

sgcUI 2026.10

October 2026

Documentation pour Delphi

Copyright © 2012-2026 eSeGeCe Software
info@esegece.com
www.esegece.com

Contents

sgcUI 26

sgcUI Overview 26
Installation 27
Plateformes et packages 29
Editions 31
.NET Packages 32

sgcPDF 33

Introduction 33
Installation 35
Architecture 38
Démarrage rapide 41
TsgcPDFDocument 44
TsgcPDFEditor 48
TsgcPDFViewer 51
TsgcPDFTextExtractor 53
TsgcPDFExporter 55
TsgcPDFOptimizer 57
TsgcPDFSignatureHandler 59
TsgcPDFSignatureVerifier 61
Texte et polices 63
Images 65
Graphisme, transparence et dégradés 66
Tableaux 68
Codes-barres 70
Annotations 72
Signets 79
Calques 81

Formulaires 82
Chiffrement 84
Signatures numériques 86
PDF/A 88
Optimisation des fichiers PDF 90
Fusionner, scinder et extraire des pages 91
Extraction de texte 92
Rendu et export 94
Gestion des erreurs et entrées malveillantes 95
Architecture du moteur de rendu 97
TsgcPDFRendererRaster 99
TsgcPDFRendererGDI 102
Fonctionnalités de rendu prises en charge 104
TsgcPDFViewerControl 106
TsgcPDFThumbnailView 113
TsgcPDFOutlineView 114
TsgcPDFCommentsView 115
TsgcPDFViewerFMX 117
TsgcPDFCommentsViewFMX 121
Polices OpenType 123
Écritures complexes et mise en forme 124
Texte chinois, japonais et coréen 126
Extraction de texte : positions des caractères 127
Limites pour les fichiers non approuvés 129
Décodeurs d'images 130
Import TIFF 132
TsgcPDFImageExtractor 134
TsgcPDFCanvas 137
EMF vers PDF 139
TsgcPDFVCLCanvasHelper 141
HTML vers PDF 142

Codes-barres 2D et GS1-128 145
TsgcPDFBarcodeControl 147
TsgcPDFBarcodeFMX 151
TsgcPDFBarcodeSymbol 154
Linux 159
macOS, iOS et Android 161
C++Builder 162
Pièces jointes 163
Factures Factur-X et ZUGFeRD 166
Enregistrement incrémental et sortie compacte 168
Fichiers volumineux 170
FDF et XFDF 171
Valeurs et apparences des formulaires 173
Signatures PAdES 175
Horodatage et validation à long terme 177
Vérification des signatures 179
TsgcPDFPAdESSigner (pont sgcSign) 181
Chiffrement par certificat 183
TsgcPDFRedactor 185
PDF balisé 188
PDF/UA et PDF/A niveau A 191
Validateur d'accessibilité 192
FAQ 194

sgcReports 196

Introduction 196
Installation 197
Getting Started 200
Report Model 202
Expressions and Text Templates 204
Data Sources 206

Groups, Headers, Footers and Columns 207
Fonts 208
Thread Safety and Server Use 209
Accessibility (PDF/UA) 211
PDF/A 212
TsgcReport 213
TsgcReportEngine 214
TsgcReportDataSetSource 216
TsgcReportJSONSource 217
TsgcReportEventSource 218
TsgcReportPDFExport 220
TsgcReportXLSXExport 222
TsgcReportCSVExport 224
TsgcReportImageExport 225
TsgcReportPreviewVCL 226
TsgcReportPreviewFMX 228
TsgcReportHTTPEndpoint 229
TsgcReportHTMLViewer 231
TsgcReportHTMLDesigner 232
TsgcReportBuilder 233
JSON Report Definitions 235
How To: Invoice with Master-Detail Data 237
How To: A Grouped Report with Totals 239
How To: Export to XLSX, CSV and PNG 241
How To: An Accessible PDF Validated with veraPDF 242
How To: Serve Reports Over HTTP 243
How To: A Linux64 Report Daemon 244
How To: A Mobile FMX Report 245
How To: The Web Report Designer 246

sgcCharts 247

Introduction 247
Architecture 248
Editions 250
Installation 251
Plateformes 253
Quick Start 256
Demos 261
TsgcChart 264
TsgcChartSeries 267
Axes 271
Title, Legend and Margins 275
Themes and Palette 278
Plot Areas 280
Colour Scale 283
TsgcChartSeries_Line, TsgcChartSeries_FastLine, TsgcChartSeries_StepLine 286
TsgcChartSeries_Area 290
TsgcChartSeries_Bar 292
TsgcChartSeries_Scatter 294
TsgcChartSeries_Candlestick, TsgcChartSeries_OHLC 296
TsgcChartSeries_Pie 299
TsgcChartSeries_Radar 303
TsgcChartSeries_Heatmap 306
TsgcChartSeries_Treemap 309
TsgcChartSeries_Funnel 313
TsgcVCLChart 316
TsgcFMXChart 319
TsgcChartRenderer 322
TsgcChartInteraction 325
TsgcChartDataSource 332
TsgcChartDataSource_Manual 334
TsgcChartDataSource_DataSet 336

TsgcChartDataSource_JSON 338
TsgcChartDataSource_CSV 340
TsgcChartDataSource_Table 342
Bridge Overview 346
TsgcChartDataSource_MQTT 347
TsgcChartDataSource_WebSocket 349
TsgcChartDataSource_SSE 351
TsgcChartDataSource_Binance 353
TsgcChartSeries_Indicator 355
TsgcChartSeries_Oscillator 360
TsgcChartSeries_Volume 364
Panes 367
TsgcChartAggregator 371
TsgcChartDrawing 374
TsgcChartBuffer 379
Decimation 382
Canvas 383
Fonts 386
Indicators 389
.NET Overview 392
Démarrage rapide : WinForms 395
Démarrage rapide : WPF 396
Quick Start: Avalonia 397
Headless Rendering 399
Series Gallery 401
Bridge 404
.NET Differences from Delphi 408
Troubleshooting 411

sgcMaps 412

Introduction 412

Présentation 413
Installation 416
Démarrage rapide VCL 418
Démarrage rapide FMX 420
Démos 422
01 Carte hors ligne 423
02 Fonctions de la carte 424
03 Services 426
04 Services hors ligne 428
05 Rendu statique 429
06 FireMonkey 431
07 Tuiles vectorielles 432
TsgcVCLMap 433
TsgcFMXMap 438
Tuiles, couches et fournisseurs 442
Tuiles vectorielles 445
Marqueurs et regroupement 448
Formes et étiquettes 451
Cartes de chaleur 453
GeoJSON, GPX et KML 455
Fonctions de géométrie 457
Rendu de cartes statiques 459
Tuiles hors ligne et téléchargeur de tuiles 462
Lecteur OpenStreetMap 464
Index de géocodage hors ligne 466
Graphes routiers et calcul d'itinéraires hors ligne 467
TsgcMapService 469
TsgcMapGeocoder 471
TsgcMapDirections 474
TsgcMapPlaces 477
TsgcMapElevation 480

TsgcMapTimeZone 482
TsgcMapRouteCalculator 484
TsgcMapOfflineRouter 487
TsgcMapOfflineGeocoder 489
TsgcMapGeoIndexBuilder 491
TsgcMapLocation 493
Outils 496
GeoIndexBuilder 497
RouteGraphBuilder 499
Lazarus 500
Présentation .NET 502
Installation .NET 505
Démarrage rapide : WinForms 507
Rendu sans interface graphique 510
Données hors ligne .NET 512
Correspondance des classes 515
Différences .NET par rapport à Delphi 518
Démos .NET 520
Dépannage 522

sgcDashboard 526

Introduction 526
Architecture 527
Éditions 529
Installation 530
Plateformes 532
Démarrage rapide VCL 536
Démarrage rapide FMX 538
Démarrage rapide C++Builder 539
Démos 540
TsgcVCLDashboard 541

TsgcVCLDashboardTile 544
TsgcFMXDashboard 546
TsgcFMXDashboardTile 549
TsgcDashboardKPI 551
TsgcVCLKPICard 554
TsgcFMXKPICard 556
TsgcDashboardSource 558
TsgcDashboardSource_WebSocket 560
TsgcDashboardSource_MQTT 562
TsgcDashboardSource_SSE 564
Disposition 566
Thèmes 568
Persistance 569
Données en direct 571
Images KPI headless 573
Présentation de .NET 574
Installation depuis NuGet 577
Démarrage rapide WinForms 579
Démarrage rapide WPF 580
Démarrage rapide Avalonia 581
Classes .NET 582
Différences de .NET par rapport à Delphi 585
Liaison de données .NET 588
Rendu headless en .NET 590
Flux en direct en .NET 592
Démos .NET 594
Dépannage 596

sgcGrid 598

Introduction 598
Architecture 599

Éditions 601
Installation 602
Plateformes 604
Performances 607
Démarrage rapide VCL 608
Démarrage rapide FMX 609
Démarrage rapide C++Builder 610
Démos 611
TsgcVCLGrid 612
TsgcFMXGrid 618
TsgcGridColumn 623
TsgcGridColumns 625
TsgcGridData 627
TsgcGridLineView 630
TsgcGridDataStore 634
TsgcGridRenderer 636
TsgcGridFormatRule 639
TsgcGridFormatRules 641
IsgcGridDataHost et IsgcUITable 642
TsgcGridSource_DataSet 644
TsgcGridFeedSource 646
TsgcGridFeedSource_WebSocket 648
TsgcGridFeedSource_MQTT 650
TsgcGridFeedSource_SSE 652
TsgcGridAIQuery 654
Mises à jour en direct et threads 656
Tri 658
Filtrage 659
Regroupement 660
Résumés 663
Lignes d'arbre 665

Maître-détail 668
Sélection et clavier 672
Redimensionnement des colonnes 674
Colonnes figées 675
Clignotement à la modification 676
Mise en forme conditionnelle 677
Modification sur place 679
Mode virtuel 681
JSON et CSV 682
Export Excel 683
Liaison à un DataSet 686
Flux en direct 687
Interroger la grille 689
Graphiques à partir de la grille 691
Thèmes 692
Rendu sans fenêtre 693
Vue d'ensemble de .NET 695
Installation depuis NuGet 698
Démarrage rapide WinForms 700
Démarrage rapide WPF 701
Démarrage rapide Avalonia 702
Classes .NET 703
Différences entre .NET et Delphi 707
Liaison de données .NET 710
Rendu sans fenêtre .NET 713
Liaison DataTable, JSON et CSV en .NET 714
Modification sur place en .NET 716
Flux en direct .NET 717
Interroger la grille en .NET 719
Performances .NET 720
Regroupement et résumés .NET 722

Lignes d'arbre et maître-détail .NET 725
Export Excel .NET 728
Démos .NET 730
Dépannage 732

sgcNavigation 734

Introduction 734
Architecture 736
Éditions 737
Installation 738
Plateformes 740
Démarrage rapide VCL 743
Démarrage rapide FMX 745
Démarrage rapide C++Builder 746
Démos 747
TsgcVCLNavigationBar 748
TsgcFMXNavigationBar 751
TsgcNavigationItem 754
TsgcNavigationItems 757
TsgcNavigationModel 759
Mode sidebar 762
Mode accordéon 764
Réduction et animation 765
Images et glyphes 767
Badges 769
Clavier et focus 770
Liens d'action 772
Thèmes 773
Rendu sans interface 774
Vue d'ensemble de .NET 777
Installation depuis NuGet 780

Démarrage rapide WinForms 781
Démarrage rapide WPF 782
Démarrage rapide Avalonia 783
Classes .NET 784
Différences de .NET avec Delphi 786
Liaison de données .NET 788
Rendu sans interface en .NET 790
Démos .NET 791
Dépannage 793

sgcToolbar 794

Introduction 794
Architecture 795
Éditions 797
Installation 798
Plateformes 800
Démarrage rapide VCL 803
Démarrage rapide FMX 805
Démarrage rapide C++Builder 806
Démos 807
TsgcVCLToolbar 808
TsgcFMXToolbar 811
TsgcVCLStatusBar 814
TsgcFMXStatusBar 816
TsgcToolbarItem 819
TsgcToolbarItems 822
TsgcStatusBarPanel 823
TsgcStatusBarPanels 825
TsgcToolbarModel 826
TsgcStatusBarModel 829
Types d'éléments 831

Boutons à bascule et groupes d'options 833
Menus déroulants 834
Débordement 835
Liens d'action 836
Images et glyphes 837
Panneaux d'état 838
Progression et indicateurs 839
Mises à jour thread-safe 840
Point de redimensionnement 842
Clavier 843
Thèmes 844
Rendu sans interface 845
Présentation de .NET 847
Installation depuis NuGet 850
Démarrage rapide WinForms 851
Démarrage rapide WPF 852
Démarrage rapide Avalonia 853
Classes .NET 854
Différences de .NET avec Delphi 857
Liaison de données .NET 860
Rendu sans interface en .NET 862
Démos .NET 863
Dépannage 865

sgcWizard 867

Introduction 867
Architecture 868
Éditions 870
Installation 871
Plateformes 873
Démarrage rapide VCL 876

Démarrage rapide FMX 878
Démarrage rapide C++Builder 880
Démos 881
TsgcVCLWizard 882
TsgcVCLWizardPage 885
TsgcFMXWizard 887
TsgcFMXWizardPage 890
TsgcVCLStepper 892
TsgcFMXStepper 894
TsgcWizardStep 896
TsgcWizardSteps 898
TsgcStepperModel 899
Navigation et pages ignorées 901
Validation 903
Finish et Cancel 904
Clavier 905
États des étapes 906
Disposition du stepper 908
Libellés des boutons et localisation 910
Thèmes 911
Conception 912
Rendu sans interface 913
Vue d'ensemble de .NET 915
Installation depuis NuGet 917
Démarrage rapide WinForms 918
Démarrage rapide WPF 919
Démarrage rapide Avalonia 920
Classes .NET 921
Différences de .NET par rapport à Delphi 924
Liaison de données .NET 926
Rendu sans interface avec .NET 928

Démos .NET 930
Dépannage 932

sgcPivotGrid 934

Introduction 934
Architecture 936
Éditions 938
Installation 939
Plateformes 941
Démarrage rapide VCL 944
Démarrage rapide FMX 946
Démarrage rapide C++Builder 948
Démarrage rapide DataSet 949
Démarrage rapide sans interface 951
Démos 953
TsgcVCLPivotGrid 954
TsgcFMXPivotGrid 958
TsgcVCLPivotFieldList 961
TsgcFMXPivotFieldList 963
TsgcPivotEngine 965
TsgcPivotField 969
TsgcPivotCalculatedField 971
TsgcPivotSource_DataSet 972
TsgcPivotSource_Table 973
TsgcPivotGridRenderer 975
TsgcPivotFormatRule 977
Zones 979
Agrégations 981
Regroupement des dates 983
Totaux 984
Filtres 986

Top N 989
Afficher les valeurs en tant que 990
Champs calculés 992
Persistance de la mise en page 994
Export 996
Développer et réduire 998
Tri 1000
Drill-down 1002
Mise en forme conditionnelle 1004
Interopérabilité avec les graphiques 1006
Clavier et accessibilité 1008
Performances 1010
TsgcPivotGridAIQuery 1012
Interrogez votre tableau croisé 1016
Présentation de .NET 1021
Installation depuis NuGet 1023
Démarrage rapide WinForms 1024
Démarrage rapide WPF 1025
Démarrage rapide Avalonia 1026
Classes .NET 1027
Différences de .NET par rapport à Delphi 1030
Liaison de données .NET 1032
Rendu sans interface .NET 1034
Démos .NET 1035
Dépannage 1036

sgcPlanner 1038

Introduction 1038
Architecture 1040
Éditions 1043
Installation 1045

Plateformes 1047
Démarrage rapide VCL 1051
Démarrage rapide FMX 1053
Démarrage rapide C++Builder 1054
Démos 1055
TsgcPlannerData 1057
TsgcVCLScheduler 1062
TsgcFMXScheduler 1066
TsgcVCLGantt 1069
TsgcFMXGantt 1073
TsgcVCLKanban 1077
TsgcFMXKanban 1081
TsgcPlannerSource_DataSet 1084
TsgcPlannerICalendar 1087
TsgcPlannerMSProject 1089
TsgcPlannerRenderer 1091
Données partagées 1094
Vues synchronisées 1096
Calendrier de travail 1098
Récurrence 1100
Échelle de temps et zoom 1103
Glisser-déposer 1105
Dependencies 1108
Planification automatique 1110
Chemin critique 1112
Lignes de base 1114
Couloirs et limites WIP 1116
Liaison DataSet 1118
Persistance JSON 1120
iCalendar 1122
XML MS Project 1124

Rendu sans fenêtre 1126
Thèmes 1128
Fonctionnalités par édition 1130
Synchronisation avec Google Agenda 1132
TsgcPlannerGoogleSync 1136
TsgcPlannerICalSubscription 1141
Vue d'ensemble de .NET 1145
Installation depuis NuGet 1148
Démarrage rapide WinForms 1149
Démarrage rapide WPF 1150
Démarrage rapide Avalonia 1152
Classes .NET 1153
Différences de .NET par rapport à Delphi 1156
Données et planification dans .NET 1158
Liaison de données dans .NET 1159
JSON, iCalendar et MS Project dans .NET 1161
Rendu sans fenêtre dans .NET 1162
Synchronisation avec Google Agenda et iCalendar en .NET 1163
Démos .NET 1165
Dépannage 1167

sgcEditors 1170

Introduction 1170
Architecture 1172
Contrôles 1175
Installation 1177
Plateformes 1179
Démarrage rapide VCL 1182
Démarrage rapide FMX 1184
Démarrage rapide C++Builder 1186
Démarrage rapide data-aware 1187

Calendrier mensuel 1189
Sélecteur de date 1193
Sélecteur d'heure 1197
Sélecteur de date et d'heure 1200
Curseur de plage 1203
Sélecteur de couleur 1206
Token edit 1209
Évaluation 1212
Spin edit 1215
Sélecteur de date DB 1218
Sélecteur d'heure DB 1220
Spin edit DB 1222
Évaluation DB 1224
Curseur de plage DB 1226
TsgcEditorModel 1228
TsgcEditorLocale 1232
TsgcEditorFieldLink 1235
Saisie de texte et IME 1238
Clavier et accessibilité 1241
Thèmes 1243
Localisation 1245
DPI 1247
Sortie sans interface 1249
Éditions 1251
Vue d'ensemble de .NET 1253
Installation depuis NuGet 1255
Démarrage rapide WinForms 1256
Démarrage rapide WPF 1257
Démarrage rapide Avalonia 1258
Classes .NET 1259
Différences de .NET par rapport à Delphi 1262

Liaison de données dans .NET 1264
Rendu sans interface dans .NET 1265
Démos .NET 1266
Dépannage 1267

sgcGauges 1269

Introduction 1269
Architecture 1270
Types de jauges 1272
Installation 1274
Plateformes 1276
Démarrage rapide VCL 1279
Démarrage rapide FMX 1281
Démarrage rapide sans interface 1283
Démarrage rapide C++Builder 1284
Jauge radiale 1285
Jauge linéaire 1287
Jauge numérique 1289
LED 1291
Barre de LED 1293
Anneau de progression 1295
Vumètre 1297
Horloge 1299
Boussole 1301
Annunciator 1303
TsgcCustomGauge 1306
TsgcGaugeScale et TsgcGaugeRange 1309
TsgcGaugePointer, TsgcGaugeThreshold et TsgcGaugeAnimation 1311
Échelles 1314
Plages 1316
Aiguilles 1318

Seuils 1320
Animation 1322
Thèmes 1324
Sortie sans interface 1326
Éditions 1328
Vue d'ensemble de la passerelle 1330
Source MQTT 1333
Source WebSocket 1335
Source d'événements envoyés par le serveur 1337
Vue d'ensemble de .NET 1339
Installation depuis NuGet 1342
Démarrage rapide WinForms 1343
Démarrage rapide WPF 1344
Démarrage rapide Avalonia 1345
Classes .NET 1346
Différences de .NET par rapport à Delphi 1349
Liaison de données dans .NET 1351
Rendu sans interface dans .NET 1352
Passerelle sgcWebSockets pour .NET 1353
Démos .NET 1355
Dépannage 1357

sgcDiagram 1359

Introduction 1359
Qu'est-ce que sgcDiagram 1361
Architecture 1362
Éditions 1364
Plateformes 1366
Démarrage rapide VCL 1370
Démarrage rapide FMX 1372
Démarrage rapide C++Builder 1373

Démarrage rapide Headless 1374
TsgcVCLDiagramView 1375
TsgcFMXDiagramView 1378
TsgcVCLDiagramPalette 1379
TsgcFMXDiagramPalette 1381
TsgcDiagramDataBinding 1382
Événements 1384
TsgcDiagramDocument 1386
TsgcDiagramNode 1389
TsgcDiagramConnector 1391
TsgcDiagramPort 1393
TsgcDiagramStyle 1394
TsgcDiagramLayer 1395
TsgcDiagramController 1396
TsgcDiagramRenderer 1398
Format JSON 1399
Disposition en couches 1402
Disposition en arbre 1404
Liaison de données 1406
Formes 1408
Connecteurs et routage 1410
Ports et collage 1412
Édition et clavier 1414
Magnétisme et repères 1416
Annuler et presse-papiers 1417
Calques 1418
Sérialisation 1419
Export 1421
Thèmes 1422
Mode visionneuse 1423
Vue d'ensemble de .NET 1424

Installation depuis NuGet 1427
Démarrage rapide WinForms 1428
Démarrage rapide WPF 1429
Démarrage rapide Avalonia 1430
Classes .NET 1431
Différences de .NET par rapport à Delphi 1433
Liaison de données dans .NET 1435
Rendu sans interface dans .NET 1437
Démos .NET 1438
Dépannage 1439

Index 1440

Présentation de sgcUI

sgcUI est la suite de composants visuels et documentaires d'eSeGeCe pour Delphi et C++Builder. Les produits partagent un dossier de sources, un fichier include de version, un installateur et cette aide. Chaque produit est concédé sous licence séparément, et l'installateur n'installe que les produits couverts par votre licence.

Les produits

Produit	À quoi il sert
sgcPDF	Une bibliothèque PDF native pour créer, modifier et rendre des documents PDF, avec texte et polices, dessin, images, sécurité, accessibilité et contrôles de visionneuse. Voir sgcPDF.
sgcReports	Un moteur de rapports sans interface, construit sur sgcPDF, pour les services Windows, les démons Linux64 et les points de terminaison HTTP ainsi que pour les applications VCL et FMX. Voir sgcReports.
sgcCharts	Graphiques de données en direct. Les sources de données ajoutent des points ou des chandeliers OHLC à un tampon circulaire, et le graphique dessine un instantané décimé, si bien que même les flux rapides s'affichent de façon fluide. Voir sgcCharts.
sgcMaps	Cartes interactives. Les contrôles de carte dessinent nativement des tuiles raster et vectorielles, sans contrôle de navigateur, et affichent par-dessus des marqueurs, des itinéraires, des formes, des étiquettes et des cartes de chaleur. Voir sgcMaps.
sgcDashboard	Un dashboard composé d'une grille de tiles. Chaque tile héberge n'importe quel contrôle, par exemple un graphique, une carte, une visionneuse PDF ou un aperçu de rapport. Voir sgcDashboard.
sgcGrid	Une grille de données en direct. Les lignes sont mises à jour par clé depuis n'importe quel thread, les cellules modifiées clignotent puis s'estompent, et les rafales de mises à jour sont regroupées en rafraîchissements temporisés. Voir sgcGrid.
sgcNavigation	Une barre de navigation qui présente les pages d'une application sous forme de liste d'éléments avec icônes, légendes, badges et en-têtes de groupe. Voir sgcNavigation.
sgcToolbar	Une barre d'outils et une barre d'état, avec des boutons, des boutons à bascule, des groupes d'options, des boutons déroulants, des libellés, des séparateurs et des espaceurs. Voir sgcToolbar.
sgcWizard	Assistants pas à pas et indicateurs d'étapes : conteneurs de pages avec une barre Stepper et une barre de boutons. Voir sgcWizard.
sgcUICore	La base commune de sgcCharts, sgcDashboard, sgcGrid, sgcNavigation, sgcToolbar et sgcWizard. Elle n'est ni vendue ni sélectionnée séparément : l'installateur l'ajoute dès qu'un de ces produits est installé.

Pour aller plus loin

- Installation explique l'installateur sgcUI et la façon dont il choisit les produits.
- Éditions énumère les éditions Registered, Trial et Community de chaque produit.
- Paquets .NET énumère les paquets NuGet de chaque produit.

Installation

Tous les produits sgcUI sont installés par un seul installateur. Vous n'avez pas à télécharger ni à exécuter un installateur distinct pour chaque produit.

Connexion

L'installateur Registered vous demande de vous connecter avec le nom d'utilisateur et le mot de passe de votre compte eSeGeCe. Vous pouvez aussi réutiliser la licence enregistrée par une installation précédente ou activer une licence hors ligne. Le serveur de licences répond avec les produits que possède votre compte, et une licence All-Access comprend tous les produits.

Choix des produits

La page des composants affiche une case à cocher par produit. Un produit que votre licence ne couvre pas est grisé et non coché. Sélectionnez les produits à installer, ainsi que les démos et les fichiers d'aide.
Certains produits en nécessitent d'autres, et l'installateur les ajoute pour vous :

- La sélection de sgcReports installe aussi sgcPDF, le moteur sur lequel il repose.
- sgcCharts, sgcDashboard, sgcGrid, sgcNavigation, sgcToolbar et sgcWizard reposent sur sgcUICore, si bien que sgcUICore est installé dès que l'un d'eux est sélectionné.

Intégrations sgcWebSockets

Le groupe **sgcWebSockets bridges** propose sgcCharts for sgcWebSockets, sgcDashboard for sgcWebSockets, sgcGrid for sgcWebSockets et sgcReports HTML for sgcWebSockets. Une passerelle n'est disponible que si sgcWebSockets est déjà installé sur la machine et que vous possédez le produit qu'elle étend. Sinon, elle est grisée.

Compilation des packages

Après avoir copié les fichiers, l'installateur compile les packages sur votre machine pour chaque version de Delphi et de C++Builder que vous avez choisie, et enregistre les packages de conception dans l'IDE. Les packages sont compilés dans l'ordre des dépendances : d'abord sgcUICore, puis les produits qui reposent dessus, ensuite sgcPDF, sgcReports et sgcMaps, et enfin les passerelles sgcWebSockets.

Passage depuis des installations autonomes

Les versions précédentes de sgcPDF, sgcReports, sgcMaps et sgcCharts avaient leurs propres installateurs. Lorsque l'installateur sgcUI trouve l'une de ces installations, il répertorie les chemins de bibliothèque et les packages de conception que l'ancienne installation a ajoutés à chaque IDE, et il ne les supprime qu'après votre confirmation de cette page. Les entrées des chemins de recherche et de navigation qui pointent encore vers le dossier d'un produit autonome sont supprimées elles aussi, de sorte que l'IDE ne trouve jamais deux copies d'une unité.

Éditions Trial et Community

- L'installateur Trial installe chaque produit dans son édition Trial, sans licence. Il existe un installateur Trial pour Delphi et un pour C++Builder.
- L'installateur Community installe sgcCharts, sgcDashboard, sgcGrid, sgcNavigation, sgcToolbar et sgcWizard, ainsi que sgcUICore. Il propose aussi sgcReports lorsqu'un sgcPDF sous licence est déjà installé, car sgcPDF n'a pas d'édition Community. L'installateur Community ne propose jamais les passerelles sgcWebSockets. Voir Éditions.

Détails des produits

Chaque produit décrit ses packages, ses prérequis et ses fichiers installés dans sa propre rubrique Installation :

- sgcPDF
- sgcReports
- sgcCharts
- sgcMaps
- sgcDashboard
- sgcGrid
- sgcNavigation
- sgcToolbar
- sgcWizard

Plateformes et packages

sgcUICore et chaque produit de composants sgcUI fournissent un package d'exécution indépendant du framework (sgc<Product>), un package d'exécution VCL (sgc<Product>VCL), un package d'exécution FireMonkey à partir de XE6 (sgc<Product>FMX), ainsi qu'un package de conception VCL et un package de conception FireMonkey (dclsgc<Product> et dclsgc<Product>FMX). sgcUICore n'a pas de package de conception. Le package neutre est compilé pour Win32 et Win64, et à partir de Delphi 10.3 aussi pour Linux64 et macOS. Le package FireMonkey est compilé pour macOS à partir de Delphi 10.3 et pour Android et iOS à partir de Delphi 10.4. Les packages C++Builder sont compilés pour Windows uniquement. Chaque produit présente ses packages, ses versions d'IDE et ses plateformes cibles dans son propre sujet :

Produit	Sujet
sgcCharts	Plateformes
sgcDashboard	Plateformes
sgcGrid	Plateformes
sgcNavigation	Plateformes
sgcToolbar	Plateformes
sgcWizard	Plateformes
sgcPivotGrid	Plateformes
sgcPlanner	Plateformes
sgcEditors	Plateformes
sgcGauges	Plateformes
sgcDiagram	Plateformes

C++Builder Win64 : packages d'exécution

C++Builder Win64 (chaîne d'outils classique, C++Builder 12 et 13) : les packages d'exécution Win64 de sgcUI sont liés par le compilateur Delphi, de sorte que les exceptions gérées dans le code sgcUI fonctionnent lorsque votre application utilise des packages d'exécution. L'éditeur de liens C++ Win64 (ilink64) a encore un défaut propre : le code Delphi qu'il lie statiquement dans une application compilée avec des packages d'exécution plante dans `_DelphiExceptionHandler` la première fois qu'un bloc `except on E: <class>` intercepte une exception dont la classe se trouve dans un package d'exécution.

- Lorsque Lier avec les packages d'exécution est activé (Options du projet, Packages, Packages d'exécution), listez chaque package d'exécution sgcUI utilisé par votre projet, afin qu'aucune unité sgcUI ne soit liée dans l'exécutable.
- Dans le .cbproj, chacun de ces packages doit figurer à la fois dans PackageImports et dans LinkPackageImports. Un package listé uniquement dans PackageImports n'est pas lié.
- Lorsque Lier avec les packages d'exécution est désactivé, tout est lié statiquement et fonctionne.
- Win32 et Win64x (Modern) ne sont pas concernés.

C++Builder 10.3 Rio, Win64 : exceptions dans les threads secondaires

Lorsqu'une exception Delphi est levée dans un thread secondaire, par exemple dans un gestionnaire d'événement exécuté dans un thread sgcWebSockets ou dans un TThread quelconque, et que le code C++ de ce thread

l'intercepte ou la fait remonter à travers une fonction C++ qui détruit un objet local tel qu'un UnicodeString temporaire, l'application plante avec une violation d'accès. C'est un défaut du runtime C++Builder 10.3 Win64 (cc64260mt.dll). Il se produit aussi sans code sgc, et C++Builder 10.1, 10.2, 10.4, 11, 12 et 13 ne sont pas concernés. Solutions : utilisez C++Builder 10.4 ou une version ultérieure, ou, dans le code Win64 exécuté dans des threads secondaires, évitez de lever des exceptions, par exemple TryStrToInt au lieu de StrToInt.

C++Builder 13 Win64x : noms de fichiers des packages

Les packages d'exécution Win64x (Modern) ont les mêmes noms de fichiers que les packages Win64 classiques, par exemple sgcUICoreC13.bpl, et tous deux sont en 64 bits, si bien que Windows ne peut pas les distinguer. Une application Win64x qui utilise des packages d'exécution et trouve d'abord le bpl Win64 ne démarre pas (0xC0000139, point d'entrée introuvable). Placez le dossier des bpl Win64x (Bpl\Win64x) dans le PATH avant le dossier Win64, ou livrez les bpl Win64x dans le même dossier que l'exécutable. Les applications liées sans packages d'exécution ne sont pas concernées.

Delphi XE7 FireMonkey : FMX.Forms en premier

Delphi XE7 uniquement : un programme FireMonkey peut se terminer par "Runtime error 216" à l'arrêt lorsque FMX.Types ou une unité FMX de sgcUI précède FMX.Forms dans l'ordre uses du programme et qu'un contrôle stylé a été utilisé (FMX libère le service de sélection avant le style par défaut). XE6, XE8 et les versions ultérieures ne sont pas concernés. Solution : placez FMX.Forms en première unité de la clause uses du .dpr, comme le fait l'IDE pour les nouvelles applications FMX.

Capturer un formulaire VCL avec PaintTo

Pour capturer un formulaire VCL contenant des contrôles sgcUI dans un bitmap avec PaintTo, verrouillez le canvas du bitmap autour de l'appel (`Bitmap.Canvas.Lock`, puis `Unlock`). PaintTo dans un canvas non verrouillé peut perdre son contexte de périphérique lorsqu'un contrôle natif du formulaire traite un message pendant la capture, et le bitmap reste vide ou partiel. Un formulaire FireMonkey utilise MakeScreenshot à la place.

Voir aussi

- Installation
- Editions

Éditions

Chaque produit sgcUI est compilé à partir de la même source dans chaque édition. L'édition ne décide que des unités et des fonctions qu'une compilation inclut.

- **Registered** : toutes les unités des produits que comprend votre licence, y compris les passerelles sgcWebSockets.
- **Trial** : tous les produits, pour évaluation. Les compilations Trial sont réalisées avec le define `SGC_TRIAL`.
- **Community** : une édition gratuite pour les produits listés ci-dessous, compilée avec le define `SGC_COMMUNITY`. Elle n'inclut jamais les passerelles sgcWebSockets.

Édition Community par produit

Produit	Community	Non inclus dans Community
sgcPDF	Non	L'installateur Community ne fournit pas sgcPDF.
sgcReports	Oui, sur un sgcPDF sous licence	La passerelle HTML de sgcReports pour sgcWebSockets. sgcReports Community n'est proposé que lorsqu'un sgcPDF sous licence est déjà installé.
sgcCharts	Oui	Les fonctions financières : indicateurs, séries financières, l'agrégateur et les outils de dessin. La passerelle sgcWebSockets.
sgcMaps	Non	L'installateur Community ne fournit pas sgcMaps.
sgcDashboard	Oui	Le glisser et le redimensionnement des tiles à l'exécution, ainsi que SaveLayout et LoadLayout. La passerelle sgcWebSockets.
sgcGrid	Oui	L'édition de cellules sur place. La passerelle sgcWebSockets.
sgcNavigation	Oui	Les liens Action sur les éléments de navigation.
sgcToolbar	Oui	Le menu de débordement et les liens Action.
sgcWizard	Oui	Rien. La compilation Community offre toutes les fonctions.
sgcUICore	Oui	Installé avec les produits Community qui en ont besoin.

Chaque produit explique ses éditions plus en détail dans sa propre aide, par exemple Éditions de sgcCharts.

Paquets .NET

Les versions .NET des produits sgcUI sont distribuées sous forme de paquets NuGet. Chaque produit possède un paquet principal, portant le nom du produit, et des paquets complémentaires pour un framework d'interface ou une intégration. Ajoutez le paquet principal et les paquets complémentaires dont votre application a besoin.

Produit	Paquet principal	Paquets complémentaires
sgcPDF	`esegece.sgcPDF`	`esegece.sgcPDF.WinForms` , `esegece.sgcPDF.sgcSign`
sgcReports	`esegece.sgcReports`	`esegece.sgcReports.WinForms` , `esegece.sgcReports.AspNetCore` , `esegece.sgcReports.HTML` , `esegece.sgcReports.sgcSign`
sgcCharts	`esegece.sgcCharts`	`esegece.sgcCharts.Rendering` , `esegece.sgcCharts.WinForms` , `esegece.sgcCharts.Wpf` , `esegece.sgcCharts.Avalonia` , `esegece.sgcCharts.sgcWebSockets`
sgcMaps	`esegece.sgcMaps`	`esegece.sgcMaps.Rendering` , `esegece.sgcMaps.WinForms` , `esegece.sgcMaps.Offline`
sgcDashboard	`esegece.sgcDashboard`	`esegece.sgcDashboard.Rendering` , `esegece.sgcDashboard.WinForms` , `esegece.sgcDashboard.Wpf` , `esegece.sgcDashboard.Avalonia` , `esegece.sgcDashboard.sgcWebSockets`
sgcGrid	`esegece.sgcGrid`	`esegece.sgcGrid.Rendering` , `esegece.sgcGrid.WinForms` , `esegece.sgcGrid.Wpf` , `esegece.sgcGrid.Avalonia` , `esegece.sgcGrid.sgcWebSockets` , `esegece.sgcGrid.sgcHTML`
sgcNavigation	`esegece.sgcNavigation`	`esegece.sgcNavigation.Rendering` , `esegece.sgcNavigation.WinForms` , `esegece.sgcNavigation.Wpf` , `esegece.sgcNavigation.Avalonia`
sgcToolbar	`esegece.sgcToolbar`	`esegece.sgcToolbar.Rendering` , `esegece.sgcToolbar.WinForms` , `esegece.sgcToolbar.Wpf` , `esegece.sgcToolbar.Avalonia`
sgcWizard	`esegece.sgcWizard`	`esegece.sgcWizard.Rendering` , `esegece.sgcWizard.WinForms` , `esegece.sgcWizard.Wpf` , `esegece.sgcWizard.Avalonia`
sgcUICore	`esegece.sgcUICore`	`esegece.sgcUICore.Rendering` , `esegece.sgcUICore.WinForms` , `esegece.sgcUICore.Wpf` , `esegece.sgcUICore.Avalonia` . Ces paquets sont des dépendances des produits reposant sur sgcUICore, si bien que NuGet les restaure automatiquement.

Paquets Community

Une compilation Community d'un paquet a le même identifiant avec le suffixe `.Community`, par exemple `esegece.sgcGrid.Community`. Consultez Éditions pour savoir ce que l'édition Community de chaque produit omet.

Introduction

sgcPDF est une bibliothèque Delphi permettant de créer, ouvrir, modifier, signer, restituer et visualiser des documents PDF. Elle est écrite en Object Pascal pur et se compile directement dans votre exécutable, de sorte qu'il n'y a aucune DLL de moteur PDF à déployer ni aucune bibliothèque tierce à licencier.

Fonctionnalités principales

- Créez des documents PDF depuis le code : texte avec les 14 polices standard ou des polices TrueType et OpenType (CFF) intégrées avec sous-ensemble (subsetting), texte Unicode y compris CJK, mise en forme des écritures complexes (liaison arabe, ligatures, texte de droite à gauche), lignes, courbes, images, transparence, dégradés et motifs.
- Dessin avec une API de type TCanvas (TsgcPDFCanvas), conversion de dessins EMF et de canevas VCL en PDF vectoriel, et conversion de HTML avec CSS en pages PDF.
- Images : import JPEG, PNG, BMP et TIFF multi-pages ; décodeurs en Pascal pur pour JPEG (baseline et progressif), JPEG 2000, CCITT G3/G4 et JBIG2 ; extraction d'images vers PNG, JPEG ou BMP.
- Tableaux avec en-têtes, bordures, couleurs de lignes alternées et sauts de page automatiques.
- Codes-barres : Code 128, GS1-128, Code 39, EAN-13, EAN-8, UPC-A, ITF, Codabar, QR Code (versions 1 à 40), DataMatrix et PDF417.
- Annotations, signets (outlines), calques de contenu optionnel et formulaires interactifs : création, remplissage, aplatissement, listes à sélection multiple, apparences de champs Unicode, import et export FDF et XFDF.
- Ouvrez des documents existants, modifiez-les, ajoutez des filigranes, en-têtes, pieds de page et numéros de page, fusionnez, scindez, extrayez, faites pivoter et supprimez des pages.
- Enregistrement incrémental qui conserve la validité des signatures, sortie compacte avec flux d'objets et flux de références croisées, et fichiers de plus de 2 Go lus à la demande.
- Fichiers intégrés avec relations PDF/A-3, et factures électroniques Factur-X, ZUGFeRD et XRechnung.
- Extraction de texte avec la position, la police et la taille de chaque caractère.
- Chiffrement par mots de passe (RC4 40 et 128 bits, AES-128, AES-256 révision 6) ou par certificats de destinataires (chiffrement à clé publique).
- Signatures numériques avec fichiers PFX, le magasin de certificats Windows ou tout signataire externe, PAdES (ETSI.CAdES.detached), horodatage de documents, DSS pour la validation à long terme, vérification des signatures, et PAdES B-B à B-LTA via le pont sgcSign optionnel.
- Caviardage réel qui supprime le texte, les pixels d'image, les graphismes et les annotations des zones marquées ou des résultats de recherche.
- PDF balisé avec une API d'arbre de structure, sortie PDF/UA-1 et PDF/A-1a, 2a et 3a, sortie PDF/A-1b, 2b et 3b, et un validateur PDF/A et PDF/UA.
- Optimiseur qui compresse les flux, supprime les objets dupliqués et inutilisés, et sous-échantillonne les images.
- Rendu de pages en Pascal pur vers PNG et BMP sur toutes les plateformes, un moteur de rendu GDI, et des contrôles de visualisation pour VCL et FireMonkey avec zoom, sélection, recherche, remplissage de formulaires et impression.
- Windows, Linux 64 bits, macOS, iOS et Android, avec Delphi 7 à 13 et C++Builder 2007 à 13.

Prérequis

- Delphi 7 à Delphi 13, C++Builder 2007 à C++Builder 13.
- Windows 32 bits et 64 bits pour toutes les fonctionnalités.
- Linux 64 bits, macOS, iOS et Android pour le noyau multiplateforme, le moteur de rendu raster et le lecteur FireMonkey (Delphi XE5 ou plus récent pour FireMonkey).
- Le moteur de rendu GDI, TsgcPDFViewer, TsgcPDFExporter et les contrôles de visualisation VCL utilisent le GDI de Windows. La signature avec des fichiers PFX ou le magasin de certificats Windows, ainsi que le chiffrement par certificat, utilisent la CryptoAPI de Windows (crypt32.dll, incluse dans Windows).

Composants

Après l'installation, les composants suivants apparaissent sur la page **sgcPDF** de la palette d'outils :

Composant	Paquet	Description
TsgcPDFDocument	Core	Le document PDF : création de pages, dessin de contenu, chargement, enregistrement, chiffrement.
TsgcPDFEditor	Core	Modification de fichiers PDF existants : filigranes, numéros de page, fusion, formulaires.
TsgcPDFTextExtractor	Core	Texte d'une page avec les polices et les positions des caractères.
TsgcPDFOptimizer	Core	Réduit la taille d'un fichier PDF.
TsgcPDFSignatureHandler	Core	Signe des documents PDF.
TsgcPDFSignatureVerifier	Core	Vérifie les signatures d'un document PDF.
TsgcPDFRedactor	Core	Supprime le texte, les images et les graphismes de zones des pages.
TsgcPDFImageExtractor	Core	Liste et enregistre les images d'un PDF.
TsgcPDFHTMLRenderer	Core	Convertit HTML et CSS en pages PDF.
TsgcPDFViewer	VCL	Une fenêtre native qui affiche, recherche et imprime un PDF.
TsgcPDFExporter	VCL	Effectue le rendu des pages en fichiers BMP, PNG ou JPEG et en vignettes.
TsgcPDFViewerControl	VCL	Contrôle de visualisation VCL : zoom, sélection, recherche, remplissage de formulaires, impression.
TsgcPDFThumbnailView	VCL	Vignettes de pages pour TsgcPDFViewerControl.
TsgcPDFOutlineView	VCL	Arborescence de signets pour TsgcPDFViewerControl.
TsgcPDFBarcodeControl	VCL	Contrôle de code-barres et de QR Code : onze symbologies, rotation, sortie PNG, SVG et PDF.
TsgcPDFViewerFMX	FMX	Contrôle de visualisation FireMonkey pour Windows, macOS, iOS, Android et Linux.
TsgcPDFBarcodeFMX	FMX	Contrôle FireMonkey de code-barres et de QR Code.

Commencez par Installation, puis Démarrage rapide.

Installation

Prérequis

- Delphi 7 à Delphi 13, ou C++Builder 2007 à C++Builder 13.
- Windows 32 bits et 64 bits pour toutes les fonctionnalités. Linux 64 bits, macOS, iOS et Android pour les unités multiplateformes (Delphi XE5 ou plus récent pour FireMonkey), voir Linux et macOS, iOS et Android.

Paquets

sgcPDF est divisé en trois paquets d'exécution, afin que les applications et services qui n'utilisent pas la VCL ni FireMonkey n'aient besoin que du noyau :

Paquet	Nécessite	Contenu
sgcPDFDxx (exécution du noyau)	rtl	Chaque unité indépendante de la plateforme : document, éditeur, analyseur, écrivain, polices, mise en forme, images et décodeurs, formulaires, annotations, chiffrement, signatures, PDF/A, PDF balisé, caviardage, pièces jointes, extraction de texte et d'images, optimiseur, TsgcPDFCanvas, HTML vers PDF et le moteur de rendu raster.
sgcPDFVCLDxx (exécution VCL)	rtl, vcl, sgcPDFDxx	Unités Windows et VCL : TsgcPDFRendererGDI, TsgcPDFViewer, TsgcPDFExporter, TsgcPDFViewerControl, TsgcPDFThumbnailView, TsgcPDFOutlineView, TsgcPDFBarcodeControl, TsgcPDFEMFPlayer et TsgcPDFVCLCanvasHelper.
sgcPDFFMXDxx (exécution FMX, XE5 et plus récent)	rtl, fmx, sgcPDFDxx	TsgcPDFViewerFMX, TsgcPDFBarcodeFMX et TsgcPDFRendererFMX.
dclsgcPDFDxx (conception)	designide, rtl, vcl, sgcPDFDxx, sgcPDFVCLDxx	Enregistre les composants du noyau et de la VCL.
dclsgcPDFFMXDxx (conception, XE5 et plus récent)	designide, rtl, fmx, sgcPDFDxx, sgcPDFFMXDxx	Enregistre TsgcPDFViewerFMX et TsgcPDFBarcodeFMX.

xx est le suffixe de version Delphi, et les fichiers se trouvent dans le dossier **Packages** :

Version de Delphi	Suffixe	Paquets FMX
Delphi 7	D7	Non
Delphi 2007	D2007	Non
Delphi 2009	D2009	Non
Delphi 2010	D2010	Non
Delphi XE à XE4	DXE ... DXE4	Non
Delphi XE5 à XE8	DXE5 ... DXE8	Oui

Version de Delphi	Suffixe	Paquets FMX
Delphi 10 Seattle	D10	Oui
Delphi 10.1 Berlin	D10_1	Oui
Delphi 10.2 Tokyo	D10_2	Oui
Delphi 10.3 Rio	D10_3	Oui
Delphi 10.4 Sydney	D10_4	Oui
Delphi 11 Alexandria	D11	Oui
Delphi 12 Athens	D12	Oui
Delphi 13	D13	Oui

Par exemple, Delphi 13 utilise sgcPDFD13.dpk, sgcPDFVCLD13.dpk, sgcPDFFMXD13.dpk, dclsgcPDFD13.dpk et dclsgcPDFFMXD13.dpk. C++Builder utilise les paquets .cbproj sgcPDFCxx, sgcPDFVCLCxx et dclsgcPDFCxx, voir C++Builder.

Installation des paquets

1. Ouvrez et compilez (ou générez) les paquets d'exécution dans cet ordre : **sgcPDFDxx.dpk**, **sgcPDFVCLDxx.dpk** et, si vous utilisez FireMonkey, **sgcPDFFMXDxx.dpk**.
2. Ouvrez le paquet de conception **dclsgcPDFDxx.dpk**, compilez-le et sélectionnez **Install**. Pour FireMonkey, faites de même avec **dclsgcPDFFMXDxx.dpk**.
3. Ajoutez le dossier **Source** de sgcPDF au chemin de la bibliothèque : *Tools* > *Options* > *Language* > *Delphi* > *Library* > *Library Path* (dans les versions plus anciennes *Tools* > *Options* > *Library*). Ajoutez-le pour chaque plateforme que vous utilisez : Windows 32 bits, Windows 64 bits, Linux 64 bits, macOS, iOS et Android.

Vérifier l'installation

Créez une nouvelle application VCL et ouvrez la palette d'outils. La page **sgcPDF** contient ces composants (TsgcPDFViewerFMX et TsgcPDFBarcodeFMX apparaissent dans les applications FireMonkey) :

Composant	Paquet	Description
TsgcPDFDocument	Core	Le document PDF : création de pages, dessin de contenu, chargement, enregistrement, chiffrement.
TsgcPDFEditor	Core	Modification de fichiers PDF existants : filigranes, numéros de page, fusion, formulaires.
TsgcPDFTextExtractor	Core	Texte d'une page avec les polices et les positions des caractères.
TsgcPDFOptimizer	Core	Réduit la taille d'un fichier PDF.
TsgcPDFSignatureHandler	Core	Signe des documents PDF.
TsgcPDFSignatureVerifier	Core	Vérifie les signatures d'un document PDF.
TsgcPDFRedactor	Core	Supprime le texte, les images et les graphismes de zones des pages.

Composant	Paquet	Description
TsgcPDFImageExtractor	Core	Liste et enregistre les images d'un PDF.
TsgcPDFHTMLRenderer	Core	Convertit HTML et CSS en pages PDF.
TsgcPDFViewer	VCL	Une fenêtre native qui affiche, recherche et imprime un PDF.
TsgcPDFExporter	VCL	Effectue le rendu des pages en fichiers BMP, PNG ou JPEG et en vignettes.
TsgcPDFViewerControl	VCL	Contrôle de visualisation VCL : zoom, sélection, recherche, remplissage de formulaires, impression.
TsgcPDFThumbnailView	VCL	Vignettes de pages pour TsgcPDFViewerControl.
TsgcPDFOutlineView	VCL	Arborescence de signets pour TsgcPDFViewerControl.
TsgcPDFBarcodeControl	VCL	Contrôle de code-barres et de QR Code : onze symbologies, rotation, sortie PNG, SVG et PDF.
TsgcPDFViewerFMX	FMX	Contrôle de visualisation FireMonkey pour Windows, macOS, iOS, Android et Linux.
TsgcPDFBarcodeFMX	FMX	Contrôle FireMonkey de code-barres et de QR Code.

Déposez un composant sur la fiche et compilez le projet.

Utiliser sgcPDF sans installation

Tous les composants peuvent aussi être créés au moment de l'exécution. Chaque composant possède un constructeur qui ne nécessite aucune fiche, par exemple `TsgcPDFDocument.Create`, de sorte que les applications console, services, serveurs web et DLL n'ont besoin que du dossier **Source** dans le chemin de recherche. La plupart des exemples de cette aide sont des programmes console.

Architecture

sgcPDF est divisé en unités par fonctionnalité. Vous n'ajoutez à la clause *uses* que les unités dont vous utilisez les types.

Unités du noyau

Unité	Contenu
sgcPDF_Document	TsgcPDFDocument, TsgcPDFPage, TsgcPDFDocumentInfo
sgcPDF_Editor	TsgcPDFEditor, TsgcPDFWatermarkOptions, sgcPDFMergeFiles, sgcPDFExtractPages, sgcPDFGetPageCount
sgcPDF_Fonts	TsgcPDFStandardFont (les 14 polices standard), sgcPDFMeasureTextWidth
sgcPDF_Fonts_TrueType	TsgcTrueTypeFont : analyse, sous-ensemble, intégration et mise en forme TrueType et OpenType (CFF)
sgcPDF_Shaping	TsgcPDFShaper : mise en forme OpenType GSUB/GPOS, liaison arabe, algorithme bidirectionnel Unicode
sgcPDF_Fonts_CFF, sgcPDF_Fonts_Type1	Programmes de polices CFF et Type 1 utilisés par les moteurs de rendu
sgcPDF_Fonts_CJK	TsgcPDFCIDFont, TsgcPDFCJKFontManager (bas niveau, Windows uniquement)
sgcPDF_ContentStream	TsgcPDFContentBuilder (opérateurs de dessin de page), TsgcPDFContentParser
sgcPDF_Graphics	TsgcPDFExtGState, TsgcPDFShading, TsgcPDFPattern, TsgcPDFGraphicsResources
sgcPDF_Canvas	TsgcPDFCanvas : dessin de style TCanvas sur les pages PDF
sgcPDF_HTML	TsgcPDFHTMLRenderer : HTML et CSS vers PDF
sgcPDF_Images	TsgcPDFImage : chargement JPEG, PNG, BMP et TIFF
sgcPDF_JPEG, sgcPDF_JPX, sgcPDF_CCITT, sgcPDF_JBIG2	Décodeurs d'images : JPEG, JPEG 2000, fax CCITT et JBIG2
sgcPDF_ColorSpaces	Espaces colorimétriques et évaluation de fonctions partagés par les moteurs de rendu et l'extracteur d'images
sgcPDF_ImageExtract	TsgcPDFImageExtractor, sgcPDFEncodePNG, sgcPDFEncodeBMP
sgcPDF_Tables	TsgcPDFTable, TsgcPDFTableRow, TsgcPDFTableCell
sgcPDF_Barcodes	TsgcPDFBarcodeGenerator, TsgcPDFBarcodeOptions : codes-barres linéaires, QR Code, DataMatrix, PDF417, GS1-128
sgcPDF_Barcode_Symbol	TsgcPDFBarcodeSymbol : mise en page de code-barres indépendante du framework avec sortie SVG, PNG et PDF

Unité	Contenu
sgcPDF_Annotations	TsgcPDFAnnotationManager et les classes d'annotation
sgcPDF_Bookmarks	TsgcPDFBookmarks, TsgcPDFBookmarkItem
sgcPDF_Layers	TsgcPDFLayerManager, TsgcPDFLayer, TsgcPDFLayerGroup
sgcPDF_Forms	TsgcPDFAcroForm, TsgcPDFFormField
sgcPDF_Attachments	TsgcPDFAttachments : fichiers intégrés et pièces jointes PDF/A-3
sgcPDF_Tagged	Arbre de structure du PDF balisé
sgcPDF_Redaction	TsgcPDFRedactor
sgcPDF_Encryption	TsgcPDFEncryptionMethod, TsgcPDFEncryptionHandler, fonctions utilitaires de hachage et de chiffrement
sgcPDF_Signatures	TsgcPDFSignatureHandler, TsgcPDFSignatureVerifier
sgcPDF_PDFA	TsgcPDFALevel, TsgcPDFAValidator, TsgcPDFAConverter, TsgcPDFXMPMetadata
sgcPDF_Optimize	TsgcPDFOptimizer, TsgcPDFOptimizeOptions, sgcPDFOptimizeFile
sgcPDF_TextExtract	TsgcPDFTextExtractor, sgcPDFExtractPageText, sgcPDFExtractAllText, sgcPDFFindText, sgcPDFCreatePageTextExtractor
sgcPDF_Renderer	TsgcPDFRenderer : l'interpréteur de contenu indépendant de la plateforme
sgcPDF_Renderer_Canvas	TsgcPDFRenderCanvas : le moteur de dessin abstrait
sgcPDF_Renderer_Fonts	Programmes de polices et contours de glyphes pour les moteurs de rendu
sgcPDF_Renderer_Raster	TsgcPDFRendererRaster, TsgcPDFRasterCanvas, sgcPDFRasterAddFontDirectory
sgcPDF_Parser	TsgcPDFParser : le lecteur PDF de bas niveau
sgcPDF_Writer	TsgcPDFWriter : l'écrivain PDF de bas niveau
sgcPDF_Filters	Filtres de flux (Flate, LZW, ASCII85, ASCIIHex, RunLength, CCITT, JBIG2, prédicteurs) et sgcPDFMaxDecodedSize
sgcPDF_Types	Modèle d'objets PDF (TsgcPDFDictionary, TsgcPDFArray, TsgcPDFStream...), TsgcPDFRect, TBytes

Unités VCL (Windows)

Unité	Contenu
sgcPDF_Renderer_GDI	TsgcPDFRendererGDI, TsgcPDFGDICanvas : rendu sur un contexte de périphérique Windows

Unité	Contenu
sgcPDF_Export	TsgcPDFExporter, TsgcPDFExportOptions, sgcPDFPageToImage, sgcPDFToImages
sgcPDF_Viewer	TsgcPDFViewer
sgcPDF_ViewerVCL	TsgcPDFViewerControl, TsgcPDFThumbnailView, TsgcPDFOutlineView
sgcPDF_EMF	TsgcPDFEMFPlayer, sgcPDFEMFAddPage
sgcPDF_CanvasVCL	TsgcPDFVCLCanvasHelper
sgcPDF_Barcode_VCL	TsgcPDFBarcodeControl

Unités FireMonkey (XE5 et plus récent)

Unité	Contenu
sgcPDF_Renderer_FMX	TsgcPDFRendererFMX, TsgcPDFFMXCanvas, sgcPDFBGRAToFMXBitmap
sgcPDF_ViewerFMX	TsgcPDFViewerFMX
sgcPDF_Barcode_FMX	TsgcPDFBarcodeFMX, TsgcPDFBarcodeTextSettingsFMX

L'unité optionnelle sgcPDF_Sign_sgcSign (signatures PAdES via sgcSign) ne fait pas partie des paquets. Ajoutez-la à votre projet lorsque vous l'utilisez.

Coordonnées et unités

Toutes les positions et tailles sont exprimées en points PDF (1/72 pouce). L'origine est le coin **inférieur gauche** de la page et Y croît vers le haut. Une page A4 fait 595 x 842 points et une page US Letter 612 x 792 points. Les rectangles passés sous forme *Left, Bottom, Right, Top* utilisent le même espace de coordonnées. Les couleurs sont données sous forme de composants R, G, B entre 0 et 1. TsgcPDFCanvas fait exception : il utilise une origine en haut à gauche et l'unité que vous choisissez, comme un TCanvas de la VCL.

Index de page

Les index de page sont indexés à partir de zéro : la première page est 0 et la dernière page est `PageCount - 1`.

Chaînes

Les méthodes qui dessinent avec les polices standard prennent un `AnsiString` (encodage WinAnsi). Les méthodes qui dessinent avec des polices TrueType, le texte extrait, le contenu des annotations, les titres de signets et les valeurs de formulaire utilisent `WideString`, de sorte qu'ils prennent en charge tout caractère Unicode que la police contient.

Démarrage rapide

Les exemples suivants sont des procédures complètes issues de programmes console qui ont été compilés et exécutés. Ajoutez les unités indiquées dans chaque section à votre clause *uses*.

Créer un PDF

Unités : sgcPDF_Fonts, sgcPDF_Document.

```
procedure CreateHelloPDF(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oPage: TsgcPDFPage;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.Info.Title := 'Hello sgcPDF';
    oPDF.Info.Author := 'eSeGeCe';
    // A4 portrait: 595 x 842 points, origin at the bottom left corner
    oPage := oPDF.AddPage(595, 842);
    oPage.DrawTextColor('Hello sgcPDF', 50, 780, sfHelveticaBold, 24, 0, 0.2, 0.6);
    oPage.DrawLine(50, 770, 545, 770, 1.5);
    oPage.DrawText('A PDF written in pure Pascal.', 50, 740, sfHelvetica, 12);
    oPage.DrawRectangle(40, 40, 515, 762, 1.0);
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Ouvrir et modifier un PDF

Unités : sgcPDF_Fonts, sgcPDF_Editor. TsgcPDFEditor charge un document, applique des modifications à chaque page et l'enregistre.

```
procedure StampExistingPDF(const AInput, AOutput: string);
var
  oEditor: TsgcPDFEditor;
  oWatermark: TsgcPDFWatermarkOptions;
begin
  oEditor := TsgcPDFEditor.Create;
  try
    oEditor.LoadFromFile(AInput);
    oWatermark.Text := 'CONFIDENTIAL';
    oWatermark.Font := sfHelveticaBold;
    oWatermark.FontSize := 48;
    oWatermark.Rotation := 45;
    oWatermark.Opacity := 0.3;
    oWatermark.R := 0.5;
    oWatermark.G := 0.5;
    oWatermark.B := 0.5;
    oEditor.AddWatermarkAllPages(oWatermark);
    oEditor.AddHeaderText('Internal report', sfHelvetica, 9);
    oEditor.AddPageNumbers(sfHelvetica, 10, 'Page %d of %d');
    oEditor.AddText(0, 'Reviewed', 450, 60, sfHelveticaOblique, 11);
    oEditor.SaveToFile(AOutput);
  finally
    oEditor.Free;
  end;
end;
```

Extraire du texte

Unités : sgcPDF_Document.

```
procedure PrintAllText(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  i: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.LoadFromFile(AFileName);
    for i := 0 to oPDF.PageCount - 1 do
    begin
      WriteLn('--- page ', i + 1, ' ---');
      WriteLn(string(oPDF.ExtractTextW(i)));
    end;
    if oPDF.FindText(0, 'Hello') then
      WriteLn('Page 1 contains "Hello"');
  finally
    oPDF.Free;
  end;
end;
```

Effectuer le rendu d'une page en image

Unités : sgcPDF_Document, sgcPDF_Export.

```
procedure ExportPages(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oExporter: TsgcPDFExporter;
  oOptions: TsgcPDFExportOptions;
begin
  oPDF := TsgcPDFDocument.Create;
  oExporter := TsgcPDFExporter.Create;
  try
    oPDF.LoadFromFile(AFileName);
    oOptions := sgcPDFDefaultExportOptions;
    oOptions.Format := efPNG;
    oOptions.DPI := 150;
    oExporter.ExportPage(oPDF, 0, 'out\page1.png', oOptions);
    oExporter.ExportThumbnail(oPDF, 0, 'out\thumb1.bmp', 200, 200);
  finally
    oExporter.Free;
    oPDF.Free;
  end;
  // one call helpers that open the file themselves
  sgcPDFPageToImage(AFileName, 'out\page1_96.png', 0, 96);
end;
```

Visualiser un PDF

Unités : sgcPDF_Viewer. TsgcPDFViewer ouvre une fenêtre native. Dans un programme console, `RunMessageLoop` maintient la fenêtre active jusqu'à sa fermeture.

```
procedure ShowViewer(const AFileName: string);
var
  oViewer: TsgcPDFViewer;
begin
  oViewer := TsgcPDFViewer.Create;
  try
    oViewer.ViewMode := vmContinuous;
    // top level window; use CreateWindow(AParent, X, Y, W, H) to embed it
```

```
    oViewer.CreateWindow(100, 100, 800, 900, 'My PDF Viewer');
    oViewer.LoadFromFile(AFileName);
    oViewer.ZoomFit;
    oViewer.Show;
    oViewer.RunMessageLoop;
  finally
    oViewer.Free;
  end;
end;
```

Voir aussi

- Texte et polices
- Rendu et export

TsgcPDFDocument

Unité : sgcPDF_Document
La classe centrale de sgcPDF. Elle contient les pages d'un document, dessine du nouveau contenu, charge des fichiers existants et enregistre le résultat, éventuellement chiffré ou au format PDF/A. Elle donne également accès aux signets, annotations, champs de formulaire et calques du document.

Création et chargement

Méthode	Description
Create / Create(AOwner)	Crée un document vide.
LoadFromFile(const AFileName: string)	Charge un fichier PDF. Appelez d'abord SetPassword pour les fichiers chiffrés.
LoadFromStream(AStream: TStream)	Charge un PDF depuis un flux.
LoadFromBytes(const AData: TBytes)	Charge un PDF depuis la mémoire.
SaveToFile(const AFileName: string)	Écrit le document dans un fichier.
SaveToStream(AStream: TStream)	Écrit le document dans un flux.

Pages

Membre	Description
AddPage: TsgcPDFPage	Ajoute une page A4 (595 x 842 points).
AddPage(AWidth, AHeight: Double): TsgcPDFPage	Ajoute une page de la taille indiquée, en points.
InsertPage(AIndex: Integer; AWidth, AHeight: Double): TsgcPDFPage	Insère une page à une position donnée.
DeletePage(AIndex: Integer)	Supprime une page.
GetPage(AIndex: Integer) / Pages[AIndex]	Renvoie une page (propriété par défaut).
PageCount: Integer	Nombre de pages.
AppendDocument(ADoc: TsgcPDFDocument)	Ajoute à la suite les pages d'un autre document.

Ressources

Méthode	Description
AddTrueTypeFont(const AFontName: WideString; ABold, AItalic: Boolean): Integer	Charge une police TrueType installée et renvoie son index de police pour TsgcPDFPage.DrawTextTT.
AddImage(const AFileName: string): Integer	Charge un fichier JPEG, PNG ou BMP et renvoie son index d'image.
AddImageFromStream(AStream: TStream): Integer	Identique à AddImage, à partir d'un flux.

Texte

Méthode	Description
ExtractText(APageIndex: Integer): AnsiString	Texte d'une page sous forme d'AnsiString.
ExtractTextW(APageIndex: Integer): WideString	Texte d'une page sous forme Unicode.
FindText(APageIndex: Integer; const AText: WideString): Boolean	True lorsque la page contient le texte.
GetPageContentStream(APageIndex: Integer): TBytes	Le flux de contenu décodé d'une page.

Sécurité et conformité

Méthode	Description
SetEncryption(AMethod: TsgcPDFEncryptionMethod; const AUserPassword, AOwnerPassword: AnsiString)	Chiffre le document lors de son enregistrement.
SetPassword(const APassword: AnsiString)	Mot de passe utilisé pour ouvrir un fichier chiffré. Appelez cette méthode avant LoadFromFile.
SetPDFALevel(ALevel: TsgcPDFALevel)	Enregistre le document au format PDF/A-1b, PDF/A-2b ou PDF/A-3b.

Propriétés

Propriété	Type	Description
Info	TsgcPDFDocumentInfo	Title, Author, Subject, Keywords, Creator et Producer.
Bookmarks	TsgcPDFBookmarks	Plan du document. Voir Signets.
Annotations	TsgcPDFAnnotationManager	Annotations de toutes les pages. Voir Annotations.
AcroForm	TsgcPDFAcroForm	Champs de formulaire interactifs. Voir Formulaires.
Layers	TsgcPDFLayerManager	Groupes de contenu optionnel. Voir Calques.

Propriété	Type	Description
GraphicsResources	TsgcPDFGraphicsResources	États de transparence, dégradés et motifs.
IsLoaded	Boolean	True après le chargement d'un document.
Parser	TsgcPDFParser	L'analyseur d'un document chargé, utilisé par l'extracteur de texte, le validateur et le moteur de rendu.
PDFALevel	TsgcPDFALevel	Le niveau PDF/A défini avec SetPDFALevel.
Version (published)	AnsiString	Version PDF écrite dans l'en-tête du fichier.

TsgcPDFPage

Renvoyée par AddPage et GetPage. Ses méthodes de dessin ajoutent des opérateurs au contenu de la page.

Méthode	Description
DrawText(const AText: AnsiString; X, Y: Double; AFont: TsgcPDFStandardFont; AFontSize: Double)	Texte avec une police standard, à partir du point de ligne de base X, Y.
DrawTextColor(..., R, G, B: Double)	Identique, avec une couleur de texte.
DrawTextTT(const AText: WideString; X, Y: Double; AFontIndex: Integer; AFontSize: Double)	Texte Unicode avec une police TrueType ajoutée avec AddTrueTypeFont.
DrawTextTTColor(..., R, G, B: Double)	Identique, avec une couleur de texte.
DrawLine(X1, Y1, X2, Y2: Double; ALineWidth: Double = 1.0)	Ligne. DrawLineColor ajoute une couleur.
DrawRectangle(X, Y, W, H: Double; ALineWidth: Double = 1.0)	Contour de rectangle. DrawRectangleFill le remplit d'une couleur.
DrawCircle(CX, CY, Radius: Double; ALineWidth: Double = 1.0)	Contour de cercle. DrawCircleFill le remplit d'une couleur.
DrawImageFromIndex(AImageIndex: Integer; X, Y, W, H: Double)	Dessine une image ajoutée avec AddImage.
DrawImageFile(const AFileName: string; X, Y, W, H: Double)	Charge et dessine une image en un seul appel.
SetOpacity(AOpacity: Double)	Opacité (de 0 à 1) pour les opérations suivantes.
DrawRectangleTransparent(X, Y, W, H: Double; R, G, B, AOpacity: Double)	Rectangle rempli semi-transparent.
DrawLinearGradient(X, Y, W, H: Double; R0, G0, B0, R1, G1, B1: Double)	Rectangle rempli d'un dégradé linéaire.

Propriétés de la page : **Width**, **Height**, **MediaBox**, **PageIndex**, **PageDict** et **Content**, le TsgcPDFContentBuilder qui donne accès à chaque opérateur de dessin (voir Graphisme).

Événements

TsgcPDFDocument n'a aucun événement.

Exemple

```
procedure CreateHelloPDF(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oPage: TsgcPDFPage;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.Info.Title := 'Hello sgcPDF';
    oPDF.Info.Author := 'eSeGeCe';
    // A4 portrait: 595 x 842 points, origin at the bottom left corner
    oPage := oPDF.AddPage(595, 842);
    oPage.DrawTextColor('Hello sgcPDF', 50, 780, sfHelveticaBold, 24, 0, 0.2, 0.6);
    oPage.DrawLine(50, 770, 545, 770, 1.5);
    oPage.DrawText('A PDF written in pure Pascal.', 50, 740, sfHelvetica, 12);
    oPage.DrawRectangle(40, 40, 515, 762, 1.0);
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Voir aussi

- Texte et polices
- Fusionner, scinder et extraire des pages

TsgcPDFEditor

Unité : sgcPDF_Editor
Modification de haut niveau d'un PDF existant. L'éditeur encapsule un TsgcPDFDocument (propriété **Document**) et ajoute des opérations en un seul appel : filigranes, en-têtes, pieds de page, numéros de page, fusion, extraction de pages, remplissage et aplatissement de formulaires.

Constructeurs

Constructeur	Description
Create / Create(AOwner: TComponent)	Crée l'éditeur avec son propre document.
Create(ADocument: TsgcPDFDocument)	Modifie une instance TsgcPDFDocument existante.

Méthodes

Méthode	Description
LoadFromFile / SaveToFile	Charge et enregistre le document.
AddBlankPage / AddBlankPage(AWidth, AHeight: Double)	Ajoute une page vide à la fin.
DeletePage(AIndex: Integer)	Supprime une page.
RotatePage(AIndex: Integer; ADegrees: Integer)	Fait pivoter une page. ADegrees doit valoir 90, 180 ou 270.
AddText / AddTextColor(APageIndex, AText, X, Y, AFont, AFontSize [, R, G, B])	Dessine du texte sur une page.
AddTransparentText(APageIndex, AText, X, Y, AFont, AFontSize, AOpacity)	Dessine du texte semi-transparent.
AddLine / AddRectangle / AddRectangleFill	Dessine des formes sur une page.
AddImageToPage(APageIndex: Integer; const AImageFile: string; X, Y, W, H: Double)	Dessine un fichier image sur une page.
AddWatermark(APageIndex; const AOptions: TsgcPDFWatermarkOptions)	Texte pivoté et semi-transparent sur une page.
AddWatermarkAllPages(const AOptions: TsgcPDFWatermarkOptions)	Identique sur toutes les pages.
AddHeaderText / AddFooterText(const AText: AnsiString; AFont; AFontSize)	Texte en haut ou en bas de chaque page.
AddPageNumbers(AFont; AFontSize: Double; const AFormat: AnsiString)	Numéros de page sur chaque page. AFormat prend deux %d : page courante et nombre total de pages.
AddTextAnnotation / AddLinkAnnotation / AddHighlightAnnotation	Ajoute des annotations.

Méthode	Description
AddBookmark(const ATitle: WideString; APageIndex: Integer)	Ajoute un signet de premier niveau. AddBookmarkChild en ajoute un imbriqué.
MergeFrom(const AFileName: string)	Ajoute à la suite toutes les pages d'un autre PDF.
ExtractPages(AFromIndex, AToIndex: Integer; const AOutputFileName: string)	Enregistre une plage de pages dans un nouveau fichier.
ExtractText(APageIndex): AnsiString / ExtractTextUnicode(APageIndex): WideString / ExtractAllText	Extraction de texte.
SearchText(APageIndex: Integer; const AText: WideString): Boolean	True lorsque la page contient le texte.
FillFormField(const AFieldName: AnsiString; const AValue: WideString)	Définit la valeur d'un champ de formulaire.
FlattenForm	Transforme les champs de formulaire en contenu de page statique.
SetEncryption(AMethod; const AUserPassword, AOwnerPassword: AnsiString)	Chiffre la sortie.
ConvertToPDFA(ALevel: TsgcPDFALevel)	Enregistre la sortie au format PDF/A.
SignDocument(const AOutputFile: string; const ACertSubjectName, AReason, ALocation: WideString)	Enregistre et signe avec un certificat du magasin Windows, trouvé par son nom de sujet.

TsgcPDFWatermarkOptions

Enregistrement (record) avec les champs Text, Font, FontSize, Rotation (degrés), Opacity (de 0 à 1) et R, G, B.

Fonctions utilitaires

Fonction	Description
sgcPDFMergeFiles(const AInputFiles: array of string; const AOutputFile: string)	Fusionne plusieurs fichiers.
sgcPDFExtractPages(const AInputFile, AOutputFile: string; AFromPage, AToPage: Integer)	Extrait une plage de pages indexée à partir de zéro.
sgcPDFGetPageCount(const AFileName: string): Integer	Nombre de pages d'un fichier.

Événements

TsgcPDFEditor n'a aucun événement.

Exemple

```
procedure StampExistingPDF(const AInput, AOutput: string);
var
  oEditor: TsgcPDFEditor;
  oWatermark: TsgcPDFWatermarkOptions;
begin
  oEditor := TsgcPDFEditor.Create;
  try
    oEditor.LoadFromFile(AInput);
    oWatermark.Text := 'CONFIDENTIAL';
    oWatermark.Font := sfHelveticaBold;
    oWatermark.FontSize := 48;
    oWatermark.Rotation := 45;
    oWatermark.Opacity := 0.3;
    oWatermark.R := 0.5;
    oWatermark.G := 0.5;
    oWatermark.B := 0.5;
    oEditor.AddWatermarkAllPages(oWatermark);
    oEditor.AddHeaderText('Internal report', sfHelvetica, 9);
    oEditor.AddPageNumbers(sfHelvetica, 10, 'Page %d of %d');
    oEditor.AddText(0, 'Reviewed', 450, 60, sfHelveticaOblique, 11);
    oEditor.SaveToFile(AOutput);
  finally
    oEditor.Free;
  end;
end;
```

Voir aussi

- Fusionner, scinder et extraire des pages
- Formulaires

TsgcPDFViewer

Unité : sgcPDF_Viewer
Un lecteur PDF implémenté comme une fenêtre Windows native. Il ne dépend pas de la VCL, il fonctionne donc dans les applications VCL (intégré dans n'importe quel contrôle avec fenêtre via son Handle) et dans les programmes console ou non visuels (comme fenêtre de premier niveau). Les pages sont dessinées avec le moteur de rendu GDI.

Fenêtre

Méthode	Description
CreateWindow(AParent: HWND; X, Y, W, H: Integer)	Crée le lecteur comme fenêtre enfant, par exemple d'un TPanel.Handle.
CreateWindow(X, Y, W, H: Integer; const ATitle: AnsiString)	Crée une fenêtre de premier niveau.
Show / Hide / Invalidate / DestroyWindow	Gestion de la fenêtre.
RunMessageLoop	Exécute une boucle de messages jusqu'à la fermeture de la fenêtre. À utiliser uniquement dans les programmes sans boucle de messages propre, comme les programmes console.

Document et navigation

Méthode	Description
LoadFromFile(const AFileName: string)	Ouvre un fichier PDF.
LoadDocument(ADocument: TsgcPDFDocument; AOwns: Boolean = False)	Affiche un document déjà en mémoire.
CloseDocument	Ferme le document.
FirstPage / LastPage / NextPage / PrevPage / GotoPage(AIndex)	Navigation.
ZoomIn / ZoomOut / ZoomFit / ZoomFitPage / SetZoom(APercent: Double)	Zoom.
FindText(const AText: WideString): Boolean / FindNext / ClearSearch	Recherche de texte.
Print / PrintPage(APageIndex: Integer)	Impression.
GetPageText(APageIndex: Integer): WideString	Texte d'une page.
ExportPageToBitmap(APageIndex: Integer; const AFileName: string; AWidth, AHeight: Integer)	Effectue le rendu d'une page vers un fichier BMP.

La fenêtre répond aussi au clavier : Gauche et Page précédente vont à la page précédente, Droite et Page suivante à la suivante, Origine et Fin à la première et à la dernière page, les touches plus et moins zooment, et Haut et Bas font défiler.

Propriétés

Propriété	Type	Description
ViewMode (published)	TsgcPDFViewer-Mode	vmSinglePage (par défaut) ou vmContinuous.
BackColor (published)	COLORREF	Couleur d'arrière-plan autour des pages. Par défaut $00C8C8C8.
ShowAnnotations (published)	Boolean	Dessine les annotations. Par défaut True.
Handle	HWND	Handle de fenêtre.
CurrentPage	Integer	Index de la page courante, à partir de zéro.
PageCount	Integer	Nombre de pages.
Zoom	Double	Zoom courant.
Document	TsgcPDFDocument	Le document affiché.

Événements

TsgcPDFViewer n'a aucun événement.

Exemple

```
procedure ShowViewer(const AFileName: string);
var
  oViewer: TsgcPDFViewer;
begin
  oViewer := TsgcPDFViewer.Create;
  try
    oViewer.ViewMode := vmContinuous;
    // top level window; use CreateWindow(AParent, X, Y, W, H) to embed it
    oViewer.CreateWindow(100, 100, 800, 900, 'My PDF Viewer');
    oViewer.LoadFromFile(AFileName);
    oViewer.ZoomFit;
    oViewer.Show;
    oViewer.RunMessageLoop;
  finally
    oViewer.Free;
  end;
end;
```

Voir aussi

- Rendu et export

TsgcPDFTextExtractor

Unité : sgcPDF_TextExtract
Extrait le texte d'une page, en conservant la position, le nom de police, la taille de police et la largeur de chaque fragment de texte. Il décode les encodages standard, les Differences de police, les tables ToUnicode et les polices CID, de sorte que le texte écrit avec des polices TrueType et CJK est renvoyé en Unicode.

Constructeurs

Constructeur	Description
Create(AParser: TsgcPDFParser; APage-Dict: TsgcPDFDictionary)	Prépare l'extraction d'une page. Passez Parser et PageDict de TsgcPDFPage d'un document chargé.
Create(AOwner: TComponent)	Constructeur de composant, utilisé par l'IDE. Utilisez le constructeur ci-dessus pour extraire du texte dans le code.

Méthodes

Méthode	Description
Extract	Analyse le contenu de la page. À appeler avant les autres méthodes.
GetText: WideString	Le texte de la page dans l'ordre de lecture, une ligne par ligne de texte.
GetTextWithPositions: WideString	Le texte avec les coordonnées de chaque fragment.
GetFragmentCount: Integer	Nombre de fragments.
GetFragment(AIndex: Integer): TsgcPDFTextFragment	Un fragment : Text, X, Y, FontSize, FontName et Width.
FindText(const ASearchText: WideS-tring; ACaseSensitive: Boolean = False): Boolean	Recherche dans le texte de la page.

Fonctions utilitaires

Fonction	Description
sgcPDFExtractPageText(AParser: TsgcPDFParser; APageIndex: Integer): WideS-tring	Texte d'une page.
sgcPDFExtractAllText(AParser: TsgcPDFParser): WideString	Texte de toutes les pages.
sgcPDFFindText(AParser: TsgcPDFParser; APageIndex: Integer; const AText: Wi-deString): Boolean	Recherche dans une page.

Un TsgcPDFParser utilisé directement doit être chargé puis analysé : appelez `LoadFromFile` puis `Parse`.

Événements

TsgcPDFTextExtractor n'a aucun événement.

Exemple

```
procedure PrintTextPositions(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oExtractor: TsgcPDFTextExtractor;
  oFragment: TsgcPDFTextFragment;
  i: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.LoadFromFile(AFileName);
    oExtractor := TsgcPDFTextExtractor.Create(oPDF.Parser,
      oPDF.Pages[0].PageDict);
    try
      oExtractor.Extract;
      for i := 0 to oExtractor.GetFragmentCount - 1 do
      begin
        oFragment := oExtractor.GetFragment(i);
        WriteLn(Format('%.1f, %.1f  %s  (%s %.1f pt)', [oFragment.X, oFragment.Y,
          string(oFragment.Text), string(oFragment.FontName), oFragment.FontSize]));
      end;
    finally
      oExtractor.Free;
    end;
  finally
    oPDF.Free;
  end;
end;
```

Voir aussi

- Extraction de texte

TsgcPDFExporter

Unité : sgcPDF_Export
Effectue le rendu des pages d'un TsgcPDFDocument en images BMP, PNG ou JPEG à l'aide du moteur de rendu GDI (TsgcPDFRendererGDI).

Méthodes

Méthode	Description
ExportPage(ADocument: TsgcPDFDocument; APageIndex: Integer; const AFileName: string)	Effectue le rendu d'une page avec la propriété Options.
ExportPage(ADocument; APageIndex; const AFileName: string; const AOptions: TsgcPDFExportOptions)	Effectue le rendu d'une page avec des options explicites.
ExportPageToStream(ADocument; APageIndex; AStream: TStream; const AOptions)	Effectue le rendu d'une page vers un flux.
ExportAllPages(ADocument; const AOutputDir, AFilePrefix: string [; const AOptions])	Effectue le rendu de toutes les pages vers un dossier.
ExportPages(ADocument; AFromPage, AToPage: Integer; const AOutputDir, AFilePrefix: string; const AOptions)	Effectue le rendu d'une plage de pages.
ExportThumbnail(ADocument; APageIndex; const AFileName: string; AMaxWidth, AMaxHeight: Integer)	Effectue le rendu d'une page mise à l'échelle pour s'ajuster à la zone indiquée.
ExportAllThumbnails(ADocument; const AOutputDir, AFilePrefix: string; AMaxWidth, AMaxHeight: Integer)	Vignettes de toutes les pages.

TsgcPDFExportOptions

Champ	Par défaut	Description
Format	efPNG	efBMP, efPNG ou efJPEG.
DPI	150	Résolution. 72 DPI restitue un pixel par point.
JPEGQuality	85	Qualité JPEG, de 1 à 100.
BackgroundColor	$00FFFFFF	Arrière-plan de la page (COLORREF).
AntiAlias	False	Lisse le rendu.

Obtenez les valeurs par défaut avec `sgcPDFDefaultExportOptions`.

Fonctions utilitaires

Fonction	Description
sgcPDFPageToImage(const APDFFile, AImageFile: string; APageIndex: Integer; ADPI: Integer = 150)	Ouvre un fichier et effectue le rendu d'une page.
sgcPDFToImages(const APDFFile, AOutputDir: string; AFormat: TsgcPDFExportFormat = efPNG; ADPI: Integer = 150)	Ouvre un fichier et effectue le rendu de toutes les pages.

Événements

TsgcPDFExporter n'a aucun événement.

Exemple

```
procedure ExportPages(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oExporter: TsgcPDFExporter;
  oOptions: TsgcPDFExportOptions;
begin
  oPDF := TsgcPDFDocument.Create;
  oExporter := TsgcPDFExporter.Create;
  try
    oPDF.LoadFromFile(AFileName);
    oOptions := sgcPDFDefaultExportOptions;
    oOptions.Format := efPNG;
    oOptions.DPI := 150;
    oExporter.ExportPage(oPDF, 0, 'out\page1.png', oOptions);
    oExporter.ExportThumbnail(oPDF, 0, 'out\thumb1.bmp', 200, 200);
  finally
    oExporter.Free;
    oPDF.Free;
  end;
  // one call helpers that open the file themselves
  sgcPDFPageToImage(AFileName, 'out\page1_96.png', 0, 96);
end;
```

Voir aussi

- Rendu et export

TsgcPDFOptimizer

Unité : sgcPDF_Optimize
Réécrit un fichier PDF pour le rendre plus petit : il compresse les flux non compressés, supprime les objets dupliqués et non référencés, et peut sous-échantillonner les images et supprimer du contenu optionnel tel que les métadonnées, les signets, les annotations, le JavaScript et les vignettes.

Méthodes

Méthode	Description
Optimize(const AInputFile, AOutputFile: string)	Optimise un fichier.
OptimizeStream(AInputStream, AOutputStream: TStream)	Optimise un flux.
GetSizeReduction: Int64	Octets économisés par la dernière optimisation.
GetLog: TStringList	Rapport de la dernière optimisation. Appartient à l'optimiseur, ne pas le libérer.

TsgcPDFOptimizeOptions

Champ	Par défaut	Description
CompressStreams	True	Compresse en Flate les flux non compressés.
RemoveDuplicateObjects	True	Fusionne les objets identiques.
RemoveUnusedObjects	True	Supprime les objets inaccessibles depuis le document.
DownsampleImages	False	Réduit les images dont la résolution dépasse MaxImageDPI.
MaxImageDPI	150	Résolution cible pour le sous-échantillonnage.
JPEGQuality	75	Qualité des images recompressées.
RemoveMetadata	False	Supprime les métadonnées XMP.
RemoveBookmarks	False	Supprime le plan (outline).
RemoveAnnotations	False	Supprime les annotations.
RemoveJavaScript	False	Supprime les actions JavaScript.
FlattenForms	False	Aplatit les champs de formulaire.
RemoveThumbnails	False	Supprime les vignettes de page intégrées.

Obtenez les valeurs par défaut avec `sgcPDFDefaultOptimizeOptions`, ou optimisez un fichier avec les valeurs par défaut en un seul appel avec `sgcPDFOptimizeFile(const AInputFile, AOutputFile: string)`.

Événements

TsgcPDFOptimizer n'a aucun événement.

Exemple

```
procedure OptimizePDF(const AInput, AOutput: string);
var
  oOptimizer: TsgcPDFOptimizer;
  oOptions: TsgcPDFOptimizeOptions;
begin
  oOptimizer := TsgcPDFOptimizer.Create;
  try
    oOptions := sgcPDFDefaultOptimizeOptions;
    oOptions.DownsampleImages := True;
    oOptions.MaxImageDPI := 150;
    oOptions.RemoveMetadata := True;
    oOptimizer.Options := oOptions;
    oOptimizer.Optimize(AInput, AOutput);
    WriteLn('Saved ', oOptimizer.GetSizeReduction, ' bytes');
    WriteLn(oOptimizer.GetLog.Text);
  finally
    oOptimizer.Free;
  end;
end;
```

Voir aussi

- Optimisation des fichiers PDF

TsgcPDFSignatureHandler

Unité : sgcPDF_Signatures
Signe des documents PDF avec un certificat X.509 et sa clé privée. La signature est une signature PKCS#7 (CMS) détachée (*/SubFilter /adbe.pkcs7.detached*) créée via la CryptoAPI de Windows, et elle est ajoutée sous forme de mise à jour incrémentale afin que les octets d'origine restent inchangés.

Certificat

Méthode	Description
LoadCertificateFromPFX(const AFileName: string; const APassword: AnsiString)	Charge un certificat et sa clé privée depuis un fichier PFX (PKCS#12).
LoadCertificateFromStore(const ASubjectName: WideString)	Recherche un certificat par nom de sujet dans le magasin MY de l'utilisateur courant.
LoadCertificateFromThumbprint(const AThumbprint: AnsiString)	Recherche un certificat par empreinte SHA-1.
ListCertificates: TStringList	Certificats du magasin. La liste doit être libérée par l'appelant.
GetCertificateInfo: TsgcPDFCertificateInfo	SubjectName, IssuerName, SerialNumber, NotBefore, NotAfter et Thumbprint du certificat chargé.

Signature

Méthode	Description
SetField(const AName: AnsiString; APageIndex: Integer; ALeft, ABottom, ARight, ATop: Double)	Champ de signature visible sur une page.
SetInvisible	Signature invisible.
SignDocument(const AInputFile, AOutputFile: string)	Signe un fichier.
SignDocumentStream(AInputStream, AOutputStream: TStream)	Signe un flux.

Propriétés

Propriété	Type	Description
SignerName (published)	WideString	Nom du signataire écrit dans la signature. Identique à Name.
Reason (published)	WideString	Motif de la signature.
Location (published)	WideString	Lieu de signature.

Propriété	Type	Description
ContactInfo (published)	AnsiString	Coordonnées de contact.
HashAlgorithm (published)	TsgcPDFSignatureHashAlgorithm	shaSHA1, shaSHA256 (par défaut), shaSHA384 ou shaSHA512.
Appearance	TsgcPDFSignatureAppearance	Apparence de signature visible : ShowName, ShowDate, ShowReason, ShowLocation, Text, FontName, FontSize, ImageData.
CertInfo	TsgcPDFCertificateInfo	Informations du certificat chargé.

Événements

TsgcPDFSignatureHandler n'a aucun événement.

Exemple

```
procedure SignPDF(const AInput, AOutput: string);
var
  oSigner: TsgcPDFSignatureHandler;
begin
  oSigner := TsgcPDFSignatureHandler.Create;
  try
    oSigner.LoadCertificateFromPFX('signer.pfx', 'test123');
    WriteLn('Signing as ', string(oSigner.GetCertificateInfo.SubjectName));
    oSigner.Reason := 'Document approval';
    oSigner.Location := 'Barcelona';
    oSigner.HashAlgorithm := shaSHA256;
    // visible signature field on page 1: Left, Bottom, Right, Top
    oSigner.SetField('Signature1', 0, 50, 50, 250, 100);
    oSigner.SignDocument(AInput, AOutput);
  finally
    oSigner.Free;
  end;
end;
```

Voir aussi

- TsgcPDFSignatureVerifier
- Signatures numériques

TsgcPDFSignatureVerifier

Unité : sgcPDF_Signatures
Vérifie les signatures numériques d'un document PDF : contrôle que chaque signature est cryptographiquement valide, que les plages d'octets signées couvrent toujours l'ensemble du fichier, et que le certificat du signataire remonte à une autorité racine de confiance dans le magasin de certificats Windows.

Méthodes

Méthode	Description
LoadDocument(const AFileName: string)	Charge un PDF signé.
GetSignatureCount: Integer	Nombre de signatures.
VerifySignature(AIndex: Integer): Boolean	True lorsque la signature correspond aux octets signés.
IsDocumentModified(AIndex: Integer): Boolean	True lorsque le fichier a été modifié après cette signature.
IsSignatureTrusted(AIndex: Integer): Boolean	True lorsque la chaîne de certification est fiable pour Windows.
GetSignatureInfo(AIndex: Integer): TsgcPDFCertificateInfo	Informations de certificat du signataire.

Un certificat de test auto-signé se vérifie comme valide mais non fiable.

Événements

TsgcPDFSignatureVerifier n'a aucun événement.

Exemple

```
procedure VerifyPDF(const AFileName: string);
var
  oVerifier: TsgcPDFSignatureVerifier;
  oCert: TsgcPDFCertificateInfo;
  i: Integer;
begin
  oVerifier := TsgcPDFSignatureVerifier.Create;
  try
    oVerifier.LoadDocument(AFileName);
    for i := 0 to oVerifier.GetSignatureCount - 1 do
    begin
      oCert := oVerifier.GetSignatureInfo(i);
      WriteLn('Signer:   ', string(oCert.SubjectName));
      WriteLn('Valid:    ', oVerifier.VerifySignature(i));
      WriteLn('Modified: ', oVerifier.IsDocumentModified(i));
      WriteLn('Trusted:  ', oVerifier.IsSignatureTrusted(i));
    end;
  finally
    oVerifier.Free;
  end;
end;
```

Voir aussi

- TsgcPDFSignatureHandler
- Signatures numériques

Texte et polices

sgcPDF dessine le texte avec deux types de polices : les 14 polices PDF standard, que tout lecteur PDF fournit, et les polices TrueType, qui sont intégrées au fichier.

Polices standard

Les polices standard se sélectionnent avec **TsgcPDFStandardFont** (unité sgcPDF_Fonts) : sfCourier, sfCourierBold, sfCourierOblique, sfCourierBoldOblique, sfHelvetica, sfHelveticaBold, sfHelveticaOblique, sfHelveticaBoldOblique, sfTimesRoman, sfTimesBold, sfTimesItalic, sfTimesBoldItalic, sfSymbol et sfZapfDingbats. Elles ne sont pas intégrées, ce qui garde les fichiers petits, mais elles ne couvrent que le jeu de caractères WinAnsi. Utilisez-les avec `TsgcPDFPage.DrawText` et `DrawTextColor`.
`sgcPDFMeasureTextWidth(AFont, AText, AFontSize)` renvoie la largeur d'une chaîne en points, ce qui permet de centrer le texte ou de l'aligner à droite.

TrueType et Unicode

`TsgcPDFDocument.AddTrueTypeFont(AFontName, ABold, AItalic)` charge une police Windows installée par son nom de famille et renvoie un index de police. Le texte dessiné avec `DrawTextTT` ou `DrawTextTTColor` est une WideString, de sorte que tout caractère contenu dans la police peut être utilisé. Lors de l'enregistrement du document, seuls les glyphes réellement utilisés sont intégrés (sous-ensemble de police) avec une table ToUnicode, afin que le texte puisse être recherché, copié et extrait.

```
procedure CreateFontsPDF(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oPage: TsgcPDFPage;
  vArial, vTimes: Integer;
  vWidth: Double;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    // TrueType fonts are loaded from the installed system fonts, subset and embedded
    vArial := oPDF.AddTrueTypeFont('Arial', False, False);
    vTimes := oPDF.AddTrueTypeFont('Times New Roman', True, False);
    oPage := oPDF.AddPage(595, 842);
    // the 14 standard fonts need no embedding
    oPage.DrawText('Standard Helvetica', 50, 780, sfHelvetica, 14);
    oPage.DrawText('Standard Courier', 50, 760, sfCourier, 14);
    // TrueType text is Unicode (WideString)
    oPage.DrawTextTT('Arial TrueType', 50, 720, vArial, 14);
    oPage.DrawTextTTColor('Times New Roman Bold', 50, 700, vTimes, 14, 0.6, 0, 0);
    oPage.DrawTextTT('Cyrillic: ' + WideString(#$041F#$0440#$0438#$0432#$0435#$0442),
      50, 670, vArial, 14);
    oPage.DrawTextTT('Greek: ' + WideString(#$0391#$03BB#$03C6#$03B1), 50, 650,
      vArial, 14);
    // measure a standard font string to right align it at x = 545
    vWidth := sgcPDFMeasureTextWidth(sfHelvetica, 'Right aligned', 12);
    oPage.DrawText('Right aligned', 545 - vWidth, 620, sfHelvetica, 12);
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Texte CJK

Le texte chinois, japonais et coréen fonctionne de la même façon : ajoutez une police TrueType contenant les caractères et dessinez avec DrawTextTT. La police est intégrée comme police CID avec l'encodage Identity-H.

L'énumération **TsgcPDFCJKFont** (unité sgcPDF_Fonts_CJK) répertorie les polices CJK Windows courantes et `sgcPDFCJKFontNameFromEnum` renvoie leurs noms de famille :

Valeur	Police	Écriture
cjkMSGothic, cjkMSMincho, cjkMeiryo	MS Gothic, MS Mincho, Meiryo	Japonais
cjkSimSun, cjkSimHei, cjkMicrosoftYaHei	SimSun, SimHei, Microsoft YaHei	Chinois simplifié
cjkMingLiU, cjkMicrosoftJhengHei	MingLiU, Microsoft JhengHei	Chinois traditionnel
cjkBatang, cjkGulim, cjkMalgunGothic	Batang, Gulim, Malgun Gothic	Coréen

```
procedure CreateCJKPDF(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oPage: TsgcPDFPage;
  vKorean: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    // any installed TrueType font that covers the script works the same way
    vKorean := oPDF.AddTrueTypeFont(sgcPDFCJKFontNameFromEnum(cjkMalgunGothic),
      False, False);
    oPage := oPDF.AddPage(595, 842);
    // "Annyeonghaseyo" in Hangul
    oPage.DrawTextTT(WideString(#$C548#$B155#$D558#$C138#$C694), 50, 780,
      vKorean, 20);
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

La police doit être installée sur la machine qui crée le PDF. Le lecteur du PDF n'en a pas besoin, car les glyphes sont intégrés.

Voir aussi

- TsgcPDFDocument
- Extraction de texte
- PDF/A

Images

sgcPDF intègre les images JPEG, PNG et BMP. Les données JPEG sont stockées telles quelles (DCTDecode) sans recompression. Les pixels PNG et BMP sont compressés en Flate.
Ajoutez une image une seule fois avec `TsgcPDFDocument.AddImage` (ou `AddImageFromStream`) et dessinez-la autant de fois que nécessaire avec `TsgcPDFPage.DrawImageFromIndex`. Le fichier ne stocke l'image qu'une seule fois. `DrawImageFile` charge et dessine un fichier en un seul appel. X et Y désignent le coin inférieur gauche de l'image, et W et H sa taille sur la page, en points.

```
procedure CreateImagesPDF(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oPage: TsgcPDFPage;
  vLogo: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    // JPEG is embedded as is (DCTDecode), PNG and BMP are recompressed
    vLogo := oPDF.AddImage('img.png');
    oPage := oPDF.AddPage(595, 842);
    // the image is added once and can be drawn many times
    oPage.DrawImageFromIndex(vLogo, 50, 600, 200, 100);
    oPage.DrawImageFromIndex(vLogo, 300, 600, 100, 50);
    // or load and draw a file in one call
    oPage.DrawImageFile('img.jpg', 50, 400, 300, 150);
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

La classe **TsgcPDFImage** (unité sgcPDF_Images) charge et inspecte une image sans document : Width, Height, BitsPerComponent, ColorSpace, HasAlpha. Les autres formats, comme TIFF, déclenchent une exception.

Voir aussi

- TsgcPDFDocument
- Optimisation des fichiers PDF

Graphisme, transparence et dégradés

TsgcPDFPage dispose de méthodes utilitaires pour les formes courantes : DrawLine, DrawRectangle, DrawRectangleFill, DrawCircle, DrawCircleFill et leurs variantes en couleur. Pour la transparence, utilisez `SetOpacity` (elle affecte les opérations qui la suivent) ou `DrawRectangleTransparent`. `DrawLinearGradient` remplit un rectangle avec un dégradé de couleur, dessiné sous forme de fines bandes de couleur allant de la première couleur à la seconde.

Le générateur de contenu

`TsgcPDFPage.Content` renvoie le **TsgcPDFContentBuilder** de la page (unité sgcPDF_ContentStream). Il expose un par un les opérateurs de dessin PDF, et tout ce qu'il écrit va dans le même contenu de page que les méthodes utilitaires :

- État graphique : SaveState, RestoreState, SetMatrix, SetLineWidth, SetLineCap, SetLineJoin, SetMiterLimit, SetDashPattern.
- Couleurs : SetStrokeColorRGB, SetFillColorRGB, SetStrokeColorCMYK, SetFillColorCMYK, SetStrokeGray, SetFillGray.
- Chemins : MoveTo, LineTo, CurveTo, CurveToV, CurveToY, Rectangle, ClosePath.
- Peinture : Stroke, CloseAndStroke, Fill, FillEvenOdd, FillAndStroke, EndPath, ClipNonZero, ClipEvenOdd.
- Texte : BeginText, EndText, SetFont, SetCharSpacing, SetWordSpacing, SetTextRise, SetTextRenderMode, SetTextScale, MoveTextPos, SetTextMatrix, ShowText.
- Autre : SetGraphicsState, DrawXObject, BeginMarkedContent, EndMarkedContent et AddOperator pour écrire n'importe quel opérateur.

Faites toujours correspondre SaveState avec RestoreState afin que les modifications ne se propagent pas au dessin suivant.

```
procedure CreateGraphicsPDF(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oPage: TsgcPDFPage;
  oContent: TsgcPDFContentBuilder;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPage := oPDF.AddPage(595, 842);
    // shapes, gradients and transparency through the page helpers
    oPage.DrawLinearGradient(50, 680, 495, 100, 0.2, 0.4, 0.8, 0.9, 0.9, 1.0);
    oPage.DrawRectangleFill(70, 520, 200, 120, 1, 1, 0);
    oPage.DrawRectangleTransparent(150, 550, 200, 120, 1, 0, 0, 0.5);
    oPage.SetOpacity(0.6);
    oPage.DrawCircleFill(450, 580, 60, 0, 0.5, 0);
    oPage.SetOpacity(1.0);
    // low level path operators through the content builder
    oContent := oPage.Content;
    oContent.SaveState;
    oContent.SetStrokeColorCMYK(0, 1, 1, 0);
    oContent.SetLineWidth(3);
    oContent.SetDashPattern([6, 3], 0);
    oContent.MoveTo(50, 400);
    oContent.CurveTo(150, 500, 250, 300, 350, 400);
    oContent.Stroke;
    oContent.RestoreState;
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Voir aussi

- TsgcPDFDocument

- Calques

Tableaux

TsgcPDFTable (unité sgcPDF_Tables) met en page des lignes et des colonnes, ajuste le texte à l'intérieur des cellules et dessine les bordures et les arrière-plans.

- Créez-le avec le nombre de colonnes et définissez les largeurs avec SetColumnWidths, SetColumnWidth ou SetColumnWidthPercent.
- SetHeaders ajoute une ligne d'en-tête. AddRowData ajoute une ligne de données à partir d'un tableau de chaînes. AddRow et AddHeaderRow renvoient un TsgcPDFTableRow pour un contrôle complet sur ses cellules (TsgcPDFTableCell avec Style, ColSpan et RowSpan).
- Style : SetDefaultFont, SetHeaderFont, SetHeaderBackground, SetAlternateRowColors, SetBorder, SetPadding, ou les enregistrements DefaultStyle et HeaderStyle.

Dessin

Méthode	Description
DrawOnPage(APage: TObject; AX, AY: Double)	Dessine le tableau entier sur un TsgcPDFPage. AX, AY est le coin supérieur gauche.
DrawOnPageWithOverflow(APage: TObject; AX, AY, AMaxHeight: Double): Integer	Dessine les lignes d'en-tête et les lignes de données qui tiennent dans AMaxHeight, et renvoie l'index de la première ligne qui ne tenait pas.
DrawMultiPage(ADocument: TObject; AX, AY, AMaxHeight: Double; APageWidth, APageHeight: Double)	Ajoute une page de la taille indiquée à un TsgcPDFDocument, dessine le tableau à partir de AX, AY et ajoute d'autres pages si nécessaire, en répétant les lignes d'en-tête sur chaque page.

```
procedure CreateTablePDF(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oTable: TsgcPDFTable;
  i: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oTable := TsgcPDFTable.Create(3);
    try
      oTable.SetColumnWidths([200, 150, 145]);
      oTable.SetHeaders(['Product', 'Quantity', 'Price']);
      oTable.SetHeaderBackground(0.2, 0.4, 0.7);
      oTable.SetAlternateRowColors(0.93, 0.95, 1.0);
      oTable.SetBorder(0.5, 0.6, 0.6, 0.6);
      for i := 1 to 80 do
        oTable.AddRowData([AnsiString('Item ' + IntToStr(i)),
          AnsiString(IntToStr(i * 3)), AnsiString(Format('%d.00', [i * 10]))]);
      // rows that do not fit continue on new A4 pages
      oTable.DrawMultiPage(oPDF, 50, 790, 740, 595, 842);
    finally
      oTable.Free;
    end;
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Voir aussi

- TsgcPDFDocument

Codes-barres

TsgcPDFBarcodeGenerator (unité sgcPDF_Barcodes) dessine des codes-barres sous forme de graphismes vectoriels, de sorte qu'ils s'impriment nettement à toute taille. Symbologies prises en charge (TsgcPDFBarcodeType) : bcCode128, bcCode39, bcEAN13, bcEAN8, bcUPCA, bcITF, bcCodabar, bcQRCode, bcDataMatrix, bcPDF417 et bcGS1_128. Les versions de QR Code et les niveaux de correction d'erreur, DataMatrix, PDF417 et GS1-128 sont décrits dans Codes-barres 2D et GS1-128.
Le premier paramètre des méthodes Draw est la page (TsgcPDFPage) ou son **générateur de contenu**, `TsgcPDFPage.Content`. Tout autre objet déclenche une exception. AX et AY désignent le coin inférieur gauche.

Méthode	Description
DrawBarcode(APage: TObject; ABarcodeType; const AData: AnsiString; AX, AY: Double)	Dessine avec les options par défaut.
DrawBarcode(APage: TObject; ABarcodeType; const AData: AnsiString; AX, AY: Double; const AOptions: TsgcPDFBarcodeOptions)	Dessine avec des options personnalisées.
DrawQRCode(APage: TObject; const AData: AnsiString; AX, AY, AModuleSize: Double)	Dessine un QR Code dont les modules ont une largeur de AModuleSize points.
EncodeLinear(ABarcodeType: TsgcPDFBarcodeType; const AData: AnsiString): TBytes	Renvoie les largeurs des barres et des espaces en modules d'une symbologie linéaire (Code 128, Code 39, EAN-13, EAN-8, UPC-A, ITF, Codabar et GS1-128), en commençant par une barre : les mêmes largeurs que celles que dessine DrawBarcode. Les types 2D lèvent une exception, ils ont leurs propres encodeurs, voir Codes-barres 2D et GS1-128.

TsgcPDFBarcodeOptions

Obtenez les valeurs par défaut avec `sgcPDFDefaultBarcodeOptions` : BarWidth 1.0, Height 50, ShowText True, TextFont sfHelvetica, TextFontSize 8, QuietZone 10, premier plan noir et arrière-plan blanc (ForegroundR/G/B, BackgroundR/G/B).

```
procedure CreateBarcodesPDF(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oPage: TsgcPDFPage;
  oBarcodes: TsgcPDFBarcodeGenerator;
  oOptions: TsgcPDFBarcodeOptions;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPage := oPDF.AddPage(595, 842);
    // barcodes are drawn into the page content builder
    oBarcodes := TsgcPDFBarcodeGenerator.Create;
    try
      oBarcodes.DrawBarcode(oPage.Content, bcCode128, 'SGC-2026-0001', 50, 700);
      oOptions := sgcPDFDefaultBarcodeOptions;
      oOptions.Height := 60;
      oOptions.BarWidth := 1.2;
      oBarcodes.DrawBarcode(oPage.Content, bcEAN13, '5901234123457', 50, 580, oOptions);
      // QR code, 3 points per module
      oBarcodes.DrawQRCode(oPage.Content, 'https://www.esegece.com', 350, 580, 3);
    finally
      oBarcodes.Free;
    end;
```

```
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Contrôles de code-barres

Pour afficher un code-barres sur une fiche, déposez un TsgcPDFBarcodeControl (VCL) ou un TsgcPDFBarcodeFMX (FireMonkey) et définissez Symbology et Text. Les deux contrôles prennent en charge les onze symbologies, y compris les symboles 2D et GS1-128 décrits dans Codes-barres 2D et GS1-128, et dessinent chaque module sur des pixels entiers. Ils hébergent un TsgcPDFBarcodeSymbol, qui fonctionne aussi sans fiche, par exemple dans un service ou sous Linux :

- SaveToPNG et SaveToSVG enregistrent le symbole sous forme d'image.
- DrawToPDF dessine le symbole sur une page PDF sous forme de rectangles vectoriels, avec les mêmes modules, la même rotation, la même zone de silence et la même ligne de texte que le contrôle.
- Runs, ModuleAt et LayoutAxes renvoient les modules et la disposition en pixels pour un dessin personnalisé.

Voir aussi

- TsgcPDFDocument
- Codes-barres 2D et GS1-128
- TsgcPDFBarcodeControl
- TsgcPDFBarcodeSymbol

Annotations

Les annotations sont gérées par `TsgcPDFDocument.Annotations` (TsgcPDFAnnotationManager, unité sgcPDF_Annotations). Chaque méthode Add prend l'index de page à partir de zéro et renvoie la nouvelle annotation afin que vous puissiez définir sa couleur (SetColor), Author, Contents, Opacity et Flags.

Méthode	Annotation
AddText(APageIndex; ALeft, ABottom, ARight, ATop: Double; const AText: WideString)	Note autocollante (TsgcPDFTextAnnotation, avec Icon et Open).
AddLink(APageIndex; ALeft, ABottom, ARight, ATop: Double; ADestPage: Integer)	Lien vers une autre page.
AddLinkURI(APageIndex; ALeft, ABottom, ARight, ATop: Double; const AURI: AnsiString)	Lien vers une adresse web.
AddFreeText(APageIndex; ALeft, ABottom, ARight, ATop: Double; const AText: WideString)	Zone de texte (SetFont, SetTextColor, Alignment).
AddHighlight / AddUnderline / AddStrikeOut(APageIndex; const AQuadPoints: array of Double)	Annotation de balisage de texte. Chaque groupe de 8 nombres forme un quadrilatère.
AddStamp(APageIndex; ALeft, ABottom, ARight, ATop: Double; const AStampName: AnsiString)	Tampon, comme Approved ou Draft.
AddLine(APageIndex; AX1, AY1, AX2, AY2: Double)	Ligne avec StartStyle et EndStyle (par exemple OpenArrow).
AddInk(APageIndex; const APoints: array of Double)	Dessin à main levée à partir de paires X, Y.
AddSquare / AddCircle(APageIndex; ALeft, ABottom, ARight, ATop: Double)	Rectangle ou ellipse (TsgcPDFShapeAnnotation, avec SetInteriorColor).
AddReply(AParent: TsgcPDFAnnotation; const AText: WideString)	Annotation de texte qui répond à AParent.

Lorsqu'un document est chargé, ses annotations existantes sont lues dans le même gestionnaire sous forme d'objets typés, de sorte que `Count`, `GetAnnotationsForPage` et `Annotations` les répertorient. Elles peuvent être modifiées puis enregistrées de nouveau, voir Lecture des annotations et Modification et enregistrement ci-dessous.

```
procedure CreateAnnotationsPDF(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oNote: TsgcPDFTextAnnotation;
  oHighlight: TsgcPDFHighlightAnnotation;
  oLine: TsgcPDFLineAnnotation;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.AddPage(595, 842).DrawText('Annotated page', 70, 790, sfHelveticaBold, 20);
    // sticky note: page index, then Left, Bottom, Right, Top
    oNote := oPDF.Annotations.AddText(0, 70, 680, 94, 704, 'Please review');
    oNote.SetColor(1, 1, 0);
    oNote.Icon := 'Comment';
    oNote.Author := 'Reviewer';
    // web link
    oPDF.Annotations.AddLinkURI(0, 70, 625, 280, 645, 'https://www.esegece.com');
    // highlight: one quadrilateral, 8 numbers (x1 y1 x2 y2 x3 y3 x4 y4)
    oHighlight := oPDF.Annotations.AddHighlight(0,
```

```
      [70, 786, 240, 786, 70, 806, 240, 806]);
    oHighlight.SetColor(1, 1, 0);
    oHighlight.Opacity := 0.5;
    oPDF.Annotations.AddStamp(0, 350, 420, 530, 470, 'Approved');
    oLine := oPDF.Annotations.AddLine(0, 70, 350, 300, 350);
    oLine.SetColor(1, 0, 0);
    oLine.EndStyle := 'OpenArrow';
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Formes

`AddSquare` et `AddCircle` ajoutent un rectangle ou une ellipse inscrite dans le rectangle donné, et renvoient une **TsgcPDFShapeAnnotation**. SetColor définit la couleur de la bordure, SetInteriorColor la couleur de remplissage (il n'y a pas de remplissage si elle n'est pas appelée), et BorderWidth la largeur de la bordure en points. Les formes sont enregistrées avec un flux d'apparence, de sorte que chaque lecteur les dessine de la même manière.

```
procedure CreateShapesPDF(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oBox, oCircle: TsgcPDFShapeAnnotation;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.AddPage(595, 842);
    // rectangle: page index, then Left, Bottom, Right, Top
    oBox := oPDF.Annotations.AddSquare(0, 70, 600, 270, 700);
    oBox.SetColor(0, 0, 1);
    oBox.SetInteriorColor(0.85, 0.9, 1);
    oBox.BorderWidth := 2;
    oBox.Contents := 'Check this block';
    // ellipse inscribed in the rectangle
    oCircle := oPDF.Annotations.AddCircle(0, 320, 600, 420, 700);
    oCircle.SetColor(1, 0, 0);
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Lecture des annotations

LoadFromFile et LoadFromStream lisent les annotations de chaque page dans `Annotations`. Chacune est créée comme la classe de son type, avec ses propriétés renseignées à partir du fichier, de sorte que le test `is` habituel ou un transtypage en donne l'accès :

Sous-type dans le fichier	Classe et propriétés lues
Text	TsgcPDFTextAnnotation : Icon, Open.
Link	TsgcPDFLinkAnnotation : DestPage, ou l'adresse web.
FreeText	TsgcPDFFreeTextAnnotation : police, couleur du texte, Alignment.
Highlight, Underline, StrikeOut	TsgcPDFHighlightAnnotation (AnnotType indique laquelle) : QuadCount, QuadPoint.
Stamp	TsgcPDFStampAnnotation : StampName.

Sous-type dans le fichier	Classe et propriétés lues
Line	TsgcPDFLineAnnotation : X1, Y1, X2, Y2, StartStyle, EndStyle.
Ink	TsgcPDFInkAnnotation : PathCount, PathPointCount, GetPathPoint.
Square, Circle	TsgcPDFShapeAnnotation : HasInteriorColor, InteriorR, InteriorG, InteriorB.
Popup, FileAttachment, Widget et autres	TsgcPDFAnnotation, la classe de base.

Chaque classe d'annotation possède ces membres, pour les annotations chargées depuis un fichier comme pour les nouvelles :

Membre	Description
Author, Contents: WideString	Auteur (/T) et texte (/Contents). Le texte enregistré en UTF-16, comme l'écrivent Acrobat et d'autres lecteurs pour les commentaires accentués ou non latins, est décodé en Unicode.
ModDate: TDateTime	Date de modification (/M) en heure locale, 0 s'il n'y en a pas.
Name: AnsiString	Nom de l'annotation (/NM).
InReplyTo: TsgcPDFAnnotation	L'annotation à laquelle celle-ci répond, ou avec laquelle elle est groupée (/IRT). nil pour une annotation de premier niveau.
ReplyType: AnsiString	Vide ou R pour une réponse, Group pour un membre d'un groupe (/RT).
IsReply: Boolean	True lorsque InReplyTo est défini et que ReplyType n'est pas Group.
BorderWidth: Double	Largeur de la bordure ou de la ligne, en points.
HasColor, ColorR, ColorG, ColorB	La couleur (/C), telle que lue dans le fichier ou définie avec SetColor. Les composantes vont de 0 à 1.
Modified: Boolean	True lorsqu'une annotation chargée a changé depuis le chargement du document, ou depuis le dernier ResetModified. Un indicateur pour l'interface utilisateur. Toujours False pour une annotation créée dans le code.
ChangedSinceLoad: Boolean	True lorsqu'une annotation chargée diffère de l'état qu'elle avait au chargement du document, de sorte que le prochain enregistrement l'écrit de nouveau. ResetModified ne le modifie pas. Toujours False pour une annotation créée dans le code.
MoveBy(ADX, ADY: Double)	Déplace l'annotation de ADX, ADY points : son Rect ainsi que les points de ligne, les points de quadrilatère ou les tracés à main levée.
SourceObjectNumber: Integer	Numéro d'objet d'une annotation chargée dans le fichier, 0 pour une annotation créée dans le code.

La géométrie des lignes, du balisage de texte et du tracé à main levée se lit avec X1, Y1, X2 et Y2 (TsgcPDFLineAnnotation), QuadCount et QuadPoint(AIndex) avec AIndex de 0 à QuadCount * 8 - 1 (TsgcPDFHighlightAnnotation), ainsi que PathCount, PathPointCount(APath) et GetPathPoint(APath, AIndex, AX, AY) (TsgcPDFInkAnnotation).

```
procedure ListAnnotations(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oAnnot: TsgcPDFAnnotation;
  oReplies: TList;
  i, j: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.LoadFromFile(AFileName);
    for i := 0 to oPDF.Annotations.Count - 1 do
    begin
      oAnnot := TsgcPDFAnnotation(oPDF.Annotations.Annotations[i]);
      // replies are listed under their parent, popups have no content
      if oAnnot.IsReply or (oAnnot.AnnotType = atPopup) then
        Continue;
      Write(Format('Page %d: %s', [oAnnot.PageIndex + 1, oAnnot.ClassName]));
      if oAnnot.Author <> '' then
        Write(', by ', oAnnot.Author);
      if oAnnot.ModDate <> 0 then
        Write(', ', DateTimeToStr(oAnnot.ModDate));
      WriteLn;
      if oAnnot.Contents <> '' then
        WriteLn('  ', oAnnot.Contents);
      if oAnnot is TsgcPDFStampAnnotation then
        WriteLn('  stamp: ', TsgcPDFStampAnnotation(oAnnot).StampName);
      oReplies := oPDF.Annotations.GetReplies(oAnnot);
      try
        for j := 0 to oReplies.Count - 1 do
          WriteLn('  reply from ', TsgcPDFAnnotation(oReplies[j]).Author, ': ',
            TsgcPDFAnnotation(oReplies[j]).Contents);
      finally
        oReplies.Free;
      end;
    end;
  finally
    oPDF.Free;
  end;
end;

procedure WriteGeometry(AAnnot: TsgcPDFAnnotation);
var
  oLine: TsgcPDFLineAnnotation;
  oMarkup: TsgcPDFHighlightAnnotation;
  oInk: TsgcPDFInkAnnotation;
  i: Integer;
  vX, vY: Double;
begin
  if AAnnot is TsgcPDFLineAnnotation then
  begin
    oLine := TsgcPDFLineAnnotation(AAnnot);
    WriteLn(Format('Line from %.1f, %.1f to %.1f, %.1f',
      [oLine.X1, oLine.Y1, oLine.X2, oLine.Y2]));
  end
  else if AAnnot is TsgcPDFHighlightAnnotation then
  begin
    // highlight, underline or strike-out: 8 numbers per quadrilateral
    oMarkup := TsgcPDFHighlightAnnotation(AAnnot);
    for i := 0 to oMarkup.QuadCount - 1 do
      WriteLn(Format('Quad %d: first corner %.1f, %.1f',
        [i, oMarkup.QuadPoint(i * 8), oMarkup.QuadPoint(i * 8 + 1)]));
  end
  else if AAnnot is TsgcPDFInkAnnotation then
  begin
    oInk := TsgcPDFInkAnnotation(AAnnot);
    for i := 0 to oInk.PathCount - 1 do
    begin
      oInk.GetPathPoint(i, 0, vX, vY);
```

```
      WriteLn(Format('Path %d: %d points, starts at %.1f, %.1f',
        [i, oInk.PathPointCount(i), vX, vY]));
    end;
  end;
  if AAnnot.HasColor then
    WriteLn(Format('Color %.2f %.2f %.2f, border %.1f',
      [AAnnot.ColorR, AAnnot.ColorG, AAnnot.ColorB, AAnnot.BorderWidth]));
end;
```

Réponses

Une réponse est une annotation dont `InReplyTo` pointe vers une autre annotation (l'entrée /IRT du fichier), comme les réponses à un commentaire dans Acrobat. LoadFromFile relie chaque réponse à son parent, y compris lorsque le parent se trouve sur une autre page. Le gestionnaire possède ces membres pour les réponses et pour la liste elle-même :

Membre	Description
GetAnnotationsForPage(APageIndex: Integer): TList	Les annotations d'une page, dans l'ordre de la liste. Libérez la liste.
GetReplies(AAnnotation: TsgcPDFAnnotation): TList	Les réponses directes d'une annotation, dans l'ordre de la liste. Libérez la liste.
AddReply(AParent: TsgcPDFAnnotation; const AText: WideString): TsgcPDFTextAnnotation	Ajoute une annotation de texte qui répond à AParent, sur sa page et avec son Rect.
Remove(AAnnotation: TsgcPDFAnnotation)	Supprime et libère une annotation, toutes les annotations qui lui répondent ou qui sont groupées avec elle (récursivement) et son popup.
LinkReplies	Résout le /IRT des annotations chargées vers l'annotation chargée portant ce numéro d'objet. LoadFromFile l'appelle, n'appelez-le vous-même qu'après LoadFromPage.
Modified: Boolean	True lorsqu'une annotation a été ajoutée ou supprimée, ou qu'une annotation chargée a changé, depuis le chargement du document ou depuis le dernier ResetModified. L'indicateur « modifications non enregistrées » pour l'interface utilisateur.
ChangedSinceLoad: Boolean	True lorsqu'une annotation a été ajoutée ou supprimée, ou qu'une annotation chargée diffère du fichier, depuis le chargement du document. ResetModified ne le modifie pas.
ResetModified	Efface Modified, par exemple après un enregistrement. Les modifications sont conservées, et le prochain enregistrement les écrit toujours. Voir Suivi des modifications.
MarkLoaded	Prend l'état actuel comme état du fichier et efface Modified et ChangedSinceLoad. Le document l'appelle après le chargement. Voir Suivi des modifications.

Les réponses sont écrites avec /IRT pointant vers leur parent par SaveToFile et par SaveIncremental, de sorte qu'elles sont relues comme des réponses par sgcPDF et par d'autres lecteurs. AddReply ne définit ni Author ni ModDate, définissez-les vous-même.

Modification et enregistrement

Les propriétés d'une annotation chargée peuvent être modifiées comme celles d'une nouvelle annotation : Contents, Author, Rect, la couleur, Flags, ModDate, etc., ou MoveBy. SaveToFile et SaveIncremental écrivent les modifications.

- SaveIncremental écrit de nouveau une annotation modifiée sous son propre numéro d'objet, après les octets d'origine du fichier. Les nouvelles annotations et les réponses sont ajoutées à leur page dans la même révision.
- Lorsque ModDate n'a pas été modifié par votre code, il est défini à l'heure actuelle.
- Le flux d'apparence du fichier est conservé lorsque seuls Contents, Author, Name, ModDate, Rect, Flags ou les entrées de réponse changent. Pour les autres modifications, les tampons et les formes reçoivent une nouvelle apparence, et les autres types perdent la leur afin que les lecteurs les redessinent à partir de leurs propriétés.
- Remove supprime une annotation avec ses réponses et son popup, et le prochain enregistrement écrit la page sans eux.

Comme SaveIncremental ne modifie pas les octets d'origine, la plage d'octets de chaque signature existante reste la même et la valeur de la signature reste valide. L'acceptation de la nouvelle révision par un validateur dépend des permissions de la signature. Une signature de certification avec la permission DocMDP P=3, ou uniquement des signatures d'approbation, autorise l'ajout et la modification de commentaires. Avec P=1 ou P=2, le validateur signale une modification non autorisée. Un enregistrement complet avec SaveToFile réécrit tout le fichier et invalide les signatures existantes, utilisez-le donc uniquement pour des documents non signés. Voir Enregistrement incrémental et Signatures numériques.

```
procedure ReviewNotes(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oList: TList;
  oAnnot: TsgcPDFAnnotation;
  oReply: TsgcPDFTextAnnotation;
  i: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.LoadFromFile(AFileName);
    oList := oPDF.Annotations.GetAnnotationsForPage(0);
    try
      for i := 0 to oList.Count - 1 do
      begin
        oAnnot := TsgcPDFAnnotation(oList[i]);
        if (oAnnot is TsgcPDFTextAnnotation) and not oAnnot.IsReply then
        begin
          // change a loaded note
          oAnnot.Contents := oAnnot.Contents + ' (checked)';
          // and answer it
          oReply := oPDF.Annotations.AddReply(oAnnot, 'Fixed in version 2');
          oReply.Author := 'Reviewer';
          oReply.ModDate := Now;
          Break;
        end;
      end;
    finally
      oList.Free;
    end;
    // appends the changes after the original bytes of the file
    if oPDF.Annotations.Modified then
      oPDF.SaveIncremental(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Suivi des modifications

Deux indicateurs indiquent ce qui a changé, et chacun a son propre rôle.

- `Modified` est l'indicateur « modifications non enregistrées » de l'interface utilisateur. `Annotations.Modified` vaut True lorsqu'une annotation a été ajoutée ou supprimée, ou qu'une annotation chargée a changé, depuis le chargement du document ou depuis le dernier ResetModified. La propriété `Modified` de chaque annotation indique quelles annotations chargées ont changé. Utilisez-les pour activer un bouton Enregistrer, ou pour interroger l'utilisateur avant la fermeture.
- `ChangedSinceLoad` compare avec le fichier. `Annotations.ChangedSinceLoad` vaut True lorsqu'une annotation a été ajoutée ou supprimée, ou qu'une annotation chargée diffère de l'état qu'elle avait au chargement du document. SaveToFile et SaveIncremental utilisent la fonction `ChangedSinceLoad` de chaque annotation pour décider quelles annotations chargées ils écrivent avec leurs modifications.

`ResetModified` efface seulement Modified. Appelez-le après un enregistrement, afin que l'interface utilisateur indique qu'il ne reste rien à enregistrer. Il ne modifie pas ChangedSinceLoad, il ne perd donc jamais une modification : le prochain SaveToFile ou SaveIncremental écrit toujours chaque modification faite depuis le chargement du document, avec les plus récentes. Il n'est pas nécessaire de recharger le fichier enregistré, et l'exemple continue de fonctionner sur le même document.
`MarkLoaded` prend l'état actuel des annotations comme état du fichier et efface les deux indicateurs. LoadFromFile et LoadFromStream l'appellent après la lecture des annotations, les applications en ont donc rarement besoin. Pour effacer l'indicateur « modifications non enregistrées » après un enregistrement, appelez plutôt ResetModified.

```
procedure MoveNoteAndSave(ADocument: TsgcPDFDocument; const AFileName: string);
var
  oAnnot: TsgcPDFAnnotation;
  i: Integer;
begin
  for i := 0 to ADocument.Annotations.Count - 1 do
  begin
    oAnnot := TsgcPDFAnnotation(ADocument.Annotations.Annotations[i]);
    if oAnnot is TsgcPDFTextAnnotation then
    begin
      // 20 points right and 20 points down
      oAnnot.MoveBy(20, -20);
      WriteLn('Note changed: ', oAnnot.Modified);
      Break;
    end;
  end;
  if ADocument.Annotations.Modified then
  begin
    ADocument.SaveIncremental(AFileName);
    // saved: clear the "unsaved changes" flag of the user interface
    ADocument.Annotations.ResetModified;
  end;
  WriteLn('Unsaved changes: ', ADocument.Annotations.Modified);
  // still True: the next save writes the note again, with the later changes
  WriteLn('Changed since load: ', ADocument.Annotations.ChangedSinceLoad);
end;
```

Rendu des annotations modifiées

Pour un usage avancé, `TsgcPDFDocument.BuildPageAnnotations(APageIndex)` renvoie les annotations d'une page telles qu'elles sont à présent, modifications comprises, sous forme de TsgcPDFArray pour la propriété Annotations d'un moteur de rendu. L'appelant libère le tableau. Les lecteurs l'utilisent pour dessiner les pages dont les annotations ont été modifiées, la plupart des applications ne l'appellent donc jamais.

Voir aussi

- TsgcPDFDocument
- Signets
- Enregistrement incrémental
- TsgcPDFViewerControl
- TsgcPDFViewerFMX
- TsgcPDFCommentsView

Signets

Les signets (le plan du document) sont gérés par `TsgcPDFDocument.Bookmarks` (TsgcPDFBookmarks, unité sgcPDF_Bookmarks). `Add` crée une entrée de premier niveau et `TsgcPDFBookmarkItem.AddChild` une entrée imbriquée. Chaque élément possède Title, PageIndex, Top, Left, Zoom, ColorR/G/B, Bold, Italic et Open.

```
procedure CreateBookmarksPDF(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oChapter: TsgcPDFBookmarkItem;
  i: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    for i := 0 to 4 do
      oPDF.AddPage(595, 842).DrawText(AnsiString('Page ' + IntToStr(i + 1)),
        50, 780, sfHelvetica, 14);
    oPDF.Bookmarks.Add('Cover', 0);
    oChapter := oPDF.Bookmarks.Add('Chapter 1', 1);
    oChapter.Bold := True;
    oChapter.Open := True;
    oChapter.AddChild('Section 1.1', 2);
    oChapter.AddChild('Section 1.2', 3);
    oPDF.Bookmarks.Add('Appendix', 4);
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Lecture des signets

Les signets d'un document chargé sont disponibles via la même propriété.

```
procedure PrintBookmarks(AItem: TsgcPDFBookmarkItem; const AIndent: string);
var
  i: Integer;
begin
  WriteLn(AIndent, string(AItem.Title), ' -> page ', AItem.PageIndex + 1);
  for i := 0 to AItem.ChildCount - 1 do
    PrintBookmarks(AItem.GetChild(i), AIndent + '  ');
end;

procedure ListBookmarks(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  i: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.LoadFromFile(AFileName);
    for i := 0 to oPDF.Bookmarks.Count - 1 do
      PrintBookmarks(oPDF.Bookmarks[i], '');
  finally
    oPDF.Free;
  end;
end;
```

Voir aussi

- TsgcPDFDocument

- Annotations

Calques

Les calques (groupes de contenu optionnel) permettent au lecteur d'afficher ou de masquer des parties d'une page. Ils sont gérés par `TsgcPDFDocument.Layers` (TsgcPDFLayerManager, unité sgcPDF_Layers).

- `AddLayer(AName, ADefaultState)` crée un calque initialement activé (lsOn) ou désactivé (lsOff). TsgcPDFLayer possède aussi Intent, Visible, Printable et Locked.
- Le contenu appartient à un calque lorsqu'il est écrit entre `BeginLayerMarkedContent(ALayer)` et `EndLayerMarkedContent`. Les deux renvoient le texte de l'opérateur, que vous ajoutez à la page avec `Content.AddOperator`.
- `AddGroup` regroupe des calques dans le panneau de calques du lecteur. `FindLayer`, `GetLayer`, `LayerCount` et `RemoveLayer` gèrent la liste, qui est aussi remplie lorsqu'un document est chargé.

```
procedure CreateLayersPDF(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oPage: TsgcPDFPage;
  oNotes: TsgcPDFLayer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.Layers.AddLayer('Background', lsOn);
    oNotes := oPDF.Layers.AddLayer('Notes', lsOff);
    oPage := oPDF.AddPage(595, 842);
    oPage.DrawText('Always visible', 50, 780, sfHelvetica, 14);
    // wrap the drawing operators that belong to the layer in marked content
    oPage.Content.AddOperator(oPDF.Layers.BeginLayerMarkedContent(oNotes));
    oPage.DrawText('Only visible when the Notes layer is on', 50, 740,
      sfHelvetica, 14);
    oPage.Content.AddOperator(oPDF.Layers.EndLayerMarkedContent);
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Voir aussi

- TsgcPDFDocument
- Graphisme

Formulaires

Les formulaires interactifs (AcroForms) sont gérés par `TsgcPDFDocument.AcroForm` (TsgcPDFAcroForm, unité sgcPDF_Forms).

Création de champs

Méthode	Champ
AddTextField(const AName: AnsiString; APageIndex: Integer; ALeft, ABottom, ARight, ATop: Double)	Zone de texte.
AddCheckBox(...)	Case à cocher.
AddRadioGroup(const AName: AnsiString) and AddRadioButton(AGroup; APageIndex; ALeft, ABottom, ARight, ATop; const AExportValue)	Boutons radio.
AddComboBox(...; const AOptions: array of WideString)	Liste déroulante.
AddListBox(...; const AOptions: array of WideString)	Liste.
AddButton(...; const ACaption: WideString)	Bouton poussoir.
AddSignatureField(...)	Champ de signature vide.

Chaque méthode renvoie un TsgcPDFFormField avec Value, DefaultValue, ReadOnly, Required, MaxLen, FontName, FontSize et Alignment. Définissez `NeedAppearances` sur True pour que les lecteurs régénèrent les apparences des champs.

```
procedure CreateFormPDF(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oPage: TsgcPDFPage;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPage := oPDF.AddPage(595, 842);
    oPage.DrawText('Name:', 50, 737, sfHelvetica, 12);
    oPage.DrawText('Country:', 50, 697, sfHelvetica, 12);
    oPage.DrawText('Subscribe:', 50, 657, sfHelvetica, 12);
    oPDF.AcroForm.AddTextField('name', 0, 160, 730, 400, 755).FontSize := 12;
    oPDF.AcroForm.AddComboBox('country', 0, 160, 690, 400, 715,
      ['Germany', 'Spain', 'United States']);
    oPDF.AcroForm.AddCheckBox('subscribe', 0, 160, 650, 178, 668);
    oPDF.AcroForm.NeedAppearances := True;
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Remplissage des champs

Lorsqu'un document est chargé, ses champs sont lus dans AcroForm. Utilisez FillField, SetCheckBox, GetFieldValue, FindField, GetField et FieldCount.

```
procedure FillFormPDF(const AInput, AOutput: string);
var
  oPDF: TsgcPDFDocument;
  i: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.LoadFromFile(AInput);
    for i := 0 to oPDF.AcroForm.FieldCount - 1 do
      WriteLn('Field: ', string(oPDF.AcroForm.GetField(i).FullName));
    oPDF.AcroForm.FillField('name', 'John Smith');
    oPDF.AcroForm.FillField('country', 'Spain');
    oPDF.AcroForm.SetCheckBox('subscribe', True);
    oPDF.SaveToFile(AOutput);
  finally
    oPDF.Free;
  end;
end;
```

Aplatissement

L'aplatissement dessine les valeurs actuelles des champs comme du contenu de page normal et supprime les champs, de sorte que les valeurs ne peuvent plus être modifiées. Utilisez `TsgcPDFAcroForm.FlattenFields` ou `TsgcPDFEditor.FlattenForm`.

```
procedure FillAndFlatten(const AInput, AOutput: string);
var
  oEditor: TsgcPDFEditor;
begin
  oEditor := TsgcPDFEditor.Create;
  try
    oEditor.LoadFromFile(AInput);
    oEditor.FillFormField('name', 'Jane Doe');
    // the field values become ordinary page content, the fields are removed
    oEditor.FlattenForm;
    oEditor.SaveToFile(AOutput);
  finally
    oEditor.Free;
  end;
end;
```

Voir aussi

- TsgcPDFDocument
- TsgcPDFEditor

Chiffrement

Appelez `SetEncryption` avant l'enregistrement pour chiffrer un document avec le gestionnaire de sécurité standard. **TsgcPDFEncryptionMethod** (unité sgcPDF_Encryption) sélectionne l'algorithme :

Valeur	Algorithme
emRC4_40	RC4 40 bits (PDF 1.1, obsolète)
emRC4_128	RC4 128 bits (PDF 1.4)
emAES_128	AES-128 (PDF 1.6)
emAES_256	AES-256 (PDF 2.0, recommandé)

Le mot de passe utilisateur est nécessaire pour ouvrir le fichier. Le mot de passe propriétaire accorde un accès complet. SetEncryption sur TsgcPDFDocument et TsgcPDFEditor accorde toutes les permissions (impression, modification, copie, annotation, remplissage de formulaires, extraction, assemblage et impression haute qualité).

```
procedure CreateEncryptedPDF(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.AddPage(595, 842).DrawText('Secret', 50, 780, sfHelveticaBold, 24);
    // user password opens the file, owner password grants full rights
    oPDF.SetEncryption(emAES_256, 'user123', 'owner456');
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Ouverture de fichiers chiffrés

Appelez `SetPassword` avant `LoadFromFile`. Le mot de passe utilisateur ou le mot de passe propriétaire permet d'ouvrir le fichier.

```
procedure OpenEncryptedPDF(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    // set the password before loading
    oPDF.SetPassword('user123');
    oPDF.LoadFromFile(AFileName);
    WriteLn(string(oPDF.ExtractTextW(0)));
  finally
    oPDF.Free;
  end;
end;
```

Accès de bas niveau

TsgcPDFEncryptionHandler implémente le gestionnaire de sécurité lui-même : `SetupEncryption` avec un ensemble TsgcPDFPermissions, `Authenticate`, et les routines de chiffrement et déchiffrement pour les chaînes et les

flux. L'unité expose aussi sgcPDFMD5, sgcPDFSHA256, sgcPDFSHA384, sgcPDFSHA512, sgcPDFRC4, sgcPDFAESEncryptCBC et sgcPDFAESDecryptCBC. Les erreurs de cette unité déclenchent **EsgcPDFEncryptionError**.

Voir aussi

- TsgcPDFDocument
- Signatures numériques

Signatures numériques

TsgcPDFSignatureHandler signe un PDF avec un certificat disposant d'une clé privée, chargé depuis un fichier PFX, depuis le magasin de certificats Windows par nom de sujet, ou par empreinte. La signature est une signature PKCS#7 détachée (adbe.pkcs7.detached) ajoutée sous forme de mise à jour incrémentale, de sorte que les octets du fichier d'origine restent inchangés au début du fichier signé.

Signature

```
procedure SignPDF(const AInput, AOutput: string);
var
  oSigner: TsgcPDFSignatureHandler;
begin
  oSigner := TsgcPDFSignatureHandler.Create;
  try
    oSigner.LoadCertificateFromPFX('signer.pfx', 'test123');
    WriteLn('Signing as ', string(oSigner.GetCertificateInfo.SubjectName));
    oSigner.Reason := 'Document approval';
    oSigner.Location := 'Barcelona';
    oSigner.HashAlgorithm := shaSHA256;
    // visible signature field on page 1: Left, Bottom, Right, Top
    oSigner.SetField('Signature1', 0, 50, 50, 250, 100);
    oSigner.SignDocument(AInput, AOutput);
  finally
    oSigner.Free;
  end;
end;
```

Appelez `SetField` pour un rectangle de signature visible, ou `SetInvisible` pour une signature sans apparence. `HashAlgorithm` sélectionne SHA-1, SHA-256 (par défaut), SHA-384 ou SHA-512.

Vérification

TsgcPDFSignatureVerifier indique, pour chaque signature, si elle est valide, si le document a été modifié après la signature, et si le certificat est fiable pour Windows.

```
procedure VerifyPDF(const AFileName: string);
var
  oVerifier: TsgcPDFSignatureVerifier;
  oCert: TsgcPDFCertificateInfo;
  i: Integer;
begin
  oVerifier := TsgcPDFSignatureVerifier.Create;
  try
    oVerifier.LoadDocument(AFileName);
    for i := 0 to oVerifier.GetSignatureCount - 1 do
    begin
      oCert := oVerifier.GetSignatureInfo(i);
      WriteLn('Signer:   ', string(oCert.SubjectName));
      WriteLn('Valid:    ', oVerifier.VerifySignature(i));
      WriteLn('Modified: ', oVerifier.IsDocumentModified(i));
      WriteLn('Trusted:  ', oVerifier.IsSignatureTrusted(i));
    end;
  finally
    oVerifier.Free;
  end;
end;
```

Signer avec l'éditeur

`TsgcPDFEditor.SignDocument(AOutputFile, ACertSubjectName, AReason, ALocation)` enregistre et signe le document modifié avec un certificat du magasin Windows en un seul appel.

Voir aussi

- Chiffrement
- PDF/A

PDF/A

PDF/A est la norme ISO pour l'archivage à long terme. sgcPDF écrit les niveaux de conformité **b** (basic) de PDF/A-1, PDF/A-2 et PDF/A-3, sélectionnés avec **TsgcPDFALevel** (unité sgcPDF_PDFA) : paA1b, paA2b et paA3b. Lorsque `SetPDFALevel` est défini, le document est enregistré avec des métadonnées XMP qui déclarent le niveau de conformité et reprennent les champs Info, une intention de sortie sRGB et un identifiant de document. Votre contenu doit aussi respecter les règles PDF/A :

- Intégrez chaque police : dessinez le texte avec des polices TrueType (AddTrueTypeFont et DrawTextTT), pas avec les polices standard.
- Ne chiffrez pas le document.
- PDF/A-1b n'autorise pas la transparence, évitez donc SetOpacity et le dessin transparent à ce niveau.

```
procedure CreatePDFA(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  vFont: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.SetPDFALevel(paA2b);
    oPDF.Info.Title := 'Archived invoice';
    oPDF.Info.Author := 'eSeGeCe';
    // PDF/A requires embedded fonts, so use a TrueType font
    vFont := oPDF.AddTrueTypeFont('Arial', False, False);
    oPDF.AddPage(595, 842).DrawTextTT('PDF/A-2b document', 50, 780, vFont, 20);
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Validation

TsgcPDFAValidator vérifie un document chargé par rapport à un niveau : métadonnées XMP, intégration des polices, espaces colorimétriques, transparence, chiffrement, annotations, actions et structure du fichier. `Errors` répertorie les problèmes trouvés. C'est une vérification rapide intégrée, pas un remplacement d'un validateur de conformité complet tel que veraPDF.

```
procedure ValidatePDFA(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oValidator: TsgcPDFAValidator;
begin
  oPDF := TsgcPDFDocument.Create;
  oValidator := TsgcPDFAValidator.Create(paA2b);
  try
    oPDF.LoadFromFile(AFileName);
    if oValidator.Validate(oPDF.Parser) then
      WriteLn('No PDF/A-2b problems found')
    else
      WriteLn(oValidator.Errors.Text);
  finally
    oValidator.Free;
    oPDF.Free;
  end;
end;
```

`TsgcPDFEditor.ConvertToPDFA(ALevel)` définit le niveau sur un document ouvert avec l'éditeur.

Voir aussi

- TsgcPDFDocument
- Texte et polices

Optimisation des fichiers PDF

TsgcPDFOptimizer réécrit un PDF pour réduire sa taille. Par défaut, il compresse les flux, fusionne les objets dupliqués et supprime les objets inutilisés. Le sous-échantillonnage des images et la suppression des métadonnées, signets, annotations, JavaScript et vignettes sont activables via `Options`.

```
procedure OptimizePDF(const AInput, AOutput: string);
var
  oOptimizer: TsgcPDFOptimizer;
  oOptions: TsgcPDFOptimizeOptions;
begin
  oOptimizer := TsgcPDFOptimizer.Create;
  try
    oOptions := sgcPDFDefaultOptimizeOptions;
    oOptions.DownsampleImages := True;
    oOptions.MaxImageDPI := 150;
    oOptions.RemoveMetadata := True;
    oOptimizer.Options := oOptions;
    oOptimizer.Optimize(AInput, AOutput);
    WriteLn('Saved ', oOptimizer.GetSizeReduction, ' bytes');
    WriteLn(oOptimizer.GetLog.Text);
  finally
    oOptimizer.Free;
  end;
end;
```

Un fichier déjà compact, comme un fichier écrit par sgcPDF lui-même, peut ne pas devenir plus petit. GetSizeReduction renvoie alors 0 et le journal indique les tailles d'entrée et de sortie.

Voir aussi

- Images

Fusionner, scinder et extraire des pages

Fonctions utilitaires en un appel

L'unité sgcPDF_Editor propose des fonctions qui agissent directement sur les fichiers :

- `sgcPDFMergeFiles(AInputFiles, AOutputFile)` fusionne des fichiers dans l'ordre indiqué.
- `sgcPDFExtractPages(AInputFile, AOutputFile, AFromPage, AToPage)` copie une plage de pages incluse, indexée à partir de zéro, vers un nouveau fichier (scission).
- `sgcPDFGetPageCount(AFileName)` renvoie le nombre de pages.

```
procedure MergeAndSplit;
begin
  // merge several files into one
  sgcPDFMergeFiles(['out\hello.pdf', 'out\bookmarks.pdf', 'out\tables.pdf'],
    'out\merged.pdf');
  WriteLn('Merged pages: ', sgcPDFGetPageCount('out\merged.pdf'));
  // extract pages 2 to 4 (zero based indexes 1 to 3) into a new file
  sgcPDFExtractPages('out\merged.pdf', 'out\pages_2_4.pdf', 1, 3);
end;
```

Avec l'éditeur

TsgcPDFEditor combine des opérations sur les pages d'un document ouvert : MergeFrom, ExtractPages, DeletePage, RotatePage et AddBlankPage.

```
procedure ReorganizePages(const AInput, AOutput: string);
var
  oEditor: TsgcPDFEditor;
begin
  oEditor := TsgcPDFEditor.Create;
  try
    oEditor.LoadFromFile(AInput);
    oEditor.MergeFrom('out\hello.pdf');
    oEditor.RotatePage(0, 90);
    oEditor.DeletePage(1);
    oEditor.AddBlankPage(595, 842);
    oEditor.SaveToFile(AOutput);
  finally
    oEditor.Free;
  end;
end;
```

Avec un TsgcPDFDocument, utilisez AppendDocument, InsertPage et DeletePage.

Voir aussi

- TsgcPDFEditor

Extraction de texte

sgcPDF extrait le texte des flux de contenu des pages. Il décode les encodages standard (WinAnsi, MacRoman, Standard), les Differences de police, les tables ToUnicode et les polices CID, de sorte que le texte écrit avec des polices TrueType ou CJK intégrées est renvoyé en Unicode. Les pages numérisées qui ne contiennent que des images n'ont pas de texte à extraire.

Texte brut

`TsgcPDFDocument.ExtractTextW` renvoie le texte d'une page et `FindText` le recherche.

```
procedure PrintAllText(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  i: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.LoadFromFile(AFileName);
    for i := 0 to oPDF.PageCount - 1 do
    begin
      WriteLn('--- page ', i + 1, ' ---');
      WriteLn(string(oPDF.ExtractTextW(i)));
    end;
    if oPDF.FindText(0, 'Hello') then
      WriteLn('Page 1 contains "Hello"');
  finally
    oPDF.Free;
  end;
end;
```

Texte avec positions

TsgcPDFTextExtractor renvoie chaque fragment de texte avec ses coordonnées, sa police et sa taille.

```
procedure PrintTextPositions(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oExtractor: TsgcPDFTextExtractor;
  oFragment: TsgcPDFTextFragment;
  i: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.LoadFromFile(AFileName);
    oExtractor := TsgcPDFTextExtractor.Create(oPDF.Parser,
      oPDF.Pages[0].PageDict);
    try
      oExtractor.Extract;
      for i := 0 to oExtractor.GetFragmentCount - 1 do
      begin
        oFragment := oExtractor.GetFragment(i);
        WriteLn(Format('%.1f, %.1f  %s  (%s %.1f pt)', [oFragment.X, oFragment.Y,
          string(oFragment.Text), string(oFragment.FontName), oFragment.FontSize]));
      end;
    finally
      oExtractor.Free;
    end;
  finally
    oPDF.Free;
  end;
```

```
end;
```

Utiliser directement l'analyseur

Les fonctions sgcPDFExtractPageText, sgcPDFExtractAllText et sgcPDFFindText fonctionnent avec un TsgcPDF-Parser. Appelez `Parse` après le chargement.

```
function GetDocumentText(const AFileName: string): WideString;
var
  oParser: TsgcPDFParser;
begin
  oParser := TsgcPDFParser.Create;
  try
    oParser.LoadFromFile(AFileName);
    oParser.Parse;
    Result := sgcPDFExtractAllText(oParser);
  finally
    oParser.Free;
  end;
end;
```

Voir aussi

- TsgcPDFTextExtractor

Rendu et export

Les pages sont rendues avec **TsgcPDFRendererGDI** (unité sgcPDF_Renderer_GDI), qui dessine le contenu de la page (chemins, texte, images, dégradés) dans un contexte de périphérique Windows. Vous l'utilisez rarement directement :

- TsgcPDFExporter enregistre les pages sous forme de fichiers BMP, PNG ou JPEG, ou de vignettes.
- TsgcPDFViewer affiche les pages dans une fenêtre.
- `TsgcPDFRendererGDI.RenderPage(ADC, AParser, APageDict, AWidth, AHeight)` dessine une page dans n'importe quel HDC, par exemple le canevas d'un TBitmap ou une imprimante.

Exporter des pages

```
procedure ExportPages(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oExporter: TsgcPDFExporter;
  oOptions: TsgcPDFExportOptions;
begin
  oPDF := TsgcPDFDocument.Create;
  oExporter := TsgcPDFExporter.Create;
  try
    oPDF.LoadFromFile(AFileName);
    oOptions := sgcPDFDefaultExportOptions;
    oOptions.Format := efPNG;
    oOptions.DPI := 150;
    oExporter.ExportPage(oPDF, 0, 'out\page1.png', oOptions);
    oExporter.ExportThumbnail(oPDF, 0, 'out\thumb1.bmp', 200, 200);
  finally
    oExporter.Free;
    oPDF.Free;
  end;
  // one call helpers that open the file themselves
  sgcPDFPageToImage(AFileName, 'out\page1_96.png', 0, 96);
end;
```

Visualisation

```
procedure ShowViewer(const AFileName: string);
var
  oViewer: TsgcPDFViewer;
begin
  oViewer := TsgcPDFViewer.Create;
  try
    oViewer.ViewMode := vmContinuous;
    // top level window; use CreateWindow(AParent, X, Y, W, H) to embed it
    oViewer.CreateWindow(100, 100, 800, 900, 'My PDF Viewer');
    oViewer.LoadFromFile(AFileName);
    oViewer.ZoomFit;
    oViewer.Show;
    oViewer.RunMessageLoop;
  finally
    oViewer.Free;
  end;
end;
```

Voir aussi

- TsgcPDFExporter
- TsgcPDFViewer

Gestion des erreurs et entrées malveillantes

sgcPDF signale les erreurs en déclenchant des exceptions. Charger un fichier endommagé ou non pris en charge, un index de page incorrect, un certificat manquant ou une image non prise en charge déclenche une exception dont Message décrit le problème (par exemple *PDF parse error: cannot find startxref* ou *sgcPDF: Invalid page index*). Les erreurs de chiffrement déclenchent **EsgcPDFEncryptionError**. Encadrez le chargement des fichiers que vous ne maîtrisez pas dans un try..except.

```
procedure OpenUntrustedPDF(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
begin
  // cap every decoded stream at 64 MB instead of the default 256 MB
  sgcPDFMaxDecodedSize := 64 * 1024 * 1024;
  oPDF := TsgcPDFDocument.Create;
  try
    try
      oPDF.LoadFromFile(AFileName);
      WriteLn('Pages: ', oPDF.PageCount);
    except
      on E: Exception do
        WriteLn('Rejected: ', E.Message);
    end;
  finally
    oPDF.Free;
  end;
end;
```

Limites pour les fichiers non approuvés

Des fichiers PDF peuvent être conçus pour épuiser la mémoire ou la pile, par exemple avec un petit flux compressé qui se décompresse en gigaoctets, ou avec des objets profondément imbriqués. L'analyseur et les décodeurs appliquent des limites et déclenchent une exception lorsqu'une limite est dépassée :

Limite	Valeur	Description
sgcPDFMaxDecodedSize	268435456 (256 Mo)	Taille décodée maximale d'un flux (Flate, LZW, ASCII85, ASCIIHex, RunLength). C'est une variable globale de sgcPDF_Filters, vous pouvez donc la réduire pour des services qui traitent des envois d'utilisateurs, ou l'augmenter pour de très grandes images légitimes.
Imbrication d'objets	512 niveaux	Tableaux et dictionnaires imbriqués dans l'analyseur.
Nombre d'objets	8388607	Numéro d'objet maximal accepté depuis les données de références croisées.
Sections de références croisées	4096	Sections et flux xref chaînés.
Taille d'image	2^{27} pixels, 1 Go de données	Plus grande image décodée.
Imbrication du flux de contenu	64 niveaux	Profondeur d'imbrication acceptée par l'analyseur du flux de contenu.

Limite	Valeur	Description
Arbre de pages	64 niveaux, 1 000 000 de nœuds	Profondeur et taille de l'arbre de pages parcouru par l'extracteur de texte.

Seul sgcPDFMaxDecodedSize peut être modifié. Les autres limites sont fixes. Définissez sgcPDFMaxDecoded-Size une seule fois au démarrage, car elle s'applique à tous les documents du processus.

Voir aussi

- FAQ

Architecture du moteur de rendu

sgcPDF dessine les pages PDF avec un seul interpréteur de contenu et plusieurs moteurs de dessin (backends). L'interpréteur lit le flux de contenu de la page, résout les polices, images, espaces colorimétriques, dégradés et annotations, et transforme chaque coordonnée en pixels de périphérique. Le backend n'a qu'à peindre les chemins, les rasters et les glyphes. La même page a donc le même aspect sur chaque backend.

Classe	Unité	Rôle
TsgcPDFRenderer	sgcPDF_Renderer	L'interpréteur de contenu. Indépendant de la plateforme, en Pascal pur.
TsgcPDFRenderCanvas	sgcPDF_Renderer_Canvas	Backend de dessin abstrait : chemins, découpage, images, rasters, calques et texte système.
TsgcPDFRasterCanvas	sgcPDF_Renderer_Raster	Rasteriseur logiciel avec anticrénelage. Dessine dans une mémoire tampon de pixels BGRA. Fonctionne sur toutes les plateformes.
TsgcPDFGDICanvas	sgcPDF_Renderer_GDI	Backend GDI Windows. Dessine sur n'importe quel contexte de périphérique (écran, bitmap, imprimante).
TsgcPDFFMXCanvas	sgcPDF_Renderer_FMX	Le rasteriseur logiciel avec sortie bitmap FireMonkey et texte système FMX. Delphi XE5 ou plus récent.

Vous n'utilisez normalement pas l'interpréteur directement. Chaque backend possède une classe de moteur de rendu déjà prête qui contient un interpréteur et un canevas :

Renderer	Sortie	Plateformes
TsgcPDFRendererRaster	Fichier ou flux PNG, fichier BMP, pixels BGRA	Windows, Linux, macOS, iOS, Android
TsgcPDFRendererGDI	Contexte de périphérique Windows, fichier BMP	Windows
TsgcPDFRendererFMX	FireMonkey TBitmap	Toutes les plateformes FMX

Propriétés communes

Toutes les classes de moteur de rendu exposent l'interpréteur sous `Renderer` et le backend sous `Canvas`, plus ces propriétés :

Propriété	Description
RenderAnnotations: Boolean	Dessine les annotations et champs de formulaire de la page après le contenu de la page. Par défaut True.
AnnotationMode: TsgcPDFAnnotRenderMode	**armView** (par défaut) ignore les annotations portant l'indicateur NoView, **armPrint** ne dessine que les annotations portant l'indicateur Print. Utilisez armPrint pour l'impression et pour les images qui doivent correspondre à une sortie imprimée. Non disponible sur TsgcPDFRendererFMX, qui utilise toujours armView.

Écrire votre propre backend

Pour dessiner vers une autre cible (une bibliothèque graphique, un convertisseur PDF vers SVG, un enregistreur de tests), dérivez de **TsgcPDFRenderCanvas**, implémentez ses méthodes abstraites et passez-la à `TsgcPDFRenderer.Create(ACanvas)`. Appelez ensuite `RenderPage(AParser, APageDict, AWidth, AHeight)`. L'interpréteur appelle le canevas dans l'espace périphérique : les coordonnées sont des pixels avec Y croissant vers le bas, les couleurs sont des octets RGB et l'alpha va de 0 à 255. Les tampons raster sont en BGRA 32 bits, de haut en bas et non prémultipliés.

Voir aussi

- TsgcPDFRendererRaster
- TsgcPDFRendererGDI
- Fonctionnalités de rendu prises en charge
- Rendu et export

TsgcPDFRendererRaster

TsgcPDFRendererRaster (unité sgcPDF_Renderer_Raster) effectue le rendu des pages PDF en images sans GDI, VCL ni FMX. Il est écrit en Pascal pur, de sorte que le même code fonctionne dans un service Windows, un serveur web Linux ou un outil console sous macOS. Utilisez-le pour créer des aperçus et des vignettes de pages sur un serveur, pour convertir un PDF en PNG, ou pour fournir les pixels d'une page à un code d'OCR ou de traitement d'images.

Méthodes

Méthode	Description
RenderPageToPNG(AParser; APageDict; AWidth, AHeight: Integer; const AFileName: string)	Effectue le rendu d'une page à AWidth x AHeight pixels sur un fond blanc et l'enregistre en PNG.
RenderPageToBitmap(AParser; APageDict; AWidth, AHeight: Integer; const AFileName: string)	Identique, enregistré sous forme de fichier BMP 32 bits.
RenderPageToImage(AParser; APageDict; AWidth, AHeight: Integer)	Effectue le rendu uniquement dans le canevas. Lisez les pixels depuis `Canvas.Bits` ou enregistrez-les avec les méthodes du canevas.
RenderPage(AParser; APageDict; ARenderWidth, ARenderHeight: Integer; AOffsetX: Integer = 0; AOffsetY: Integer = 0)	Dessine sur le contenu actuel du canevas sans le redimensionner ni l'effacer. Dimensionnez d'abord le canevas avec `Canvas.SetSize` .
Render(AContentData: TBytes; AMediaBox: TsgcPDFRect; ...)	Effectue le rendu d'un flux de contenu brut (sans ressources).

AParser est le `TsgcPDFDocument.Parser` d'un document chargé et APageDict est `TsgcPDFDocument.Pages[i].PageDict`. La rotation de la page (/Rotate) est appliquée.

TsgcPDFRasterCanvas

La propriété `Canvas` contient le raster de la page :

Membre	Description
Width, Height	Taille du raster en pixels.
Bits: TBytes	Les pixels : BGRA, de haut en bas, 4 * Width octets par ligne. Le canal alpha vaut toujours 255.
SetSize(AWidth, AHeight)	Redimensionne le raster et le remplit de blanc.
Clear(R, G, B: Byte)	Remplit le raster d'une couleur.
SaveToPNGFile, SaveToPNGStream, SaveToBMPFile	Enregistre le raster.

Exemple : page vers PNG

```
procedure RenderPageToPNG(const APDFFile, APNGFile: string; ADPI: Integer);
var
  oPDF: TsgcPDFDocument;
  oRenderer: TsgcPDFRendererRaster;
  oPage: TsgcPDFPage;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.LoadFromFile(APDFFile);
    oPage := oPDF.Pages[0];
    oRenderer := TsgcPDFRendererRaster.Create;
    try
      // page size in points (1/72 inch) to pixels
      oRenderer.RenderPageToPNG(oPDF.Parser, oPage.PageDict,
        Round(oPage.Width * ADPI / 72), Round(oPage.Height * ADPI / 72),
        APNGFile);
    finally
      oRenderer.Free;
    end;
  finally
    oPDF.Free;
  end;
end;
```

Exemple : toutes les pages, pixels et flux

```
procedure RenderAllPagesToPixels(const APDFFile: string);
var
  oPDF: TsgcPDFDocument;
  oRenderer: TsgcPDFRendererRaster;
  oStream: TFileStream;
  i: Integer;
begin
  // extra folder with TrueType fonts for text in non embedded fonts
  sgcPDFRasterAddFontDirectory('/opt/myapp/fonts');
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.LoadFromFile(APDFFile);
    oRenderer := TsgcPDFRendererRaster.Create;
    try
      // annotations are drawn by default; armPrint draws the printable ones
      oRenderer.AnnotationMode := armPrint;
      for i := 0 to oPDF.PageCount - 1 do
      begin
        oRenderer.RenderPageToImage(oPDF.Parser, oPDF.Pages[i].PageDict,
          800, 1131);
        // Canvas.Bits: BGRA pixels, top-down, 4 * Width bytes per row
        WriteLn(Format('page %d: %d x %d, %d bytes', [i + 1,
          oRenderer.Canvas.Width, oRenderer.Canvas.Height,
          Length(oRenderer.Canvas.Bits)]));
        oStream := TFileStream.Create(Format('out\page%d.png', [i + 1]),
          fmCreate);
        try
          oRenderer.Canvas.SaveToPNGStream(oStream);
        finally
          oStream.Free;
        end;
      end;
    finally
      oRenderer.Free;
    end;
  finally
    oPDF.Free;
```

```
  end;
end;
```

Polices pour le texte non intégré

Le texte dans des polices intégrées (TrueType, OpenType, CFF, Type 1 et Type 3) est dessiné à partir du programme de police du PDF. Le texte dans des polices non intégrées, comme les 14 polices standard, est dessiné avec une police TrueType du système portant le même nom ou un substitut proche. Le moteur de rendu analyse ces dossiers :

- Windows : %WINDIR%\Fonts et %LOCALAPPDATA%\Microsoft\Windows\Fonts.
- Linux : /usr/share/fonts, /usr/local/share/fonts, ~/.fonts et ~/.local/share/fonts.
- macOS : /System/Library/Fonts, /Library/Fonts et ~/Library/Fonts.
- Android : /system/fonts.

Appelez `sgcPDFRasterAddFontDirectory(const ADirectory: string)` avant le rendu pour ajouter un autre dossier (il est analysé récursivement). Sur un serveur Linux minimal ou un conteneur sans polices, installez un paquet de polices (par exemple fonts-dejavu ou fonts-liberation) ou fournissez vos propres polices et enregistrez leur dossier.

Limites

La taille du raster est limitée à CS_CANVAS_MAX_RASTER pixels (67 108 864, par exemple 8192 x 8192) pour protéger le processus contre les allocations énormes. Scindez les sorties plus grandes en tuiles avec `RenderPage` et les paramètres de décalage.

Voir aussi

- Architecture du moteur de rendu
- Fonctionnalités de rendu prises en charge
- Linux
- TsgcPDFRendererGDI

TsgcPDFRendererGDI

TsgcPDFRendererGDI (unité sgcPDF_Renderer_GDI) effectue le rendu des pages PDF sur un contexte de périphérique Windows. Il utilise le même interpréteur de contenu que le moteur de rendu raster et dessine via **TsgcPDFGDICanvas**. TsgcPDFViewer, TsgcPDFViewerControl, TsgcPDFThumbnailView, TsgcPDFExporter et l'impression sont construits sur lui. Windows uniquement.

Méthodes

Méthode	Description
RenderPage(ADC: HDC; AParser; APageDict; ARenderWidth, ARenderHeight: Integer; AOffsetX: Integer = 0; AOffsetY: Integer = 0)	Dessine une page sur un contexte de périphérique : un canevas de fiche ou de contrôle (`Canvas.Handle`), un bitmap mémoire ou une imprimante.
Render(ADC: HDC; AContentData: TBytes; AMediaBox: TsgcPDFRect; ...)	Dessine un flux de contenu brut.
RenderPageToBitmap(AParser; APageDict; AWidth, AHeight: Integer; const AFileName: string)	Effectue le rendu d'une page vers un fichier BMP.
RenderToBitmap(AContentData: TBytes; AMediaBox: TsgcPDFRect; AWidth, AHeight: Integer; const AFileName: string)	Effectue le rendu d'un flux de contenu brut vers un fichier BMP.

Propriétés : `Renderer`, `Canvas`, `RenderAnnotations` et `AnnotationMode`, voir Architecture du moteur de rendu.

Exemple : page vers fichier bitmap

```
procedure RenderPageWithGDI(const APDFFile, ABMPFile: string);
var
  oPDF: TsgcPDFDocument;
  oRenderer: TsgcPDFRendererGDI;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.LoadFromFile(APDFFile);
    oRenderer := TsgcPDFRendererGDI.Create;
    try
      // skip annotations and form fields, draw the page content only
      oRenderer.RenderAnnotations := False;
      oRenderer.RenderPageToBitmap(oPDF.Parser, oPDF.Pages[0].PageDict,
        1240, 1754, ABMPFile);
    finally
      oRenderer.Free;
    end;
  finally
    oPDF.Free;
  end;
end;
```

Exemple : peindre sur un contexte de périphérique

```
procedure PaintPage(ADC: HDC; APDF: TsgcPDFDocument; APageIndex, AWidth,
  AHeight: Integer);
var
  oRenderer: TsgcPDFRendererGDI;
begin
  oRenderer := TsgcPDFRendererGDI.Create;
  try
    // draws on any device context: a form canvas, a bitmap or a printer
    oRenderer.RenderPage(ADC, APDF.Parser, APDF.Pages[APageIndex].PageDict,
      AWidth, AHeight);
  finally
    oRenderer.Free;
  end;
end;
```

Voir aussi

- Architecture du moteur de rendu
- TsgcPDFRendererRaster
- TsgcPDFViewerControl

Fonctionnalités de rendu prises en charge

Cette page répertorie ce que dessine l'interpréteur de contenu. Elle s'applique à tous les moteurs de rendu (raster, GDI et FMX) et aux lecteurs construits sur eux.

Graphisme

- Chemins : lignes, courbes de Bézier, rectangles, remplissage avec les règles nonzero et even-odd, trait, découpage.
- Largeur de ligne, terminaison, jonction, limite de biseau et motifs de tirets.
- Opérateurs de couleur pour chaque espace colorimétrique (cs, CS, sc, SC, scn, SCN, g, rg, k et leurs variantes de trait).
- Alpha constant de remplissage et de trait (ExtGState /ca et /CA).
- Modes de fusion : Normal et Multiply sont composités exactement. Les autres modes de fusion sont dessinés comme Normal.
- Dégradés : basés sur une fonction (type 1), axial (type 2), radial (type 3), maillages de Gouraud en forme libre et en treillis (types 4 et 5), correctifs de Coons et de produit tensoriel (types 6 et 7), utilisés avec l'opérateur sh et comme motifs de dégradé.
- Motifs de pavage, colorés et non colorés.
- XObjects de formulaire avec leur matrice et leur rectangle englobant.

Espaces colorimétriques

DeviceGray, DeviceRGB, DeviceCMYK, CalGray, CalRGB, Lab, ICCBased (via son alternative ou son nombre de composants), Indexed, Separation et DeviceN, avec des fonctions de transformation de teinte échantillonnées (type 0), exponentielles (type 2) et de calculateur PostScript (type 4). CMJN est converti en RVB avec une formule simple, sans gestion des couleurs ICC.

Images

- XObjects image et images en ligne (BI, ID, EI).
- Filtres : DCTDecode (JPEG baseline et progressif), JPXDecode (JPEG 2000), CCITTFaxDecode (Groupe 3 et Groupe 4), JBIG2Decode, FlateDecode et LZWDecode avec prédicteurs, RunLengthDecode, ASCII85Decode et ASCIIHexDecode. Voir Décodeurs d'images.
- 1, 2, 4, 8 et 16 bits par composant, tableaux /Decode, masques d'image (masques pochoir), masques de couleur clé, flux /Mask et masques doux /SMask avec /Matte.
- Placement avec rotation, inclinaison et retournement, avec réduction d'échelle lissée.

Texte et polices

- Polices TrueType intégrées (FontFile2), polices CFF et OpenType CFF (FontFile3), polices Type 1 (FontFile), et polices Type 3 (glyphes dessinés comme des flux de contenu).
- Polices simples avec encodages WinAnsi, MacRoman, Standard et /Differences, polices symboliques, et polices composites (Type0) avec Identity-H, Identity-V et les CMaps CJK prédéfinis.
- Écriture verticale (Identity-V et les CMaps verticaux, avec les métriques /W2).
- Espacement des caractères et des mots, mise à l'échelle horizontale, décalage vertical, et les modes de rendu de texte (remplissage, trait, découpage, invisible).
- Les polices qui ne sont pas intégrées sont dessinées avec une police système installée du même nom ou un substitut. Le petit texte est suréchantillonné afin de rester lisible à basse résolution.

Annotations et formulaires

- Flux d'apparence des annotations (/AP /N), avec l'état /AS des cases à cocher et boutons radio, les indicateurs Hidden, NoView et Print, et la correspondance du rectangle englobant de l'apparence avec le /Rect de l'annotation.
- Un dessin de repli simple pour les types d'annotations courants sans flux d'apparence.
- Lorsque le formulaire définit **/NeedAppearances** sur true, les champs de texte, les champs de choix et les boutons sans apparence normale sont dessinés à partir de leurs valeurs de champ. `sgcPDFWidgetAppearanceGenerated(AParser, AWidget)` indique si un widget est dessiné de cette façon.
- Définissez `RenderAnnotations := False` pour ne dessiner que le contenu de la page, et `AnnotationMode := armPrint` pour dessiner ce qu'une imprimante imprimerait.

Non pris en charge

- Les dictionnaires de masque doux dans ExtGState (/SMask autre que /None) sont ignorés, et les groupes de transparence ne sont ni isolés ni détourés (knockout).
- Le contenu optionnel (calques) n'est pas évalué : le contenu des calques masqués est dessiné.
- Les modes de fusion autres que Normal et Multiply sont approximés en Normal.

Voir aussi

- Architecture du moteur de rendu
- Décodeurs d'images
- Polices OpenType

TsgcPDFViewerControl

TsgcPDFViewerControl (unité sgcPDF_ViewerVCL) est un contrôle VCL qui affiche des documents PDF dans vos fiches. Déposez-le sur une fiche, ou créez-le au moment de l'exécution, et chargez un fichier. Les pages sont rendues par TsgcPDFRendererGDI en arrière-plan dans un cache de pages borné, de sorte que les gros documents s'ouvrent rapidement et défilent sans à-coups. Il se trouve sur la page de palette **sgcPDF** avec les contrôles compagnons TsgcPDFThumbnailView, TsgcPDFOutlineView et TsgcPDFCommentsView.
Fonctionnalités : disposition continue ou page unique, modes de zoom, navigation au clavier, à la molette de la souris et à l'outil main, sélection et copie de texte, recherche avec mise en surbrillance, navigation par liens, remplissage interactif de formulaires, révision et modification des annotations, et impression. Compatible haute résolution (High DPI) sur Delphi 10.3 et plus récent.

Propriétés publiées

Propriété	Description
Document: TsgcPDFDocument	Affiche un TsgcPDFDocument dont vous êtes propriétaire. Laissez-le vide pour laisser le contrôle charger les fichiers lui-même.
FileName: string	Charge un fichier (aussi au moment de la conception).
PageIndex: Integer	Page courante, à partir de zéro.
Zoom: Double	Zoom en pourcentage. Le définir bascule ZoomMode sur zmCustom.
ZoomMode: TsgcPDFViewerZoomMode	zmFitWidth (par défaut), zmFitPage, zmActualSize ou zmCustom.
ViewMode: TsgcPDFViewerViewMode	pvmContinuous (par défaut, toutes les pages les unes sous les autres) ou pvmSinglePage.
PageGap: Integer	Espace entre les pages, en pixels. Par défaut 10.
BackgroundColor: TColor	Couleur autour des pages. Par défaut clAppWorkSpace.
ShowPageShadow: Boolean	Dessine une ombre sous chaque page. Par défaut True.
CacheSize: Integer	Nombre de pages rendues conservées en mémoire. Par défaut 12.
SelectionMode: TsgcPDFViewerSelectionMode	psmText (par défaut) sélectionne le texte avec la souris, psmHand fait glisser les pages.
SelectionColor, SearchColor, SearchCurrentColor: TColor	Couleurs de la sélection de texte, de tous les résultats de recherche et du résultat courant.
FormFilling: Boolean	Permet à l'utilisateur de remplir les champs de texte, cases à cocher, boutons radio, listes et listes déroulantes. Par défaut True.
OpenLinksInBrowser: Boolean	Ouvre les liens http, https et mailto avec l'application par défaut. Par défaut False.
AnnotationEditing: Boolean	Permet à l'utilisateur de sélectionner, déplacer, redimensionner et supprimer des annotations, de modifier leur texte et d'en ajouter de nouvelles. Les liens sont alors sélectionnés au lieu d'être suivis. Définir cette propriété à False efface la sé-

Propriété	Description
	lection et conserve l'outil, car les outils fonctionnent aussi lorsque AnnotationEditing vaut False. Par défaut False.
AnnotationTool: TsgcPDFViewerAnnotationTool	Outil pour la prochaine annotation placée sur une page, que AnnotationEditing vaille True ou False : vatNone (par défaut), vatNote, vatFreeText, vatHighlight, vatUnderline, vatStrikeOut, vatLine, vatArrow, vatSquare, vatCircle, vatInk ou vatStamp.
AnnotationAuthor: string	Auteur des annotations et des réponses que l'utilisateur ajoute.
AnnotationColor: TColor	Couleur des nouvelles annotations. clDefault (par défaut) utilise la couleur habituelle de chaque outil.
AnnotationStampName: string	Nom du tampon placé par vatStamp, comme Approved (par défaut), Draft ou Confidential.
ShowAnnotationPopups: Boolean	Ouvre un popup lorsqu'on clique sur une annotation ayant un contenu ou des réponses. Par défaut True.

Événements

Événement	Description
OnPageChange: TNotifyEvent	La page courante a changé.
OnZoomChange: TNotifyEvent	Le zoom a changé.
OnLinkClick(Sender; APageIndex: Integer; const AURI: string; var AHandled: Boolean)	Un lien a été cliqué. AURI est vide pour un lien vers une page du document. Définissez AHandled sur True pour annuler l'action par défaut.
OnFieldChange(Sender; AField: TsgcPDFFormField)	L'utilisateur a modifié la valeur d'un champ de formulaire.
OnSearchProgress(Sender; APageIndex, APageCount: Integer; var ACancel: Boolean)	Appelé pour chaque page pendant que FindText effectue une recherche. Définissez ACancel pour arrêter.
OnAnnotationClick(Sender; AAnnotation: TsgcPDFAnnotation; var AHandled: Boolean)	Une annotation ayant un contenu ou des réponses a été cliquée. Définissez AHandled à True pour ignorer le popup.
OnAnnotationSelect(Sender; AAnnotation: TsgcPDFAnnotation)	L'annotation sélectionnée a changé. AAnnotation vaut nil lorsque la sélection a été effacée.
OnAnnotationAdd(Sender; AAnnotation: TsgcPDFAnnotation)	L'utilisateur a ajouté une annotation avec un outil, ou une réponse dans le popup.
OnAnnotationChange(Sender; AAnnotation: TsgcPDFAnnotation)	L'utilisateur a déplacé, redimensionné ou modifié une annotation.
OnAnnotationDelete(Sender; AAnnotation: TsgcPDFAnnotation; var AAllow: Boolean)	Une annotation est sur le point d'être supprimée avec ses réponses. Définissez AAllow à False pour la conserver.

Méthodes publiques

Méthode	Description
LoadFromFile, LoadFromStream, Close	Ouvre et ferme des documents.
GotoPage, NextPage, PrevPage, FirstPage, LastPage	Navigation.
ZoomIn, ZoomOut	Modifie le zoom d'un cran.
Print(AFromPage: Integer = 0; AToPage: Integer = -1; const ATitle: string = '')	Imprime une plage de pages (à partir de zéro, -1 correspond à la dernière page) sur l'imprimante courante, avec les annotations en mode impression.
RenderPageToBitmap(APageIndex: Integer; ABitmap: TBitmap; AZoomPercent: Double = 100; ADPI: Integer = 0)	Effectue le rendu d'une page dans un bitmap VCL.
FindText(const AText: WideString; ACaseSensitive: Boolean = False; AWholeWord: Boolean = False; ABackward: Boolean = False): Boolean	Trouve et met en surbrillance chaque résultat dans le document et fait défiler jusqu'au premier.
FindNext, FindPrevious, ClearSearch, GetSearchMatch(AIndex)	Se déplace entre les résultats. SearchResults est le nombre de résultats et SearchIndex le résultat courant.
SelectText(APageIndex, AFirstChar, ACharCount), SelectPage, SelectAll, ClearSelection, CopyToClipboard	Sélection de texte. SelectedText et HasSelection lisent la sélection.
PageText(APageIndex): TsgcPDFTextExtractor	Texte d'une page avec les positions des caractères (appartient au lecteur).
DeviceToPage, PageToDevice, GetPageRect, PageRotation	Convertit entre les pixels client et l'espace utilisateur PDF.
WidgetCount, GetWidget(AIndex), EditWidget	Accède aux widgets de champs de formulaire et démarre l'édition de l'un d'eux.
ClearCache, RenderVisiblePages, CachedPageCount	Contrôle du cache de pages.
AnnotationAt(X, Y: Integer): TsgcPDFAnnotation	L'annotation la plus haute à un point client, nil s'il n'y en a pas. Les réponses, les popups, les widgets de formulaire et les annotations masquées sont ignorés, et les liens ne sont renvoyés que lorsque AnnotationEditing vaut True.
SelectedAnnotation, SelectAnnotation(AAnnotation), DeleteSelectedAnnotation: Boolean	L'annotation sélectionnée. SelectAnnotation(nil) efface la sélection. DeleteSelectedAnnotation fonctionne lorsque AnnotationEditing vaut True, déclenche OnAnnotationDelete et renvoie True lorsque l'annotation a été supprimée.
ShowAnnotationPopup(AAnnotation), CloseAnnotationPopup	Ouvrent le popup d'une annotation depuis le code, et le ferment.
GotoAnnotation(AAnnotation)	Fait défiler jusqu'à ce que l'annotation soit visible. Lorsque AnnotationEditing vaut True, elle est aussi sélectionnée.

Méthode	Description
AnnotationsChanged	Appelez-la après avoir modifié Document.Annotations dans le code. Le lecteur se repeint et abandonne la sélection ou le popup d'une annotation supprimée.

Souris et clavier

- La molette de la souris fait défiler, Ctrl + molette zoome.
- Les touches fléchées, Page précédente, Page suivante, Espace, Origine et Fin font défiler et changent de page.
- Ctrl+A sélectionne tout le texte du document, Ctrl+C et Ctrl+Inser copient la sélection.
- Tab se déplace entre les champs de formulaire pendant le remplissage.
- Un clic sur une annotation ayant un contenu ou des réponses ouvre son popup. Lorsque AnnotationEditing vaut True, un clic sélectionne une annotation, un glissement la déplace, les poignées la redimensionnent, Suppr la supprime et les touches fléchées la déplacent (de 10 points avec Maj).
- Échap pendant un glissement d'outil, de déplacement ou de redimensionnement n'annule que ce glissement. Sinon, il ferme le popup de l'annotation, efface la sélection et annule l'outil d'annotation.

Exemple : lecteur avec vignettes et signets

```
constructor TMainForm.CreateViewer(AOwner: TComponent);
begin
  CreateNew(AOwner);
  Width := 1100;
  Height := 800;
  // page thumbnails on the left
  FThumbnails := TsgcPDFThumbnailView.Create(Self);
  FThumbnails.Parent := Self;
  FThumbnails.Align := alLeft;
  FThumbnails.Width := 160;
  FThumbnails.ThumbnailWidth := 120;
  // bookmarks (outline) tree on the right
  FOutline := TsgcPDFOutlineView.Create(Self);
  FOutline.Parent := Self;
  FOutline.Align := alRight;
  FOutline.Width := 220;
  // the viewer fills the rest of the form
  FViewer := TsgcPDFViewerControl.Create(Self);
  FViewer.Parent := Self;
  FViewer.Align := alClient;
  FViewer.ZoomMode := zmFitWidth;
  FViewer.ViewMode := pvmContinuous;
  FViewer.OnPageChange := OnPageChangeEvent;
  FViewer.OnLinkClick := OnLinkClickEvent;
  FViewer.OnFieldChange := OnFieldChangeEvent;
  FViewer.OnSearchProgress := OnSearchProgressEvent;
  // link the side panels to the viewer
  FThumbnails.Viewer := FViewer;
  FOutline.Viewer := FViewer;
  FViewer.LoadFromFile('sample.pdf');
end;
```

Exemple : événements

```
procedure TMainForm.OnPageChangeEvent(Sender: TObject);
begin
  Caption := Format('Page %d of %d, zoom %.0f%%',
    [FViewer.PageIndex + 1, FViewer.PageCount, FViewer.Zoom]);
end;
```

```
procedure TMainForm.OnLinkClickEvent(Sender: TObject; APageIndex: Integer;
  const AURI: string; var AHandled: Boolean);
begin
  // AURI is empty for links to a page of the same document
  if AURI <> '' then
    AHandled := MessageDlg('Open ' + AURI + '?', mtConfirmation,
      [mbYes, mbNo], 0) <> mrYes;
end;

procedure TMainForm.OnFieldChangeEvent(Sender: TObject;
  AField: TsgcPDFFormField);
begin
  // a form field was edited in the viewer: save the filled form
  FViewer.Document.SaveToFile('filled.pdf');
end;

procedure TMainForm.OnSearchProgressEvent(Sender: TObject;
  APageIndex, APageCount: Integer; var ACancel: Boolean);
begin
  Caption := Format('Searching page %d of %d', [APageIndex + 1, APageCount]);
end;
```

Exemple : recherche et copie

```
procedure TMainForm.SearchAndCopy;
begin
  // every match of the document is highlighted, the first one is shown
  if FViewer.FindText('invoice', False, True) then
  begin
    ShowMessage(Format('%d matches', [FViewer.SearchResults]));
    FViewer.FindNext;
  end;
  // select the text of the current page and copy it to the clipboard
  FViewer.SelectPage(FViewer.PageIndex);
  if FViewer.HasSelection then
    FViewer.CopyToClipboard;
end;
```

Exemple : impression

```
procedure TMainForm.PrintPages;
begin
  // pages 1 to 3 (zero based indexes) on the default printer
  FViewer.Print(0, 2, 'My document');
end;
```

Annotations

Le lecteur dessine les annotations de `Document.Annotations`. Cliquer sur une annotation qui a un contenu ou des réponses ouvre un popup avec son auteur, sa date, son contenu et ses réponses. Gérez OnAnnotationClick, ou définissez ShowAnnotationPopups à False, pour afficher votre propre interface utilisateur à la place.
Avec **AnnotationEditing** à True, l'utilisateur peut aussi modifier les annotations :

- Un clic sélectionne une annotation et affiche ses poignées. Faites glisser l'annotation pour la déplacer, ou une poignée pour la redimensionner.
- Suppr supprime l'annotation sélectionnée avec ses réponses. OnAnnotationDelete peut la refuser.
- Les touches fléchées déplacent l'annotation sélectionnée de 1 point, ou de 10 points avec Maj.
- Le texte du popup peut être modifié, et sa zone Répondre ajoute une réponse écrite par AnnotationAuthor.
- Échap pendant un glissement d'outil, de déplacement ou de redimensionnement n'annule que ce glissement. Sinon, il ferme le popup, efface la sélection et annule l'outil en cours.

Définissez **AnnotationTool** pour placer une nouvelle annotation : cliquez sur une page, ou faites glisser dessus, et l'annotation est créée avec AnnotationAuthor et AnnotationColor. OnAnnotationAdd est déclenché et l'outil revient à vatNone.

Outil	Annotation placée
vatNone	Aucun outil (par défaut). Les clics sélectionnent les annotations.
vatNote	Note autocollante (TsgcPDFTextAnnotation).
vatFreeText	Zone de texte (TsgcPDFFreeTextAnnotation).
vatHighlight, vatUnderline, vatStrikeOut	Balisage de texte (TsgcPDFHighlightAnnotation) sur le texte que vous sélectionnez avec la souris.
vatLine, vatArrow	Ligne, ou ligne avec une flèche à son extrémité (TsgcPDFLineAnnotation).
vatSquare, vatCircle	Rectangle ou ellipse (TsgcPDFShapeAnnotation).
vatInk	Dessin à main levée (TsgcPDFInkAnnotation).
vatStamp	Tampon nommé AnnotationStampName (TsgcPDFStampAnnotation).

L'enregistrement reste à la charge de l'application. Le lecteur ne modifie le document qu'en mémoire, et `Document.Annotations.Modified` indique s'il y a quelque chose à enregistrer. Appelez `Document.SaveIncremental` pour ajouter les modifications et conserver la validité des signatures existantes, ou `Document.SaveToFile` pour un enregistrement complet. Voir Annotations. Après avoir modifié `Document.Annotations` dans le code, appelez AnnotationsChanged. Pour afficher les annotations sous forme de liste de commentaires à côté du lecteur, associez-lui un TsgcPDFCommentsView.

Exemple : annotations

```
procedure TMainForm.SetupAnnotations;
begin
  FViewer.AnnotationEditing := True;
  FViewer.AnnotationAuthor := 'Reviewer';
  // clDefault (the default) uses the usual colour of each tool
  FViewer.AnnotationColor := clRed;
  FViewer.AnnotationStampName := 'Draft';
  FViewer.OnAnnotationSelect := OnAnnotationSelectEvent;
  FViewer.OnAnnotationAdd := OnAnnotationAddEvent;
  FViewer.OnAnnotationChange := OnAnnotationChangeEvent;
  FViewer.OnAnnotationDelete := OnAnnotationDeleteEvent;
  FViewer.OnAnnotationClick := OnAnnotationClickEvent;
end;

procedure TMainForm.NoteButtonClick(Sender: TObject);
begin
  // the next click on a page places a sticky note
  FViewer.AnnotationTool := vatNote;
end;

procedure TMainForm.DeleteButtonClick(Sender: TObject);
begin
  FViewer.DeleteSelectedAnnotation;
end;

procedure TMainForm.SaveButtonClick(Sender: TObject);
begin
  if FViewer.Document.Annotations.Modified then
    // the original bytes are kept, and so are the signatures of the file
    FViewer.Document.SaveIncremental(ChangeFileExt(FViewer.FileName, '') +
      '_reviewed.pdf');
```

```
end;

procedure TMainForm.OnAnnotationSelectEvent(Sender: TObject;
  AAnnotation: TsgcPDFAnnotation);
begin
  // AAnnotation is nil when the selection is cleared
  DeleteButton.Enabled := AAnnotation <> nil;
end;

procedure TMainForm.OnAnnotationAddEvent(Sender: TObject;
  AAnnotation: TsgcPDFAnnotation);
begin
  StatusBar.SimpleText := Format('%s added on page %d',
    [AAnnotation.ClassName, AAnnotation.PageIndex + 1]);
end;

procedure TMainForm.OnAnnotationChangeEvent(Sender: TObject;
  AAnnotation: TsgcPDFAnnotation);
begin
  StatusBar.SimpleText := 'Changed: ' + AAnnotation.Contents;
end;

procedure TMainForm.OnAnnotationDeleteEvent(Sender: TObject;
  AAnnotation: TsgcPDFAnnotation; var AAllow: Boolean);
begin
  AAllow := MessageDlg('Delete this annotation and its replies?',
    mtConfirmation, [mbYes, mbNo], 0) = mrYes;
end;

procedure TMainForm.OnAnnotationClickEvent(Sender: TObject;
  AAnnotation: TsgcPDFAnnotation; var AHandled: Boolean);
begin
  // no popup for stamps in this application
  AHandled := AAnnotation is TsgcPDFStampAnnotation;
end;
```

Remplissage de formulaires

Avec FormFilling à True, cliquer sur un champ de texte ou de choix ouvre un éditeur au-dessus du champ, et cliquer sur une case à cocher ou un bouton radio bascule son état. Les valeurs des champs sont écrites dans le document (TsgcPDFDocument.AcroForm) et la page est repeinte. Gérez OnFieldChange pour réagir, et enregistrez le document avec `Document.SaveToFile` pour conserver les valeurs saisies.

Voir aussi

- TsgcPDFThumbnailView
- TsgcPDFOutlineView
- TsgcPDFCommentsView
- TsgcPDFViewerFMX
- TsgcPDFViewer
- Annotations

TsgcPDFThumbnailView

TsgcPDFThumbnailView (unité sgcPDF_ViewerVCL) affiche une colonne défilante de vignettes de pages pour un TsgcPDFViewerControl. Cliquez sur une vignette pour accéder à cette page. La page courante est mise en surbrillance, et les vignettes sont rendues à la demande et mises en cache.

Membre	Description
Viewer: TsgcPDFViewerControl	Le lecteur dont le document est affiché. Les vignettes suivent son document et sa page courante.
ThumbnailWidth: Integer	Largeur d'une vignette en pixels. Par défaut 120.
ItemCount	Nombre de vignettes (pages).
GetItemRect(AIndex): TRect, ItemAtPos(X, Y): Integer	Disposition des vignettes en coordonnées client.
RenderVisibleThumbnails, CachedThumbnailCount	Contrôle du rendu et du cache.

Les propriétés VCL habituelles (Align, Color, Font, PopupMenu...) ainsi que OnClick, OnMouseDown et OnResize sont publiées. Voir l'exemple dans TsgcPDFViewerControl.

Voir aussi

- TsgcPDFViewerControl
- TsgcPDFOutlineView

TsgcPDFOutlineView

TsgcPDFOutlineView (unité sgcPDF_ViewerVCL) est une arborescence qui affiche les signets (outlines) du document d'un TsgcPDFViewerControl. Sélectionner un nœud fait défiler le lecteur jusqu'à la destination du signet.

Membre	Description
Viewer: TsgcPDFViewer-Control	Le lecteur à suivre. L'arborescence est reconstruite lorsque le lecteur charge un autre document.
RebuildOutline	Recharge l'arborescence à partir du document.
NavigateToNode(ANode: TTreeNode)	Accède à la destination d'un nœud.

Elle descend de TCustomTreeView et publie les propriétés habituelles d'une arborescence (Items, Indent, ShowButtons, ShowLines, ShowRoot, HideSelection...) ainsi que les événements OnClick et OnChange.

Voir aussi

- TsgcPDFViewerControl
- TsgcPDFCommentsView
- Signets

TsgcPDFCommentsView

TsgcPDFCommentsView (unité sgcPDF_ViewerVCL) est une arborescence qui répertorie les commentaires du document d'un TsgcPDFViewerControl, comme le volet des commentaires d'un lecteur PDF. Toutes les annotations sont répertoriées, sauf les champs de formulaire, les liens et les popups.

Membre	Description
Viewer: TsgcPDF-ViewerControl	Le lecteur à suivre. La liste est reconstruite lorsque le lecteur charge ou ferme un document et lorsque ses annotations changent.
GroupByPage: Boolean	Répertorie les commentaires sous un nœud par page, avec le texte Page N (nombre). Lorsque la valeur est False, les commentaires forment une liste plate dans l'ordre des pages. Par défaut True.
ShowReplies: Boolean	Affiche les réponses d'un commentaire comme ses nœuds enfants. Lorsque la valeur est False, seuls les commentaires de premier niveau sont répertoriés. Par défaut True.
RebuildComments	Recharge la liste à partir du document du lecteur. La liste suit déjà le lecteur, et après avoir modifié Document.Annotations dans le code, la méthode AnnotationsChanged du lecteur la reconstruit aussi.
SelectedAnnotation: TsgcPDFAnnotation	Lecture seule. L'annotation du nœud sélectionné, nil lorsqu'aucun nœud ou un nœud de page est sélectionné.

Chaque nœud se lit `Auteur, date: contenu`. Le contenu est affiché sur une seule ligne, et les textes longs sont coupés à 120 caractères. Lorsque le contenu est vide, le type de l'annotation est affiché à la place, comme Highlight ou Stamp (Approved). Une annotation sans auteur affiche (no author), et la date est omise lorsque l'annotation n'en a pas.

- Sélectionner un nœud fait défiler le lecteur jusqu'à l'annotation avec GotoAnnotation. Lorsque AnnotationEditing du lecteur vaut True, l'annotation est aussi sélectionnée dans le lecteur.
- Un double-clic ouvre le popup de l'annotation.
- La touche Suppr supprime l'annotation sélectionnée et ses réponses lorsque AnnotationEditing du lecteur vaut True. OnAnnotationDelete du lecteur peut la refuser.
- Lorsque l'utilisateur sélectionne une annotation dans le lecteur, son nœud est sélectionné dans la liste.

Elle descend de TCustomTreeView et publie les propriétés habituelles d'une arborescence (Items, Indent, ShowButtons, ShowLines, ShowRoot, HideSelection...) ainsi que les événements OnClick et OnChange. L'arborescence est en lecture seule, et HideSelection vaut False par défaut afin que le commentaire sélectionné reste visible.

Exemple : volet des commentaires

```
procedure TMainForm.CreateCommentsPanel;
begin
  FComments := TsgcPDFCommentsView.Create(Self);
  FComments.Parent := Self;
  FComments.Align := alRight;
  FComments.Width := 280;
  // one flat list instead of the "Page N (count)" groups
  FComments.GroupByPage := False;
  FComments.ShowReplies := True;
  FComments.OnChange := OnCommentChangeEvent;
  // from now on the list follows the document, the annotations and the
  // selection of the viewer
  FComments.Viewer := FViewer;
  // a selected comment is also selected in the viewer, Delete removes it
  FViewer.AnnotationEditing := True;
end;
```

```
procedure TMainForm.OnCommentChangeEvent(Sender: TObject; Node: TTreeNode);
var
  oAnnot: TsgcPDFAnnotation;
begin
  // nil for a page group node
  oAnnot := FComments.SelectedAnnotation;
  if oAnnot <> nil then
    StatusBar.SimpleText := Format('%s, page %d', [oAnnot.Author,
      oAnnot.PageIndex + 1]);
end;
```

Voir aussi

- TsgcPDFViewerControl
- TsgcPDFOutlineView
- TsgcPDFCommentsViewFMX
- Annotations

TsgcPDFViewerFMX

TsgcPDFViewerFMX (unité sgcPDF_ViewerFMX) est un contrôle FireMonkey qui affiche des documents PDF sous Windows, macOS, iOS, Android et Linux (FMXLinux). Les pages sont dessinées par le rasteriseur logiciel en Pascal pur (TsgcPDFRendererRaster avec sortie bitmap FMX) dans une disposition verticale continue, et conservées dans un cache de pages borné. Il nécessite Delphi XE5 ou plus récent et est installé par les paquets FMX, sur la page de palette **sgcPDF**.

Propriétés publiées

Propriété	Description
Document: TsgcPDFDocument	Affiche un TsgcPDFDocument dont vous êtes propriétaire.
PageIndex: Integer	Page courante, à partir de zéro.
Zoom: Single	Facteur de zoom, 1 correspond à 100 %. Le définir bascule ZoomMode sur fzmCustom.
ZoomMode: TsgcPDFViewerFMXZoomMode	fzmFitWidth (par défaut), fzmFitPage ou fzmCustom.
PageGap: Single	Espace entre les pages.
BackgroundColor: TAlphaColor	Couleur autour des pages. Par défaut $FF808080.
CacheSize: Integer	Nombre de pages rendues conservées en mémoire. Par défaut 8.
OnPageChange, OnZoomChange, OnLayerChange: TNotifyEvent	Notifications de page, de zoom et de visibilité des calques.
OnLinkClick(Sender; APageIndex: Integer; const AURI: string; var AHandled: Boolean)	Un lien a été touché ou cliqué. Les liens vers des pages du document sont suivis sauf si AHandled est défini.
AnnotationEditing: Boolean	Permet à l'utilisateur de sélectionner, déplacer, redimensionner et supprimer des annotations, de modifier leur texte et d'en ajouter de nouvelles. Définir cette propriété à False efface la sélection et conserve l'outil, car les outils fonctionnent aussi lorsque AnnotationEditing vaut False. Un popup ouvert est rouvert dans le nouveau mode. Par défaut False.
AnnotationTool: TsgcPDFViewerAnnotationTool	Outil pour la prochaine annotation placée sur une page, que AnnotationEditing vaille True ou False : vatNone (par défaut), vatNote, vatFreeText, vatHighlight, vatUnderline, vatStrikeOut, vatLine, vatArrow, vatSquare, vatCircle, vatInk ou vatStamp.
AnnotationAuthor: string	Auteur des annotations et des réponses que l'utilisateur ajoute.
AnnotationColor: TAlphaColor	Couleur des nouvelles annotations. TAlphaColorRec.Null (par défaut) utilise la couleur habituelle de chaque outil.
AnnotationStampName: string	Nom du tampon placé par vatStamp. Par défaut Approved.
ShowAnnotationPopups: Boolean	Ouvre un popup lorsqu'on touche ou clique sur une annotation ayant un contenu ou des réponses. Par défaut True.

Propriété	Description
OnAnnotationClick(Sender; AAnnotation: TsgcPDFAnnotation; var AHandled: Boolean)	Une annotation ayant un contenu ou des réponses a été touchée ou cliquée. Définissez AHandled à True pour ignorer le popup.
OnAnnotationSelect(Sender; AAnnotation: TsgcPDFAnnotation)	L'annotation sélectionnée a changé. AAnnotation vaut nil lorsque la sélection a été effacée.
OnAnnotationAdd(Sender; AAnnotation: TsgcPDFAnnotation)	L'utilisateur a ajouté une annotation avec un outil, ou une réponse dans le popup.
OnAnnotationChange(Sender; AAnnotation: TsgcPDFAnnotation)	L'utilisateur a déplacé, redimensionné ou modifié une annotation.
OnAnnotationDelete(Sender; AAnnotation: TsgcPDFAnnotation; var AAllow: Boolean)	Une annotation est sur le point d'être supprimée avec ses réponses. Définissez AAllow à False pour la conserver.

Méthodes publiques

LoadFromFile, LoadFromStream, Close, GotoPage, NextPage, PrevPage, FirstPage, LastPage, ZoomIn, ZoomOut, ClearCache, SetScrollPosition, GetPageRect, DeviceToPage, PageToDevice, PageRotation et CachedPageCount. Calques (contenu optionnel) : GetLayers, SetLayerVisible et LayerVisible, comme dans TsgcPDFViewerControl. Lecture seule : PageCount, ScrollX, ScrollY, ContentWidth, ContentHeight et FileName.
Annotations : AnnotationAt(X, Y: Single), SelectedAnnotation (lecture et écriture), SelectAnnotation, DeleteSelectedAnnotation (renvoie True lorsque l'annotation a été supprimée), ShowAnnotationPopup, CloseAnnotationPopup, GotoAnnotation et AnnotationsChanged, que vous appelez après avoir modifié Document.Annotations dans le code. Elles fonctionnent comme dans TsgcPDFViewerControl : DeleteSelectedAnnotation ne fonctionne que lorsque AnnotationEditing vaut True, et GotoAnnotation fait défiler jusqu'à ce que l'annotation soit visible et, lorsque AnnotationEditing vaut True, la sélectionne aussi.

Tactile, souris et clavier

- Faites glisser avec le doigt ou la souris pour défiler et déplacer la vue.
- Pincez pour zoomer (le contrôle active le geste interactif Zoom).
- La molette de la souris fait défiler, Maj + molette fait défiler horizontalement, Ctrl + molette zoome.
- Page précédente, Page suivante, Origine, Fin et les touches fléchées naviguent.
- Lorsque AnnotationEditing vaut False, touchez un lien pour le suivre, et touchez une annotation ayant un contenu ou des réponses pour ouvrir son popup.
- Lorsque AnnotationEditing vaut True, un toucher sélectionne une annotation et un toucher sur un espace vide efface la sélection. Un double-clic ouvre le popup. Un glissement sur l'annotation sélectionnée la déplace, un glissement sur l'une de ses poignées la redimensionne, et un glissement ailleurs déplace la vue. Suppr supprime l'annotation sélectionnée, et les touches fléchées la déplacent de 1 point, ou de 10 points avec Maj.
- Échap pendant un glissement d'outil, de déplacement ou de redimensionnement n'annule que ce glissement. Sinon, il ferme le popup de l'annotation, efface la sélection et annule l'outil d'annotation.

Exemple

```
constructor TMainForm.CreateViewer(AOwner: TComponent);
begin
  CreateNew(AOwner);
  FViewer := TsgcPDFViewerFMX.Create(Self);
  FViewer.Parent := Self;
  FViewer.Align := TAlignLayout.Client;
  FViewer.ZoomMode := fzmFitWidth;
  FViewer.BackgroundColor := TAlphaColors.Lightgray;
  FViewer.OnPageChange := OnPageChangeEvent;
  // on mobile, open a file from the application documents folder
```

```
  FViewer.LoadFromFile('sample.pdf');
end;

procedure TMainForm.OnPageChangeEvent(Sender: TObject);
begin
  Caption := Format('Page %d of %d', [FViewer.PageIndex + 1,
    FViewer.PageCount]);
end;
```

Annotations

Le lecteur FMX offre la même prise en charge des annotations que TsgcPDFViewerControl, avec les mêmes outils et les mêmes événements. Toucher ou cliquer sur une annotation qui a un contenu ou des réponses ouvre un popup avec son auteur, sa date, son contenu et ses réponses. Lorsque AnnotationEditing vaut True, le texte du popup peut être modifié et sa zone Répondre ajoute une réponse écrite par AnnotationAuthor.

- Lorsque AnnotationEditing vaut True, un toucher ou un clic sélectionne une annotation et un double-clic ouvre son popup. Un glissement sur l'annotation sélectionnée la déplace, ses poignées la redimensionnent et Suppr la supprime avec ses réponses. Les touches fléchées déplacent l'annotation sélectionnée de 1 point, ou de 10 points avec Maj. Chaque modification déclenche OnAnnotationChange.
- Échap pendant un glissement d'outil, de déplacement ou de redimensionnement n'annule que ce glissement. Sinon, il ferme le popup, efface la sélection et annule l'outil en cours.
- AnnotationTool place une annotation par un toucher ou un glissement, puis revient à vatNone. Lorsque AnnotationEditing vaut True, la nouvelle annotation est sélectionnée. Une nouvelle note ouvre son popup, lorsque ShowAnnotationPopups vaut True, afin que son texte puisse être saisi. Un nouveau texte libre ouvre un éditeur sur place. Ctrl+Entrée, ou un toucher en dehors de l'éditeur, ajoute l'annotation avec le texte saisi. Échap, ou un texte vide, l'abandonne.
- Le lecteur FMX n'a pas de sélection de texte, de sorte que vatHighlight, vatUnderline et vatStrikeOut marquent le texte situé dans le rectangle que vous faites glisser, un quadrilatère par ligne. Lorsque le rectangle ne contient aucun texte, rien n'est ajouté et l'outil reste actif.
- Les positions sont en coordonnées du contrôle (Single) : AnnotationAt(X, Y) et PageToDevice(APageIndex, APDFX, APDFY): TPointF.

L'enregistrement reste à la charge de l'application, comme dans le lecteur VCL : appelez `Document.SaveIncremental` ou `Document.SaveToFile` lorsque `Document.Annotations.Modified` vaut True. Après l'enregistrement, `Document.Annotations.ResetModified` efface cet indicateur, et le prochain enregistrement écrit toujours chaque modification faite depuis le chargement du document. Voir Annotations. Pour afficher les annotations sous forme de liste de commentaires à côté du lecteur, associez-lui un TsgcPDFCommentsViewFMX.

Exemple : annotations

```
procedure TMainForm.SetupAnnotations;
begin
  FViewer.AnnotationEditing := True;
  FViewer.AnnotationAuthor := 'Reviewer';
  // TAlphaColorRec.Null (the default) uses the usual colour of each tool
  FViewer.AnnotationColor := TAlphaColorRec.Red;
  FViewer.OnAnnotationAdd := OnAnnotationAddEvent;
  FViewer.OnAnnotationChange := OnAnnotationChangeEvent;
  FViewer.OnAnnotationDelete := OnAnnotationDeleteEvent;
end;

procedure TMainForm.HighlightButtonClick(Sender: TObject);
begin
  // drag a rectangle over the text to highlight it
  FViewer.AnnotationTool := vatHighlight;
end;

procedure TMainForm.SaveButtonClick(Sender: TObject);
begin
  if FViewer.Document.Annotations.Modified then
    FViewer.Document.SaveIncremental(TPath.Combine(TPath.GetDocumentsPath,
      'reviewed.pdf'));
end;
```

```
procedure TMainForm.OnAnnotationAddEvent(Sender: TObject;
  AAnnotation: TsgcPDFAnnotation);
begin
  Caption := Format('%s added on page %d', [AAnnotation.ClassName,
    AAnnotation.PageIndex + 1]);
end;

procedure TMainForm.OnAnnotationChangeEvent(Sender: TObject;
  AAnnotation: TsgcPDFAnnotation);
begin
  Caption := 'Changed: ' + AAnnotation.Contents;
end;

procedure TMainForm.OnAnnotationDeleteEvent(Sender: TObject;
  AAnnotation: TsgcPDFAnnotation; var AAllow: Boolean);
begin
  // only the annotations of this user can be deleted
  AAllow := AAnnotation.Author = FViewer.AnnotationAuthor;
end;
```

Rendu vers un bitmap FMX

TsgcPDFRendererFMX (unité sgcPDF_Renderer_FMX) effectue le rendu d'une page dans un TBitmap FMX, par exemple pour un aperçu personnalisé ou pour partager une image de page. Le texte dans des polices non intégrées et non disponibles sous forme de fichiers TrueType est dessiné avec la mise en page de texte FMX de la plateforme.

```
procedure RenderPageToFMXBitmap(ADocument: TsgcPDFDocument;
  APageIndex: Integer; ABitmap: TBitmap);
var
  oRenderer: TsgcPDFRendererFMX;
begin
  oRenderer := TsgcPDFRendererFMX.Create;
  try
    // 800 x 1131 pixels, the bitmap is resized to the page raster
    oRenderer.RenderPageToBitmap(ADocument.Parser,
      ADocument.Pages[APageIndex].PageDict, 800, 1131, ABitmap);
  finally
    oRenderer.Free;
  end;
end;
```

La sélection de texte, la recherche et le remplissage de formulaires ne sont disponibles que dans le lecteur VCL.

Voir aussi

- TsgcPDFViewerControl
- TsgcPDFCommentsViewFMX
- macOS, iOS et Android
- Architecture du moteur de rendu
- Annotations

TsgcPDFCommentsViewFMX

TsgcPDFCommentsViewFMX (unité sgcPDF_ViewerFMX) est la version FireMonkey de TsgcPDFCommentsView. C'est un TTreeView qui répertorie les commentaires du document d'un TsgcPDFViewerFMX. Toutes les annotations sont répertoriées, sauf les champs de formulaire, les liens et les popups. Il nécessite Delphi XE5 ou plus récent et est installé par les packages FMX, sur la page de palette **sgcPDF**.

Membre	Description
Viewer: TsgcPDFViewerFMX	Le lecteur à suivre. La liste est reconstruite lorsque le lecteur charge ou ferme un document et lorsque ses annotations changent.
GroupByPage: Boolean	Répertorie les commentaires sous un élément par page, avec le texte Page N (nombre). Lorsque la valeur est False, les commentaires forment une liste plate dans l'ordre des pages. Par défaut True.
ShowReplies: Boolean	Affiche les réponses d'un commentaire comme ses éléments enfants. Lorsque la valeur est False, seuls les commentaires de premier niveau sont répertoriés. Par défaut True.
RebuildComments	Recharge la liste à partir du document du lecteur. La liste suit déjà le lecteur, et après avoir modifié Document.Annotations dans le code, la méthode AnnotationsChanged du lecteur la reconstruit aussi.
SelectedAnnotation: TsgcPDFAnnotation	Lecture seule. L'annotation de l'élément sélectionné, nil lorsqu'aucun élément ou un élément de page est sélectionné.

Chaque élément se lit `Auteur, date: contenu`. Le contenu est affiché sur une seule ligne, et les textes longs sont raccourcis. Lorsque le contenu est vide, le type de l'annotation est affiché à la place, comme Highlight. Une annotation sans auteur affiche (no author), et la date est omise lorsque l'annotation n'en a pas.

- Sélectionner un élément fait défiler le lecteur jusqu'à l'annotation avec GotoAnnotation. Lorsque AnnotationEditing du lecteur vaut True, l'annotation est aussi sélectionnée dans le lecteur.
- Un double-clic ouvre le popup de l'annotation.
- La touche Suppr supprime l'annotation sélectionnée et ses réponses lorsque AnnotationEditing du lecteur vaut True. OnAnnotationDelete du lecteur peut la refuser.
- Lorsque l'utilisateur sélectionne une annotation dans le lecteur, son élément est sélectionné dans la liste.

Les autres propriétés et événements, comme Align, Width et OnChange, sont ceux du TTreeView FMX. Les éléments sont créés à partir du document et ne sont pas enregistrés dans la fiche.

Exemple : volet des commentaires

```
procedure TMainForm.CreateCommentsPanel;
begin
  FComments := TsgcPDFCommentsViewFMX.Create(Self);
  FComments.Parent := Self;
  FComments.Align := TAlignLayout.Right;
  FComments.Width := 280;
  FComments.GroupByPage := True;
  FComments.ShowReplies := True;
  FComments.OnChange := OnCommentChangeEvent;
  // from now on the list follows the document, the annotations and the
  // selection of the viewer
  FComments.Viewer := FViewer;
  // a selected comment is also selected in the viewer, Delete removes it
  FViewer.AnnotationEditing := True;
end;

procedure TMainForm.OnCommentChangeEvent(Sender: TObject);
var
```

```
  oAnnot: TsgcPDFAnnotation;
begin
  // nil for a page group item
  oAnnot := FComments.SelectedAnnotation;
  if oAnnot <> nil then
    Caption := Format('%s, page %d', [oAnnot.Author, oAnnot.PageIndex + 1]);
end;
```

Voir aussi

- TsgcPDFViewerFMX
- TsgcPDFCommentsView
- Annotations

Polices OpenType

`TsgcPDFDocument.AddTrueTypeFont` (une police installée) et `AddTrueTypeFontFromFile` (un fichier .ttf, .otf ou la première face d'un fichier .ttc) acceptent les deux variantes de polices OpenType :

Contours de police	Intégré comme
TrueType (table glyf, fichiers .ttf et la plupart des .otf)	Police Type0 avec un descendant CIDFontType2 et un programme FontFile2 en sous-ensemble.
PostScript / CFF (table CFF, fichiers .otf qui commencent par OTTO)	Police Type0 avec un descendant CIDFontType0 et un programme FontFile3 en sous-ensemble de sous-type CIDFontType0C.

Dans les deux cas, seuls les glyphes utilisés dans le document sont intégrés, une table ToUnicode est écrite pour que le texte puisse être copié et recherché, et les largeurs de glyphes sont écrites dans /W. `TsgcTrueTypeFont.IsCFF` indique quelle variante possède une police chargée. Le texte est dessiné avec `TsgcPDFPage.DrawTextTT` ou `DrawTextTTColor`, qui appliquent aussi les fonctionnalités de mise en page OpenType décrites dans Écritures complexes et mise en forme.

Exemple

```
procedure CreateOpenTypeCFFPDF(const AOTFFile, AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oPage: TsgcPDFPage;
  vFont: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPage := oPDF.AddPage(595, 842);
    // an .otf font with PostScript (CFF) outlines is embedded as a
    // CIDFontType0 font with a subset FontFile3 /CIDFontType0C program
    vFont := oPDF.AddTrueTypeFontFromFile(AOTFFile);
    oPage.DrawTextTT('OpenType CFF text, subset and embedded', 50, 780,
      vFont, 16);
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

AddTrueTypeFontFromFile est le moyen d'utiliser des polices sur des serveurs, sous Linux et sur les plateformes mobiles, où la police peut ne pas être installée. `FindTrueTypeFont(AFamilyName, ABold, AItalic)` renvoie l'index d'une police déjà ajoutée, ou -1.

Voir aussi

- Écritures complexes et mise en forme
- Texte chinois, japonais et coréen
- Texte et polices

Écritures complexes et mise en forme

Le texte dessiné avec une police TrueType ou OpenType intégrée (`DrawTextTT`, `DrawTextTTColor`, TsgcPDFCanvas et TsgcPDFHTMLRenderer avec des polices TrueType, et les apparences de champs de formulaire) passe par le moteur de mise en forme OpenType de sgcPDF (unité sgcPDF_Shaping, classe TsgcPDFShaper). Il est écrit en Pascal pur et fonctionne sur toutes les plateformes. Vous passez le texte dans l'ordre logique, tel qu'il est saisi, et le moteur produit les glyphes dans l'ordre visuel :

- **Arabe** : formes de liaison (isol, init, medi, fina), ligatures obligatoires (rlig) telles que lam alef, et positionnement des signes diacritiques.
- **Hébreu** et autre texte de droite à gauche, avec des caractères mis en miroir tels que les crochets.
- **Texte bidirectionnel** : l'algorithme bidirectionnel Unicode (UAX #9) ordonne les segments mixtes de gauche à droite et de droite à gauche, nombres inclus. Le sens du paragraphe provient du premier caractère fort.
- **Ligatures et formes contextuelles** : ccmp, locl, liga, clig et calt de la table GSUB.
- **Crénage et positionnement des signes** : kern, mark, mkmk et curs de la table GPOS, ou la table kern historique lorsque la police n'a pas de crénage GPOS.

Lorsque le texte contient des segments de droite à gauche, le texte logique est aussi écrit comme /ActualText, de sorte que le copier-coller et l'extraction de texte renvoient le texte dans l'ordre où il a été saisi.

Exemple : arabe, hébreu et crénage

```
procedure CreateMultilingualPDF(const AFontFile, AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oPage: TsgcPDFPage;
  vFont: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPage := oPDF.AddPage(595, 842);
    // a TrueType or OpenType font file; the used glyphs are embedded
    vFont := oPDF.AddTrueTypeFontFromFile(AFontFile);
    // Latin with kerning and ligatures from the GPOS / GSUB tables
    oPage.DrawTextTT('Office affinity, AVATAR', 50, 780, vFont, 18);
    // Arabic: joining forms and right to left order are applied
    oPage.DrawTextTT(WideString(#$0645#$0631#$062D#$0628#$0627), 50, 740,
      vFont, 18);
    // Hebrew mixed with numbers (Unicode bidirectional algorithm)
    oPage.DrawTextTT(WideString(#$05E9#$05DC#$05D5#$05DD) + ' 2026', 50, 700,
      vFont, 18);
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Utiliser directement le moteur de mise en forme

`TsgcTrueTypeFont.ShapeText(const AText: WideString; ABaseLevel: Integer = -1): TsgcPDFShapedGlyphs` renvoie les glyphes positionnés (GlyphID, Cluster, Text, XAdvance, YAdvance, XOffset, YOffset et le Level bidi) dans l'ordre visuel, avec les avancées en unités de police. ABaseLevel vaut 0 pour la gauche vers la droite, 1 pour la droite vers la gauche et -1 pour le détecter. `EncodeShapedText(AText, AFontSize)` renvoie les opérateurs de flux de contenu pour un texte mis en forme. L'unité sgcPDF_Shaping expose aussi les fonctions utilitaires bidi `sgcPDFBidiVisual`, `sgcPDFBidiLevels` et `sgcPDFBidiReorder`.

```
procedure ShowShapedGlyphs(const AText: WideString);
var
  oFont: TsgcTrueTypeFont;
  vGlyphs: TsgcPDFShapedGlyphs;
  i: Integer;
begin
```

```
  oFont := TsgcTrueTypeFont.Create;
  try
    oFont.LoadFromFile('C:\Windows\Fonts\arial.ttf');
    // -1: the paragraph direction comes from the first strong character
    vGlyphs := oFont.ShapeText(AText, -1);
    for i := 0 to Length(vGlyphs) - 1 do
      WriteLn(Format('glyph %d  cluster %d  advance %d  level %d',
        [vGlyphs[i].GlyphID, vGlyphs[i].Cluster, vGlyphs[i].XAdvance,
        vGlyphs[i].Level]));
  finally
    oFont.Free;
  end;
end;
```

Champs de formulaire

Les valeurs de champs de formulaire contenant des caractères en dehors de l'encodage de police standard (cyrillique, grec, CJK, arabe, hébreu) obtiennent des apparences dessinées avec une police TrueType intégrée et mises en forme de la même façon. Définissez `TsgcPDFAcroForm.AppearanceFontFile` ou `AppearanceFontName` pour choisir la police. Lorsque les deux sont vides, la première police installée qui couvre les caractères est utilisée (Arial Unicode MS, Segoe UI, Arial, Yu Gothic, MS Gothic, Microsoft YaHei, Malgun Gothic, Noto Sans).

Limitations

- Les écritures qui nécessitent un moteur de mise en forme dédié, comme les écritures indiennes (devanagari, bengali, tamoul) et les écritures d'Asie du Sud-Est (thaï, khmer), sont dessinées avec les fonctionnalités génériques uniquement, sans réorganisation syllabique.
- Les caractères explicites d'incorporation et de forçage bidi (LRE, RLE, LRO, RLO, isolats) sont ignorés.

Voir aussi

- Polices OpenType
- Texte chinois, japonais et coréen
- Formulaires

Texte chinois, japonais et coréen

Pour écrire du texte chinois, japonais ou coréen, ajoutez une police contenant les caractères avec `AddTrueTypeFont` ou `AddTrueTypeFontFromFile` et dessinez avec `DrawTextTT`. La police est intégrée sous forme de sous-ensemble (seuls les glyphes utilisés), de sorte que le fichier reste petit même avec de grandes polices CJK, et il s'affiche de la même façon sur tous les ordinateurs. Les collections TrueType (.ttc) sont acceptées : la première face de la collection est utilisée.

```
procedure CreateJapanesePDF(const AFontFile, AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oPage: TsgcPDFPage;
  vFont: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPage := oPDF.AddPage(595, 842);
    // for example C:\Windows\Fonts\msgothic.ttc (the first face is used)
    vFont := oPDF.AddTrueTypeFontFromFile(AFontFile);
    oPage.DrawTextTT(WideString(#$65E5#$672C#$8A9E#$306E#$30C6#$30AD#$30B9#$30C8),
      50, 780, vFont, 20);
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Polices Windows courantes : MS Gothic, Yu Gothic et Meiryo (japonais), SimSun et Microsoft YaHei (chinois simplifié), MingLiU et Microsoft JhengHei (chinois traditionnel), Batang, Gulim et Malgun Gothic (coréen). Sous Linux, utilisez par exemple les polices Noto Sans CJK ou WenQuanYi avec AddTrueTypeFontFromFile.

Lecture et rendu des documents CJK

L'extraction de texte et les moteurs de rendu prennent en charge les documents CJK créés par d'autres outils : polices composites avec les encodages Identity-H et Identity-V et les CMaps CJK prédéfinis, polices CID intégrées (TrueType et CFF), et écriture verticale. Les documents qui utilisent des polices CJK sans les intégrer sont rendus avec une police installée du système.

TsgcPDFCJKFontManager

L'unité sgcPDF_Fonts_CJK (Windows uniquement) conserve les anciennes classes de bas niveau TsgcPDFCIDFont et TsgcPDFCJKFontManager ainsi que les fonctions utilitaires `sgcPDFIsCJKCodePoint` et `TsgcPDFCJKFontManager.ContainsCJK`. Ces classes ne sont pas connectées à TsgcPDFDocument. Utilisez AddTrueTypeFont et DrawTextTT pour tout nouveau code.

Voir aussi

- Polices OpenType
- Écritures complexes et mise en forme

Extraction de texte : positions des caractères

Outre le texte brut et les fragments, **TsgcPDFTextExtractor** (unité sgcPDF_TextExtract) renvoie chaque caractère avec sa position, regroupé en mots et en lignes dans l'ordre de lecture. C'est ce qu'utilise le lecteur VCL pour la sélection de texte et la mise en surbrillance des résultats de recherche, et vous pouvez l'utiliser pour la recherche de texte avec coordonnées, l'extraction de données de mises en page fixes ou le caviardage du texte trouvé.

API caractères, mots et lignes

Méthode	Description
GetCharCount, GetChar(AIndex): TsgcPDFTextChar	Caractères dans l'ordre de lecture de GetText. TsgcPDFTextChar comporte le quadrilatère du glyphe (X1, Y1 à X4, Y4, suivant le sens du texte), le rectangle englobant (Left, Bottom, Right, Top), l'origine (OriginX, OriginY), FontSize, LineIndex, WordIndex (-1 pour un espace) et Synthetic (un espace inséré par l'extracteur entre deux segments de texte).
GetWordCount, GetWord(AIndex): TsgcPDFTextWord	Mots sous forme d'une plage de caractères (FirstChar, CharCount), avec LineIndex et rectangle englobant.
GetLineCount, GetLine(AIndex): TsgcPDFTextLine	Lignes sous forme d'une plage de caractères avec leur rectangle englobant.
GetCharsText: WideString	Les caractères assemblés avec CRLF entre les lignes. Même texte que GetText.
CharAt(AX, AY: Double; ATolerance: Double = 0): Integer	Index du caractère à un point de l'espace utilisateur, ou -1. ATolerance élargit les rectangles des caractères.
IncludeAnnotations: Boolean	Extrait aussi le texte affiché par les flux d'apparence visibles des annotations (valeurs de champs de formulaire, texte libre et tampons). Par défaut False.

`sgcPDFCreatePageTextExtractor(AParser, APageIndex)` crée un extracteur pour une page et exécute Extract (à libérer par l'appelant, nil lorsque la page n'existe pas). Toutes les positions sont exprimées dans l'espace utilisateur PDF : en points, origine dans le coin inférieur gauche.

Exemple

```
procedure ListCharacterPositions(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oText: TsgcPDFTextExtractor;
  vChar: TsgcPDFTextChar;
  vWord: TsgcPDFTextWord;
  i: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.LoadFromFile(AFileName);
    oText := TsgcPDFTextExtractor.Create(oPDF.Parser, oPDF.Pages[0].PageDict);
    try
      // also read the text shown by annotation appearance streams
```

```
      oText.IncludeAnnotations := True;
      oText.Extract;
      for i := 0 to oText.GetWordCount - 1 do
      begin
        vWord := oText.GetWord(i);
        WriteLn(Format('word %d: line %d, box %.1f %.1f %.1f %.1f',
          [i, vWord.LineIndex, vWord.Left, vWord.Bottom, vWord.Right,
          vWord.Top]));
      end;
      if oText.GetCharCount > 0 then
      begin
        vChar := oText.GetChar(0);
        WriteLn(Format('first char at %.1f, %.1f size %.1f',
          [vChar.OriginX, vChar.OriginY, vChar.FontSize]));
        // hit test: the character under a point of the page (user space)
        WriteLn('char at point: ', oText.CharAt(vChar.OriginX + 1,
          vChar.OriginY + 2, 1));
      end;
    finally
      oText.Free;
    end;
  finally
    oPDF.Free;
  end;
end;
```

Extraction améliorée

- Le texte à l'intérieur des XObjects de formulaire est extrait, avec la matrice du formulaire appliquée aux positions.
- /ActualText du contenu marqué remplace les glyphes qu'il couvre, de sorte que les ligatures, l'arabe mis en forme et les marques de coupure de mots sont extraits comme le texte d'origine.
- Les polices sans table ToUnicode sont mappées via leur encodage, /Differences et les noms de glyphes.

Voir aussi

- TsgcPDFTextExtractor
- Extraction de texte
- TsgcPDFViewerControl

Limites pour les fichiers non approuvés

sgcPDF sert souvent à ouvrir des fichiers venant de l'extérieur : envois d'utilisateurs, pièces jointes d'e-mails, documents numérisés. Chaque analyseur et décodeur vérifie ses données d'entrée et déclenche une exception sur des données corrompues plutôt que de lire hors limites. Les allocations sont plafonnées par ces variables globales, que vous pouvez réduire (ou augmenter) avant de charger des documents :

Variable	Unité	Par défaut	Limite
sgcPDFMaxDecodedSize: Int64	sgcPDF_Filters	268435456 (256 Mo)	Taille décodée maximale d'un flux (Flate, LZW, RunLength, ASCII85, ASCIIHex, CCITT, JBIG2). Protège contre les bombes de décompression.
sgcPDFJPEGMaxPixels: Int64	sgcPDF_JPEG	67108864	Trame JPEG maximale (largeur x hauteur).
sgcPDFJPEGMaxMemory: Int64	sgcPDF_JPEG	1073741824 (1 Go)	Mémoire de travail du décodeur JPEG pour une image.
sgcPDFJPXMaxPixels: Int64	sgcPDF_JPX	67108864	Image JPEG 2000 maximale (largeur x hauteur).
CS_CANVAS_MAX_RASTER (constante)	sgcPDF_Renderer_Canvas	67108864	Image matricielle maximale construite par les moteurs de rendu, en pixels.

```
procedure ConfigureLimits;
begin
  // largest decoded stream (default 256 MB); larger streams raise
  sgcPDFMaxDecodedSize := 64 * 1024 * 1024;
end;
```

D'autres protections sont intégrées : limites de récursion pour les XObjects de formulaire imbriqués, les motifs, les glyphes Type 3, les arbres de pages et les plans, détection de cycles dans les références d'objets, et caches bornés dans les lecteurs. Une image endommagée est ignorée ou dessinée partiellement, et le reste de la page est quand même rendu.
Ces variables sont globales au processus. Définissez-les une seule fois au démarrage, pas pendant que d'autres threads chargent des documents.

Voir aussi

- Gestion des erreurs
- Décodeurs d'images

Décodeurs d'images

sgcPDF inclut des décodeurs en Pascal pur pour chaque compression d'image utilisée dans les fichiers PDF. Les moteurs de rendu, les lecteurs, TsgcPDFImageExtractor et l'optimiseur les utilisent, et vous pouvez les appeler directement. Ils fonctionnent sur toutes les plateformes et ne nécessitent aucune DLL.

Format	Unité et fonction	Pris en charge
JPEG (DCTDecode)	sgcPDF_JPEG: sgcPDFDecodeJPEG, TsgcPDFJPEGDecoder	Huffman baseline et progressif, 8 bits, 1, 3 ou 4 composants, tous facteurs d'échantillonnage, marqueurs de redémarrage, transformations JFIF et Adobe (YCbCr, YCCK, CMYK). Le codage arithmétique, le JPEG sans perte et le JPEG 12 bits déclenchent EsgcPDFJPEGError.
JPEG 2000 (JPXDecode)	sgcPDF_JPX: sgcPDFDecodeJPX	Fichiers JP2 et flux de code bruts, tuiles, tous les ordres de progression, ondelettes réversibles 5/3 et irréversibles 9/7, sous-échantillonnage des composants, 1 à 16 bits, palettes, canaux alpha, sYCC.
CCITT Groupe 3 et Groupe 4 (CCITTFaxDecode)	sgcPDF_CCITT: sgcPDFCCITTFaxDecode, sgcPDFCCITTFaxDecodeParms	Groupe 3 unidimensionnel (MH), Groupe 3 bidimensionnel (MR) et Groupe 4 (MMR), EndOfLine, EncodedByteAlign, EndOfBlock, BlackIs1.
JBIG2 (JBIG2Decode)	sgcPDF_JBIG2: sgcPDFJBIG2Decode	Régions génériques (arithmétique et MMR), dictionnaires de symboles et régions de texte avec codage arithmétique, /JBIG2Globals. Les segments de demi-teinte, de motif, de raffinement et les symboles et textes codés en Huffman ne sont pas pris en charge et déclenchent une exception.

Signatures

- `sgcPDFDecodeJPEG(const AData: TBytes; out AWidth, AHeight, AComponents: Integer): TBytes` et une surcharge avec `AColorTransform: Integer` (la valeur PDF /ColorTransform, -1 si absente). Renvoie des échantillons 8 bits entrelacés.
- `sgcPDFDecodeJPX(const AData: TBytes; out AInfo: TsgcPDFJPXInfo; AKeep16: Boolean = False): TBytes.` AInfo fournit Width, Height, Components, ColorComponents, HasAlpha, BitsPerComponent, ColorSpace (sgcJPXColorGray, sgcJPXColorRGB, sgcJPXColorCMYK, sgcJPXColorICC) et le profil ICC.
- `sgcPDFCCITTFaxDecode(const AData: TBytes; AK, AColumns, ARows: Integer; AEndOfLine, AEncodedByteAlign, AEndOfBlock, ABlackIs1: Boolean; ADamagedRowsBeforeError: Integer = 0): TBytes.` AK < 0 correspond au Groupe 4, 0 au Groupe 3 MH, > 0 au Groupe 3 MR. Renvoie des lignes de 1 bit alignées sur un octet.
- `sgcPDFJBIG2Decode(const AData, AGlobals: TBytes; out AWidth, AHeight: Integer): TBytes.` Renvoie des lignes de 1 bit selon la convention PDF (0 correspond au noir).

Exemple

```
procedure DecodeJPEGFile(const AFileName: string);
var
  vPixels: TBytes;
  vWidth, vHeight, vComponents: Integer;
begin
  // 1 (gray), 3 (RGB) or 4 (CMYK) bytes per pixel
  vPixels := sgcPDFDecodeJPEG(ReadFileBytes(AFileName), vWidth, vHeight,
    vComponents);
  WriteLn(Format('%d x %d, %d components', [vWidth, vHeight, vComponents]));
```

```
end;
```

Tous les décodeurs rejettent les données corrompues avec une exception et plafonnent leurs allocations, voir Limites pour les fichiers non approuvés.

Voir aussi

- TsgcPDFImageExtractor
- Import TIFF
- Fonctionnalités de rendu prises en charge

Import TIFF

TsgcPDFImage et TsgcPDFDocument.AddImage chargent les fichiers TIFF en plus de JPEG, PNG et BMP. Cela facilite la transformation de documents numérisés et de fichiers fax en PDF.

Fichiers TIFF pris en charge

- Petit et grand boutisme (II et MM), bandes et tuiles.
- Compression : aucune, PackBits, LZW, Deflate (avec prédicteur horizontal), CCITT Groupe 3 et Groupe 4, et JPEG.
- Images bileveau, niveaux de gris, RVB, palette et CMJN, 1 à 16 bits par échantillon (16 bits sont réduits à 8).
- Fichiers multi-pages : chaque page (IFD) peut être ajoutée.

Les pages CCITT et JPEG en bande unique sont intégrées sans réencodage, sous forme de flux CCITTFaxDecode et DCTDecode, de sorte qu'une page de fax numérisée conserve sa taille et sa qualité d'origine. Les autres pages sont compressées en Flate.

API

Membre	Description
TsgcPDFDocument.AddImage(const AFileName: string; APageIndex: Integer): Integer	Ajoute la page APageIndex (à partir de zéro) d'un fichier TIFF multi-pages. Les autres formats n'acceptent que la page 0.
TsgcPDFImage.TIFFPageCount(const AData: TBytes): Integer	Fonction de classe : nombre de pages d'un fichier TIFF.
TsgcPDFImage.LoadTIFF(const AData: TBytes; APageIndex: Integer = 0)	Charge une page dans un TsgcPDFImage (puis ajoutez-la avec TsgcPDFDocument.AddImageObject).
TsgcPDFImage.IsCCITT: Boolean	True lorsque la page est intégrée sous forme de données CCITT sans réencodage.

Exemple : TIFF multi-pages vers PDF

```
procedure TIFFToPDF(const ATIFFFile, APDFFile: string);
var
  oPDF: TsgcPDFDocument;
  oPage: TsgcPDFPage;
  vPages, vImage, i: Integer;
begin
  // number of pages (IFDs) of the TIFF file
  vPages := TsgcPDFImage.TIFFPageCount(ReadFileBytes(ATIFFFile));
  oPDF := TsgcPDFDocument.Create;
  try
    for i := 0 to vPages - 1 do
    begin
      oPage := oPDF.AddPage(595, 842);
      // APageIndex selects the TIFF page; CCITT G3/G4 and JPEG strips are
      // embedded without re-encoding
      vImage := oPDF.AddImage(ATIFFFile, i);
      oPage.DrawImageFromIndex(vImage, 50, 300, 495, 371);
    end;
    oPDF.SaveToFile(APDFFile);
  finally
```

```
    oPDF.Free;
  end;
end;
```

Voir aussi

- Images
- Décodeurs d'images

TsgcPDFImageExtractor

TsgcPDFImageExtractor (unité sgcPDF_ImageExtract) répertorie les images d'un PDF et les enregistre sous forme de fichiers PNG, JPEG ou BMP, ou renvoie leurs pixels. Il trouve les XObjects image, les images en ligne et les images à l'intérieur des XObjects de formulaire (et éventuellement à l'intérieur des annotations), avec la position où chaque image est peinte. Il se trouve sur la page de palette sgcPDF.

Propriétés

Propriété	Description
Document: TsgcPDFDocument	Un document chargé.
Parser: TsgcPDFParser	Un analyseur explicite. Il prend le pas sur Document.
IncludeInline: Boolean	Répertorie les images en ligne (BI, ID, EI). Par défaut True.
IncludeAnnotations: Boolean	Répertorie les images peintes par les flux d'apparence des annotations. Par défaut False.
IncludeUnused: Boolean	Répertorie les images des ressources de la page que le contenu ne peint jamais. Par défaut True.
UniqueOnly: Boolean	Répertorie chaque objet image une seule fois. UseCount indique combien de fois il est peint. Par défaut False.
IncludeAlpha: Boolean	Ajoute la transparence /SMask ou /Mask comme canal alpha. Par défaut True.
MaxFormDepth: Integer	Imbrication maximale des XObjects de formulaire. Par défaut 16.
FileNamePattern: string	Nom de fichier utilisé par ExtractAll, sans extension : %p numéro de page, %i numéro d'image, %n numéro d'image dans la page, %o numéro d'objet, %r nom de ressource.
Count, Items[AIndex]: TsgcPDFImageItem	Les images trouvées. L'analyse s'exécute à la demande.
Errors: TStringList	Images qu'ExtractAll n'a pas pu enregistrer.

Méthodes

Méthode	Description
Scan, ScanPage(APageIndex), Clear	Répertorie les images de toutes les pages ou d'une seule page.
SaveToFile(AIndex; const AFileName: string)	Choisit le format d'après l'extension : .png, .jpg, .jpeg, .bmp. Les extensions .jp2, .jpx, .j2k, .jb2, .bin et .raw enregistrent les données du flux telles que stockées.
SaveAsPNG, SaveAsJPEG, SaveAsBMP, SaveToStream(AIndex; AStream; AFormat)	Enregistre une image. AFormat : pifAuto (images JPEG en JPEG, JPEG 2000 tel que stocké, le reste en PNG), pifPNG, pifJPEG, pifBMP ou pifRaw.

Méthode	Description
ExtractAll(const AFolder: string; AFormat = pifAuto): Integer	Enregistre chaque image répertoriée et renvoie le nombre de fichiers écrits.
GetPixels(AIndex; out AWidth, AHeight, AChannels: Integer): TBytes	Pixels 8 bits décodés, de haut en bas : 1 Gray, 2 Gray et alpha, 3 RGB, 4 RGBA.
GetRawBytes, GetJPEGBytes, GetPNGBytes, GetBMPBytes, CanSaveAsJPEG	Données d'image en mémoire. La sortie JPEG est disponible pour les images DCTDecode, qui sont enregistrées sans recompression.

TsgcPDFImageItem décrit une image : PageIndex, ObjectNumber, Name, Source (pisXObject ou pisInline), Width, Height, BitsPerComponent, ColorSpace, Filters, Matrix, Rect (la zone peinte, en points), Placed, InAnnotation, HasSMask, HasMask, IsImageMask, UseCount et Stream.

Exemple : lister et enregistrer des images

```
procedure ExtractImages(const APDFFile, AFolder: string);
var
  oPDF: TsgcPDFDocument;
  oExtractor: TsgcPDFImageExtractor;
  oItem: TsgcPDFImageItem;
  i: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.LoadFromFile(APDFFile);
    oExtractor := TsgcPDFImageExtractor.Create(nil);
    try
      oExtractor.Document := oPDF;
      oExtractor.UniqueOnly := True;
      for i := 0 to oExtractor.Count - 1 do
      begin
        oItem := oExtractor.Items[i];
        WriteLn(Format('page %d: %dx%d %s [%s] at %.0f,%.0f', [oItem.PageIndex +
          1, oItem.Width, oItem.Height, oItem.ColorSpace, oItem.Filters,
          oItem.Rect.Left, oItem.Rect.Bottom]));
      end;
      // one file per image: page number and image number in the name
      oExtractor.FileNamePattern := 'page%p_img%n';
      WriteLn('files written: ', oExtractor.ExtractAll(AFolder, pifPNG));
      // or a single image in the format given by the extension
      if oExtractor.Count > 0 then
        oExtractor.SaveToFile(0, IncludeTrailingPathDelimiter(AFolder) +
          'first.bmp');
    finally
      oExtractor.Free;
    end;
  finally
    oPDF.Free;
  end;
end;
```

Exemple : pixels

```
procedure ReadImagePixels(const APDFFile: string);
var
  oPDF: TsgcPDFDocument;
  oExtractor: TsgcPDFImageExtractor;
  vPixels: TBytes;
  vWidth, vHeight, vChannels: Integer;
```

```
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.LoadFromFile(APDFFile);
    oExtractor := TsgcPDFImageExtractor.Create(nil);
    try
      oExtractor.Document := oPDF;
      oExtractor.ScanPage(0);
      if oExtractor.Count > 0 then
      begin
        // decoded samples, 8 bits per channel, top-down rows
        vPixels := oExtractor.GetPixels(0, vWidth, vHeight, vChannels);
        WriteLn(Format('%d x %d, %d channels, %d bytes', [vWidth, vHeight,
          vChannels, Length(vPixels)]));
      end;
    finally
      oExtractor.Free;
    end;
  finally
    oPDF.Free;
  end;
end;
```

Les encodeurs sont aussi disponibles sous forme de fonctions : `sgcPDFEncodePNG` et `sgcPDFEncodeBMP(const APixels: TBytes; AWidth, AHeight, AChannels: Integer): TBytes`.

Voir aussi

- Décodeurs d'images
- Images

TsgcPDFCanvas

TsgcPDFCanvas (unité sgcPDF_Canvas) dessine sur une page PDF avec une API calquée sur le TCanvas de la VCL : objets Pen, Brush et Font, MoveTo et LineTo, Rectangle, Ellipse, Arc, Pie, TextOut, TextRect, etc. Tout est écrit sous forme de graphismes vectoriels et de texte réel dans le contenu de la page. C'est le moyen le plus simple de porter vers PDF du code d'impression TCanvas existant, et c'est multiplateforme (sans VCL, sans GDI).

Coordonnées

L'origine du canevas est le coin **supérieur gauche** de la page et Y croît vers le bas, comme un TCanvas. Choisissez l'unité dans le constructeur : `Create(APage: TsgcPDFPage; AUnit: TsgcPDFCanvasUnit = pcuPoints; ADPI: Double = 96)`, avec pcuPoints, pcuPixels (à ADPI), pcuMillimeters ou pcuInches. Translate, Rotate, Scale, MultiplyTransform, SetTransform et ResetTransform modifient la transformation.

Principaux membres

Membre	Description
Pen	Color, Width (0 dessine la ligne visible la plus fine), Style (ppsSolid, ppsDash, ppsDot, ppsDashDot, ppsDashDotDot, ppsClear), Cap, Join, MiterLimit, GapFill et GapColor.
Brush	Color et Style (pbsSolid, pbsClear).
Font	Name, Size (points), Style (pfsBold, pfsItalic, pfsUnderline, pfsStrikeOut) et Color.
Opacity: Double	Alpha de remplissage et de trait, de 0 à 1.
TrueTypeFonts: Boolean	Les polices autres qu'Arial/Helvetica, Times et Courier sont intégrées comme TrueType lorsqu'elles sont installées. False fait correspondre chaque police à l'une des 14 polices standard.
MoveTo, LineTo, Rectangle, FillRect, FrameRect, RoundRect, Ellipse, Arc, ArcTo, Pie, Chord, Polygon, Polyline, PolylineTo, PolyBezier, PolyBezierTo	Lignes et formes, avec les conventions de paramètres de TCanvas.
BeginPath, EndPath, CloseFigure, FillPath, StrokePath, StrokeAndFillPath, ClipPath	Chemins.
IntersectClipRect, ExcludeClipRect, ResetClip, SaveState, RestoreState	Découpage (clipping) et état.
TextOut, TextOutEx, TextRect, TextWidth, TextHeight, TextAscent, TextDescent	Texte, avec alignement, retour à la ligne automatique, avancées de caractères et rotation (TextOutEx).
DrawImage(X, Y, W, H; AImageIndex | AData: TBytes | AStream | AImage: TsgcPDFImage)	Images.
EndDraw	Ferme le groupe de découpage ouvert. Appelez cette méthode lorsque vous terminez de dessiner une page.

Les couleurs sont des valeurs TsgcPDFCanvasColor dans la disposition TColor ($00BBGGRR). Construisez-les avec `sgcPDFCanvasRGB(R, G, B)`.

Exemple

```
procedure DrawWithCanvas(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oCanvas: TsgcPDFCanvas;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    // millimeters, origin at the top left corner, Y grows downwards
    oCanvas := TsgcPDFCanvas.Create(oPDF.AddPage(595, 842), pcuMillimeters);
    try
      oCanvas.Pen.Color := sgcPDFCanvasRGB(0, 64, 128);
      oCanvas.Pen.Width := 0.5;
      oCanvas.Brush.Color := sgcPDFCanvasRGB(220, 235, 250);
      oCanvas.RoundRect(20, 20, 190, 60, 6, 6);
      oCanvas.Font.Name := 'Arial';
      oCanvas.Font.Size := 18;
      oCanvas.Font.Style := [pfsBold];
      oCanvas.TextRect(20, 20, 190, 60, 'Drawn with TsgcPDFCanvas', ptaCenter,
        ptvCenter);
      oCanvas.Brush.Color := sgcPDFCanvasRGB(255, 200, 0);
      oCanvas.Pie(20, 80, 80, 140, 80, 110, 50, 80);
      oCanvas.Pen.Style := ppsDash;
      oCanvas.MoveTo(100, 80);
      oCanvas.LineTo(190, 140);
      // semi transparent overlay
      oCanvas.Opacity := 0.4;
      oCanvas.Brush.Color := sgcPDFCanvasRGB(255, 0, 0);
      oCanvas.Ellipse(120, 90, 170, 130);
      oCanvas.EndDraw;
    finally
      oCanvas.Free;
    end;
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Voir aussi

- EMF vers PDF
- TsgcPDFVCLCanvasHelper
- HTML vers PDF
- Graphisme

EMF vers PDF

TsgcPDFEMFPlayer (unité sgcPDF_EMF, Windows uniquement) rejoue les métafichiers améliorés Windows (EMF) dans un TsgcPDFCanvas. Les lignes, formes, chemins, textes et bitmaps du métafichier deviennent des graphismes vectoriels, du texte réel et des images dans le PDF, de sorte que les graphiques et rapports dessinés par d'autres composants restent nets à tout niveau de zoom et que le texte peut être recherché.

API

Membre	Description
Create(ACanvas: TsgcPDFCanvas)	Le canevas qui reçoit le dessin.
Play(AHandle: HENHMETAFILE; X, Y, W, H: Double)	Rejoue un métafichier de sorte que son cadre remplisse le rectangle (unités du canevas).
PlayFromFile, PlayFromStream, PlayFrom-Bytes	Identique, depuis un fichier .emf, un flux ou des octets.
GetFrameSize(AHandle; out AWidth, AHeight: Double)	Procédure de classe : taille du cadre du métafichier en points.
RecordCount, SkippedRecords	Nombre d'enregistrements rejoués et d'enregistrements non pris en charge.
sgcPDFEMFAddPage(ADocument; AHandle | AFileName): TsgcPDFPage	Ajoute une page à la taille du cadre du métafichier et y dessine le métafichier.

Exemple

```
procedure EMFToPDF(const AEMFFile, APDFFile: string);
var
  oPDF: TsgcPDFDocument;
  oCanvas: TsgcPDFCanvas;
  oPlayer: TsgcPDFEMFPlayer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    // a page with the size of the metafile frame
    sgcPDFEMFAddPage(oPDF, AEMFFile);
    // or play a metafile into a rectangle of any page
    oCanvas := TsgcPDFCanvas.Create(oPDF.AddPage(595, 842));
    try
      oPlayer := TsgcPDFEMFPlayer.Create(oCanvas);
      try
        oPlayer.PlayFromFile(AEMFFile, 50, 50, 300, 225);
        WriteLn(Format('records %d, skipped %d', [oPlayer.RecordCount,
          oPlayer.SkippedRecords]));
      finally
        oPlayer.Free;
      end;
      oCanvas.EndDraw;
    finally
      oCanvas.Free;
    end;
    oPDF.SaveToFile(APDFFile);
  finally
    oPDF.Free;
```

```
  end;
end;
```

Voir aussi

- TsgcPDFCanvas
- TsgcPDFVCLCanvasHelper

TsgcPDFVCLCanvasHelper

TsgcPDFVCLCanvasHelper (unité sgcPDF_CanvasVCL) relie les graphismes VCL à un TsgcPDFCanvas. Windows et VCL uniquement.

Membre	Description
Create(ACanvas: TsgcPDFCanvas)	Le canevas cible.
DrawMetafile(AMetafile: TMetafile; X, Y, W, H: Double)	Rejoue un TMetafile sous forme de graphismes vectoriels.
DrawGraphic(AGraphic: TGraphic; X, Y, W, H: Double)	Dessine tout TGraphic : les métafichiers sous forme de graphismes vectoriels, les bitmaps, images PNG et JPEG sous forme d'images.
DrawControl(AControl: TWinControl; X, Y, W, H: Double)	Dessine un contrôle et ses enfants (par exemple un graphique ou une grille) sous forme de graphismes vectoriels, via un métafichier amélioré.
ControlToMetafile(AControl: TWinControl): TMetafile	Fonction de classe : enregistre un contrôle dans un nouveau métafichier (à libérer par l'appelant).

X, Y, W et H sont exprimés en unités du canevas.

Exemple

```
procedure GraphicToPDF(AGraphic: TGraphic; const APDFFile: string);
var
  oPDF: TsgcPDFDocument;
  oCanvas: TsgcPDFCanvas;
  oHelper: TsgcPDFVCLCanvasHelper;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oCanvas := TsgcPDFCanvas.Create(oPDF.AddPage(595, 842));
    try
      oHelper := TsgcPDFVCLCanvasHelper.Create(oCanvas);
      try
        // a TMetafile is played as vector graphics, other graphics as images
        oHelper.DrawGraphic(AGraphic, 50, 50, 400, 300);
      finally
        oHelper.Free;
      end;
      oCanvas.EndDraw;
    finally
      oCanvas.Free;
    end;
    oPDF.SaveToFile(APDFFile);
  finally
    oPDF.Free;
  end;
end;
```

Voir aussi

- TsgcPDFCanvas
- EMF vers PDF

HTML vers PDF

TsgcPDFHTMLRenderer (unité sgcPDF_HTML) convertit du HTML avec CSS en pages PDF paginées. Il dispose de son propre moteur de mise en page écrit en Pascal pur, il n'a donc besoin d'aucun navigateur et fonctionne sur toutes les plateformes. Utilisez-le pour des factures, rapports, lettres et archives d'e-mails générés à partir de modèles HTML. Il se trouve sur la page de palette sgcPDF.

Méthodes

Méthode	Description
HTMLToPDFFile(const AHTML, APDFFileName: string; const ABasePath: string = '')	Crée un fichier PDF à partir d'une chaîne HTML.
HTMLFileToPDFFile(const AHTMLFileName, APDFFileName: string)	Crée un fichier PDF à partir d'un fichier HTML.
RenderHTML(const AHTML: string; ADocument: TsgcPDFDocument; const ABasePath: string = ''): Integer	Ajoute les pages à un document et renvoie le nombre de pages ajoutées. RenderHTMLW prend une WideString, RenderHTMLBytes des octets bruts (l'encodage provient du BOM, de <meta charset> ou d'une détection UTF-8) et RenderHTMLFile un fichier.

ABasePath résout les chemins relatifs des images et des feuilles de style. Dans les versions de Delphi antérieures à 2009, un AnsiString peut contenir de l'UTF-8 ou du Windows-1252, ce qui est détecté.

Propriétés

Propriété	Description
PageWidth, PageHeight: Double	Taille de la page en points. La règle CSS @page size la remplace.
Margins: TsgcPDFHTMLMargins	Left, Top, Right et Bottom en points. La règle CSS @page margin les remplace.
DefaultFontName, DefaultFontSize	Police du texte sans font-family ni font-size.
TrueTypeFonts: Boolean	Intègre les familles de polices autres que les 14 standard sous forme de polices TrueType. Nécessaire pour le texte en dehors du jeu de caractères Windows-1252.
HeaderText, FooterText: string	Texte centré dans les marges du haut et du bas. {page} et {pages} sont remplacés par le numéro de page et le nombre total de pages.
MaxPages: Integer	S'arrête après ce nombre de pages.
OnPageHeader, OnPageFooter(Sender; ACanvas: TsgcPDFCanvas; APageNumber, APageCount: Integer)	Dessinez votre propre en-tête et pied de page avec un TsgcPDFCanvas.
LastPageCount	Pages ajoutées par le dernier rendu.

HTML pris en charge

- Éléments de bloc : html, body, div, p, h1 à h6, blockquote, pre, hr, address, center, section, article, header, footer, nav, aside, main, figure, figcaption, form, fieldset, details, summary.
- Listes : ul, ol et li (disc, circle, square, decimal, decimal-leading-zero, alpha minuscule et majuscule, latin et romain), dl, dt et dd.
- Éléments en ligne : span, a, b, strong, i, em, u, ins, s, strike, del, small, big, sub, sup, code, kbd, samp, tt, cite, var, dfn, br.
- Tableaux : table, caption, thead, tbody, tfoot, tr, th et td, avec colspan et rowspan. Les lignes d'en-tête sont répétées sur chaque page.
- Images : img avec des fichiers JPEG, PNG, BMP et TIFF (chemins locaux relatifs à ABasePath) et des URI data: en base64, avec width et height.
- Liens : a href devient une annotation de lien.
- Feuilles de style : blocs <style>, attributs style et <link rel="stylesheet"> vers des fichiers locaux.

CSS pris en charge

- Sélecteurs : élément, classe, id, combinateurs de descendance et d'enfant, :first-child, avec spécificité, héritage et !important.
- Modèle de boîte : margin, padding, border (largeur, style solid, dashed et dotted, couleur, par côté), width, max-width, height, background-color, border-collapse et border-spacing.
- Texte : color, font, font-family, font-size (y compris les mots-clés, em, rem, %, px, pt, mm, cm, in), font-weight, font-style, line-height, letter-spacing, text-align (y compris justify), text-indent, text-decoration, vertical-align, white-space.
- Listes et affichage : list-style, list-style-type, display (block, inline, list-item, table et ses parties, none).
- Pagination : page-break-before, page-break-after, page-break-inside, break-before, break-after, break-inside, et @page avec size (A3, A4, A5, B5, letter, legal, landscape) et margin. Les lignes, images et lignes de tableau ne sont jamais scindées entre les pages.
- Couleurs sous forme de noms, #rgb, #rrggbb, rgb() et rgba().

Non pris en charge : JavaScript, les floats, le positionnement absolute et fixed, flexbox, grid, les polices web (@font-face), les images chargées via HTTP, SVG et les contrôles de formulaire. Les propriétés non prises en charge sont ignorées.

Exemple : chaîne HTML vers PDF

```
procedure HTMLToPDF(const APDFFile: string);
var
  oHTML: TsgcPDFHTMLRenderer;
begin
  oHTML := TsgcPDFHTMLRenderer.Create(nil);
  try
    oHTML.PageWidth := 595;
    oHTML.PageHeight := 842;
    oHTML.FooterText := 'Page {page} of {pages}';
    oHTML.HTMLToPDFFile(
      '<html><head><style>' +
      'h1 { color: #003366; border-bottom: 1px solid #999 }' +
      'td, th { border: 1px solid #ccc; padding: 4px }' +
      '.total { font-weight: bold; text-align: right }' +
      '</style></head><body>' +
      '<h1>Invoice 2026-001</h1>' +
      '<p>Thank you for your order. See <a href="https://www.esegece.com">' +
      'our website</a>.</p>' +
      '<table><tr><th>Item</th><th>Amount</th></tr>' +
      '<tr><td>sgcPDF license</td><td class="total">299.00</td></tr></table>' +
      '<ul><li>Delivery by download</li><li>One year of updates</li></ul>' +
      '</body></html>', APDFFile);
  finally
    oHTML.Free;
  end;
end;
```

Exemple : ajouter du HTML à un document

```
function AppendHTML(ADocument: TsgcPDFDocument;
  const AHTMLFile: string): Integer;
var
  oHTML: TsgcPDFHTMLRenderer;
begin
  oHTML := TsgcPDFHTMLRenderer.Create(nil);
  try
    oHTML.Margins.Left := 50;
    oHTML.Margins.Right := 50;
    oHTML.DefaultFontName := 'Arial';
    oHTML.DefaultFontSize := 11;
    // relative image and stylesheet paths resolve from the HTML file folder
    Result := oHTML.RenderHTMLFile(AHTMLFile, ADocument);
  finally
    oHTML.Free;
  end;
end;
```

Voir aussi

- TsgcPDFCanvas
- Tableaux

Codes-barres 2D et GS1-128

TsgcPDFBarcodeGenerator (unité sgcPDF_Barcodes) dessine des symboles 2D et des codes-barres GS1 sous forme de graphismes vectoriels, en plus des codes-barres linéaires décrits dans Codes-barres. Le paramètre APage de chaque méthode Draw accepte un TsgcPDFPage ou son générateur de contenu (`Page.Content`). Tout autre objet déclenche une exception.

Symbologie	Méthode	Options
QR Code	DrawQRCode(APage; const AData: AnsiString; AX, AY, AModuleSize: Double; AErrorLevel: TsgcPDFQRErrorLevel; AMode: TsgcPDFQRMode; AVersion: Integer)	Versions 1 à 40 (0 choisit la plus petite qui convient ; une version fixe déclenche une exception si les données ne rentrent pas), correction d'erreur qrecL, qrecM, qrecQ, qrecH, modes qrmAuto, qrmNumeric, qrmAlphanumeric, qrmByte. Les surcharges plus courtes utilisent les valeurs par défaut.
DataMatrix ECC200	DrawDataMatrix(APage; const AData: AnsiString; AX, AY, AModuleSize: Double; AShape: TsgcPDFDataMatrixShape = dmsSquare)	24 tailles carrées et 6 rectangulaires, dmsSquare, dmsRectangle ou dmsAuto (le plus petit symbole quelle que soit la forme). Encodage ASCII et Base256.
PDF417	DrawPDF417(APage; const AData: AnsiString; AX, AY, AModuleWidth: Double; ASecurityLevel: Integer = -1; AColumns: Integer = 0)	Niveaux de sécurité 0 à 8 (-1 automatique), 1 à 30 colonnes de données (0 automatique), 3 à 90 lignes. Compactage texte, numérique et octet.
GS1-128	DrawBarcode(APage; bcGS1_128; const AData: AnsiString; AX, AY: Double [; const AOptions])	Identifiants d'application entre parenthèses, par exemple (01)09501101530003(17)261231. FNC1 est inséré et la clé de contrôle GTIN est vérifiée.

DataMatrix et PDF417 sont également disponibles via DrawBarcode avec bcDataMatrix et bcPDF417, en utilisant les champs QRVersion, QRErrorLevel, QRMode, DataMatrixShape, PDF417SecurityLevel, PDF417Columns et PDF417RowHeight de TsgcPDFBarcodeOptions. Les encodeurs `EncodeQRCodeMatrix`, `EncodeDataMatrix`, `EncodePDF417` et `EncodeGS1_128`, et `EncodeLinear` pour toute symbologie linéaire, renvoient la matrice de modules ou les largeurs de barres si vous voulez dessiner le symbole vous-même. TsgcPDFBarcodeSymbol fait ce travail pour toute symbologie : il ajoute la zone de silence, la ligne de texte et la rotation, et c'est le symbole qu'affichent les contrôles de code-barres VCL et FireMonkey.

Exemple

```
procedure Create2DBarcodes(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oPage: TsgcPDFPage;
  oBarcodes: TsgcPDFBarcodeGenerator;
  oOptions: TsgcPDFBarcodeOptions;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPage := oPDF.AddPage(595, 842);
    oBarcodes := TsgcPDFBarcodeGenerator.Create;
    try
      // QR Code: level H, alphanumeric mode, version 0 = smallest that fits
      oBarcodes.DrawQRCode(oPage, 'HTTPS://WWW.ESEGECE.COM', 50, 650, 3,
        qrecH, qrmAlphanumeric, 0);
      // DataMatrix ECC200, square or rectangular symbols
      oBarcodes.DrawDataMatrix(oPage, 'SGC-PDF-2026', 250, 700, 3);
      oBarcodes.DrawDataMatrix(oPage, 'SGC-PDF-2026', 350, 700, 3,
```

```
        dmsRectangle);
      // PDF417: security level 5, 4 data columns
      oBarcodes.DrawPDF417(oPage, 'sgcPDF PDF417 sample text', 50, 520, 1, 5, 4);
      // GS1-128: application identifiers in parentheses
      oOptions := sgcPDFDefaultBarcodeOptions;
      oOptions.Height := 40;
      oBarcodes.DrawBarcode(oPage, bcGS1_128, '(01)09501101530003(17)261231',
        50, 400, oOptions);
    finally
      oBarcodes.Free;
    end;
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Non implémenté : l'encodage DataMatrix C40 et Text, ainsi que la segmentation en mode mixte du QR Code. Le résultat reste valide, seulement moins compact pour certaines données.

Voir aussi

- Codes-barres
- TsgcPDFCanvas
- TsgcPDFBarcodeSymbol
- TsgcPDFBarcodeControl

TsgcPDFBarcodeControl

TsgcPDFBarcodeControl (unité sgcPDF_Barcode_VCL) est un contrôle VCL qui affiche un code-barres ou un symbole 2D sur une fiche. Il héberge un TsgcPDFBarcodeSymbol, qui encode le texte avec les encodeurs de TsgcPDFBarcodeGenerator, et dessine un rectangle plein par série de modules sombres, de sorte que chaque module commence et finit sur un pixel entier et que le symbole reste net pour les lecteurs. Il prend en charge les onze symbologies de sgcPDF : Code 128, Code 39, EAN-13, EAN-8, UPC-A, ITF, Codabar, GS1-128, QR Code, DataMatrix et PDF417.
Le contrôle dessine uniquement avec le canevas de la VCL, sans GDI+, et fonctionne donc dès Delphi 7. Il est installé par les paquets VCL, sur la page de palette **sgcPDF**. Le même symbole peut être enregistré en PNG ou en SVG, ou dessiné sur une page PDF avec les mêmes modules qu'à l'écran.

Propriétés publiées

Propriété	Description
Symbology: TsgcPDFBarcodeType	La symbologie : bcCode128 (par défaut), bcCode39, bcEAN13, bcEAN8, bcUPCA, bcITF, bcCodabar, bcGS1_128, ou les symbologies 2D bcQRCode, bcDataMatrix et bcPDF417.
Text: string	Les données à encoder. QR Code, DataMatrix et PDF417 encodent les octets UTF-8 du texte. Une symbologie linéaire n'accepte que son propre jeu de caractères, et GS1-128 attend les identifiants d'application entre parenthèses.
QuietZone: Integer	La marge claire en modules. -1 (par défaut) utilise la valeur normalisée de la symbologie : 10 pour les symboles linéaires, à gauche et à droite, et 4 pour QR Code, 1 pour DataMatrix et 2 pour PDF417, sur les quatre côtés.
BarHeight: Integer	La hauteur des barres d'un symbole linéaire en modules. 0 (par défaut) remplit la hauteur disponible.
ShowText: Boolean	Dessine la ligne lisible sous les barres d'un symbole linéaire. True par défaut. Les symboles 2D n'ont pas de ligne de texte.
Rotation: TsgcPDFBarcodeRotation	br0 (par défaut), br90, br180 ou br270 : quarts de tour dans le sens horaire. La ligne de texte tourne avec le symbole.
Fit: TsgcPDFBarcodeFit	bfFitInteger (par défaut) : le plus grand nombre entier de pixels par module qui tient, centré. bfModuleSize : exactement ModuleSize pixels par module, centré. bfStretch : le symbole remplit le contrôle et les modules peuvent différer d'un pixel.
ModuleSize: Integer	Pixels par module pour bfModuleSize. 2 par défaut. La valeur est mise à l'échelle avec la fiche sur les écrans haute densité.
Color: TColor	L'arrière-plan et les modules clairs. clWhite par défaut.
BarColor: TColor	Les modules sombres. clBlack par défaut.
Font, ParentFont	La police, le style et la couleur de la ligne de texte. Sa hauteur suit la taille du module (8 modules dans une bande de 12 modules sous les barres), donc Font.Size est ignoré. Une police TrueType est toujours utilisée.
Transparent: Boolean	Laisse l'arrière-plan non peint, de sorte que le parent apparaît entre les barres. False par défaut.

Propriété	Description
QRErrorLevel: TsgcPDFQRErrorLevel	Correction d'erreurs de QR Code : qrecL, qrecM (par défaut), qrecQ ou qrecH.
QRMode: TsgcPDFQRMode	Mode d'encodage de QR Code : qrmByte (par défaut), qrmAuto, qrmNumeric ou qrmAlphanumeric.
QRVersion: Integer	0 (par défaut) choisit la plus petite version de QR Code qui convient, de 1 à 40 fixe la version.
DataMatrixShape: TsgcPDFDataMatrixShape	dmsSquare (par défaut), dmsRectangle ou dmsAuto (le plus petit symbole de toute forme).
PDF417SecurityLevel: Integer	Niveau de sécurité PDF417 de 0 à 8, -1 (par défaut) est automatique.
PDF417Columns: Integer	Colonnes de données PDF417 de 1 à 30, 0 (par défaut) est automatique.
PDF417RowHeight: Integer	La hauteur d'une ligne PDF417 en modules. 3 par défaut.
Align, Anchors, Visible, Hint	Propriétés VCL standard.

Événements

Événement	Description
OnError: TsgcPDFBarcodeErrorEvent	Appelé avec Sender et aError (Symbol.LastError) lorsque le texte ne peut pas être encodé, une fois à chaque modification du symbole. Le contrôle dessine alors un fin cadre hachuré. Le dessin ne lève jamais d'exception.
OnClick: TNotifyEvent	Le contrôle a reçu un clic.

Méthodes publiques

Méthode	Description
IsValid: Boolean	True lorsque le texte peut être encodé. Symbol.LastError indique pourquoi il ne le peut pas.
PaintTo(aCanvas: TCanvas; const aRect: TRect)	Dessine le symbole dans aRect de n'importe quel canevas, par exemple un bitmap ou le canevas d'une imprimante, comme le contrôle se dessine lui-même : l'arrière-plan (sauf avec Transparent), les modules et la ligne de texte, ou le cadre hachuré lorsque le texte n'est pas valide.
SaveToPNG(const aFileName: string; aModuleSize: Integer), SaveToPNG(aStream: TStream; aModuleSize: Integer)	Enregistre le symbole sous forme d'image PNG avec aModuleSize pixels par module, BarColor sur Color. La ligne de texte utilise la police de Font et BarColor.
SaveToSVG(const aFileName: string; aModuleSize: Integer), SaveToSVG(aStream: TStream; aModuleSize: Integer)	Enregistre le symbole en SVG avec aModuleSize unités par module, avec les mêmes couleurs et la même police que SaveToPNG.
Symbol: TsgcPDFBarcodeSymbol	Le symbole hébergé, en lecture seule. Le contrôle le possède et utilise son OnChange. Servez-vous-en pour dessiner le symbole sur une page PDF (DrawToPDF) ou pour lire Runs, ModuleWidth et ModuleHeight.

SaveToPNG et SaveToSVG lèvent une exception avec Symbol.LastError lorsque le texte n'est pas valide, avant la création de tout fichier.

Exemple

```
procedure TMainForm.CreateBarcode;
begin
  FBarcode := TsgcPDFBarcodeControl.Create(Self);
  FBarcode.Parent := Self;
  FBarcode.SetBounds(16, 16, 320, 140);
  FBarcode.Symbology := bcEAN13;
  FBarcode.Text := '5901234123457';
  FBarcode.BarHeight := 60;
  FBarcode.BarColor := clNavy;
  FBarcode.Font.Name := 'Arial';
  FBarcode.OnError := OnBarcodeError;
end;

procedure TMainForm.OnBarcodeError(Sender: TObject; const aError: string);
begin
  // the control draws a hatched frame, the event tells why
  Caption := aError;
end;

procedure TMainForm.SaveBarcode(const aFileName: string);
begin
  if FBarcode.IsValid then
  begin
    // 4 pixels per module, BarColor on Color
    FBarcode.SaveToPNG(ChangeFileExt(aFileName, '.png'), 4);
    FBarcode.SaveToSVG(ChangeFileExt(aFileName, '.svg'), 4);
  end;
end;
```

Ajouter le contrôle à un PDF

TsgcPDFBarcodeSymbol.DrawToPDF dessine le symbole du contrôle sur une page PDF sous forme de rectangles vectoriels, avec les mêmes modules, la même rotation, la même zone de silence et la même ligne de texte qu'à l'écran. DrawToPDF attend les couleurs au format $RRGGBB, alors qu'un TColor de la VCL est au format $BBGGRR : échangez donc d'abord les octets du rouge et du bleu.

```
procedure AddBarcodeToPDF(aControl: TsgcPDFBarcodeControl;
  const aFileName: string);
var
  oPDF: TsgcPDFDocument;
  oPage: TsgcPDFPage;
  vFore, vBack: Cardinal;
begin
  // TColor is $BBGGRR and DrawToPDF takes $RRGGBB: swap red and blue
  vFore := Cardinal(ColorToRGB(aControl.BarColor));
  vFore := ((vFore and $FF) shl 16) or (vFore and $FF00) or
    ((vFore shr 16) and $FF);
  vBack := Cardinal(ColorToRGB(aControl.Color));
  vBack := ((vBack and $FF) shl 16) or (vBack and $FF00) or
    ((vBack shr 16) and $FF);
  oPDF := TsgcPDFDocument.Create;
  try
    oPage := oPDF.AddPage(595, 842);
    // the modules, rotation, quiet zone and text line of the control,
    // fitted in a 250 x 100 points box whose bottom left corner is 50, 680
    aControl.Symbol.DrawToPDF(oPage, 50, 680, 250, 100, vFore, vBack,
      sfHelvetica);
    oPDF.SaveToFile(aFileName);
  finally
    oPDF.Free;
  end;
end;
```

Voir aussi

- TsgcPDFBarcodeFMX
- TsgcPDFBarcodeSymbol
- Codes-barres
- Codes-barres 2D et GS1-128

TsgcPDFBarcodeFMX

TsgcPDFBarcodeFMX (unité sgcPDF_Barcode_FMX) est la version FireMonkey de TsgcPDFBarcodeControl. Il héberge le même TsgcPDFBarcodeSymbol et dessine un rectangle par série de modules sombres. La mise en page est calculée en pixels du périphérique, avec l'échelle du canevas, de sorte que chaque module commence et finit sur un pixel entier du périphérique et reste net sur les écrans haute densité. Le contrôle limite son dessin à ses bornes.
Il nécessite Delphi XE5 ou plus récent (avec des compilateurs plus anciens, l'unité est vide) et est installé par les paquets FMX, sur la page de palette **sgcPDF**.

Propriétés publiées

Propriété	Description
Symbology: TsgcPDFBarcodeType	La symbologie : bcCode128 (par défaut), bcCode39, bcEAN13, bcEAN8, bcUPCA, bcITF, bcCodabar, bcGS1_128, ou les symbologies 2D bcQRCode, bcDataMatrix et bcPDF417.
Text: string	Les données à encoder. QR Code, DataMatrix et PDF417 encodent les octets UTF-8 du texte. Une symbologie linéaire n'accepte que son propre jeu de caractères, et GS1-128 attend les identifiants d'application entre parenthèses.
QuietZone: Integer	La marge claire en modules. -1 (par défaut) utilise la valeur normalisée de la symbologie : 10 pour les symboles linéaires, à gauche et à droite, et 4 pour QR Code, 1 pour DataMatrix et 2 pour PDF417, sur les quatre côtés.
BarHeight: Integer	La hauteur des barres d'un symbole linéaire en modules. 0 (par défaut) remplit la hauteur disponible.
ShowText: Boolean	Dessine la ligne lisible sous les barres d'un symbole linéaire. True par défaut. Les symboles 2D n'ont pas de ligne de texte.
Rotation: TsgcPDFBarcodeRotation	br0 (par défaut), br90, br180 ou br270 : quarts de tour dans le sens horaire. La ligne de texte tourne avec le symbole.
Fit: TsgcPDFBarcodeFit	bfFitInteger (par défaut) : le plus grand nombre entier de pixels du périphérique par module qui tient, centré. bfModuleSize : ModuleSize unités logiques par module, arrondies à des pixels entiers du périphérique, centré. bfStretch : le symbole remplit le contrôle et les modules peuvent différer d'un pixel du périphérique.
ModuleSize: Integer	Unités logiques par module pour bfModuleSize, comme pour toute autre taille FireMonkey. 2 par défaut.
Color: TAlphaColor	L'arrière-plan et les modules clairs. Blanc par défaut.
BarColor: TAlphaColor	Les modules sombres. Noir par défaut.
TextSettings: TsgcPDFBarcodeTextSettingsFMX	Font (famille et style) et FontColor de la ligne de texte. Sa hauteur suit la taille du module, donc Font.Size est ignoré.
Transparent: Boolean	Laisse l'arrière-plan non peint. False par défaut.
QRErrorLevel: TsgcPDFQRErrorLevel	Correction d'erreurs de QR Code : qrecL, qrecM (par défaut), qrecQ ou qrecH.

Propriété	Description
QRMode: TsgcPDFQRMode	Mode d'encodage de QR Code : qrmByte (par défaut), qrmAuto, qrmNumeric ou qrmAlphanumeric.
QRVersion: Integer	0 (par défaut) choisit la plus petite version de QR Code qui convient, de 1 à 40 fixe la version.
DataMatrixShape: TsgcPDFDataMatrixShape	dmsSquare (par défaut), dmsRectangle ou dmsAuto (le plus petit symbole de toute forme).
PDF417SecurityLevel: Integer	Niveau de sécurité PDF417 de 0 à 8, -1 (par défaut) est automatique.
PDF417Columns: Integer	Colonnes de données PDF417 de 1 à 30, 0 (par défaut) est automatique.
PDF417RowHeight: Integer	La hauteur d'une ligne PDF417 en modules. 3 par défaut.
Align, Anchors, Margins, Position, Width, Height, Size, Visible, Hint, ShowHint	Propriétés FireMonkey standard. Size est publiée à partir de Delphi XE7, Hint et ShowHint à partir de Delphi XE6.

Événements

Événement	Description
OnError: TsgcPDFBarcodeErrorEvent	Appelé avec Sender et aError (Symbol.LastError) lorsque le texte ne peut pas être encodé, une fois à chaque modification du symbole. Le contrôle dessine alors un fin cadre hachuré. Le dessin ne lève jamais d'exception.
OnClick: TNotifyEvent	Le contrôle a reçu un clic ou une pression.

Méthodes publiques

Méthode	Description
IsValid: Boolean	True lorsque le texte peut être encodé. Symbol.LastError indique pourquoi il ne le peut pas.
SaveToPNG(const aFileName: string; aModuleSize: Integer), SaveToPNG(aStream: TStream; aModuleSize: Integer)	Enregistre le symbole sous forme d'image PNG avec aModuleSize pixels par module, BarColor sur Color. La ligne de texte utilise la famille de TextSettings.Font.
SaveToSVG(const aFileName: string; aModuleSize: Integer), SaveToSVG(aStream: TStream; aModuleSize: Integer)	Enregistre le symbole en SVG avec aModuleSize unités par module, avec les mêmes couleurs et la même police que SaveToPNG.
Symbol: TsgcPDFBarcodeSymbol	Le symbole hébergé, en lecture seule. Le contrôle le possède et utilise son OnChange.

SaveToPNG et SaveToSVG lèvent une exception avec Symbol.LastError lorsque le texte n'est pas valide, avant la création de tout fichier.

Couleurs

Un TAlphaColor est au format $AARRGGBB, et les méthodes de TsgcPDFBarcodeSymbol attendent $RRGGBB et ignorent l'octet de poids fort. Color et BarColor se passent donc tels quels à DrawToPDF, SaveToSVG et SaveToPNG du symbole, sans échange d'octets.

Exemple

```
procedure TMainForm.CreateQRCode;
begin
  FQRCode := TsgcPDFBarcodeFMX.Create(Self);
  FQRCode.Parent := Self;
  FQRCode.Align := TAlignLayout.Client;
  FQRCode.Symbology := bcQRCode;
  FQRCode.QRErrorLevel := qrecH;
  FQRCode.Text := 'https://www.esegece.com';
  FQRCode.BarColor := TAlphaColors.Navy;
  FQRCode.OnError := OnBarcodeError;
end;

procedure TMainForm.OnBarcodeError(Sender: TObject; const aError: string);
begin
  Caption := aError;
end;

procedure TMainForm.SaveQRCode;
begin
  // on mobile, write to the documents folder of the application
  if FQRCode.IsValid then
    FQRCode.SaveToPNG(TPath.Combine(TPath.GetDocumentsPath, 'qrcode.png'), 8);
end;

procedure TMainForm.AddQRCodeToPDF(const aFileName: string);
var
  oPDF: TsgcPDFDocument;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    // a TAlphaColor is $AARRGGBB and DrawToPDF ignores the top byte
    FQRCode.Symbol.DrawToPDF(oPDF.AddPage(595, 842), 50, 600, 200, 200,
      FQRCode.BarColor, FQRCode.Color);
    oPDF.SaveToFile(aFileName);
  finally
    oPDF.Free;
  end;
end;
```

Voir aussi

- TsgcPDFBarcodeControl
- TsgcPDFBarcodeSymbol
- macOS, iOS et Android

TsgcPDFBarcodeSymbol

TsgcPDFBarcodeSymbol (unité sgcPDF_Barcode_Symbol) est le code-barres indépendant du framework qui se trouve derrière TsgcPDFBarcodeControl et TsgcPDFBarcodeFMX. Il encode un code-barres avec les encodeurs de TsgcPDFBarcodeGenerator, le dispose sur une grille de modules et l'écrit sous forme de texte SVG, d'image PNG ou de rectangles vectoriels sur une page PDF. Il n'utilise ni VCL, ni FMX, ni GDI, donc il fait partie du paquet core et fonctionne aussi dans les services, les applications console et sous Linux 64 bits.

La grille de modules

- La grille contient tout le symbole après la rotation, zone de silence comprise et, pour une symbologie linéaire avec ShowText, la bande de la ligne lisible.
- La bande de texte mesure 12 modules de haut et se place sous les barres avant la rotation. Le texte mesure 8 modules de haut, sa ligne de base est à 10 modules du bord de la bande situé près des barres, et il est centré le long de la bande. Les caractères de contrôle s'affichent comme des espaces.
- Un symbole linéaire dont BarHeight vaut 0 compte 50 modules de barres dans ModuleHeight et dans la sortie SVG et PNG. Les contrôles et DrawToPDF font plutôt remplir la hauteur disponible à ses barres.
- Rotation fait tourner le symbole dans le sens horaire par quarts de tour.
- Les couleurs sont au format $RRGGBB et l'octet de poids fort est ignoré, donc un TAlphaColor de FireMonkey se passe tel quel, alors que les octets du rouge et du bleu d'un TColor de la VCL doivent être échangés.

Propriétés publiées

Propriété	Description
Symbology: TsgcPDFBarcodeType	La symbologie : bcCode128 (par défaut), bcCode39, bcEAN13, bcEAN8, bcUPCA, bcITF, bcCodabar, bcGS1_128, ou les symbologies 2D bcQRCode, bcDataMatrix et bcPDF417.
Text: string	Les données à encoder. QR Code, DataMatrix et PDF417 encodent les octets UTF-8 du texte. Une symbologie linéaire n'accepte que son propre jeu de caractères, et GS1-128 attend les identifiants d'application entre parenthèses.
QuietZone: Integer	La marge claire en modules. -1 (par défaut) utilise la valeur normalisée de la symbologie : 10 pour les symboles linéaires, à gauche et à droite, et 4 pour QR Code, 1 pour DataMatrix et 2 pour PDF417, sur les quatre côtés.
BarHeight: Integer	La hauteur des barres d'un symbole linéaire en modules. 0 (par défaut) remplit la hauteur disponible.
ShowText: Boolean	Dessine la ligne lisible sous les barres d'un symbole linéaire. True par défaut. Les symboles 2D n'ont pas de ligne de texte.
Rotation: TsgcPDFBarcodeRotation	br0 (par défaut), br90, br180 ou br270 : quarts de tour dans le sens horaire. La ligne de texte tourne avec le symbole.
QRErrorLevel: TsgcPDFQRErrorLevel	Correction d'erreurs de QR Code : qrecL, qrecM (par défaut), qrecQ ou qrecH.
QRMode: TsgcPDFQRMode	Mode d'encodage de QR Code : qrmByte (par défaut), qrmAuto, qrmNumeric ou qrmAlphanumeric.
QRVersion: Integer	0 (par défaut) choisit la plus petite version de QR Code qui convient, de 1 à 40 fixe la version.

Propriété	Description
DataMatrixShape: TsgcPDFDataMatrixShape	dmsSquare (par défaut), dmsRectangle ou dmsAuto (le plus petit symbole de toute forme).
PDF417SecurityLevel: Integer	Niveau de sécurité PDF417 de 0 à 8, -1 (par défaut) est automatique.
PDF417Columns: Integer	Colonnes de données PDF417 de 1 à 30, 0 (par défaut) est automatique.
PDF417RowHeight: Integer	La hauteur d'une ligne PDF417 en modules. 3 par défaut.
OnChange: TNotifyEvent	Une propriété a changé. Les contrôles de code-barres utilisent l'OnChange de leur Symbol pour se redessiner, ne le remplacez donc pas à cet endroit.

Méthodes et propriétés publiques

Membre	Description
Encode: Boolean	Encode le symbole lorsqu'une propriété a changé depuis le dernier appel et renvoie True lorsque le texte est valide. Les autres méthodes l'appellent au besoin.
LastError: string	La raison pour laquelle le texte ne peut pas être encodé.
ModuleWidth, ModuleHeight: Integer	La taille de la grille tournée en modules, zone de silence et bande de texte comprises. 0 lorsque le texte n'est pas valide.
ModuleAt(aCol, aRow: Integer): Boolean	True lorsque le module en aCol, aRow de la grille tournée est sombre.
Runs: TsgcPDFBarcodeRuns	Les modules sombres sous forme de rectangles TsgcPDFBarcodeRun (Col, Row, Cols, Rows), fusionnés par ligne puis entre lignes, triés par ligne et par colonne. Vide lorsque le texte n'est pas valide.
TextBounds(out aCol, aRow, aCols, aRows: Integer): Boolean	La bande de la ligne de texte sur la grille tournée. False lorsque le symbole ne dessine aucun texte : symbologies 2D, ShowText désactivé ou texte non valide.
FitModuleSize(aWidth, aHeight: Integer): Integer	La plus grande taille entière de module avec laquelle le symbole tient dans aWidth par aHeight pixels, 0 lorsqu'un pixel par module ne tient pas. Pour un symbole linéaire avec BarHeight 0, seules la bande de texte et un module de barres comptent en travers des barres, car les barres remplissent la hauteur.
LayoutAxes(aFit: TsgcPDFBarcodeFit; aModuleSize, aLeft, aTop, aWidth, aHeight: Integer; out aX, aY: TsgcPDFBarcodeAxis)	La disposition complète en pixels de la grille dans un cadre, celle avec laquelle les contrôles dessinent. Voir Dessin personnalisé.
SaveToSVG(const aFileName: string; aModuleSize: Integer; aFore, aBack: Cardinal; const aFontName: string), SaveToSVG(aStream: TStream; ...)	Écrit le symbole en SVG avec aModuleSize unités par module, aFore sur aBack. La ligne de texte utilise la famille de polices aFontName.
SaveToPNG(const aFileName: string; aModuleSize: Integer; aFore, aBack: Cardinal; const aFontName:	Écrit le symbole sous forme d'image PNG avec aModuleSize pixels par module, via le canevas raster en Pascal pur. La ligne de texte utilise la police TrueType du système d'exploitation aFontName, ou une

Membre	Description
string), SaveToPNG(aStream: TStream; ...)	police sans empattement si elle n'est pas installée. Sans police utilisable, le PNG n'a pas de ligne de texte.
DrawToPDF(aPage: TObject; aX, aY, aWidth, aHeight: Double; aFore, aBack: Cardinal; aFont: TsgcPDFStandardFont = sfHelvetica)	Dessine le symbole sur une page PDF. Voir Dessiner sur une page PDF.
Assign(Source: TPersistent)	Copie les propriétés publiées d'un autre symbole.

SaveToSVG et SaveToPNG lèvent une exception lorsque aModuleSize est inférieur à 1, et avec LastError lorsque le texte n'est pas valide, avant la création de tout fichier.

Dessiner sur une page PDF

DrawToPDF dessine le symbole sous forme de graphismes vectoriels, de sorte qu'une page PDF montre exactement ce que montre un contrôle de code-barres. Il remplace la copie manuelle des propriétés d'un contrôle vers TsgcPDFBarcodeOptions : la rotation, la zone de silence, la hauteur des barres et la ligne de texte proviennent du symbole. TsgcPDFBarcodeGenerator.DrawBarcode reste la façon la plus courte de dessiner un code-barres directement à partir de données.

Paramètre	Description
aPage: TObject	Un TsgcPDFPage ou son générateur de contenu, `TsgcPDFPage.Content`, comme pour DrawBarcode. Tout autre objet lève une exception.
aX, aY: Double	Le coin inférieur gauche du cadre en points. L'axe y du PDF est dirigé vers le haut.
aWidth, aHeight: Double	La taille du cadre en points. Un cadre vide lève une exception.
aFore: Cardinal	La couleur des barres et de la ligne de texte, $RRGGBB.
aBack: Cardinal	La couleur peinte sous tout le symbole, zone de silence comprise, $RRGGBB.
aFont: TsgcPDFStandardFont	La police PDF standard de la ligne de texte. sfHelvetica par défaut.

- Taille : le symbole prend la plus grande taille de module qui tient dans le cadre, pas seulement en points entiers, et y est centré. Un symbole linéaire avec BarHeight 0 remplit le cadre en travers des barres, comme FitModuleSize et LayoutAxes.
- Sortie : le rectangle aBack, puis un rectangle plein par entrée de Runs en aFore, de sorte que chaque bord de module tombe exactement sur la grille, puis la ligne de texte, le tout dans un état graphique sauvegardé.
- Texte : la ligne de texte est dessinée dans aFont avec une taille de 8 modules, sa ligne de base à 10 modules du bord de la bande situé près des barres, centrée sur la bande, et elle tourne dans le sens horaire avec le symbole.
- Erreurs : un texte non valide lève une exception avec LastError et ne dessine rien.

Exemple

Une étiquette GS1-128 dessinée deux fois sur une page PDF, la seconde fois tournée d'un quart de tour, et enregistrée en SVG et en PNG, sans fiche et sans contrôle :

```
procedure CreateShippingLabel(const aFileName: string);
var
  oSymbol: TsgcPDFBarcodeSymbol;
  oPDF: TsgcPDFDocument;
  oPage: TsgcPDFPage;
```

```
begin
  oSymbol := TsgcPDFBarcodeSymbol.Create;
  try
    oSymbol.Symbology := bcGS1_128;
    oSymbol.Text := '(01)09501101530003(17)261231';
    oSymbol.BarHeight := 30;
    if not oSymbol.Encode then
      raise Exception.Create(oSymbol.LastError);
    oPDF := TsgcPDFDocument.Create;
    try
      oPage := oPDF.AddPage(595, 842);
      // black on white in a 300 x 90 points box, bottom left corner 50, 700
      oSymbol.DrawToPDF(oPage, 50, 700, 300, 90, $000000, $FFFFFF);
      // a quarter turn clockwise, navy, the text line in Courier
      oSymbol.Rotation := br90;
      oSymbol.DrawToPDF(oPage.Content, 450, 350, 90, 300, $000080, $FFFFFF,
        sfCourier);
      oPDF.SaveToFile(aFileName);
    finally
      oPDF.Free;
    end;
    // the same symbol as SVG (2 units per module) and PNG (3 pixels)
    oSymbol.Rotation := br0;
    oSymbol.SaveToSVG(ChangeFileExt(aFileName, '.svg'), 2, $000000, $FFFFFF,
      'Arial');
    oSymbol.SaveToPNG(ChangeFileExt(aFileName, '.png'), 3, $000000, $FFFFFF,
      'Arial');
  finally
    oSymbol.Free;
  end;
end;
```

Dessin personnalisé

LayoutAxes calcule la disposition complète en pixels avec laquelle les contrôles dessinent, pour un cadre de aWidth par aHeight pixels dont le coin supérieur gauche est en aLeft, aTop. aFit fixe la taille du module (aModuleSize pixels avec bfModuleSize), le symbole est centré et un symbole linéaire avec BarHeight 0 remplit le cadre en travers des barres. Elle renvoie un TsgcPDFBarcodeAxis par axe, et `sgcPDFBarcodeAxisPos(aAxis, aIndex)` renvoie le pixel de la ligne de grille aIndex (de 0 à Count). Les séries voisines partagent leurs bords, donc aucun espace n'apparaît entre les modules.

```
procedure PaintSymbol(aSymbol: TsgcPDFBarcodeSymbol; aCanvas: TCanvas;
  const aRect: TRect);
var
  vX, vY: TsgcPDFBarcodeAxis;
  vRuns: TsgcPDFBarcodeRuns;
  i: Integer;
begin
  if not aSymbol.Encode then
    Exit;
  // the largest whole module size that fits, centred in aRect
  aSymbol.LayoutAxes(bfFitInteger, 0, aRect.Left, aRect.Top,
    aRect.Right - aRect.Left, aRect.Bottom - aRect.Top, vX, vY);
  aCanvas.Brush.Style := bsSolid;
  aCanvas.Brush.Color := clBlack;
  vRuns := aSymbol.Runs;
  for i := 0 to Length(vRuns) - 1 do
    aCanvas.FillRect(Rect(sgcPDFBarcodeAxisPos(vX, vRuns[i].Col),
      sgcPDFBarcodeAxisPos(vY, vRuns[i].Row),
      sgcPDFBarcodeAxisPos(vX, vRuns[i].Col + vRuns[i].Cols),
      sgcPDFBarcodeAxisPos(vY, vRuns[i].Row + vRuns[i].Rows)));
end;
```

Types et constantes

Nom	Description
TsgcPDFBarcodeRotation	br0, br90, br180, br270 : quarts de tour dans le sens horaire.
TsgcPDFBarcodeFit	bfFitInteger, bfModuleSize, bfStretch : la façon dont les contrôles et LayoutAxes dimensionnent les modules.
TsgcPDFBarcodeRun, TsgcPDFBarcodeRuns	Un rectangle de modules sombres sur la grille tournée (Col, Row, Cols, Rows) et le tableau que renvoie Runs.
TsgcPDFBarcodeAxis	La façon dont les lignes de grille d'un axe correspondent à des pixels entiers : Origin, Module (pixels par module) et Count (modules) ; avec Module 0 (bfStretch), Span pixels sont répartis sur Count modules. FillStart, FillCount et FillPixels décrivent la bande qui remplit le cadre, les barres d'un symbole linéaire avec BarHeight 0.
sgcPDFBarcodeAxisPos(const aAxis: TsgcPDFBarcodeAxis; aIndex: Integer): Integer	Le pixel de la ligne de grille aIndex sur un axe de LayoutAxes.
TsgcPDFBarcodeErrorEvent	`procedure(Sender: TObject; const aError: string) of object` , l'événement OnError des deux contrôles de code-barres.
CS_SGC_PDF_BARCODE_TEXT_BAND, CS_SGC_PDF_BARCODE_TEXT_SIZE, CS_SGC_PDF_BARCODE_TEXT_BASELINE	La ligne de texte en modules : hauteur de bande 12, taille de police 8, ligne de base 10.
CS_SGC_PDF_BARCODE_BAR_HEIGHT	50, la hauteur des barres en modules d'un symbole linéaire avec BarHeight 0 dans ModuleHeight, SVG et PNG.

Voir aussi

- TsgcPDFBarcodeControl
- TsgcPDFBarcodeFMX
- Codes-barres
- Codes-barres 2D et GS1-128

Linux

Le noyau de sgcPDF se compile et fonctionne sous Linux 64 bits (Delphi 10.2 Tokyo et plus récent avec la plate-forme Linux64). Cela couvre le travail serveur habituel : création, modification, fusion, remplissage, chiffrement et lecture de documents PDF, et rendu de pages en images, par exemple dans un service web ou une tâche par lots.

Disponible sous Linux

- TsgcPDFDocument, TsgcPDFEditor ainsi que l'analyseur et l'écrivain, y compris les enregistrements incrémentaux, les flux d'objets compressés et les fichiers de plus de 2 Go.
- Texte, polices TrueType et OpenType (chargées depuis des fichiers avec AddTrueTypeFontFromFile, ou depuis les dossiers de polices système), mise en forme, tableaux, codes-barres, annotations, signets, calques, formulaires, pièces jointes, PDF balisé, sortie et validation PDF/A.
- Chiffrement par mot de passe (RC4 et AES), caviardage, l'optimiseur.
- Extraction de texte avec positions des caractères, TsgcPDFImageExtractor et tous les décodeurs d'images.
- TsgcPDFRendererRaster : page vers PNG ou BMP, et pixels de page.
- TsgcPDFCanvas et TsgcPDFHTMLRenderer.
- Signatures numériques via l'événement OnExternalSign (par exemple avec sgcSign ou votre propre HSM), voir Signatures numériques.

Windows uniquement

- TsgcPDFRendererGDI, TsgcPDFViewer, TsgcPDFExporter et les contrôles VCL TsgcPDFViewerControl, TsgcPDFThumbnailView et TsgcPDFOutlineView.
- TsgcPDFEMFPlayer et TsgcPDFVCLCanvasHelper.
- La signature et la vérification avec des certificats provenant de fichiers PFX ou du magasin de certificats Windows, ainsi que le chiffrement par certificat (clé publique), qui utilisent la CryptoAPI de Windows. Sous Linux, ces appels déclenchent une exception.
- Les classes CJK de bas niveau de sgcPDF_Fonts_CJK (le texte CJK lui-même fonctionne via AddTrueTypeFontFromFile).

Polices sous Linux

`AddTrueTypeFont` recherche les polices installées dans /usr/share/fonts et /usr/local/share/fonts. Le moteur de rendu raster analyse aussi ~/.fonts et ~/.local/share/fonts, ainsi que d'autres dossiers ajoutés avec `sgcPDFRasterAddFontDirectory`. Les serveurs et conteneurs minimaux n'ont souvent aucune police du tout : installez un paquet tel que fonts-dejavu-core, fonts-liberation ou fonts-noto, ou déployez vos propres polices avec l'application et utilisez `AddTrueTypeFontFromFile`. Les documents avec des polices intégrées s'affichent correctement sans aucune police installée.

Exemple : vignettes côté serveur

Le même code que sous Windows :

```
procedure RenderPageToPNG(const APDFFile, APNGFile: string; ADPI: Integer);
var
  oPDF: TsgcPDFDocument;
  oRenderer: TsgcPDFRendererRaster;
  oPage: TsgcPDFPage;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.LoadFromFile(APDFFile);
    oPage := oPDF.Pages[0];
    oRenderer := TsgcPDFRendererRaster.Create;
    try
```

```
      // page size in points (1/72 inch) to pixels
      oRenderer.RenderPageToPNG(oPDF.Parser, oPage.PageDict,
        Round(oPage.Width * ADPI / 72), Round(oPage.Height * ADPI / 72),
        APNGFile);
    finally
      oRenderer.Free;
    end;
  finally
    oPDF.Free;
  end;
end;
```

Voir aussi

- macOS, iOS et Android
- TsgcPDFRendererRaster
- Installation

macOS, iOS et Android

À partir de Delphi XE5, sgcPDF peut être utilisé dans des applications FireMonkey pour macOS, iOS et Android, ainsi que pour Windows et Linux (FMXLinux). Les mêmes unités non Windows que sous Linux sont disponibles, plus le lecteur FireMonkey.

Fonctionnalité	Unités	Remarques
Créer, modifier, lire et extraire	Unités du noyau sgcPDF	Comme sous Linux. Chargez les polices depuis des fichiers déployés avec l'application (AddTrueTypeFontFromFile), car les plateformes mobiles n'exposent pas leurs polices système sous forme de fichiers de manière portable.
Contrôle de visualisation	sgcPDF_ViewerFMX	TsgcPDFViewerFMX : défilement, zoom par pincement, ajustement à la largeur ou à la page, liens.
Images de page	sgcPDF_Renderer_FMX, sgcPDF_Renderer_Raster	TsgcPDFRendererFMX effectue le rendu dans un TBitmap FMX, TsgcPDFRendererRaster en PNG, BMP ou pixels.
Signatures	sgcPDF_Signatures	Via OnExternalSign uniquement (pas de CryptoAPI).

Installation

Installez les paquets FMX (sgcPDFFMXDxx et dclsgcPDFFMXDxx) en plus des paquets du noyau pour obtenir TsgcPDFViewerFMX dans la palette. Pour les cibles mobiles, l'IDE compile les unités du dossier Source dans votre application : ajoutez le dossier Source de sgcPDF au chemin de la bibliothèque de chaque plateforme (Android 32 et 64 bits, iOS Device 64 bits, macOS 64 bits et ARM 64 bits, Linux 64 bits).

Déployer des fichiers

Utilisez le gestionnaire de déploiement pour ajouter des fichiers PDF et des polices à l'application, et ouvrez-les depuis le dossier des documents, par exemple avec `TPath.Combine(TPath.GetDocumentsPath, 'sample.pdf')` (unité System.IOUtils).

Voir aussi

- TsgcPDFViewerFMX
- Linux
- Architecture du moteur de rendu

C++Builder

sgcPDF prend en charge C++Builder 2007 à C++Builder 13 avec des paquets d'exécution et de conception, sur la plateforme Win32 pour toutes les versions et sur Win64 à partir de C++Builder XE6 (y compris la plateforme Win64 Modern de C++Builder 13).

Paquets

Le dossier Packages contient, pour chaque version de C++Builder, le paquet d'exécution du noyau **sgcPDFCxx.cbproj**, le paquet d'exécution VCL **sgcPDFVCLCxx.cbproj** et le paquet de conception **dclsgcPDFCxx.cbproj**, par exemple sgcPDFC13.cbproj, sgcPDFVCLC13.cbproj et dclsgcPDFC13.cbproj. Générez les deux paquets d'exécution, puis générez et installez le paquet de conception. La génération produit l'en-tête .hpp de chaque unité et les fichiers d'import .lib ou .a.

Utiliser sgcPDF depuis C++

- Incluez l'en-tête de chaque unité que vous utilisez, par exemple sgcPDF_Document.hpp, sgcPDF_Fonts.hpp et sgcPDF_Renderer_Raster.hpp.
- Les classes et fonctions portent les mêmes noms qu'en Delphi, dans l'espace de noms de leur unité (par exemple `Sgcpdf_document::TsgcPDFDocument`). La plupart des en-têtes sont aussi importés dans l'espace de noms global par les directives using générées.
- Créez les composants avec `new` et libérez-les avec `delete`. Les composants qui descendent de TComponent prennent un propriétaire dans le constructeur (passez NULL pour aucun).
- Les paramètres WideString et AnsiString acceptent les classes de chaînes C++Builder du même nom, TBytes est `System::DynamicArray<System::Byte>`.

Les en-têtes annulent la définition des macros Windows qui entrent en collision avec des noms de méthodes (DrawText, FindText, GetObject, GetCharWidth, CreateWindow, CertNameToStr), de sorte que des méthodes telles que `TsgcPDFTextExtractor::FindText` peuvent être appelées normalement.

Voir aussi

- Installation

Pièces jointes

Un PDF peut transporter d'autres fichiers en son sein : les données source d'un rapport, une feuille de calcul, le XML d'une facture. **TsgcPDFDocument.Attachments** (classe TsgcPDFAttachments, unité sgcPDF_Attachments) ajoute, liste, extrait et supprime ces fichiers intégrés. Ils sont écrits dans l'arbre de noms `/EmbeddedFiles` du catalogue et, avec leur relation, dans le tableau `/AF` du catalogue exigé par PDF/A-3.

Ajout de fichiers

```
procedure CreateWithAttachments(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oFile: TsgcPDFAttachment;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.AddPage(595, 842).DrawText('Report with attachments', 50, 780,
      sfHelveticaBold, 18);
    // from bytes: name, data, MIME type, description, relationship
    oPDF.Attachments.AddFromBytes('data.csv', StrBytes('id;qty'#13#10'1;10'),
      'text/csv', 'Source data of the report', afrSource);
    // from a file on disk, the file name becomes the attachment name
    oFile := oPDF.Attachments.AddFromFile(Dir + 'notes.txt', 'text/plain',
      'Notes');
    oFile.Relationship := afrSupplement;
    // the viewer opens the attachments panel
    oPDF.Attachments.ShowAttachmentsPanel := True;
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Méthode	Description
AddFromBytes(AName; AData; AMimeType; ADescription; ARelationship)	Intègre AData sous le nom AName. Seuls AName et AData sont obligatoires.
AddFromStream(AName; AStream; ...)	Intègre le reste de AStream.
AddFromFile(AFileName; AMimeType; ADescription; ARelationship)	Intègre un fichier depuis le disque. La pièce jointe prend le nom du fichier.
Delete(AIndex), Clear, IndexOf(AName)	Supprime des pièces jointes et en recherche une par son nom.

Les noms doivent être uniques : ajouter un second fichier portant le même nom déclenche une exception. Chaque pièce jointe stocke sa taille, ses dates de création et de modification et une somme de contrôle MD5, comme le recommande la spécification.

Relationship (AFRelationship)

ARelationship, ou la propriété `Relationship` de TsgcPDFAttachment, indique à un lecteur comment le fichier se rapporte au document (PDF 2.0 et PDF/A-3) :

Valeur	Signification
afrSource	La source d'origine du contenu du document, par exemple la feuille de calcul à partir de laquelle un rapport a été créé.
afrData	Données utilisées pour construire le document, par exemple les valeurs d'un graphique.
afrAlternative	Une représentation alternative du contenu, par exemple le XML d'une facture électronique.
afrSupplement	Une représentation supplémentaire, par exemple une copie exploitable par une machine.
afrEncryptedPayload	Un document de charge utile chiffré.
afrFormData	Données associées aux champs de formulaire du document.
afrSchema	Une définition de schéma d'un objet associé.
afrUnspecified	Aucune relation indiquée (par défaut).

Lecture et extraction

Les documents chargés remplissent automatiquement la liste Attachments. Chaque TsgcPDFAttachment expose Name, Description, MimeType, Relationship, CreationDate, ModDate et Size, et renvoie ses données avec `GetBytes`, `SaveToStream` ou `SaveToFile`.

```
procedure ListAttachments(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  i: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.LoadFromFile(AFileName);
    for i := 0 to oPDF.Attachments.Count - 1 do
      WriteLn(string(oPDF.Attachments[i].Name), ' ',
        oPDF.Attachments[i].Size, ' bytes ',
        string(oPDF.Attachments[i].MimeType));
    i := oPDF.Attachments.IndexOf('data.csv');
    if i >= 0 then
      oPDF.Attachments[i].SaveToFile(Dir + 'extracted.csv');
    // remove one attachment and save a copy without it
    oPDF.Attachments.Delete(oPDF.Attachments.IndexOf('notes.txt'));
    oPDF.SaveToFile(Dir + 'attachments_deleted.pdf');
  finally
    oPDF.Free;
  end;
end;
```

Pièces jointes et PDF/A

Niveau	Règle
PDF/A-1b	Les fichiers intégrés ne sont pas autorisés : en ajouter un déclenche une exception.

Niveau	Règle
PDF/A-2b	Seuls les fichiers qui sont eux-mêmes des documents PDF/A. Définissez `Attachments.PDFACompliantFiles` sur True pour le confirmer avant de les ajouter, sinon l'ajout déclenche une exception.
PDF/A-3b	Tout type de fichier, chacun avec son AFRelationship dans le tableau /AF du catalogue.

Définissez `ShowAttachmentsPanel` sur True pour écrire `/PageMode /UseAttachments`, afin que le lecteur ouvre son panneau de pièces jointes. Les pièces jointes sont conservées par les documents chiffrés et par SaveIncremental.

Voir aussi

- Factures Factur-X et ZUGFeRD
- PDF/A
- TsgcPDFDocument

Factures Factur-X et ZUGFeRD

Factur-X (France) et ZUGFeRD 2 (Allemagne) sont le même format hybride de facture électronique : un document PDF/A-3 lisible par une personne, avec la facture structurée (XML UN/CEFACT Cross Industry Invoice) intégrée pour le logiciel de comptabilité. XRechnung utilise le même conteneur avec son propre nom de fichier.
sgcPDF construit le conteneur PDF. Le XML de la facture lui-même est créé par votre application (ou votre ERP) et transmis sous forme d'octets à **Attachments.AddFacturX** :

```
procedure CreateFacturX(const AInvoiceXML, AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oXML: TMemoryStream;
  vXML: TBytes;
  vFont: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  oXML := TMemoryStream.Create;
  try
    // the invoice XML (CII) is created by your application
    oXML.LoadFromFile(AInvoiceXML);
    SetLength(vXML, oXML.Size);
    Move(oXML.Memory^, vXML[0], oXML.Size);
    oPDF.Info.Title := 'Invoice 2026-001';
    // PDF/A requires embedded fonts
    vFont := oPDF.AddTrueTypeFont('Arial', False, False);
    oPDF.AddPage(595, 842).DrawTextTT('Invoice 2026-001', 50, 780, vFont, 18);
    // embeds factur-x.xml with the relationship of the profile, adds the
    // Factur-X XMP extension schema and saves the document as PDF/A-3b
    oPDF.Attachments.AddFacturX(vXML, fxpEN16931);
    oPDF.SaveToFile(AFileName);
  finally
    oXML.Free;
    oPDF.Free;
  end;
end;
```

Ce que fait AddFacturX

- Intègre le XML sous le nom **factur-x.xml** (**xrechnung.xml** pour fxpXRechnung, ou le AFileName que vous passez) avec le type MIME text/xml.
- Définit la relation exigée par le profil : **Data** pour MINIMUM et BASIC WL, qui ne sont pas des factures complètes, et **Alternative** pour BASIC, EN 16931, EXTENDED et XRECHNUNG.
- Ajoute le schéma d'extension XMP Factur-X (fx:DocumentType, fx:DocumentFileName, fx:Version, fx:ConformanceLevel) aux métadonnées.
- Fait passer un document sans niveau PDF/A au niveau **PDF/A-3b**. Un document déjà défini en PDF/A-1b ou PDF/A-2b déclenche une exception, car Factur-X exige PDF/A-3.
- Remplace un XML de facture précédent : un document contient une seule facture.

Profile (TsgcPDFFacturXProfile)	ConformanceLevel	Relationship
fxpMinimum	MINIMUM	Data
fxpBasicWL	BASIC WL	Data
fxpBasic	BASIC	Alternative
fxpEN16931	EN 16931	Alternative
fxpExtended	EXTENDED	Alternative

Profile (TsgcPDFFacturXProfile)	ConformanceLevel	Relationship
fxpXRechnung	XRECHNUNG	Alternative

Le paramètre optionnel AVersion définit fx:Version (1.0 par défaut). Les règles PDF/A s'appliquent à tout le document, dessinez donc le texte avec des polices TrueType intégrées (AddTrueTypeFont et DrawTextTT).
Les documents chargés indiquent la facture dans `Attachments.FacturX`, `FacturXProfile` et `FacturXFileName`.
L'exemple ci-dessus a été vérifié avec veraPDF (PDF/A-3b : PASS) et la facture a été extraite sans modification par la bibliothèque Python factur-x.

Voir aussi

- Pièces jointes
- PDF/A

Enregistrement incrémental et sortie compacte

SaveIncremental

TsgcPDFDocument.SaveIncremental met à jour un PDF chargé en ajoutant une nouvelle révision : les octets d'origine sont écrits sans modification, suivis uniquement des objets nouveaux et modifiés, d'une section de références croisées du même type que celle d'origine (table ou flux) et d'un trailer avec `/Prev`. Les entrées `/ID` et `/Encrypt` sont conservées.
Utilisez-le pour modifier des documents signés sans casser leurs signatures, pour ajouter rapidement des annotations ou des valeurs de formulaire à de gros fichiers, et pour conserver l'historique d'un document.

```
procedure AppendChanges(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.LoadFromFile(AFileName);
    oPDF.Info.Title := 'Reviewed';
    oPDF.Annotations.AddText(0, 300, 700, 320, 720, 'Checked by QA');
    // the original bytes are kept, only the changes are appended
    oPDF.SaveIncremental(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Il existe deux surcharges : `SaveIncremental(AStream: TStream)` écrit le fichier mis à jour complet vers un flux, et `SaveIncremental(AFileName: string)` écrit un fichier, qui peut être le fichier depuis lequel le document a été chargé. Un document sans modification produit les octets d'origine.
Les modifications d'annotations sont enregistrées de la même manière. Une annotation modifiée du fichier est écrite de nouveau sous son propre numéro d'objet, et les nouvelles annotations et réponses sont ajoutées dans la nouvelle révision, de sorte que la plage d'octets de chaque signature existante reste la même. Un validateur accepte la révision lorsque la signature autorise les commentaires : une signature de certification avec la permission DocMDP P=3, ou uniquement des signatures d'approbation. Avec P=1 ou P=2, il signale une modification non autorisée. Voir Annotations.

Limites

SaveIncremental déclenche une exception, et un enregistrement complet (SaveToFile ou SaveToStream) est nécessaire, lorsque :

- le document n'a pas été chargé depuis un PDF (nouveau document) ;
- d'autres documents ont été ajoutés avec AppendDocument ;
- de nouveaux calques ou de nouveaux champs de formulaire ont été ajoutés (remplir des champs existants ne pose pas de problème) ;
- le chiffrement a été modifié ;
- le document a été caviardé après avoir été ouvert avec un mot de passe (voir TsgcPDFRedactor).

CompressObjects

Avec **CompressObjects** défini sur True, un enregistrement complet écrit les objets qui ne sont pas des flux dans des flux d'objets compressés et les références croisées sous forme de flux de références croisées compressé (PDF 1.5). Les documents comportant de nombreux petits objets (pages, annotations, champs de formulaire, arbres de structure) sont 39 à 57 % plus petits dans les tests. Le chiffrement est pris en charge. La propriété est ignorée pour PDF/A-1, qui n'autorise pas les flux d'objets.

```
procedure SaveCompact(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  i: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    for i := 1 to 50 do
      oPDF.AddPage(595, 842).DrawText(AnsiString('Page ' + IntToStr(i)), 50,
        780, sfHelvetica, 12);
    // object streams and a compressed cross-reference stream (PDF 1.5)
    oPDF.CompressObjects := True;
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Voir aussi

- Fichiers volumineux
- Horodatage et validation à long terme
- Optimisation des fichiers PDF
- Annotations

Fichiers volumineux

sgcPDF utilise partout des décalages de fichier 64 bits, il lit et écrit donc des fichiers de plus de 2 Go, et n'a pas besoin de charger un gros fichier en mémoire pour le traiter.

MemoryThreshold

TsgcPDFDocument.LoadFromFile charge en mémoire les fichiers jusqu'à **MemoryThreshold** octets (64 Mo par défaut). Un fichier plus volumineux est ouvert en lecture partagée et ses objets sont lus à la demande via un cache de lecture, de sorte qu'ouvrir un fichier de 3 Go ne lit que ses références croisées et les objets que vous utilisez. Définissez MemoryThreshold sur 0 pour lire chaque fichier à la demande.

```
procedure OpenLargeFile(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    // files larger than MemoryThreshold (default 64 MB) are read on demand
    // from the file instead of being loaded into memory
    oPDF.MemoryThreshold := 16 * 1024 * 1024;
    oPDF.LoadFromFile(AFileName);
    WriteLn('Pages: ', oPDF.PageCount, ', read on demand: ',
      oPDF.Parser.StreamBacked);
    WriteLn(string(oPDF.ExtractTextW(0)));
  finally
    oPDF.Free;
  end;
end;
```

Comportement

- `Parser.StreamBacked` vaut True lorsque le document est lu à la demande, et `Parser.Size` est la taille du fichier sous forme d'Int64.
- Le fichier reste ouvert jusqu'à ce qu'un autre document soit chargé ou que le document soit libéré. D'autres processus peuvent le lire mais pas le supprimer ni l'écraser entre-temps.
- SaveIncremental fonctionne sur un document lu à la demande et peut écrire dans le même fichier : les octets d'origine sont copiés par blocs, le fichier est remplacé et l'analyseur rouvre le nouveau fichier.
- LoadFromStream et LoadFromBytes chargent toujours en mémoire. Pour lire un flux à la demande, utilisez directement l'analyseur : `TsgcPDFParser.LoadFromStream(AStream, AOwnsStream)`. Le flux doit rester valide et inchangé pendant l'utilisation de l'analyseur.
- Un processus 32 bits ne peut pas contenir 2 Go dans un seul tableau : charger une telle source en mémoire déclenche une exception qui demande une lecture à la demande.

Voir aussi

- Enregistrement incrémental et sortie compacte
- Gestion des erreurs et entrées malveillantes

FDF et XFDF

Les fichiers FDF (ISO 32000, 12.7.8) et XFDF (ISO 19444-1, XML) ne contiennent que les valeurs des champs de formulaire d'un PDF. Utilisez-les pour envoyer des données remplies à un serveur, pré-remplir un formulaire depuis une base de données ou transférer des valeurs entre deux versions d'un formulaire. TsgcPDFAcroForm (`TsgcPDFDocument.AcroForm`) exporte et importe les deux formats.

Exportation

```
procedure ExportFormData(const APDFFile, AFDFFile, AXFDFFile: string);
var
  oPDF: TsgcPDFDocument;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.LoadFromFile(APDFFile);
    // the optional second parameter names the PDF the data belongs to
    oPDF.AcroForm.ExportFDF(AFDFFile, ExtractFileName(APDFFile));
    oPDF.AcroForm.ExportXFDF(AXFDFFile, ExtractFileName(APDFFile));
  finally
    oPDF.Free;
  end;
end;
```

Méthode	Description
ExportFDF(AStream or AFile-Name; ASourceFile)	Écrit un fichier FDF avec la valeur de chaque champ sous son nom complet. ASourceFile (optionnel) est écrit comme /F.
ExportXFDF(AStream or AFile-Name; ASourceFile)	Écrit un fichier XFDF (UTF-8). ASourceFile (optionnel) est écrit comme <f href>.

Importation

```
procedure ImportFormData(const APDFFile, AXFDFFile, AOutput: string);
var
  oPDF: TsgcPDFDocument;
  vCount: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.LoadFromFile(APDFFile);
    vCount := oPDF.AcroForm.ImportXFDF(AXFDFFile);
    WriteLn(vCount, ' values set, ', oPDF.AcroForm.ImportUnknownCount,
      ' unknown fields');
    // the appearances are regenerated from the imported values
    oPDF.SaveToFile(AOutput);
  finally
    oPDF.Free;
  end;
end;
```

`ImportFDF` et `ImportXFDF` définissent la valeur de chaque champ trouvé par son nom complet (par exemple `address.city`) et renvoient le nombre de valeurs définies. Les noms qui ne correspondent à aucun champ sont comptés dans `ImportUnknownCount`. Les apparences des champs modifiés sont régénérées lors de l'enregistrement du document.
Les valeurs texte sont en Unicode dans les deux formats, les cases à cocher et les boutons radio utilisent leur valeur d'export (par exemple Off ou Yes), et les listes à sélection multiple écrivent une valeur par option sélectionnée.

Voir aussi

- Valeurs et apparences des formulaires
- Formulaires interactifs

Valeurs et apparences des formulaires

```
procedure CreateForm(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oList: TsgcPDFFormField;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.AddPage(595, 842);
    oPDF.AcroForm.AddTextField('name', 0, 50, 740, 300, 760);
    oPDF.AcroForm.AddCheckBox('agree', 0, 50, 700, 65, 715);
    oList := oPDF.AcroForm.AddListBox('languages', 0, 50, 600, 200, 680,
      ['Delphi', 'C++', 'Pascal', 'Python']);
    // multi-select list box: flag bit 22 (MultiSelect)
    oList.Flags := oList.Flags or $200000;
    oList.SetValues(['Delphi', 'Pascal']);
    // Cyrillic value: the appearance embeds a font that covers it
    oPDF.AcroForm.AppearanceFontName := 'Arial';
    oPDF.AcroForm.FillField('name', 'Иван Петров');
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Listes à sélection multiple

Une liste avec l'indicateur MultiSelect (bit 22 de `Flags`, valeur $200000) peut contenir plusieurs valeurs. `SetValues` les sélectionne, `ValueCount` et `GetValues(AIndex)` les lisent, et `Value` renvoie la première. Les valeurs sont écrites sous forme de tableau dans /V et comme indices /I des options sélectionnées, et elles sont conservées par FDF et XFDF.

Apparences Unicode

L'apparence d'une valeur de champ contenant des caractères en dehors de l'encodage de police standard (cyrillique, grec, CJK, arabe, hébreu...) est dessinée avec une police TrueType ou OpenType intégrée, enregistrée sous `/SgcU` dans les ressources d'apparence et dans le dictionnaire /DR du formulaire, afin que la valeur s'affiche et s'imprime correctement même dans les lecteurs qui ne régénèrent pas les apparences.

Propriété	Description
AppearanceFontFile: string	Fichier de police (.ttf, .otf) utilisé pour ces valeurs. Prend le pas sur AppearanceFontName.
AppearanceFontName: WideString	Police installée utilisée pour ces valeurs.

Lorsque les deux sont vides, la première police installée parmi Arial Unicode MS, Segoe UI, Arial, Yu Gothic, MS Gothic, Microsoft YaHei, Malgun Gothic et Noto Sans qui couvre les caractères est utilisée. Sous Linux ou sur un serveur sans ces polices, définissez AppearanceFontFile.

NeedAppearances

`AcroForm.NeedAppearances` écrit `/NeedAppearances true`, ce qui demande au lecteur de régénérer les apparences des champs à l'ouverture du fichier. Cette valeur est True par défaut pour les nouveaux formulaires. Les documents chargés conservent la valeur du fichier. sgcPDF écrit toujours ses propres apparences, de sorte que les champs

s'affichent correctement aussi dans les lecteurs qui ignorent cet indicateur ; définissez-le sur False pour les fichiers PDF/A, car les normes PDF/A n'autorisent pas qu'il soit à true.

Voir aussi

- FDF et XFDF
- Formulaires interactifs

Signatures PAdES

PAdES (ETSI EN 319 142-1) est le profil européen des signatures PDF utilisé pour les signatures électroniques avancées et qualifiées au sens d'eIDAS. TsgcPDFSignatureHandler crée des signatures PAdES baseline B-B avec la CryptoAPI de Windows, accepte un CMS produit par n'importe quel signataire externe, et ajoute des horodatages de document et des données de validation pour les niveaux B-T, B-LT et B-LTA.

SubFilter

SubFilter (TsgcPDFSignatureSubFilter)	Écrit comme	Utilisation
pssPKCS7Detached (par défaut)	/adbe.pkcs7.detached	Signatures PDF classiques, compatibles avec tous les lecteurs.
pssCAdESDetached	/ ETSI.CAdES.detached	Signatures PAdES baseline. Le CMS porte l'attribut ESS signing-certificate-v2.

```
procedure SignPAdES(const AInput, AOutput, APFXFile: string);
var
  oSigner: TsgcPDFSignatureHandler;
begin
  oSigner := TsgcPDFSignatureHandler.Create;
  try
    oSigner.LoadCertificateFromPFX(APFXFile, 'test123');
    // PAdES baseline: /SubFilter /ETSI.CAdES.detached
    oSigner.SubFilter := pssCAdESDetached;
    oSigner.HashAlgorithm := shaSHA256;
    oSigner.SignerName := 'John Smith';
    oSigner.Reason := 'Approved';
    oSigner.Location := 'Barcelona';
    oSigner.SetField('Signature1', 0, 350, 50, 550, 110);
    oSigner.SignDocument(AInput, AOutput);
  finally
    oSigner.Free;
  end;
end;
```

Signataires externes (OnExternalSign)

Lorsque la clé privée n'est pas accessible à la CryptoAPI (un service de signature distant, un HSM, un intergiciel de carte à puce, un coffre de clés cloud), affectez **OnExternalSign**. Aucun certificat n'a besoin d'être chargé : sgcPDF prépare le dictionnaire de signature et les plages d'octets, et l'événement renvoie le CMS.

```
procedure TSignEvents.DoExternalSign(Sender: TObject; const AData: TBytes;
  const ADigest: TBytes; var ASignature: TBytes);
begin
  // AData: the bytes covered by /ByteRange, ADigest: their SHA-256 digest.
  // Return a complete DER encoded detached CMS SignedData
  ASignature := SignWithRemoteService(AData);
end;

procedure SignExternal(const AInput, AOutput: string; AEvents: TSignEvents);
var
  oSigner: TsgcPDFSignatureHandler;
begin
  oSigner := TsgcPDFSignatureHandler.Create;
  try
    // no certificate is loaded: the CMS comes from the event
```

```
    oSigner.OnExternalSign := AEvents.DoExternalSign;
    oSigner.SignerName := 'Remote signer';
    oSigner.SetInvisible;
    oSigner.SignDocument(AInput, AOutput);
  finally
    oSigner.Free;
  end;
end;
```

Para-mètre	Description
AData	Les octets couverts par /ByteRange (le fichier sans l'espace réservé /Contents).
ADigest	L'empreinte de AData avec HashAlgorithm, pour les services qui signent une empreinte.
ASigna-ture	Renvoyez un SignedData CMS détaché complet encodé en DER. Il doit tenir dans Placeholder-Size octets (8192 par défaut, augmentez-le pour les longues chaînes de certification ou les ho-rodatages intégrés).

Pour des signatures PAdES complètes construites avec sgcSign (tout fournisseur de clés, horodatages de signa-ture, OCSP), utilisez le pont TsgcPDFPAdESSigner.

Voir aussi

- Horodatage et validation à long terme
- Vérification des signatures
- Signatures numériques
- TsgcPDFSignatureHandler

Horodatage et validation à long terme

Une signature ne peut être validée que tant que son certificat est valide et que son statut de révocation peut être vérifié. La validation à long terme (LTV) garde une signature vérifiable au-delà, en intégrant les données de validation dans le document et en protégeant le tout avec un horodatage. TsgcPDFSignatureHandler écrit les deux sous forme de mises à jour incrémentales, de sorte que les signatures précédentes restent valides.

Horodatages de document

TimestampDocument (et TimestampDocumentStream) ajoute une signature invisible de type `/DocTimeStamp` avec SubFilter `/ETSI.RFC3161`. Le jeton d'horodatage provient de l'événement **OnTimestamp** : sgcPDF transmet l'empreinte à horodater, et votre code envoie une requête RFC 3161 à une autorité d'horodatage (TSA) et renvoie le TimeStampToken de la réponse.

```
procedure TSignEvents.DoTimestamp(Sender: TObject; const ADigest: TBytes;
  AHashAlgorithm: TsgcPDFSignatureHashAlgorithm; var AToken: TBytes);
begin
  // send an RFC 3161 request with ADigest to your time-stamp authority and
  // return the TimeStampToken of the response (DER)
  AToken := RequestTimeStampToken(ADigest, AHashAlgorithm);
end;

procedure AddDocumentTimestamp(const AInput, AOutput: string;
  AEvents: TSignEvents);
var
  oSigner: TsgcPDFSignatureHandler;
begin
  oSigner := TsgcPDFSignatureHandler.Create;
  try
    oSigner.OnTimestamp := AEvents.DoTimestamp;
    // appends a /DocTimeStamp (ETSI.RFC3161) as an incremental update
    oSigner.TimestampDocument(AInput, AOutput);
  finally
    oSigner.Free;
  end;
end;
```

OnTimestamp n'est utilisé que pour les horodatages de document. Les horodatages de signature à l'intérieur du CMS (PAdES B-T) sont ajoutés par TsgcPDFPAdESSigner.

Document Security Store (DSS)

Le DSS (`/DSS` dans le catalogue) contient les certificats, réponses OCSP et CRL dont un validateur a besoin, ainsi qu'une entrée `/VRI` pour chaque signature du document.

```
procedure AddValidationData(const AInput, AOutput: string);
var
  oSigner: TsgcPDFSignatureHandler;
begin
  oSigner := TsgcPDFSignatureHandler.Create;
  try
    // certificates of the chain, CRLs and OCSP responses (DER)
    oSigner.AddDSSCertificate(LoadBytes(Pki + 'ica.der'));
    oSigner.AddDSSCertificate(LoadBytes(Pki + 'root.der'));
    oSigner.AddDSSCRL(LoadBytes(Pki + 'ica.crl'));
    oSigner.AddDSSCRL(LoadBytes(Pki + 'root.crl'));
    // writes /DSS with a /VRI entry per signature (incremental update)
    oSigner.AddDSSDocument(AInput, AOutput);
  finally
    oSigner.Free;
  end;
```

```
end;
```

Méthode	Description
AddDSSCertificate(AData)	Ajoute un certificat DER (AC intermédiaire, AC racine, certificat de la TSA).
AddDSSOCSPResponse(AData)	Ajoute une réponse OCSP DER.
AddDSSCRL(AData)	Ajoute une liste de révocation de certificats (CRL) DER.
ClearDSS	Efface les données collectées.
AddDSSDocument(AInputFile, AOutputFile), AddDSSStream(AInput, AOutput)	Écrit le DSS sous forme de mise à jour incrémentale d'un document signé.

Niveaux

Niveau	Comment
B-B	Signez avec SubFilter pssCAdESDetached.
B-T	B-B avec un horodatage de signature, ou un horodatage de document ajouté juste après la signature.
B-LT	Ajoutez le DSS avec la chaîne et les données de révocation.
B-LTA	Ajoutez un horodatage de document après le DSS. Répétez le DSS et l'horodatage avant l'expiration du certificat d'horodatage.

Le résultat des exemples de cette page a été validé avec pyHanko : la signature CAdES est intacte, valide et fiable, les modifications qui la suivent ne sont reconnues que comme des mises à jour LTA, et l'horodatage de document est intact et fiable.

Voir aussi

- Signatures PAdES
- TsgcPDFPAdESSigner
- Vérification des signatures

Vérification des signatures

TsgcPDFSignatureVerifier vérifie chaque signature et chaque horodatage de document d'un PDF. `GetSignature(AIndex)` renvoie un TsgcPDFSignatureInfo avec les détails de chacune.

```
procedure VerifyAll(const AFileName: string);
var
  oVerifier: TsgcPDFSignatureVerifier;
  oSig: TsgcPDFSignatureInfo;
  i: Integer;
begin
  oVerifier := TsgcPDFSignatureVerifier.Create;
  try
    oVerifier.LoadDocument(AFileName);
    for i := 0 to oVerifier.GetSignatureCount - 1 do
    begin
      oVerifier.VerifySignature(i);
      oSig := oVerifier.GetSignature(i);
      WriteLn(string(oSig.FieldName), ' ', string(oSig.SubFilter));
      WriteLn('  doc time-stamp: ', oSig.IsDocTimeStamp);
      WriteLn('  signer: ', string(oSig.CertInfo.SubjectName));
      WriteLn('  signed: ', DateTimeToStr(oSig.SignDate));
      WriteLn('  valid: ', oSig.IsValid, ', trusted: ', oSig.IsTrusted,
        ', modified: ', oSig.IsModified);
    end;
  finally
    oVerifier.Free;
  end;
end;
```

Propriété	Description
FieldName	Nom du champ de signature.
SubFilter	Par exemple adbe.pkcs7.detached, ETSI.CAdES.detached ou ETSI.RFC3161.
IsDocTimeS-tamp	True pour un horodatage de document (/Type /DocTimeStamp). Son jeton est vérifié par rapport aux octets couverts.
SignDate	L'entrée /M de la signature, ou le genTime (UTC) du jeton d'un horodatage de document.
CertInfo	SubjectName, IssuerName, SerialNumber, NotBefore, NotAfter et Thumbprint du certificat du signataire (ou de la TSA).
IsValid	La signature CMS correspond aux plages d'octets signées.
IsTrusted	Le certificat du signataire remonte à une racine de confiance pour Windows. Vérifié séparément : une signature valide provenant d'un signataire inconnu est valide mais non fiable.
IsModified	La signature ne couvre pas l'ensemble du fichier. Les mises à jour incrémentales après la signature (remplissage de formulaire, annotations, DSS, horodatages) la définissent.
Name, Rea-son, Location	Les valeurs du dictionnaire de signature.

VerifySignature(AIndex) renvoie IsValid. IsSignatureTrusted(AIndex) et IsDocumentModified(AIndex) renvoient les deux autres résultats. La confiance est évaluée avec le magasin de certificats Windows, les certificats de test sont donc signalés comme non fiables. Pour une validation de politique PAdES par rapport à une liste de confiance spécifique, utilisez un validateur dédié tel que sgcSign.

Voir aussi

- Signatures PAdES
- TsgcPDFSignatureVerifier

TsgcPDFPAdESSigner (pont sgcSign)

TsgcPDFPAdESSigner (unité **sgcPDF_Sign_sgcSign**) crée des signatures PAdES baseline B-B, B-T, B-LT et B-LTA avec **sgcSign**, la bibliothèque de signature numérique eSeGeCe. sgcPDF gère la partie PDF (champ, apparence, mises à jour incrémentales, DSS, horodatage de document) et sgcSign gère la cryptographie : le CMS avec les attributs exigés par PAdES, le fournisseur de clés, les clients TSA et OCSP.
Comme la clé privée est accessible via un fournisseur de clés sgcSign, tout magasin de clés pris en charge par sgcSign fonctionne : fichiers PFX et PEM, jetons PKCS#11 et cartes à puce, magasin de certificats Windows et HSM cloud.

Prérequis

- Une licence de sgcSign et son dossier Source dans le chemin de recherche du projet, à côté de sgcPDF.
- sgcPDF_Sign_sgcSign.pas est une unité optionnelle : elle ne fait **pas** partie des paquets sgcPDF. Ajoutez-la à votre projet.

Exemple

```
procedure SignPAdESLT(const AInput, AOutput: string);
var
  oKey: TsgcPEMKeyProvider;
  oSigner: TsgcPDFPAdESSigner;
begin
  oKey := TsgcPEMKeyProvider.Create(nil);
  oSigner := TsgcPDFPAdESSigner.Create(nil);
  try
    // any sgcSign key provider: PEM, PFX, PKCS#11, Windows store, cloud HSM
    oKey.LoadFromFile(Pki + 'signer.pem', Pki + 'signer.key');
    oSigner.KeyProvider := oKey;
    oSigner.Level := paBLT;
    oSigner.TSAURL := 'http://timestamp.digicert.com';
    // validation material: CA certificates and CRLs (or an OCSPClient)
    oSigner.AddCertificate(LoadDER(Pki + 'ica.der'));
    oSigner.AddCertificate(LoadDER(Pki + 'root.der'));
    oSigner.AddCRL(LoadDER(Pki + 'ica.crl'));
    oSigner.AddCRL(LoadDER(Pki + 'root.crl'));
    // signature field and appearance through the sgcPDF handler
    oSigner.Handler.SetField('Signature1', 0, 350, 50, 550, 110);
    oSigner.Handler.SignerName := 'John Smith';
    oSigner.Handler.Reason := 'Approved';
    oSigner.SignPDFFile(AInput, AOutput);
    oSigner.KeyProvider := nil;
  finally
    oSigner.Free;
    oKey.Free;
  end;
end;
```

Cet exemple a été compilé avec Delphi 13 par rapport aux sources sgcPDF et sgcSign et exécuté avec une véritable TSA publique : pyHanko valide le résultat comme une signature ETSI.CAdES.detached intacte et fiable, avec un horodatage de signature valide et un /DSS.

Membre	Description
KeyProvider: IsgcKeyProvider	Le fournisseur de clés sgcSign avec le certificat du signataire et la clé privée.
Level: TsgcPDFPAdESLevel	paBB, paBT, paBLT ou paBLTA. Par défaut paBB.

Membre	Description
TSAURL: string / TSAClient: IsgcTSA-Client	Autorité d'horodatage pour B-T et supérieur : une URL (un client TSA sgcSign interne est utilisé) ou votre propre client.
OCSPClient: IsgcOCSPClient	Optionnel. Lorsqu'il est affecté, la réponse OCSP du certificat du signataire est récupérée et stockée pour B-LT.
HashAlgorithm	shaSHA256 (par défaut), shaSHA384 ou shaSHA512.
AddCertificate, AddCRL, AddOCSPResponse, ClearValidationData	Matériel de validation. Les certificats sont aussi intégrés dans le CMS ; les CRL et les réponses OCSP vont dans le /DSS.
Handler: TsgcPDFSignatureHandler	Champ de signature, apparence, motif, lieu et taille de l'espace réservé.
SignPDF(AInput, AOutput), SignPDFFile(AInputFile, AOutputFile)	Signe au niveau sélectionné.
ExtendToLTA(AInput, AOutput)	Ajoute le DSS et un horodatage de document à un PDF déjà signé.

Le pont n'écrit pas d'attribut CMS signing-time, que PAdES interdit à côté de l'entrée /M du dictionnaire de signature.

Voir aussi

- Signatures PAdES
- Horodatage et validation à long terme

Chiffrement par certificat

Le chiffrement par mot de passe partage un même secret avec tous les lecteurs. Le chiffrement par certificat (clé publique), le gestionnaire de sécurité `/Adobe.PubSec`, chiffre plutôt la clé du document pour le certificat X.509 de chaque destinataire : seuls les propriétaires des clés privées correspondantes peuvent ouvrir le fichier, aucun mot de passe n'est échangé, et chaque destinataire peut recevoir des permissions différentes.

Chiffrement

```
procedure EncryptForRecipients(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.AddPage(595, 842).DrawText('For the finance team only', 50, 780,
      sfHelvetica, 14);
    // every recipient gets its own permissions
    oPDF.AddEncryptionRecipient(Pki + 'signer.der', [ppPrint, ppCopy,
      ppExtract]);
    oPDF.AddEncryptionRecipient(Pki + 'tsa.pem', [ppPrint]);
    // emAES_256 writes /Adobe.PubSec with /SubFilter /adbe.pkcs7.s5
    oPDF.SetCertificateEncryption(emAES_256);
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Méthode	Description
AddEncryptionRecipient(ACertificate: TBytes; APermissions)	Ajoute un destinataire à partir d'un certificat DER ou PEM.
AddEncryptionRecipient(ACertificateFile: string; APermissions)	Identique, à partir d'un fichier .cer, .der ou .pem.
AddEncryptionRecipientFromStore(ACertificateId; APermissions; AStoreName)	Un certificat du magasin Windows, par empreinte SHA-1 (hex) ou partie du sujet. AStoreName vaut MY par défaut.
SetCertificateEncryption(AMethod)	Active le chiffrement pour les destinataires ajoutés. emAES_256 et emAES_128 écrivent /SubFilter /adbe.pkcs7.s5 (filtres de chiffrement), emRC4_128 et emRC4_40 l'ancien /adbe.pkcs7.s4.

APermissions est un ensemble de TsgcPDFPermission : ppPrint, ppModify, ppCopy, ppAnnotate, ppFillForms, ppExtract, ppAssemble et ppPrintHigh.

Ouverture

```
procedure OpenWithCertificate(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    // the private key of one recipient opens the document
```

```
    oPDF.SetDecryptionCertificate(Pki + 'signer.pfx', 'test123');
    oPDF.LoadFromFile(AFileName);
    WriteLn(string(oPDF.ExtractTextW(0)));
    WriteLn('Can print: ', ppPrint in oPDF.Encryption.Permissions);
    WriteLn('Can modify: ', ppModify in oPDF.Encryption.Permissions);
  finally
    oPDF.Free;
  end;
end;
```

Définissez la clé privée avant LoadFromFile ou LoadFromStream avec `SetDecryptionCertificate(APFXFileName, APassword)`, `SetDecryptionCertificate(APFX: TBytes, APassword)` ou `SetDecryptionCertificateFromStore(AStoreName)`, qui essaie les certificats disposant d'une clé privée dans le magasin de l'utilisateur courant. `Encryption.Permissions` contient alors les permissions du destinataire qui a ouvert le fichier.

Plateforme

Le chiffrement et le déchiffrement par certificat utilisent la CryptoAPI de Windows (crypt32.dll, incluse dans Windows), ils ne sont donc disponibles que sous Windows. Le fichier produit dans l'exemple a été déchiffré indépendamment par pyHanko avec le PFX du premier destinataire.

Remarques de sécurité

- Les clés de fichier, les vecteurs d'initialisation AES et les sels proviennent du CSPRNG du système d'exploitation (CryptGenRandom sous Windows, /dev/urandom sous Linux, macOS, iOS et Android).
- emAES_256 écrit la révision 6 du gestionnaire de sécurité de PDF 2.0 (ISO 32000-2). Les fichiers avec la révision 5 obsolète (Acrobat 9) sont quand même ouverts.
- Le chiffrement ne peut être modifié que par un enregistrement complet : SaveIncremental déclenche une exception s'il a été modifié.

Voir aussi

- Chiffrement
- TsgcPDFRedactor

TsgcPDFRedactor

TsgcPDFRedactor (unité sgcPDF_Redaction) effectue un caviardage réel : le texte, les pixels d'image, les graphismes vectoriels et les annotations à l'intérieur des zones marquées sont supprimés du fichier, pas seulement recouverts par un rectangle noir. Le contenu caviardé ne peut être sélectionné, copié, extrait ni récupéré depuis le résultat.

```
procedure RedactFile(const AInput, AOutput: string);
var
  oRedactor: TsgcPDFRedactor;
  vOptions: TsgcPDFRedactionOptions;
begin
  oRedactor := TsgcPDFRedactor.Create;
  try
    // every occurrence of a text, on all pages
    oRedactor.AddText('John Smith');
    // an area of page 0: left, bottom, right, top in points
    oRedactor.AddArea(0, 45, 735, 300, 752);
    oRedactor.AddArea(0, 50, 500, 250, 650);
    vOptions := oRedactor.Options;
    vOptions.OverlayText := 'REDACTED';
    vOptions.FillColorR := 0;
    vOptions.FillColorG := 0;
    vOptions.FillColorB := 0;
    vOptions.RemoveMetadata := True;
    oRedactor.Options := vOptions;
    oRedactor.Apply(AInput, AOutput);
    WriteLn('Text matches: ', oRedactor.TextMatches);
    WriteLn('Removed glyphs: ', oRedactor.RemovedGlyphs);
    WriteLn('Removed paths: ', oRedactor.RemovedPaths);
  finally
    oRedactor.Free;
  end;
end;
```

Marquage

Méthode	Description
AddArea(APageIndex; ARect: TsgcPDFRect)	Un rectangle en points, dans l'espace utilisateur par défaut de la page (origine en bas à gauche, avant le /Rotate de la page).
AddArea(APageIndex; ALeft, ABottom, ARight, ATop)	Identique, avec des coordonnées.
AddQuad(APageIndex; AQuad: TsgcPDFRedactionQuad)	Tout quadrilatère convexe, pour du texte pivoté. sgcPDFRedactionQuadFromRect convertit un rectangle.
AddText(AText; ACaseSensitive; AWholeWord; APageIndex)	Chaque occurrence de AText sur une page, ou sur toutes les pages lorsque APageIndex vaut -1. Les correspondances sont recherchées à l'intérieur d'une ligne de texte, et les suites d'espaces correspondent à n'importe quel espace.
Clear	Supprime toutes les marques.

Application

Méthode	Description
Apply(ADocument: TsgcPDFDocument)	Caviarde un document chargé : il est enregistré, caviardé puis rechargé à partir des octets caviardés.
Apply(AInputFile, AOutputFile)	Caviarde un fichier.
ApplyToStream(AInput, AOutput), ApplyToBytes(AData)	Caviarde un flux ou un tableau d'octets.

Le résultat est toujours un nouveau fichier complet (enregistrement complet) : les octets d'origine, qu'une mise à jour incrémentale conserverait, ne sont jamais écrits. Après Apply, les statistiques TextMatches, RemovedGlyphs, RedactedImages, RemovedImages, RemovedPaths et RemovedAnnotations sont disponibles, et `GetLog` renvoie un journal de l'opération.

Options

Champ de TsgcPDFRedactionOptions	Description
FillColorR, FillColorG, FillColorB	Couleur (de 0 à 1) des rectangles peints sur les zones. Par défaut noir.
OverlayText, OverlayFontSize	Texte dessiné centré dans chaque zone (Helvetica), 0 ajuste à la zone.
RemoveAnnotationsInArea	Supprime les annotations dont le rectangle croise une zone, y compris les liens et les widgets de champs de formulaire avec leurs champs. Par défaut True.
RemoveMetadata	Supprime le dictionnaire /Info et les métadonnées XMP.
RemoveAttachments	Supprime les fichiers intégrés et les annotations de pièces jointes.
RemoveJavaScript	Supprime le JavaScript, /OpenAction, les actions supplémentaires et les actions JavaScript, Launch, SubmitForm et ImportData.

Documents chiffrés

Ouvrez le document avec son mot de passe et appelez `Apply(ADocument)`. Le contenu est caviardé déchiffré et le document conserve ses paramètres de chiffrement, de sorte que le prochain SaveToFile ou SaveToStream l'écrit à nouveau chiffré. SaveIncremental déclenche une exception après un caviardage. Les surcharges fichier, flux et octets rejettent une entrée chiffrée, car elles ne prennent aucun mot de passe.

```
procedure RedactEncrypted(const AFileName, APassword, AOutput: string);
var
  oPDF: TsgcPDFDocument;
  oRedactor: TsgcPDFRedactor;
begin
  oPDF := TsgcPDFDocument.Create;
  oRedactor := TsgcPDFRedactor.Create;
  try
    oPDF.SetPassword(AnsiString(APassword));
    oPDF.LoadFromFile(AFileName);
    oRedactor.AddText('Salary', False, True);
    oRedactor.Apply(oPDF);
    // full save: written encrypted again with the same settings
    oPDF.SaveToFile(AOutput);
```

```
  finally
    oRedactor.Free;
    oPDF.Free;
  end;
end;
```

Comment le contenu est supprimé

- **Texte** : chaque glyphe dont le rectangle croise une zone est supprimé du flux de contenu ; les glyphes restants conservent leurs positions.
- **Images** : les images que sgcPDF peut décoder, avec 1 à 8 bits par composant, sont réécrites avec les pixels caviardés peints dans la couleur de remplissage (RedactedImages). Les images qui ne peuvent pas être décodées, par exemple avec un filtre non pris en charge, trop grandes ou avec 16 bits par composant, sont entièrement supprimées (RemovedImages).
- **Graphismes vectoriels** : un sous-chemin entièrement situé à l'intérieur d'une zone est supprimé (RemovedPaths). Les sous-chemins qui ne font que croiser une zone restent, sous le rectangle de remplissage. Les chemins de découpage ne sont pas modifiés.
- **Les XObjects de formulaire** sont caviardés récursivement. Un XObject de formulaire imbriqué trop profondément ou avec un filtre non pris en charge est supprimé dans son intégralité.
- Une page dont le flux de contenu utilise un filtre non pris en charge déclenche EsgcPDFRedactionError plutôt que d'être laissée non caviardée.

Dans l'exemple, pypdf ne trouve aucune trace des mots caviardés dans le résultat, ni dans le texte extrait ni dans aucun flux de contenu, et le fichier n'a qu'une seule révision.

Voir aussi

- Chiffrement par certificat
- Extraction de texte

PDF balisé

Un PDF balisé (ISO 32000-1, 14.8) contient un arbre de structure logique en plus du contenu des pages : titres, paragraphes, listes, tableaux, figures avec texte alternatif et liens, dans l'ordre de lecture. Les lecteurs d'écran, la réorganisation sur les appareils mobiles et l'export de texte l'utilisent, et il est exigé par la conformité PDF/UA et PDF/A niveau A.

Construction de la structure

Définissez **TsgcPDFDocument.Tagged** sur True et encadrez les appels de dessin de chaque page entre `Page.Tagging.BeginElement` et `EndElement`. Le balisage de page écrit les opérateurs de contenu marqué (BDC/EMC avec MCID) et les relie à l'élément ; `StructureTree` rassemble les éléments, le ParentTree et le RoleMap, et écrit / MarkInfo, /Lang et les /StructParents de la page lors de l'enregistrement du document.

```
procedure CreateTaggedPDF(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oPage: TsgcPDFPage;
  oTag: TsgcPDFPageTagging;
  oFigure, oCell: TsgcPDFStructElement;
  vFont: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.Info.Title := 'Quarterly report';
    oPDF.Tagged := True;
    oPDF.StructureTree.Lang := 'en-US';
    vFont := oPDF.AddTrueTypeFont('Arial', False, False);
    oPage := oPDF.AddPage(595, 842);
    oTag := oPage.Tagging;
    // running header: not part of the logical content
    oTag.BeginArtifact(akPagination, asHeader);
    oPage.DrawTextTT('ACME Corp.', 50, 810, vFont, 8);
    oTag.EndArtifact;
    oTag.BeginElement(stH1);
    oPage.DrawTextTT('Quarterly report', 50, 770, vFont, 20);
    oTag.EndElement;
    oTag.BeginElement(stP);
    oPage.DrawTextTT('Sales grew in every region.', 50, 740, vFont, 11);
    oTag.EndElement;
    // figures need an alternate text
    oFigure := oTag.BeginElement(stFigure);
    oFigure.Alt := 'Blue bar chart of the sales per region';
    oFigure.SetBBox(50, 600, 250, 700);
    oPage.DrawRectangleFill(50, 600, 200, 100, 0.2, 0.4, 0.8);
    oTag.EndElement;
    // grouping elements (no content) are opened on the tree
    oPDF.StructureTree.BeginElement(stTable);
    oPDF.StructureTree.BeginElement(stTR);
    oCell := oTag.BeginElement(stTH);
    oCell.Scope := tsColumn;
    oPage.DrawTextTT('Region', 50, 570, vFont, 11);
    oTag.EndElement;
    oCell := oTag.BeginElement(stTH);
    oCell.Scope := tsColumn;
    oPage.DrawTextTT('Sales', 200, 570, vFont, 11);
    oTag.EndElement;
    oPDF.StructureTree.EndElement;
    oPDF.StructureTree.BeginElement(stTR);
    oTag.BeginElement(stTD);
    oPage.DrawTextTT('North', 50, 550, vFont, 11);
    oTag.EndElement;
```

```
    oTag.BeginElement(stTD);
    oPage.DrawTextTT('1200', 200, 550, vFont, 11);
    oTag.EndElement;
    oPDF.StructureTree.EndElement;
    oPDF.StructureTree.EndElement;
    // a link: the annotation added inside the Link element is attached to it
    oTag.BeginElement(stLink).Alt := 'eSeGeCe web site';
    oPage.DrawTextTT('www.esegece.com', 50, 520, vFont, 11);
    oPDF.Annotations.AddLinkURI(0, 50, 515, 150, 532,
      'https://www.esegece.com');
    oTag.EndElement;
    // PDF/UA-1 and PDF/A-2a in the same file
    oPDF.SetPDFALevel(paA2a);
    oPDF.PDFUA := True;
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Membre	Description
Page.Tagging.BeginElement(AType; ATitle)	Ouvre un élément dont le contenu est dessiné sur cette page, en tant qu'enfant de l'élément courant. Renvoie le TsgcPDFStructElement.
StructureTree.BeginElement(AType; ATitle)	Ouvre un élément de regroupement sans contenu de page (Table, TR, L, Sect, Part...).
EndElement	Ferme l'élément courant (sur le balisage de page ou sur l'arbre, en correspondance avec l'appel Begin).
BeginCustomElement(ATypeName...) and StructureTree.AddRole	Types de structure personnalisés, mappés vers un type standard par le /RoleMap.
Page.Tagging.BeginArtifact(AKind; ASubtype), EndArtifact	Contenu qui ne fait pas partie du document : en-têtes et pieds de page répétés, numéros de page, filigranes, décoration. AKind : akPagination, akLayout, akPage, akBackground. ASubtype : asHeader, asFooter, asWatermark.
StructureTree.Lang	Langue naturelle du document (/Lang du catalogue), par exemple en-US.

TsgcPDFStructType comporte tous les types standard : stDocument, stPart, stArt, stSect, stDiv, stH, stH1 à stH6, stP, stL, stLI, stLbl, stLBody, stTable, stTHead, stTBody, stTFoot, stTR, stTH, stTD, stFigure, stCaption, stLink, stSpan, stQuote, stNote, stReference, stCode, stBlockQuote, stFormula, stTOC, stTOCI et stAnnot.

Propriétés de l'élément

Propriété	Description
Alt	Texte alternatif. Obligatoire pour Figure et Formula : enregistrer une Figure sans Alt déclenche EsgcPDFTaggedError.
ActualText, Expansion, Lang, Title, ID	Texte de remplacement, développement d'abréviation, langue de l'élément, titre et identifiant.
SetBBox(ALeft, ABottom, ARight, ATop)	Rectangle englobant d'une Figure, Formula ou Table.

Propriété	Description
Scope, RowSpan, ColSpan, Headers	Cellules de tableau : portée d'un TH (tsRow, tsColumn, tsBoth), fusions et identifiants des cellules d'en-tête.

Liens et annotations

Une annotation ajoutée avec `Annotations.AddLinkURI` (ou une autre méthode Add) pendant qu'un élément Link est ouvert lui est rattachée comme référence d'objet, avec son /StructParent. `Page.Tagging.AddAnnotation` en rattache une explicitement.

Tableaux

Un TsgcPDFTable dessiné sur un document balisé est automatiquement balisé en Table, THead, TBody, TR, TH et TD. Avec DrawMultiPage, les lignes d'en-tête répétées sur les pages suivantes sont balisées comme artefacts de pagination, afin qu'un lecteur d'écran ne les lise qu'une seule fois.

```
procedure CreateTaggedTable(const AFileName: string);
var
  oPDF: TsgcPDFDocument;
  oTable: TsgcPDFTable;
  i: Integer;
begin
  oPDF := TsgcPDFDocument.Create;
  try
    oPDF.Tagged := True;
    oPDF.StructureTree.Lang := 'en-US';
    oTable := TsgcPDFTable.Create(2);
    try
      oTable.SetHeaders(['Product', 'Quantity']);
      for i := 1 to 60 do
        oTable.AddRowData([AnsiString('Item ' + IntToStr(i)),
          AnsiString(IntToStr(i))]);
      // tagged as Table / TR / TH / TD; the header repeated on the next
      // pages is tagged as a pagination artifact
      oTable.DrawMultiPage(oPDF, 50, 790, 495, 595, 842);
    finally
      oTable.Free;
    end;
    oPDF.SaveToFile(AFileName);
  finally
    oPDF.Free;
  end;
end;
```

Les documents balisés chargés conservent leur arbre de structure lors de l'enregistrement.

Voir aussi

- PDF/UA et PDF/A niveau A
- Validateur d'accessibilité
- Tableaux

PDF/UA et PDF/A niveau A

PDF/UA-1

PDF/UA-1 (ISO 14289-1) est la norme pour les PDF universellement accessibles, exigée par les lois sur l'accessibilité pour les documents publics dans de nombreux pays. Définissez **TsgcPDFDocument.PDFUA** sur True : le document est enregistré avec pdfuaid:part 1 et dc:title dans les métadonnées XMP, et /DisplayDocTitle dans les préférences du lecteur.
L'enregistrement déclenche une exception lorsqu'une exigence PDF/UA ne peut pas être respectée :

- le document n'est pas balisé (`Tagged` vaut False) ;
- `StructureTree.Lang` est vide ;
- `Info.Title` est vide ;
- du texte est dessiné avec l'une des 14 polices standard, qui ne sont pas intégrées : utilisez AddTrueType-Font (ou AddTrueTypeFontFromFile) et DrawTextTT ;
- le document est chiffré sans la permission ppExtract, dont les technologies d'assistance ont besoin.

PDF/A niveau A

Le niveau A (accessible) de PDF/A ajoute les exigences du PDF balisé au niveau B. Passez **paA1a**, **paA2a** ou **paA3a** à SetPDFALevel, et balisez le contenu comme pour PDF/UA. PDF/UA et PDF/A peuvent être combinés dans le même fichier :

```
oPDF.SetPDFALevel(paA2a);
oPDF.PDFUA := True;
```

Le fichier créé dans PDF balisé utilise exactement cette combinaison et passe la validation veraPDF pour PDF/UA-1 et PDF/A-2a.

TsgcPDFALevel	Norme
paA1b, paA1a	PDF/A-1 (ISO 19005-1), niveau B ou A
paA2b, paA2a	PDF/A-2 (ISO 19005-2), niveau B ou A
paA3b, paA3a	PDF/A-3 (ISO 19005-3), niveau B ou A. Autorise tout fichier intégré.

Liste de contrôle

- Définissez Info.Title et StructureTree.Lang.
- Intégrez chaque police (TrueType ou OpenType).
- Balisez tout le contenu : chaque texte et graphisme appartient à un élément ou à un artefact.
- Donnez un texte alternatif à chaque élément Figure et Formula.
- Utilisez les titres dans l'ordre (H1, puis H2...), des cellules TH avec Scope dans les tableaux, et des éléments Link pour les liens.
- Validez le fichier enregistré avec le validateur.

Voir aussi

- PDF balisé
- Validateur d'accessibilité
- PDF/A

Validateur d'accessibilité

TsgcPDFAValidator (unité sgcPDF_PDFA) vérifie un fichier enregistré par rapport à un niveau PDF/A. Pour les niveaux A, il vérifie également la structure, et lorsque **PDFUA** est défini sur True, il exécute aussi les vérifications PDF/UA-1.

```
procedure ValidateAccessible(const AFileName: string);
var
  oParser: TsgcPDFParser;
  oValidator: TsgcPDFAValidator;
  i: Integer;
begin
  oParser := TsgcPDFParser.Create;
  oValidator := TsgcPDFAValidator.Create(paA2a);
  try
    oParser.LoadFromFile(AFileName);
    oParser.Parse;
    // PDF/UA-1 checks in addition to the PDF/A-2a ones
    oValidator.PDFUA := True;
    if oValidator.Validate(oParser) then
      WriteLn('PDF/A-2a and PDF/UA-1: no errors')
    else
      for i := 0 to oValidator.Errors.Count - 1 do
        WriteLn('Error: ', oValidator.Errors[i]);
    for i := 0 to oValidator.Warnings.Count - 1 do
      WriteLn('Warning: ', oValidator.Warnings[i]);
  finally
    oValidator.Free;
    oParser.Free;
  end;
end;
```

Ce qui est vérifié

Méthode	Vérifications
CheckStructure	Arbre de structure, /MarkInfo /Marked, /Lang.
CheckContentTagged	Chaque opérateur de texte et de dessin de chaque page appartient à un élément de structure ou à un artefact.
CheckStructureElements	Texte alternatif des éléments Figure et Formula, séquence des titres, structure des tableaux et cellules d'en-tête.
CheckFontUnicode	Chaque police dispose d'une table ToUnicode ou d'un encodage standard, afin que le texte puisse être lu.
CheckPDFUA	pdfuaid:part, dc:title, /DisplayDocTitle, annotations dans la structure, /Tabs /S de la page.

Validate exécute toutes les vérifications du niveau (et PDF/UA lorsqu'il est activé) et renvoie True lorsqu'il n'y a aucune erreur. Les méthodes individuelles peuvent être appelées séparément.

Erreurs et avertissements

`Errors` répertorie les problèmes qui rompent la conformité. Chaque message commence par la clause de la norme à laquelle il se réfère, afin que vous puissiez la consulter, par exemple :

```
[PDF/A-2a 6.7.3.3] [PDF/UA-1 7.1] Page 1: 1 content operator(s) neither tagged nor marked as artifact (first: TJ)
[PDF/UA-1 5] XMP metadata without pdfuaid:part 1
```

`Warnings` répertorie les problèmes qui ne sont pas des erreurs au sens de la norme mais qui nuisent à l'accessibilité, par exemple :

```
[PDF/UA-1 7.4.2] Heading level skipped: H1 followed by H3
```

Le validateur est une vérification intégrée rapide pour vos propres sorties. Il ne remplace pas un contrôleur de conformité complet tel que veraPDF ou PAC pour des documents provenant d'autres sources.

Voir aussi

- PDF/UA et PDF/A niveau A
- PDF balisé
- PDF/A

FAQ

Dois-je déployer une DLL ?

Non. sgcPDF est compilé directement dans votre exécutable. Il utilise uniquement des bibliothèques qui font partie de Windows : GDI pour le rendu et le chargement des polices, et la CryptoAPI pour les signatures numériques.

Quelles plateformes sont prises en charge ?

Windows 32 bits et 64 bits avec Delphi 7 à Delphi 13 et C++Builder 2007 à 13. Le noyau, le moteur de rendu raster et le lecteur FireMonkey fonctionnent également sous Linux 64 bits, macOS, iOS et Android. Voir Installation.

Où se trouve l'origine des coordonnées de la page ?

Dans le coin inférieur gauche de la page, en points (1/72 pouce). L'axe Y croît vers le haut. Voir Architecture.

Pourquoi les caractères accentués ou non latins n'apparaissent-ils pas ?

Les polices standard ne couvrent que le jeu de caractères WinAnsi. Utilisez une police TrueType avec AddTrueTypeFont et DrawTextTT. Voir Texte et polices.

Comment écrire du texte chinois, japonais ou coréen ?

Ajoutez une police TrueType CJK installée, par exemple le nom renvoyé par sgcPDFCJKFontNameFromEnum(cjkMSGothic), et dessinez avec DrawTextTT.

Comment ouvrir un PDF protégé par mot de passe ?

Appelez SetPassword avant LoadFromFile. Voir Chiffrement.

Mon code-barres n'apparaît pas.

Passez le générateur de contenu de page (`Page.Content`) à TsgcPDFBarcodeGenerator, et non la page. Voir Codes-barres.

sgcPDFExtractAllText renvoie une chaîne vide.

Un TsgcPDFParser doit être analysé après son chargement : appelez Parse après LoadFromFile. TsgcPDFDocument.LoadFromFile le fait pour vous.

Pourquoi la signature est-elle valide mais non fiable ?

Le certificat du signataire ne remonte pas à une autorité racine de confiance pour Windows, ce qui est normal pour des certificats de test auto-signés. Utilisez un certificat émis par une autorité de certification de confiance en production.

L'optimiseur n'a pas réduit mon fichier.

Les fichiers déjà compressés et sans objets dupliqués ne peuvent pas être réduits davantage. Activez DownsampleImages ou les options Remove pour les réduire encore. Voir Optimisation des fichiers PDF.

Puis-je limiter la mémoire utilisée à l'ouverture de fichiers envoyés par des utilisateurs ?

Oui. Réduisez sgcPDFMaxDecodedSize. Voir Gestion des erreurs et entrées malveillantes.

sgcPDF fonctionne-t-il sous Linux ?

Oui. Les unités du noyau se compilent avec le compilateur Delphi Linux 64 bits : créez, modifiez, chiffrez avec des mots de passe, extrayez le texte et effectuez le rendu des pages en PNG avec TsgcPDFRendererRaster dans une application serveur ou console. Chargez les polices depuis des fichiers avec AddTrueTypeFontFromFile. Voir Linux.

Puis-je utiliser sgcPDF depuis C++Builder ?

Oui. Les paquets pour C++Builder 2007 à 13 sont inclus et les en-têtes générés sont prêts à l'emploi. Voir C++Builder.

Comment créer un fichier PDF/A-3 avec une facture intégrée (Factur-X, ZUGFeRD) ?

Créez le XML de la facture dans votre application et passez-le à Attachments.AddFacturX. Le document est enregistré en PDF/A-3b avec le XML et les métadonnées Factur-X. Voir Factures Factur-X et ZUGFeRD.

sgcPDF peut-il créer des signatures PAdES ?

Oui. Définissez SubFilter sur pssCAdESDetached pour PAdES B-B, puis ajoutez les données de validation avec AddDSSDocument et un horodatage de document avec TimestampDocument pour B-LT et B-LTA. Avec sgcSign, TsgcPDFPAdESSigner crée chaque niveau en un seul appel avec n'importe quel fournisseur de clés sgcSign. Voir Signatures PAdES.

Un rectangle noir masque-t-il le texte en dessous ?

Non, le texte reste présent dans le fichier. Utilisez TsgcPDFRedactor, qui supprime le texte, les pixels d'image et les graphismes situés sous les zones caviardées. Voir TsgcPDFRedactor.

Comment créer un PDF accessible ?

Balisez le contenu avec Page.Tagging, définissez Info.Title et StructureTree.Lang, intégrez les polices et définissez PDFUA sur True. Voir PDF balisé.

Introduction

sgcReports est un moteur de rapports sans interface, neutre vis-à-vis de la plateforme, pour Delphi, construit sur sgcPDF. Il vise d'abord la génération de rapports serveur et web : services Windows, démons Linux64 et points de terminaison HTTP, et il s'exécute sans modification dans des applications de bureau et mobiles VCL et FMX.
Le moteur produit par défaut une sortie PDF accessible (PDF/UA-1) et PDF d'archivage (PDF/A-3). Un rapport peut également être exporté en XLSX, CSV ou PNG à partir des mêmes pages préparées.

Sans interface par conception

Le noyau (modèle de rapport, expressions, sources de données, polices, mise en page, moteur et exports) n'a aucune dépendance VCL ou FMX et aucune DLL externe. Il se compile et s'exécute sur Windows, Linux64, macOS, iOS et Android. Les composants d'aperçu (`TsgcReportPreviewVCL`, `TsgcReportPreviewFMX`) et le visualiseur navigateur sont des packages complémentaires optionnels.

Accessible et archivable par défaut

L'export PDF balise automatiquement l'arbre de structure : titres, paragraphes, tableaux avec lignes d'en-tête, figures avec texte alternatif et artefacts, de sorte que la propriété `PDFUA` vaut `True` par défaut. La sortie d'archivage PDF/A-3 est contrôlée par la propriété `PDFA` et peut être combinée avec PDF/UA. Voir Accessibilité et PDF/A.

Plateformes

Delphi 7, 2007, 2009, 2010, XE à XE8, 10 à 10.4, 11, 12 et 13. Windows, Linux64, macOS, iOS et Android via FireMonkey. Le noyau se compile sur Delphi 7 sans generics, méthodes anonymes, variables inline ni méthodes d'enregistrement.

Éditions et prérequis

Package	Nécessite	Nécessaire pour
`sgcReports` (noyau)	sgcPDF	Modèle de rapport, expressions, sources de données, polices, mise en page, moteur, export PDF/XLSX/CSV/PNG, API fluide, définitions JSON. sgcPDF est requis dans toutes les configurations.
`sgcReportsVCL` / `sgcReportsFMX`	noyau, VCL ou FMX	`TsgcReportPreviewVCL` ou `TsgcReportPreviewFMX`.
`sgcReportsHTML`	noyau, sgcWebSockets (édition Professional ou supérieure), sgcHTML	`TsgcReportHTTPEndpoint`, `TsgcReportHTMLViewer` et `TsgcReportHTMLDesigner`. sgcWebSockets n'est nécessaire que pour ce package ; il n'est jamais utilisé par le noyau.

Suivant : Installation, puis Prise en main.

Installation

sgcReports fait partie de la distribution sgcUI. Un seul installateur couvre les produits sgcUI (sgcUICore, sgcCharts, sgcPDF, sgcReports, sgcMaps, sgcDashboard, sgcGrid, sgcNavigation, sgcToolbar et sgcWizard) et n'installe que les produits inclus dans votre licence. sgcReports nécessite sgcPDF, et l'installateur s'en charge.

Prérequis

- Delphi 7, 2007, 2009, 2010, XE à XE8, 10 à 10.4, 11, 12 ou 13 (ou la version C++Builder correspondante, C2007 à C13).
- sgcPDF, même version de Delphi. L'installateur sgcUI l'installe avec sgcReports.
- sgcWebSockets édition Professional ou supérieure, et sgcHTML, uniquement lorsque le package `sgcReportsHTML` est installé.

Installation avec l'installateur sgcUI

1. Lancez l'installateur sgcUI et connectez-vous avec le nom d'utilisateur et le mot de passe de votre compte eSeGeCe.
2. Choisissez les versions de Delphi et de C++Builder dans lesquelles installer.
3. Sur la page des composants, sélectionnez **sgcReports (installs sgcPDF)** et tout autre produit souhaité. Le serveur de licences renvoie les produits que vous possédez, et l'installateur affiche une case à cocher par produit. Les produits que votre licence n'inclut pas sont grisés. Une licence All-Access déverrouille tous les produits.
4. L'installateur copie les fichiers, puis compile les packages sur votre machine pour chaque IDE sélectionné, dans l'ordre des dépendances, et installe les packages de conception.

Ce qui est fourni avec sgcReports

- **sgcPDF**, le moteur sur lequel repose sgcReports. Sélectionner sgcReports sélectionne aussi sgcPDF, et sgcPDF est installé chaque fois que sgcReports l'est.
- **sgcReports HTML pour sgcWebSockets**, listé sous **sgcWebSockets bridges**. Il installe les packages `sgcReportsHTML` (point de terminaison HTTP, visualiseur navigateur et concepteur web) pour Delphi 10 Seattle et versions ultérieures. L'installateur ne le propose que si sgcWebSockets est déjà installé sur la machine et que votre licence inclut sgcReports. Sinon, il est grisé.

Dossiers installés

Tous les produits s'installent dans un seul dossier, par défaut `Documents\esegece\sgcUI` (`sgcUI_Trial` pour la version d'essai et `sgcUI_Community` pour l'édition Community).

Dossier	Contenu
`Source`	Le code source de chaque produit installé, y compris sgcReports et sgcPDF. C'est l'unique chemin de bibliothèque de sgcUI.
`Packages`	Les packages de chaque produit installé, un jeu par produit et par version d'IDE.
`Lib<V>`	Les unités compilées pour chaque version d'IDE et chaque plateforme, par exemple `LibD13` .
`Demos\Reports`	Les démos de sgcReports.

L'installateur ajoute `sgcUI\Source` au chemin de bibliothèque et au chemin de navigation de chaque IDE et plate-forme sélectionnés, ainsi que le dossier des unités compilées. sgcPDF se trouve dans le même dossier `Source`, il n'a donc besoin d'aucun chemin propre.

Trial et Community

- **Trial** : l'installateur Trial de sgcUI installe tous les produits en mode d'essai, y compris sgcReports et sgcPDF. Un build d'essai affiche un avis au démarrage (`sgcReports_License.pas`). Un build sous licence n'en affiche pas.
- **Community** : sgcPDF n'a pas d'édition Community, l'installateur Community gratuit de sgcUI ne l'installe donc pas. Il ne propose sgcReports que si un sgcPDF sous licence est déjà installé sur la machine, soit depuis une installation sgcUI enregistrée qui inclut sgcPDF, soit depuis une installation autonome de sgcPDF, et il compile sgcReports avec ce sgcPDF. Sans lui, sgcReports est grisé et l'installateur vous demande d'installer d'abord sgcPDF. Les passerelles sgcWebSockets ne font pas partie de l'installateur Community.

Migration depuis des installations autonomes de sgcPDF et de sgcReports

Si sgcPDF ou sgcReports ont déjà été installés avec leur propre installateur autonome, l'installateur sgcUI détecte cette installation et liste ses chemins de bibliothèque et ses packages de conception sur la page **Previous installations**. Quand vous continuez, il les supprime de l'IDE avant de compiler les nouveaux packages, de sorte qu'il n'y ait jamais deux copies d'une unité dans le chemin. La page n'est affichée que lorsqu'une ancienne installation est trouvée. Après la migration, les deux produits sont compilés depuis l'unique dossier `sgcUI\Source`.

Compiler les packages manuellement

Pour compiler vous-même les packages, ouvrez-les pour votre version d'IDE depuis `sgcUI\Packages`. `<V>` est le suffixe de version Delphi, par exemple `sgcReportsD13.dpk`.

Package	Contenu
`sgcPDF<V>` , `sgcPDFVCL<V>` , `sgcPDFFMX<V>` / `dclsgcPDF<V>` , `dclsgcPDFFMX<V>`	sgcPDF, le moteur dont sgcReports a besoin (runtime / design time).
`sgcReports<V>`	Moteur principal et exports (runtime).
`sgcReportsVCL<V>` / `dclsgcReports<V>`	Aperçu VCL (runtime / design time).
`sgcReportsFMX<V>` / `dclsgcReportsFMX<V>`	Aperçu FMX (runtime / design time).
`sgcReportsHTML<V>` / `dclsgcReportsHTML<V>`	Point de terminaison HTTP, visualiseur navigateur et concepteur web (runtime / design time).

Compilez-les dans l'ordre des dépendances : d'abord les packages runtime de sgcPDF, puis le noyau de sgcReports, puis VCL ou FMX, puis les packages design time, et `sgcReportsHTML` en dernier. Installez ensuite les packages design time `dcl*`. `TsgcReport` apparaît sur la page de palette **sgcReports** après l'installation du package design time.

Chemin de recherche

Ajoutez `sgcUI\Source` au chemin de bibliothèque de votre version de Delphi (Tools > Options > Language > Delphi Options > Library > Library path), ou ajoutez-le au chemin de recherche des unités du projet. sgcReports et sgcPDF partagent ce dossier, une seule entrée couvre donc les deux.

C++Builder

Ouvrez le projet `.cbproj` correspondant à votre version de C++Builder sous `sgcUI\Packages`, par exemple `sgcReportsC13.cbproj`, et compilez-le ; les en-têtes `.hpp` générés apparaissent à côté du package compilé.

Prise en main

Il existe trois façons de construire un rapport : l'API de modèle (remplir un composant TsgcReport avec des pages, des bandes et des éléments), l'API fluide (chaîner des appels sur un TsgcReportBuilder), ou une définition de rapport JSON chargée à l'exécution. Les trois aboutissent au même TsgcReportEngine et aux mêmes exports.

1. L'API de modèle

Déposez un `TsgcReport`, un `TsgcReportEngine` et un `TsgcReportPDFExport` sur un formulaire (ou créez-les en code) et remplissez le rapport avec une page, une bande et un élément texte :

```
var
  oPage: TsgcReportPage;
  oBand: TsgcReportBand;
  oPages: TsgcReportPreparedPages;
begin
  sgcReport.Units := ruPoints;
  with sgcReport.Styles.Add do
  begin
    Name := 'Title';
    Font.Name := 'Helvetica';
    Font.Size := 18;
    Font.Bold := True;
  end;

  oPage := sgcReport.Pages.Add;
  oPage.PaperSize := rpsA4;
  oPage.Margins.Left := 40;
  oPage.Margins.Top := 40;
  oPage.Margins.Right := 40;
  oPage.Margins.Bottom := 40;

  oBand := oPage.Bands.Add;
  oBand.BandType := rbtReportTitle;
  oBand.Height := 40;
  with oBand.Items.Add do
  begin
    Kind := rikText;
    Name := 'Title1';
    Text.Text := 'Hello, sgcReports';
    Width := 400;
    Height := 24;
    StyleName := 'Title';
  end;

  sgcEngine.Report := sgcReport;
  oPages := sgcEngine.Prepare;
  try
    sgcPDFExport.SaveToFile(oPages, 'hello.pdf');
  finally
    oPages.Free;
  end;
end;
```

2. L'API fluide

Le même rapport avec `TsgcReportBuilder`, qui ne nécessite aucun composant sur le formulaire :

```
var
  vBuilder: TsgcReportBuilder;
begin
```

```
  vBuilder := TsgcReportBuilder.Create;
  try
    vBuilder.Units(ruPoints).Style('Title').Font('Helvetica', 18).Bold.EndStyle
      .Page.Paper(rpsA4).Margins(40, 40, 40, 40)
      .ReportTitle(40).Text('Hello, sgcReports', 0, 0, 400, 24).Style('Title')
      .EndBand.EndPage;
    vBuilder.SaveToPDF('hello.pdf');
  finally
    vBuilder.Free;
  end;
end;
```

Chaque appel de style, de page et de bande retourne un générateur limité à une portée (`TsgcReportStyleBuilder`, `TsgcReportPageBuilder`, `TsgcReportBandBuilder`) ; `EndStyle`, `EndBand` et `EndPage` reviennent à la portée englobante afin que la chaîne puisse continuer.
Voir TsgcReportBuilder pour la chaîne complète (styles, pages, bandes, éléments, données, assistants d'export).

3. Une définition de rapport JSON

Un rapport construit de l'une ou l'autre façon peut être enregistré en JSON et rechargé sans recompilation :

```
var
  vJSON: string;
  vLoaded: TsgcReport;
begin
  vJSON := sgcReportSaveToJSON(sgcReport);
  vLoaded := TsgcReport.Create(nil);
  try
    sgcReportLoadFromJSON(vLoaded, vJSON);
    // vLoaded now has the same pages, bands, items and styles as sgcReport
  finally
    vLoaded.Free;
  end;
end;
```

Voir Définitions de rapport JSON pour le format d'enveloppe, les règles de nommage et le schéma généré.

Étapes suivantes

- Modèle de rapport : pages, bandes, éléments, styles et unités.
- Sources de données : lire des enregistrements depuis un `TDataSet`, du JSON ou des événements.
- Guide pratique : facture avec des données maître-détail.

Modèle de rapport

Un rapport (TsgcReport) est un arbre de pages, de bandes, d'éléments et de styles, construit sur `TCollection`, il se sérialise donc comme un DFM tel n'importe quel autre composant. Le même arbre peut aussi être chargé depuis et enregistré vers JSON, voir Définitions de rapport JSON.

Unités

`TsgcReport.Units` (`TsgcReportUnits` : `ruMillimeters`, `ruPoints`, `ruInches`) est l'unité dans laquelle chaque `Left`, `Top`, `Width`, `Height` et marge est saisi. En interne, la mise en page et l'export PDF fonctionnent toujours en points (1/72 de pouce) ; le rapport convertit à la lecture et à l'écriture de ces propriétés.

Pages

`TsgcReportPage` (dans `TsgcReport.Pages`) possède `PaperSize` (`rpsA3`, `rpsA4`, `rpsA5`, `rpsLetter`, `rpsLegal`, `rpsCustom`), `Orientation`, `Width` et `Height` (utilisées uniquement lorsque `PaperSize` vaut `rpsCustom`), `Margins`, `Columns` et `ColumnSpacing`. Un rapport peut avoir plusieurs définitions de page ; les sous-rapports `rbtDetailData` référencent une page par son nom (`TsgcReportItem.SubreportPageName`).

Bandes

`TsgcReportBand` (dans `TsgcReportPage.Bands`) possède un `BandType` qui détermine quand et à quelle fréquence elle s'imprime :

BandType	S'imprime
rbtReportTitle	Une fois, avant le premier en-tête de page.
rbtReportSummary	Une fois, après le dernier groupe et les données détail.
rbtPageHeader	En haut de chaque page.
rbtPageFooter	En bas de chaque page.
rbtColumnHeader	En haut de chaque colonne, sur une page multi-colonnes.
rbtColumnFooter	En bas de chaque colonne.
rbtGroupHeader	Une fois par valeur distincte de `GroupExpression`, `RepeatOnEveryPage` pouvant la répéter lorsqu'un groupe s'étend sur un saut de page.
rbtGroupFooter	Une fois à la fin de chaque groupe, une fois ses agrégats complétés.
rbtMasterData	Parcourt les enregistrements de `DataSourceName` ; n'imprime rien elle-même.
rbtDetailData	Parcourt les enregistrements de `DataSourceName` sous l'enregistrement maître courant.
rbtChild	Imprimée immédiatement après sa bande parente via `ChildBandName`, partageant le même enregistrement.

Autres propriétés de bande : `Height`, `CanGrow` et `CanShrink` (agrandir ou rétrécir pour s'adapter à leur élément le plus haut), `KeepTogether` (ne pas découper la bande à travers un saut de page), `StartNewPage`, `PrintOnFirstPage` / `PrintOnLastPage`, `VisibleExpression` et `StyleName` (un style par défaut pour ses éléments).

Éléments et types

`TsgcReportItem` (dans `TsgcReportBand.Items`) possède un `Kind` (`TsgcReportItemKind`) :

Kind	Sous-options	Notes
rikText	`Text` (`TsgcReportTextOptions`)	`Text.Text` est un modèle de texte, voir Expressions et modèles de texte.
rikImage	`Image` (`TsgcReportImageOptions`)	Octets d'image intégrée, sérialisés sous forme de `PictureData` .
rikLine	—	Un trait horizontal ou vertical.
rikShape	`Shape` (`TsgcReportShapeOptions`)	`ShapeKind` : rectangle, rectangle arrondi, ellipse, ligne.
rikBarcode	`Barcode` (`TsgcReportBarcodeOptions`)	11 symbologies, voir `TsgcReportBarcodeSymbology` ; `Barcode.Value` est un modèle de texte.
rikSubreport	`SubreportPageName`	Exécute sur place une autre page du même rapport.
rikPageBreak	—	Force une nouvelle page.

Propriétés communes des éléments : `Left`, `Top`, `Width`, `Height`, `StyleName`, `Visible` / `VisibleExpression`, `CanGrow` / `CanShrink`, `AltText` (accessibilité) et `TagRole` (voir Accessibilité).

Styles

`TsgcReportStyle` (dans `TsgcReport.Styles`) regroupe `Font`, `BackColor`, `Frame`, `Padding`, `HAlign` et `VAlign` sous un `Name` qu'une bande ou un élément référence par `StyleName`. Les couleurs sont des `TsgcReportColor`, un cardinal neutre vis-à-vis de la plateforme `$00RRGGBB` (et non `Graphics.TColor`) ; `sgcReportColorNone`, `sgcReportColorBlack` et `sgcReportColorWhite` sont prédéfinis.

Expressions et modèles de texte

sgcReports compile chaque expression une fois, puis l'évalue une fois par enregistrement. Les valeurs sont des Variants.

Champs, variables et portée

Un champ s'écrit `Source.Field`, où `Source` est le `Name` d'une source de données ajoutée au moteur (`TsgcReportEngine.AddDataSource`) et `Field` est l'un de ses noms de champ. Un identifiant nu se résout comme une variable du moteur (`TsgcReportEngine.Variables`, paires `Name=Value`), également accessible depuis `OnGetVariable`. Deux identifiants sont toujours disponibles : `Page` (numéro de page courant) et `TotalPages`, résolu seulement après la dernière page (différé, deux passes, "page X sur Y").
Un pied de groupe ou un pied de page voit les valeurs de champ du **dernier** enregistrement de son groupe ou de sa page, et non l'enregistrement suivant qui a déclenché la rupture.

Opérateurs

Arithmétiques `+ - * /`, de comparaison `= <> < <= > >=`, logiques `and or not`, concaténation de chaînes avec `+`, et parenthèses pour le regroupement.

Fonctions

Fonction	Description
IIF(cond, a, b)	a lorsque cond est vrai, sinon b.
IsNull(v)	True lorsque v est Null ou vide.
Coalesce(v1, v2, ...)	Premier argument qui n'est pas Null.
Upper(s)	Majuscules.
Lower(s)	Minuscules.
Trim(s)	Supprime les espaces en début et fin.
Length(s)	Longueur de la chaîne.
Copy(s, from, count)	Sous-chaîne.
Pos(sub, s)	Position basée sur 1, 0 si non trouvé.
Replace(s, from, to)	Remplace chaque occurrence.
FormatFloat(fmt, v)	Formate un nombre.
FormatDateTime(fmt, v)	Formate une date ou une heure.
Str(v)	Convertit en chaîne.
Round(v)	Arrondit à l'entier le plus proche.
Trunc(v)	Tronque vers zéro.

Abs(v)	Valeur absolue.
Int(v)	Partie entière.
MinOf(a, b)	Le plus petit des deux.
MaxOf(a, b)	Le plus grand des deux.
Now()	Date et heure actuelles.
Date()	Date actuelle.
Year(v)	Année d'une date.
Month(v)	Mois d'une date.
Day(v)	Jour d'une date.

Agrégats

`Sum`, `Count`, `Avg`, `Min` et `Max` prennent un argument, par exemple `Sum(Lines.Qty * Lines.Price)`. Un agrégat s'accumule sur la portée dans laquelle il est imprimé : dans un pied de groupe, il se réinitialise à chaque groupe, dans un pied de page, il se réinitialise à chaque page, dans le résumé du rapport, il s'exécute pour tout le rapport.

Sémantique de Null

Un champ manquant ou une variable non définie s'évalue à Null, qui se propage à travers l'arithmétique (`Null + 1` vaut Null) et se compare comme n'étant ni égal ni différent de toute valeur non-Null, conformément aux règles des Variants Delphi. Utilisez `IsNull` ou `Coalesce` pour substituer une valeur par défaut avant le formatage.

Modèles de texte

`TsgcReportTextTemplate` (utilisé par `TsgcReportItem.Text.Text` et `TsgcReportItem.Barcode.Value`) est du texte littéral avec des expressions intégrées entre crochets, par exemple `'Invoice [Invoices.Number], total [Sum(Lines.Qty * Lines.Price)]'`. Tout ce qui est en dehors de `[ ]` est recopié tel quel, de sorte qu'une valeur de code-barres telle que `STOCK-2026-09` sans crochets est imprimée littéralement, zéros non significatifs inclus. `TsgcReportTextOptions.DisplayFormat` (ou `Barcode.DisplayFormat` le cas échéant) formate le résultat évalué d'un modèle à expression unique de la même manière que le ferait `FormatFloat` ou `FormatDateTime`.

Sources de données

Une source de données est un descendant de `TsgcReportDataSource`, ajouté à un TsgcReportEngine avec `AddDataSource`. Sa propriété `SourceName` (et non le `Name` du composant) est ce que les expressions de bande référencent comme `Source.Field`, et une bande `rbtMasterData` ou `rbtDetailData` la référence sous le même nom via `DataSourceName`.

Types

Classe	Lit depuis
TsgcReportDataSetSource	N'importe quel `TDataSet` (VCL `TClientDataSet`, FireDAC, dbExpress, ...).
TsgcReportJSONSource	Un tableau JSON d'objets, depuis une chaîne, un fichier ou un flux.
TsgcReportEventSource	Gestionnaires d'événements `OnFirst`, `OnNext`, `OnEof` et `OnGetValue`, pour des données qui n'ont aucune forme de jeu de données ou JSON, comme un appel REST paginé.

Maître-détail

`TsgcReportDataSource` (la base commune) possède `MasterSource`, `MasterFields` et `DetailFields` (séparés par des points-virgules, en même nombre des deux côtés). Lorsqu'une bande détail avec `DataSourceName` pointant vers la source détail est imbriquée sous une bande maître, le moteur redémarre la source détail et l'associe à l'enregistrement maître courant sur ces champs avant chaque itération détail. C'est ainsi que la démo de facture relie les lignes de facture à leur facture, voir Guide pratique : facture avec des données maître-détail.

Enregistrements JSON

`TsgcReportJSONSource.RecordsPath` est un chemin en notation pointée vers le tableau d'enregistrements ; laissez-le vide lorsque la racine JSON est elle-même le tableau. Les noms de champs sont découverts à partir des membres du premier enregistrement.

Sécurité des threads

Une source de données n'est pas partagée entre des rapports préparés simultanément ; chaque thread doit utiliser ses propres instances de source (et son propre moteur). Voir Sécurité des threads et utilisation serveur.

Groupes, en-têtes, pieds de page et colonnes

Groupes

Une bande `rbtGroupHeader` avec une `GroupExpression` (par exemple `'Sales.Region'`) démarre un nouveau groupe chaque fois que la valeur de cette expression change ; sa bande `rbtGroupFooter` correspondante ferme le groupe une fois ses agrégats finalisés. Définissez `RepeatOnEveryPage` sur l'en-tête pour la réimprimer lorsque le groupe s'étend sur un saut de page, et `StartNewPage` pour forcer chaque groupe sur sa propre page (utilisé pour une page par facture). Les groupes s'imbriquent : une page peut avoir plusieurs paires `rbtGroupHeader` / `rbtGroupFooter`, évaluées dans l'ordre d'apparition des bandes.

En-têtes et pieds de page et de colonne

`rbtPageHeader` et `rbtPageFooter` s'impriment sur chaque page. `rbtColumnHeader` et `rbtColumnFooter` s'impriment sur chaque colonne lorsque `TsgcReportPage.Columns` est supérieur à 1. `PrintOnFirstPage` et `PrintOnLastPage` (par défaut `True`) suppriment un en-tête ou un pied de page sur la première ou la dernière page.

Maintien groupé

`TsgcReportBand.KeepTogether := True` force l'ensemble de la bande sur une seule page : si elle ne tient pas dans l'espace restant sur la page courante, le moteur démarre une nouvelle page avant de l'imprimer, plutôt que de la découper. Utilisé sur les blocs de totaux et d'autres bandes qui ne doivent pas être coupées par un saut de page.

Colonnes

`TsgcReportPage.Columns` (par défaut 1) et `ColumnSpacing` disposent les bandes maître et détail en colonnes de type journal ; le moteur remplit une colonne avant de passer à la suivante, en imprimant `rbtColumnHeader` / `rbtColumnFooter` à chaque limite de colonne.

Sous-rapports

Un élément `rikSubreport` (`TsgcReportItem.SubreportPageName`) exécute une autre `TsgcReportPage` du même rapport sur place, à la position de l'élément. La page de sous-rapport peut avoir ses propres bandes et ses propres références de source de données, et est préparée dans le cadre de la même exécution.

Page X sur Y

`Page` et `TotalPages` sont les deux identifiants réservés au texte de pagination, typiquement `'Page [Page] of [TotalPages]'` dans une bande `rbtPageFooter`. `TotalPages` est différé : le moteur exécute tout le rapport une première fois pour compter les pages, puis résout chaque expression différée lors d'une seconde passe avant que les pages ne soient transmises à un export.

Polices

`TsgcReportFontManager` (unité `sgcReports_Fonts`) découvre, mesure et applique un repli des polices à la fois pour le moteur de mise en page (retour à la ligne, `CanGrow`) et pour l'export PDF. Il n'a aucun état global ; chaque moteur et chaque export référence un gestionnaire via sa propriété `FontManager`.

Gestionnaire par défaut

Lorsque `FontManager` n'est pas affecté, le moteur et les exports utilisent le `sgcReportDefaultFontManager` à l'échelle du processus, une instance partagée construite à partir des polices installées sur le système, créée une seule fois lors de la première utilisation. Affectez votre propre `TsgcReportFontManager` lorsqu'un rapport a besoin de polices que le gestionnaire par défaut ne possède pas, ou pour garder le chargement des polices isolé entre des rapports concurrents.

Découverte

Le gestionnaire analyse les dossiers de polices de la plateforme : le dossier Fonts de Windows, les répertoires de polices standard de Linux, les emplacements de polices système de macOS et iOS, et les chemins de polices et de ressources d'Android. Les polices TrueType découvertes sont indexées par nom de famille, graisse et style (gras, italique).

Polices standard et repli

Les 14 polices PDF standard (Helvetica, Times, Courier et leurs variantes gras et italique, plus Symbol et ZapfDingbats) sont mappées par défaut ; un style avec `Font.Name := 'Helvetica'` s'affiche sans intégration sauf si l'export a besoin d'une police intégrable (PDF/A, ou un caractère que la police de base ne peut pas représenter). Le gestionnaire substitue alors une police TrueType à la famille demandée, et se replie encore davantage, glyphe par glyphe, sur une autre police installée lorsque la police courante n'a pas de glyphe pour un caractère, ce qui permet à une police de base latine de tout de même rendre du texte CJK ou cyrillique présent dans les données.

Intégration

L'export PDF intègre les polices TrueType réellement utilisées, réduites en sous-ensemble aux glyphes référencés par le document, sauf lorsque PDF/A exige l'intégration complète d'une police déjà partiellement intégrée. `TsgcReportPDFExport.FontManager` résout les polices des éléments qui n'ont pas de police explicite et, pour une sortie PDF/A, les substituts TrueType des 14 polices standard via `TsgcReportFontManager.ResolveEmbeddable`.

Plateformes mobiles et sans interface

iOS n'offre aucun dossier de polices système lisible depuis une application en bac à sable, et un démon Linux64 peut s'exécuter sur une image minimale avec peu de polices installées. Dans les deux cas, livrez une police TrueType (par exemple DejaVu Sans, sous licence libre de copyleft) à côté de l'exécutable ou dans le paquet de l'application, et laissez la découverte de polices s'y replier ; voir Guide pratique : un rapport FMX mobile.

Sécurité des threads et utilisation serveur

sgcReports est conçu pour la génération de rapports serveur et web, où de nombreux rapports sont préparés et exportés en même temps sur des threads différents.

Règle

Un moteur, son rapport, ses sources de données et ses exports ne sont pas partagés entre threads. Chaque thread de travail doit créer et utiliser son propre `TsgcReportEngine`, son propre `TsgcReport` (ou partager une définition de rapport en lecture seule et ne cloner que le moteur, puisque le moteur détient l'état d'exécution), ses propres instances de source de données, et ses propres composants d'export. Rien dans les classes de modèle, de mise en page ou d'export n'utilise de verrou global autour de l'état propre d'un rapport, de sorte que deux threads partageant un moteur ou un export corrompent les deux exécutions.

Ce qui peut être partagé

Un `TsgcReportFontManager` peut être partagé en lecture seule entre threads une fois ses polices découvertes, y compris le `sgcReportDefaultFontManager` partagé au niveau du processus : la découverte et la mesure des polices ne modifient aucun état par appel dont d'autres threads dépendraient. `TsgcReportFontManager.ResolveEmbeddable` ne modifie pas non plus le paramètre `EmbedStandardFonts` du gestionnaire, il est donc également sûr de l'appeler sur un gestionnaire partagé entre threads. Une `TsgcReportDefinition` (ou les fonctions simples `sgcReportLoadFromJSON` / `sgcReportSaveToJSON`) peut être appelée depuis n'importe quel thread, à condition que chaque appel utilise sa propre instance de `TsgcReport`.

Modèle : un pool de workers

Chargez une seule définition de rapport JSON partagée en lecture seule au démarrage. Chaque thread de travail effectue alors, pour chaque tâche : créer un `TsgcReport`, appeler `sgcReportLoadFromJSON` dessus, créer un `TsgcReportEngine` et les sources de données pour cette tâche, les ajouter, `Prepare`, exporter, puis libérer tout ce que le thread a créé. C'est la conception utilisée par la démo démon Linux, vérifiée par un test de charge à 8 threads sur Win32, Win64 et Linux64.

Applications serveur sous Windows

`sgcReportsConfigureForServer` (unité `sgcReports_Types`) est un réglage optionnel pour les serveurs qui préparent et exportent des rapports sur de nombreux threads à la fois. Sous Windows, avec Delphi 2006 ou une version ultérieure, il définit `System.NeverSleepOnMMThreadContention` à `True`, de sorte que le gestionnaire de mémoire du RTL attend activement au lieu de se mettre en veille lorsque deux threads se disputent la même taille de bloc ; il n'a aucun effet sur les autres plateformes ni avec les compilateurs plus anciens. sgcReports ne l'appelle jamais lui-même, il n'a donc aucun effet tant que l'application ne l'appelle pas elle-même, et une fois qu'elle le fait, le réglage s'applique à l'ensemble du processus, à chaque allocation mémoire de l'application, pas seulement aux exports de rapports.
Appelez-le une seule fois, au démarrage de l'application, avant la création de tout thread de travail, dans un serveur multithread. Mesuré avec 8 threads exportant chacun 25 rapports sur Win64, le débit est passé d'environ x1,8 à environ x5 par rapport à un seul thread, sur deux exécutions (de 13,6 s à 4,4 s, et de 7,8 s à 2,8 s). Ce réglage fait aussi consommer plus de CPU aux threads pendant leur attente ; laissez-le donc désactivé dans les applications monothread.

```
program sgcReportsServerStartup;

{$APPTYPE CONSOLE}

uses
  sgcReports_Types;

begin
```

```
  // call once, at application start up, before any worker thread is created
  sgcReportsConfigureForServer;
end.
```

Point de terminaison HTTP

TsgcReportHTTPEndpoint enregistre un moteur longue durée par nom de rapport et sert les requêtes concurrentes qui lui sont adressées ; voir le sujet de ce composant pour savoir comment il synchronise l'accès au moteur partagé et à son cache de pages préparées.

Accessibilité (PDF/UA)

TsgcReportPDFExport.PDFUA vaut `True` par défaut : chaque PDF exporté est balisé pour PDF/UA-1 (ISO 14289-1) sauf désactivation explicite. Le balisage écrit un arbre de structure (titres, paragraphes, tableaux avec lignes d'en-tête, figures avec texte alternatif, artefacts), le `/Lang` du document, le titre et l'identifiant `pdfuaid`, et intègre chaque police utilisée (les polices standard sont remplacées par un substitut TrueType via le gestionnaire de polices, puisqu'un PDF balisé ne peut pas s'appuyer sur une police de base non intégrée).

Règles de balisage automatique

Le rôle de chaque élément dans l'arbre de structure est dérivé de sa bande et de son type sauf s'il est remplacé :

Source	Rôle par défaut
Élément texte dans une bande `rbtReportTitle`	Titre (le niveau dépend de `TagRole` s'il est défini, sinon un titre de premier niveau).
Élément texte dans une bande `rbtGroupHeader`	Titre.
Autres éléments texte	Paragraphe.
Un tableau construit à partir de lignes de détail répétées sous une bande d'en-tête de colonne	Tableau, avec les cellules de la bande d'en-tête de colonne marquées comme cellules d'en-tête de tableau.
Éléments image et code-barres	Figure, utilisant `AltText` comme texte alternatif.
Lignes et formes décoratives	Artefact (exclu de l'ordre de lecture présenté aux technologies d'assistance).

TagRole

Définissez `TsgcReportItem.TagRole` pour remplacer le rôle automatique : `rtrAuto` (par défaut), `rtrHeading1`, `rtrHeading2`, `rtrHeading3`, `rtrParagraph`, `rtrFigure`, `rtrArtifact`, `rtrTableHeader` ou `rtrTableCell`. Définissez toujours `AltText` sur un élément balisé `rtrFigure`, car le texte alternatif est ce que PDF/UA exige pour qu'une figure soit conforme.

Ordre de lecture

L'arbre de structure suit l'ordre dans lequel les éléments sont imprimés, de haut en bas, de gauche à droite au sein d'une bande, ce qui, pour une mise en page de rapport normale, correspond déjà à l'ordre de lecture voulu ; il n'y a pas d'étape de tri manuel séparée.

Vérification avec veraPDF

Voir Guide pratique : un PDF accessible validé avec veraPDF pour la commande qui vérifie un fichier PDF/UA-1 par rapport au profil ISO.

PDF/A

TsgcReportPDFExport.PDFA (`TsgcReportPDFAMode`) sélectionne la sortie d'archivage : `rpaNone` (PDF simple par défaut), `rpaA3b` (PDF/A-3b, reproductibilité visuelle) ou `rpaA3a` (PDF/A-3a, ajoute des exigences de balisage et de langue en plus de A-3b). PDF/A-3 permet d'intégrer des fichiers arbitraires dans le document, ce qui en fait le choix d'archivage pour la facturation électronique et les flux similaires où la source lisible par machine (par exemple le XML de la facture) voyage à l'intérieur du même PDF.

Combinaison avec PDF/UA

`PDFA = rpaA3a` et `PDFUA = True` peuvent être définis ensemble : les deux exigent un arbre de structure balisé, une langue déclarée et des polices intégrées, de sorte que l'export produit un seul arbre de structure qui satisfait les deux revendications de conformité. C'est la combinaison utilisée pour les factures et autres documents qui doivent être à la fois accessibles et archivables ; voir le guide pratique de la facture et le guide pratique veraPDF.

Polices

PDF/A exige que chaque police soit intégrée. Lorsqu'un style utilise par nom une police standard des 14, l'export substitue et intègre une police TrueType via le gestionnaire de polices plutôt que de s'appuyer sur la police de base, ce que PDF/A interdit comme seule représentation du texte ; voir Polices.

Sortie reproductible

`FixedCreationDate` et `FixedDocumentID` sur `TsgcReportPDFExport` fixent les dates de création et de modification et l'identifiant de fichier à une valeur fournie par l'appelant plutôt qu'à l'heure actuelle, de sorte que des exports répétés du même rapport produisent des fichiers identiques au bit près ; utile pour les tests de fichiers de référence et la déduplication d'archives. Définir l'un ou l'autre désactive les flux d'objets PDF/A afin que les valeurs restent modifiables.

TsgcReport

Unité : sgcReports_Model
La définition du rapport : pages, bandes, éléments et styles. Basé sur `TComponent` et construit sur `TCollection`, il se sérialise donc comme un DFM tel n'importe quel autre composant et peut aussi être chargé depuis ou enregistré vers JSON, voir Définitions de rapport JSON. Voir Modèle de rapport pour la description complète des pages, bandes, éléments et styles.

Propriétés

Propriété	Type	Description
Unités	TsgcReportUnits	Unité dans laquelle chaque propriété géométrique (Left, Top, Width, Height, marges) est saisie : ruMillimeters, ruPoints ou ruInches.
Pages	TsgcReportPages	Collection de TsgcReportPage.
Styles	TsgcReportStyles	Collection de TsgcReportStyle, référencée par StyleName depuis les bandes et les éléments.
Titre	string	Titre du rapport ; également le titre par défaut du document PDF.
Auteur	string	Auteur du rapport.

Exemple

Voir Prise en main pour un exemple complet de l'API de modèle (styles, une page, une bande, un élément texte) et Guide pratique : facture avec des données maître-détail pour un rapport complet.

TsgcReportEngine

Unité : sgcReports_Engine
Exécute un TsgcReport face à ses sources de données : parcourt les relations maître-détail et les groupes, évalue les expressions et les agrégats, et produit la liste d'affichage des pages préparées que les exports et les composants d'aperçu lisent. Basé sur `TComponent`.

Propriétés

Propriété	Type	Description
Report	TsgcReport	La définition du rapport à exécuter.
Variables	TStrings	Paramètres utilisateur sous forme de Name=Value, exposés aux expressions comme identifiants nus.
FontManager	TsgcReportFontManager	Non publiée (public). Nil utilise le sgcReportDefaultFontManager partagé. Voir Polices.
OnBeforePrintBand	TsgcReportBeforePrintBandEvent	Se déclenche avant l'impression d'une bande ; permet au gestionnaire de modifier la visibilité ou les valeurs.
OnGetVariable	TsgcReportGetVariableEvent	Résout un identifiant introuvable dans Variables.

Méthodes

Méthode	Retourne	Description
AddDataSource(aSource: TsgcReportDataSource)	—	Enregistre une source de données sous son SourceName ; les expressions de bande la référencent ensuite comme SourceName.Field.
RemoveDataSource(aSource: TsgcReportDataSource)	—	Désenregistre une source de données.
ClearDataSources	—	Supprime toutes les sources de données enregistrées.
Prepare	TsgcReportPreparedPages	Exécute le rapport ; l'appelant possède et libère le résultat.
Prepare(aPages: TsgcReportPreparedPages)	—	Surcharge qui prépare dans un TsgcReportPreparedPages existant, préalablement vidé.

Exemple

```
sgcEngine.Report := sgcReport;
sgcEngine.AddDataSource(srcInvoices);
sgcEngine.AddDataSource(srcLines);
oPages := sgcEngine.Prepare;
try
  pdfExport.SaveToFile(oPages, 'invoice.pdf');
```

```
finally
  oPages.Free;
end;
```

TsgcReportDataSetSource

Unité : sgcReports_Data
Source de données qui lit des enregistrements depuis n'importe quel `TDataSet` (`TClientDataSet`, FireDAC, dbExpress, etc.). Descend de `TsgcReportDataSource`, qui fournit les propriétés maître-détail communes `MasterSource`, `MasterFields` et `DetailFields`, voir Sources de données.

Propriétés

Propriété	Type	Description
DataSet	TDataSet	Le jeu de données à lire. Laissé fermé, il est ouvert et fermé par la source autour d'une exécution ; déjà ouvert, son état et sa position sont restaurés ensuite.
SourceName	string	Hérité. Le nom que les expressions de bande utilisent comme Source.Field ; indépendant du Name du composant.
MasterSource	TsgcReportDataSource	Hérité. La source maître dont celle-ci est un détail.
MasterFields	string	Hérité. Noms des champs maîtres séparés par des points-virgules.
DetailFields	string	Hérité. Noms des champs détail séparés par des points-virgules, en même nombre que MasterFields.

Exemple

```
srcInvoices := TsgcReportDataSetSource.Create(Self);
srcInvoices.SourceName := 'Invoices';
srcInvoices.DataSet := cdsInvoices;

srcLines := TsgcReportDataSetSource.Create(Self);
srcLines.SourceName := 'Lines';
srcLines.DataSet := cdsLines;
srcLines.MasterSource := srcInvoices;
srcLines.MasterFields := 'Number';
srcLines.DetailFields := 'Invoice';
```

Voir Guide pratique : facture avec des données maître-détail pour le rapport complet qui utilise cette paire de sources.

TsgcReportJSONSource

Unité : sgcReports_Data
Source de données qui lit des enregistrements depuis un tableau JSON d'objets, fourni sous forme de chaîne, de nom de fichier ou de flux. Descend de `TsgcReportDataSource`, elle prend donc également en charge `MasterSource`, `MasterFields` et `DetailFields`.

Propriétés

Propriété	Type	Description
JSON	string	Le texte JSON, lorsqu'il est défini directement.
FileName	string	Un fichier depuis lequel lire le JSON.
RecordsPath	string	Chemin en notation pointée vers le tableau d'enregistrements ; vide lorsque la racine JSON est elle-même le tableau.

Méthodes

Méthode	Description
LoadFromStream(aStream: TStream)	Analyse le JSON UTF-8 du flux immédiatement ; remplace JSON et FileName jusqu'à ce que l'un des deux change.

Exemple

sales.json, un tableau sous "sales" :

```
{"sales": [
  {"region": "North", "customer": "Acme", "amount": 1200.50},
  {"region": "North", "customer": "Globex", "amount": 340.00}
]}
```

```
srcSales := TsgcReportJSONSource.Create(Self);
srcSales.SourceName := 'Sales';
srcSales.FileName := 'sales.json';
srcSales.RecordsPath := 'sales';
sgcEngine.AddDataSource(srcSales);
```

Voir Guide pratique : un rapport groupé avec totaux.

TsgcReportEventSource

Unité : sgcReports_Data
Source de données pilotée entièrement par des événements, pour des enregistrements qui n'ont aucune forme `TDataSet` ou JSON, par exemple une réponse REST paginée récupérée à la demande. Sans gestionnaire `OnEof`, la source n'a aucun enregistrement.

Propriétés

Propriété	Type	Description
FieldNames	TStrings	Les noms de champs que cette source expose, puisqu'il n'y a aucun jeu de données ni objet JSON pour les découvrir.
OnFirst	TNotifyEvent	Déclenché pour réinitialiser l'itération au premier enregistrement.
OnNext	TNotifyEvent	Déclenché pour avancer à l'enregistrement suivant.
OnEof	TsgcReportEofEvent	`procedure(Sender: TObject; var aEof: Boolean) of object`. Définissez aEof à True lorsqu'il n'y a plus d'enregistrements.
OnGetValue	TsgcReportGetValueEvent	`procedure(Sender: TObject; const aFieldName: string; var aValue: Variant) of object`. Retourne la valeur d'un champ de l'enregistrement courant.

Exemple

`OnFirst`, `OnNext`, `OnEof` et `OnGetValue` sont des événements de type pointeur de méthode (déclarés `of object`), ils prennent donc des méthodes nommées d'un formulaire ou d'une classe, et non des procédures anonymes. Cet exemple utilise deux champs, `FIndex` et `FRows`, déclarés sur le formulaire englobant avec les quatre méthodes de gestion ci-dessous.

```
procedure TForm1.DoFirst(Sender: TObject);
begin
  FIndex := -1;
end;

procedure TForm1.DoNext(Sender: TObject);
begin
  Inc(FIndex);
end;

procedure TForm1.DoEof(Sender: TObject; var aEof: Boolean);
begin
  aEof := FIndex >= High(FRows);
end;

procedure TForm1.DoGetValue(Sender: TObject; const aFieldName: string;
  var aValue: Variant);
begin
  if aFieldName = 'region' then aValue := FRows[FIndex][0]
  else if aFieldName = 'amount' then aValue := StrToFloat(FRows[FIndex][1]);
end;

procedure TForm1.SetupEventSource;
begin
```

```
  FRows[0] := ['North', '1200.5'];
  FRows[1] := ['South', '340'];
  FIndex := -1;

  srcEvt.FieldNames.CommaText := 'region,amount';
  srcEvt.OnFirst := DoFirst;
  srcEvt.OnNext := DoNext;
  srcEvt.OnEof := DoEof;
  srcEvt.OnGetValue := DoGetValue;
end;
```

`FRows` est déclaré comme `array [0..1] of TArray<string>`. Ce modèle fonctionne sur toutes les versions de Delphi prises en charge, de Delphi 7 à Delphi 13, puisqu'il affecte des pointeurs de méthode ordinaires plutôt que des méthodes anonymes.

TsgcReportPDFExport

Unité : sgcReports_Export_PDF
Rend les pages préparées via sgcPDF : polices TrueType intégrées et en sous-ensemble, polices de base 14, texte de droite à gauche, rotation, découpage, texte justifié, formes, images et codes-barres vectoriels. Produit par défaut une sortie balisée PDF/UA-1 et, en option, une sortie d'archivage PDF/A-3. Basé sur `TComponent`.

Propriétés

Propriété	Type	Par défaut	Description
Compress	Boolean	True	Flux d'objets et référence croisée compressée ; les flux de contenu sont toujours compressés.
PDFA	TsgcReportPDFAMode	rpaNone	rpaNone, rpaA3b ou rpaA3a. Voir PDF/A.
PDFUA	Boolean	True	Écrit l'arbre de structure PDF/UA-1 ; combinable avec PDFA. Voir Accessibilité.
Title / Author / Subject / Keywords	string	—	Métadonnées du document PDF.
Language	string	—	Langue naturelle BCP 47, écrite par l'export balisé.
Creator / Producer	string	—	Métadonnées du document PDF.
OnAfterRender	TsgcReportPDFDocumentEvent	—	Se déclenche une fois que toutes les pages sont rendues, avant l'enregistrement du document.
FontManager	TsgcReportFontManager	—	Non publiée (public). Voir Polices.
FixedCreationDate	TDateTime	0	Non publiée (public). Fixe les dates de création et de modification PDF/A pour une sortie identique au bit près.
FixedDocumentID	string	—	Non publiée (public). Fixe le /ID au MD5 de cette valeur.

Méthodes

Méthode	Description
SaveToStream(aPages: TsgcReportPreparedPages; aStream: TStream)	Rend les pages préparées dans un flux.
SaveToFile(aPages: TsgcReportPreparedPages; const aFileName: string)	Rend les pages préparées dans un fichier.

Exemple : PDF accessible et archivable

```
pdfExport.PDFA := rpaA3a;
pdfExport.PDFUA := True;
pdfExport.Language := 'en-US';
pdfExport.Title := sgcReport.Title;
oPages := sgcEngine.Prepare;
try
  pdfExport.SaveToFile(oPages, 'invoice_pdfa3a_ua1.pdf');
finally
  oPages.Free;
end;
```

TsgcReportXLSXExport

Unité : sgcReports_Export_XLSX
Exporte les pages préparées sous forme d'une feuille de calcul d'un classeur .xlsx. Descend de `TsgcReportGridExport`, qui transforme les pages préparées en une grille (une ligne par instance de bande, colonnes fusionnées à partir des positions des éléments dans une tolérance) partagée avec TsgcReportCSVExport. Le paquet est un ZIP avec des entrées STORED (Content_Types, relations, classeur, une feuille et des styles) utilisant des chaînes inline, écrit sans aucune bibliothèque zip externe.

Propriétés (propres)

Propriété	Type	Description
SheetName	string	Nom de la feuille de calcul.
FixedDateTime	TDateTime	Heure de modification stockée dans les entrées ZIP ; 0 (par défaut) utilise l'heure actuelle. Définissez-la pour une sortie identique au bit près.

Propriétés (héritées de TsgcReportGridExport)

Propriété	Type	Par défaut	Description
ExcludePageBands	Boolean	True	Exclut de la grille les bandes en-tête de page, pied de page, en-tête de colonne et pied de colonne.
IncludeBandTypes	TsgcReportBandTypes	rbtReportTitle..rbtChild	Quels types de bandes contribuent des lignes.
PageFrom / PageTo	Integer	0	Plage de pages basée sur 1 ; 0 signifie toutes les pages.
DetectNumbers	Boolean	True	Les cellules texte qui s'analysent comme un nombre sont écrites comme des cellules numériques.
ColumnTolerance	Double	—	Les positions gauches plus proches que cette valeur (points) sont traitées comme la même colonne.
DecimalSeparator / ThousandSeparator	Char	locale	Non publiée (public). Utilisée par DetectNumbers.

Méthodes (héritées)

Méthode	Description
SaveToStream(aPages, aStream)	Écrit le classeur dans un flux.
SaveToFile(aPages, const aFileName)	Écrit le classeur dans un fichier.

BuildGrid(aPages): TsgcReportGrid	Construit la grille intermédiaire ; l'appelant possède le résultat.

Exemple

```
xlsxExport.SheetName := 'Sales';
xlsxExport.SaveToFile(oPages, 'sales.xlsx');
```

TsgcReportCSVExport

Unité : sgcReports_Export_CSV
Exporte les pages préparées en texte CSV, en utilisant la même grille basée sur la mise en page que TsgcReportXLSXExport (voir les propriétés `TsgcReportGridExport` sur cette page : ExcludePageBands, IncludeBandTypes, PageFrom, PageTo, DetectNumbers, ColumnTolerance).

Propriétés (propres)

Propriété	Type	Par défaut	Description
Delimiter	Char	','	Délimiteur de champ.
QuoteChar	Char	'"'	Caractère de citation ; #0 désactive la mise entre guillemets.
LineBreak	string	—	Séparateur de ligne.
WriteBOM	Boo-lean	True	Écrit une marque d'ordre des octets (BOM) UTF-8 au début du fichier.

Méthodes

Méthode	Retourne	Description
GetCSV(aPages: TsgcReportPrepared-Pages)	string	Construit le texte CSV en mémoire.
SaveToStream(aPages, aStream)	—	Écrit le CSV dans un flux.
SaveToFile(aPages, const aFileName)	—	Hérité de TsgcReportGridExport ; écrit le CSV dans un fichier.

Exemple

```
csvExport.Delimiter := ';';
csvExport.SaveToFile(oPages, 'sales.csv');
```

TsgcReportImageExport

Unité : sgcReports_Export_Image
Exporte une image PNG par page, au DPI choisi, via le moteur de rendu raster sgcPDF, sur toutes les plateformes. Basé sur `TComponent`.

Propriétés

Propriété	Type	Par défaut	Description
DPI	Integer	150	Pixels par pouce des images.
PageFrom / PageTo	Integer	0	Plage de pages basée sur 1 pour SaveToFiles ; 0 signifie la première / dernière page.
BackgroundColor	TsgcReportColor	sgcReportColorWhite	Couleur de remplissage avant le rendu ; sgcReportColorNone signifie blanc.
FontManager	TsgcReportFontManager	—	Non publiée (public). Résout les polices pour les éléments sans police explicite.

Méthodes

Méthode	Retourne	Description
PageCount(aPages: TsgcReportPreparedPages)	Integer	Nombre de pages que SaveToFiles écrirait, PageFrom / PageTo appliqués.
SaveToStream(aPages, aPageIndex: Integer, aStream: TStream)	—	Écrit une page (basée sur 0, PageFrom / PageTo ignorés) en PNG.
SaveToFiles(aPages, const aFileNamePattern: string)	—	Écrit un fichier PNG par page de la plage ; aFileNamePattern contient %d, remplacé par le numéro de page basé sur 1.
ClearCache	—	Élimine le PDF mis en cache utilisé en interne pour la rastérisation ; le prochain export le régénère.

Exemple

```
imgExport.DPI := 150;
imgExport.SaveToFiles(oPages, 'sales_%d.png');
```

TsgcReportPreviewVCL

Unité : sgcReports_Preview_VCL
Panneau d'aperçu VCL avec navigation, zoom, impression et export PDF. Descend de `TCustomPanel` et héberge en interne un `TsgcPDFViewerControl`, de sorte qu'un rapport est toujours rendu en PDF pour l'aperçu.

Propriétés

Propriété	Type	Par défaut	Description
Engine	TsgcReportEngine	—	Le moteur que Prepare exécute.
ShowToolbar	Boolean	True	Affiche la barre d'outils intégrée de navigation et de zoom.
OnPageChange	TNotifyEvent	—	Se déclenche lorsque la page affichée change.
PageCount	Integer	—	Non publiée (public), lecture seule.
CurrentPage	Integer	—	Non publiée (public). Basée sur 1, 0 lorsque rien n'est affiché.
Zoom	Double	—	Non publiée (public). Pourcentage, 100 correspond à la taille réelle.
FontManager	TsgcReportFontManager	—	Non publiée (public). Voir LoadPrepared.
Viewer	TsgcPDFViewerControl	—	Non publiée (public), lecture seule. Le visualiseur sgcPDF sous-jacent.

Méthodes

Méthode	Description
Prepare	Exécute Engine.Prepare et affiche le résultat.
LoadPrepared(aPages: TsgcReportPreparedPages)	Affiche des pages préparées ailleurs, rendues avec FontManager, sinon Engine.FontManager, sinon un gestionnaire interne avec les polices système ; passez le gestionnaire avec lequel les pages ont été préparées.
Clear	Efface l'aperçu.
FirstPage / PriorPage / NextPage / LastPage	Navigation.
ZoomIn / ZoomOut / FitWidth / FitPage	Contrôle du zoom.
Print(const aTitle: string = '')	Imprime le document affiché.
SaveToPDF(const aFileName: string)	Écrit le PDF affiché par l'aperçu.

Exemple

```
sgcPreview.Engine := sgcEngine;
sgcPreview.Prepare;
lblStatus.Caption := Format('%d pages prepared', [sgcPreview.PageCount]);
```

TsgcReportPreviewFMX

Unité : sgcReports_Preview_FMX
Aperçu FireMonkey pour ordinateur de bureau et mobile, reflétant TsgcReportPreviewVCL sur un `TControl` et un `TsgcPDFViewerFMX`. Fonctionne sur Windows, Android et iOS.

Propriétés

Propriété	Type	Par défaut	Description
Engine	TsgcReportEngine	—	Le moteur que Prepare exécute.
ShowToolbar	Boolean	True	Affiche la barre d'outils intégrée de navigation et de zoom.
OnPageChange	TNotifyEvent	—	Se déclenche lorsque la page affichée change.
PageCount	Integer	—	Non publiée (public), lecture seule.
CurrentPage	Integer	—	Non publiée (public). Basée sur 1.
Zoom	Double	—	Non publiée (public). Pourcentage, 100 correspond à la taille réelle.
Viewer	TsgcPDFViewerFMX	—	Non publiée (public), lecture seule.

Les méthodes et le reste de la surface (Prepare, LoadPrepared, navigation, zoom, SaveToPDF) sont identiques à TsgcReportPreviewVCL. Voir Guide pratique : un rapport FMX mobile pour un exemple complet, y compris le déploiement des polices sur iOS et Android.

TsgcReportHTTPEndpoint

Unité : sgcReports_HTTP (package sgcReportsHTML)
Sert des rapports enregistrés via un serveur HTTP sgcWebSockets : pages PDF, XLSX, CSV, PNG et un JSON d'information, avec des paramètres de requête en liste blanche et un cache de pages préparées. Nécessite sgcWebSockets édition Professional ou supérieure. Basé sur `TComponent`.

Propriétés

Propriété	Type	Par défaut	Description
Server	TsgcHTTPServer_Base	—	Le serveur HTTP à accrocher. Facultatif : HandleRequest peut être appelée à la place depuis votre propre OnCommandGet.
BasePath	string	—	Préfixe d'URL que possède le point de terminaison, par exemple /reports.
CacheSize	Integer	8	Ensembles de paramètres préparés conservés par rapport ; 0 désactive le cache.
MaxParamLength	Integer	256	Longueur maximale acceptée pour la valeur d'un paramètre de requête.
DefaultDPI / MaxDPI	Integer	96 / 300	DPI utilisé et autorisé pour la route de page PNG.
OnBeforePrepare	TsgcReportHTTPBeforePrepareEvent	—	Se déclenche avant qu'un rapport ne soit préparé pour une requête ; peut valider ou rejeter des paramètres (par exemple répondre 403).
DesignerEnabled	Boolean	False	Active les routes du concepteur (schema, definition, preview) utilisées par TsgcReportHTMLDesigner.
MaxDefinitionSize	Integer	1048576	Taille maximale acceptée du corps de la définition de rapport, en octets.
OnBeforeSaveDefinition	TsgcReportHTTPBeforeSaveDefinitionEvent	—	Se déclenche avant qu'une définition envoyée par le concepteur ne soit enregistrée.

Méthodes

Méthode	Retourne	Description
RegisterReport(const aName: string; aEngine: TsgcReportEngine)	—	Enregistre un moteur sous un nom ; le moteur n'est pas possédé. aName : lettres, chiffres, '_' et '-'.
UnregisterReport(const aName: string)	—	Désenregistre un rapport.

FindEngine(const aName: string)	TsgcReportEngine	Recherche un moteur enregistré.
GetReportParameters(const aName, aList: TStrings)	Boolean	Copie les Variables du moteur (Name=Value) d'un rapport ; False si inconnu.
ClearCache(const aName: string = '')	—	Élimine les pages préparées mises en cache d'un rapport, ou de tous les rapports lorsque aName est vide.
GetReportPath(const aName: string)	string	Chemin d'URL d'un rapport sans extension, par exemple /reports/sales.
HandleRequest(ARequestInfo, AResponseInfo)	Boolean	Sert une requête qui appartient à BasePath et retourne True ; à appeler depuis votre propre OnCommandGet lorsque Server n'est pas affecté.

Routes

Pour un rapport enregistré sous "sales" sous BasePath "/reports" : `/reports/sales.pdf`, `/reports/sales.xlsx`, `/reports/sales.csv`, `/reports/sales/page/1.png?dpi=96` et `/reports/sales/info` (JSON : nom, titre, nombre de pages, paramètres). Seuls les paramètres de requête dont le nom existe dans les Variables du moteur atteignent le rapport.

Routes du concepteur

Disponibles uniquement lorsque `DesignerEnabled` vaut `True` (chaque route répond 404 sinon), et utilisées par TsgcReportHTMLDesigner :

Méthode	Route	Description
GET	BasePath/schema.json	Schéma JSON des définitions de rapport, utilisé pour piloter l'inspecteur de propriétés du concepteur.
GET	BasePath/{name}/definition	La définition de rapport courante, au format JSON.
PUT or POST	BasePath/{name}/definition	Valide le corps, déclenche `OnBeforeSaveDefinition`, et remplace le rapport du moteur enregistré.
POST	BasePath/{name}/preview.png?page=1&dpi=96	Rend la définition présente dans le corps de la requête sans remplacer le rapport stocké ; la réponse porte l'en-tête `X-Report-Page-Count`.

Le corps de requête des routes de définition et d'aperçu doit être `application/json` et ne pas dépasser `MaxDefinitionSize` ; une requête qui échoue à l'une ou l'autre de ces vérifications, ou qui échoue à la validation du schéma, est rejetée avec un corps d'erreur JSON (`ok`, `path`, `message`), où `path` est le chemin JSON du membre fautif.
Voir Guide pratique : servir des rapports via HTTP pour un serveur complet, et Guide pratique : le concepteur de rapport web pour le concepteur lui-même.

TsgcReportHTMLViewer

Unité : sgcReports_HTML_Viewer (package sgcReportsHTML)
Visualiseur navigateur pour un rapport servi par un TsgcReportHTTPEndpoint : navigation entre les pages, un formulaire de paramètres et des liens de téléchargement, intégré dans une page sgcHTML. Descend de `TsgcHTMLComponent`.

Propriétés

Propriété	Type	Par défaut	Description
Endpoint	TsgcReportHTTPEndpoint	—	Le point de terminaison depuis lequel le visualiseur lit les routes du rapport.
ReportName	string	—	Nom du rapport à afficher, tel qu'enregistré sur le point de terminaison.
DPI	Integer	96	DPI utilisé pour les images de page.
ShowParameters	Boolean	—	Affiche un formulaire de saisie pour les paramètres du rapport (Variables du moteur).
ShowToolbar	Boolean	—	Affiche la navigation entre les pages et les liens de téléchargement.

Exemple

```
reportViewer.Endpoint := reportEndpoint;
reportViewer.ReportName := 'sales';
reportViewer.ShowParameters := True;
```

Voir Guide pratique : servir des rapports via HTTP, qui héberge le visualiseur à "/" à côté du point de terminaison à "/reports".

TsgcReportHTMLDesigner

Unité : sgcReports_HTML_Designer (package sgcReportsHTML)
Concepteur de bandes basé sur le navigateur qui modifie une définition de rapport JSON en la validant par rapport au schéma, en affichant un aperçu en direct via le même point de terminaison. Descend de `TsgcHTMLComponent`. Nécessite `TsgcReportHTTPEndpoint.DesignerEnabled := True`. Voir Guide pratique : le concepteur de rapport web.
La page qu'il affiche est un concepteur de bandes complet en glisser-déposer, et non un simple éditeur de formulaire :

- Une arborescence des pages, bandes et éléments à gauche, pilotée par la définition courante.
- Une zone de dessin de bande, en millimètres quelle que soit la propriété `Units` propre au rapport, où les éléments sont déplacés et redimensionnés au pointeur, s'alignent sur la grille et peuvent être ajustés au clavier.
- Un inspecteur de propriétés généré à partir de `schema.json`, qui suit ainsi les champs et les énumérations de la définition du rapport plutôt qu'un formulaire codé en dur.
- Save, qui envoie la définition à `{name}/definition` et met en évidence le champ fautif lorsque le serveur la rejette, à partir du chemin JSON de la réponse d'erreur.
- Preview, qui envoie la définition à `{name}/preview.png` et affiche la page rendue sans enregistrer.
- Annuler et rétablir sur l'historique de modification en mémoire, et un bouton Download JSON qui enregistre la définition courante sous forme de fichier.

Propriétés

Propriété	Type	Description
Endpoint	TsgcReportHTTPEndpoint	Le point de terminaison depuis lequel le concepteur lit les routes de schéma, de définition et d'aperçu ; doit avoir `DesignerEnabled` défini sur `True` .
ReportName	string	Nom de la définition de rapport en cours de modification.
DesignerHeight	Integer	Hauteur, en pixels CSS, du concepteur intégré, par défaut 640.

Exemple

```
reportEndpoint.DesignerEnabled := True;
reportDesigner.Endpoint := reportEndpoint;
reportDesigner.ReportName := 'sales';
```

TsgcReportBuilder

Unité : sgcReports_Fluent
API fluide code-first : styles, pages, bandes et éléments construits avec des appels chaînés qui retournent `Self` ou un sous-générateur limité à une portée, de sorte qu'un rapport entier soit un seul arbre d'expressions. Fonctionne sur Delphi 7 (une classe simple, sans méthode anonyme requise). Construit et possède son propre `TsgcReport` et `TsgcReportEngine`, sauf si un rapport existant est passé au constructeur.

Constructeurs

Constructeur	Description
Create	Crée et possède un nouveau TsgcReport.
Create(aReport: TsgcReport)	Construit dans un rapport existant, non possédé.

Méthodes

Méthode	Retourne	Description
Title / Author / Subject / Keywords / Language(const aValue: string)	TsgcReport-Builder	Définit la propriété TsgcReport correspondante.
Units(aUnits: TsgcReportUnits)	TsgcReport-Builder	Définit TsgcReport.Units.
Style(const aName: string)	TsgcReport-StyleBuilder	Recherche ou ajoute le style aName ; chaînez Font, Bold, Italic, Underline, Color, BackColor, Frame, Padding sur le résultat.
Page(const aName: string = '')	TsgcReport-PageBuilder	Ajoute une page à chaque appel ; chaînez les appels PaperSize, Margins, Band et Text/Bar-code/Image/Line.
Data(aSource: TsgcReportDataSource)	TsgcReport-Builder	Ajoute une source de données au moteur ; non possédée.
Variable(const aName, aValue: string)	TsgcReport-Builder	Définit une variable du moteur.
FontManager(aManager: TsgcReport-FontManager)	TsgcReport-Builder	Affecte le gestionnaire de polices du moteur ; non possédé.
Flush	—	Écrit les modificateurs en attente de la bande ou de l'élément courant.
Prepare	TsgcReport-Prepared-Pages	Prépare le rapport ; l'appelant possède le résul-tat.
Prepare(aPages: TsgcReportPrepared-Pages)	—	Surcharge qui prépare dans un objet pages exis-tant.
SaveToPDF(const aFileName) / SaveToPDFStream(aStream)	—	Prépare et exporte via PDFExport ; Title, Author, Subject, Keywords et Language du rapport sont

		copiés dans PDFExport avant chaque enregistrement.
SaveToXLSX / SaveToXLSXStream	—	Prépare et exporte vers XLSX.
SaveToCSV / SaveToCSVStream	—	Prépare et exporte vers CSV.
SaveToPNG(const aFileNamePattern: string; aDPI: Integer = 150)	—	Un fichier PNG par page ; aFileNamePattern contient %d.

Propriétés

Propriété	Type	Description
Report	TsgcReport	Le rapport, avec les modificateurs en attente écrits.
Engine	TsgcReportEngine	Le moteur du générateur, connectable à un TsgcReportPreviewVCL / TsgcReportPreviewFMX.
PDFExport	TsgcReportPDFExport	Créé à la première utilisation, possédé par le générateur ; définissez PDFA, PDFUA ou Compress avant l'enregistrement.

Exemple

Voir Prise en main pour un exemple minimal, et Guide pratique : un rapport groupé avec totaux pour un rapport complet construit entièrement avec l'API fluide.

Définitions de rapport JSON

Unité : sgcReports_Definition
Charge et enregistre un TsgcReport complet au format JSON, de sorte qu'un rapport créé une fois avec l'API de modèle ou l'API fluide peut être modifié, stocké et redéployé sans recompilation.

Enveloppe

```
{"format": "sgcReports", "version": 1, "report": { ... }}
```

Un membre chaîne facultatif `"$schema"` est accepté dans l'enveloppe.

Conventions

- Chaque propriété publiée du modèle est écrite avec son nom Delphi en lowerCamelCase (`DataSourceName` devient `dataSourceName`, `HAlign` devient `hAlign`) ; la propriété `Name` du composant du rapport ne fait pas partie de la définition.
- Les énumérations sont écrites sous forme de leurs identifiants Delphi (`"rbtMasterData"`), les ensembles sous forme de tableaux d'identifiants, les booléens sous forme de `true` / `false`, les entiers et les flottants sous forme de nombres JSON ; les couleurs sont des entiers `$00RRGGBB`.
- Les sous-objets `TPersistent` sont des objets imbriqués, les collections sont des tableaux d'objets d'élément, les `TStrings` sont des tableaux de chaînes.
- Les valeurs égales à leur valeur par défaut sont omises, exactement comme dans un DFM.
- Les octets de l'image intégrée d'un élément image sont écrits sous forme de la chaîne base64 `pictureData` de son objet `image`.

Le chargement efface d'abord le rapport. Toute erreur déclenche `ESgcReportDefinitionError` avec le chemin JSON du membre fautif, par exemple `report.pages[0].bands[2].height`. Un membre inconnu est une erreur sauf si `IgnoreUnknown` vaut `True`. Les flux et les fichiers sont en UTF-8 sans BOM à l'enregistrement ; un BOM est toléré au chargement.

Fonctions

Fonction	Description
sgcReportSaveToJSON(aReport: TsgcReport; aIndent: Boolean = True): string	Sérialise un rapport en chaîne JSON.
sgcReportLoadFromJSON(aReport: TsgcReport; const aJSON: string)	Charge un rapport à partir d'une chaîne JSON.
sgcReportSaveToJSONFile / sgcReportLoadFromJSONFile	Variantes fichier.
sgcReportSaveToJSONStream / sgcReportLoadFromJSONStream	Variantes flux.
sgcReportGenerateSchema: string	Schéma JSON (draft 2020-12) de la définition, généré à partir de la RTTI ; utilisé comme sgcReports.schema.json.

TsgcReportDefinition

La classe derrière les fonctions simples, pour les appelants qui ont besoin de contrôler `IgnoreUnknown` ou `Indent` : `SaveToJSON`, `LoadFromJSON`, `SaveToStream` / `LoadFromStream`, `SaveToFile` / `LoadFromFile`, et les propriétés `IgnoreUnknown` (par défaut False) et `Indent` (par défaut True, indentation de deux espaces et sauts de ligne LF).

Exemple

Voir Prise en main et Guide pratique : le concepteur de rapport web, qui modifie un definition.json et un data.json.

Guide pratique : facture avec des données maître-détail

Ce guide parcourt la démo `ReportInvoice` (`delphi\Demos\Delphi\ReportInvoice`) : deux `TClientDataSet` (factures et lignes de facture) lus via deux TsgcReportDataSetSource, liés par `MasterSource`, imprimés à raison d'une page par facture avec un en-tête de colonne répété, un sous-total, la TVA et un total, un code QR avec le numéro de facture, et la page X sur Y. Le rapport est affiché dans un TsgcReportPreviewVCL et exporté en fichier PDF/A-3a qui est également PDF/UA-1.

Lier les deux sources

```
srcInvoices.SourceName := 'Invoices';
srcInvoices.DataSet := cdsInvoices;

srcLines.SourceName := 'Lines';
srcLines.DataSet := cdsLines;
srcLines.MasterSource := srcInvoices;
srcLines.MasterFields := 'Number';
srcLines.DetailFields := 'Invoice';

sgcEngine.AddDataSource(srcInvoices);
sgcEngine.AddDataSource(srcLines);
```

Une page par facture

L'en-tête de groupe qui démarre chaque facture définit `StartNewPage`, et sa `GroupExpression` est le numéro de facture, de sorte qu'un nouveau groupe (et une nouvelle page) démarre à chaque changement :

```
oBand := oPage.Bands.Add;
oBand.BandType := rbtGroupHeader;
oBand.Name := 'InvoiceHeader';
oBand.Height := 110;
oBand.GroupExpression := 'Invoices.Number';
oBand.StartNewPage := True;
```

L'élément titre à l'intérieur est balisé `rtrHeading1` pour PDF/UA, et la valeur du code QR est un modèle de texte contenant le numéro de facture, et non une expression seule, de sorte que le texte littéral environnant serait conservé :

```
oItem := oBand.Items.Add;
oItem.Kind := rikText;
oItem.Text.Text := 'INVOICE';
oItem.TagRole := rtrHeading1;
...
oItem := oBand.Items.Add;
oItem.Kind := rikBarcode;
oItem.Barcode.Symbology := rbsQRCode;
oItem.Barcode.Value := '[Invoices.Number]';
oItem.AltText := 'QR code with the invoice number';
```

Le tableau des lignes et son en-tête répété

L'en-tête de colonne du tableau des lignes est une seconde bande `rbtGroupHeader` sur la même `GroupExpression`, avec `RepeatOnEveryPage := True` afin qu'elle se réimprime chaque fois que les lignes d'une longue facture se poursuivent sur une seconde page. La bande maître parcourt les factures et n'imprime rien ; la bande détail parcourt les lignes :

```
oBand := oPage.Bands.Add; // LinesHeader
oBand.BandType := rbtGroupHeader;
oBand.GroupExpression := 'Invoices.Number';
oBand.RepeatOnEveryPage := True;

oBand := oPage.Bands.Add; // InvoiceData (master), prints nothing
oBand.BandType := rbtMasterData;
oBand.DataSourceName := 'Invoices';

oBand := oPage.Bands.Add; // LineData (detail)
oBand.BandType := rbtDetailData;
oBand.DataSourceName := 'Lines';
```

Totaux et TVA

Le pied de groupe voit les champs de la dernière ligne imprimée et de la facture elle-même, de sorte que son taux de TVA provient directement de `Invoices.VATRate` même si le dernier enregistrement parcouru était une ligne :

```
// Subtotal
'[Sum(Lines.Qty * Lines.Price)]'
// VAT, using the invoice's own field from the group footer
'VAT [Invoices.VATRate] %'
'[Sum(Lines.Qty * Lines.Price) * Invoices.VATRate / 100]'
// Total
'[Sum(Lines.Qty * Lines.Price) * (1 + Invoices.VATRate / 100)]'
```

PDF accessible et archivable

```
// PDFA and PDFUA are set on pdfExport in the form
oPages := sgcEngine.Prepare;
try
  pdfExport.SaveToFile(oPages, 'invoice_pdfa3a_ua1.pdf');
finally
  oPages.Free;
end;
```

Le projet complet et compilable est `delphi\Demos\Delphi\ReportInvoice\sgcReports_Demo_Invoice.dpr`.

Guide pratique : un rapport groupé avec totaux

Ce guide parcourt la démo `ReportFluent` (`delphi\Demos\Delphi\ReportFluent`) : une liste de ventes lue depuis `sales.json`, groupée par région, avec un sous-total par région et un total général, construite entièrement avec TsgcReportBuilder. Un équivalent en source JSON et API de modèle, groupé par région et client, est la démo `ReportGroups`.

Source de données et styles

```
FSales := TsgcReportJSONSource.Create(Self);
FSales.SourceName := 'Sales';
FSales.FileName := ExtractFilePath(ParamStr(0)) + 'sales.json';

FBuilder := TsgcReportBuilder.Create;
FBuilder.Title('Sales by region').Author('eSeGeCe').Units(ruMillimeters)
  .Style('Base').Font('Arial', 10).EndStyle
  .Style('Title').Font('Arial', 18).Bold.EndStyle
  .Style('Group').Font('Arial', 11).Bold.BackColor($EEEEEE).EndStyle
```

Page, titre et légendes de colonnes

```
  .Page.Paper(rpsA4).Margins(15)
  .ReportTitle(14).Style('Title').Text('Sales by region', 0, 0, 180, 10)
    .TagRole(rtrHeading1).EndBand
  .PageHeader(8).Style('Base').Bold.Text('Customer', 2, 0, 68, 6)
    .Text('Product', 70, 0, 60, 6).Text('Amount', 130, 0, 50, 6)
    .AlignRight.EndBand
```

Grouper par région, une ligne par vente, sous-total

```
const
  CS_TOTAL = '[FormatFloat(''#,0.00'', Sum(Sales.Amount))]';
begin
  ...
  .GroupHeader('Sales.Region', 9).Style('Group')
    .Text('Region: [Sales.Region]', 0, 1, 180, 7).TagRole(rtrHeading2).EndBand
  .MasterData('Sales', 6).Style('Base').Text('[Sales.Customer]', 2, 0, 68, 6)
    .Text('[Sales.Product]', 70, 0, 60, 6).Text('[Sales.Amount]', 130, 0, 50, 6)
    .AlignRight.Format('#,0.00').EndBand
  .GroupFooter(9).Style('Base')
    .Text('Region total ' + CS_TOTAL, 80, 1, 100, 6).AlignRight.Bold.EndBand
```

`GroupHeader` accepte soit une expression nue (`'Sales.Region'`), soit une expression entourée de `[ ]`. L'agrégat `Sum` dans un pied de groupe se réinitialise à chaque région ; la même expression dans la bande de résumé ci-dessous s'exécute pour tout le rapport.

Total général et numéros de page

```
  .Summary(12).Style('Base').Line(0, 2, 180, 2)
    .Text('Grand total ' + CS_TOTAL, 80, 4, 100, 7).AlignRight.Bold.EndBand
  .PageFooter(8).Style('Base').Text('Page [Page] of [TotalPages]', 0, 1, 180, 6)
    .AlignCenter.EndBand
  .EndPage.Data(FSales);
```

Aperçu et export

```
FPreview.Engine := FBuilder.Engine;
FBuilder.Flush;
FPreview.Prepare;
...
FBuilder.SaveToPDF(OutputFile('sales.pdf'));
FBuilder.SaveToXLSX(OutputFile('sales.xlsx'));
FBuilder.SaveToCSV(OutputFile('sales.csv'));
FBuilder.SaveToPNG(OutputFile('sales_%d.png'), 96);
```

Le projet complet et compilable est `delphi\Demos\Delphi\ReportFluent\sgcReports_Demo_Fluent.dpr` ; le même modèle de regroupement et de totaux avec une source `TDataSet` ou API de modèle est également illustré par `ReportGroups`.

Guide pratique : exporter en XLSX, CSV et PNG

Ce guide parcourt la démo `ReportExports` (`delphi\Demos\Delphi\ReportExports`) : un rapport est préparé une fois, puis exporté via quatre composants différents. Les pages préparées (`TsgcReportPreparedPages`) sont réutilisées pour chaque export, de sorte que la mise en page et le parcours des données ne s'exécutent qu'une seule fois.

```
sgcEngine.Report := sgcReport;
sgcEngine.AddDataSource(srcOrders);
FPages := sgcEngine.Prepare;
sgcPreview.LoadPrepared(FPages);
```

PDF

```
pdfExport.SaveToFile(FPages, OutputFolder + 'orders.pdf');
```

XLSX

TsgcReportXLSXExport transforme les pages préparées en une grille (une ligne de feuille de calcul par instance de bande) avant d'écrire le classeur :

```
xlsxExport.SheetName := 'Orders';
xlsxExport.SaveToFile(FPages, OutputFolder + 'orders.xlsx');
```

CSV

```
csvExport.Delimiter := ',';
csvExport.SaveToFile(FPages, OutputFolder + 'orders.csv');
```

PNG, un fichier par page

```
imageExport.DPI := 150;
imageExport.SaveToFiles(FPages, OutputFolder + 'orders_%d.png');
```

Notes

- Par défaut (`ExcludePageBands = True`) les en-têtes et pieds de page et de colonne ne deviennent pas des lignes de grille dans l'export XLSX et CSV, seules les bandes de titre de rapport, de groupe et de données le deviennent.
- `DetectNumbers` (par défaut `True`) écrit une cellule texte qui s'analyse comme un nombre en tant que cellule XLSX numérique plutôt qu'en texte.
- `SaveToFiles` nomme les fichiers en substituant le numéro de page basé sur 1 à `%d` dans le motif, de sorte que `orders_%d.png` produit `orders_1.png`, `orders_2.png`, et ainsi de suite.

Le projet complet et compilable est `delphi\Demos\Delphi\ReportExports\sgcReports_Demo_Exports.dpr`.

Guide pratique : un PDF accessible validé avec veraPDF

Ce guide parcourt la démo `ReportPDFUA` (`delphi\Demos\Delphi\ReportPDFUA`) : un titre H1, des titres de groupe H2, un tableau avec des lignes d'en-tête, une image avec texte alternatif et un code-barres, exportés en PDF/UA-1, puis en PDF/A-3b, puis à la fois en PDF/A-3a et PDF/UA-1 ensemble. Voir Accessibilité et PDF/A pour les règles de balisage et les niveaux de conformité.

Baliser le contenu

```
// title
oItem.TagRole := rtrHeading1;
// group heading
oItem.TagRole := rtrHeading2;
// logo, needs alternate text to satisfy PDF/UA
oItem.Kind := rikImage;
oItem.Image.FileName := 'logo.png';
oItem.AltText := 'eSeGeCe logo';
// decorative rule: excluded from the reading order
oItem.Kind := rikLine;
oItem.TagRole := rtrArtifact;
```

Un tableau construit à partir d'une bande d'en-tête de groupe ou de colonne suivie de lignes de détail est balisé automatiquement ; rien d'autre n'est requis pour que ses cellules d'en-tête soient marquées `rtrTableHeader`.

Exporter les trois variantes

```
// PDF/UA-1 only
pdfExport.PDFA := rpaNone;
pdfExport.PDFUA := True;
pdfExport.Language := 'en-US';
pdfExport.SaveToFile(oPages, 'report_ua1.pdf');

// PDF/A-3b only (visual reproducibility, no tagging required)
pdfExport.PDFA := rpaA3b;
pdfExport.PDFUA := False;
pdfExport.SaveToFile(oPages, 'report_a3b.pdf');

// both: PDF/A-3a and PDF/UA-1
pdfExport.PDFA := rpaA3a;
pdfExport.PDFUA := True;
pdfExport.SaveToFile(oPages, 'report_a3a_ua1.pdf');
```

Valider avec veraPDF

veraPDF est un validateur PDF/A et PDF/UA open source adossé à l'ISO, exécuté séparément de sgcReports en tant qu'outil en ligne de commande. Vérifiez chaque fichier par rapport à la variante correspondante :

```
verapdf --flavour ua1 report_ua1.pdf
verapdf --flavour 3b  report_a3b.pdf
verapdf --flavour 3a  report_a3a_ua1.pdf
verapdf --flavour ua1 report_a3a_ua1.pdf
```

Une exécution réussie rapporte `compliant: true` pour la variante demandée ; un échec liste la clause ISO et l'objet qui la viole, ce qui est le moyen le plus rapide de trouver un `AltText` manquant ou un élément qui aurait dû être balisé `rtrArtifact`.
Le projet complet et compilable est `delphi\Demos\Delphi\ReportPDFUA\sgcReports_Demo_PDFUA.dpr`.

Guide pratique : servir des rapports via HTTP

Ce guide parcourt la démo `ReportHTTPServer` (`delphi\Demos\Delphi\ReportHTTPServer`) : un `TsgcWebSocketHTTPServer` sur le port 8080 avec un TsgcReportHTTPEndpoint servant un rapport "sales", et une page sgcHTML à `/` hébergeant un TsgcReportHTMLViewer. Nécessite le package `sgcReportsHTML` : sgcWebSockets (édition Professional ou supérieure), sgcHTML et sgcReports.

Enregistrer le rapport et accrocher le serveur

```
reportEndpoint.BasePath := '/reports';
reportEndpoint.Server := httpServer;
reportEndpoint.RegisterReport('sales', salesEngine);

httpServer.Port := 8080;
httpServer.Active := True;
```

Routes servies

- `/reports/sales.pdf`, `/reports/sales.xlsx`, `/reports/sales.csv`
- `/reports/sales/page/1.png?dpi=96`
- `/reports/sales/info` (JSON : nom, titre, nombre de pages, paramètres)

Restreindre les paramètres

Le point de terminaison n'accepte que les paramètres de requête dont le nom existe dans les `Variables` du moteur de rapport, de sorte qu'un rapport avec une seule variable nommée `Region` n'accepte que `?Region=...` `OnBeforePrepare` peut valider davantage la valeur, par exemple en rejetant tout ce qui est en dehors d'une liste autorisée avant que le rapport ne soit préparé :

```
salesEngine.Variables.Values['Region'] := 'All';
// OnBeforePrepare checks the incoming Region against an allowed list
// (All, North, South, East, West) and can reject the request (403)
// or load the rows for that region before the report is prepared.
```

Servir la page du visualiseur

```
reportViewer.Endpoint := reportEndpoint;
reportViewer.ReportName := 'sales';
reportViewer.ShowParameters := True;
```

Notes de compilation

Compilez avec le dossier `Source` de sgcWebSockets sur le chemin de recherche. Définissez `SGC_CUSTOM_INDY` lors de l'édition de liens avec les packages sgcWebSockets publiés (construits sur le fork sgcIndy), ou `SGC_REPORTS_STOCK_INDY` lorsque sgcWebSockets a été construit avec l'Indy fourni avec l'IDE.
Le projet complet et compilable est `delphi\Demos\Delphi\ReportHTTPServer\sgcReports_Demo_HTTPServer.dpr`.

Guide pratique : un démon de rapports Linux64

Ce guide parcourt la démo `ReportLinuxDaemon` (`delphi\Demos\Delphi\ReportLinuxDaemon`), une application console qui s'exécute sans modification sur Windows et Linux64, puisque le noyau sgcReports n'a aucune dépendance VCL ou FMX. Voir Sécurité des threads et utilisation serveur pour le modèle de pool qu'il utilise.

Fichiers de tâches

Le démon surveille un dossier de tâches à la recherche de fichiers de tâche `*.json`. Une tâche nomme une définition de rapport, un fichier de données et un format de sortie (`pdf`, `xlsx`, `csv` ou `png`) ; les chemins relatifs de la tâche se résolvent par rapport au dossier propre du fichier de tâche. Une tâche est prise en charge en la renommant en `.json.worki` la sortie est écrite à côté, et la tâche se termine en `.json.done` ou `.json.failed`.

Pool de workers

Un pool de 4 threads de travail rend les tâches simultanément. Chaque thread charge la définition de rapport de la tâche avec `sgcReportLoadFromJSON` dans son propre `TsgcReport`, crée son propre `TsgcReportEngine` et sa propre source de données, prépare, exporte, puis libère tout ce qu'il a créé, en suivant le modèle partagé et en lecture seule décrit dans Sécurité des threads et utilisation serveur.

Exécution

```
sgcReports_Demo_LinuxDaemon JOBS_FOLDER
sgcReports_Demo_LinuxDaemon --sample FOLDER   -- writes sample.job plus its report definition and data
```

SIGTERM ou SIGINT sur Linux, et Ctrl+C sur Windows, arrêtent le démon proprement : aucune nouvelle tâche n'est prise en charge, les tâches en cours se terminent, et le processus se termine avec le code 0.

Unité systemd

`sgcreports-daemon.service` exécute l'exécutable compilé sous un utilisateur dédié, surveille `/var/lib/sgcreports/jobs`, redémarre en cas d'échec et s'arrête sur SIGTERM avec un délai de 60 secondes pour que les tâches en cours se terminent. Les journaux vont vers stdout, à lire avec :

```
journalctl -u sgcreports-daemon -f
```

Le projet complet et compilable est `delphi\Demos\Delphi\ReportLinuxDaemon\sgcReports_Demo_LinuxDaemon.dpr`.

Guide pratique : un rapport FMX mobile

Ce guide parcourt la démo `ReportMobileFMX` (`delphi\Demos\Delphi\ReportMobileFMX`), un petit bon de livraison A5 (un code QR avec le numéro du bon, des lignes d'articles, le total des unités et une ligne de signature) construit avec l'API fluide et affiché dans un TsgcReportPreviewFMX. Cible Win32, Win64, Android64 et iOS depuis un seul projet.

Emplacement d'enregistrement

Le bouton Save PDF écrit `sgcReports_DeliveryNote.pdf` dans le dossier de documents de l'application et affiche le chemin :

```
uses System.IOUtils;
...
vBuilder.SaveToPDF(TPath.Combine(TPath.GetDocumentsPath,
  'sgcReports_DeliveryNote.pdf'));
```

Polices sur mobile

La démo utilise les polices déjà présentes sur le système via la découverte de polices (voir Polices). iOS n'offre aucun dossier de polices système lisible depuis une application en bac à sable, donc lorsqu'aucune police TrueType adaptée n'est trouvée, la démo charge à la place `DejaVuSans.ttf` depuis son propre dossier ; `DejaVu-LICENSE.txt` est la licence de cette police et est fourni avec elle.

Déployer la police

Dans l'IDE, Project > Deployment, ajoutez `DejaVuSans.ttf` pour chaque plateforme mobile avec ces chemins distants :

Plateforme	Chemin distant
Android64	assets\internal\
iOS	StartUp\Documents\

Sur Windows, la police est lue depuis le dossier de l'exécutable, copiez-la donc à cet endroit lorsque le système n'a aucune police adaptée installée.
Le projet complet et compilable est `delphi\Demos\Delphi\ReportMobileFMX\sgcReports_Demo_MobileFMX.dpr` ; il nécessite sgcReports, sgcPDF et FireMonkey.

Guide pratique : le concepteur de rapport web

TsgcReportHTMLDesigner est un concepteur de bandes basé sur le navigateur qui modifie une définition de rapport JSON en la validant par rapport à `sgcReports.schema.json`, avec un aperçu en direct servi par le même TsgcReportHTTPEndpoint qu'utilise un visualiseur ou une application pour servir le rapport terminé.

L'activer

```
reportEndpoint.DesignerEnabled := True;
reportDesigner.Endpoint := reportEndpoint;
reportDesigner.ReportName := 'sales';
```

`DesignerEnabled` active les routes de schéma, de définition et d'aperçu que la page du concepteur appelle ; laissez-la `False` sur un point de terminaison de production qui ne sert que des rapports terminés, et ne l'activez que là où l'édition est prévue, protégée par votre propre authentification.

Ce que vous apporte le concepteur navigateur

La page servie par `TsgcReportHTMLDesigner` est un concepteur de bandes complet en glisser-déposer : une arborescence des pages, bandes et éléments ; une zone de dessin de bande en millimètres avec déplacement, redimensionnement, alignement et ajustement au clavier ; un inspecteur de propriétés généré à partir de `schema.json` ; Save (avec le champ fautif mis en évidence à partir du chemin JSON d'une erreur de validation côté serveur) et Preview ; annuler et rétablir ; et un bouton Download JSON. Voir TsgcReportHTMLDesigner pour la liste complète des propriétés, et TsgcReportHTTPEndpoint pour les routes du concepteur qu'il appelle (`schema.json`, `{name}/definition` et `{name}/preview.png`).

Modifier une définition sans le concepteur navigateur

La démo `ReportJSONDefinition` (`delphi\Demos\Delphi\ReportJSONDefinition`) montre le même aller-retour depuis du code Delphi, utile pour un éditeur de bureau ou pour des tests automatisés d'une définition : elle modifie `definition.json` et son `data.json` dans deux zones de texte, un bouton Render charge la définition avec `sgcReportLoadFromJSON` et affiche l'aperçu, les erreurs sont rapportées dans la barre d'état avec le chemin JSON du membre fautif (voir Définitions de rapport JSON), et un bouton Save definition réécrit le rapport courant en JSON avec `sgcReportSaveToJSON`.
Le projet complet et compilable pour l'éditeur basé sur le code est `delphi\Demos\Delphi\ReportJSONDefinition\sgcReports_Demo_JSONDefinition.dpr`.

Introduction

sgcCharts est une bibliothèque de composants Delphi pour les graphiques de données en direct. Une source de données ajoute des points ou des chandeliers OHLC à un tampon circulaire, et un canevas dessine un instantané décimé de ce tampon, si bien qu'un flux qui publie des milliers de mises à jour par seconde continue de s'afficher à une cadence d'images stable.
La bibliothèque cible VCL, FireMonkey et le rendu sans interface graphique (sortie PNG ou SVG sans fenêtre), si bien que le même code de tampon et de source de données dessine sur un formulaire, sur un écran mobile ou à l'intérieur d'une application console ou d'un service.
Cette aide couvre les parties de sgcCharts qui existent aujourd'hui : les sources de données qui alimentent un tampon (TsgcChartDataSource et ses classes descendantes), le pont sgcWebSockets qui alimente un tampon depuis une connexion active (Présentation du pont), ainsi que les tampons et les fonctions de décimation eux-mêmes (TsgcChartBuffer).

- Présentation : le modèle, les canevas, les tampons et les sources de données, et comment ils s'articulent.
- Éditions : ce que Registered, Trial et Community incluent chacune.
- Démarrage rapide : état actuel du parcours de prise en main.
- Sources de données : manuelle, dataset, JSON et CSV.
- Pont : MQTT, WebSocket, SSE et Binance, via sgcWebSockets.
- Référence : les classes de tampon et les fonctions de décimation.

Architecture

sgcCharts sépare les données qu'un graphique affiche de la surface sur laquelle il est dessiné, si bien que les mêmes données peuvent alimenter un formulaire VCL, une vue FireMonkey ou un rendu PNG/SVG sans interface graphique.

Un modèle, plusieurs canevas

Les données d'un graphique vivent dans un tampon, pas dans un contrôle. **TsgcChartBuffer** conserve des points X/Y et **TsgcChartOHLCBuffer** conserve des chandeliers OHLC (ouverture, plus haut, plus bas, clôture, volume). Les deux sont de simples classes sans dépendance à VCL ou FireMonkey, si bien qu'elles peuvent être créées, alimentées et lues depuis une application console, un service ou un thread d'arrière-plan. Le dessin est une étape séparée : un canevas lit un instantané décimé d'un tampon et le peint, en utilisant un canevas raster (`sgcUICore_Canvas_Raster`) ou un canevas SVG (`sgcUICore_Canvas_SVG`) derrière la surface partagée `sgcUICore_Canvas`.
La couche de canevas, de raster, de police TrueType, de SVG et de thèmes qui dessine ces graphiques ne fait pas partie de sgcCharts lui-même. C'est le package partagé **sgcUICore** (unités Delphi `sgcUICore_*` dans le package d'exécution sgcUICore, packages .NET `esegece.sgcUICore` et `esegece.sgcUICore.Rendering`). Il est installé automatiquement avec sgcCharts et partagé avec les autres produits visuels d'eSeGeCe.

Tampons circulaires avec décimation min/max

Les deux classes de tampon sont des tampons circulaires à capacité fixe. En mode `cbmOverwrite`, un tampon plein supprime son point le plus ancien au prochain ajout ; en mode `cbmGrow`, sa capacité double à la place. Chaque méthode publique d'un tampon prend un verrou interne, si bien qu'un thread d'alimentation peut ajouter des données pendant qu'un thread de rendu lit un instantané sans que les deux n'entrent en collision. **Snapshot** et **SnapshotRange** copient les données dont l'appelant a besoin dans un enregistrement possédé par l'appelant sous ce verrou et le libèrent aussitôt ; l'appelant décime et dessine ensuite à partir de la copie, hors du verrou.
`sgcChartDecimate` réduit un instantané à environ un ensemble de points par colonne de pixels, en utilisant `dmMinMax` (le premier, le minimum, le maximum et le dernier point de chaque colonne, ce qui garde une polyligne en direct visuellement exacte) ou `dmLTTB` (Largest Triangle Three Buckets, pour une série statique). `sgcChartDecimateOHLC` fusionne de la même manière les chandeliers qui tombent dans la même colonne de pixels. Voir TsgcChartBuffer et Décimation.

Graphique, séries et axes

TsgcChart est le modèle de graphique : un composant non visuel qui détient les axes, les séries, le titre, la légende, les marges, le thème et la palette, mais ne dessine rien par lui-même. Une série est elle-même un composant, pas une partie imbriquée du graphique ; assigner sa propriété **Chart** est ce qui l'enregistre, et une série lit son propre tampon directement (via l'un des types de série concrets) plutôt qu'en passant par le graphique. Une vue appelle **TsgcChart.Render** pour peindre tout le modèle sur un TsgcUICanvas, si bien que le même graphique, avec les mêmes séries et axes, peut être affiché par plus d'une vue, ou être rendu sans interface graphique en PNG ou SVG sans aucune vue du tout. Voir TsgcChart et TsgcChartSeries.

Sources de données

Une classe descendante de **TsgcChartDataSource** lit des enregistrements depuis quelque part (une application appelant **AppendValue** directement, un `TDataSet`, un document JSON ou du texte CSV) et ajoute les points ou chandeliers correspondants aux tampons assignés à ses propriétés **Buffer** et **OHLCBuffer**. L'endroit où vivent les champs d'un enregistrement est décrit une seule fois, dans **Mapping**, et chaque source de données le lit de la même manière. Voir Sources de données.

Pont sgcWebSockets

Le paquet optionnel `sgcChartsWS` ajoute des sources de données qui lisent depuis une connexion sgcWebSockets active : un sujet MQTT, un flux de messages WebSocket, un flux Server-Sent Events, ou l'API de données de marché publique Binance. Chacune se rattache à l'événement existant du composant client qu'elle cible, si bien que le gestionnaire propre de l'application continue de se déclencher aux côtés du graphique. Ce paquet nécessite sgcWebSockets ; le paquet de base `sgcCharts` n'en a pas besoin. Voir Pont.

Éditions

Voir Éditions pour ce que Registered, Trial et Community incluent chacune.

Éditions

sgcCharts est compilé depuis la même source pour chaque édition ; l'édition ne change que les unités qu'une compilation inclut.

Édition	Inclut
Registered	Toutes les unités : les sources de données de base, les indicateurs et panneaux financiers, et les sources de données en direct (streaming) y compris le pont sgcWebSockets. Aucune notice d'essai.
Trial	Les mêmes unités que Registered. Une compilation avec `SGC_TRIAL` affiche une notice unique une fois la fenêtre de version nommée dans la compilation dépassée ; rien d'autre ne change.
Community	Le noyau de graphiques complet sans limite fonctionnelle : les huit séries (ligne, ligne rapide, ligne en escalier, aire, barre/colonne, nuage de points, chandelier et OHLC), les vues VCL, FMX et sans interface graphique, les tampons, les sources de données de base (manuelle, dataset, JSON, CSV) et l'interaction. Une compilation avec `SGC_COMMUNITY` exclut le pack financier (panneaux, séries indicateur et volume, agrégateur, outils de dessin, marqueur de dernière valeur et bande bid/ask), et l'édition Community ne fournit pas les unités et paquets du pont sgcWebSockets.

L'édition est un choix effectué à la compilation dans `sgcVer.inc` (`SGC_COMMUNITY`, qui désactive aussi `SGC_TRIAL`), pas un commutateur à l'exécution.

Installation

sgcCharts fait partie de la distribution sgcUI. Un seul installateur couvre les produits sgcUI (sgcUICore, sgcCharts, sgcPDF, sgcReports, sgcMaps, sgcDashboard, sgcGrid, sgcNavigation, sgcToolbar et sgcWizard) et n'installe que les produits inclus dans votre licence.

Installation avec l'installateur sgcUI

1. Lancez l'installateur sgcUI et connectez-vous avec le nom d'utilisateur et le mot de passe de votre compte eSeGeCe.
2. Choisissez les versions de Delphi et de C++Builder dans lesquelles installer.
3. Sur la page des composants, sélectionnez **sgcCharts** et tout autre produit souhaité. Le serveur de licences renvoie les produits que vous possédez, et l'installateur affiche une case à cocher par produit. Les produits que votre licence n'inclut pas sont grisés. Une licence All-Access déverrouille tous les produits.
4. L'installateur copie les fichiers, puis compile les packages sur votre machine pour chaque IDE sélectionné, dans l'ordre des dépendances, et installe les packages de conception.

Ce qui est fourni avec sgcCharts

- **sgcUICore**, la base commune de sgcCharts et des produits de mise en page. Il n'a pas de case à cocher propre. L'installateur l'installe automatiquement dès que sgcCharts est sélectionné.
- **sgcCharts pour sgcWebSockets**, listé sous **sgcWebSockets bridges**. Il installe les packages sgcChartsWS avec les sources de données en direct de la passerelle sgcWebSockets. L'installateur ne le propose que si sgcWebSockets est déjà installé sur la machine et que votre licence inclut sgcCharts. Sinon, il est grisé.

Dossiers installés

Tous les produits s'installent dans un seul dossier, par défaut `Documents\esegece\sgcUI` (`sgcUI_Trial` pour la version d'essai et `sgcUI_Community` pour l'édition Community).

Dossier	Contenu
`Source`	Le code source de chaque produit installé. C'est l'unique chemin de bibliothèque de sgcUI.
`Packages`	Les packages de chaque produit installé, un jeu par produit et par version d'IDE.
`Lib<V>`	Les unités compilées pour chaque version d'IDE et chaque plateforme, par exemple `LibD13` .
`Demos\Charts`	Les démos de sgcCharts. Voir Démos.

L'installateur ajoute `sgcUI\Source` au chemin de bibliothèque et au chemin de navigation de chaque IDE et plateforme sélectionnés, ainsi que le dossier des unités compilées. sgcUICore se trouve dans le même dossier `Source`, il n'a donc besoin d'aucun chemin propre.

Trial et Community

- **Trial** : l'installateur Trial de sgcUI installe tous les produits en mode d'essai, y compris sgcCharts.
- **Community** : l'installateur Community gratuit de sgcUI inclut sgcCharts, avec sgcDashboard, sgcGrid, sgcNavigation, sgcToolbar et sgcWizard. Les passerelles sgcWebSockets ne font pas partie de l'installateur Community, et sgcPDF et sgcMaps ne font pas partie de Community. Voir Éditions pour ce que la version Community laisse de côté.

Migration depuis une installation autonome de sgcCharts

Si sgcCharts a déjà été installé avec son propre installateur autonome, l'installateur sgcUI détecte cette installation, ainsi que le dossier sgcUICore créé par l'installateur autonome. Il liste leurs chemins de bibliothèque et leurs packages de conception sur la page **Previous installations**. Quand vous continuez, il les supprime de l'IDE avant de compiler les nouveaux packages, de sorte qu'il n'y ait jamais deux copies d'une unité dans le chemin. La page n'est affichée que lorsqu'une ancienne installation est trouvée.

Compiler les packages manuellement

Pour compiler vous-même les packages, ouvrez-les pour votre version d'IDE depuis `sgcUI\Packages`. `<V>` est le suffixe de version Delphi, par exemple `sgcChartsD13.dpk`.

Package	Contenu
`sgcUICore<V>`	La base commune sgcUICore (exécution).
`sgcCharts<V>` / `dclsgcCharts<V>`	Les composants sgcCharts (exécution / conception).
`sgcChartsWS<V>` / `dclsgcChartsWS<V>`	La passerelle sgcWebSockets (exécution / conception). Nécessite sgcWebSockets.

Compilez-les dans l'ordre des dépendances : `sgcUICore`, puis `sgcCharts`, puis `dclsgcCharts`, et enfin les packages de la passerelle. Installez ensuite les packages de conception `dcl*` et ajoutez `sgcUI\Source` au chemin de bibliothèque de votre version de Delphi (Tools > Options > Language > Delphi Options > Library > Library path). Pour C++Builder, ouvrez le projet `.cbproj` correspondant à votre version, par exemple `sgcChartsC13.cbproj`, et compilez-le. Les en-têtes `.hpp` générés apparaissent à côté du package compilé.

.NET

Rien ne change pour .NET. sgcCharts reste distribué sous forme de ses propres packages NuGet (`esegece.sgcCharts` et ses packages compagnons). Voir la présentation .NET.

Plateformes

Cible	Prise en charge
Delphi	Delphi 7, 2007, 2009, 2010, XE à XE8, 10 à 10.4, 11, 12 et 13.
C++Builder	C++Builder 2007, 2010, XE à XE8, 10 à 10.4, 11, 12 et 13. Les packages génèrent les en-têtes .hpp et les fichiers .lib qu'un projet C++Builder inclut et lie.
VCL	TsgcVCLChart sous Windows, Win32 et Win64, toutes les versions de Delphi.
FireMonkey	TsgcFMXChart à partir de Delphi XE6 (SGC_CHARTS_FMX), sous Win32, Win64, macOS, iOS et Android.
Linux64	Les unités indépendantes du framework, TsgcChart et TsgcChartRenderer avec la sortie PNG et SVG headless, à partir de Delphi 10.2 (SGC_CHARTS_LINUX).
.NET	Un port C# pour WinForms, WPF et Avalonia, avec la sortie PNG et SVG headless, voir Présentation de .NET.

Packages

sgcCharts fournit un package d'exécution indépendant du framework et un package d'exécution par framework, de sorte qu'une application FireMonkey pour Android, iOS ou macOS ne lie jamais la VCL. Chaque nom de package se termine par le suffixe de l'IDE, par exemple sgcChartsD13.dpk pour Delphi 13 et sgcChartsC13.cbproj pour C++Builder 13.

Package	Type	Contenu	Requiert
sgcCharts	Exécution, indépendant du framework	Chaque unité du produit qui n'utilise ni la VCL ni FireMonkey.	rtl, dbrtl, sgcUICore
sgcChartsVCL	Exécution, VCL	sgcCharts_VCL (TsgcVCLChart)	rtl, vcl, sgcUICore, sgcUICoreVCL, sgcCharts
sgcChartsFMX	Exécution, FireMonkey, à partir de XE6	sgcCharts_FMX (TsgcFMXChart)	rtl, fmx, sgcUICore, sgcUICoreFMX, sgcCharts
dclsgcCharts	Conception, VCL	Enregistre TsgcChart, TsgcChartRenderer et TsgcVCLChart dans la page de palette **SGC Charts**.	designide, rtl, vcl, sgcCharts, sgcChartsVCL
dclsgcChartsFMX	Conception, FireMonkey, à partir de XE6	Enregistre TsgcFMXChart dans la page de palette **SGC Charts**.	designide, rtl, fmx, sgcCharts, sgcChartsFMX, dclsgcCharts
sgcChartsWS, dclsgcChartsWS	Exécution et conception, passerelle sgcWebSockets	La passerelle de données en direct, indépendante du framework.	rtl, sgcCharts, sgcWebSockets

sgcUICore suit la même répartition : sgcUICore, sgcUICoreVCL et sgcUICoreFMX, sans package de conception. dclsgcChartsFMX requiert dclsgcCharts, installez donc d'abord dclsgcCharts.

Versions de l'IDE et plateformes cibles

Delphi	sgcCharts	sgc-ChartsVCL	sgcChartsFMX	dclsgcCharts, dclsgcChartsFMX
7, 2007, 2009, 2010, XE	Win32	Win32	Aucune	Win32
XE2 à XE5	Win32, Win64	Win32, Win64	Aucune	Win32
XE6 à 10.2	Win32, Win64	Win32, Win64	Win32, Win64	Win32
10.3	Win32, Win64, Linux64, OSX64	Win32, Win64	Win32, Win64, OSX64	Win32
10.4	Win32, Win64, Linux64, OSX64, Android, An-droid64, iOSDevice64	Win32, Win64	Win32, Win64, OSX64, Android, Android64, iOSDevice64	Win32
11	Win32, Win64, Linux64, OSX64, OSXARM64, Android, Android64, iOSDevice64, iOSSi-mARM64	Win32, Win64	Win32, Win64, OSX64, OSXARM64, Android, Android64, iOSDe-vice64, iOSSimARM64	Win32
12, 13	Comme 11	Win32, Win64	Comme 11	Win32, Win64

De XE6 à 10.2, le package FireMonkey est compilé pour Windows uniquement. Les packages de conception sont chargés par l'IDE, ils sont donc compilés pour la plateforme de l'IDE uniquement.

C++Builder	sgcCharts, sgcChartsV-CL	sgcChartsFMX	dclsgcCharts, dclsgc-ChartsFMX
2007, 2010, XE, XE2	Win32	Aucune	Win32 (dclsgcCharts uniquement)
XE3 à XE5	Win32, Win64	Aucune	Win32 (dclsgcCharts uniquement)
XE6 à 12	Win32, Win64	Win32, Win64	Win32
13	Win32, Win64, Win64x	Win32, Win64, Win64x	Win32, Win64x

Les projets C++Builder portent les mêmes noms avec le suffixe C, par exemple sgcChartsC13.cbproj, sgcChartsVCLC13.cbproj, sgcChartsFMXC13.cbproj, dclsgcChartsC13.cbproj et dclsgcChartsFMXC13.cbproj. Les packages C++Builder sont compilés pour Windows uniquement. Les notes C++Builder qui s'appliquent à chaque produit sgcUI (packages d'exécution sous Win64, C++Builder 10.3 sous Win64, noms de fichiers des packages Win64x) se trouvent dans le sujet Plateformes et packages de l'aide de la suite sgcUI.

FireMonkey sur Android, iOS et macOS

Les contrôles FireMonkey (TsgcFMXChart) ciblent macOS à partir de Delphi 10.3, Android et iOS à partir de Delphi 10.4, et macOS ARM et le simulateur iOS à partir de Delphi 11. Dans un projet FireMonkey pour ces plateformes :

- Utilisez les contrôles FireMonkey. Les contrôles VCL et sgcChartsVCL ne fonctionnent que sous Windows.
- Déposez le contrôle sur un formulaire, ou ajoutez son unité à la clause uses : sgcCharts_FMX.

- Sélectionnez la plateforme cible dans la fenêtre Projets. L'installateur ajoute le dossier Source de sgcUI et les unités compilées de chaque plateforme (LibD<version>\<platform>) au chemin de bibliothèque de chaque plateforme qu'il compile. Sans l'installateur, ajoutez le dossier Source de sgcUI au chemin de bibliothèque de la plateforme cible (Outils, Options, Langage, Delphi, Bibliothèque).
- Les applications Android et iOS sont liées statiquement, aucun package d'exécution n'est donc déployé.

Linux64

Le package indépendant du framework sgcCharts est compilé pour Linux64 à partir de Delphi 10.3. Une application Linux64, par exemple un programme console ou un serveur, utilise TsgcChart et TsgcChartRenderer, qui rend un graphique en PNG ou SVG sans fenêtre. Aucun package VCL ou FireMonkey n'est compilé pour Linux64, car FireMonkey sous Linux nécessite FMX pour Linux (FMXLinux), un module complémentaire tiers.

Voir aussi

- Installation
- Éditions
- Architecture
- Dépannage

Démarrage Rapide

sgcCharts sépare le **modèle** de graphique de la **vue** qui le dessine. Un TsgcChart détient les axes, les séries et les données ; il ne dessine rien par lui-même. Une vue, un contrôle VCL, un contrôle FireMonkey ou un moteur de rendu sans interface graphique, peint ce modèle chaque fois qu'il change. Ce guide construit un graphique étape par étape : d'abord sur un formulaire VCL, puis alimenté avec des données en direct, puis rendu sans aucune vue pour un service ou une application console.

1. Le modèle de graphique et la vue VCL

Créez un **TsgcChart** et donnez-lui un axe du bas et un axe de gauche, puis créez un **TsgcVCLChart** et faites pointer sa propriété **Chart** vers le modèle. La vue s'enregistre elle-même comme écouteur et se redessine chaque fois que le modèle change ; rien d'autre n'est nécessaire pour voir un graphique (vide) sur le formulaire.

```
uses
  Forms, Classes, SysUtils, Controls,
  sgcCharts_Chart, sgcCharts_Types,
  sgcCharts_VCL;

type
  TfrmMain = class(TForm)
  private
    FChart: TsgcChart;
    FView: TsgcVCLChart;
  public
    constructor Create(aOwner: TComponent); override;
  end;

constructor TfrmMain.Create(aOwner: TComponent);
begin
  inherited CreateNew(aOwner);
  Caption := 'sgcCharts Quick Start';
  Width := 820;
  Height := 480;

  FChart := TsgcChart.Create(Self);
  FChart.Axes.Add.Name := 'bottom';
  FChart.Axes[0].Position := capBottom;
  FChart.Axes.Add.Name := 'left';
  FChart.Axes[1].Position := capLeft;

  FView := TsgcVCLChart.Create(Self);
  FView.Parent := Self;
  FView.Align := alClient;
  FView.Chart := FChart;
end;
```

2. Ajouter une série et des données statiques

Une série est un composant à part entière. Créez une TsgcChartSeries_Line, faites pointer sa propriété **Chart** vers le modèle, nommez les axes contre lesquels elle trace, et appelez **AddXY** une fois par point. Ajoutez les deux lignes ci-dessous au constructeur ci-dessus, après la création de la vue.

```
uses
  // ... as above, plus:
  Math,
  sgcCharts_Series;

// inside TfrmMain, add a field:
//   FSeries: TsgcChartSeries_Line;
```

```
// inside the constructor, after FView.Chart := FChart:
var
  i: Integer;
begin
  FSeries := TsgcChartSeries_Line.Create(Self);
  FSeries.Chart := FChart;
  FSeries.AxisX := 'bottom';
  FSeries.AxisY := 'left';
  for i := 0 to 99 do
    FSeries.AddXY(i, Sin(i / 10) * 50 + 50);
end;
```

3. Alimenter avec des données en direct

Pour continuer à ajouter des points après l'affichage du formulaire, déposez un `TTimer` et appelez **AddY** depuis son événement **OnTimer**. **AddY** ajoute un point à la prochaine valeur X en cours, ce qui est pratique lorsque X est un index d'échantillon ou un compteur de ticks plutôt qu'une valeur que l'appelant possède déjà.

```
uses
  // ... as above, plus:
  ExtCtrls;

// inside TfrmMain, add fields:
//   FTimer: TTimer;
//   FTicks: Integer;
// and a method:
//   procedure DoTimer(aSender: TObject);

// inside the constructor, after the series loop:
FTimer := TTimer.Create(Self);
FTimer.Interval := 50;
FTimer.OnTimer := DoTimer;
FTimer.Enabled := True;

procedure TfrmMain.DoTimer(aSender: TObject);
begin
  Inc(FTicks);
  FSeries.AddY(Sin(FTicks / 5) * 50 + 50);
end;
```

Un minuteur appelant **AddY** convient pour un test rapide, mais une application réelle qui trace un flux en direct devrait plutôt utiliser l'un des composants source de données : **TsgcChartDataSource_Manual** pour des valeurs que l'application possède déjà en main, **TsgcChartDataSource_DataSet** pour un `TDataSet`, ou **TsgcChartDataSource_JSON** / **TsgcChartDataSource_CSV** pour un document ou un fichier. Ils ajoutent des ajouts thread-safe, des notifications limitées en fréquence et (pour la source dataset) un mappage de champs qu'un simple minuteur n'offre pas. Pour un flux qui arrive déjà via MQTT, un WebSocket, Server-Sent Events ou un flux de marché Binance, les sources de données du pont se branchent directement sur le client sgcWebSockets correspondant et ne nécessitent aucun sondage.
Mis bout à bout, les étapes 1 à 3 donnent le programme complet et compilable ci-dessous. Le formulaire n'utilise aucun `.dfm`, il construit sa propre mise en page en code, ce qui explique pourquoi le constructeur appelle **CreateNew** au lieu de l'habituel **Create** hérité.

```
program QuickStartVCL;

{$APPTYPE GUI}

uses
  Forms, Classes, SysUtils, Math, ExtCtrls, Controls,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Axis, sgcCharts_Types,
  sgcCharts_VCL;

type
  TfrmMain = class(TForm)
  private
```

```
    FChart: TsgcChart;
    FView: TsgcVCLChart;
    FSeries: TsgcChartSeries_Line;
    FTimer: TTimer;
    FTicks: Integer;
    procedure DoTimer(aSender: TObject);
  public
    constructor Create(aOwner: TComponent); override;
  end;

constructor TfrmMain.Create(aOwner: TComponent);
var
  i: Integer;
begin
  inherited CreateNew(aOwner);
  Caption := 'sgcCharts Quick Start';
  Width := 820;
  Height := 480;

  FChart := TsgcChart.Create(Self);
  FChart.Axes.Add.Name := 'bottom';
  FChart.Axes[0].Position := capBottom;
  FChart.Axes.Add.Name := 'left';
  FChart.Axes[1].Position := capLeft;

  FView := TsgcVCLChart.Create(Self);
  FView.Parent := Self;
  FView.Align := alClient;
  FView.Chart := FChart;

  FSeries := TsgcChartSeries_Line.Create(Self);
  FSeries.Chart := FChart;
  FSeries.AxisX := 'bottom';
  FSeries.AxisY := 'left';
  for i := 0 to 99 do
    FSeries.AddXY(i, Sin(i / 10) * 50 + 50);

  FTicks := 0;
  FTimer := TTimer.Create(Self);
  FTimer.Interval := 50;
  FTimer.OnTimer := DoTimer;
  FTimer.Enabled := True;
end;

procedure TfrmMain.DoTimer(aSender: TObject);
begin
  Inc(FTicks);
  FSeries.AddY(Sin(FTicks / 5) * 50 + 50);
end;

var
  vForm: TfrmMain;
begin
  Application.Initialize;
  Application.CreateForm(TfrmMain, vForm);
  Application.Run;
end.
```

4. Rendu sans interface graphique, sans aucune vue

Un service, un module ISAPI ou une application console n'ont pas de formulaire sur lequel déposer une vue, et n'en ont pas besoin : le même **TsgcChart** et les mêmes séries effectuent leur rendu via un TsgcChartRenderer au lieu d'un **TsgcVCLChart**. **TsgcChartRenderer** n'a aucune dépendance VCL ou FMX, si bien que ce programme ne se lie qu'aux unités de base de sgcCharts.

```
program QuickStartHeadless;

{$APPTYPE CONSOLE}

uses
  SysUtils, Math,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Axis, sgcCharts_Types,
  sgcCharts_Renderer;

var
  vChart: TsgcChart;
  vSeries: TsgcChartSeries_Line;
  vRenderer: TsgcChartRenderer;
  i: Integer;
begin
  vChart := TsgcChart.Create(nil);
  try
    vChart.Title.Text := 'Quick Start (headless)';
    vChart.Axes.Add.Name := 'bottom';
    vChart.Axes[0].Position := capBottom;
    vChart.Axes.Add.Name := 'left';
    vChart.Axes[1].Position := capLeft;

    vSeries := TsgcChartSeries_Line.Create(nil);
    try
      vSeries.Chart := vChart;
      vSeries.AxisX := 'bottom';
      vSeries.AxisY := 'left';
      for i := 0 to 99 do
        vSeries.AddXY(i, Sin(i / 10) * 50 + 50);

      vRenderer := TsgcChartRenderer.Create(nil);
      try
        vRenderer.Chart := vChart;
        vRenderer.Width := 800;
        vRenderer.Height := 400;
        vRenderer.RenderToPNGFile('quickstart.png');
      finally
        vRenderer.Free;
      end;
    finally
      vSeries.Free;
    end;
  finally
    vChart.Free;
  end;
end.
```

L'exécuter écrit `quickstart.png` à côté de l'exécutable, sans fenêtre, sans boucle de messages et sans bureau requis. Voir TsgcChartRenderer pour effectuer le rendu vers SVG, vers un flux, ou vers un tampon de pixels brut à la place.

Pour aller plus loin

- Séries : ligne, ligne rapide, ligne en escalier, aire, barre, nuage de points et chandelier, et ce que chacune dessine.
- Vues : **TsgcVCLChart**, **TsgcFMXChart** et **TsgcChartRenderer**.
- Interaction : zoom, panoramique, réticule et infobulles sur une vue visuelle.
- Financial : indicateurs, oscillateurs, volume et graphiques financiers multi-panneaux.
- Sources de données : flux manuel, dataset, JSON et CSV.
- Sources de données du pont : flux MQTT, WebSocket, SSE et Binance en direct via sgcWebSockets.
- Démos : les projets de démonstration exécutables sur lesquels se base ce démarrage rapide.

Voir aussi

- TsgcChart
- TsgcChartSeries
- TsgcVCLChart
- TsgcChartRenderer
- TsgcChartBuffer

Démos

La distribution sgcCharts fournit un projet de démonstration pour chaque scénario de cette aide : un flux en direct, plusieurs axes, un graphique en chandeliers depuis un CSV, un graphique lié à un dataset, les thèmes et le HiDPI, un flux en direct relié via sgcWebSockets, un graphique mobile FireMonkey, et le rendu sans interface graphique sans aucune vue. Les projets Delphi se trouvent sous `Demos\Delphi\<Name>\` et leurs équivalents C++Builder sous `Demos\CBuilder\<Name>\` ; chaque dossier de démonstration comporte un `readme.html` avec les détails.

Démos VCL

Ces démonstrations placent un **TsgcChart**, un **TsgcVCLChart**, les séries et les sources de données sur un formulaire à la conception. Elles se compilent avec Delphi 7 à Delphi 13, Win32 et Win64.

Démo	Ce qui est illustré	Dossier
LiveLine	Une **TsgcChartSeries_FastLine** avec un tampon circulaire de 1 000 000 de points, alimentée depuis un thread d'arrière-plan via une **TsgcChartDataSource_Manual** à raison de 1 000 à 200 000 points par seconde, avec un **TimeWindow** de 10 s / 60 s / tampon entier, un commutateur de thème clair/sombre, et un réticule et une infobulle sur un million de points.	Delphi\LiveLine\
MultiAxis	Trois séries ligne sur trois axes de valeur (gauche, droite et un second axe gauche), un axe date-heure en bas et la légende à droite ; chaque série choisit ses axes avec **AxisX** et **AxisY**, avec zoom à la molette, zoom par boîte, panoramique et réinitialisation sur chaque axe.	Delphi\MultiAxis\
Candlestick	Une année de données OHLCV quotidiennes depuis un CSV compilé dans l'exécutable, chargée par **TsgcChartDataSource_CSV** dans une série de chandeliers avec le volume en barres sur un second axe ; les noms de colonnes sont mappés sur X, ouverture, plus haut, plus bas, clôture et volume avec des dates ISO 8601, et un commutateur bascule entre chandeliers et barres OHLC.	Delphi\Candlestick\
DataSetChart	Un `TClientDataSet` de 1 000 lignes, affiché dans un `TDBGrid` et lu par deux composants **TsgcChartDataSource_DataSet** dans une série ligne et une série barre ; **XField** et **YField** nomment les champs du dataset, et modifier une ligne dans la grille puis recharger met à jour le graphique.	Delphi\DataSetChart\
Thèmes	Le même graphique dans le thème clair et le thème sombre côte à côte, avec un sélecteur d'échelle (1.0, 1.5, 2.0) et l'export en PNG et SVG via **TsgcVCLChart.SaveToPNG** et **SaveToSVG**. Le projet est `Themes\ChartThemes.dpr` .	Delphi\Themes\
PieDonut	Une **TsgcChartSeries_Pie** et un anneau côte à côte, alimentés en direct depuis une petite `IsgcUITable` éditée dans une grille de chaînes via **TsgcChartDataSource_Table**, avec un curseur de seuil Others, l'éclatement au clic, un sélecteur de mode d'étiquettes et l'export en PNG et SVG.	Delphi\PieDonut\
RadarCompare	Trois produits sur huit critères avec **TsgcChartSeries_Radar** : rempli ou contour, grille polygonale ou circulaire, et un minuteur qui fait évoluer les scores.	Delphi\RadarCompare\
HeatmapLive	Un spectrogramme **TsgcChartSeries_Heatmap** qui ajoute 30 lignes par seconde sur un axe de date et heure avec **Capacity** 300, et une matrice région par mois chargée depuis une `IsgcUITable` avec des	Delphi\HeatmapLive\

	axes de catégories, des étiquettes de valeur et une échelle de couleurs continue ou par paliers.	
Treemap-Funnel	Un **TsgcChartSeries_Treemap** à trois niveaux avec descente et fil d'Ariane, et un **TsgcChartSeries_Funnel** de ventes en direct à six étapes.	Delphi\TreemapFunnel\

Démos du pont sgcWebSockets

Ces démonstrations alimentent les graphiques avec des données en direct provenant de composants sgcWebSockets via les sources de données du pont. Elles nécessitent sgcWebSockets installé et se compilent avec Delphi 2010 à Delphi 13, Win32 et Win64. Les clients, serveurs sgcWebSockets et les sources de données sont placés sur le formulaire à la conception.

Démo	Ce qui est illustré	Dossier
MQTT-Sensors	Trois **TsgcChartDataSource_MQTT** partagent un `TsgcWSPClient_MQTT` et tracent les sujets `sgccharts/sensors/temperature`, `humidity` et `pressure`, abonnés une seule fois à `sgccharts/sensors/+`, contre test.mosquitto.org via WebSocket ou un broker local en TCP brut ; un bouton Simulate publie des valeurs aléatoires toutes les 100 ms si bien que la démonstration fonctionne avec n'importe quel broker, et la pression obtient son propre axe droit.	Delphi\MQTTSensors\
BinanceLive	Deux **TsgcChartDataSource_Binance** sur un même `TsgcWSAPI_Binance` : un flux kline alimente une série Candlestick qui met à jour le chandelier en formation à chaque message, et un flux de transactions alimente une série Line ; les 60 dernières bougies se chargent via l'API REST Binance avant l'abonnement, et le symbole (BTCUSDT ou ETHUSDT) et l'intervalle (1m ou 5m) peuvent être changés en cours de connexion.	Delphi\BinanceLive\
WebSocket-Feed	Un `TsgcWebSocketServer` à l'intérieur de la démonstration diffuse `{"t":<unix ms>,"v":<value>}` toutes les 20 ms, et une **TsgcChartDataSource_WebSocket** sur un `TsgcWebSocketClient` le trace, en lisant **XPath** `t` comme des millisecondes Unix et **YPath** `v` ; le serveur et le client s'exécutent dans le même programme en boucle locale sur 127.0.0.1.	Delphi\WebSocketFeed\
SSE-Feed	Un flux d'événements servi à l'intérieur de la démonstration, lu avec `TsgcHTTP1Client.GetSSE` dans un thread de travail et tracé par une **TsgcChartDataSource_SSE**, qui découpe le flux brut en événements et lit leurs données JSON tandis que le propre **OnSSEMessage** du formulaire continue de se déclencher et annule le flux à l'arrêt.	Delphi\SSEFeed\

Démos FMX

Cette démonstration place un TsgcFMXChart sur un formulaire FireMonkey. Elle se compile avec Delphi 12 et 13, pour Win32, Win64, macOS (OSX64, OSXARM64), iOSDevice64 et Android64.

Démo	Ce qui est illustré	Dossier
FMX-Mobile	Un contrôle à onglets avec une **TsgcChartSeries_FastLine** en direct (un tampon circulaire de 200 000 points alimenté depuis un thread d'arrière-plan à 20 000 points par seconde) et une **TsgcChartSeries_Candlestick** avec 500 chandeliers, dimensionnée pour le tactile ; zoom par pincement et panoramique sur l'axe temporel, le réticule et l'infobulle, un bouton Reset, un **Ti-**	Delphi\FMXMobile\

	meWindow de 10 s / 30 s, un commutateur de thème clair/sombre, et les images par seconde depuis **TsgcFMXChart.FrameCount**.	

Démos de rendu sans interface graphique

Ces démonstrations effectuent le rendu de graphiques sans aucun contrôle visuel, via TsgcChartRenderer, qui n'a aucune dépendance VCL ou FMX et n'a donc besoin d'aucun des deux installés. Elles s'exécutent dans des programmes console, des services Windows, des modules ISAPI et des démons Linux64.

Démo	Ce qui est illustré	Dossier
Headless	Un programme console qui construit en mémoire une série Line de 10 000 points et une série Candlestick de 500 chandeliers et effectue le rendu de chacune vers un fichier PNG et SVG (`line.png` , `line.svg` , `candles.png` , `candles.svg`), en affichant la taille et le CRC32 de chaque fichier. Les données sont générées par arithmétique entière et le rasteriseur est identique bit à bit sur toutes les plateformes, si bien que les fichiers PNG écrits sur Win64 et sur Linux64 partagent le même CRC32. S'exécute sur Win32, Win64 et Linux64.	Delphi\Headless\
HeadlessService	Un service Windows, installé avec `HeadlessService.exe /install` et retiré avec `/uninstall` , qui ajoute une valeur à un graphique en ligne et le rend vers `output\chart.png` toutes les 10 secondes ; comme un service n'a pas de bureau, le graphique est dessiné par **TsgcChartRenderer** plutôt que par un contrôle visuel, et les erreurs de rendu vont dans le journal des événements Windows. S'exécute sur Win32 et Win64.	Delphi\HeadlessService\

Voir aussi

- Démarrage rapide
- Sources de données
- Sources de données du pont
- TsgcChartRenderer

TsgcChart

Unité : sgcCharts_Chart
TsgcChart est le modèle du graphique : un descendant de `TComponent` qui contient les axes, les séries, le titre, sous-titre, légende et marges et le thème et la palette, mais qui ne dessine rien par lui-même et ne dépend d'aucun contrôle visuel. Une vue (un contrôle VCL ou FireMonkey, ou un moteur de rendu sans interface) appelle **AddListener** pour savoir quand le modèle a changé et **Render** pour le dessiner sur un TsgcUICanvas. Le modèle et le dessin étant séparés, le même TsgcChart peut être affiché par plusieurs vues à la fois, ou rendu directement dans un fichier PNG ou SVG sans aucune vue.

Propriétés

Propriété	Type	Description
Theme	TsgcUI-Theme	`utLight` , `utDark` ou `utCustom` . Par défaut `utLight` . Voir Thèmes et palette.
Axes	TsgcChar-tAxes	La collection d'axes. Voir Axes.
Title, Subtitle	TsgcChart-Title	Voir Titre, légende et marges.
Legend	TsgcChart-Legend	Voir Titre, légende et marges.
Margins	TsgcChart-Margins	Voir Titre, légende et marges.
BackgroundColor, PlotBackgroundCo-lor	TsgcUIColor	Stocké uniquement lorsque **Theme** vaut `utCustom` . Définir directement l'une ou l'autre bascule **Theme** sur `utCustom` , sauf lorsque le composant se charge depuis un flux.
Palette	TsgcChart-Palette	Les 8 couleurs automatiques des séries. Voir Thèmes et palette.
MaxFPS	Integer	Limite de rafraîchissement que les vues sont censées respecter. Par défaut 60.
TimeWindow	Double	Secondes, par défaut 0 (désactivé). Tant que non nul, un axe du bas numérique ou date-heure automatique affiche `[newest X - Time-Window, newest X]` au lieu de la plage complète des données.
PlotRect	TsgcUIRect (lecture seule)	Le rectangle de tracé calculé par le dernier **Render**.
Scale	Double	Facteur HiDPI. La vue le définit avant d'appeler **Render** ; le modifier ne déclenche aucune notification.
SeriesCount	Integer (lec-ture seule)	Nombre de séries dont la propriété **Chart** pointe vers ce graphique.
Series[Index: Inte-ger]	TsgcChart-Series (lec-ture seule, default)	La série à Index, base 0.

Méthodes

Méthode	Retour	Description
BeginUpdate / EndUpdate		Regroupe plusieurs modifications afin que les écouteurs ne soient notifiés qu'une seule fois, à **EndUpdate**, au lieu d'après chaque modification.
Invalidate		Notifie immédiatement les écouteurs, ou diffère la notification jusqu'au **EndUpdate** correspondant lorsqu'une mise à jour est ouverte.
AddListener(aEvent: TNotifyEvent)		Enregistre un événement que la vue rappelle quand le graphique doit être redessiné.
RemoveListener(aEvent: TNotifyEvent)		Désenregistre un écouteur ajouté avec **AddListener**.
Render(aCanvas: TsgcUICanvas; const aRect: TsgcUIRect)		Dessine l'arrière-plan, le titre, le sous-titre, la légende, les axes et chaque série visible à l'intérieur de **aRect** (pixels d'appareil) de **aCanvas**. Voir Canvas.
FindAxis(const aName: string)	TsgcChartAxis	L'axe portant ce Name (insensible à la casse), nil si aucun.
AxisRect(aAxis: TsgcChartAxis)	TsgcUIRect	La bande que le dernier **Render** a réservée pour aAxis en dehors du tracé ; un rectangle vide lorsqu'elle est inconnue.
AreaCount, AreaRect(aIndex), AreaMode(aIndex), AreaAt(X, Y)	Integer, TsgcUIRect, TsgcChartPlotMode, Integer	Les zones de tracé du dernier **Render**, leurs rectangles et leurs modes, et la zone située sous un pixel. Voir Zones de tracé.
SeriesAreaRect(aSeries), IsSeriesDrawn(aSeries)	TsgcUIRect, Boolean	L'endroit où le dernier **Render** a dessiné une série, et s'il l'a dessinée.

Modèle et vues

TsgcChart ne possède aucune surface de dessin propre : c'est un simple composant, pas un contrôle, si bien qu'il peut être construit, alimenté et rendu depuis un service ou une application console exactement comme depuis un formulaire. Les séries sont elles-mêmes des composants dotés d'une propriété **Chart** (voir TsgcChartSeries) ; assigner cette propriété enregistre la série auprès du graphique, et la vider (ou détruire la série) la retire. Une vue n'a besoin que de trois choses du modèle : **AddListener** pour être informée du moment où redessiner, **Render** pour effectuer le dessin, et **Scale** défini au préalable sur son facteur HiDPI.

Mises à jour et notifications

Chaque modification du graphique, de ses séries ou de ses axes appelle **Invalidate**, qui notifie immédiatement les écouteurs enregistrés. Regrouper plusieurs modifications entre **BeginUpdate** et **EndUpdate** diffère cette notification jusqu'au **EndUpdate** correspondant, de sorte qu'une vue se redessine une seule fois au lieu d'une fois par modification.

Exemple : construire et rendre un graphique en code

```
uses
  SysUtils, Math,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Axis, sgcCharts_Types,
  sgcUICore_Canvas, sgcUICore_Canvas_Raster, sgcUICore_Canvas_SVG;

var
  vChart: TsgcChart;
  vSeries: TsgcChartSeries_Line;
  vAxisX, vAxisY: TsgcChartAxis;
  vRaster: TsgcUICanvas_Raster;
  vSVG: TsgcUICanvas_SVG;
  i: Integer;
begin
  vChart := TsgcChart.Create(nil);
  try
    vChart.Title.Text := 'Monthly Revenue';

    vAxisX := vChart.Axes.Add;
    vAxisX.Name := 'x';
    vAxisX.Position := capBottom;

    vAxisY := vChart.Axes.Add;
    vAxisY.Name := 'y';
    vAxisY.Position := capLeft;

    vSeries := TsgcChartSeries_Line.Create(nil);
    try
      vSeries.Chart := vChart;
      vSeries.AxisX := 'x';
      vSeries.AxisY := 'y';
      for i := 0 to 99 do
        vSeries.AddXY(i, Sin(i / 10) * 100);

      vRaster := TsgcUICanvas_Raster.Create(800, 400);
      try
        vChart.Scale := 1;
        vChart.Render(vRaster, sgcUIRect(0, 0, 800, 400));
        vRaster.SaveToPNGFile('chart.png');
      finally
        vRaster.Free;
      end;

      vSVG := TsgcUICanvas_SVG.Create(800, 400);
      try
        vChart.Render(vSVG, sgcUIRect(0, 0, 800, 400));
        vSVG.SaveToFile('chart.svg');
      finally
        vSVG.Free;
      end;
    finally
      vSeries.Free;
    end;
  finally
    vChart.Free;
  end;
end;
```

Voir aussi

- TsgcChartSeries
- Axes
- Zones de tracé
- Canvas

TsgcChartSeries

Unité : sgcCharts_Chart
TsgcChartSeries est la base abstraite de chaque série : un descendant de `TComponent` qu'un TsgcChart affiche via sa propriété **Chart**. Elle définit les propriétés de couleur, de visibilité, de légende et de liaison aux axes que toutes les séries partagent, ainsi que deux membres abstraits, **GetDataRange** et **Draw**, que chaque type concret implémente. Les types de série concrets (ligne, aire, ligne en escalier, barre, nuage de points, chandelier, OHLC et les autres) sont des composants à part entière, déclarés dans `sgcCharts_Series` et documentés séparément.

Propriétés

Propriété	Type	Description
Chart	TsgcChart	Le graphique qui affiche cette série. L'assigner enregistre la série auprès du graphique ; la vider, ou la définir sur un autre graphique, la désenregistre d'abord de l'ancien.
Title	string	Texte affiché dans la légende. Voir **LegendText**.
Color	TsgcUIColor	0 (valeur par défaut) signifie une couleur de palette automatique. Voir **EffectiveColor**.
Visible	Boolean	Par défaut True.
ShowInLegend	Boolean	Par défaut True.
AxisX, AxisY	string	Le **Name** de l'axe sur lequel cette série est tracée. Voir Axes.
Decimation	TsgcChartDecimation	`dmNone` , `dmMinMax` ou `dmLTTB` . Par défaut `dmMinMax` . Voir Décimation.
Index	Integer (lecture seule)	Position de la série dans la liste des séries du graphique, -1 lorsqu'elle n'est liée à aucun graphique.

Méthodes

Méthode	Retour	Description
Changed		Demande à **Chart** de se redessiner (appelle son **Invalidate**) ; ne fait rien lorsque **Chart** est nil.
EffectiveColor	TsgcUIColor	**Color**, ou la couleur de palette du graphique à **Index** lorsque **Color** vaut 0.
XAxis	TsgcChartAxis	L'axe nommé par **AxisX**, résolu sur **Chart** ; le premier axe du bas lorsque **AxisX** est vide, nil si introuvable.

YAxis	TsgcChartAxis	L'axe nommé par **AxisY**, résolu sur **Chart** ; le premier axe de gauche lorsque **AxisY** est vide, nil si introuvable.
GetDataRange(var aXMin, aXMax, aYMin, aYMax: Double)	Boolean (abstrait)	Les bornes de données de la série ; False lorsqu'elle n'a pas de données. Implémenté par le type de série concret.
Draw(aCanvas: TsgcUICanvas; aXAxis, aYAxis: TsgcChartAxis; const aPlot: TsgcUIRect; aScale: Double)		Abstrait. Dessine la série à l'intérieur de **aPlot**, déjà rogné à celui-ci ; les valeurs sont converties avec **aXAxis.ValueToPixel** et **aYAxis.ValueToPixel**.
DrawLegendSymbol(aCanvas: TsgcUICanvas; const aRect: TsgcUIRect; aScale: Double)		Dessine le symbole de légende. Par défaut, un carré plein de **EffectiveColor** ; les types concrets le redéfinissent.
LegendText	string	**Title** lorsqu'il est défini, sinon le **Name** du composant.
PlotMode	TsgcChartPlotMode (virtual)	Mode que cette série donne à sa zone de tracé : `cpmXY` (par défaut), `cpmPolar` ou `cpmCell` . Voir Zones de tracé.
DrawArea(aCanvas; const aRect; const aSeries: TsgcChartSeriesArray; aCount: Integer; aScale: Double)	(virtual)	Dessine une zone polaire (appelée sur le premier radar de la zone, avec tous les radars dessinés dans **aSeries**) ou une cellule (**aSeries** ne contient que cette série). Le canevas est rogné à la zone. Par défaut, ne fait rien.
HitTestArea(const aRect; const aSeries; aCount: Integer; X, Y: Double; var aHit: TsgcChartHitInfo)	Boolean (virtual)	Test de survol d'une zone polaire ou d'une cellule, avec les arguments de **DrawArea**. Par défaut, renvoie False.
HitTestPoint(const aPlot: TsgcUIRect; X, Y: Double; var aHit: TsgcChartHitInfo)	Boolean (virtual)	Test de survol d'une série XY que l'interaction ne connaît pas, comme la carte de chaleur. Par défaut, renvoie False.
TooltipText(const aHit: TsgcChartHitInfo)	string (virtual)	Infobulle d'un survol de cette série ; une chaîne vide (par défaut) utilise le format de l'interaction. Plusieurs lignes sont dessinées comme de simples lignes.
AreaClick(const aHit: TsgcChartHitInfo)	(virtual)	Un clic sur un point touché de cette série, après **OnSeriesClick**. L'éclatement (camembert) et la descente (treemap) s'exécutent ici. Par défaut, ne fait rien.
LegendItemCount	Integer (virtual)	Entrées de légende de la série, 1 par défaut ; un camembert en renvoie une par secteur.
MeasureLegendItem(aCanvas; aItem: Integer; const aFont; aScale: Double; out aWidth, aHeight: Double)	(virtual)	Taille de l'entrée de légende **aItem**. Par défaut, le symbole, un espace et **LegendText** sur une ligne.
DrawLegendItem(aCanvas; aItem: Integer; const aRect; const aFont; aScale: Double)	(virtual)	Dessine l'entrée de légende **aItem** dans **aRect**. Par défaut, appelle **DrawLegendSymbol** et dessine **LegendText**.
GetDataCategories(aAxis: TsgcChartAxis; aList: TStrings)	Boolean (virtual)	Textes de catégories à l'exécution pour un axe de catégories sans **Categories** ; True lorsque **aList** a été remplie. La carte de chaleur l'utilise pour les éti-

		quettes de ses lignes et de ses cellules. Par défaut, renvoie False.

Liaison au graphique et aux axes

Une série n'appartient pas à son graphique : définir **Chart** ne fait que l'enregistrer, si bien que l'application continue de posséder et de libérer l'instance de la série (ou lui donne un propriétaire à la manière Delphi habituelle). **AxisX** et **AxisY** lient la série à l'un des axes de **Chart.Axes** par **Name** (insensible à la casse) ; un nom vide se replie sur le premier axe du bas pour X et le premier axe de gauche pour Y, si bien qu'un graphique avec un axe de chaque type n'oblige pas chaque série à les nommer.

Couleur et palette

Laisser **Color** à 0 fait que la série prend la couleur de **Palette** du graphique correspondant à son **Index**, en parcourant les 8 couleurs de la palette. Définir **Color** sur une valeur non nulle remplace la palette pour cette seule série. Voir Thèmes et palette.

Décimation

Decimation contrôle la façon dont une série concrète réduit un grand tampon à environ un ensemble de points par colonne de pixels avant le dessin. Voir Décimation pour ce que conserve chaque mode.

Points d'extension des zones de tracé

Chaque point d'extension a par défaut le comportement XY, de sorte qu'une série qui ne redéfinit que **GetDataRange** et **Draw** reste une série XY. Les séries camembert, radar, entonnoir et treemap redéfinissent **PlotMode**, **DrawArea** et **HitTestArea** ; la carte de chaleur reste une série XY et redéfinit **HitTestPoint** et **GetDataCategories**. Voir Zones de tracé pour savoir comment un mode est choisi.

Exemple : une série de type ligne liée à des axes nommés

```
uses
  SysUtils, Math,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Axis, sgcCharts_Types,
  sgcUICore_Canvas, sgcUICore_Canvas_Raster;

var
  vChart: TsgcChart;
  vSeries: TsgcChartSeries_Line;
  vBottom, vLeft: TsgcChartAxis;
  i: Integer;
  vRaster: TsgcUICanvas_Raster;
begin
  vChart := TsgcChart.Create(nil);
  try
    // ... keep the axes Axes.Add returns: Axes[0] and Axes[1] are the
    // default X and Y axes, not the new ones; no series uses the defaults
    // here, so they are hidden
    vChart.Axes.Find('X').Visible := False;
    vChart.Axes.Find('Y').Visible := False;
    vBottom := vChart.Axes.Add;
    vBottom.Name := 'bottom';
    vBottom.Position := capBottom;
    vLeft := vChart.Axes.Add;
    vLeft.Name := 'left';
    vLeft.Position := capLeft;

    vSeries := TsgcChartSeries_Line.Create(nil);
    try
```

```
      vSeries.Chart := vChart;
      vSeries.Title := 'Signal';
      vSeries.AxisX := 'bottom';
      vSeries.AxisY := 'left';
      for i := 0 to 199 do
        vSeries.AddXY(i, Sin(i / 15) * 50 + 50);

      Writeln('series index: ', vSeries.Index);
      Writeln('effective colour: $', IntToHex(vSeries.EffectiveColor, 8));

      vRaster := TsgcUICanvas_Raster.Create(600, 300);
      try
        vChart.Render(vRaster, sgcUIRect(0, 0, 600, 300));
        vRaster.SaveToPNGFile('series.png');
      finally
        vRaster.Free;
      end;
    finally
      vSeries.Free;
    end;
  finally
    vChart.Free;
  end;
end;
```

Voir aussi

- TsgcChart
- Axes
- Décimation
- Zones de tracé

TsgcChartAxis et TsgcChartAxes

Unité : sgcCharts_Axis
TsgcChartAxes est la `TOwnedCollection` exposée en tant que TsgcChart.**Axes** ; chaque élément est un **TsgcChartAxis** (un `TCollectionItem`). Un graphique peut contenir plusieurs axes du même côté, ce qui permet à un graphique d'afficher deux échelles de valeurs à la fois : chaque série se lie à un axe par son nom (voir TsgcChartSeries).

Propriétés de TsgcChartAxis

Propriété	Type	Description
Name	string	Identifiant auquel les séries se lient via **AxisX** / **AxisY**, et que **TsgcChart.FindAxis** recherche.
AxisType	TsgcChartAxisType	`catNumeric` (par défaut), `catLogarithmic`, `catDateTime` ou `catCategory`.
Position	TsgcChartAxisPosition	`capLeft` (par défaut), `capRight`, `capTop` ou `capBottom`. Les axes du haut et du bas sont horizontaux (X) ; ceux de gauche et de droite sont verticaux (Y). Voir **IsHorizontal**.
Title, TitleStyle	string, TsgcUITextStyle	Titre de l'axe et son style de texte.
LabelStyle	TsgcUITextStyle	Style de texte des étiquettes de graduation.
Visible	Boolean	Par défaut True.
Automatic	Boolean	Par défaut True. Lorsque True, la plage provient des données des séries liées à cet axe. Lorsque False, **Minimum** / **Maximum** sont utilisés tels quels.
Minimum, Maximum	Double	Plage manuelle, utilisée quand **Automatic** est False.
Inverted	Boolean	Par défaut False.
LabelFormat	string	Vide pour un formatage automatique. Pour `catNumeric` / `catLogarithmic`, un masque `FormatFloat` indépendant de la locale (`'0.00'`, `'#,##0.0'`). Pour `catDateTime`, les lettres de format date-heure ci-dessous.
GridVisible, GridColor, LineColor	Boolean, TsgcUIColor	Visibilité des lignes de grille, et couleurs des lignes de grille et d'axe. **GridVisible** par défaut True.
TickLength	Double	Longueur des graduations en pixels d'appareil à l'échelle 1.
MinTickSpacing	Double	Nombre minimal de pixels d'appareil à l'échelle 1 entre deux graduations. Par défaut 50.
Categories	TStrings	Étiquettes de graduation pour un axe `catCategory`, une par position.

Méthodes de TsgcChartAxis

Méthode	Retour	Description
IsHorizontal	Boolean	True lorsque **Position** vaut `capTop` ou `capBottom`.
SetDataRange(aMin, aMax: Double; aExact: Boolean = False)		Communique les bornes de données auxquelles un axe automatique doit s'élargir pour arrondir aux valeurs rondes ; **aExact** affiche **[aMin, aMax]** tel quel, sans élargissement (utilisé pour **TimeWindow**).
ClearDataRange		Efface les bornes définies par **SetDataRange**.
SetDataCategories(aList: TStrings)		Textes de catégories à l'exécution d'un axe de catégories sans **Categories** ; le graphique les définit pendant le rendu à partir de **GetDataCategories** des séries, comme **SetDataRange**. Ils ne sont jamais enregistrés dans le flux.
ClearDataCategories		Efface les textes définis par **SetDataCategories**.
ZoomRange(aMin, aMax: Double)		Restreint la plage visible à **[aMin, aMax]**.
ResetZoom		Efface une plage définie par **ZoomRange**.
Zoomed	Boolean (lecture seule)	True tant qu'une plage de zoom est active.
VisibleMin, VisibleMax	Double (lecture seule)	La plage actuellement en vigueur : la plage de zoom, sinon la plage manuelle ou de données.
Calculate(aCanvas: TsgcUICanvas; aPixelLength: Double; aScale: Double = 1)		Calcule les graduations pour un tracé de **aPixelLength** pixels le long de cet axe. **aCanvas** mesure les étiquettes pour l'éclaircissage et peut être nil.
TickCount, TickValue[Index], TickLabel[Index]	Integer, Double, string	Les graduations calculées par le dernier **Calculate**. **TickLabel** est vide pour une étiquette supprimée par l'éclaircissage.
Step, DateTimeUnit	Double, TsgcChartDateTimeUnit	L'espacement entre les graduations, et, pour un axe `catDateTime`, l'unité à laquelle cet espacement a été arrondi.
ValueToPixel(aValue, aStart, aLength: Double)	Double	Convertit une valeur de données en une position de pixel d'appareil.
PixelToValue(aPixel, aStart, aLength: Double)	Double	L'inverse de **ValueToPixel**.
MeasureSize(aCanvas: TsgcUICanvas; aScale: Double = 1)	Double	Épaisseur dont cet axe a besoin en dehors du tracé (graduations, étiquettes, titre) ; 0 lorsqu'il n'est pas **Visible**.
Draw, DrawGrid		Dessine l'axe, ou ses lignes de grille, pour le dernier **Calculate**.

Membres de TsgcChartAxes

Membre	Description
Add: TsgcChartAxis	Ajoute un nouvel axe et le retourne.
Find(const aName: string): TsgcChartAxis	Premier axe portant ce Name (insensible à la casse), nil si aucun.
Items[Index: Integer]: TsgcChartAxis (default)	Axe à Index.

Plage automatique et zoom

Tant que **Automatic** est True, le graphique alimente l'axe avec les bornes de données combinées de chaque série qui lui est liée, à chaque rendu, via **SetDataRange** ; un axe automatique élargit cette plage pour arrondir aux valeurs de graduation, sauf si le graphique appelle **SetDataRange** avec **aExact** à True, ce qu'il fait pour **TsgcChart.TimeWindow**. Définir **Automatic** sur False et fournir **Minimum** / **Maximum** fixe la plage à la place. **ZoomRange** et **ResetZoom** s'appliquent par-dessus l'une ou l'autre, pour un zoom interactif ; **VisibleMin** / **VisibleMax** indiquent la plage réellement en vigueur.

Format des étiquettes

Pour un axe `catDateTime`, **LabelFormat** utilise des lettres de format date-heure indépendantes de la locale d'exécution : `yyyy`, `yy`, `mmmm`, `mmm`, `mm`, `m`, `dd`, `d`, `hh`, `h`, `nn`, `n`, `ss`, `s`, `zzz`. Le texte entre guillemets est recopié tel quel. `mm` désigne toujours le mois ; utilisez `nn` pour les minutes. Laissé vide, l'axe choisit automatiquement un format à partir du **DateTimeUnit** de l'espacement des graduations.

Catégories

Un axe `catCategory` place une graduation par élément de **Categories**, dans l'ordre de la liste, au lieu de calculer des graduations numériques ; les séries qui lui sont liées se tracent à des positions entières correspondant à ces éléments. Lorsque **Categories** est vide, une série peut fournir les textes à l'exécution : une carte de chaleur étiquette ainsi ses lignes et ses cellules.

Axes de valeurs multiples

Un graphique n'est pas limité à un seul axe X et un seul axe Y : ajouter un second axe avec la même **Position** qu'un axe existant, et y faire pointer certaines séries via **AxisX** / **AxisY**, donne à ces séries leur propre échelle pendant que les autres continuent d'utiliser le premier axe. Une série qui laisse **AxisX** ou **AxisY** vide se lie respectivement au premier axe du bas et au premier axe de gauche.

Exemple : un axe du bas date-heure et un axe de gauche numérique

```
uses
  SysUtils, Math,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Axis, sgcCharts_Types,
  sgcUICore_Canvas, sgcUICore_Canvas_Raster;

var
  vChart: TsgcChart;
  vAxisX, vAxisY: TsgcChartAxis;
  vSeries: TsgcChartSeries_Line;
  vNow: TDateTime;
```

```
  i: Integer;
  vRaster: TsgcUICanvas_Raster;
begin
  vChart := TsgcChart.Create(nil);
  try
    vAxisX := vChart.Axes.Add;
    vAxisX.Name := 'time';
    vAxisX.Position := capBottom;
    vAxisX.AxisType := catDateTime;
    vAxisX.LabelFormat := 'dd/mm hh:nn';

    vAxisY := vChart.Axes.Add;
    vAxisY.Name := 'value';
    vAxisY.Position := capLeft;
    vAxisY.LabelFormat := '0.00';

    vSeries := TsgcChartSeries_Line.Create(nil);
    try
      vSeries.Chart := vChart;
      vSeries.AxisX := 'time';
      vSeries.AxisY := 'value';
      vNow := Now;
      for i := 0 to 59 do
        vSeries.AddXY(vNow + (i / 1440), 50 + Sin(i / 5) * 10);

      { Step and VisibleMin / VisibleMax are only computed by Render (it
        calls Calculate on every axis), so read them after rendering }
      vRaster := TsgcUICanvas_Raster.Create(700, 350);
      try
        vChart.Render(vRaster, sgcUIRect(0, 0, 700, 350));
        vRaster.SaveToPNGFile('axes.png');

        Writeln('X ticks: ', vAxisX.TickCount, ', unit: ', Ord(vAxisX.DateTimeUnit));
        Writeln('Y visible range: ', vAxisY.VisibleMin, ' .. ', vAxisY.VisibleMax);
      finally
        vRaster.Free;
      end;
    finally
      vSeries.Free;
    end;
  finally
    vChart.Free;
  end;
end;
```

Voir aussi

- TsgcChart
- TsgcChartSeries

Titre, légende et marges

Unité : sgcCharts_Chart
Trois petites classes `TPersistent` portent la décoration périphérique du graphique : **TsgcChartTitle** (utilisée à la fois pour TsgcChart.**Title** et **Subtitle**), **TsgcChartLegend** (**TsgcChart.Legend**) et **TsgcChartMargins** (**TsgcChart.Margins**). Toutes trois déclenchent une notification de changement interne qui invalide le graphique, de sorte que les modifier à l'exécution provoque un nouveau tracé automatique.

Propriétés de TsgcChartTitle

Propriété	Type	Description
Text	string	Le texte du titre. Un **Text** vide ne réserve aucun espace ; le titre est omis lorsqu'il est vide ou non **Visible**.
TextStyle	TsgcUITextStyle	Taille, gras et couleur de la police. Voir ci-dessous.
Visible	Boolean	Par défaut True.
Alignment	TAlignment	Par défaut `taCenter`.

Propriétés de TsgcChartLegend

Propriété	Type	Description
Visible	Boolean	Par défaut True.
Position	TsgcChartLegendPosition	`clpTop` (par défaut), `clpBottom`, `clpLeft` ou `clpRight`.
TextStyle	TsgcUITextStyle	Police des entrées de la légende.
BackgroundColor	TsgcUIColor	Par défaut 0 (alpha 0, transparent).
Padding	Integer	Par défaut 4, pixels d'appareil à l'échelle 1.

Une série apparaît dans la légende lorsque son propre **ShowInLegend** vaut True (voir TsgcChartSeries) ; le texte de légende est le **LegendText** de la série.

Propriétés de TsgcChartMargins

Propriété	Type	Description
Left, Top, Right, Bottom	Integer	Marge extérieure en pixels d'appareil à l'échelle 1, autour de l'ensemble du graphique, en dehors du titre, de la légende et des axes. Par défaut 10 de chaque côté.

Propriétés de TsgcUITextStyle

Unité : sgcCharts_Types. Utilisée par **TsgcChartTitle.TextStyle**, **TsgcChartLegend.TextStyle** et **TsgcChartAxis.TitleStyle** / **LabelStyle**.

Propriété	Type	Description
Size	Double	Taille de police en points. Par défaut 11.
Bold	Boolean	Par défaut False.
Color	TsgcUIColor	

ToFontStyle(aScale: Double = 1) retourne le **TsgcUIFontStyle** correspondant pour un appel de canevas, avec **Size** multiplié par **aScale**.

Éléments de légende et barres de couleurs

Une série peut ajouter plusieurs entrées à la légende, et les lignes de la légende prennent la hauteur de leur entrée la plus haute. Un camembert ajoute une entrée par secteur dessiné, Others en dernier ; un entonnoir une par étape dessinée ; un treemap en mode `ctcGroup` une par groupe de premier niveau. Une carte de chaleur, et un treemap en mode `ctcValue`, ajoutent une barre de couleurs avec le minimum et le maximum de leur échelle de couleurs, horizontale ou verticale selon la position de la légende. **ShowInLegend** à False masque toutes les entrées d'une série, et une série ignorée par sa zone de tracé n'en affiche aucune.

Exemple : titre, sous-titre, position de la légende et marges

```
uses
  SysUtils, Math,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Axis, sgcCharts_Types,
  sgcUICore_Canvas, sgcUICore_Canvas_Raster;

var
  vChart: TsgcChart;
  vSeries: TsgcChartSeries_Line;
  i: Integer;
  vRaster: TsgcUICanvas_Raster;
begin
  vChart := TsgcChart.Create(nil);
  try
    vChart.Title.Text := 'Server Load';
    vChart.Subtitle.Text := 'last 24 hours';
    vChart.Subtitle.TextStyle.Size := 9;

    vChart.Legend.Position := clpBottom;
    vChart.Legend.Padding := 6;

    vChart.Margins.Left := 16;
    vChart.Margins.Top := 16;
    vChart.Margins.Right := 16;
    vChart.Margins.Bottom := 16;

    vChart.Axes.Add.Position := capBottom;
    vChart.Axes.Add.Position := capLeft;

    vSeries := TsgcChartSeries_Line.Create(nil);
    try
      vSeries.Chart := vChart;
      vSeries.Title := 'CPU %';
      for i := 0 to 49 do
        vSeries.AddXY(i, 40 + Sin(i / 6) * 20);
```

```
      vRaster := TsgcUICanvas_Raster.Create(640, 360);
      try
        vChart.Render(vRaster, sgcUIRect(0, 0, 640, 360));
        vRaster.SaveToPNGFile('title_legend.png');
      finally
        vRaster.Free;
      end;
    finally
      vSeries.Free;
    end;
  finally
    vChart.Free;
  end;
end;
```

Voir aussi

- TsgcChart
- TsgcChartSeries

Thèmes et palette

Unité : sgcCharts_Chart (TsgcChartPalette), sgcCharts_Types (TsgcUITheme)
TsgcChart.**Theme** choisit en une seule fois les couleurs intégrées d'arrière-plan, de texte, de grille et de ligne d'axe, ainsi que la palette de séries par défaut ; **Palette** est la liste de 8 couleurs que les séries utilisent automatiquement par index.

TsgcUITheme

Valeur	Description
utLight	Par défaut. Arrière-plan clair, texte sombre, grille claire.
utDark	Arrière-plan sombre, texte clair, grille sombre.
utCustom	Laisse **BackgroundColor**, **PlotBackgroundColor** et **Palette** exactement tels quels ; le graphique n'applique aucune couleur intégrée.

`utLight` et `utDark` possèdent chacun leur propre **Palette** par défaut (8 couleurs, utilisées dans l'ordre d'index des séries). Définir directement **BackgroundColor**, **PlotBackgroundColor** ou une couleur de **Palette** fait automatiquement basculer **Theme** sur `utCustom`, sauf lorsque le graphique se charge depuis un flux ; à partir de là, redéfinir **Theme** sur `utLight` ou `utDark` réapplique les couleurs de ce thème et écrase les couleurs personnalisées.

Propriétés de TsgcChartPalette

Propriété	Type	Description
Colors[Index: Integer] (default)	TsgcUIColor	L'index est pris modulo 8, de sorte qu'il ne provoque jamais d'erreur pour un index de série hors limites.
Color1 .. Color8	TsgcUIColor	Les 8 couleurs sous forme de propriétés publiées, index 0 à 7.

Méthodes de TsgcChartPalette

Méthode	Description
LoadColors(const aColors: array of TsgcUIColor)	Copie jusqu'à 8 couleurs de **aColors** dans **Color1**.. **Color8** dans l'ordre.

Une série dont **Color** vaut 0 (valeur par défaut) prend **Palette.Colors[Series.Index]** via **EffectiveColor** ; voir TsgcChartSeries.

Exemple : thème sombre, puis une palette personnalisée

```
uses
  SysUtils, Math,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Axis, sgcCharts_Types,
  sgcUICore_Canvas, sgcUICore_Canvas_Raster;

var
```

```
  vChart: TsgcChart;
  vSeries1, vSeries2: TsgcChartSeries_Line;
  i: Integer;
  vRaster: TsgcUICanvas_Raster;
begin
  vChart := TsgcChart.Create(nil);
  try
    vChart.Theme := utDark;
    vChart.Axes.Add.Position := capBottom;
    vChart.Axes.Add.Position := capLeft;

    vSeries1 := TsgcChartSeries_Line.Create(nil);
    vSeries2 := TsgcChartSeries_Line.Create(nil);
    try
      vSeries1.Chart := vChart;
      vSeries1.Title := 'A';
      vSeries2.Chart := vChart;
      vSeries2.Title := 'B';
      for i := 0 to 39 do
      begin
        vSeries1.AddXY(i, Sin(i / 5) * 20 + 50);
        vSeries2.AddXY(i, Cos(i / 5) * 20 + 50);
      end;

      { override the first automatic colour; Theme switches to utCustom }
      vChart.Palette.Color1 := $FFFFA500;
      Writeln('theme after palette edit: ', Ord(vChart.Theme));

      vRaster := TsgcUICanvas_Raster.Create(640, 320);
      try
        vChart.Render(vRaster, sgcUIRect(0, 0, 640, 320));
        vRaster.SaveToPNGFile('theme_palette.png');
      finally
        vRaster.Free;
      end;
    finally
      vSeries1.Free;
      vSeries2.Free;
    end;
  finally
    vChart.Free;
  end;
end;
```

Voir aussi

- TsgcChart
- TsgcChartSeries

Zones de tracé

Unit: sgcCharts_Chart
Une zone de tracé est le tracé d'un graphique, ou un panneau visible lorsque le graphique comporte des panneaux. À chaque rendu, le graphique attribue à chaque zone un mode, calculé d'après les séries visibles qui lui sont liées :

Mode	Séries	Ce que la zone dessine
`cpmXY`	Courbe, aire, barres, nuage de points, chandeliers, carte de chaleur et les autres séries XY	Axes, grille, et toutes les séries XY via **Draw**.
`cpmPolar`	TsgcChartSeries_Radar	Une grille polaire ; le premier radar dessine tous les radars de la zone par-dessus.
`cpmCell`	TsgcChartSeries_Pie, TsgcChartSeries_Funnel, TsgcChartSeries_Treemap	Une grille de cellules, une par série, chacune avec un bandeau de titre ; aucun axe.

La règle de mélange

XY l'emporte dès qu'une série XY est visible dans la zone. Sinon, la première série visible choisit entre polaire et cellule. Les séries qui ne correspondent pas au mode de leur zone sont ignorées : elles ne sont pas dessinées, ne répondent pas au survol ou au clic et n'apparaissent pas dans la légende. Pour afficher un camembert à côté d'un graphique en courbes, donnez au graphique deux panneaux et liez chaque série à l'axe de son panneau, ou utilisez deux graphiques.

Membres de zone de TsgcChart

Méthode	Valeur de retour	Description
AreaCount	Integer (lecture seule)	Zones de tracé du dernier **Render** : 1 pour le tracé, ou une par panneau visible dans l'ordre de la collection.
AreaRect(aIndex: Integer)	TsgcUIRect	Rectangle d'une zone, vide lorsque **aIndex** est hors limites.
AreaMode(aIndex: Integer)	TsgcChartPlotMode	Mode d'une zone, `cpmXY` lorsque **aIndex** est hors limites.
AreaAt(X, Y: Double)	Integer	Index de la zone située sous un pixel du graphique, -1 s'il n'y en a pas.
SeriesAreaRect(aSeries: TsgcChartSeries)	TsgcUIRect	La cellule d'une série à cellule dessinée, la zone d'une série polaire dessinée, le rectangle de tracé du panneau d'une série XY, vide pour toute autre série.
IsSeriesDrawn(aSeries: TsgcChartSeries)	Boolean	True lorsque le dernier **Render** a dessiné la série : visible, du mode de sa zone et, pour une série XY, avec ses deux axes.

Les cellules sont disposées en grille qui remplit la zone. Avec plusieurs cellules, chacune commence sous un bandeau de titre qui contient le **Title** de la série. L'interaction achemine le survol, le clic et les infobulles vers les séries

de tous les modes, mais le zoom, le déplacement et le réticule ne fonctionnent que dans les zones XY. L'auteur d'une série implémente les points d'extension listés dans TsgcChartSeries.

Exemple : deux séries à cellule, puis une série XY

```
uses
  SysUtils,
  sgcUICore_Theme, sgcUICore_Canvas, sgcUICore_Canvas_Raster,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Series_Category,
  sgcCharts_Series_Pie, sgcCharts_Series_Funnel;

const
  CS_MODES: array [TsgcChartPlotMode] of string = ('XY', 'polar', 'cell');

var
  vChart: TsgcChart;
  vPie: TsgcChartSeries_Pie;
  vFunnel: TsgcChartSeries_Funnel;
  vLine: TsgcChartSeries_Line;
  vRaster: TsgcUICanvas_Raster;
  vRect: TsgcUIRect;
  i: Integer;
begin
  vChart := TsgcChart.Create(nil);
  vRaster := TsgcUICanvas_Raster.Create(800, 400);
  vPie := TsgcChartSeries_Pie.Create(nil);
  vFunnel := TsgcChartSeries_Funnel.Create(nil);
  vLine := TsgcChartSeries_Line.Create(nil);
  try
    vPie.Chart := vChart;
    vPie.Title := 'Share';
    vPie.SetValue('Web', 55);
    vPie.SetValue('Store', 30);
    vPie.SetValue('Phone', 15);

    vFunnel.Chart := vChart;
    vFunnel.Title := 'Pipeline';
    vFunnel.SetValue('Leads', 900);
    vFunnel.SetValue('Offers', 420);
    vFunnel.SetValue('Orders', 160);

    // two cell series: the plot area is split into two cells
    vChart.Render(vRaster, sgcUIRect(0, 0, 800, 400));
    vRaster.SaveToPNGFile('cells.png');
    Writeln('areas: ', vChart.AreaCount, ', mode: ',
      CS_MODES[vChart.AreaMode(0)]);
    vRect := vChart.SeriesAreaRect(vFunnel);
    Writeln('funnel cell: ', FormatFloat('0', vRect.Left), ', ',
      FormatFloat('0', vRect.Top), ' to ', FormatFloat('0', vRect.Right),
      ', ', FormatFloat('0', vRect.Bottom));

    // one visible XY series turns the area into an XY area:
    // the pie and the funnel are skipped
    vLine.Chart := vChart;
    vLine.Title := 'Orders per day';
    for i := 0 to 9 do
      vLine.AddXY(i, 10 + i * i);
    vChart.Render(vRaster, sgcUIRect(0, 0, 800, 400));
    vRaster.SaveToPNGFile('xy.png');
    Writeln('mode: ', CS_MODES[vChart.AreaMode(0)], ', pie drawn: ',
      vChart.IsSeriesDrawn(vPie), ', line drawn: ',
      vChart.IsSeriesDrawn(vLine));
  finally
    vLine.Free;
    vFunnel.Free;
    vPie.Free;
```

```
    vRaster.Free;
    vChart.Free;
  end;
end.
```

Voir aussi

- TsgcChart
- TsgcChartSeries
- Panneaux
- TsgcChartInteraction

Échelle de couleurs

Unit: sgcCharts_ColorScale
TsgcChartColorScale associe une valeur à une couleur au moyen d'une table de correspondance (LUT) de 256 entrées construite à partir de deux ou trois repères de couleur. C'est la propriété **ColorScale** de TsgcChartSeries_Heatmap et de TsgcChartSeries_Treemap en mode `ctcValue`, et la légende l'affiche sous la forme d'une barre de couleurs avec le minimum et le maximum.

Propriétés

Propriété	Type	Description
MinColor	TsgcUIColor	Couleur du minimum ; 0 (par défaut) utilise le fond de tracé du thème.
MidColor	TsgcUIColor	Couleur du milieu de la plage ; 0 (par défaut) signifie deux repères.
MaxColor	TsgcUIColor	Couleur du maximum ; 0 (par défaut) utilise la couleur de la série, sinon la première couleur de la palette.
Steps	Integer	0 (par défaut) est une échelle continue ; de 2 à 32, c'est une palette par paliers avec ce nombre de bandes.
Automatic	Boolean	Par défaut True : la plage va du minimum au maximum des cellules ou tuiles visibles. False utilise **Minimum** et **Maximum**.
Minimum, Maximum	Double	La plage fixe. Par défaut 0 et 100.
NullColor	TsgcUIColor	Couleur des cellules NaN ; 0 (par défaut) les laisse non peintes.
RangeMin, RangeMax	Double (lecture seule)	La plage définie par le dernier rendu.
LUT[aIndex]	TsgcUIColor (lecture seule)	Une entrée de la table de correspondance, de 0 à 255.
OnChange	TNotifyEvent	Se déclenche lorsqu'une propriété change ; la série propriétaire se redessine.

Méthodes

Méthode	Valeur de retour	Description
Prepare(aDefaultMinColor, aDefaultMaxColor: TsgcUIColor; const aDataMin, aDataMax: Double)		Appelée par la série avant de dessiner : résout les couleurs à zéro, reconstruit la LUT lorsqu'un repère ou **Steps** a changé et définit la plage. Une plage de largeur nulle associe toute valeur à l'entrée 128.
IndexOf(const aValue: Double)	Integer	Entrée de la LUT pour une valeur, de 0 à 255 ; -1 pour NaN.

ColorOf(const aValue: Double)	TsgcUI-Color	Couleur de la LUT pour une valeur ; **NullColor** pour NaN.
DrawBar, MeasureColorBar, DrawColor-Bar		Dessinent la barre et la barre de couleurs de la légende avec ses étiquettes ; utilisées par les éléments de légende de la série.

Exemple : une échelle de feux tricolores par paliers sur une plage fixe

```
uses
  SysUtils,
  sgcUICore_Theme, sgcUICore_Canvas, sgcUICore_Canvas_Raster,
  sgcCharts_Chart, sgcCharts_ColorScale, sgcCharts_Series_Heatmap;

var
  vChart: TsgcChart;
  vHeat: TsgcChartSeries_Heatmap;
  vScale: TsgcChartColorScale;
  vRaster: TsgcUICanvas_Raster;
  vValues: array [0 .. 9] of Double;
  vHour, j: Integer;
begin
  vChart := TsgcChart.Create(nil);
  try
    vHeat := TsgcChartSeries_Heatmap.Create(nil);
    try
      vHeat.Chart := vChart;
      vHeat.Title := 'Load %';
      // three colour stops and five steps on a fixed 0..100 range
      vScale := vHeat.ColorScale;
      vScale.MinColor := $FF2E7D32;
      vScale.MidColor := $FFFDD835;
      vScale.MaxColor := $FFC62828;
      vScale.Steps := 5;
      vScale.Automatic := False;
      vScale.Minimum := 0;
      vScale.Maximum := 100;
      vScale.NullColor := $FFBDBDBD;

      // one line per hour (X), ten servers per line (Y)
      vHeat.BeginUpdate;
      try
        for vHour := 0 to 23 do
        begin
          for j := 0 to High(vValues) do
            vValues[j] := (vHour * 7 + j * 13) mod 100;
          vHeat.AppendLine(vHour, vValues);
        end;
      finally
        vHeat.EndUpdate;
      end;

      vRaster := TsgcUICanvas_Raster.Create(800, 360);
      try
        vChart.Render(vRaster, sgcUIRect(0, 0, 800, 360));
        vRaster.SaveToPNGFile('scale.png');
      finally
        vRaster.Free;
      end;

      // the range and the colours of the last render
      Writeln('range: ', FormatFloat('0', vScale.RangeMin), ' to ',
        FormatFloat('0', vScale.RangeMax));
```

```
      Writeln('42 is LUT entry ', vScale.IndexOf(42), ', colour $',
        IntToHex(vScale.ColorOf(42), 8));
    finally
      vHeat.Free;
    end;
  finally
    vChart.Free;
  end;
end.
```

Voir aussi

- TsgcChartSeries_Heatmap
- TsgcChartSeries_Treemap
- Titre, légende et marges
- THEMEFR

TsgcChartSeries_Line, TsgcChartSeries_FastLine, TsgcChartSeries_StepLine

Unit: sgcCharts_Series
Ces trois séries descendent de **TsgcChartSeries_XY**, la base commune de toute série X/Y (ligne, ligne rapide, ligne en escalier, aire, barre et nuage de points, voir TsgcChartSeries pour les propriétés partagées par toutes les séries). TsgcChartSeries_XY possède le tampon de points et les méthodes qui l'alimentent ; **TsgcChartSeries_Line** dessine une polyligne avec un motif de tirets optionnel et des marqueurs de points ; **TsgcChartSeries_FastLine** est une variante plus fine destinée aux données en direct volumineuses ; **TsgcChartSeries_StepLine** descend de TsgcChartSeries_Line et dessine une ligne en escalier au lieu de segments droits.

TsgcChartSeries_XY

TsgcChartSeries_XY possède un **Buffer** de points X/Y et implémente les méthodes de données que partagent chaque série ligne, aire, ligne en escalier, barre et nuage de points. Elle n'est pas abstraite, mais elle est normalement utilisée via l'une de ses classes descendantes plutôt que seule.

Propriété	Type	Description
Buffer	TsgcChartBuffer (lecture seule)	Le tampon circulaire sous-jacent. Voir TsgcChartBuffer and TsgcChartOHLCBuffer.
Capacity	Integer	Nombre maximal de points que le tampon conserve. Par défaut 100000.
BufferMode	TsgcChartBufferMode	`cbmOverwrite` (par défaut) supprime le point le plus ancien une fois le tampon plein ; `cbmGrow` double la capacité à la place.

Méthodes

Méthode	Valeur de retour	Description
AddXY(const aX, aY: Double)		Ajoute un point et appelle **Changed**.
AddY(const aY: Double)		Ajoute un point avec un X égal au X du point le plus récent plus 1 (0 pour un tampon vide) et appelle **Changed**. Utile lorsque X est un index séquentiel plutôt qu'une coordonnée réelle.
Clear		Vide le tampon et appelle **Changed**.

TsgcChartSeries_Line

TsgcChartSeries_Line dessine les points du tampon sous forme de polyligne, avec un motif de tirets optionnel et des marqueurs de points optionnels.

Propriété	Type	Description
LineWidth	Double	Pixels physiques à l'échelle 1. Par défaut 2.
Dash	TsgcChartDash-Style	`cdsSolid`, `cdsDash`, `cdsDot` ou `cdsDashDot`. Par défaut `cdsSolid`.
Markers	TsgcChartSeries-Markers	Masqués par défaut (**Visible** vaut False). Voir TsgcChartSeriesMarkers ci-dessous.

TsgcChartSeries_FastLine

TsgcChartSeries_FastLine est une ligne fine destinée aux données en direct volumineuses : elle n'a ni marqueurs ni motif de tirets, et ses segments se rejoignent par une jonction biseautée plutôt que la jonction arrondie qu'utilise TsgcChartSeries_Line, ce qui la rend économe à redessiner à une fréquence de mise à jour élevée.

Propriété	Type	Description
LineWidth	Double	Pixels physiques à l'échelle 1. Par défaut 1.

TsgcChartSeries_StepLine

TsgcChartSeries_StepLine descend de TsgcChartSeries_Line, elle possède donc aussi **LineWidth**, **Dash** et **Markers**, mais elle relie les points consécutifs par un segment horizontal et un segment vertical au lieu d'une diagonale droite.

Propriété	Type	Description
StepMode	TsgcChartStepMode	`csmBefore`, `csmAfter` ou `csmCenter`. Par défaut `csmAfter`.

Avec **csmAfter**, une valeur est maintenue jusqu'à ce que le point suivant soit atteint, si bien que le palier se produit juste au niveau de chaque point. Avec **csmBefore**, la valeur est atteinte au point précédent, si bien que le palier se produit juste avant celui-ci. Avec **csmCenter**, le palier se produit à mi-chemin entre les deux points.

TsgcChartSeriesMarkers

TsgcChartSeriesMarkers est le petit objet persistant exposé par la propriété **Markers** de TsgcChartSeries_Line et de TsgcChartSeries_Scatter. Il dessine une forme centrée sur chaque point visible.

Propriété	Type	Description
Visible	Boolean	Indique si les marqueurs sont dessinés. La valeur par défaut dépend de la série propriétaire : False pour TsgcChartSeries_Line et ses classes descendantes, True pour TsgcChartSeries_Scatter.
Shape	TsgcChartMarkerShape	`cmsSquare`, `cmsCircle`, `cmsTriangle` ou `cmsDiamond`. Par défaut `cmsCircle`.
Size	Integer	Diamètre en pixels physiques à l'échelle 1. Par défaut 6.

Exemple : séries ligne, ligne rapide et ligne en escalier sur un même graphique

```
uses
  SysUtils, Math,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Axis, sgcCharts_Types,
  sgcUICore_Canvas, sgcUICore_Canvas_Raster;

var
  vChart: TsgcChart;
  vLine: TsgcChartSeries_Line;
  vFast: TsgcChartSeries_FastLine;
  vStep: TsgcChartSeries_StepLine;
  vRaster: TsgcUICanvas_Raster;
  i: Integer;
begin
  vChart := TsgcChart.Create(nil);
  try
    vChart.Axes.Add.Name := 'x';
    vChart.Axes[0].Position := capBottom;
    vChart.Axes.Add.Name := 'y';
    vChart.Axes[1].Position := capLeft;

    vLine := TsgcChartSeries_Line.Create(nil);
    vFast := TsgcChartSeries_FastLine.Create(nil);
    vStep := TsgcChartSeries_StepLine.Create(nil);
    try
      vLine.Chart := vChart;
      vLine.AxisX := 'x';
      vLine.AxisY := 'y';
      vLine.Dash := cdsDash;
      vLine.Markers.Visible := True;
      vLine.Markers.Shape := cmsDiamond;
      for i := 0 to 29 do
        vLine.AddXY(i, Sin(i / 5) * 40 + 60);

      vFast.Chart := vChart;
      vFast.AxisX := 'x';
      vFast.AxisY := 'y';
      vFast.LineWidth := 3;
      for i := 0 to 29 do
        vFast.AddXY(i, Cos(i / 5) * 20 + 20);

      vStep.Chart := vChart;
      vStep.AxisX := 'x';
      vStep.AxisY := 'y';
      vStep.StepMode := csmCenter;
      for i := 0 to 9 do
        vStep.AddXY(i * 3, (i mod 2) * 30);

      Writeln('line dash: ', Ord(vLine.Dash));
      Writeln('step mode: ', Ord(vStep.StepMode));

      vRaster := TsgcUICanvas_Raster.Create(600, 300);
      try
        vChart.Render(vRaster, sgcUIRect(0, 0, 600, 300));
        vRaster.SaveToPNGFile('line.png');
      finally
        vRaster.Free;
      end;
    finally
      vLine.Free;
      vFast.Free;
      vStep.Free;
    end;
  finally
```

```
    vChart.Free;
  end;
end.
```

Voir aussi

- TsgcChart
- TsgcChartSeries
- TsgcChartSeries_Area
- TsgcChartSeries_Scatter

TsgcChartSeries_Area

Unité : sgcCharts_Series
TsgcChartSeries_Area descend de TsgcChartSeries_Line, elle dessine donc la même polyligne (avec **LineWidth**, **Dash** et **Markers**) et remplit en plus la zone entre cette ligne et une valeur de référence.

Propriétés

Propriété	Type	Description
Baseline	Double	Le remplissage va de la ligne vers le bas (ou vers le haut, si la ligne se trouve en dessous) jusqu'à cette valeur. Par défaut 0.
FillOpacity	Double	0..1. Par défaut 0.35.
Gradient	Boolean	False (par défaut) remplit avec une couleur unie à **FillOpacity**. True remplit avec un dégradé vertical qui va de la couleur de remplissage à **FillOpacity** jusqu'à totalement transparent à **Baseline**.

Exemple : une série de type aire avec une ligne de base surélevée et un remplissage en dégradé

```
uses
  SysUtils, Math,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Axis, sgcCharts_Types,
  sgcUICore_Canvas, sgcUICore_Canvas_Raster;

var
  vChart: TsgcChart;
  vArea: TsgcChartSeries_Area;
  vRaster: TsgcUICanvas_Raster;
  i: Integer;
begin
  vChart := TsgcChart.Create(nil);
  try
    vChart.Axes.Add.Name := 'x';
    vChart.Axes[0].Position := capBottom;
    vChart.Axes.Add.Name := 'y';
    vChart.Axes[1].Position := capLeft;

    vArea := TsgcChartSeries_Area.Create(nil);
    try
      vArea.Chart := vChart;
      vArea.AxisX := 'x';
      vArea.AxisY := 'y';
      vArea.Baseline := 10;
      vArea.FillOpacity := 0.5;
      vArea.Gradient := True;

      for i := 0 to 59 do
        vArea.AddXY(i, Sin(i / 8) * 25 + 40);

      Writeln('baseline: ', vArea.Baseline: 0: 1);
      Writeln('fill opacity: ', vArea.FillOpacity: 0: 2);

      vRaster := TsgcUICanvas_Raster.Create(600, 300);
```

```
      try
        vChart.Render(vRaster, sgcUIRect(0, 0, 600, 300));
        vRaster.SaveToPNGFile('area.png');
      finally
        vRaster.Free;
      end;
    finally
      vArea.Free;
    end;
  finally
    vChart.Free;
  end;
end.
```

Voir aussi

- TsgcChart
- TsgcChartSeries
- TsgcChartSeries_Line, TsgcChartSeries_FastLine, TsgcChartSeries_StepLine

TsgcChartSeries_Bar

Unit: sgcCharts_Series
TsgcChartSeries_Bar descend de **TsgcChartSeries_XY** (voir TsgcChartSeries_XY pour **Buffer**, **AddXY**, **AddY**, **Clear**, **Capacity** et **BufferMode**) et dessine une barre par point.

Propriétés

Propriété	Type	Description
Decimation	TsgcChartDecimation	Par défaut `dmNone` pour une série de type barre, contrairement à la valeur par défaut `dmMinMax` de la classe de base TsgcChartSeries : les barres sont censées être affichées à raison d'une par colonne de pixels ou plus, sans être décimées. Voir Décimation.
Orientation	TsgcChartBarOrientation	`cboVertical` (par défaut) ou `cboHorizontal`. Une série de barres horizontale associe son X (position) à l'axe vertical et son Y (valeur) à l'axe horizontal, de sorte que les barres s'étendent latéralement plutôt que vers le haut.
BarWidthPercent	Integer	Part du créneau X occupée par la barre, ou par le groupe de barres, 1..100. Par défaut 70.
Grouping	TsgcChartBarGrouping	`cbgNone` (par défaut), `cbgGrouped` ou `cbgStacked`. Voir ci-dessous.

Groupement et empilement

Avec **cbgGrouped**, les séries de barres visibles qui partagent les mêmes axes, la même orientation et le même groupement sont placées côte à côte dans chaque créneau X. Avec **cbgStacked**, ces séries sont empilées à la place, dans l'ordre des séries : les valeurs positives sont empilées du côté positif et les valeurs négatives du côté négatif, de sorte qu'une série ayant une valeur négative à un X donné ne pousse pas la pile positive vers le haut. **cbgNone** dessine chaque série de barres séparément, en la superposant aux autres au même X.

Exemple : deux séries de barres groupées

```
uses
  SysUtils,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Axis, sgcCharts_Types,
  sgcUICore_Canvas, sgcUICore_Canvas_Raster;

var
  vChart: TsgcChart;
  vBar1, vBar2: TsgcChartSeries_Bar;
  vRaster: TsgcUICanvas_Raster;
  i: Integer;
begin
  vChart := TsgcChart.Create(nil);
  try
    vChart.Axes.Add.Name := 'x';
    vChart.Axes[0].Position := capBottom;
    vChart.Axes.Add.Name := 'y';
    vChart.Axes[1].Position := capLeft;

    vBar1 := TsgcChartSeries_Bar.Create(nil);
```

```
    vBar2 := TsgcChartSeries_Bar.Create(nil);
    try
      vBar1.Chart := vChart;
      vBar1.AxisX := 'x';
      vBar1.AxisY := 'y';
      vBar1.Title := 'Budget';
      vBar1.Orientation := cboVertical;
      vBar1.Grouping := cbgGrouped;
      vBar1.BarWidthPercent := 80;

      vBar2.Chart := vChart;
      vBar2.AxisX := 'x';
      vBar2.AxisY := 'y';
      vBar2.Title := 'Actual';
      vBar2.Orientation := cboVertical;
      vBar2.Grouping := cbgGrouped;
      vBar2.BarWidthPercent := 80;

      for i := 0 to 5 do
      begin
        vBar1.AddXY(i, 40 + i * 5);
        vBar2.AddXY(i, 35 + i * 6);
      end;

      Writeln('bar1 orientation: ', Ord(vBar1.Orientation));
      Writeln('bar grouping: ', Ord(vBar1.Grouping));

      vRaster := TsgcUICanvas_Raster.Create(600, 300);
      try
        vChart.Render(vRaster, sgcUIRect(0, 0, 600, 300));
        vRaster.SaveToPNGFile('bars.png');
      finally
        vRaster.Free;
      end;
    finally
      vBar1.Free;
      vBar2.Free;
    end;
  finally
    vChart.Free;
  end;
end.
```

Voir aussi

- TsgcChart
- TsgcChartSeries
- Décimation

TsgcChartSeries_Scatter

Unit: sgcCharts_Series
TsgcChartSeries_Scatter descend de **TsgcChartSeries_XY** (voir TsgcChartSeries_XY pour **Buffer**, **AddXY**, **AddY**, **Clear**, **Capacity** et **BufferMode**) et dessine ses points uniquement sous forme de marqueurs, sans ligne de liaison.

Propriétés

Propriété	Type	Description
Decimation	TsgcChartDecimation	Par défaut `dmNone` pour une série de type nuage de points, contrairement à la valeur par défaut `dmMinMax` de la classe de base TsgcChartSeries : un nuage de points n'a pas de ligne à garder visuellement exacte, donc chaque point est conservé. Voir Décimation.
Markers	TsgcChartSeriesMarkers	Visibles par défaut (contrairement aux Markers des séries ligne). Voir TsgcChartSeriesMarkers pour **Shape** et **Size** ; mettre **Visible** à False masque les points.

Exemple : une série de type nuage de points avec des marqueurs triangulaires

```
uses
  SysUtils, Math,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Axis, sgcCharts_Types,
  sgcUICore_Canvas, sgcUICore_Canvas_Raster;

var
  vChart: TsgcChart;
  vScatter: TsgcChartSeries_Scatter;
  vRaster: TsgcUICanvas_Raster;
  i: Integer;
begin
  vChart := TsgcChart.Create(nil);
  try
    vChart.Axes.Add.Name := 'x';
    vChart.Axes[0].Position := capBottom;
    vChart.Axes.Add.Name := 'y';
    vChart.Axes[1].Position := capLeft;

    vScatter := TsgcChartSeries_Scatter.Create(nil);
    try
      vScatter.Chart := vChart;
      vScatter.AxisX := 'x';
      vScatter.AxisY := 'y';
      vScatter.Markers.Shape := cmsTriangle;
      vScatter.Markers.Size := 8;

      for i := 0 to 49 do
        vScatter.AddXY(i, Sin(i / 3) * 30 + (i mod 7));

      Writeln('decimation: ', Ord(vScatter.Decimation));
      Writeln('markers visible: ', vScatter.Markers.Visible);

      vRaster := TsgcUICanvas_Raster.Create(600, 300);
      try
```

```
        vChart.Render(vRaster, sgcUIRect(0, 0, 600, 300));
        vRaster.SaveToPNGFile('scatter.png');
      finally
        vRaster.Free;
      end;
    finally
      vScatter.Free;
    end;
  finally
    vChart.Free;
  end;
end.
```

Voir aussi

- TsgcChart
- TsgcChartSeries
- TsgcChartSeries_Line, TsgcChartSeries_FastLine, TsgcChartSeries_StepLine
- Décimation

TsgcChartSeries_Candlestick, TsgcChartSeries_OHLC

Unit: sgcCharts_Series
Ces deux séries descendent de **TsgcChartSeries_OHLCBase**, la base commune de toute série open/high/low/close. TsgcChartSeries_OHLCBase possède un tampon OHLC au lieu du tampon X/Y utilisé par les séries de la famille ligne, et implémente les méthodes qui l'alimentent ; **TsgcChartSeries_Candlestick** dessine chaque barre comme un corps de chandelier plein ou creux avec des mèches, et **TsgcChartSeries_OHLC** dessine chaque barre comme une ligne verticale avec des repères d'ouverture et de clôture.

TsgcChartSeries_OHLCBase

Propriété	Type	Description
OHLC-Buffer	TsgcChartOHLCBuffer (lecture seule)	Le tampon circulaire sous-jacent des chandeliers. Voir TsgcChartBuffer and TsgcChartOHLCBuffer.
Capacity	Integer	Nombre maximal de chandeliers que le tampon conserve. Par défaut 100000.
Buffer-Mode	TsgcChartBufferMode	`cbmOverwrite` (par défaut) supprime le chandelier le plus ancien une fois le tampon plein ; `cbmGrow` double la capacité à la place.
UpColor	TsgcUIColor	Couleur utilisée quand la clôture d'un chandelier est supérieure ou égale à son ouverture. Par défaut un vert, $FF1BAF7A.
DownColor	TsgcUIColor	Couleur utilisée quand la clôture d'un chandelier est inférieure à son ouverture. Par défaut un rouge, $FFE34948.

Méthodes

Méthode	Valeur de retour	Description
AddOHLC(const aX, aOpen, aHigh, aLow, aClose: Double; const aVolume: Double = 0)		Ajoute un chandelier et appelle **Changed**.
UpdateLastOHLC(const aX, aOpen, aHigh, aLow, aClose: Double; const aVolume: Double = 0)		Remplace le chandelier le plus récent lorsque son X est égal à **aX**, sinon en ajoute un nouveau, et appelle **Changed**. Destiné à un flux en direct qui continue de mettre à jour le chandelier en cours de formation jusqu'à la fin de sa période.
Clear		Vide le tampon et appelle **Changed**.

TsgcChartSeries_Candlestick

TsgcChartSeries_Candlestick dessine chaque chandelier comme un corps entre l'ouverture et la clôture, coloré avec **UpColor** ou **DownColor**, plus une mèche du plus haut au plus bas.

Propriété	Type	Description
WickColor	TsgcUI-Color	Par défaut 0, ce qui signifie que la mèche prend la couleur du corps du chandelier. Une valeur non nulle la remplace pour chaque chandelier.
BodyWidth-Percent	Integer	Part du créneau du chandelier occupée par le corps, 1..100. Par défaut 70.
HollowUp	Boolean	Par défaut False. True dessine les chandeliers haussiers (clôture >= ouverture) comme un contour dans **UpColor** au lieu d'un corps plein.

TsgcChartSeries_OHLC

TsgcChartSeries_OHLC dessine chaque barre comme une ligne verticale du plus haut au plus bas, avec un court repère à gauche pour l'ouverture et un court repère à droite pour la clôture, coloré avec **UpColor** ou **DownColor**.

Propriété	Type	Description
TickWidth-Percent	Integer	Part du créneau du chandelier occupée par les repères d'ouverture et de clôture, 1..100. Par défaut 70.
LineWidth	Double	Pixels physiques à l'échelle 1. Par défaut 1.

Exemple : une série de chandeliers et une série OHLC

```
uses
  SysUtils,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Axis, sgcCharts_Types,
  sgcUICore_Canvas, sgcUICore_Canvas_Raster;

var
  vChart: TsgcChart;
  vCandle: TsgcChartSeries_Candlestick;
  vOHLC: TsgcChartSeries_OHLC;
  vRaster: TsgcUICanvas_Raster;
  i: Integer;
  vOpen, vClose: Double;
begin
  vChart := TsgcChart.Create(nil);
  try
    vChart.Axes.Add.Name := 'x';
    vChart.Axes[0].Position := capBottom;
    vChart.Axes.Add.Name := 'y';
    vChart.Axes[1].Position := capLeft;

    vCandle := TsgcChartSeries_Candlestick.Create(nil);
    vOHLC := TsgcChartSeries_OHLC.Create(nil);
    try
      vCandle.Chart := vChart;
      vCandle.AxisX := 'x';
      vCandle.AxisY := 'y';
      vCandle.WickColor := $FF000000;
      vCandle.BodyWidthPercent := 60;
      vCandle.HollowUp := True;

      for i := 0 to 18 do
      begin
        vOpen := 100 + i;
        vClose := vOpen + (i mod 3) - 1;
        vCandle.AddOHLC(i, vOpen, vOpen + 3, vClose - 3, vClose);
      end;
```

```
      vCandle.UpdateLastOHLC(18, 118, 125, 110, 121);

      vOHLC.Chart := vChart;
      vOHLC.AxisX := 'x';
      vOHLC.AxisY := 'y';
      vOHLC.TickWidthPercent := 50;
      vOHLC.LineWidth := 2;
      for i := 0 to 18 do
        vOHLC.AddOHLC(i, 90 - i, 95 - i, 80 - i, 85 - i);

      Writeln('candlestick index: ', vCandle.Index);
      Writeln('ohlc index: ', vOHLC.Index);

      vRaster := TsgcUICanvas_Raster.Create(600, 300);
      try
        vChart.Render(vRaster, sgcUIRect(0, 0, 600, 300));
        vRaster.SaveToPNGFile('candles.png');
      finally
        vRaster.Free;
      end;
    finally
      vCandle.Free;
      vOHLC.Free;
    end;
  finally
    vChart.Free;
  end;
end.
```

Voir aussi

- TsgcChart
- TsgcChartSeries
- Panneaux, pour scinder une série de chandeliers dans son propre panneau sous l'axe des prix

TsgcChartSeries_Pie

Unit: sgcCharts_Series_Pie
TsgcChartSeries_Pie dessine un camembert ou, avec **InnerRadiusPercent** supérieur à 0, un anneau (donut). Elle descend de **TsgcChartSeries_Category** (unité `sgcCharts_Series_Category`), la base de toute série qui contient des valeurs nommées : un élément par secteur, conservé dans la collection **Slices**, qui sert aussi d'éditeur en mode conception et de stockage thread-safe pour les données en direct. Un camembert est une série à cellule : il obtient sa propre cellule de la zone de tracé et ne dessine aucun axe, voir Zones de tracé.

Propriétés

Propriété	Type	Description
Slices	TsgcChartCategoryItems	Les secteurs, un **TsgcChartPieSlice** chacun (**Text**, **Value**, **Color** et **Exploded**). Les couleurs des secteurs suivent l'index de la collection, de sorte qu'un secteur garde sa couleur avec des données en direct.
InnerRadiusPercent	Double	0 (par défaut) dessine un camembert ; jusqu'à 90, dessine un anneau dont le trou représente ce pourcentage du rayon.
StartAngle	Double	Angle du premier bord de secteur en degrés, 0 (par défaut) correspond à 12 heures.
Clockwise	Boolean	Par défaut True.
Labels	TsgcChartSliceLabels	`cslNone` , `cslInside` , `cslOutside` (par défaut) ou `cslAuto` . Voir Étiquettes ci-dessous.
LabelFormat	string	Par défaut `{text} ({percent})` . Les jetons s'écrivent sous la forme d'un nom entre accolades : `{text}` , `{value}` et `{percent}` .
ValueFormat	string	Masque FormatFloat du jeton de valeur ; vide (par défaut), il affiche jusqu'à 2 décimales.
PercentDecimals	Integer	Nombre de décimales du jeton de pourcentage. Par défaut 1.
LabelStyle	TsgcUITextStyle	Police des étiquettes. **Color** 0 utilise la couleur de texte du thème à l'extérieur du camembert, et un texte blanc ou foncé, choisi selon la couleur du secteur, à l'intérieur.
LeaderLineColor	TsgcUIColor	Couleur des lignes de repère des étiquettes extérieures ; 0 (par défaut) utilise la couleur du secteur.
BorderColor	TsgcUIColor	Couleur des bordures des secteurs ; 0 (par défaut) utilise la couleur de fond du graphique.
BorderWidth	Double	Épaisseur des bordures des secteurs. Par défaut 1.
ExplodeOffset	Integer	Distance dont un secteur éclaté est tiré vers l'extérieur, en pixels physiques à l'échelle 1. Par défaut 10.
ExplodeOnClick	Boolean	Par défaut False. À True, un clic sur un secteur bascule son indicateur **Exploded** (**OthersExploded** pour le secteur Others).

Others-Threshold	Double	Un pourcentage, 0 (par défaut) désactive la fonction. Les secteurs inférieurs à ce seuil sont fusionnés en un seul secteur Others, dessiné en dernier, lorsqu'au moins deux sont concernés.
OthersText	string	Texte du secteur Others. Par défaut `Others`.
OthersColor	TsgcUIColor	Couleur du secteur Others ; 0 (par défaut) utilise la couleur de texte atténuée du thème.
OthersExploded	Boolean (public)	Le secteur Others est tiré hors du camembert.
Total	Double (lecture seule)	Somme des valeurs dessinées lors du dernier rendu.

Méthodes

Méthode	Valeur de retour	Description
SliceColor(aIndex: Integer)	TsgcUIColor	La couleur avec laquelle le secteur **aIndex** (un index de collection, ou **CS_CHART_OTHERS_INDEX**) est dessiné : son **Color**, sinon la couleur de palette du graphique correspondant à **aIndex**. Le secteur Others utilise **OthersColor**, sinon la couleur de texte atténuée du thème.

Méthodes des séries de catégories

Les séries camembert, radar, entonnoir et treemap partagent ces méthodes de **TsgcChartSeries_Category**. Chacune peut être appelée depuis n'importe quel thread : le stockage possède son propre verrou, et une écriture qui ne modifie pas une valeur ne déclenche pas de redessin.

Méthode	Valeur de retour	Description
SetValue(const aText: string; aValue: Double)	Integer	Définit la valeur du premier élément portant ce texte (sensible à la casse), ou ajoute un nouvel élément. Renvoie l'index de l'élément. La dernière valeur d'un texte l'emporte ; les valeurs ne sont jamais cumulées.
SetValueAt(aIndex: Integer; aValue: Double)		Définit la valeur de l'élément à **aIndex**, en ajoutant des éléments au texte vide et à la valeur 0 jusqu'à **aIndex** lorsque la collection est plus courte.
AddItem(const aText: string; aValue: Double; aColor: TsgcUIColor = 0)	Integer	Ajoute un élément, y compris lorsque le texte est déjà utilisé. Renvoie son index.
DeleteItem(aIndex: Integer)		Supprime un élément.
ClearItems		Supprime tous les éléments.
IndexOfText(const aText: string)	Integer	Index du premier élément portant ce texte (sensible à la casse), -1 s'il n'y en a aucun.
ItemCount	Integer	Nombre d'éléments.

BeginUpdate / EndUpdate		Les écritures effectuées entre les deux ne redessinent le graphique qu'une fois, au dernier **EndUpdate**.
BeginReload / EndReload		Marquage et balayage : le dernier **EndReload** supprime les éléments qui n'ont pas été écrits depuis le premier **BeginReload**, de sorte qu'un rechargement conserve les couleurs et les indicateurs des éléments restants. C'est aussi une mise à jour, avec un seul redessin à la fin.
GetSnapshot(var aSnap: TsgcChartCategorySnapshot)		Copie les éléments dans **aSnap** sous verrou lorsqu'ils ont changé depuis la dernière copie.

Les jetons d'étiquette de toutes les séries de catégories sont développés par **sgcChartFormatLabel**, qui prend le pourcentage et la part précédente en unités de pourcentage (25 signifie 25 %) et ajoute le signe `%`.

Étiquettes

cslInside centre chaque étiquette dans son secteur. **cslOutside** place les étiquettes en deux colonnes, à gauche et à droite du camembert, reliées à leurs secteurs par des lignes de repère ; les étiquettes d'une colonne sont espacées de façon à ne jamais se chevaucher. **cslAuto** dessine une étiquette à l'intérieur lorsqu'elle tient dans la corde et l'épaisseur de l'anneau, et à l'extérieur sinon. Une étiquette plus large que sa place est raccourcie pour y tenir.

Test de survol, infobulle et légende

Un survol ou un clic sur un secteur renseigne **Index** et **SubIndex** avec l'index de collection (**CS_CHART_OTHERS_INDEX**, -2, pour le secteur Others), **YValue** avec la valeur, **Percent** avec la part et **Text** avec le texte du secteur. L'infobulle affiche `{text}: {value} ({percent})`. La légende affiche une entrée par secteur dessiné, Others en dernier. Voir TsgcChartInteraction.

Exemple : un anneau avec Others et un secteur éclaté

```
uses
  SysUtils,
  sgcUICore_Theme, sgcUICore_Canvas, sgcUICore_Canvas_Raster,
  sgcCharts_Chart, sgcCharts_Series_Category, sgcCharts_Series_Pie;

var
  vChart: TsgcChart;
  vDonut: TsgcChartSeries_Pie;
  vRaster: TsgcUICanvas_Raster;
  i: Integer;
begin
  vChart := TsgcChart.Create(nil);
  try
    vDonut := TsgcChartSeries_Pie.Create(nil);
    try
      vDonut.Chart := vChart;
      vDonut.Title := 'Revenue by region';
      vDonut.InnerRadiusPercent := 55; // a donut ring
      vDonut.Labels := cslAuto;
      vDonut.LabelFormat := '{text}: {value} ({percent})';
      vDonut.ValueFormat := '#,##0';
      vDonut.OthersThreshold := 5; // slices below 5% go into Others
      vDonut.ExplodeOnClick := True;

      // one repaint for the whole batch
      vDonut.BeginUpdate;
      try
        vDonut.SetValue('North', 420);
        vDonut.SetValue('South', 310);
```

```
        vDonut.SetValue('East', 275);
        vDonut.SetValue('West', 190);
        vDonut.SetValue('Islands', 12);
        vDonut.SetValue('Export', 9);
      finally
        vDonut.EndUpdate;
      end;
      // a live update: the latest value of a text wins
      vDonut.SetValue('South', 335);
      // pull the first slice out of the ring
      TsgcChartPieSlice(vDonut.Slices[0]).Exploded := True;

      vRaster := TsgcUICanvas_Raster.Create(640, 400);
      try
        vChart.Render(vRaster, sgcUIRect(0, 0, 640, 400));
        vRaster.SaveToPNGFile('donut.png');
      finally
        vRaster.Free;
      end;

      Writeln('total: ', FormatFloat('0', vDonut.Total));
      for i := 0 to vDonut.ItemCount - 1 do
        Writeln(vDonut.Slices[i].Text, ' $', IntToHex(vDonut.SliceColor(i), 8));
    finally
      vDonut.Free;
    end;
  finally
    vChart.Free;
  end;
end.
```

Voir aussi

- TsgcChartSeries
- Zones de tracé
- TsgcChartInteraction
- Titre, légende et marges

TsgcChartSeries_Radar

Unit: sgcCharts_Series_Radar
TsgcChartSeries_Radar dessine des valeurs sur des rayons autour d'un centre, sous la forme d'une ligne polygonale ou d'un polygone rempli. C'est une série de catégories (voir les méthodes des séries de catégories de la rubrique du camembert) : chaque élément de **Points** est un rayon. Un radar donne à sa zone de tracé le mode polaire, et tous les radars d'une zone partagent une même grille polaire dont les rayons sont l'union des textes de leurs éléments, dans l'ordre de première utilisation. L'échelle radiale est **YAxis** de la série : **Automatic**, **Minimum**, **Maximum**, le nombre de graduations et le format des étiquettes de cet axe s'appliquent ; les valeurs hors de sa plage sont ramenées au bord.

Propriétés

Propriété	Type	Description
Points	TsgcChartCategoryItems	Un élément par rayon : **Text** nomme le rayon et **Value** est la valeur radiale.
Filled	Boolean	Par défaut False. True remplit le polygone.
FillOpacity	Byte	Alpha du remplissage. Par défaut $40.
LineWidth	Double	Épaisseur du contour en pixels physiques à l'échelle 1 ; 0 ne dessine aucun contour.
Markers	TsgcChartSeriesMarkers	Repères des sommets, visibles par défaut.
GridShape	TsgcChartRadarGrid	`crgPolygon` (par défaut) ou anneaux `crgCircle`. Utilisé lorsqu'il s'agit du premier radar dessiné de sa zone.
StartAngle	Double	Angle du premier rayon en degrés, 0 (par défaut) correspond à 12 heures, dans le sens horaire. Utilisé lorsqu'il s'agit du premier radar dessiné de sa zone.
SpokeLabelStyle	TsgcUITextStyle	Police des textes des rayons ; **Color** 0 utilise la couleur de texte du thème. Utilisé lorsqu'il s'agit du premier radar dessiné de sa zone.

Les propriétés **GridVisible** et **GridColor** de l'axe Y contrôlent les anneaux et les rayons, et sa propriété **Visible** affiche ou masque les étiquettes de graduation. Un anneau extérieur est ajouté lorsqu'aucune graduation ne tombe sur le bord.

Géométrie du dernier rendu

Méthode	Valeur de retour	Description
CenterX, CenterY, Radius	Double (lecture seule)	Centre et rayon de la grille polaire.
SpokeCount	Integer (lecture seule)	Nombre de rayons.
SpokeText[Index]	string (lecture seule)	Texte d'un rayon.

VertexPoint(aSpoke: Integer; out aX, aY: Double)	Boolean	Pixel du sommet de cette série sur le rayon **aSpoke**. False (et le centre) pour un rayon manquant, une valeur non finie, ou lorsque la série n'a pas été dessinée.

Test de survol et infobulle

Un survol ou un clic renvoie le sommet le plus proche des radars de la zone dans le rayon de clic, à partir de la géométrie du dernier rendu. **Index** est le rayon, **SubIndex** l'élément, **YValue** la valeur, **Text** le texte du rayon et **Distance** la distance en pixels ; **Percent** et **ZValue** valent NaN. L'infobulle affiche `Product A, Speed: 80`.

Exemple : deux produits sur six critères

```
uses
  SysUtils,
  sgcUICore_Theme, sgcUICore_Canvas, sgcUICore_Canvas_Raster,
  sgcCharts_Chart, sgcCharts_Axis, sgcCharts_Series_Category,
  sgcCharts_Series_Radar;

const
  CS_CRITERIA: array [0 .. 5] of string = ('Speed', 'Price', 'Support',
    'Security', 'Features', 'Design');
  CS_PRODUCT_A: array [0 .. 5] of Double = (80, 65, 90, 70, 85, 60);
  CS_PRODUCT_B: array [0 .. 5] of Double = (60, 90, 70, 85, 55, 75);

var
  vChart: TsgcChart;
  vAxis: TsgcChartAxis;
  vA, vB: TsgcChartSeries_Radar;
  vRaster: TsgcUICanvas_Raster;
  vX, vY: Double;
  i: Integer;
begin
  vChart := TsgcChart.Create(nil);
  try
    // the radial scale of a radar is its Y axis
    vAxis := vChart.Axes.Find('Y');
    vAxis.Automatic := False;
    vAxis.Minimum := 0;
    vAxis.Maximum := 100;

    vA := TsgcChartSeries_Radar.Create(nil);
    vB := TsgcChartSeries_Radar.Create(nil);
    try
      vA.Chart := vChart;
      vA.Title := 'Product A';
      vA.Filled := True;
      vA.FillOpacity := $50;
      // the first radar of the area decides the grid
      vA.GridShape := crgCircle;

      vB.Chart := vChart;
      vB.Title := 'Product B';
      vB.LineWidth := 2;

      for i := 0 to High(CS_CRITERIA) do
      begin
        vA.SetValue(CS_CRITERIA[i], CS_PRODUCT_A[i]);
        vB.SetValue(CS_CRITERIA[i], CS_PRODUCT_B[i]);
      end;

      vRaster := TsgcUICanvas_Raster.Create(500, 500);
      try
```

```
        vChart.Render(vRaster, sgcUIRect(0, 0, 500, 500));
        vRaster.SaveToPNGFile('radar.png');
      finally
        vRaster.Free;
      end;

      // the geometry of the last render
      Writeln('spokes: ', vA.SpokeCount, ', radius: ',
        FormatFloat('0', vA.Radius));
      for i := 0 to vA.SpokeCount - 1 do
        if vA.VertexPoint(i, vX, vY) then
          Writeln(vA.SpokeText[i], ' at ', FormatFloat('0', vX), ', ',
            FormatFloat('0', vY));
    finally
      vB.Free;
      vA.Free;
    end;
  finally
    vChart.Free;
  end;
end.
```

Voir aussi

- TsgcChartSeries
- Zones de tracé
- TsgcChartInteraction
- Axes

TsgcChartSeries_Heatmap

Unit: sgcCharts_Series_Heatmap
TsgcChartSeries_Heatmap dessine une matrice de valeurs sous forme de cellules colorées sur les axes X et Y. C'est une série XY (un descendant direct de **TsgcChartSeries**) : elle partage donc son tracé avec des séries en courbes, en barres ou en nuage de points, et utilise tels quels les axes numériques, de date et heure ou de catégories. Les valeurs sont conservées dans un stockage thread-safe de lignes, **Data** (**TsgcChartHeatmapData**) : avec **Orientation** `chmColumns`, une ligne est une colonne placée à une clé X et ses cellules s'étendent le long de Y ; avec `chmRows`, une ligne est une rangée placée à une clé Y. Avec **Capacity** supérieur à 0, le stockage ne conserve que les lignes les plus récentes, l'idée du tampon circulaire appliquée à une matrice, ce qui transforme une carte de chaleur en spectrogramme défilant. Les couleurs proviennent d'une échelle de couleurs.

Propriétés

Propriété	Type	Description
Orientation	TsgcChartHeatmapOrientation	`chmColumns` (par défaut) ou `chmRows`.
Capacity	Integer	0 (par défaut) laisse le stockage grandir ; au-dessus de 0, ne conserve que les lignes les plus récentes.
CellOrigin	Double	Centre de la cellule 0 sur l'axe des cellules. Par défaut 0.
CellStep	Double	Distance entre les centres des cellules, supérieure à 0. Par défaut 1.
LineStep	Double	Étendue d'une ligne sur l'axe des lignes ; 0 (par défaut) va jusqu'à mi-chemin des lignes voisines.
ColorScale	TsgcChartColorScale	Couleurs des cellules. Un **MaxColor** de 0 prend la **Color** de la série, sinon la première couleur de la palette ; un **MinColor** de 0 prend le fond du tracé. Voir Échelle de couleurs.
ShowValues	Boolean	Par défaut False. True écrit la valeur dans chaque cellule peinte du chemin vectoriel.
ValueFormat	string	Masque FormatFloat des textes de valeur ; vide (par défaut), il affiche jusqu'à 2 décimales.
ValueStyle	TsgcUITextStyle	Police des textes de valeur ; **Color** 0 choisit un texte blanc ou foncé selon la luminance de la cellule.
CellBorderColor	TsgcUIColor	Bordure dessinée à l'intérieur de chaque cellule peinte du chemin vectoriel ; 0 (par défaut) n'en dessine aucune.
Data	TsgcChartHeatmapData (lecture seule)	Le stockage des lignes. Les écritures faites directement sur **Data** ne redessinent pas le graphique ; les méthodes de la série ci-dessous, si.
Cells[aLine, aCell]	Double	Valeur d'une cellule ; NaN hors limites. Une écriture hors limites est ignorée. NaN est une cellule nulle, jamais peinte et jamais touchée.
ImagePath	Boolean (lecture seule)	True lorsque le dernier rendu a rempli une seule image, False lorsqu'il a dessiné des rectangles.
DrawnCells	Integer (lecture seule)	Cellules visibles lors du dernier rendu.

Méthodes

Méthode	Valeur de retour	Description
BeginUpdate / EndUpdate		Les écritures effectuées entre les deux ne redessinent le graphique qu'une fois. Depuis n'importe quel thread.
Clear		Supprime toutes les lignes.
SetSize(aLines, aCells: Integer)		Dimensionne la matrice à **aLines** lignes de **aCells** cellules.
AppendLine(const aValues: array of Double)		Ajoute une ligne identifiée par le compteur de lignes courant du stockage.
AppendLine(const aKey: Double; const aValues: array of Double)		Ajoute une ligne à la clé X (colonnes) ou Y (rangées) **aKey**, par exemple un horodatage. Une ligne plus longue que le nombre de cellules actuel élargit toutes les lignes et les complète avec des cellules nulles.
SetLineLabels(const aLabels: array of string) / SetLineLabels(aLabels: TStrings)		Textes des lignes, affichés par un axe de lignes de catégories.
SetCellLabels(const aLabels: array of string) / SetCellLabels(aLabels: TStrings)		Textes des cellules, affichés par un axe de cellules de catégories.

Axes et dessin

Sur un axe de catégories, une ligne ou une cellule se place à son index logique, quels que soient sa clé ou **CellOrigin**, de sorte qu'un stockage défilant reste aligné avec ses étiquettes, qui parviennent à l'axe par **GetDataCategories**. Chaque bord de cellule est aligné sur un pixel physique entier. La série choisit le chemin de dessin d'après le nombre et la taille des cellules visibles, jamais d'après le canevas : jusqu'à **CS_CHART_HEATMAP_VECTOR_CELLS** (20 000) cellules d'au moins **CS_CHART_HEATMAP_VECTOR_MIN_SIZE** (3) pixels sont dessinées sous forme de rectangles avec bordures et textes de valeur ; les matrices plus grandes sont remplies comme une seule image, sans bordures ni textes. Le résultat est identique sur les canevas raster, SVG, VCL et FMX.

Test de survol, infobulle et légende

Un survol ou un clic renseigne **Area** avec `chaPlot`, **Index** avec la ligne, **SubIndex** avec la cellule, **XValue** et **YValue** avec le centre de la cellule, **ZValue** avec la valeur et **Text** avec `ligne, cellule` (le texte de l'axe lorsqu'une étiquette est vide) ; **Distance** vaut 0 et **Percent** vaut NaN. L'infobulle affiche `Sales, Mar, South: 64`. Lorsqu'une série en courbe est dessinée par-dessus, un point de courbe situé dans le rayon de clic l'emporte sur la cellule. La légende affiche une barre de couleurs avec le minimum et le maximum de l'échelle.

Exemple : une matrice région par mois

```
uses
  SysUtils, Math,
  sgcUICore_Theme, sgcUICore_Canvas, sgcUICore_Canvas_Raster,
  sgcCharts_Types, sgcCharts_Chart, sgcCharts_HeatmapData,
  sgcCharts_Series_Heatmap;
```

```
var
  vChart: TsgcChart;
  vHeat: TsgcChartSeries_Heatmap;
  vRaster: TsgcUICanvas_Raster;
  i: Integer;
begin
  vChart := TsgcChart.Create(nil);
  try
    // category axes: one line per month on X, one cell per region on Y
    vChart.Axes.Find('X').AxisType := catCategory;
    vChart.Axes.Find('Y').AxisType := catCategory;

    vHeat := TsgcChartSeries_Heatmap.Create(nil);
    try
      vHeat.Chart := vChart;
      vHeat.Title := 'Sales';
      vHeat.Orientation := chmColumns;
      vHeat.ColorScale.MinColor := $FFE3F2FD;
      vHeat.ColorScale.MaxColor := $FF0D47A1;
      vHeat.ShowValues := True;
      vHeat.ValueFormat := '0';
      vHeat.CellBorderColor := $FFFFFFFF;

      // one repaint for the whole matrix
      vHeat.BeginUpdate;
      try
        vHeat.SetCellLabels(['North', 'South', 'East', 'West']);
        vHeat.SetLineLabels(['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun']);
        for i := 0 to 5 do
          vHeat.AppendLine([40 + i * 5, 30 + i * 3, 55 - i * 4, 20 + i * 8]);
      finally
        vHeat.EndUpdate;
      end;
      // a null cell is not painted and never hit
      vHeat.Cells[2, 1] := NaN;

      vRaster := TsgcUICanvas_Raster.Create(700, 400);
      try
        vChart.Render(vRaster, sgcUIRect(0, 0, 700, 400));
        vRaster.SaveToPNGFile('heatmap.png');
      finally
        vRaster.Free;
      end;

      Writeln('cells drawn: ', vHeat.DrawnCells, ', one image: ',
        vHeat.ImagePath);
      Writeln('Apr, West: ', FormatFloat('0', vHeat.Cells[3, 3]));
    finally
      vHeat.Free;
    end;
  finally
    vChart.Free;
  end;
end.
```

Voir aussi

- TsgcChartSeries
- Zones de tracé
- TsgcChartInteraction
- Échelle de couleurs
- Axes

TsgcChartSeries_Treemap

Unit: sgcCharts_Series_Treemap
TsgcChartSeries_Treemap dessine une hiérarchie sous forme de rectangles imbriqués dont les aires suivent les valeurs. C'est une série de catégories (voir les méthodes des séries de catégories de la rubrique du camembert) : le **Text** de chaque élément de **Nodes** est un chemin tel que `Europe/Spain/Madrid`, découpé par **PathDelimiter**, et les groupes sont les sommes de leurs enfants. La disposition est en rectangles quasi carrés (squarified) à chaque niveau, avec marges et bandeaux d'en-tête, et conserve l'ordre des éléments de même niveau lors des mises à jour de valeurs en direct. Un clic descend dans un groupe et un fil d'Ariane au-dessus des tuiles permet de remonter. Un treemap est une série à cellule, voir Zones de tracé.

Propriétés

Propriété	Type	Description
Nodes	TsgcChartCategoryItems	Les feuilles : un chemin dans **Text**, la valeur dans **Value**. La **Color** d'un élément de premier niveau remplace sa couleur de palette.
PathDelimiter	Char	Séparateur des segments du chemin. Par défaut `/`.
MaxDepth	Integer	Niveaux dessinés sous la racine courante ; en dessous de 1, aucune limite. Par défaut 2. Les groupes plus profonds sont dessinés repliés, sous la forme d'une seule tuile.
ColorMode	TsgcChartTreemapColorMode	`ctcGroup` (par défaut) : une couleur de palette par groupe de premier niveau, éclaircie (thème clair) ou assombrie (thème sombre) à chaque niveau. `ctcValue` : tuiles colorées au moyen de **ColorScale**, sur la plage des tuiles situées sous la racine.
ColorScale	TsgcChartColorScale	Couleurs de `ctcValue` ; un **MaxColor** de 0 prend la **Color** de la série, sinon la première couleur de la palette. Voir Échelle de couleurs.
Levels	TsgcChartTreemapLevels	Bordure, marge et en-tête par niveau ; **Items[0]** est le niveau 1, les enfants de la racine courante. Voir TsgcChartTreemapLevel ci-dessous.
LabelFormat	string	Par défaut `{text}`. Les jetons s'écrivent sous la forme d'un nom entre accolades : `{text}`, `{value}` et `{percent}` ; text est le dernier segment du chemin et percent est la part de la racine courante.
ValueFormat	string	Masque FormatFloat du jeton de valeur ; vide (par défaut), il affiche jusqu'à 2 décimales.
PercentDecimals	Integer	Par défaut 1.
LabelStyle	TsgcUITextStyle	Police des étiquettes ; **Color** 0 choisit un texte blanc ou foncé selon le remplissage.
StableLayout	Boolean	Par défaut True : une mise à jour de valeur conserve l'ordre des éléments de même niveau, de sorte que les tuiles ne sautent pas avec des données en direct.
DrillOnClick	Boolean	Par défaut True : un clic sur une tuile descend dans son ancêtre de niveau 1, un clic sur un élément du fil d'Ariane remonte jusqu'à lui.

OnDrill	TNotifyEvent	Se déclenche lorsque **DrillPath** change.
DrillPath	string (public)	Chemin de la racine courante ; vide, il désigne la racine de l'arbre. Le définir déclenche **OnDrill**.
Layout	TsgcChart-Treema-pLayout (lec-ture seule)	L'arbre et la géométrie du dernier rendu : nœuds, nœuds dessinés, le fil d'Ariane (**Crumbs**) et sa boîte.

TsgcChartTreemapLevel

Propriété	Type	Description
BorderColor	TsgcUICo-lor	0 (par défaut) utilise la couleur de fond du graphique.
BorderWidth	Double	Pixels physiques à l'échelle 1 ; 0 ne dessine aucune bordure.
Padding	Double	Espace entre un groupe de ce niveau et ses enfants, en pixels physiques à l'échelle 1.
ShowHeader	Boolean	Un bandeau d'en-tête portant le nom du groupe au-dessus des enfants.

Un nouvel élément de niveau prend les valeurs par défaut de son niveau : 2 pixels de marge et un en-tête pour le premier élément, 1 pixel et aucun en-tête pour les autres. Un niveau sans élément utilise une bordure de 1 pixel.

Méthodes

Méthode	Valeur de retour	Description
DrillDown(const aPath: string)		Fait d'un chemin de nœud la racine courante ; un chemin vide désigne la racine de l'arbre.
DrillUp		Fait du parent de la racine courante la nouvelle racine courante.
DrillReset		Fait de la racine de l'arbre la racine courante.
NodeColor(aNode: Inte-ger)	TsgcUIColor	Remplissage du nœud de disposition **aNode** lors du dernier ren-du, 0 s'il n'est pas dessiné.
CrumbRect(aIndex: In-teger)	TsgcUIRect	Boîte de l'élément **aIndex** du fil d'Ariane (dans l'ordre de **Layout.Crumbs**) lors du dernier rendu ; vide s'il n'est pas dessi-né. Le premier élément est la racine, avec le texte `All` .

Étiquettes, test de survol et légende

Une étiquette s'affiche lorsque sa boîte est au moins aussi large que trois caractères et haute d'une ligne ; un texte plus long est raccourci pour y tenir, une tuile plus petite n'affiche aucune étiquette. Un survol ou un clic renvoie la tuile la plus profonde sous le point : **Index** est l'élément (-1 pour un groupe, qui n'a pas d'élément propre), **SubIn-dex** la profondeur, **YValue** la valeur, **Percent** la part de la racine courante et **Text** le chemin. Pour un élément du fil d'Ariane, **Percent** vaut NaN. Avec `ctcGroup`, la légende liste les groupes de premier niveau de tout l'arbre ; avec `ctcValue`, elle affiche une barre de couleurs.

Exemple : un arbre à trois niveaux avec descente

```
uses
  SysUtils,
  sgcUICore_Theme, sgcUICore_Canvas, sgcUICore_Canvas_Raster,
  sgcCharts_Chart, sgcCharts_Series_Category, sgcCharts_Series_Treemap;

var
  vChart: TsgcChart;
  vTree: TsgcChartSeries_Treemap;
  vLevel: TsgcChartTreemapLevel;
  vRaster: TsgcUICanvas_Raster;
begin
  vChart := TsgcChart.Create(nil);
  try
    vTree := TsgcChartSeries_Treemap.Create(nil);
    try
      vTree.Chart := vChart;
      vTree.Title := 'Sales';
      vTree.PathDelimiter := '/';
      vTree.MaxDepth := 2; // continents and countries below the root
      vTree.ColorMode := ctcGroup;
      vTree.LabelFormat := '{text} {percent}';
      // level 1: 3 pixels of padding and a header strip
      vLevel := vTree.Levels.Add;
      vLevel.Padding := 3;
      vLevel.ShowHeader := True;

      vTree.BeginUpdate;
      try
        vTree.SetValue('Europe/Spain/Madrid', 120);
        vTree.SetValue('Europe/Spain/Barcelona', 95);
        vTree.SetValue('Europe/France/Paris', 140);
        vTree.SetValue('Europe/France/Lyon', 45);
        vTree.SetValue('America/USA/New York', 160);
        vTree.SetValue('America/USA/Chicago', 70);
        vTree.SetValue('America/Mexico/Monterrey', 40);
        vTree.SetValue('Asia/Japan/Tokyo', 130);
        vTree.SetValue('Asia/India/Mumbai', 60);
      finally
        vTree.EndUpdate;
      end;

      vRaster := TsgcUICanvas_Raster.Create(800, 500);
      try
        vChart.Render(vRaster, sgcUIRect(0, 0, 800, 500));
        vRaster.SaveToPNGFile('treemap.png');
        // drill into one continent: a breadcrumb shows the path
        vTree.DrillDown('Europe');
        vChart.Render(vRaster, sgcUIRect(0, 0, 800, 500));
        vRaster.SaveToPNGFile('treemap_europe.png');
      finally
        vRaster.Free;
      end;

      Writeln('root: ', vTree.DrillPath);
      vTree.DrillUp;
      Writeln('root after DrillUp: "', vTree.DrillPath, '"');
    finally
      vTree.Free;
    end;
  finally
    vChart.Free;
  end;
end.
```

Voir aussi

- TsgcChartSeries
- Zones de tracé
- TsgcChartInteraction
- Échelle de couleurs

TsgcChartSeries_Funnel

Unit: sgcCharts_Series_Funnel
TsgcChartSeries_Funnel dessine les étapes d'un processus, comme un pipeline de ventes, sous la forme d'un entonnoir qui se rétrécit jusqu'à un col. C'est une série de catégories (voir les méthodes des séries de catégories de la rubrique du camembert) : chaque élément de **Stages** est une étape, de haut en bas dans l'ordre de la collection. Un entonnoir est une série à cellule, voir Zones de tracé.

Propriétés

Propriété	Type	Description
Stages	TsgcChartCategoryItems	Les étapes, de haut en bas.
NeckWidthPercent	Double	Largeur du col, en pourcentage de la largeur de l'entonnoir. Par défaut 30.
NeckHeightPercent	Double	Hauteur du col, en pourcentage de la hauteur de l'entonnoir. Par défaut 25.
ProportionalHeight	Boolean	False (par défaut) : hauteurs d'étapes égales. True : hauteurs selon la valeur.
Gap	Integer	Espace entre deux étapes, en pixels physiques à l'échelle 1. Par défaut 2.
Labels	TsgcChartSliceLabels	`cslOutside` (par défaut) dessine une colonne à droite avec des lignes de repère dans la couleur de l'étape ; `cslInside`, `cslAuto` et `cslNone` fonctionnent comme pour le camembert.
LabelFormat	string	Par défaut `{text}: {value} ({percent})`. Les jetons s'écrivent sous la forme d'un nom entre accolades : `{text}`, `{value}` et `{percent}` (part de la première étape) et `{previous}` (part de l'étape située au-dessus, 100 % pour la première).
ValueFormat	string	Masque FormatFloat du jeton de valeur ; vide (par défaut), il affiche jusqu'à 2 décimales.
PercentDecimals	Integer	Par défaut 1.
LabelStyle	TsgcUITextStyle	**Color** 0 utilise la couleur de texte du thème à l'extérieur de l'entonnoir, et un texte blanc ou foncé, choisi selon la couleur de l'étape, à l'intérieur.
BorderColor	TsgcUIColor	0 (par défaut) utilise la couleur de fond du graphique.
BorderWidth	Double	Par défaut 1.

Méthodes et géométrie du dernier rendu

Méthode	Valeur de retour	Description

StageColor(aIndex: Integer)	TsgcUIColor	La couleur avec laquelle l'étape **aIndex** est dessinée : son **Color**, sinon la couleur de palette du graphique correspondant à **aIndex**.
HalfWidthAt(aY: Double)	Double	Demi-largeur de la forme à **aY** ; 0 au-dessus ou au-dessous de l'entonnoir.
StageAt(X, Y: Double)	Integer	Position de l'étape dessinée (ou de son étiquette extérieure) en **X, Y**, -1 s'il n'y en a pas.
DrawnCount	Integer (lecture seule)	Nombre d'étapes dessinées.
DrawnStages[aPos]	TsgcChartFunnelStage (lecture seule)	Une étape dessinée, de haut en bas : **Index** (index de collection), **Value**, **Percent** (de la première étape), **Previous** (de l'étape située au-dessus), **Top**, **Bottom**, les demi-largeurs, la place et la boîte de l'étiquette, et les points de la ligne de repère.
CenterX, FunnelRect, NeckTop, NeckWidth	Double, TsgcUIRect (lecture seule)	Le centre horizontal, le rectangle englobant de la forme, le haut du col et sa largeur.

Test de survol, infobulle et légende

Un survol ou un clic sur une étape ou sur son étiquette extérieure renseigne **Index** et **SubIndex** avec l'index de collection, **YValue** avec la valeur, **Percent** avec la part de la première étape et **Text** avec le texte de l'étape. L'infobulle affiche `{text}: {value} ({percent})`. La légende affiche une entrée par étape dessinée.

Exemple : un pipeline de ventes

```
uses
  SysUtils,
  sgcUICore_Theme, sgcUICore_Canvas, sgcUICore_Canvas_Raster,
  sgcCharts_Chart, sgcCharts_Series_Category, sgcCharts_Series_Funnel;

var
  vChart: TsgcChart;
  vFunnel: TsgcChartSeries_Funnel;
  vStage: TsgcChartFunnelStage;
  vRaster: TsgcUICanvas_Raster;
  i: Integer;
begin
  vChart := TsgcChart.Create(nil);
  try
    vFunnel := TsgcChartSeries_Funnel.Create(nil);
    try
      vFunnel.Chart := vChart;
      vFunnel.Title := 'Sales pipeline';
      vFunnel.NeckWidthPercent := 25;
      vFunnel.NeckHeightPercent := 20;
      vFunnel.Labels := cslOutside;
      vFunnel.LabelFormat := '{text}: {value} ({previous} of the stage above)';
      vFunnel.ValueFormat := '#,##0';

      vFunnel.BeginUpdate;
      try
        vFunnel.SetValue('Visits', 12000);
        vFunnel.SetValue('Sign-ups', 4200);
        vFunnel.SetValue('Trials', 1800);
        vFunnel.SetValue('Quotes', 640);
        vFunnel.SetValue('Orders', 310);
      finally
        vFunnel.EndUpdate;
```

```
      end;

      vRaster := TsgcUICanvas_Raster.Create(700, 420);
      try
        vChart.Render(vRaster, sgcUIRect(0, 0, 700, 420));
        vRaster.SaveToPNGFile('funnel.png');
      finally
        vRaster.Free;
      end;

      // the stages of the last render, top to bottom
      for i := 0 to vFunnel.DrawnCount - 1 do
      begin
        vStage := vFunnel.DrawnStages[i];
        Writeln(vFunnel.Stages[vStage.Index].Text, ': ',
          FormatFloat('0.0', vStage.Percent), '% of the first stage');
      end;
    finally
      vFunnel.Free;
    end;
  finally
    vChart.Free;
  end;
end.
```

Voir aussi

- TsgcChartSeries
- Zones de tracé
- TsgcChartInteraction

TsgcVCLChart

Unit: sgcCharts_VCL
TsgcVCLChart est la vue VCL d'un modèle TsgcChart : une classe descendante de `TCustomControl`, réservée à Windows, qui peint le graphique assigné à sa propriété **Chart**. Le contrôle enregistre un écouteur sur le graphique et se redessine chaque fois que le modèle change ; le modèle lui-même ne porte aucun code de dessin, si bien que le même graphique peut aussi être affiché par un TsgcFMXChart ou être rendu sans interface graphique par un TsgcChartRenderer en même temps.

Propriétés

Propriété	Type	Description
Chart	TsgcChart	Le modèle de graphique que ce contrôle peint. L'assigner enregistre le contrôle comme écouteur ; l'effacer, ou détruire le graphique, empêche le contrôle de dessiner quoi que ce soit.
Frame-Count	Cardinal (lecture seule)	Nombre d'images complètes rendues depuis la création du contrôle.
Overlay-Frame-Count	Cardinal (lecture seule)	Nombre d'images de superposition seule rendues depuis la création du contrôle. Voir **Redessins de superposition uniquement** ci-dessous.

Les propriétés habituelles de `TCustomControl` sont également publiées sans changement : **Align**, **Anchors**, **Color**, **Constraints**, **DragCursor**, **DragKind**, **DragMode**, **Enabled**, **ParentColor**, **ParentShowHint**, **PopupMenu**, **ShowHint**, **TabOrder**, **TabStop**, **Visible**, ainsi que les événements de souris, de glisser-déposer et de redimensionnement.

Méthodes

Méthode	Valeur de retour	Description
SaveToPNG(const aFileName: string)		Effectue le rendu de **Chart** à la taille client et à la résolution (PPI) actuelles du contrôle dans un nouveau fichier PNG. Ne fait rien lorsque **Chart** est nil.
SaveToSVG(const aFileName: string)		Effectue le rendu de **Chart** à la taille client et à la résolution (PPI) actuelles du contrôle dans un fichier SVG. Ne fait rien lorsque **Chart** est nil.

Limitation par MaxFPS

MaxFPS n'est pas une propriété de ce contrôle, elle appartient au modèle de graphique (voir TsgcChart). Une notification de changement provenant du graphique se contente de marquer le contrôle comme devant être redessiné et de programmer un redessin, au plus **Chart.MaxFPS** fois par seconde : si la prochaine image est déjà due, le contrôle est invalidé immédiatement, sinon un minuteur à usage unique l'invalide quand l'image devient due. Les images sont cadencées par rapport à une échéance glissante (échéance précédente plus 1000 divisé par **MaxFPS**), et non par rapport à un intervalle fixe depuis le dernier dessin, de sorte que la granularité du minuteur Windows ne laisse pas le débit effectif dériver sous le plafond. Une notification qui arrive sur un thread d'alimentation ne touche jamais directement le contrôle : elle poste un message privé que le thread principal récupère avant de programmer le redessin.

Redessins de superposition uniquement

Le contrôle enregistre aussi un second écouteur, de superposition, sur le graphique. Un changement qui ne fait que déplacer le réticule, l'infobulle ou une sélection en bande élastique (**TsgcChart.InvalidateOverlay**) ne déclenche pas de redessin complet : les axes, les séries, l'arrière-plan, le titre et la légende ne sont pas redessinés. À la place, les pixels du dernier rendu complet sont recopiés dans le raster interne et seul **TsgcChart.RenderOverlay** est dessiné par-dessus, ce qui explique pourquoi **OverlayFrameCount** et **FrameCount** sont comptabilisés séparément. Un rendu complet se produit malgré tout chaque fois que le contrôle doit être redessiné pour une autre raison, comme un redimensionnement, un changement de résolution (PPI), ou une modification des données, des axes, du thème ou de la mise en page du graphique.

Exemple : tracer un graphique dans un formulaire VCL

```
uses
  Forms, Classes, SysUtils, Math, ExtCtrls, Controls,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Axis, sgcCharts_Types,
  sgcCharts_VCL;

type
  TfrmMain = class(TForm)
  private
    FChart: TsgcChart;
    FView: TsgcVCLChart;
    FSeries: TsgcChartSeries_Line;
  public
    constructor Create(aOwner: TComponent); override;
  end;

constructor TfrmMain.Create(aOwner: TComponent);
var
  i: Integer;
begin
  inherited CreateNew(aOwner);
  Caption := 'sgcCharts VCL Sample';
  Width := 820;
  Height := 480;

  FChart := TsgcChart.Create(Self);
  FChart.Axes.Add.Name := 'bottom';
  FChart.Axes[0].Position := capBottom;
  FChart.Axes.Add.Name := 'left';
  FChart.Axes[1].Position := capLeft;

  FSeries := TsgcChartSeries_Line.Create(Self);
  FSeries.Chart := FChart;
  FSeries.AxisX := 'bottom';
  FSeries.AxisY := 'left';
  for i := 0 to 99 do
    FSeries.AddXY(i, Sin(i / 10) * 50 + 50);

  FView := TsgcVCLChart.Create(Self);
  FView.Parent := Self;
  FView.Align := alClient;
  FView.Chart := FChart;
end;

// elsewhere, e.g. from a menu action or before closing:
FView.SaveToPNG('chart.png');
```

Voir aussi

- TsgcChart
- TsgcFMXChart

- TsgcChartRenderer
- TsgcChartSeries_Line
- TsgcChartInteraction
- Panneaux

TsgcFMXChart

Unit: sgcCharts_FMX
TsgcFMXChart est la vue FireMonkey d'un modèle TsgcChart : une classe descendante de `TControl` qui peint le graphique assigné à sa propriété **Chart** sur Windows, macOS, iOS et Android. L'unité ne compile en rien tant que `SGC_CHARTS_FMX` n'est pas défini, ce qui nécessite Delphi XE6 ou une version ultérieure. Le modèle ne porte aucun code de dessin, si bien que le même graphique peut aussi être affiché par un TsgcVCLChart ou être rendu sans interface graphique par un TsgcChartRenderer en même temps.

Propriétés

Propriété	Type	Description
Chart	TsgcChart	Le modèle de graphique que ce contrôle peint. L'assigner enregistre le contrôle comme écouteur ; l'effacer, ou détruire le graphique, empêche le contrôle de dessiner quoi que ce soit.
Frame-Count	Cardinal (lecture seule)	Nombre d'images complètes rendues depuis la création du contrôle.
Overlay-Frame-Count	Cardinal (lecture seule)	Nombre d'images de superposition seule rendues depuis la création du contrôle. Voir **Redessins de superposition uniquement** ci-dessous.

Les propriétés habituelles de `TControl` sont également publiées sans changement : **Align**, **Anchors**, **ClipChildren**, **ClipParent**, **Cursor**, **DragMode**, **EnableDragHighlight**, **Enabled**, **Locked**, **Height**, **HitTest**, **Padding**, **Opacity**, **Margins**, **PopupMenu**, **Position**, **RotationAngle**, **RotationCenter**, **Scale**, **Visible**, **Width**, **TouchTargetExpansion**, ainsi que les événements de souris, de glisser-déposer et de redimensionnement.

Méthodes

Méthode	Valeur de retour	Description
SaveToPNG(const aFileName: string)		Effectue le rendu de **Chart** à la taille et à l'échelle de scène actuelles du contrôle dans un nouveau fichier PNG. Ne fait rien lorsque **Chart** est nil.
SaveToSVG(const aFileName: string)		Effectue le rendu de **Chart** à la taille et à l'échelle de scène actuelles du contrôle dans un fichier SVG. Ne fait rien lorsque **Chart** est nil.
DetectPixelOrder(var aData: TBitmapData) [class function]	TsgcFMXChartPixelOrder	Écrit un pixel de test dans les données de bitmap mappées et le relit pour déterminer si le bitmap de la plateforme stocke l'ordre des canaux `fpoBGRA`, `fpoRGBA` ou un autre (`fpoOther`). Utilisé en interne avant de copier les pixels rendus dans un bitmap FMX ; la plupart des applications ne l'appellent jamais directement.
CopyToBitmapData(aRasterizer: TsgcUIRasterizer; var aData: TBitmapData) [class procedure]		Convertit les pixels ARGB droits de **aRasterizer** en pixels prémultipliés de données de bitmap mappées, sur la plus petite des deux tailles. Utilisé en interne

		par **Paint** ; la plupart des applications ne l'appellent jamais directement.

Limitation par MaxFPS

MaxFPS n'est pas une propriété de ce contrôle, elle appartient au modèle de graphique (voir TsgcChart). Une notification de changement provenant du graphique se contente de marquer le contrôle comme devant être redessiné et de programmer un redessin, au plus **Chart.MaxFPS** fois par seconde : si la prochaine image est déjà due, le contrôle est redessiné immédiatement, sinon un minuteur à usage unique le redessine quand l'image devient due. Les images sont cadencées par rapport à une échéance glissante (échéance précédente plus 1000 divisé par **MaxFPS**), et non par rapport à un intervalle fixe depuis le dernier dessin. Une notification qui arrive sur un thread d'alimentation ne touche jamais directement le contrôle : elle place un appel en file d'attente vers le thread principal avec **TThread.Queue** (au plus un appel en attente), et le thread principal programme le redessin ; le destructeur retire un appel encore en file d'attente.

Redessins de superposition uniquement

Le contrôle enregistre aussi un second écouteur, de superposition, sur le graphique. Un changement qui ne fait que déplacer le réticule, l'infobulle ou une sélection en bande élastique (**TsgcChart.InvalidateOverlay**) ne déclenche pas de redessin complet : les axes, les séries, l'arrière-plan, le titre et la légende ne sont pas redessinés. À la place, les pixels du dernier rendu complet sont recopiés dans le raster interne et seul **TsgcChart.RenderOverlay** est dessiné par-dessus, ce qui explique pourquoi **OverlayFrameCount** et **FrameCount** sont comptabilisés séparément. Un rendu complet se produit malgré tout chaque fois que le contrôle doit être redessiné pour une autre raison, comme un redimensionnement, un changement d'échelle de scène, ou une modification des données, des axes, du thème ou de la mise en page du graphique.

Exemple : tracer un graphique dans un formulaire FMX

```
uses
  System.SysUtils, System.Types, System.Math, System.Classes,
  FMX.Types, FMX.Forms, FMX.Controls,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Axis, sgcCharts_Types,
  sgcCharts_FMX;

type
  TfrmMain = class(TForm)
  private
    FChart: TsgcChart;
    FView: TsgcFMXChart;
    FSeries: TsgcChartSeries_Line;
  public
    constructor Create(aOwner: TComponent); override;
  end;

constructor TfrmMain.Create(aOwner: TComponent);
var
  i: Integer;
begin
  inherited CreateNew(aOwner);
  Caption := 'sgcCharts FMX Sample';
  Width := 820;
  Height := 480;

  FChart := TsgcChart.Create(Self);
  FChart.Axes.Add.Name := 'bottom';
  FChart.Axes[0].Position := capBottom;
  FChart.Axes.Add.Name := 'left';
  FChart.Axes[1].Position := capLeft;

  FSeries := TsgcChartSeries_Line.Create(Self);
  FSeries.Chart := FChart;
  FSeries.AxisX := 'bottom';
```

```
  FSeries.AxisY := 'left';
  for i := 0 to 99 do
    FSeries.AddXY(i, Sin(i / 10) * 50 + 50);

  FView := TsgcFMXChart.Create(Self);
  FView.Parent := Self;
  FView.Align := TAlignLayout.Client;
  FView.Chart := FChart;
end;

// elsewhere, e.g. from an action or before closing:
FView.SaveToPNG('chart.png');
```

Voir aussi

- TsgcChart
- TsgcVCLChart
- TsgcChartRenderer
- TsgcChartSeries_Line
- TsgcChartInteraction
- Panneaux

TsgcChartRenderer

Unit: sgcCharts_Renderer
TsgcChartRenderer est la vue sans interface graphique d'un modèle TsgcChart : un simple `TComponent` sans dépendance VCL ou FMX et sans fenêtre propre, qui rend donc un graphique dans un fichier ou un flux PNG, un document SVG ou un tampon de pixels ARGB brut, depuis un programme console, un service Windows, un module ISAPI ou un démon Linux64. N'ayant aucune surface visuelle, il n'est pas affecté par **Chart.MaxFPS** : il n'y a ni boucle d'images ni minuteur, chaque appel de rendu peint le graphique de façon synchrone au moment même où il est appelé.

Propriétés

Propriété	Type	Description
Chart	TsgcChart	Le modèle de graphique que ce moteur de rendu peint. Chaque méthode Render* lève **EsgcChartRendererError** lorsque cette valeur est nil.
Width	Integer	Largeur logique par défaut en pixels utilisée par les surcharges sans taille. Par défaut 800.
Height	Integer	Hauteur logique par défaut en pixels utilisée par les surcharges sans taille. Par défaut 500.
Scale	Double	Pixels physiques par pixel logique. Par défaut 1. La sortie fait `Round(Width * Scale)` par `Round(Height * Scale)` pixels physiques ; **Chart.Scale** est réglé à cette valeur pour le rendu puis restauré à sa valeur précédente ensuite.
CompressPNG	Boolean	Indique si la sortie PNG est compressée par déflation. Par défaut True lorsque la bibliothèque a été compilée avec `SGC_CHARTS_DEFLATE` , sinon par défaut False.

Méthodes

Méthode	Valeur de retour	Description
RenderToPNGFile(const aFileName: string) RenderToPNGFile(const aFileName: string; aWidth, aHeight: Integer)		Effectue le rendu dans un fichier PNG, à la taille **Width**/**Height** ou à la taille logique indiquée sans modifier **Width** et **Height**.
RenderToPNGStream(aStream: TStream) RenderToPNGStream(aStream: TStream; aWidth, aHeight: Integer)		Effectue le rendu PNG (RGBA, 8 bits) vers un flux.
RenderToSVG RenderToSVG(aWidth, aHeight: Integer)	string	Effectue le rendu et retourne un document SVG 1.1 (ASCII pur) sous forme de chaîne.
RenderToSVGFile(const aFileName: string) RenderToSVGFile(const aFileName: string; aWidth, aHeight: Integer)		Effectue le rendu SVG vers un fichier.
RenderToSVGStream(aStream: TStream) RenderToSVGStream(aStream: TStream; aWidth, aHeight: Integer)		Effectue le rendu SVG vers un flux.

RenderToBuffer(var aBuffer: TsgcUICardinalArray; out aPixelWidth, aPixelHeight: Integer) RenderToBuffer(var aBuffer: TsgcUICardinalArray; out aPixelWidth, aPixelHeight: Integer; aWidth, aHeight: Integer)		Effectue le rendu en pixels $AARRGGBB droits (non prémultipliés), de haut en bas, avec un pas de ligne égal à **aPixelWidth** ; **aPixelWidth** et **aPixelHeight** retournent la taille physique.

Threads

Chaque méthode de rendu peut être appelée depuis n'importe quel thread. Chaque **TsgcChartRenderer** possède ses propres canevas, un raster et un SVG, protégés par un verrou propre à cette instance de moteur de rendu. En plus de cela, chaque rendu effectué par un **TsgcChartRenderer** quel qu'il soit dans le processus passe par un verrou supplémentaire partagé par toute l'unité, car **TsgcChart.Render** conserve l'état de mise en page sur le graphique lui-même pendant son exécution. Pendant qu'un rendu est en cours, le graphique ne doit pas être modifié structurellement, c'est-à-dire qu'aucune série ni aucun axe ne doit être ajouté, supprimé ou libéré, et qu'aucune propriété affectant la mise en page ne doit être modifiée ; ajouter des données via les tampons des séries (**Buffer.Append**, **OHLCBuffer.Append**, les sources de données) est sûr, car les séries dessinent à partir d'un instantané pris sous le verrou propre du tampon.
Le verrou de rendu partagé n'est pris que par **TsgcChartRenderer**. Une vue visuelle, TsgcVCLChart ou TsgcFMXChart, ne le prend pas ; il ne faut donc pas effectuer le rendu du même graphique avec un moteur de rendu dans un thread pendant qu'une vue visuelle le peint au même moment.

Exemple : rendu sans interface graphique vers PNG et SVG

```
uses
  SysUtils, Math,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Axis, sgcCharts_Types,
  sgcCharts_Renderer;

var
  vChart: TsgcChart;
  vSeries: TsgcChartSeries_Line;
  vRenderer: TsgcChartRenderer;
  i: Integer;
begin
  vChart := TsgcChart.Create(nil);
  try
    vChart.Title.Text := 'Headless Render';
    vChart.Axes.Add.Name := 'bottom';
    vChart.Axes[0].Position := capBottom;
    vChart.Axes.Add.Name := 'left';
    vChart.Axes[1].Position := capLeft;

    vSeries := TsgcChartSeries_Line.Create(nil);
    try
      vSeries.Chart := vChart;
      vSeries.AxisX := 'bottom';
      vSeries.AxisY := 'left';
      for i := 0 to 99 do
        vSeries.AddXY(i, Sin(i / 10) * 100);

      vRenderer := TsgcChartRenderer.Create(nil);
      try
        vRenderer.Chart := vChart;
        vRenderer.Width := 800;
        vRenderer.Height := 400;
        vRenderer.RenderToPNGFile('renderer.png');
        vRenderer.RenderToSVGFile('renderer.svg');
      finally
        vRenderer.Free;
      end;
    finally
      vSeries.Free;
```

```
    end;
  finally
    vChart.Free;
  end;
end;
```

Voir aussi

- TsgcChart
- TsgcVCLChart
- TsgcFMXChart
- TsgcChartSeries_Line
- Panneaux

TsgcChartInteraction

Unit: sgcCharts_Interaction
TsgcChartInteraction est l'API d'entrée d'un TsgcChart : un `TPersistent`, sans dépendance à un framework, qui transforme les appels bruts de souris, de molette et de gestes en lectures de réticule, clics sur des points, zoom, panoramique et, dans l'édition financière, en comportement d'outil de dessin et de séparateur de panneau. Il ne porte aucune dépendance à VCL, FMX ou à une quelconque API de fenêtrage ; une vue transmet ce qu'elle reçoit de ses propres événements d'entrée (un TsgcVCLChart, un TsgcFMXChart, ou une boucle hôte personnalisée) sous forme de coordonnées en pixels du graphique, le même espace en pixels physiques que le rectangle donné à **TsgcChart.Render**. Chaque graphique possède exactement une instance d'interaction, créée dans son constructeur et exposée en tant que **TsgcChart.Interaction** ; les applications la configurent là plutôt que d'en créer une elles-mêmes.

Propriétés

Propriété	Type	Description
Crosshair	Boolean	Peint le réticule et les lectures d'axe pendant le survol du tracé. Par défaut True.
CrosshairColor	TsgcUIColor	0 (par défaut) utilise la couleur du thème.
Tooltip	Boolean	Affiche l'infobulle du point le plus proche pendant le survol. Par défaut True.
ZoomMode	TsgcChartZoomMode	`czmNone`, `czmX`, `czmY` ou `czmXY` : sur quels axes agissent le zoom et le panoramique. Par défaut `czmX`.
WheelZoom	Boolean	La molette zoome autour du curseur ; Maj + molette effectue un panoramique horizontal à la place. Par défaut True.
BoxZoom	Boolean	Un glisser du bouton gauche dessine une bande élastique et zoome dessus. Par défaut True. Ctrl + glisser du bouton gauche effectue toujours un panoramique à la place, et un glisser du bouton droit ou du milieu effectue un panoramique indépendamment de cette propriété.
Pan	Boolean	Active le panoramique par glisser. Par défaut True.
AutoScrollResumeDelay	Integer	Millisecondes sans interaction avant qu'un zoom horizontal ne soit réinitialisé et qu'un **TsgcChart.TimeWindow** ne défile à nouveau ; 0 signifie ne jamais reprendre automatiquement. Par défaut 3000. Voir Défilement automatique.
SnapToPoint	Boolean	Le réticule saute au point le plus proche : le plus proche en X, puis le plus proche en Y parmi les séries à ce X (sinon le plus proche en distance de pixels quand False). Par défaut True.
DrawingTool	TsgcChartDrawingTool	Édition financière uniquement. `cdtNone` (par défaut) permet à un appui du bouton gauche de sélectionner et de faire glisser un dessin existant ; toute autre valeur (`cdtTrendLine`, `cdtHorizontalLine`, `cdtVerticalLine`, `cdtRectangle`, `cdtFibonacci`) fait que le prochain glisser du bouton gauche dans le tracé crée un dessin de ce type (appui = premier ancrage, relâchement = second ancrage), après quoi la propriété revient à `cdtNone`.

Paused	Boolean (lecture seule)	True tant que le défilement automatique est mis en pause par un zoom ou un panoramique. Voir Défilement automatique.
Dragging	Boolean (lecture seule)	True tant qu'un zoom par boîte, un panoramique, un glissement de séparateur ou de dessin est en cours.
DragMode	TsgcChartDragMode (lecture seule)	`cdmNone` , `cdmBox` , `cdmPan` , ou, dans l'édition financière, `cdmSplitter` (un séparateur de panneau est glissé, voir Panneaux), `cdmDrawCreate` ou `cdmDrawMove` (un dessin ou l'un de ses ancrages est glissé).
Chart	TComponent (lecture seule)	Le graphique auquel cette interaction est attachée.

Méthodes

Méthode	Valeur de retour	Description
MouseMove(X, Y: Double)		Alimente un déplacement de pointeur, en pixels de graphique. Met à jour le réticule, poursuit un glissement en cours, ou déclenche **TsgcChart.OnCrosshairMove** au repos au-dessus du tracé.
MouseDown(X, Y: Double; aButton: TsgcChartMouseButton; aShift: TsgcChartShiftState)		Alimente un appui de bouton. **aButton** vaut `cmbLeft` , `cmbRight` ou `cmbMiddle` ; **aShift** est un ensemble de `cskShift` , `cskCtrl` , `cskAlt` . Démarre un zoom par boîte, un panoramique, un glissement de séparateur ou de dessin selon les propriétés ci-dessus.
MouseUp(X, Y: Double; aButton: TsgcChartMouseButton; aShift: TsgcChartShiftState)		Alimente un relâchement de bouton. Termine le glissement démarré par **MouseDown**, ou, lorsque le pointeur n'a jamais dépassé le seuil de glissement, résout un clic et déclenche **TsgcChart.OnSeriesClick**.
MouseWheel(X, Y: Double; aDelta: Integer; aShift: TsgcChartShiftState)		Alimente un cran de molette. **aDelta** vaut 120 par cran, positif en s'éloignant de l'utilisateur (zoom avant).
MouseLeave		Alimente la sortie du pointeur du contrôle, masquant le réticule et l'infobulle.
DoubleClick(X, Y: Double)		Alimente un double clic, en pixels de graphique. Appelle **ResetZoom**.
Gesture(aKind: TsgcChartGestureKind; aCenterX, aCenterY, aFactorOrDX, aDY: Double)		Alimente un geste tactile. Pour `cgZoom` , **aFactorOrDX** est l'échelle du pincement (supérieure à 1 zoome en avant) autour de **aCenterX, aCenterY** ; pour `cgPan` , **aFactorOrDX** et **aDY** sont le déplacement du doigt en pixels.
ResetZoom		Réinitialise le zoom de chaque axe et reprend le défilement automatique.

DeleteSelectedDrawing	Boolean	Édition financière uniquement. Supprime le dessin sélectionné de **TsgcChart.Drawings** sauf s'il est Locked ; une vue l'appelle pour la touche Suppr. True quand un dessin a été supprimé.
HitTest(X, Y: Double)	TsgcChartHitInfo	Ce qui se trouve sous une position en pixels du graphique après le dernier **TsgcChart.Render**, sans effet de bord. Voir HitTest et TsgcChartHitInfo.
CheckAutoScroll	Boolean	Appelée par **TsgcChart.Render** avant que les plages ne soient calculées. Réinitialise le zoom horizontal une fois que **AutoScrollResumeDelay** s'est écoulé depuis la dernière interaction ; True quand ce fut le cas. Voir Défilement automatique.
DrawOverlay(aCanvas: TsgcUICanvas; const aRect: TsgcUIRect; aScale: Double)		Appelée par **TsgcChart.Render** après le contenu du graphique. Peint le réticule, l'infobulle et la bande élastique à l'intérieur de **aRect** de **aCanvas**.

Événements

Ces trois événements sont publiés sur **TsgcChart** lui-même, pas sur TsgcChartInteraction : c'est l'interaction qui les déclenche sur le graphique auquel elle est attachée, si bien qu'une application les assigne sur le graphique. Voir TsgcChart.

Événement	Type	Description
TsgcChart.OnCrosshairMove	TsgcChartHitEvent	Le pointeur s'est déplacé au-dessus du tracé au repos (aucun glissement en cours) ; **aHit** est le résultat de **HitTest** à la position du pointeur.
TsgcChart.OnSeriesClick	TsgcChartHitEvent	Un clic gauche est tombé dans le rayon de clic d'un point ; **aHit.Series** et **aHit.Index** l'identifient.
TsgcChart.OnZoom	TNotifyEvent	Déclenché après un zoom interactif, un panoramique ou **ResetZoom**.

TsgcChartHitEvent est déclaré comme :

```
TsgcChartHitEvent = procedure(Sender: TObject; const aHit: TsgcChartHitInfo)
  of object;
```

HitTest et TsgcChartHitInfo

HitTest indique ce que recouvre une position en pixels du graphique, en utilisant le rectangle de tracé calculé par le dernier **TsgcChart.Render** ; elle ne change rien et peut être appelée sans risque à tout moment, y compris en dehors de **MouseMove**, **MouseDown** ou **MouseUp**. Elle retourne un enregistrement **TsgcChartHitInfo** avec les champs suivants :

Champ	Type	Description
Area	TsgcChartHitArea	La partie du graphique sous la position : `chaNone` , `chaPlot` , `chaAxis` , `chaLegend` ou `chaTitle` .
Axis	TsgcChartAxis	L'axe sous la position quand **Area** vaut `chaAxis` .

Series	TCom-ponent	Le **TsgcChartSeries** du point le plus proche, nil quand il n'y en a aucun. Déclaré comme TComponent car TsgcChartInteraction est compilé avant Tsgc-Chart et les types de série.
SeriesIndex	Integer	Index de **Series** dans le graphique, -1 quand il n'y en a aucun.
Index	Integer	Index logique du point dans le tampon, 0 est le plus ancien, -1 quand il n'y en a aucun.
XValue, YValue	Double	Valeurs du point. **YValue** est la clôture d'un chandelier.
IsOHLC	Boolean	True quand le point provient d'une série OHLC ou chandelier.
Open, High, Low, Close	Double	Renseignés uniquement quand **IsOHLC** vaut True.
PixelX, PixelY	Double	Position du point en pixels de graphique.
Distance	Double	Distance en pixels entre la position interrogée et le point ; 0 à l'intérieur d'une barre ou d'un chandelier.
CursorX, CursorY	Double	La position interrogée, telle que donnée à **HitTest**.
SubIndex	Integer	Élément de la série sous la position : un secteur, un point de radar, une cellule de carte de chaleur, une profondeur de treemap ou une étape d'entonnoir ; -1 s'il n'y en a aucun.
ZValue	Double	Troisième valeur de l'élément (la valeur d'une cellule de carte de chaleur), NaN s'il n'y en a pas.
Percent	Double	Part de l'élément dans sa série (camembert, entonnoir, treemap), NaN s'il n'y en a pas.
Text	string	Texte de catégorie de l'élément (secteur, rayon, cellules, chemin de nœud ou étape), vide s'il n'y en a pas.

TsgcChart.OnSeriesClick ne se déclenche que lorsque **Distance** se trouve dans un petit rayon de clic du point ; un clic qui tombe sur une zone de tracé vide appelle quand même **HitTest** en interne mais ne déclenche pas l'événement.

Zones de tracé : survol, clic et actions des séries

Le survol, le clic et les infobulles atteignent les séries de chaque mode de zone de tracé : cellules de camembert, d'entonnoir et de treemap, zones polaires des radars, et la carte de chaleur dans une zone XY. Le zoom, le déplacement, le réticule et la réinitialisation du zoom par double-clic ne fonctionnent que dans les zones XY. Il n'y a aucun nouvel événement de graphique : le survol déclenche **OnCrosshairMove** et un clic déclenche **OnSeriesClick**, tous deux avec les champs de survol ci-dessus ; après **OnSeriesClick**, la série exécute sa propre action de clic via **AreaClick**, de sorte qu'un camembert avec **ExplodeOnClick** tire le secteur vers l'extérieur et qu'un treemap avec **DrillOnClick** descend dans le groupe. Une série peut fournir son propre texte d'infobulle, dessiné sous forme de lignes simples. Avec une carte de chaleur sous une série en courbe, un point de courbe situé dans le rayon de clic l'emporte, sinon c'est la cellule de la carte de chaleur.

```
uses
  SysUtils,
  sgcUICore_Theme, sgcUICore_Canvas, sgcUICore_Canvas_Raster,
  sgcCharts_Chart, sgcCharts_Interaction, sgcCharts_Series_Category,
  sgcCharts_Series_Pie;

var
```

```
  vChart: TsgcChart;
  vPie: TsgcChartSeries_Pie;
  vRaster: TsgcUICanvas_Raster;
  vRect: TsgcUIRect;
  vHit: TsgcChartHitInfo;
  vX, vY: Double;
begin
  vChart := TsgcChart.Create(nil);
  vRaster := TsgcUICanvas_Raster.Create(400, 400);
  vPie := TsgcChartSeries_Pie.Create(nil);
  try
    vPie.Chart := vChart;
    vPie.ExplodeOnClick := True;
    vPie.SetValue('Web', 50);
    vPie.SetValue('Store', 30);
    vPie.SetValue('Phone', 20);
    // the hit test uses the geometry of the last render
    vChart.Render(vRaster, sgcUIRect(0, 0, 400, 400));

    // a point right of 12 o'clock: inside the first slice
    vRect := vChart.SeriesAreaRect(vPie);
    vX := (vRect.Left + vRect.Right) / 2 + 20;
    vY := (vRect.Top + vRect.Bottom) / 2 - 40;
    vHit := vChart.Interaction.HitTest(vX, vY);
    if vHit.Series = vPie then
      Writeln('slice ', vHit.SubIndex, ' ', vHit.Text, ': ',
        FormatFloat('0.#', vHit.Percent), '%');

    // a click: OnSeriesClick fires, then the pie explodes the slice
    vChart.Interaction.MouseDown(vX, vY, cmbLeft, []);
    vChart.Interaction.MouseUp(vX, vY, cmbLeft, []);
    Writeln('exploded: ',
      TsgcChartPieSlice(vPie.Slices[vHit.SubIndex]).Exploded);
  finally
    vPie.Free;
    vRaster.Free;
    vChart.Free;
  end;
end.
```

Défilement automatique

Un graphique avec un **TsgcChart.TimeWindow** non nul garde normalement son axe horizontal affichant les données les plus récentes. Zoomer ou faire un panoramique sur les axes horizontaux fige ce comportement afin que la plage zoomée reste à l'écran pendant que de nouvelles données continuent d'arriver ; **Paused** indique True tant que c'est le cas. Il n'y a pas de méthode séparée de mise en pause ou de reprise : **TsgcChart.Render** appelle **CheckAutoScroll** à chaque rendu, et une fois que **AutoScrollResumeDelay** millisecondes se sont écoulées sans interaction, le zoom horizontal est réinitialisé et la fenêtre temporelle défile à nouveau ; **AutoScrollResumeDelay** mis à 0 désactive la reprise automatique, laissant **ResetZoom** comme seul moyen de revenir en arrière. Aucun minuteur n'est impliqué, la vérification n'a lieu qu'au moment du rendu.

Exemple : alimenter une entrée synthétique et lire le résultat

TsgcChartInteraction prend de simples coordonnées, aucun formulaire VCL ou FMX n'est donc nécessaire pour l'exercer, un programme console peut le piloter directement une fois que le graphique a été rendu au moins une fois (afin que **PlotRect** soit connu) :

```
uses
  SysUtils, Math,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Axis, sgcCharts_Types,
  sgcUICore_Canvas, sgcUICore_Canvas_Raster, sgcCharts_Interaction;
```

```
type
  THandler = class
    procedure OnCrosshair(Sender: TObject; const aHit: TsgcChartHitInfo);
    procedure OnClick(Sender: TObject; const aHit: TsgcChartHitInfo);
  end;

procedure THandler.OnCrosshair(Sender: TObject; const aHit: TsgcChartHitInfo);
begin
  Writeln('crosshair: seriesIndex=', aHit.SeriesIndex, ' index=', aHit.Index,
    ' x=', aHit.XValue: 0: 2, ' y=', aHit.YValue: 0: 2);
end;

procedure THandler.OnClick(Sender: TObject; const aHit: TsgcChartHitInfo);
begin
  if aHit.Series <> nil then
    Writeln('click: series=', TsgcChartSeries(aHit.Series).Name, ' index=',
      aHit.Index)
  else
    Writeln('click: no point near cursor');
end;

var
  vChart: TsgcChart;
  vSeries: TsgcChartSeries_Line;
  vRaster: TsgcUICanvas_Raster;
  vHandler: THandler;
  vHit: TsgcChartHitInfo;
  vAxisX, vAxisY: TsgcChartAxis;
  vClickX, vClickY: Double;
  i: Integer;
begin
  vChart := TsgcChart.Create(nil);
  vHandler := THandler.Create;
  try
    // TsgcChart already owns a bottom and a left axis, and Interaction
    // (created automatically, as vChart.Interaction)
    vSeries := TsgcChartSeries_Line.Create(nil);
    try
      vSeries.Name := 'Signal';
      vSeries.Chart := vChart;
      for i := 0 to 99 do
        vSeries.AddXY(i, Sin(i / 10) * 50 + 50);

      vChart.OnCrosshairMove := vHandler.OnCrosshair;
      vChart.OnSeriesClick := vHandler.OnClick;

      vRaster := TsgcUICanvas_Raster.Create(400, 300);
      try
        vChart.Scale := 1;
        vChart.Render(vRaster, sgcUIRect(0, 0, 400, 300));

        // hover: fires OnCrosshairMove through the handler above
        vChart.Interaction.MouseMove(200, 150);

        // HitTest reports the point nearest to a chart pixel position
        vHit := vChart.Interaction.HitTest(200, 150);
        Writeln('HitTest: seriesIndex=', vHit.SeriesIndex, ' index=',
          vHit.Index, ' x=', vHit.XValue: 0: 2, ' y=', vHit.YValue: 0: 2);

        // click exactly on that point: fires OnSeriesClick
        vAxisX := vSeries.XAxis;
        vAxisY := vSeries.YAxis;
        vClickX := vAxisX.ValueToPixel(vHit.XValue, vChart.PlotRect.Left,
          vChart.PlotRect.Right - vChart.PlotRect.Left);
        vClickY := vAxisY.ValueToPixel(vHit.YValue, vChart.PlotRect.Top,
          vChart.PlotRect.Bottom - vChart.PlotRect.Top);
        vChart.Interaction.MouseDown(vClickX, vClickY, cmbLeft, []);
```

```
        vChart.Interaction.MouseUp(vClickX, vClickY, cmbLeft, []);
      finally
        vRaster.Free;
      end;
    finally
      vSeries.Free;
    end;
  finally
    vHandler.Free;
    vChart.Free;
  end;
end.
```

Ceci affiche le résultat du réticule, le résultat équivalent de **HitTest**, et la série et l'index résolus par le clic, par exemple :

```
crosshair: seriesIndex=0 index=46 x=46.00 y=0.32
HitTest: seriesIndex=0 index=46 x=46.00 y=0.32
click: series=Signal index=46
```

Voir aussi

- TsgcChart
- Axes
- TsgcChartSeries_Line
- TsgcVCLChart
- TsgcFMXChart
- Panneaux
- Zones de tracé

TsgcChartDataSource

Unité : sgcCharts_DataSource
Classe de base de chaque source de données de graphique. Elle ajoute des points à un **Buffer** (TsgcChartBuffer) et des chandeliers à un **OHLCBuffer** (**TsgcChartOHLCBuffer**). Les deux sont définis en code et ne sont pas possédés par la source de données. Chaque ajout passe par le verrou propre au tampon, si bien qu'un thread d'alimentation peut ajouter des données pendant que le thread de rendu lit un instantané.
Mapping indique à la source de données où se trouvent les valeurs X, Y et OHLC dans chaque enregistrement entrant : un chemin JSON pour une entrée JSON, un nom de colonne ou un index de colonne basé sur 0 pour une entrée CSV, ou un nom de champ pour un dataset. Un **XPath** vide fait de X l'index d'enregistrement basé sur 0. Un enregistrement produit un point lorsque **Buffer** est assigné et que **YPath** est défini, et un chandelier lorsque **OHLCBuffer** est assigné et que **OpenPath**, **HighPath**, **LowPath** et **ClosePath** sont définis (**VolumePath** est optionnel ; un volume manquant vaut 0). Les enregistrements dont X ou les valeurs ne peuvent pas être lus ou convertis sont ignorés.
Mapping.TimeFormat convertit X : `ctfNumber` conserve le nombre, `ctfUnixSeconds` et `ctfUnixMilliseconds` transforment un temps Unix en `TDateTime` UTC, `ctfISO8601` analyse un texte ISO 8601 (un décalage `Z` ou `+hh:mm` est converti en UTC ; un texte sans fuseau est conservé tel quel) et conserve un nombre simple en tant que `TDateTime`. Les nombres sont toujours analysés avec `.` comme séparateur décimal, quelle que soit la locale.

Propriétés

Propriété	Type	Par défaut	Description
Active	Boolean	True	False fait que la source de données ignore chaque ajout.
Mapping	TsgcChartDataMapping	—	XPath, YPath, OpenPath, HighPath, LowPath, ClosePath, VolumePath, LabelPath et TimeFormat. Voir ci-dessus, et Séries de catégories et cartes de chaleur ci-dessous pour **LabelPath**.
Throttle	Integer	0	Millisecondes entre les notifications **OnDataChanged**. Voir ci-dessous.
OnBeforeAppend	TsgcChartBeforeAppendEvent	nil	Se déclenche avant qu'un point ne soit ajouté, avec `aX` et `aY` modifiables et `aAccept` défini à True ; définir `aAccept` à False ignore le point.
OnBeforeAppendOHLC	TsgcChartBeforeAppendOHLCEvent	nil	Se déclenche avant qu'un chandelier ne soit ajouté, avec `aX`, `aOpen`, `aHigh`, `aLow`, `aClose`, `aVolume` modifiables et `aAccept` défini à True ; définir `aAccept` à False ignore le chandelier.
OnDataChanged	TNotifyEvent	nil	Se déclenche après l'ajout d'un ou plusieurs points ou chandeliers. Voir Notification ci-dessous.

Méthodes

Méthode	Description
BeginUpdate / EndUpdate	Diffère les notifications jusqu'au **EndUpdate** correspondant, qui envoie alors une seule notification (limitée en fréquence) si quelque chose a été ajouté entre-temps. S'imbrique librement ; seul le **EndUpdate** le plus externe peut se déclencher.
Clear	Efface **Buffer** et **OHLCBuffer**, vide une série de catégories ou une carte de chaleur liée, et réinitialise l'index d'enregistrement utilisé quand **XPath** est vide.

Notification et threads

OnDataChanged est limité en fréquence : après un ajout, l'événement se déclenche immédiatement si la dernière notification date d'au moins **Throttle** ms, sinon un thread auxiliaire le déclenche à la fermeture de la fenêtre de limitation. Il n'y a jamais plus d'une notification en attente, si bien qu'une rafale d'ajouts produit un seul événement par fenêtre. **Throttle** = 0 notifie à chaque ajout (toujours regroupé tant qu'une notification est en attente).
Delphi 2010 et versions ultérieures livrent **OnDataChanged** dans le thread principal, via `TThread.Queue` ; une application console ou un service doit appeler `CheckSynchronize` pour le recevoir. Delphi 7 à 2009 ne disposent pas de `TThread.Queue` ; l'événement s'y déclenche donc dans le thread qui a effectué l'ajout, ou dans le thread de limitation, et le gestionnaire doit se synchroniser lui-même avant de toucher un contrôle.

Séries de catégories et cartes de chaleur

La propriété **Series** lie une série X/Y ou en chandeliers, mais aussi une série de catégories (camembert, radar, entonnoir, treemap) ou une carte de chaleur. **Mapping.LabelPath** est le chemin JSON ou la colonne CSV du texte de l'élément. Un enregistrement de catégorie avec une étiquette appelle **SetValue**(étiquette, Y), de sorte que la dernière valeur d'un texte l'emporte ; sans correspondance d'étiquette, il définit l'élément à X arrondi à l'index le plus proche (X est l'index d'enregistrement lorsque **XPath** est vide). Un enregistrement dont l'étiquette est mappée mais absente ou vide n'écrit aucun élément, et **OnBeforeAppend** filtre toujours (X, Y). Un enregistrement de carte de chaleur ajoute une ligne : un enregistrement JSON dont **YPath** est un tableau (la clé est X lorsque **XPath** est défini, sinon le compteur de lignes courant), ou les cellules d'un enregistrement CSV ; les valeurs qui ne sont pas des nombres sont des cellules nulles. Un chargement écrit la série liée entre son **BeginUpdate** et son **EndUpdate**, de sorte qu'elle ne se redessine qu'une fois, et une source de données n'écrit jamais dans une série au moment de la conception, de sorte que les données en direct n'atteignent jamais un DFM. Les sources dataset et table utilisent **LabelField** et **LabelColumn** à la place de **LabelPath**.

Descendants

- TsgcChartDataSource_Manual : alimentée directement depuis le code.
- TsgcChartDataSource_DataSet : charge depuis un `TDataSet`.
- TsgcChartDataSource_JSON : charge depuis un document JSON.
- TsgcChartDataSource_CSV : charge depuis du texte CSV.
- TsgcChartDataSource_Table: charge depuis une table qui implémente `IsgcUITable`, comme un modèle de données sgcGrid.
- Les sources de données passerelle, alimentées par une connexion sgcWebSockets en direct.

TsgcChartDataSource_Manual

Unité : sgcCharts_DataSource
Le composant descend de TsgcChartDataSource et hérite de **Buffer**, **OHLCBuffer**, **Mapping**, **Active**, **Throttle**, **BeginUpdate**/**EndUpdate**, **Clear** ainsi que des événements avant-ajout et de changement de données. Il n'ajoute aucune nouvelle propriété, seulement les trois méthodes qu'une application appelle directement lorsqu'elle dispose déjà des valeurs et n'a pas besoin de **Mapping** pour les trouver dans un enregistrement.

Méthodes

Méthode	Retour	Description
AppendValue(const aX, aY: Double)	Boolean	Ajoute un point à **Buffer**. Renvoie False lorsque la source de données est inactive, que **Buffer** est nil, ou que **OnBeforeAppend** a rejeté le point.
AppendOHLC(const aX, aOpen, aHigh, aLow, aClose: Double; const aVolume: Double = 0)	Boolean	Ajoute un chandelier à **OHLCBuffer**. Renvoie False lorsque la source de données est inactive, que **OHLCBuffer** est nil, ou que **OnBeforeAppendOHLC** a rejeté le chandelier.
AppendJSON(const aJSON: string)	Boolean	Lit un objet JSON, ou un tableau d'objets ou de tableaux, via **Mapping**, de la même façon que TsgcChartDataSource_JSON lit un document. Renvoie False pour un JSON mal formé ; elle ne lève jamais d'exception pour cela.
AppendCategory(const aText: string; aValue: Double)	Boolean	Écrit la valeur de **aText** dans la série de catégories liée (une insertion ou mise à jour ; X de **OnBeforeAppend** est l'index de l'élément). False lorsque la source de données est inactive, qu'aucune série de catégories n'est liée, ou que **OnBeforeAppend** l'a rejetée.
AppendLine(aKey: Double; const aValues: array of Double)	Boolean	Ajoute une ligne à **aKey** à la carte de chaleur liée. False lorsque la source de données est inactive ou qu'aucune carte de chaleur n'est liée.

Exemple : ajouter des points depuis le code

```
var
  vBuffer: TsgcChartBuffer;
  vSource: TsgcChartDataSource_Manual;
  i: Integer;
begin
  vBuffer := TsgcChartBuffer.Create(1024, cbmOverwrite);
  try
    vSource := TsgcChartDataSource_Manual.Create(nil);
    try
      vSource.Buffer := vBuffer;
      for i := 0 to 99 do
        vSource.AppendValue(i, Sin(i / 10) * 100);
    finally
      vSource.Free;
    end;
  finally
    vBuffer.Free;
  end;
end;
```

Exemple : alimenter un camembert et une carte de chaleur

```
uses
  SysUtils,
  sgcCharts_Chart, sgcCharts_DataSource, sgcCharts_Series_Category,
  sgcCharts_Series_Pie, sgcCharts_Series_Heatmap;

var
  vPie: TsgcChartSeries_Pie;
  vHeat: TsgcChartSeries_Heatmap;
  vPieSource, vHeatSource: TsgcChartDataSource_Manual;
  i: Integer;
begin
  vPie := TsgcChartSeries_Pie.Create(nil);
  vHeat := TsgcChartSeries_Heatmap.Create(nil);
  vPieSource := TsgcChartDataSource_Manual.Create(nil);
  vHeatSource := TsgcChartDataSource_Manual.Create(nil);
  try
    // a category series: AppendCategory writes the value of a text
    vPieSource.Series := vPie;
    vPieSource.AppendCategory('Chrome', 64);
    vPieSource.AppendCategory('Safari', 19);
    vPieSource.AppendCategory('Chrome', 66); // replaces 64
    // JSON records: Mapping.LabelPath names the item text
    vPieSource.Mapping.LabelPath := 'browser';
    vPieSource.Mapping.YPath := 'share';
    vPieSource.AppendJSON('[{"browser":"Edge","share":5},' +
      '{"browser":"Firefox","share":3}]');
    for i := 0 to vPie.ItemCount - 1 do
      Writeln(vPie.Slices[i].Text, ' = ', FormatFloat('0', vPie.Slices[i].Value));

    // a heatmap: AppendLine adds one line with its key
    vHeatSource.Series := vHeat;
    vHeatSource.AppendLine(0, [1, 2, 3]);
    vHeatSource.AppendLine(1, [4, 5, 6]);
    Writeln('line 1, cell 2 = ', FormatFloat('0', vHeat.Cells[1, 2]));
  finally
    vHeatSource.Free;
    vPieSource.Free;
    vHeat.Free;
    vPie.Free;
  end;
end.
```

Voir aussi

- TsgcChartDataSource
- TsgcChartBuffer

TsgcChartDataSource_DataSet

Unité : sgcCharts_DataSource_DataSet
Charge des données de graphique depuis les lignes d'un `TDataSet`. Le composant descend de TsgcChartDataSource et hérite de **Buffer**, **OHLCBuffer**, **Active**, **Throttle** ainsi que des événements avant-ajout et de changement de données ; il n'utilise pas la propriété héritée **Mapping**, car les champs sont nommés directement.
XField, **YField** et de **OpenField** à **VolumeField** nomment les champs. Un **XField** vide fait de X l'index d'enregistrement basé sur 0. Un champ date/heure donne son `TDateTime` comme X ; un champ numérique est converti par **Mapping.TimeFormat** (secondes ou millisecondes Unix) ; un champ texte est analysé comme du texte CSV. Une ligne avec un X ou un Y null ne produit aucun point, une ligne avec un open, high, low ou close null ne produit aucun chandelier, et un volume null vaut 0.

Propriétés

Propriété	Type	Description
DataSet	TDataSet	Le dataset depuis lequel les lignes sont chargées.
XField	string	Champ contenant X. Vide signifie l'index d'enregistrement basé sur 0.
YField	string	Champ contenant Y.
OpenField, HighField, LowField, CloseField	string	Champs contenant les valeurs OHLC. Les quatre doivent être définis pour qu'une ligne produise un chandelier.
VolumeField	string	Champ contenant le volume. Optionnel ; un champ null ou non défini vaut 0.
LabelField	string	Champ de texte d'élément d'une série de catégories liée (camembert, radar, entonnoir, treemap) : chaque ligne appelle **SetValue**(texte, Y), et une ligne dont le texte est nul ou vide n'écrit aucun élément. Sans **LabelField**, chaque ligne définit l'élément à son X (l'index d'enregistrement lorsque **XField** est vide). Voir Séries de catégories et cartes de chaleur.

Méthodes

Méthode	Description
LoadFromDataSet	Ajoute chaque ligne du **DataSet** ouvert, les contrôles étant désactivés puis l'enregistrement courant restauré ensuite, et envoie une seule notification à la fin. Ne fait rien lorsque **DataSet** est nil ou fermé.
Refresh	Efface les tampons et rappelle **LoadFromDataSet**. Ne fait rien lorsque **DataSet** est nil ou fermé.

Exemple : charger un tampon depuis un dataset

```
var
  vBuffer: TsgcChartBuffer;
  vSource: TsgcChartDataSource_DataSet;
  vDataSet: TClientDataSet;
begin
  vBuffer := TsgcChartBuffer.Create(4096, cbmOverwrite);
```

```
  try
    vDataSet := TClientDataSet.Create(nil);
    try
      vDataSet.FieldDefs.Add('ts', ftFloat);
      vDataSet.FieldDefs.Add('price', ftFloat);
      vDataSet.CreateDataSet;
      vDataSet.Open;
      vDataSet.AppendRecord([1.0, 100.5]);
      vDataSet.AppendRecord([2.0, 101.25]);

      vSource := TsgcChartDataSource_DataSet.Create(nil);
      try
        vSource.Buffer := vBuffer;
        vSource.DataSet := vDataSet;
        vSource.XField := 'ts';
        vSource.YField := 'price';
        vSource.LoadFromDataSet;
      finally
        vSource.Free;
      end;
    finally
      vDataSet.Free;
    end;
  finally
    vBuffer.Free;
  end;
end;
```

Voir aussi

- TsgcChartDataSource
- TsgcChartBuffer

TsgcChartDataSource_JSON

Unité : sgcCharts_DataSource_JSON
Charge des données de graphique depuis un document JSON. Le composant descend de TsgcChartDataSource et hérite de **Buffer**, **OHLCBuffer**, **Mapping**, **Active**, **Throttle** ainsi que des événements avant-ajout et de changement de données.
ArrayPath sélectionne les enregistrements ; un **ArrayPath** vide désigne la racine du document. Un tableau d'objets ou de tableaux donne un enregistrement par élément, un objet donne un enregistrement. Chaque enregistrement est ensuite lu via **Mapping**, exactement comme décrit sur TsgcChartDataSource. Les chargements ajoutent aux tampons, donc appelez d'abord **Clear** pour remplacer les données au lieu de les ajouter, et ils envoient une seule notification à la fin. Un JSON mal formé ou un **ArrayPath** manquant lève `EsgcChartJSONError`.

Propriétés

Propriété	Type	Description
ArrayPath	string	Chemin vers le tableau (ou l'objet) d'enregistrements dans le document. Vide signifie la racine du document.

Méthodes

Méthode	Description
LoadFromString(const aJSON: string)	Charge depuis une chaîne JSON.
LoadFromStream(aStream: TStream)	Charge depuis un flux de texte UTF-8 ; un BOM est optionnel.
LoadFromFile(const aFileName: string)	Charge depuis un fichier UTF-8 ; un BOM est optionnel.

Exemple : charger des chandeliers depuis un tableau JSON

```
var
  vOHLC: TsgcChartOHLCBuffer;
  vSource: TsgcChartDataSource_JSON;
const
  CS_JSON = '[{"t":1700000000,"o":100,"h":105,"l":99,"c":103,"v":10}]';
begin
  vOHLC := TsgcChartOHLCBuffer.Create(4096, cbmOverwrite);
  try
    vSource := TsgcChartDataSource_JSON.Create(nil);
    try
      vSource.OHLCBuffer := vOHLC;
      vSource.Mapping.XPath := 't';
      vSource.Mapping.TimeFormat := ctfUnixSeconds;
      vSource.Mapping.OpenPath := 'o';
      vSource.Mapping.HighPath := 'h';
      vSource.Mapping.LowPath := 'l';
      vSource.Mapping.ClosePath := 'c';
      vSource.Mapping.VolumePath := 'v';
      vSource.LoadFromString(CS_JSON);
    finally
      vSource.Free;
    end;
  finally
    vOHLC.Free;
```

```
  end;
end;
```

Voir aussi

- TsgcChartDataSource
- TsgcChartDataSource_Manual (AppendJSON)
- TsgcChartBuffer

TsgcChartDataSource_CSV

Unité : sgcCharts_DataSource_CSV
Charge des données de graphique depuis du texte CSV (RFC 4180). Le composant descend de TsgcChartDataSource et hérite de **Buffer**, **OHLCBuffer**, **Mapping**, **Active**, **Throttle** ainsi que des événements avant-ajout et de changement de données.
Les champs sont séparés par **Delimiter** ; un champ peut être mis entre guillemets avec ", et un champ entre guillemets peut contenir le délimiteur, des sauts de ligne et des guillemets doublés. Les lignes se terminent par CRLF, LF ou CR ; les lignes vides sont ignorées. Lorsque **HasHeader** vaut True, le premier enregistrement nomme les colonnes.
Les chemins de **Mapping** sélectionnent les colonnes : un nom d'en-tête (insensible à la casse) ou un index de colonne basé sur 0. Les nombres utilisent **DecimalSeparator** (. par défaut) quelle que soit la locale. Une ligne dont les valeurs mappées sont manquantes ou non numériques est ignorée. Les chargements ajoutent aux tampons (appelez d'abord **Clear** pour remplacer les données) et envoient une seule notification à la fin.

Propriétés

Propriété	Type	Par défaut	Description
Delimiter	Char	,	Séparateur de champs.
HasHeader	Boolean	True	Lorsque True, le premier enregistrement nomme les colonnes et les chemins de Mapping peuvent utiliser ces noms au lieu d'un index de colonne.
DecimalSeparator	Char	.	Séparateur décimal utilisé pour analyser les champs numériques, quelle que soit la locale.

Méthodes

Méthode	Description
LoadFromString(const aText: string)	Charge depuis une chaîne CSV.
LoadFromStream(aStream: TStream)	Charge depuis un flux de texte UTF-8 ; un BOM est optionnel.
LoadFromFile(const aFileName: string)	Charge depuis un fichier UTF-8 ; un BOM est optionnel.

Exemple : charger des points depuis un fichier CSV

```
var
  vBuffer: TsgcChartBuffer;
  vSource: TsgcChartDataSource_CSV;
begin
  vBuffer := TsgcChartBuffer.Create(8192, cbmOverwrite);
  try
    vSource := TsgcChartDataSource_CSV.Create(nil);
    try
      vSource.Buffer := vBuffer;
      vSource.HasHeader := True;
      vSource.Mapping.XPath := 'time';
      vSource.Mapping.YPath := 'value';
      vSource.LoadFromFile('sensor.csv');
    finally
```

```
      vSource.Free;
    end;
  finally
    vBuffer.Free;
  end;
end;
```

Voir aussi

- TsgcChartDataSource
- TsgcChartBuffer

TsgcChartDataSource_Table

Unité : sgcCharts_DataSource_Table
Charge les données du graphique depuis toute table qui implémente l'interface `IsgcUITable` de sgcUICore (unité `sgcUICore_Table`), par exemple le modèle de données d'une grille sgcGrid, sans dépendance entre les deux produits. Le composant dérive de TsgcChartDataSource et hérite de **Buffer**, **Series**, **Active**, **Throttle** ainsi que des événements avant ajout et de changement de données ; il n'utilise pas les chemins **Mapping** hérités, car les colonnes sont nommées directement.
XColumn et **YColumn** nomment les colonnes, comparées sans tenir compte de la casse aux noms de colonnes de la table (pour sgcGrid, le **FieldName** de la colonne). Un **XColumn** vide fait de X l'index de ligne base 0. Une valeur date/heure donne son `TDateTime` comme X, une valeur numérique est convertie par **Mapping.TimeFormat** et une valeur texte est analysée comme du texte CSV. Une ligne dont X ou Y est vide ou non convertible ne donne aucun point. Les lignes sont lues dans l'ordre de la table, qui pour sgcGrid est la vue triée et filtrée.
La source enregistre un écouteur sur la table. Chaque modification de la table recharge toute la table lorsque **Active** vaut True et que la version de la table a changé depuis le dernier chargement. Le rechargement s'exécute dans le thread qui a modifié la table, puis la notification du graphique est délivrée comme pour les autres sources de données. Une table liée avec **SetTable** doit être déliée avec `SetTable(nil)` avant d'être libérée ; un composant **Table** est délié automatiquement.

Propriétés

Propriété	Type	Description
Table	TComponent	Un composant qui implémente `IsgcUITable`. Affecter un composant qui ne l'implémente pas déclenche une exception.
XColumn	string	Colonne contenant X. Vide signifie l'index de ligne base 0.
YColumn	string	Colonne contenant Y.
LabelColumn	string	Colonne de texte d'élément d'une série de catégories liée, ou plusieurs colonnes séparées par `;`, dont les textes sont joints avec le **PathDelimiter** d'un treemap (de sorte qu'une table avec des colonnes Region, Country et City alimente un treemap à trois niveaux) et avec `/` sinon ; les parties finales vides sont ignorées. Pour une carte de chaleur, les étiquettes des lignes.

Méthodes

Méthode	Description
SetTable	Lie tout objet qui implémente `IsgcUITable`, par exemple le modèle **Data** d'une grille sgcGrid. nil délie.
GetTable	Renvoie la table liée, nil si aucune n'est liée.
LoadFromTable	Ajoute toutes les lignes de la table, en conservant le verrou de la table pendant la lecture, et envoie une notification à la fin.
Refresh	Vide les tampons et recharge la table. Ne fait rien si aucune table n'est liée.

Séries de catégories et cartes de chaleur

Avec une série de catégories, chaque ligne appelle **SetValue**(étiquette, Y), et une ligne dont l'étiquette est vide n'écrit aucun élément ; sans **LabelColumn**, chaque ligne définit l'élément à son X (l'index de ligne lorsque **XColumn** est vide). Un rechargement s'exécute entre **BeginReload** et **EndReload** de la série, de sorte que les éléments conservent leurs couleurs et leurs indicateurs et que les éléments des lignes supprimées disparaissent. Avec une carte de chaleur, chaque chargement remplace la matrice par une ligne par ligne de table (la clé est X lorsque **XColumn** est défini, sinon l'index de ligne). Les cellules sont les colonnes de **YColumn** séparées par ; ou, lorsque **YColumn** est vide, toutes les colonnes numériques sauf **XColumn** et les colonnes d'étiquette ; les étiquettes des cellules sont les noms de colonnes. C'est la voie à suivre pour un résultat de tableau croisé ou une grille.

Exemple : un camembert alimenté depuis une table

```
uses
  Classes, SysUtils, Variants,
  sgcUICore_Table, sgcCharts_Chart, sgcCharts_DataSource_Table,
  sgcCharts_Series_Category, sgcCharts_Series_Pie;

type
  { a small table with two columns, Region (text) and Sales (number); a
    live table also keeps its listeners and calls them after a change }
  TSalesTable = class(TComponent, IsgcUITable)
  private
    FRegions: array of string;
    FSales: array of Double;
  public
    procedure AddRow(const aRegion: string; aSales: Double);
    // IsgcUITable
    function TableRowCount: Integer;
    function TableColCount: Integer;
    function TableColName(aCol: Integer): string;
    function TableValue(aRow, aCol: Integer): Variant;
    function TableText(aRow, aCol: Integer): string;
    procedure TableLock;
    procedure TableUnlock;
    function TableVersion: Cardinal;
    procedure TableAddListener(aListener: TNotifyEvent);
    procedure TableRemoveListener(aListener: TNotifyEvent);
  end;

procedure TSalesTable.AddRow(const aRegion: string; aSales: Double);
begin
  SetLength(FRegions, Length(FRegions) + 1);
  SetLength(FSales, Length(FSales) + 1);
  FRegions[High(FRegions)] := aRegion;
  FSales[High(FSales)] := aSales;
end;

function TSalesTable.TableRowCount: Integer;
begin
  Result := Length(FRegions);
end;

function TSalesTable.TableColCount: Integer;
begin
  Result := 2;
end;

function TSalesTable.TableColName(aCol: Integer): string;
begin
  if aCol = 0 then
    Result := 'Region'
  else
    Result := 'Sales';
```

```
end;

function TSalesTable.TableValue(aRow, aCol: Integer): Variant;
begin
  if aCol = 0 then
    Result := FRegions[aRow]
  else
    Result := FSales[aRow];
end;

function TSalesTable.TableText(aRow, aCol: Integer): string;
begin
  if aCol = 0 then
    Result := FRegions[aRow]
  else
    Result := FloatToStr(FSales[aRow]);
end;

procedure TSalesTable.TableLock;
begin
end;

procedure TSalesTable.TableUnlock;
begin
end;

function TSalesTable.TableVersion: Cardinal;
begin
  Result := Length(FRegions);
end;

procedure TSalesTable.TableAddListener(aListener: TNotifyEvent);
begin
end;

procedure TSalesTable.TableRemoveListener(aListener: TNotifyEvent);
begin
end;

var
  vTable: TSalesTable;
  vPie: TsgcChartSeries_Pie;
  vSource: TsgcChartDataSource_Table;
  i: Integer;
begin
  vTable := TSalesTable.Create(nil);
  vPie := TsgcChartSeries_Pie.Create(nil);
  vSource := TsgcChartDataSource_Table.Create(nil);
  try
    vTable.AddRow('North', 420);
    vTable.AddRow('South', 310);
    vTable.AddRow('East', 275);

    vSource.Series := vPie;
    vSource.LabelColumn := 'Region'; // the item texts
    vSource.YColumn := 'Sales'; // the values
    vSource.Table := vTable;
    vSource.Refresh;

    for i := 0 to vPie.ItemCount - 1 do
      Writeln(vPie.Slices[i].Text, ' = ', FormatFloat('0', vPie.Slices[i].Value));
  finally
    vSource.Free;
    vPie.Free;
    vTable.Free;
```

```
  end;
end.
```

Voir aussi

- TsgcChartDataSource
- TsgcChartSeries_Pie
- TsgcChartSeries_Heatmap

Présentation de la passerelle

Le package `sgcChartsWS` ajoute des sources de données de graphique alimentées par une connexion sgcWebSockets en direct. Il s'agit d'un package distinct du package principal `sgcCharts` et il nécessite sgcWebSockets pour être compilé ; un projet qui utilise uniquement les sources de données principales (TsgcChartDataSource et ses descendants manuel, dataset, JSON et CSV) n'a pas du tout besoin de sgcWebSockets.

Composants

- TsgcChartDataSource_MQTT : trace les publications MQTT reçues par un `TsgcWSPClient_MQTT`.
- TsgcChartDataSource_WebSocket : trace les messages JSON reçus par un `TsgcWebSocketClient`.
- TsgcChartDataSource_SSE : trace les événements Server-Sent Events reçus par un `TsgcHTTP1Client`.
- TsgcChartDataSource_Binance : s'abonne à un flux de données de marché public Binance et le trace via un `TsgcWSAPI_Binance`.

Chaînage des gestionnaires

Chaque source de données passerelle fonctionne de la même façon : tant qu'elle est **Active** et que sa propriété client est assignée, elle installe son propre gestionnaire d'événement sur le client (`OnMQTTPublish`, `OnMessage`, `OnSSEMessage`, ou l'événement de message du client Binance) et conserve le gestionnaire qui était déjà présent. Chaque message entrant est d'abord tracé, puis transmis à ce gestionnaire précédent, de sorte que le gestionnaire propre à l'application continue de se déclencher exactement comme si la source de données de graphique n'était pas présente.
Définir **Active** sur False, vider la propriété client, ou détruire la source de données restaure le gestionnaire qui était présent auparavant, sauf si l'application a remplacé le gestionnaire entre-temps. Cette accroche ne se produit qu'à l'exécution, jamais à la conception, de sorte qu'une application qui assigne son propre gestionnaire après l'attachement de la source de données doit l'assigner avant de définir la propriété client (ou avant de basculer **Active**) pour conserver les deux gestionnaires.
Les messages arrivent dans le thread depuis lequel le client sous-jacent notifie. Les ajouts passent par le verrou et la notification limitée en fréquence de la classe de base TsgcChartDataSource, si bien que tout mode de notification peut être utilisé sans risque avec une source de données passerelle.

TsgcChartDataSource_MQTT

Unité : sgcChartsWS_DataSource_MQTT
Source de données de graphique alimentée par un client MQTT de sgcWebSockets. Le composant descend de TsgcChartDataSource et hérite de **Buffer**, **OHLCBuffer**, **Mapping**, **Throttle** ainsi que des événements avant-ajout et de changement de données. Voir Présentation de la passerelle pour le fonctionnement du chaînage des gestionnaires.
TopicFilter sélectionne les publications qui sont tracées, avec les caractères génériques MQTT `+` (un niveau) et `#` (ce niveau et tous les niveaux inférieurs) ; un **TopicFilter** vide trace chaque publication.
PayloadFormat `cpfJSON` lit la charge utile via **Mapping** (un objet JSON ou un tableau d'enregistrements) ; `cpfNumber` analyse la charge utile comme un nombre (Y, avec `.` comme séparateur décimal quel que soit la locale) et utilise `Now` comme X.

Propriétés

Propriété	Type	Par défaut	Description
Active	Boolean	True	Installe ou supprime l'accroche sur **Client**.
Client	TsgcWSPClient_MQTT	nil	Le client MQTT dont `OnMQTTPublish` est accroché.
PayloadFormat	TsgcChartMQTTPayloadFormat	cpfJSON	`cpfJSON` ou `cpfNumber` . Voir ci-dessus.
TopicFilter	string	''	Filtre de sujet MQTT (`+` , `#`). Vide, il trace chaque publication.

Méthodes

Méthode	Retour	Description
ProcessPublish(const aTopic, aText: string)	Boolean	Trace une publication comme si elle provenait de **Client**. Renvoie False lorsque le sujet ne correspond pas à **TopicFilter** ou que la charge utile ne peut pas être lue.

Exemple : traçage de relevés de capteurs depuis MQTT

La propriété **Client** reçoit le composant de protocole `TsgcWSPClient_MQTT`, qui est lui-même attaché au client de transport (ici un `TsgcWebSocketClient`) via sa propre propriété **Client**, exactement comme tout client MQTT sgcWebSockets est câblé.

```
var
  vBuffer: TsgcChartBuffer;
  vSource: TsgcChartDataSource_MQTT;
  vWSClient: TsgcWebSocketClient;
  vMQTT: TsgcWSPClient_MQTT;
begin
  vBuffer := TsgcChartBuffer.Create(4096, cbmOverwrite);
  try
    vWSClient := TsgcWebSocketClient.Create(nil);
    try
      vMQTT := TsgcWSPClient_MQTT.Create(nil);
      try
```

```
        vMQTT.Client := vWSClient;
        vWSClient.Host := 'broker.example.com';
        vWSClient.Port := 15675;

        vSource := TsgcChartDataSource_MQTT.Create(nil);
        try
          vSource.Buffer := vBuffer;
          vSource.Client := vMQTT;
          vSource.TopicFilter := 'sensors/+/temperature';
          vSource.PayloadFormat := cpfJSON;
          vSource.Mapping.XPath := 'ts';
          vSource.Mapping.TimeFormat := ctfUnixSeconds;
          vSource.Mapping.YPath := 'value';

          vWSClient.Active := True;
          vMQTT.Subscribe('sensors/+/temperature');
        finally
          vSource.Free;
        end;
      finally
        vMQTT.Free;
      end;
    finally
      vWSClient.Free;
    end;
  finally
    vBuffer.Free;
  end;
end;
```

Voir aussi

- Présentation de la passerelle
- TsgcChartDataSource

TsgcChartDataSource_WebSocket

Unité : sgcChartsWS_DataSource_WebSocket
Source de données de graphique alimentée par un client WebSocket de sgcWebSockets. Le composant descend de TsgcChartDataSource et hérite de **Buffer**, **OHLCBuffer**, **Mapping**, **Throttle** ainsi que des événements avant-ajout et de changement de données. Voir Présentation de la passerelle pour le fonctionnement du chaînage des gestionnaires.
Chaque message texte reçu par le client est traité comme du JSON et lu via **Mapping**. **MessageFilter** est un chemin optionnel qui doit exister dans le message pour qu'il soit tracé ; laissez-le vide pour tracer chaque message.

Propriétés

Propriété	Type	Par défaut	Description
Active	Boolean	True	Installe ou supprime l'accroche sur **Client**.
Client	TsgcWebSocketClient	nil	Le client WebSocket dont `OnMessage` est accroché.
MessageFilter	string	''	Un chemin qui doit exister dans le message pour qu'il soit tracé. Vide, il trace chaque message.

Méthodes

Méthode	Retour	Description
ProcessMessage(const aText: string)	Boolean	Trace un message comme s'il provenait de **Client**. Renvoie False lorsque le message n'est pas du JSON valide ou que **MessageFilter** n'y existe pas.

Exemple : traçage d'un flux JSON via WebSocket

```
var
  vBuffer: TsgcChartBuffer;
  vSource: TsgcChartDataSource_WebSocket;
  vClient: TsgcWebSocketClient;
begin
  vBuffer := TsgcChartBuffer.Create(4096, cbmOverwrite);
  try
    vClient := TsgcWebSocketClient.Create(nil);
    try
      vClient.URL := 'wss://example.com/feed';

      vSource := TsgcChartDataSource_WebSocket.Create(nil);
      try
        vSource.Buffer := vBuffer;
        vSource.Client := vClient;
        vSource.Mapping.XPath := 'ts';
        vSource.Mapping.TimeFormat := ctfUnixMilliseconds;
        vSource.Mapping.YPath := 'price';

        vClient.Active := True;
      finally
        vSource.Free;
      end;
```

```
    finally
      vClient.Free;
    end;
  finally
    vBuffer.Free;
  end;
end;
```

Voir aussi

- Présentation de la passerelle
- TsgcChartDataSource

TsgcChartDataSource_SSE

Unité : sgcChartsWS_DataSource_SSE
Source de données de graphique alimentée par le flux Server-Sent Events d'un client HTTP sgcWebSockets. Le composant descend de TsgcChartDataSource et hérite de **Buffer**, **OHLCBuffer**, **Mapping**, **Throttle** ainsi que des événements avant-ajout et de changement de données. Voir Présentation de la passerelle pour le fonctionnement du chaînage des gestionnaires.
Les données de chaque événement SSE sont lues comme du JSON via **Mapping**. Un événement scindé sur plusieurs segments de flux est mis en mémoire tampon en interne jusqu'à ce qu'il soit complet.

Propriétés

Propriété	Type	Par défaut	Description
Active	Boolean	True	Installe ou supprime l'accroche sur **Client**.
Client	TsgcHTTP1Client	nil	Le client HTTP dont `OnSSEMessage` est accroché.

Méthodes

Méthode	Retour	Description
ProcessChunk(const aText: string)	Integer	Trace les événements complets d'un segment de flux, comme s'ils provenaient de **Client**. Un événement incomplet attend le segment suivant. Retourne le nombre d'événements dont les données étaient du JSON valide.
ResetStream	—	Supprime un événement incomplet mis en mémoire tampon.

Exemple : traçage d'un flux SSE

```
var
  vBuffer: TsgcChartBuffer;
  vSource: TsgcChartDataSource_SSE;
  vClient: TsgcHTTP1Client;
begin
  vBuffer := TsgcChartBuffer.Create(4096, cbmOverwrite);
  try
    vClient := TsgcHTTP1Client.Create(nil);
    try
      vSource := TsgcChartDataSource_SSE.Create(nil);
      try
        vSource.Buffer := vBuffer;
        vSource.Client := vClient;
        vSource.Mapping.XPath := 'ts';
        vSource.Mapping.TimeFormat := ctfUnixSeconds;
        vSource.Mapping.YPath := 'value';

        vClient.GetSSE('https://example.com/stream');
      finally
        vSource.Free;
      end;
    finally
      vClient.Free;
```

```
    end;
  finally
    vBuffer.Free;
  end;
end;
```

Voir aussi

- Présentation de la passerelle
- TsgcChartDataSource

TsgcChartDataSource_Binance

Unité : sgcChartsWS_DataSource_Binance
Source de données de graphique alimentée par le client API Binance de sgcWebSockets. Le composant descend de TsgcChartDataSource et hérite de **Buffer**, **OHLCBuffer**, **Throttle** ainsi que des événements avant-ajout et de changement de données (il n'utilise pas la propriété héritée **Mapping**, car la forme des messages Binance est fixe pour chaque flux). Voir Présentation de la passerelle pour le fonctionnement du chaînage des gestionnaires.
Stream sélectionne ce qui est tracé : `cbsKLine` trace des chandeliers (dans **OHLCBuffer**) construits à partir des événements kline/chandelier pour **Symbol** et **Interval** ; `cbsTicker` et `cbsTrade` tracent une valeur (dans **Buffer**) à partir du flux ticker ou trade correspondant pour **Symbol**. **ClosedOnly**, lorsque `cbsKLine` est utilisé, ne trace que les klines dont le chandelier est clôturé plutôt que chaque mise à jour en cours.
Avec `cbsTrade`, **OnTrade** signale chaque transaction de **Symbol** avec son horodatage, son prix et sa quantité (le champ `q`) avant qu'elle ne soit tracée. Dans l'édition financière, assignez un TsgcChartAggregator à **Aggregator** pour construire des chandeliers à partir des transactions : chaque transaction est alors transmise à `Aggregator.AddTrade(time, price, quantity)` au lieu de **Buffer**, de sorte que les chandeliers de la série de l'agrégateur portent le volume échangé, les transactions elles-mêmes ne sont pas tracées sous forme de ligne et `OnBeforeAppend` ne se déclenche pas pour elles. Avec **Aggregator** à nil, les transactions sont tracées dans **Buffer** comme auparavant.

Propriétés

Propriété	Type	Par défaut	Description
Active	Boolean	True	Installe ou supprime l'accroche sur le client de **API**.
Aggregator	TsgcChartAggregator	nil	Édition financière. Avec `Stream = cbsTrade`, chaque transaction est transmise à `Aggregator.AddTrade` avec son horodatage, son prix et sa quantité au lieu d'être ajoutée à **Buffer**. Réinitialisé à nil quand l'agrégateur est libéré.
API	TsgcWSAPI_Binance	nil	Le composant API Binance dont les messages du client sont accrochés.
ClosedOnly	Boolean	False	Avec `Stream = cbsKLine`, ne trace que les chandeliers clôturés.
Interval	TsgcWSBinanceChartIntervals	bci1m	Intervalle de la kline, utilisé quand `Stream = cbsKLine`.
Stream	TsgcChartBinanceStream	cbsKLine	`cbsKLine`, `cbsTicker` ou `cbsTrade`.
Symbol	string	''	Le symbole Binance à tracer, par exemple `BTCUSDT`.

Méthodes

Méthode	Retour	Description
Subscribe	Integer	S'abonne à **Stream** pour **Symbol** (et **Interval** pour les klines). Retourne l'identifiant renvoyé par l'API.
Unsubscribe	Integer	Se désabonne du même flux. Retourne l'identifiant renvoyé par l'API.

ProcessMessage(const aText: string)	Boolean	Trace un message Binance comme s'il provenait de **API.Client**. Renvoie False lorsqu'il ne s'agit pas d'un message correspondant à **Stream**/**Symbol**/**Interval**, ou que rien n'a été tracé.

Événements

Événement	Description
OnTrade(Sender: TObject; aTime: TDateTime; aPrice, aQty: Double)	Public. Avec `Stream = cbsTrade`, se déclenche pour chaque transaction de **Symbol** tant que la source est active, avant que la transaction ne soit ajoutée ou agrégée : aTime est l'horodatage de la transaction (UTC), aPrice le prix et aQty la quantité (0 lorsque le message ne comporte pas de `q`).

Exemple : traçage des klines Binance en direct

```
var
  vOHLC: TsgcChartOHLCBuffer;
  vSource: TsgcChartDataSource_Binance;
  vAPI: TsgcWSAPI_Binance;
begin
  vOHLC := TsgcChartOHLCBuffer.Create(2048, cbmOverwrite);
  try
    vAPI := TsgcWSAPI_Binance.Create(nil);
    try
      vSource := TsgcChartDataSource_Binance.Create(nil);
      try
        vSource.OHLCBuffer := vOHLC;
        vSource.API := vAPI;
        vSource.Stream := cbsKLine;
        vSource.Symbol := 'BTCUSDT';
        vSource.Interval := bci1m;
        vSource.ClosedOnly := False;

        vSource.Subscribe;
      finally
        vSource.Free;
      end;
    finally
      vAPI.Free;
    end;
  finally
    vOHLC.Free;
  end;
end;
```

Voir aussi

- Présentation de la passerelle
- TsgcChartDataSource

TsgcChartSeries_Indicator

Unité : sgcCharts_Series_Financial
Éditions Registered et Trial uniquement (exclu de l'édition Community ; voir Éditions).
TsgcChartSeries_Indicator est la base abstraite de chaque série d'indicateur financier : un descendant de TsgcChartSeries qui lit les barres d'une autre série, sa **Source**, une série chandelier ou OHLC ou une série X/Y dont le Y tient lieu d'open, high, low et close sans volume, et conserve ses propres sorties calculées dans des tableaux internes au lieu de dessiner elle-même les données sources. Les indicateurs concrets, TsgcChartSeries_SMA, TsgcChartSeries_EMA, TsgcChartSeries_WMA, TsgcChartSeries_Bollinger et TsgcChartSeries_VWAP, se tracent sur le même axe que le prix, et sont donc généralement ajoutés par-dessus une série chandelier. La famille des oscillateurs (TsgcChartSeries_Oscillator et ses descendants RSI et Stochastic), MACD, ATR et TsgcChartSeries_Volume descendent également de TsgcChartSeries_Indicator mais sont documentés dans leurs propres sujets car ils sont normalement dessinés dans un panneau séparé, en dessous du prix.

Propriétés

Propriété	Type	Description
Source	TsgcChart-Series	La série que cet indicateur suit : une série chandelier / OHLC, ou une série X/Y (son Y est utilisé comme open, high, low et close, sans volume). La modifier efface les sorties et force un calcul complet au prochain **Sync**.
LineWidth	Double	Pixels d'appareil à l'échelle 1. Par défaut 1.5.
OutputCount	Integer (lecture seule)	Nombre de lignes de sortie que l'indicateur calcule par barre, 1 pour un indicateur à une seule ligne comme SMA, plus pour un indicateur à plusieurs lignes comme Bollinger.
PointCount	Integer (lecture seule)	Nombre de barres traitées, 0 étant la plus ancienne.
XValues[aIndex: Integer]	Double (lecture seule)	Le X (généralement l'heure de la barre) de la barre traitée à aIndex.
Values[aLine, aIndex: Integer]	Double (lecture seule)	Sortie aLine (base 0, jusqu'à **OutputCount** - 1) de la barre traitée à aIndex. NaN pendant la période de chauffe de l'indicateur, lorsque pas assez de barres n'ont encore été vues pour produire une valeur.

Méthodes

Méthode	Retour	Description
Sync		Aligne les sorties sur **Source**. Voir Synchronisation avec la source.
Recalculate		Force un calcul complet au prochain **Sync**, par exemple après avoir réécrit sur place d'anciennes barres de **Source**.
GetDataRange(var aXMin, aXMax, aYMin, aYMax: Double)	Boolean	Appelle d'abord **Sync**, puis retourne les bornes X et Y des sorties ; False lorsqu'il n'y en a aucune.

Draw(aCanvas: TsgcUICanvas; aXAxis, aYAxis: TsgcChartAxis; const aPlot: TsgcUIRect; aScale: Double)		Peint chaque sortie sous forme de ligne de **Effective-Color**. Un descendant comme Bollinger la redéfinit pour dessiner des bandes ou des barres à la place.
DrawLegendSymbol(aCanvas: TsgcUI-Canvas; const aRect: TsgcUIRect; aS-cale: Double)		Peint le symbole de légende de cet indicateur.

Synchronisation avec la source

Un indicateur ne copie jamais les données de **Source** ; il lit ses barres et ne conserve que ses propres sorties. **Sync** est ce qui réaligne ces sorties sur **Source**, et elle est appelée automatiquement par **GetDataRange** (donc à chaque rendu) et par le test d'impact, tout en restant disponible pour un appel direct. Ce qu'elle fait dépend de ce qui a changé depuis le dernier appel :

- Aucune sortie encore, un changement de paramètre (comme définir **Period**), ou une source effacée ou re-chargée (moins de barres qu'auparavant traitées, ou la barre traitée la plus récente a désormais un X diffé-rent) : un calcul complet, réinitialisant l'indicateur et lui refournissant chaque barre, avec le même résultat qu'un calcul depuis zéro.
- Barres ajoutées à **Source** depuis le dernier Sync : seules les nouvelles barres sont fournies.
- La barre traitée la plus récente a changé sur place, le cas du chandelier en direct (voir **TsgcChartSeries_OHLCBase.UpdateLastOHLC**) : seule cette dernière sortie est recalculée.
- Barres supprimées en tête d'un tampon circulaire plein (**BufferMode** = `cbmOverwrite` sur la source) : les sor-ties des barres supprimées sont abandonnées et le calcul se poursuit là où il en était, si bien que les sorties restantes conservent l'historique avec lequel elles ont été calculées.

Sync détecte le cas applicable en comparant le nombre de barres de **Source**, le X de sa barre la plus ancienne et de la barre traitée la plus récente, ainsi que les valeurs OHLCV de cette barre la plus récente. Appelez **Recalcu-late** après avoir réécrit sur place d'anciennes barres de **Source**, car Sync ne peut pas détecter ce cas par elle-même. Les sorties encore dans leur période de chauffe, avant que l'indicateur n'ait vu assez de barres pour pro-duire une valeur, se relisent comme NaN via **Values** et ne sont pas dessinées. Tout s'exécute dans le thread princi-pal ; le tampon source est lu sous son propre verrou.

TsgcChartSeries_SMA

Moyenne mobile simple du close. Une ligne de sortie, index 0.

Propriété	Type	Description
Period	Integer	Nombre de barres moyennées. Par défaut 20.

TsgcChartSeries_EMA

Moyenne mobile exponentielle du close, pondérant les barres récentes plus fortement qu'une moyenne simple. Une ligne de sortie, index 0.

Propriété	Type	Description
Period	Integer	Nombre de barres à partir duquel le facteur de lissage est dérivé. Par défaut 20.

TsgcChartSeries_WMA

Moyenne mobile pondérée linéairement du close, accordant le poids le plus élevé à la barre la plus récente de la période et le plus faible à la plus ancienne. Une ligne de sortie, index 0.

Propriété	Type	Description

Period	Integer	Nombre de barres pondérées. Par défaut 20.

TsgcChartSeries_Bollinger

Bandes de Bollinger : une moyenne mobile du close avec une bande supérieure et une bande inférieure à un nombre choisi d'écarts-types de celle-ci. Comme les deux bandes suivent la même échelle de prix que la source, Bollinger se trace sur le prix comme les moyennes mobiles, et non dans son propre panneau. Trois lignes de sortie : 0 est la bande médiane, 1 la bande supérieure, 2 la bande inférieure.

Propriété	Type	Description
Period	Integer	Nombre de barres sur lequel la bande médiane et l'écart-type sont calculés. Par défaut 20.
Deviations	Double	Nombre d'écarts-types séparant les bandes supérieure et inférieure de la bande médiane. Par défaut 2.
BandFill	Boolean	Remplit la zone entre la bande supérieure et la bande inférieure. Par défaut True.

TsgcChartSeries_VWAP

Prix moyen pondéré par le volume. Elle nécessite une source qui porte du volume, une série OHLC ou chandelier alimentée via **AddOHLC** avec un volume non nul ; une source X/Y n'a pas de volume et effectue une moyenne comme si chaque barre pesait le même poids. Le X de la source est pris comme heure de la barre. Une ligne de sortie, index 0.

Propriété	Type	Description
SessionStart	TDateTime	Fraction de jour à laquelle une session commence, 0 (valeur par défaut) signifiant minuit ; le calcul se réinitialise à chaque limite de session.

Exemple : SMA, EMA et bandes de Bollinger sur une série chandelier

```
uses
  SysUtils, Math,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Series_Financial,
  sgcCharts_Axis, sgcCharts_Types, sgcUICore_Canvas, sgcUICore_Canvas_Raster;

var
  vChart: TsgcChart;
  vAxisX, vAxisY: TsgcChartAxis;
  vPrice: TsgcChartSeries_Candlestick;
  vSMA: TsgcChartSeries_SMA;
  vBollinger: TsgcChartSeries_Bollinger;
  vRaster: TsgcUICanvas_Raster;
  i: Integer;
  vClose, vOpen: Double;
begin
  vChart := TsgcChart.Create(nil);
  try
    vAxisX := vChart.Axes.Add;
    vAxisX.Name := 'bottom';
    vAxisX.Position := capBottom;
    vAxisY := vChart.Axes.Add;
    vAxisY.Name := 'left';
    vAxisY.Position := capLeft;
```

```
    vPrice := TsgcChartSeries_Candlestick.Create(nil);
    vSMA := TsgcChartSeries_SMA.Create(nil);
    vBollinger := TsgcChartSeries_Bollinger.Create(nil);
    try
      vPrice.Chart := vChart;
      vPrice.AxisX := 'bottom';
      vPrice.AxisY := 'left';

      vClose := 100;
      for i := 0 to 79 do
      begin
        vOpen := vClose;
        vClose := vOpen + Sin(i / 6) * 2 + (Random - 0.5);
        vPrice.AddOHLC(i, vOpen, Max(vOpen, vClose) + 1,
          Min(vOpen, vClose) - 1, vClose, 1000 + Random(500));
      end;

      vSMA.Chart := vChart;
      vSMA.AxisX := 'bottom';
      vSMA.AxisY := 'left';
      vSMA.Source := vPrice;
      vSMA.Period := 20;

      vBollinger.Chart := vChart;
      vBollinger.AxisX := 'bottom';
      vBollinger.AxisY := 'left';
      vBollinger.Source := vPrice;
      vBollinger.Period := 20;
      vBollinger.Deviations := 2;

      vSMA.Sync;
      vBollinger.Sync;
      Writeln('SMA last value: ',
        vSMA.Values[0, vSMA.PointCount - 1]: 0: 4);
      Writeln('Bollinger upper/middle/lower: ',
        vBollinger.Values[1, vBollinger.PointCount - 1]: 0: 4, ' / ',
        vBollinger.Values[0, vBollinger.PointCount - 1]: 0: 4, ' / ',
        vBollinger.Values[2, vBollinger.PointCount - 1]: 0: 4);

      vRaster := TsgcUICanvas_Raster.Create(800, 400);
      try
        vChart.Render(vRaster, sgcUIRect(0, 0, 800, 400));
        vRaster.SaveToPNGFile('indicator.png');
      finally
        vRaster.Free;
      end;
    finally
      vBollinger.Free;
      vSMA.Free;
      vPrice.Free;
    end;
  finally
    vChart.Free;
  end;
end;
```

Voir aussi

- TsgcChartSeries_Oscillator
- TsgcChartSeries_Volume
- TsgcChartAggregator
- TsgcChartDrawing
- Panneaux
- TsgcChartSeries_Candlestick

- TsgcChartSeries
- Éditions

TsgcChartSeries_Oscillator

Unité : sgcCharts_Series_Financial
Éditions Registered et Trial uniquement (exclu de l'édition Community ; voir Éditions).
TsgcChartSeries_Oscillator est la base des indicateurs de type 0..100 : un descendant de TsgcChartSeries_Indicator qui fixe ses bornes Y à 0..100 quelles que soient les données et qui peut dessiner une paire de lignes guides en pointillés de surachat / survente. TsgcChartSeries_RSI et TsgcChartSeries_Stochastic en descendent. **TsgcChartSeries_MACD** et **TsgcChartSeries_ATR** sont documentés plus loin sur cette même page car ils sont normalement dessinés eux aussi dans leur propre panneau, mais ils descendent directement de TsgcChartSeries_Indicator, pas de TsgcChartSeries_Oscillator : leur sortie est une valeur dérivée du prix sans plafond naturel de 0..100, ils conservent donc une plage Y flottante au lieu d'une plage fixe. Les quatre indicateurs sont normalement placés dans un panneau qui leur est propre, en dessous du panneau de prix, car leur échelle ne correspond pas au prix.

Propriétés

Propriété	Type	Description
GuideLines	Boolean	Dessine les lignes guides de surachat et de survente comme des lignes horizontales en pointillés. Par défaut True.

Overbought et Oversold ne sont pas publiées sur la classe de base elle-même ; chaque oscillateur concret les publie avec son propre niveau par défaut, car RSI et Stochastic ne s'accordent pas sur l'emplacement de ces niveaux. Voir TsgcChartSeries_RSI et TsgcChartSeries_Stochastic ci-dessous.
Chaque membre de TsgcChartSeries_Indicator, y compris **Source**, les propriétés de type **Period** sur les classes concrètes, **Sync**, **PointCount** et **Values**, s'applique toujours ici ; voir Synchronisation avec la source pour la façon dont un indicateur se maintient aligné sur sa série **Source**.

TsgcChartSeries_RSI

Indice de force relative du close. Une ligne de sortie, index 0. Les lignes guides sont par défaut à 30 et 70.

Propriété	Type	Description
Period	Integer	Nombre de barres sur lequel le gain et la perte moyens sont calculés. Par défaut 14.
Overbought	Double	Niveau de la ligne guide. Par défaut 70.
Oversold	Double	Niveau de la ligne guide. Par défaut 30.

TsgcChartSeries_Stochastic

Oscillateur stochastique. Deux lignes de sortie : 0 est %K, 1 est %D (une moyenne mobile de %K). Les lignes guides sont par défaut à 20 et 80.

Propriété	Type	Description
KPeriod	Integer	Période rétrospective pour %K. Par défaut 14.
KSmoothing	Integer	Période de lissage appliquée à %K. Par défaut 3.

DPeriod	Integer	Période de la moyenne mobile de %K qui produit %D. Par défaut 3.
DColor	TsgcUI-Color	Couleur de la ligne %D ; %K est dessinée en **EffectiveColor** comme un indicateur à une seule ligne. Par défaut une couleur d'accent orange.
Over-bought	Double	Niveau de la ligne guide. Par défaut 80.
Oversold	Double	Niveau de la ligne guide. Par défaut 20.

TsgcChartSeries_MACD

Convergence / divergence des moyennes mobiles du close. Descend directement de TsgcChartSeries_Indicator : son échelle suit les différences de prix à partir desquelles elle est construite, elle n'a donc pas de plage fixe 0..100 ni de lignes guides de surachat / survente. Elle redéfinit toutefois la plage Y afin que la ligne de base zéro que l'histogramme chevauche reste toujours visible. Trois lignes de sortie : 0 est la ligne MACD, 1 la ligne de signal, 2 l'histogramme (ligne MACD moins ligne de signal).

Propriété	Type	Description
FastPeriod	Integer	Période de l'EMA rapide. Par défaut 12.
SlowPeriod	Integer	Période de l'EMA lente. Par défaut 26.
SignalPe-riod	Integer	Période de l'EMA de la ligne MACD qui produit la ligne de signal. Par défaut 9.
SignalColor	TsgcUI-Color	Couleur de la ligne de signal ; la ligne MACD utilise **EffectiveColor** et l'histogramme est dessiné sous forme de barres de hausse / baisse. Par défaut une couleur d'accent orange.

TsgcChartSeries_ATR

Amplitude vraie moyenne, une mesure de volatilité. Descend directement de TsgcChartSeries_Indicator pour la même raison que MACD : sa valeur est une plage de prix positive non bornée, pas un ratio 0..100, elle conserve donc une plage Y flottante et n'a pas de lignes guides. Une ligne de sortie, index 0.

Propriété	Type	Description
Period	Integer	Nombre de barres moyennées. Par défaut 14.

Exemple : RSI et MACD dans leurs propres panneaux

```
uses
  SysUtils, Math,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Series_Financial,
  sgcCharts_Axis, sgcCharts_Types, sgcUICore_Canvas, sgcUICore_Canvas_Raster;

var
  vChart: TsgcChart;
  vAxisX, vAxisY: TsgcChartAxis;
  vPrice: TsgcChartSeries_Candlestick;
  vRSI: TsgcChartSeries_RSI;
  vMACD: TsgcChartSeries_MACD;
  vRaster: TsgcUICanvas_Raster;
  i: Integer;
  vClose, vOpen: Double;
begin
```

```
  vChart := TsgcChart.Create(nil);
  try
    vAxisX := vChart.Axes.Add;
    vAxisX.Name := 'bottom';
    vAxisX.Position := capBottom;
    vAxisY := vChart.Axes.Add;
    vAxisY.Name := 'left';
    vAxisY.Position := capLeft;

    vPrice := TsgcChartSeries_Candlestick.Create(nil);
    vRSI := TsgcChartSeries_RSI.Create(nil);
    vMACD := TsgcChartSeries_MACD.Create(nil);
    try
      vPrice.Chart := vChart;
      vPrice.AxisX := 'bottom';
      vPrice.AxisY := 'left';

      vClose := 100;
      for i := 0 to 79 do
      begin
        vOpen := vClose;
        vClose := vOpen + Sin(i / 6) * 2 + (Random - 0.5);
        vPrice.AddOHLC(i, vOpen, Max(vOpen, vClose) + 1,
          Min(vOpen, vClose) - 1, vClose, 1000 + Random(500));
      end;

      vRSI.Chart := vChart;
      vRSI.AxisX := 'bottom';
      vRSI.AxisY := 'left';
      vRSI.Source := vPrice;
      vRSI.Period := 14;

      vMACD.Chart := vChart;
      vMACD.AxisX := 'bottom';
      vMACD.AxisY := 'left';
      vMACD.Source := vPrice;
      vMACD.FastPeriod := 12;
      vMACD.SlowPeriod := 26;
      vMACD.SignalPeriod := 9;

      vRSI.Sync;
      vMACD.Sync;
      Writeln('RSI last value: ', vRSI.Values[0, vRSI.PointCount - 1]: 0: 4);
      Writeln('MACD line/signal/histogram: ',
        vMACD.Values[0, vMACD.PointCount - 1]: 0: 4, ' / ',
        vMACD.Values[1, vMACD.PointCount - 1]: 0: 4, ' / ',
        vMACD.Values[2, vMACD.PointCount - 1]: 0: 4);

      vRaster := TsgcUICanvas_Raster.Create(800, 400);
      try
        vChart.Render(vRaster, sgcUIRect(0, 0, 800, 400));
        vRaster.SaveToPNGFile('oscillator.png');
      finally
        vRaster.Free;
      end;
    finally
      vMACD.Free;
      vRSI.Free;
      vPrice.Free;
    end;
  finally
    vChart.Free;
  end;
end;
```

Voir aussi

- TsgcChartSeries_Indicator
- TsgcChartSeries_Volume
- Panneaux
- TsgcChartSeries_Candlestick
- TsgcChartSeries
- Éditions

TsgcChartSeries_Volume

Unit: sgcCharts_Series_Financial
Éditions Registered et Trial uniquement (exclu de l'édition Community ; voir Éditions).
TsgcChartSeries_Volume dessine les barres de volume d'une **Source** OHLC, colorées selon que cette barre a clôturé à la hausse ou à la baisse. Elle descend directement de TsgcChartSeries_Indicator (pas de TsgcChartSeries_Oscillator) : le volume n'a pas de plafond fixe, elle conserve donc une plage Y flottante avec sa ligne de base épinglée à zéro plutôt qu'une échelle 0..100. Elle calcule deux sorties par barre, 0 est le volume et 1 est la direction (1 pour clôture >= ouverture, 0 sinon), mais seule la sortie 0 compte pour les limites de l'axe Y, si bien que l'indicateur de direction n'étire jamais l'axe. Comme RSI, Stochastic, MACD et ATR, elle est normalement placée dans son propre panneau sous le prix, car son échelle ne correspond pas à celle du prix. Voir Synchronisation avec la source pour la façon dont elle garde ses sorties alignées avec **Source**, les mêmes règles s'appliquent ici.

Propriétés

Propriété	Type	Description
UpColor	TsgcUIColor	Couleur d'une barre dont la clôture est supérieure ou égale à son ouverture. Par défaut, la couleur de hausse du graphique avec une opacité réduite.
DownColor	TsgcUIColor	Couleur d'une barre dont la clôture est inférieure à son ouverture. Par défaut, la couleur de baisse du graphique avec une opacité réduite.
BarWidthPercent	Integer	Part du créneau X occupée par une barre, de 1 à 100. Par défaut 70.

Tous les membres de TsgcChartSeries_Indicator s'appliquent aussi : **Source**, **Sync**, **Recalculate**, **PointCount**, **XValues** et **Values**. Lisez le volume de la barre traitée à aIndex avec `Values[0, aIndex]` et son indicateur de hausse / baisse avec `Values[1, aIndex]`.

Méthodes

Méthode	Valeur de retour	Description
Draw(aCanvas: TsgcUICanvas; aXAxis, aYAxis: TsgcChartAxis; const aPlot: TsgcUIRect; aScale: Double)		Redéfinie pour peindre des barres de volume, les barres haussières en **UpColor** et les barres baissières en **DownColor**, au lieu de la ligne que dessine par défaut TsgcChartSeries_Indicator.
DrawLegendSymbol(aCanvas: TsgcUICanvas; const aRect: TsgcUIRect; aScale: Double)		Redéfinie pour utiliser le symbole de légende en barre au lieu du symbole en ligne.

Exemple : barres de volume sous une série de chandeliers

```
uses
  SysUtils, Math,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Series_Financial,
  sgcCharts_Axis, sgcCharts_Types, sgcUICore_Canvas, sgcUICore_Canvas_Raster;

var
```

```
  vChart: TsgcChart;
  vAxisX, vAxisY: TsgcChartAxis;
  vPrice: TsgcChartSeries_Candlestick;
  vVolume: TsgcChartSeries_Volume;
  vRaster: TsgcUICanvas_Raster;
  i: Integer;
  vClose, vOpen, vVol: Double;
begin
  vChart := TsgcChart.Create(nil);
  try
    vAxisX := vChart.Axes.Add;
    vAxisX.Name := 'bottom';
    vAxisX.Position := capBottom;
    vAxisY := vChart.Axes.Add;
    vAxisY.Name := 'left';
    vAxisY.Position := capLeft;

    vPrice := TsgcChartSeries_Candlestick.Create(nil);
    vVolume := TsgcChartSeries_Volume.Create(nil);
    try
      vPrice.Chart := vChart;
      vPrice.AxisX := 'bottom';
      vPrice.AxisY := 'left';

      vClose := 100;
      for i := 0 to 79 do
      begin
        vOpen := vClose;
        vClose := vOpen + Sin(i / 6) * 2 + (Random - 0.5);
        vVol := 1000 + Random(500);
        vPrice.AddOHLC(i, vOpen, Max(vOpen, vClose) + 1,
          Min(vOpen, vClose) - 1, vClose, vVol);
      end;

      vVolume.Chart := vChart;
      vVolume.AxisX := 'bottom';
      vVolume.AxisY := 'left';
      vVolume.Source := vPrice;
      vVolume.BarWidthPercent := 70;

      vVolume.Sync;
      Writeln('last volume: ',
        vVolume.Values[0, vVolume.PointCount - 1]: 0: 2);
      Writeln('last bar up: ',
        vVolume.Values[1, vVolume.PointCount - 1] <> 0);

      vRaster := TsgcUICanvas_Raster.Create(800, 400);
      try
        vChart.Render(vRaster, sgcUIRect(0, 0, 800, 400));
        vRaster.SaveToPNGFile('volume.png');
      finally
        vRaster.Free;
      end;
    finally
      vVolume.Free;
      vPrice.Free;
    end;
  finally
    vChart.Free;
  end;
end;
```

Voir aussi

- TsgcChartSeries_Indicator

- TsgcChartSeries_Oscillator
- Panneaux
- TsgcChartSeries_Candlestick
- TsgcChartSeries
- Éditions

Panneaux

Unité : sgcCharts_Chart
Éditions Registered et Trial uniquement (exclu de l'édition Community ; voir Éditions).
Un TsgcChart possède normalement une seule zone de tracé. Lorsque **Panes** contient une ou plusieurs entrées, cette zone de tracé est divisée en bandes horizontales empilées de haut en bas, chacune avec son propre axe vertical et ses propres séries, tandis que chaque panneau continue de partager les axes horizontaux du graphique. C'est la disposition dont a besoin un graphique de prix avec un RSI, un MACD ou un volume en dessous : les chandeliers de prix dans un panneau, un oscillateur dans un autre en dessous, les deux défilant et zoomant ensemble car ils partagent le même axe du bas. **TsgcChartPane** est une bande de la collection, **TsgcChartPanes** la collection elle-même, et un axe rejoint un panneau via **TsgcChartAxis.Pane**.

Propriétés de TsgcChartPane

Propriété	Type	Description
Name	string	Recherché par **TsgcChartAxis.Pane** et par **TsgcChartPanes.Find** (insensible à la casse).
Height-Percent	Double	Poids du panneau : les panneaux visibles se partagent la hauteur du tracé proportionnellement à cette valeur, qui n'est pas un pourcentage littéral devant totaliser 100. Par défaut 100.
Visible	Boolean	Un panneau masqué n'occupe aucune hauteur de tracé et ses axes et séries ne sont pas dessinés. Par défaut True.

Méthodes de TsgcChartPanes

Méthode	Retour	Description
Add	TsgcChartPane	Ajoute un panneau au bas de la pile.
Find(const aName: string)	TsgcChartPane	Premier panneau portant ce Name (insensible à la casse), nil si aucun.

Propriétés et méthodes de TsgcChart

Membre	Type	Description
Panes	TsgcChartPanes (propriété)	Vide (valeur par défaut) signifie une seule zone de tracé ; une collection non vide divise le tracé entre les panneaux visibles, de haut en bas dans l'ordre de la collection.
HasPanes	Boolean (méthode)	True lorsque le dernier **Render** a divisé le tracé en panneaux.
PaneRect(aIndex: Integer)	TsgcUIRect (méthode)	Rectangle du panneau à cet index de collection lors du dernier **Render**, vide lorsque le panneau est masqué.
AxisPlotRect(aAxis: TsgcChartAxis)	TsgcUIRect (méthode)	Rectangle de tracé utilisé par aAxis lors du dernier **Render** : le panneau d'un axe vertical lorsque le graphique a des panneaux (vide lorsque son panneau est masqué), sinon l'ensemble du **PlotRect**.

SeriesPlotRect(aSeries: TsgcChartSeries)	TsgcUIRect (méthode)	Rectangle de tracé du panneau de aSeries, le panneau de son axe vertical.
SplitterAt(X, Y: Double)	Integer (méthode)	Index du panneau au-dessus du séparateur sous X, Y (pixels du graphique, l'espace entre deux panneaux visibles), -1 lorsque le point n'est pas sur un séparateur.
MoveSplitter(aIndex: Integer; Y: Double)	(méthode)	Déplace vers Y (pixels du graphique) le séparateur situé sous le panneau à aIndex : modifie le **HeightPercent** de ce panneau et du suivant visible, leur somme étant conservée.

Lier un axe et une série à un panneau

TsgcChartAxis.Pane (une chaîne, vide par défaut) nomme le panneau auquel appartient un axe vertical ; un **Pane** vide se résout vers le premier panneau visible. Un axe horizontal ignore complètement **Pane**, puisque chaque panneau partage les mêmes axes horizontaux. Une série ne possède pas de propriété **Pane** propre : elle est dessinée dans le panneau de son **YAxis**, si bien que placer une série dans un panneau ne dépend que de l'axe vertical nommé par son **AxisY**, et du **Pane** propre à cet axe. Un graphique avec un panneau de prix et un panneau RSI en dessous possède typiquement trois axes, un axe du bas partagé et un axe de gauche par panneau, chaque axe de gauche ayant son **Pane** défini sur le **Name** du panneau correspondant.

Faire glisser le séparateur

Les panneaux ne sont pas redimensionnés automatiquement : un hôte qui laisse l'utilisateur faire glisser l'espace entre deux panneaux appelle **SplitterAt** lors d'un clic de souris pour savoir si le pointeur se trouve sur un séparateur (et quel panneau se trouve au-dessus), puis appelle **MoveSplitter** avec cet index de panneau à chaque déplacement de la souris pendant le glissement, en transmettant le nouveau Y en pixels du graphique. **MoveSplitter** maintient constante la somme des **HeightPercent** des deux panneaux de part et d'autre du séparateur, si bien que les panneaux plus bas dans la pile ne sont pas affectés. C'est exactement ainsi que TsgcChartInteraction pilote le glissement du séparateur depuis les événements souris VCL et FMX ; un code qui effectue le rendu sans vue et ne reçoit jamais d'entrée souris n'a aucune raison d'appeler l'une ou l'autre méthode.

Exemple : un panneau de prix et un panneau RSI partageant l'axe du bas

```
uses
  SysUtils, Math,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Series_Financial,
  sgcCharts_Axis, sgcCharts_Types, sgcUICore_Canvas, sgcUICore_Canvas_Raster;

var
  vChart: TsgcChart;
  vPrice: TsgcChartSeries_Candlestick;
  vRSI: TsgcChartSeries_RSI;
  vAxisBottom, vAxisPrice, vAxisRSI: TsgcChartAxis;
  vRaster: TsgcUICanvas_Raster;
  i, vSplitter: Integer;
  vClose, vOpen: Double;
  vPricePane, vRSIPane: TsgcUIRect;
begin
  vChart := TsgcChart.Create(nil);
  try
    vChart.Panes.Add.Name := 'price';
    vChart.Panes[0].HeightPercent := 70;
    vChart.Panes.Add.Name := 'rsi';
    vChart.Panes[1].HeightPercent := 30;

    vAxisBottom := vChart.Axes.Add;
    vAxisBottom.Name := 'bottom';
    vAxisBottom.Position := capBottom;
```

```
    vAxisPrice := vChart.Axes.Add;
    vAxisPrice.Name := 'price';
    vAxisPrice.Position := capLeft;
    vAxisPrice.Pane := 'price';

    vAxisRSI := vChart.Axes.Add;
    vAxisRSI.Name := 'rsi';
    vAxisRSI.Position := capLeft;
    vAxisRSI.Pane := 'rsi';

    vPrice := TsgcChartSeries_Candlestick.Create(nil);
    vRSI := TsgcChartSeries_RSI.Create(nil);
    try
      vPrice.Chart := vChart;
      vPrice.AxisX := 'bottom';
      vPrice.AxisY := 'price';

      vClose := 100;
      for i := 0 to 79 do
      begin
        vOpen := vClose;
        vClose := vOpen + Sin(i / 6) * 2 + (Random - 0.5);
        vPrice.AddOHLC(i, vOpen, Max(vOpen, vClose) + 1,
          Min(vOpen, vClose) - 1, vClose, 1000 + Random(500));
      end;

      vRSI.Chart := vChart;
      vRSI.AxisX := 'bottom';
      vRSI.AxisY := 'rsi';
      vRSI.Source := vPrice;
      vRSI.Period := 14;

      vRaster := TsgcUICanvas_Raster.Create(800, 400);
      try
        vChart.Render(vRaster, sgcUIRect(0, 0, 800, 400));
        Writeln('has panes: ', vChart.HasPanes);

        vPricePane := vChart.PaneRect(0);
        vRSIPane := vChart.PaneRect(1);
        Writeln('price pane bottom: ', vPricePane.Bottom: 0: 2);
        Writeln('rsi pane top: ', vRSIPane.Top: 0: 2);

        vSplitter := vChart.SplitterAt((vPricePane.Left + vPricePane.Right) / 2,
          vPricePane.Bottom);
        Writeln('splitter above pane: ', vSplitter);
        if vSplitter >= 0 then
        begin
          vChart.MoveSplitter(vSplitter, vPricePane.Bottom + 20);
          vChart.Render(vRaster, sgcUIRect(0, 0, 800, 400));
          Writeln('price pane bottom after drag: ',
            vChart.PaneRect(0).Bottom: 0: 2);
        end;

        vRaster.SaveToPNGFile('panes.png');
      finally
        vRaster.Free;
      end;
    finally
      vRSI.Free;
      vPrice.Free;
    end;
  finally
    vChart.Free;
  end;
end;
```

Voir aussi

- TsgcChartSeries_Indicator
- TsgcChartSeries_Oscillator
- TsgcChartSeries_Volume
- TsgcChartInteraction
- Axes
- TsgcChart
- Éditions

TsgcChartAggregator

Unité : sgcCharts_Aggregator
Éditions Registered et Trial uniquement (exclu de l'édition Community ; voir Éditions).
TsgcChartAggregator est un `TComponent` qui construit les chandeliers de **Series**, une série chandelier ou OHLC, à partir d'un flux de transactions : appelez **AddTrade(aTime, aPrice, aQty)** pour chaque transaction, aTime étant un `TDateTime` UTC. Chaque chandelier couvre un compartiment de **Interval** et son X est le début de ce compartiment (l'heure d'ouverture, la même convention qu'une kline Binance). Une transaction dans le compartiment du chandelier en cours de formation met à jour son high, low, close et volume sur place ; une transaction dans un compartiment ultérieur scelle ce chandelier et en ajoute un nouveau ouvert au début de son propre compartiment, **FillGaps** insérant éventuellement un chandelier plat pour chaque compartiment vide entre les deux ; une transaction plus ancienne que le chandelier en formation est ignorée et comptée dans **DroppedCount**. La première transaction après la création, ou après **Reset**, poursuit le chandelier le plus récent de **Series** lorsque ce chandelier se trouve dans le même compartiment (historique déjà chargé avant le début des transactions en direct), et est abandonnée lorsque le chandelier le plus récent appartient à un compartiment ultérieur.
Les compartiments sont calculés en millisecondes entières, alignés sur UTC : les secondes, minutes et heures s'alignent sur l'horloge, les jours à 00:00, les semaines le lundi à 00:00 et les mois le premier jour du mois calendaire. **SessionAlign** décale chaque limite d'un nombre de minutes, pour les sessions qui ne débutent pas à 00:00 UTC.

Propriétés

Propriété	Type	Description
Series	TsgcChartSeries_OHLCBase	La série de chandeliers qui reçoit les chandeliers. La modifier, ou la libération de la série, oublie le chandelier en formation de la même façon que **Reset**.
Interval	TsgcChartAggregatorInterval	Taille du compartiment : `cai1s`, `cai5s`, `cai15s`, `cai30s`, `cai1m`, `cai3m`, `cai5m`, `cai15m`, `cai30m`, `cai1h`, `cai2h`, `cai4h`, `cai6h`, `cai12h`, `cai1d`, `cai1w` ou `cai1Month`. Par défaut `cai1m`. La modifier oublie le chandelier en formation.
SessionAlign	Integer	Minutes ajoutées à chaque limite UTC, 0 (valeur par défaut) signifiant des limites UTC simples. La modifier oublie le chandelier en formation.
FillGaps	Boolean	Insère un chandelier plat (open = high = low = close = le close précédent, volume 0) pour chaque compartiment vide entre deux transactions. Par défaut False.
DroppedCount	Integer (lecture seule)	Nombre de transactions ignorées car plus anciennes que le compartiment du chandelier en formation.

Méthodes

Méthode	Retour	Description
AddTrade(aTime: TDateTime; aPrice, aQty: Double)		Alimente une transaction. Met à jour **Series** de la même façon que le feraient **AddOHLC** et **UpdateLastOHLC**, elle doit donc être appelée depuis le même thread que ceux-ci.

Reset		Oublie le chandelier en formation et efface **DroppedCount**. **Series** elle-même n'est pas modifiée.
BucketStart(aTime: TDateTime)	TDateTime	Début du compartiment dans lequel tombe aTime, en respectant **Interval** et **SessionAlign**.

Alimenter l'agrégateur

Les transactions Binance sont agrégées par un TsgcChartDataSource_Binance en quatre étapes : déposer une série chandelier ou OHLC sur le graphique (une TsgcChartSeries_Volume dans son propre panneau peut afficher le volume des chandeliers) ; assigner cette série à **Series** et choisir **Interval** ; définir le `Stream` de la source de données sur `cbsTrade` ; assigner l'agrégateur à sa propriété `Aggregator`. Chaque transaction atteint alors **AddTrade** avec son horodatage, son prix et sa quantité, si bien que les chandeliers portent le volume échangé, et les transactions ne sont pas tracées sous forme de ligne. Toute autre alimentation appelle elle-même **AddTrade**, depuis quel que soit le thread qui produit les transactions, tant que c'est le thread depuis lequel **Series** est par ailleurs mise à jour.

Exemple : agréger des transactions en chandeliers d'une minute

```
uses
  SysUtils, DateUtils, Math,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Series_Financial,
  sgcCharts_Aggregator, sgcCharts_Axis, sgcCharts_Types,
  sgcUICore_Canvas, sgcUICore_Canvas_Raster;

var
  vChart: TsgcChart;
  vPrice: TsgcChartSeries_Candlestick;
  vAggregator: TsgcChartAggregator;
  vAxisBottom, vAxisLeft: TsgcChartAxis;
  vRaster: TsgcUICanvas_Raster;
  vBase: TDateTime;
begin
  vChart := TsgcChart.Create(nil);
  try
    { TsgcChart already carries two default axes ('X' and 'Y'); adding named
      ones and keeping a reference to each avoids indexing into Axes, whose
      order includes those defaults }
    vAxisBottom := vChart.Axes.Add;
    vAxisBottom.Name := 'bottom';
    vAxisBottom.Position := capBottom;
    vAxisBottom.AxisType := catDateTime;

    vAxisLeft := vChart.Axes.Add;
    vAxisLeft.Name := 'left';
    vAxisLeft.Position := capLeft;

    vPrice := TsgcChartSeries_Candlestick.Create(nil);
    vAggregator := TsgcChartAggregator.Create(nil);
    try
      vPrice.Chart := vChart;
      vPrice.AxisX := 'bottom';
      vPrice.AxisY := 'left';

      vAggregator.Series := vPrice;
      vAggregator.Interval := cai1m;
      vAggregator.FillGaps := True;

      vBase := EncodeDateTime(2026, 9, 28, 9, 0, 0, 0);
      vAggregator.AddTrade(vBase, 100, 1);
```

```
      vAggregator.AddTrade(vBase + 10 / SecsPerDay, 100.5, 2);
      vAggregator.AddTrade(vBase + 40 / SecsPerDay, 99.8, 1);
      // two minutes after the base trade: seals the forming candle and,
      // with FillGaps, inserts the flat candle for the empty minute between
      vAggregator.AddTrade(vBase + 2 * 60 / SecsPerDay, 101, 3);
      // older than the forming candle: dropped, counted in DroppedCount
      vAggregator.AddTrade(vBase, 90, 1);

      Writeln('bucket start of the base trade: ',
        DateTimeToStr(vAggregator.BucketStart(vBase)));
      Writeln('candles built: ', vPrice.OHLCBuffer.Count);
      Writeln('dropped trades: ', vAggregator.DroppedCount);

      vRaster := TsgcUICanvas_Raster.Create(800, 400);
      try
        vChart.Render(vRaster, sgcUIRect(0, 0, 800, 400));
        vRaster.SaveToPNGFile('aggregator.png');
      finally
        vRaster.Free;
      end;
    finally
      vAggregator.Free;
      vPrice.Free;
    end;
  finally
    vChart.Free;
  end;
end;
```

Voir aussi

- TsgcChartSeries_Indicator
- TsgcChartDrawing
- Panneaux
- Présentation de la passerelle
- TsgcChartDataSource_Binance
- TsgcChartSeries_Candlestick
- Éditions

TsgcChartDrawing

Unité : sgcCharts_Drawing
Éditions Registered et Trial uniquement (exclu de l'édition Community ; voir Éditions).
TsgcChart.Drawings contient des éléments **TsgcChartDrawing**, lignes de tendance, lignes horizontales et verticales, rectangles et retracements de Fibonacci, ancrés sur des valeurs de données (**X1**, **Y1**, **X2**, **Y2**) des axes nommés par **AxisX** et **AxisY**, de sorte qu'un dessin suit le zoom, le panoramique et les données en direct au lieu de rester à une position de pixel fixe. **TsgcChart.RenderContent** dessine les éléments après les séries et les axes, rognés au panneau de leur axe vertical. Aux côtés des dessins, une série porte ses propres marqueurs de prix, **TsgcChartSeries.LastValueMarker** et **TsgcChartSeries.BidAskBand**, tous deux des `TPersistent` et tous deux documentés sur cette page.

Propriétés de TsgcChartDrawing

Propriété	Type	Description
Kind	TsgcChartDrawingKind	`cdkTrendLine` (le segment entre les deux ancrages), `cdkHorizontalLine` (une ligne traversant le panneau à Y1), `cdkVerticalLine` (une ligne traversant le panneau à X1), `cdkRectangle` (la boîte des deux ancrages, contour plus un remplissage léger) ou `cdkFibonacci` (les niveaux de retracement entre X1 et X2). Par défaut `cdkTrendLine` .
X1, Y1, X2, Y2	Double	Ancrages, comme valeurs de données de **AxisX** (X) et **AxisY** (Y). Pour `cdkFibonacci` , 0 % se situe à Y2 et 100 % à Y1 (glisser du début du mouvement à sa fin) ; les niveaux couvrent toute la largeur du panneau lorsque X1 est égal à X2.
AxisX	string	Vide signifie le premier axe du bas, la même convention qu'une série.
AxisY	string	Vide signifie le premier axe de gauche ; le dessin est peint dans le panneau de cet axe. Voir Panneaux.
Color	TsgcUIColor	0 (valeur par défaut) signifie la première couleur de palette du graphique.
LineWidth	Double	Pixels d'appareil à l'échelle 1. Par défaut 1.5.
Dash	TsgcChartDrawingDash	`cddSolid` , `cddDash` ou `cddDot` . Par défaut `cddSolid` .
Visible	Boolean	Par défaut True.
Locked	Boolean	Un dessin verrouillé peut toujours être sélectionné, mais pas déplacé ni supprimé de façon interactive. Par défaut False.

Méthode de TsgcChartDrawing

Méthode	Retour	Description
SetAnchors(const aX1, aY1, aX2, aY2: Double)		Définit les quatre ancrages avec une seule notification de changement, au lieu d'une par propriété.

TsgcChartDrawings (TsgcChart.Drawings)

TsgcChartDrawings est le `TOwnedCollection` derrière **TsgcChart.Drawings**.

Membre	Retour	Description
Add	Tsgc-Chart-Drawing	Ajoute un dessin.
AnchorPixels(aIndex: Integer; out aX1, aY1, aX2, aY2: Double; out aPlot: TsgcUIRect)	Boolean	Ancrages du dessin à aIndex, en pixels du graphique, à la date du dernier **Render** ; retourne aussi le rectangle du panneau dans lequel le dessin a été peint.
PixelToValue(aIndex: Integer; X, Y: Double; out aX, aY: Double)	Boolean	Valeurs de données du pixel du graphique X, Y sur les axes du dessin à aIndex, à la date du dernier **Render**.
HitTest(X, Y: Double; out aHandle: Integer)	Integer	Index du dessin sous X, Y (pixels du graphique), le plus au premier plan d'abord, -1 si aucun. aHandle est l'ancrage sous le point, 0 ou 1, ou -1 pour un impact sur le corps.
DeleteSelected	Boolean	Supprime **Selected** sauf s'il est **Locked** ; True lorsqu'il a été supprimé.
Draw(aCanvas: TsgcUICanvas; aScale: Double)		Peint chaque dessin visible. Appelée par **TsgcChart.RenderContent**.
DrawSelection(aCanvas: TsgcUI-Canvas; aScale: Double)		Peint les poignées de **Selected**, faisant partie de la sur-couche d'interaction.
SaveToJSON	string	Un objet avec un tableau "drawings", un objet par dessin : "kind" (trendline, hline, vline, rectangle ou fibonacci), "x1", "y1", "x2", "y2", "axisX", "axisY", "color" (ARGB sous forme de nombre non signé), "lineWidth", "dash" (solid, dash ou dot), "visible" et "locked". Les nombres utilisent "." quelle que soit la locale et conservent chaque chiffre du double.
LoadFromJSON(const aJSON: string)		Remplace les dessins (un tableau racine est également accepté) ; un élément d'un type inconnu est ignoré, et un JSON mal formé lève **EsgcChartJSONError**.
Selected	Integer (pro-priété)	Index du dessin sélectionné, -1 si aucun.

Dessin interactif et sélection

TsgcChartInteraction.DrawingTool crée un dessin avec la souris : l'appui pose le premier ancrage, le glissement pose le second, le relâchement le termine et l'outil revient à `cdtNone`. Sans outil actif, un clic sélectionne le dessin sous le pointeur (**Drawings.Selected**), un glissement sur son corps le déplace, un glissement sur une poignée déplace cet ancrage, et **TsgcChartInteraction.DeleteSelectedDrawing**, lié à la touche Suppr dans les vues VCL et FMX, le supprime via **DeleteSelected**. Voir TsgcChartInteraction pour le câblage souris et clavier ; un code sans vue appelle plutôt directement **Add**, **HitTest** et **Selected**.

TsgcChartLastValueMarker

Une ligne en pointillés traversant le panneau à la dernière valeur d'une série, avec une étiquette pour cette valeur sur l'axe vertical, du côté du tracé.

Propriété	Type	Description
Visible	Boolean	Par défaut False.
Color	TsgcUI-Color	0 (valeur par défaut) signifie la couleur de la série ; pour une série de chandeliers, la couleur de hausse ou de baisse du dernier chandelier.

TsgcChartBidAskBand

Bid et Ask, définis par le code, dessinés comme une bande légère entre les deux prix, une ligne à chacun et une étiquette pour chacun sur l'axe vertical.

Propriété	Type	Description
Visible	Boolean	Par défaut False.
Bid, Ask	Double	Définis ensemble avec **SetPrices** pour ne déclencher qu'une seule notification de changement.
BidColor, Ask-Color	TsgcUI-Color	Par défaut une couleur d'accent verte (bid) et rouge (ask).

SetPrices(const aBid, aAsk: Double) définit les deux prix avec une seule notification, au lieu de deux.
Les deux marqueurs, comme les dessins, sont peints pour une série X/Y (son dernier Y) et pour une série de chandeliers (son dernier close) ; une série affiche l'un, les deux, ou aucun, indépendamment des autres séries du même graphique.

Exemple : une ligne de tendance, le marqueur de dernière valeur et une bande bid/ask

```
uses
  SysUtils, Math,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Series_Financial,
  sgcCharts_Drawing, sgcCharts_Axis, sgcCharts_Types,
  sgcUICore_Canvas, sgcUICore_Canvas_Raster;

var
  vChart: TsgcChart;
  vPrice: TsgcChartSeries_Candlestick;
  vTrend: TsgcChartDrawing;
  vAxisBottom, vAxisLeft: TsgcChartAxis;
  vRaster: TsgcUICanvas_Raster;
  i: Integer;
  vClose, vOpen: Double;
  vJSON: string;
  vHandle, vHit: Integer;
  vX1, vY1, vX2, vY2: Double;
  vPlot: TsgcUIRect;
begin
  vChart := TsgcChart.Create(nil);
  try
    { TsgcChart already carries two default axes ('X' and 'Y'); adding named
      ones and keeping a reference to each avoids indexing into Axes, whose
      order includes those defaults }
    vAxisBottom := vChart.Axes.Add;
    vAxisBottom.Name := 'bottom';
    vAxisBottom.Position := capBottom;

    vAxisLeft := vChart.Axes.Add;
    vAxisLeft.Name := 'left';
```

```
    vAxisLeft.Position := capLeft;

    vPrice := TsgcChartSeries_Candlestick.Create(nil);
    try
      vPrice.Chart := vChart;
      vPrice.AxisX := 'bottom';
      vPrice.AxisY := 'left';

      vClose := 100;
      for i := 0 to 79 do
      begin
        vOpen := vClose;
        vClose := vOpen + Sin(i / 6) * 2 + (Random - 0.5);
        vPrice.AddOHLC(i, vOpen, Max(vOpen, vClose) + 1,
          Min(vOpen, vClose) - 1, vClose, 1000 + Random(500));
      end;

      vPrice.LastValueMarker.Visible := True;
      vPrice.BidAskBand.Visible := True;
      vPrice.BidAskBand.SetPrices(vClose - 0.5, vClose + 0.5);

      vTrend := vChart.Drawings.Add;
      vTrend.Kind := cdkTrendLine;
      vTrend.AxisX := 'bottom';
      vTrend.AxisY := 'left';
      vTrend.SetAnchors(10, 95, 70, 105);

      vRaster := TsgcUICanvas_Raster.Create(800, 400);
      try
        vChart.Render(vRaster, sgcUIRect(0, 0, 800, 400));

        Writeln('drawings: ', vChart.Drawings.Count);
        if vChart.Drawings.AnchorPixels(0, vX1, vY1, vX2, vY2, vPlot) then
        begin
          vHit := vChart.Drawings.HitTest(vX1, vY1, vHandle);
          Writeln('hit test at anchor 1: drawing=', vHit, ' handle=', vHandle);
        end;

        vJSON := vChart.Drawings.SaveToJSON;
        Writeln('JSON length: ', Length(vJSON));
        vChart.Drawings.LoadFromJSON(vJSON);
        Writeln('drawings after reload: ', vChart.Drawings.Count);

        vChart.Drawings.Selected := 0;
        Writeln('deleted: ', vChart.Drawings.DeleteSelected);
        Writeln('drawings after delete: ', vChart.Drawings.Count);

        vRaster.SaveToPNGFile('drawing.png');
      finally
        vRaster.Free;
      end;
    finally
      vPrice.Free;
    end;
  finally
    vChart.Free;
  end;
end;
```

Voir aussi

- TsgcChartSeries_Indicator
- TsgcChartAggregator
- Panneaux
- TsgcChartInteraction

- TsgcChartSeries_Candlestick
- TsgcChartSeries
- Éditions

TsgcChartBuffer and TsgcChartOHLCBuffer

Unit: sgcCharts_Buffer
Tampons circulaires pour les séries de graphique. **TsgcChartBuffer** conserve des points X/Y, **TsgcChartOHLCBuffer** conserve X, Open, High, Low, Close et Volume. X est un `Double` (un `TDateTime` est stocké comme un `Double`). L'ajout est en O(1). En mode `cbmOverwrite`, un tampon plein supprime son point le plus ancien au prochain ajout, en mode `cbmGrow` la capacité double.
Chaque méthode publique prend le verrou du tampon, si bien qu'un thread d'alimentation peut ajouter des données pendant que le thread de rendu lit. Le thread de rendu doit appeler **Snapshot** ou **SnapshotRange**, qui copient les données dans un enregistrement possédé par l'appelant sous le verrou et le libèrent aussitôt, puis décimer et dessiner à partir de cette copie ; voir Décimation.
X est censé être monotone non décroissant. Append maintient l'indicateur **Monotonic** : il passe à False dès qu'un point arrive avec un X inférieur au précédent, et seul **Clear** le réinitialise. Quand **Monotonic** vaut True, **SnapshotRange** trouve la plage visible par recherche binaire et ajoute le point le plus proche en dehors de chaque bord, si bien que les lignes qui sortent du tracé sont dessinées. Quand il vaut False, **SnapshotRange** se rabat sur un balayage linéaire qui ne copie que les points à l'intérieur de la plage, et les fonctions de décimation retournent les données sans décimation.

Propriétés et méthodes partagées (TsgcChartBufferBase)

Membre	Description
Capacity: Integer	Capacité actuelle en points ou chandeliers. L'augmenter réalloue ; la réduire en dessous de **Count** élimine les données les plus anciennes.
Count: Integer	Nombre de points ou de chandeliers actuellement conservés.
Mode: TsgcChartBufferMode	`cbmOverwrite` (supprime le point le plus ancien une fois plein) ou `cbmGrow` (double la capacité une fois plein).
Monotonic: Boolean	True tant que chaque X ajouté a été >= au X précédent. Voir ci-dessus.
Lock / Unlock	Prend ou libère le verrou propre du tampon, pour un appelant qui a besoin que plusieurs appels soient atomiques.
Clear	Vide le tampon et réinitialise **Monotonic** à True.

TsgcChartBuffer

Membre	Description
Create(aCapacity: Integer = 1024; aMode: TsgcChartBufferMode = cbmOverwrite)	Constructeur.
Append(const aX, aY: Double)	Ajoute un point.
AppendBatch(const aX, aY: array of Double)	Ajoute plusieurs points en un seul appel verrouillé.
First(out aX, aY: Double): Boolean	Point le plus ancien. False lorsque le tampon est vide.
Last(out aX, aY: Double): Boolean	Point le plus récent. False lorsque le tampon est vide.

Snapshot(var aSnap: TsgcChartBufferSnapshot)	Copie chaque point dans `aSnap`. Les tableaux sont réutilisés entre les appels et peuvent être plus longs que `aSnap.Count`.
SnapshotRange(const aXMin, aXMax: Double; var aSnap: TsgcChartBufferSnapshot)	Copie les points visibles dans [aXMin, aXMax], plus les points de bord décrits ci-dessus quand **Monotonic**.
X[aIndex], Y[aIndex]: Double	Ordre logique : 0 est le point le plus ancien, `Count - 1` le plus récent.

TsgcChartOHLCBuffer

Membre	Description
Create(aCapacity: Integer = 1024; aMode: TsgcChartBufferMode = cbmOverwrite)	Constructeur.
Append(const aX, aOpen, aHigh, aLow, aClose: Double; const aVolume: Double = 0)	Ajoute un chandelier.
AppendBatch(const aX, aOpen, aHigh, aLow, aClose, aVolume: array of Double)	Ajoute plusieurs chandeliers en un seul appel verrouillé. `aVolume` peut être vide, le volume est alors 0.
First / Last(out aX, aOpen, aHigh, aLow, aClose, aVolume: Double): Boolean	Chandelier le plus ancien / le plus récent. False lorsque le tampon est vide.
UpdateLast(const aX, aOpen, aHigh, aLow, aClose, aVolume: Double): Boolean	Remplace les valeurs du chandelier le plus récent lorsque son X est égal à `aX`, pour un chandelier en direct encore en formation. Retourne False et ne change rien lorsque le tampon est vide ou que le chandelier le plus récent a un autre X.
Snapshot(var aSnap: TsgcChartOHLCSnapshot)	Copie chaque chandelier dans `aSnap`.
SnapshotRange(const aXMin, aXMax: Double; var aSnap: TsgcChartOHLCSnapshot)	Copie les chandeliers visibles dans [aXMin, aXMax].
X[aIndex], Open[aIndex], High[aIndex], Low[aIndex], Close[aIndex], Volume[aIndex]: Double	Ordre logique : 0 est le chandelier le plus ancien, `Count - 1` le plus récent.

Exemple : ajouter des données et prendre un instantané

```
var
  vBuffer: TsgcChartBuffer;
  vSnap: TsgcChartBufferSnapshot;
  i: Integer;
begin
  vBuffer := TsgcChartBuffer.Create(1024, cbmOverwrite);
  try
    for i := 0 to 999 do
      vBuffer.Append(i, Sin(i / 20) * 50);

    vBuffer.Snapshot(vSnap);
    Writeln(Format('points: %d  first x: %g  last x: %g',
      [vSnap.Count, vSnap.X[0], vSnap.X[vSnap.Count - 1]]));
  finally
```

```
    vBuffer.Free;
  end;
end;
```

Voir aussi

- Décimation
- TsgcChartDataSource

Décimation

Unit: sgcCharts_Buffer
Fonctions sans dépendance à un framework qui réduisent un instantané de tampon à environ un ensemble de points par colonne de pixels, pour un tracé large de `aPixelWidth` pixels affichant `[aXMin, aXMax]`. Un instantané non monotone, une plage vide, ou `aPixelWidth < 1` est copié sans décimation dans chaque cas ci-dessous.

Valeur / Fonction	Description
TsgcChartDecimation = (dmNone, dmMinMax, dmLTTB)	`dmNone` copie chaque point. `dmMinMax` découpe X en colonnes de pixels et conserve, pour chaque colonne, le premier, le minimum, le maximum et le dernier point dans l'ordre de X (au plus 4), si bien que la polyligne reste visuellement exacte ; les points à gauche de `aXMin` partagent une colonne supplémentaire, les points à droite de `aXMax` une autre. `dmLTTB` retourne `2 × aPixelWidth` points choisis par Largest Triangle Three Buckets, destiné aux séries statiques.
sgcChartDecimate(const aSnap: TsgcChartBufferSnapshot; const aXMin, aXMax: Double; aPixelWidth: Integer; aMode: TsgcChartDecimation; var aOutX, aOutY: TsgcChartDoubleArray; out aOutCount: Integer)	Décime un instantané de points selon `aMode`. `aOutX` et `aOutY` sont agrandis quand ils sont trop courts, jamais réduits.
sgcChartLTTB(const aX, aY: TsgcChartDoubleArray; aCount, aThreshold: Integer; var aOutX, aOutY: TsgcChartDoubleArray; out aOutCount: Integer)	Largest Triangle Three Buckets directement sur une paire de tableaux X/Y. Retourne exactement `Min(aThreshold, aCount)` points, le premier et le dernier toujours inclus.
sgcChartDecimateOHLC(const aSnap: TsgcChartOHLCSnapshot; const aXMin, aXMax: Double; aPixelWidth: Integer; var aOut: TsgcChartOHLCSnapshot)	Fusionne les chandeliers qui tombent dans la même colonne de pixels : X et ouverture du premier, plus haut le plus élevé, plus bas le plus faible, clôture du dernier, somme des volumes. `aOut.Count` contient le résultat.

Exemple : décimer un instantané pour un tracé de 800 px

```
var
  vBuffer: TsgcChartBuffer;
  vSnap: TsgcChartBufferSnapshot;
  vOutX, vOutY: TsgcChartDoubleArray;
  vOutCount, i: Integer;
begin
  vBuffer := TsgcChartBuffer.Create(100000, cbmOverwrite);
  try
    for i := 0 to 99999 do
      vBuffer.Append(i, Sin(i / 200) * 100);

    vBuffer.Snapshot(vSnap);
    sgcChartDecimate(vSnap, 0, 99999, 800, dmMinMax, vOutX, vOutY, vOutCount);
    Writeln(Format('decimated from %d to %d points', [vSnap.Count, vOutCount]));
  finally
    vBuffer.Free;
  end;
end;
```

Voir aussi

- TsgcChartBuffer and TsgcChartOHLCBuffer

TsgcUICanvas

Unit: sgcUICore_Canvas, avec deux moteurs : sgcUICore_Canvas_Raster (**TsgcUICanvas_Raster**) et sgcUICore_Canvas_SVG (**TsgcUICanvas_SVG**)
TsgcUICanvas est la surface de dessin abstraite sur laquelle TsgcChart.**Render** peint. Le paquet de base fournit deux moteurs concrets : **TsgcUICanvas_Raster** peint dans un tampon ARGB en mémoire et l'enregistre en PNG, **TsgcUICanvas_SVG** construit un document SVG. Les deux partagent la même API de chemins, de peinture et de texte, si bien qu'un graphique se rend de façon identique avec l'un ou l'autre. Une application se contente normalement de créer le canevas, d'appeler **Render**, et d'enregistrer le résultat ; les primitives de dessin ci-dessous sont également publiques pour ajouter des superpositions personnalisées.

Types

Type	Description
TsgcUIColor = Cardinal	Droit (non prémultiplié) `$AARRGGBB` .
TsgcUIRect	`Left` , `Top` , `Right` , `Bottom` : Double, pixels physiques ; Right / Bottom sont les bords extrêmes.
TsgcUIFontStyle	`Size` : Double, `Bold` : Boolean, `Color` : TsgcUIColor.
TsgcUILineJoin, TsgcUILineCap	`(ljMiter, ljRound, ljBevel)` , `(lcButt, lcRound, lcSquare)` .
TsgcUITextHAlign, TsgcUITextVAlign	`(thaLeft, thaCenter, thaRight)` , `(tvaTop, tvaMiddle, tvaBaseline, tvaBottom)` .
TsgcUIFillRule	`frNonZero` , `frEvenOdd` .

Méthodes

Méthode	Description
Clear(aColor: TsgcUIColor)	Remplit toute la surface.
BeginPath, MoveTo, LineTo, QuadTo, CubicTo, Arc, RoundRect, Rectangle, ClosePath	Construction de chemins. **Arc** prend des angles en radians, 0 sur +X, balayage positif dans le sens horaire à l'écran.
FillPath(aColor: TsgcUIColor; aRule: TsgcUIFillRule = frNonZero)	Remplit le chemin courant ; conservé jusqu'au prochain **BeginPath**.
FillPathLinearGradient(aX1, aY1, aX2, aY2: Double; const aStops: array of TsgcUIGradientStop; aRule: TsgcUIFillRule = frNonZero)	Remplit le chemin courant avec un dégradé linéaire.
StrokePath(aColor, aWidth[, aJoin, aCap[, aDash, aDashOffset]])	Deux surcharges : avec ou sans motif de tirets.
DrawText, MeasureText, TextWidth, LineHeight, FitText	Dessin et mesure de texte avec un **TsgcUIFontStyle**. Voir Polices.
DrawImage(const aImage: TsgcUIImage; aX, aY: Integer; aOpacity: Double = 1)	Copie (blit) une image ARGB.

PushClip(aLeft, aTop, aRight, aBottom: Integer), PopClip	Rectangle de découpe en pixels physiques, Right / Bottom exclusifs ; les découpes imbriquées s'intersectent.
SetTransform(aA, aB, aC, aD, aE, aF: Double), ResetTransform	Transformation affine : `x' = A*x + C*y + E`, `y' = B*x + D*y + F`.
SetFontFile(const aRegular, aBold: string), GetFont(aBold: Boolean)	Polices utilisées pour mesurer et dessiner le texte ; un nom de fichier vide sélectionne la police Roboto intégrée. Voir Polices.
Width, Height	Taille de la surface en pixels physiques.

Ajouts de TsgcUICanvas_Raster

Membre	Description
Create(aWidth, aHeight: Integer)	Surcharge qui alloue son propre rasteriseur.
Create(aRasterizer: TsgcUIRasterizer)	Surcharge qui dessine sur un rasteriseur existant.
SetSize(aWidth, aHeight: Integer)	Redimensionne la surface.
SaveToPNGStream(aStream: TStream; aCompress: Boolean = True)	Encode la surface en PNG.
SaveToPNGFile(const aFileName: string; aCompress: Boolean = True)	Encode et écrit le PNG sur le disque.
Rasterizer	TsgcUIRasterizer (lecture seule), le rasteriseur sous-jacent.

Ajouts de TsgcUICanvas_SVG

Membre	Description
Create(aWidth, aHeight: Integer)	Crée un document vide de la taille indiquée.
SetSize(aWidth, aHeight: Integer)	Nouveau document vide de la taille indiquée.
SaveToString	string, le document complet avec la déclaration XML ; les découpes encore ouvertes sont fermées dans la sortie.
SaveToStream(aStream: TStream), SaveToFile(const aFileName: string)	Écrit le document.

Rendu d'un graphique en PNG ou SVG

Créez le canevas à la taille cible, appelez **TsgcChart.Render** avec un **TsgcUIRect** le couvrant, puis enregistrez. C'est tout le chemin de code dont un rapport sans interface graphique ou un export planifié a besoin, sans aucun contrôle VCL ou FireMonkey impliqué.

Exemple : le même graphique rendu en PNG et en SVG

```
uses
  SysUtils, Math,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_Axis, sgcCharts_Types,
  sgcUICore_Canvas, sgcUICore_Canvas_Raster, sgcUICore_Canvas_SVG;

var
  vChart: TsgcChart;
  vSeries: TsgcChartSeries_Line;
  i: Integer;
  vRaster: TsgcUICanvas_Raster;
  vSVG: TsgcUICanvas_SVG;
begin
  vChart := TsgcChart.Create(nil);
  try
    vChart.Axes.Add.Position := capBottom;
    vChart.Axes.Add.Position := capLeft;

    vSeries := TsgcChartSeries_Line.Create(nil);
    try
      vSeries.Chart := vChart;
      for i := 0 to 59 do
        vSeries.AddXY(i, Sin(i / 8) * 30 + 50);

      vRaster := TsgcUICanvas_Raster.Create(500, 250);
      try
        vChart.Render(vRaster, sgcUIRect(0, 0, 500, 250));
        vRaster.SaveToPNGFile('canvas.png');
      finally
        vRaster.Free;
      end;

      vSVG := TsgcUICanvas_SVG.Create(500, 250);
      try
        vChart.Render(vSVG, sgcUIRect(0, 0, 500, 250));
        Writeln('SVG length: ', Length(vSVG.SaveToString));
        vSVG.SaveToFile('canvas.svg');
      finally
        vSVG.Free;
      end;
    finally
      vSeries.Free;
    end;
  finally
    vChart.Free;
  end;
end;
```

Voir aussi

- TsgcChart
- Polices

TsgcUIFont

Unit: sgcUICore_Font
TsgcUIFont est le moteur de texte TrueType en Pascal pur que chaque TsgcUICanvas utilise pour mesurer et dessiner le texte ; il analyse lui-même le fichier de police, sans dépendance à la pile de texte du système d'exploitation. Chaque canevas crée son propre **TsgcUIFont** normal et gras via **SetFontFile** / **GetFont**, si bien qu'une application en crée rarement un directement, sauf pour mesurer du texte en dehors d'un canevas.

Propriétés

Propriété	Type	Description
Loaded	Boolean (lecture seule)	True une fois qu'un fichier ou un flux de police a été analysé.
UnitsPerEm, Ascender, Descender, LineGap	Integer (lecture seule)	Métriques de police en unités de police.
GlyphCount	Integer (lecture seule)	
CmapFormat	Integer (lecture seule)	Le format de table caractère-vers-glyphe trouvé dans le fichier.
HasKerning	Boolean (lecture seule)	True uniquement lorsque le fichier de police chargé possède lui-même une table de crénage.
Kerning	Boolean	Par défaut True. Applique la table de crénage de la police, si elle en a une, pendant la mesure et le dessin ; mettez-la à False pour désactiver le crénage même sur une police qui possède une table.
CacheCapacity, CacheCount	Integer	Taille et nombre d'entrées actuel du cache interne de contours de glyphes.

Méthodes

Méthode	Valeur de retour	Description
LoadFromFile(const aFileName: string)		Analyse un fichier de police TrueType (contours glyf). Lève **EsgcUIFont** pour une police CFF (contours PostScript), ou tout autre fichier qui n'est pas une police TrueType utilisable.
LoadFromStream(aStream: TStream)		Identique, depuis un flux.
LoadDefault(aBold: Boolean = False)		Charge la police intégrée à sgcCharts (Roboto Regular ou Bold). Réussit toujours ; aucun fichier requis.
LoadPlatformDefault(aBold: Boolean = False)		Police système de l'interface utilisateur choisie au mieux pour la plateforme actuelle (Segoe UI sous Windows, Roboto sous Android, DejaVu Sans ou Liberation Sans sous Linux, Helvetica ou Arial sous macOS/iOS) ; se rabat sur

		LoadDefault lorsqu'aucun de ces fichiers n'est trouvé, si bien qu'elle ne lève jamais d'exception.
CharToGlyph(aCode: Cardinal)	Integer	Point de code Unicode vers identifiant de glyphe ; 0 (.notdef) lorsqu'il n'est pas mappé.
GlyphAdvance(aGlyph: Integer)	Integer	Largeur de progression en unités de police.
GetKerning(aLeft, aRight: Integer)	Integer	Ajustement de la table de crénage en unités de police pour une paire de glyphes ; 0 en l'absence de paire.
IsCompositeGlyph(aGlyph: Integer)	Boolean	
GetGlyphOutline(aGlyph: Integer)	TsgcUIGlyphOutline	Contour mis en cache en unités de police, possédé par la police, valide jusqu'au prochain appel susceptible de l'évincer ; nil pour un identifiant de glyphe invalide.
MeasureText(const aText: string; aSize: Double; out aWidth, aAscent, aDescent: Double)		
TextWidth(const aText: string; aSize: Double)	Double	
LineHeight(aSize: Double)	Double	
FitText(const aText: string; aSize, aMaxWidth: Double)	string	Tronque **aText** pour tenir dans **aMaxWidth**.
DrawText(aRaster: TsgcUIRasterizer; const aText: string; aX, aY: Double; aSize: Double; aColor: Cardinal; aAngle: Double = 0)		Dessine directement sur un rasteriseur, en dessous du niveau **TsgcUICanvas**.
TextToCodePoints(const aText: string) (class function)	TsgcUICodePointArray	

Police intégrée et polices de la plateforme

sgcCharts intègre une compilation TrueType de Roboto afin que chaque graphique dessine du texte sans fichier externe : **LoadDefault** la charge directement, et c'est ce que sélectionne un nom de fichier vide passé à **TsgcUICanvas.SetFontFile**. **LoadPlatformDefault** essaie d'abord la police d'interface propre à la plateforme et ne se rabat sur la police intégrée que lorsqu'elle ne trouve ou ne peut analyser aucun fichier utilisable, ce qui en fait le choix par défaut le plus sûr pour une application qui veut correspondre au système d'exploitation hôte. La compilation Roboto intégrée ne porte aucune table de crénage, si bien que **HasKerning** retourne False pour elle même si **Kerning** reste True ; une police TrueType personnalisée chargée avec sa propre table de crénage est crénée automatiquement.

Seuls les contours TrueType sont pris en charge

TsgcUIFont lit directement la table de contours `glyf` ; elle n'a pas d'analyseur CFF (contours PostScript). De nombreuses polices système qui ressemblent à de simples fichiers TrueType, comme Helvetica ou SF sur macOS, sont en réalité au format CFF, et **LoadFromFile** / **LoadFromStream** lèvent **EsgcUIFont** pour elles. Lorsqu'une police par défaut de la plateforme s'avère être au format CFF, **LoadPlatformDefault** le détecte et passe au candidat suivant, en terminant par la police intégrée si rien d'utilisable n'est trouvé.

Exemple : police intégrée, police de la plateforme et mesure de texte

```
uses
  SysUtils,
  sgcUICore_Font;

var
  vFont: TsgcUIFont;
  vWidth, vAscent, vDescent: Double;
begin
  vFont := TsgcUIFont.Create;
  try
    vFont.LoadDefault(False);
    Writeln('embedded Roboto loaded: ', vFont.Loaded);
    Writeln('embedded Roboto has kerning: ', vFont.HasKerning);

    vFont.MeasureText('sgcCharts', 14, vWidth, vAscent, vDescent);
    Writeln(Format('width=%.2f ascent=%.2f descent=%.2f',
      [vWidth, vAscent, vDescent]));
  finally
    vFont.Free;
  end;

  vFont := TsgcUIFont.Create;
  try
    vFont.LoadPlatformDefault(False);
    Writeln('platform font loaded: ', vFont.Loaded);
    Writeln('platform font glyph count: ', vFont.GlyphCount);
  finally
    vFont.Free;
  end;
end;
```

Voir aussi

- Canevas

Indicateurs

Unit: sgcCharts_Indicators *(édition financière uniquement, compilé sous `SGC_CHARTS_FINANCIAL` ; une coquille vide sinon)*
Calculateurs d'indicateurs techniques incrémentaux, sans dépendance à un framework : **TsgcChartIndicatorSMA**, **EMA**, **WMA**, **Bollinger**, **RSI**, **MACD**, **Stochastic**, **ATR** et **VWAP**, tous descendant de **TsgcChartIndicatorCalc**. Ils ne dépendent pas de TsgcChart et peuvent alimenter une série supplémentaire, un moteur de signaux, ou du simple code de reporting.

Schéma commun

Membre	Description
Add(...)	Alimente une nouvelle barre et retourne la dernière sortie. O(1) amorti, sauf Stochastic qui parcourt sa fenêtre de plus haut/plus bas (O(KPeriod)).
UpdateLast(...)	Remplace l'entrée de la barre ajoutée le plus récemment, pour un chandelier en direct encore en formation : l'état sauvegardé avant le dernier **Add** est restauré et la nouvelle valeur est ajoutée à nouveau, si bien que le résultat correspond à un calcul refait depuis zéro avec la barre modifiée. Sur un calculateur vide, elle se comporte comme **Add**.
Calculate(const aInput: array of Double; ...)	Réinitialise le calculateur et fait passer tout un tableau par **Add**, si bien que les résultats par lot et incrémentaux correspondent. Le calculateur conserve ensuite son état, si bien que des barres en direct peuvent être ajoutées après un chargement d'historique.
Reset	Efface tout l'état ; **Count** revient à 0.
Count	Integer (lecture seule). Barres ajoutées depuis le dernier **Reset**.

Préchauffage

Chaque sortie vaut `NaN` (`Math.NaN`, à tester avec `Math.IsNan`) jusqu'à la fin de la période de préchauffage du calculateur ; une entrée `NaN` est ignorée, elle n'entre donc pas dans la fenêtre et la sortie pour cette barre est aussi `NaN`. Les index de barre ci-dessous commencent à 0.

Définitions

Calculateur	Constructeur	Définition
TsgcChartIndicatorSMA	Create(aPeriod: Integer = 20)	Moyenne arithmétique des dernières valeurs **Period** ; première valeur à la barre Period-1.
TsgcChartIndicatorEMA	Create(aPeriod: Integer = 20)	alpha = 2/(Period+1), amorcé avec la SMA des premières valeurs Period à la barre Period-1, puis Value := X * alpha + Prev * (1 - alpha).
TsgcChartIndicatorWMA	Create(aPeriod: Integer = 20)	Moyenne mobile pondérée linéairement, poids 1..Period (Period sur la plus récente), divisée par Period(Period+1)/2 ; première valeur à la barre Period-1.

TsgcChartIndicatorBollinger	Create(aPeriod: Integer = 20; const aDeviations: Double = 2)	Middle = SMA(Period), écart-type de population (division par Period), Upper / Lower = Middle +/- Deviations * StdDev ; première valeur à la barre Period-1.
TsgcChartIndicatorRSI	Create(aPeriod: Integer = 14)	Lissage de Wilder. Change(i) = Close(i) - Close(i-1) ; le premier gain/perte moyen est la moyenne des premiers changements Period (barre Period), puis Avg := (Avg * (Period-1) + Current) / Period. RSI = 100 - 100 / (1 + AvgGain / AvgLoss) ; 100 quand AvgLoss vaut 0, 50 quand les deux valent 0.
TsgcChartIndicatorMACD	Create(aFastPeriod: Integer = 12; aSlowPeriod: Integer = 26; aSignalPeriod: Integer = 9)	Line = EMA(Fast) - EMA(Slow), première Line à la barre Slow-1. Signal = EMA(Signal) de la Line, amorcé avec la SMA des premières valeurs Signal de la Line, première à la barre Slow+Signal-2. Histogram = Line - Signal.
TsgcChartIndicatorStochastic	Create(aKPeriod: Integer = 14; aKSmoothing: Integer = 3; aDPeriod: Integer = 3)	Raw = 100 * (Close - LowestLow) / (HighestHigh - LowestLow) sur les derniers plus haut/plus bas KPeriod (50 quand HighestHigh = LowestLow), première à la barre KPeriod-1. K = SMA(KSmoothing) de Raw (KSmoothing = 1 est le stochastique rapide), D = SMA(DPeriod) de K.
TsgcChartIndicatorATR	Create(aPeriod: Integer = 14)	Wilder. TR(i) = Max(High-Low, |High-PrevClose|, |Low-PrevClose|) à partir de la barre 1. Le premier ATR est la moyenne des premiers TR Period (barre Period), puis ATR := (ATR * (Period-1) + TR) / Period.
TsgcChartIndicatorVWAP	Create(const aSessionStart: TDateTime = 0)	Sum(TypicalPrice * Volume) / Sum(Volume) cumulatif, TypicalPrice = (High+Low+Close)/3. Les sommes redémarrent à la première barre de chaque session ; la session d'une barre est Floor(Time - SessionStart), si bien que SessionStart = 0 se réinitialise chaque jour calendaire et 9.5/24 se réinitialise à 09:30. Pas de préchauffage ; NaN tant que le volume de la session est 0.

Exemple : SMA et RSI, en flux et par lot

```
{$IFDEF SGC_CHARTS_FINANCIAL}
uses
  SysUtils, Math,
  sgcCharts_Buffer, sgcCharts_Indicators;

var
  vSMA: TsgcChartIndicatorSMA;
  vRSI: TsgcChartIndicatorRSI;
  vCloses: TsgcChartDoubleArray;
  vOut: TsgcChartDoubleArray;
  i: Integer;
  vValue: Double;
begin
  SetLength(vCloses, 40);
  for i := 0 to 39 do
    vCloses[i] := 100 + Sin(i / 4) * 5;

  vSMA := TsgcChartIndicatorSMA.Create(10);
  try
    for i := 0 to High(vCloses) do
    begin
      vValue := vSMA.Add(vCloses[i]);
      if (i = High(vCloses)) and not IsNan(vValue) then
        Writeln(Format('SMA(10) last = %.4f (count=%d)', [vValue, vSMA.Count]));
    end;

    vSMA.Calculate(vCloses, vOut);
```

```
    Writeln('batch SMA matches streaming last: ',
      SameValue(vOut[High(vOut)], vSMA.Value, 1E-9));
  finally
    vSMA.Free;
  end;

  vRSI := TsgcChartIndicatorRSI.Create(14);
  try
    for i := 0 to High(vCloses) do
      vValue := vRSI.Add(vCloses[i]);
    Writeln(Format('RSI(14) last = %.2f', [vRSI.Value]));
  finally
    vRSI.Free;
  end;
end;
{$ENDIF}
```

Voir aussi

- TsgcChartSeries

Présentation .NET

sgcCharts .NET est un portage managé en C# de la bibliothèque Delphi sgcCharts, pas un simple wrapper autour de celle-ci : chaque unité Delphi a un équivalent C# sous l'espace de noms **esegece.sgcCharts**, utilisant les mêmes noms de classes **Tsgc***, si bien qu'un code écrit contre l'API Delphi se lit de la même façon en C#. Le portage est réparti en six paquets NuGet afin qu'un projet n'intègre que la plateforme qu'il cible.

Paquets

PackageId	Contient	Frameworks cibles
esegece.sgcCharts	Le modèle de graphique de base : **TsgcChart**, les axes, les panneaux, les séries, les sources de données, les tampons, les dessins, les fonctions d'aide mathématiques/de formatage et JSON, et le canevas SVG. Aucune dépendance native.	netstandard2.0, net472, net6.0, net8.0, net9.0
esegece.sgcCharts.Rendering	**TsgcUICanvas_Raster**, le canevas raster SkiaSharp utilisé pour la sortie PNG sans interface graphique sous Windows, Linux et macOS.	netstandard2.0, net472, net6.0, net8.0, net9.0
esegece.sgcCharts.WinForms	**TsgcWinFormsChart**, la vue Windows Forms.	net472, net8.0-windows, net9.0-windows
esegece.sgcCharts.Wpf	**TsgcWpfChart**, la vue WPF.	net472, net8.0-windows, net9.0-windows
esegece.sgcCharts.Avalonia	**TsgcAvaloniaChart**, la vue multiplateforme (Windows, Linux, macOS).	net8.0, net9.0
esegece.sgcCharts.sgcWebSockets	Le pont en direct : des sources de données de graphique alimentées par une connexion **sgcWebSockets** (messages WebSocket, publications MQTT, server-sent events, flux de marché Binance). Dépend du paquet NuGet **esegece.sgcWebSockets**.	netstandard2.0, net8.0, net9.0

Éditions Community et Registered

La compilation par défaut de chaque paquet ci-dessus (sans commutateur) est l'édition **Registered**, payante : pas de constante `SGC_COMMUNITY`, pas de logo intégré, pas de notice de démarrage. Compiler ou empaqueter avec `-p:SgcCommunity=true` produit à la place l'édition **Community** (gratuite), ajoutant un suffixe **.Community** à l'identifiant de paquet de chaque paquet sauf le pont (**esegece.sgcCharts.sgcWebSockets**), que l'édition Community ne fournit pas du tout, la même règle que suit l'édition Community de Delphi pour ses propres unités de pont. La compilation Community conserve les huit séries, chandelier et OHLC inclus, ainsi que toutes les vues, et

ne retire que le pack financier (panneaux, séries indicateur et volume, agrégateur, outils de dessin, marqueur de dernière valeur et bande bid/ask).
Community est gratuite pour un usage aussi bien commercial que non commercial par de petites équipes : particuliers et organisations dont le chiffre d'affaires brut annuel est inférieur à 1 000 000 EUR et qui comptent au plus 5 développeurs. Les organisations atteignant ou dépassant l'un de ces seuils ont besoin d'une licence Commerciale (Registered).

L'API reflète Delphi

Les classes de vue conservent la dénomination Delphi : **TsgcWinFormsChart** et **TsgcWpfChart** sont les équivalents .NET du **TsgcVCLChart** Delphi, et **TsgcAvaloniaChart** est l'équivalent du **TsgcFMXChart** Delphi.
Les notifications du modèle de base, comme **TsgcChart.OnChange**, **TsgcChartDataSource.OnDataChanged**, **TsgcChartDataSource.OnBeforeAppend**, **TsgcChart.OnCrosshairMove**, **OnSeriesClick** et **OnZoom**, sont de simples propriétés de délégué `get; set;`, pas des membres `event` C#. Assigner un nouveau gestionnaire remplace le précédent, exactement comme un simple pointeur de méthode Delphi. Le seul endroit où un véritable événement multicast C# (`+=` / `-=`) intervient est celui où le pont **esegece.sgcCharts.sgcWebSockets** s'abonne à un client du portage .NET de sgcWebSockets ; voir Présentation du pont et Différences .NET par rapport à Delphi.
La plupart des propriétés tableau indexées de Delphi deviennent de véritables indexeurs C# : **TsgcChartAxes**, **TsgcChartPanes**, **TsgcChartDrawings** et **TsgcChartPalette** exposent tous `this[int Index]`, et **TsgcChartJSONNode** expose `this[int aIndex]` pour ses éléments de tableau. Les tampons de points derrière une série font exception : **TsgcChartBuffer.X(Index)** / **Y(Index)**, **TsgcChartOHLCBuffer.X(Index)** / **Open(Index)** / **High(Index)** / **Low(Index)** / **Close(Index)** / **Volume(Index)**, ainsi que **XValues(Index)** / **Values(Line, Index)** des séries financières sont des méthodes ordinaires, car C# n'autorise qu'une seule signature d'indexeur par classe et chacune de ces classes a besoin de plusieurs accesseurs par index nommés indépendamment, ou à plusieurs paramètres, sur la même classe. Voir Différences .NET par rapport à Delphi.

Vues et canevas libérables

Les classes qui détiennent une surface Skia native persistante ou un bitmap GDI dimensionné au contrôle implémentent `IDisposable` et doivent être libérées, sous peine de fuite de leur mémoire native :

- **TsgcUICanvas_Raster** (esegece.sgcCharts.Rendering) : détient le bitmap SkiaSharp, le canevas, les peintures et les polices.
- **TsgcChartRenderer** : détient son canevas raster.
- **TsgcWinFormsChart** : libère son canevas raster, son bitmap Skia et son bitmap GDI+, ainsi que son minuteur de limitation, depuis un `Dispose(bool)` redéfini.
- **TsgcWpfChart** : libère son canevas raster et son bitmap Skia.
- Le cadre interne (**TsgcAvaloniaChartFrame**) de **TsgcAvaloniaChart** : libère ses images Skia avant et arrière ainsi que ses canevas raster.

Le modèle de graphique de base lui-même (**TsgcChart**, séries, sources de données) ne détient aucune ressource native et n'a pas besoin d'être libéré pour cette raison ; il implémente néanmoins `IDisposable` à certains endroits pour exécuter une notification de libération de style Delphi, comme détacher une série de sa source ou une source de données de son client.

Métriques de texte identiques à Delphi

Chaque canevas mesure le texte avec **TsgcUIFont**, un lecteur TrueType porté depuis la partie METRICS du `sgcUICore_Font.pas` Delphi : il lit une table `cmap` au format 4 ou 12 pour la correspondance caractère/glyphe, les largeurs de progression depuis `hmtx`, les paires de crénage depuis une sous-table `kern` au format 0, et l'ascendant, le descendant et l'interligne depuis `hhea`. Il charge les mêmes fichiers intégrés **Roboto-Regular.ttf** et **Roboto-Bold.ttf** que la bibliothèque Delphi embarque (licence Apache 2.0), si bien qu'une mise en page calculée en .NET correspond au bit près à la mise en page Delphi. SkiaSharp ne sert qu'à rastériser les contours de glyphes déjà positionnés par le lecteur de métriques, un glyphe positionné par caractère, anticrénelé avec positionnement sous-pixel et sans hinting ; il ne mesure jamais le texte lui-même, ce qui garde chaque moteur, raster comme SVG, aligné avec Delphi.

Voir aussi

- Différences .NET par rapport à Delphi
- TsgcChart
- TsgcVCLChart

- Présentation du pont

Démarrage rapide : WinForms

TsgcWinFormsChart est la vue Windows Forms d'un modèle TsgcChart : une classe descendante de `Control` qui peint le graphique assigné à sa propriété **Chart**. Elle est fournie dans le paquet **esegece.sgcCharts.WinForms**, ciblant `net472`, `net8.0-windows` et `net9.0-windows`, et elle porte une `ProjectReference` (ou une `PackageReference`, lorsqu'elle est consommée depuis NuGet) vers **esegece.sgcCharts.Rendering** de façon transitive, si bien qu'un projet WinForms n'a besoin de référencer que **esegece.sgcCharts.WinForms** lui-même. Le contrôle est dessiné par le même canevas raster Skia que le TsgcVCLChart Delphi, directement dans une section DIB GDI persistante, limité à **Chart.MaxFPS**, et il est sûr de l'alimenter avec des données depuis un thread d'arrière-plan.

Propriétés

Propriété	Type	Description
Chart	Tsgc-Chart	Le modèle de graphique que ce contrôle peint. L'assigner enregistre le contrôle comme écouteur ; l'effacer, ou libérer le graphique, empêche le contrôle de dessiner quoi que ce soit.
Frame-Count	uint (lecture seule)	Nombre d'images complètes rendues depuis la création du contrôle.
Overlay-Frame-Count	uint (lecture seule)	Nombre d'images de superposition seule rendues depuis la création du contrôle, pour un changement qui ne fait que déplacer le réticule, l'infobulle ou une sélection en bande élastique.

Les propriétés habituelles de `Control` sont également publiques et inchangées : **Dock**, **Anchor**, **BackColor**, **Enabled**, **Visible**, **TabIndex**, **TabStop**, **ContextMenuStrip**, ainsi que les événements de souris, de glisser-déposer et de redimensionnement.

Méthodes

Méthode	Valeur de retour	Description
SaveToPNG(string aFileName)	void	Effectue le rendu de **Chart** à la taille client et à la résolution (DPI) actuelles du contrôle dans un nouveau fichier PNG. Ne fait rien lorsque **Chart** est null.
SaveToSVG(string aFileName)	void	Effectue le rendu de **Chart** à la taille client et à la résolution (DPI) actuelles du contrôle dans un fichier SVG. Ne fait rien lorsque **Chart** est null.

Exemple : tracer un graphique dans un formulaire WinForms

Voir aussi

- Présentation .NET
- Démarrage rapide : WPF
- TsgcVCLChart
- TsgcChartSeries_Line

Démarrage rapide : WPF

TsgcWpfChart est la vue WPF d'un modèle TsgcChart : une classe descendante de `Control` qui peint le graphique assigné à sa propriété de dépendance **Chart**. Elle est fournie dans le paquet **esegece.sgcCharts.Wpf**, ciblant `net472`, `net8.0-windows` et `net9.0-windows`, et elle porte une `ProjectReference` (ou une `PackageReference`, lorsqu'elle est consommée depuis NuGet) vers **esegece.sgcCharts.Rendering** de façon transitive, si bien qu'un projet WPF n'a besoin de référencer que **esegece.sgcCharts.Wpf** lui-même. Le contrôle est dessiné par le même canevas raster Skia que le TsgcVCLChart Delphi, directement dans le back buffer d'un `WriteableBitmap` persistant, limité à **Chart.MaxFPS**, et il est sûr de l'alimenter avec des données depuis un thread d'arrière-plan.

Propriétés

Propriété	Type	Description
Chart	TsgcChart	Le modèle de graphique que ce contrôle peint, adossé à la propriété de dépendance **ChartProperty**. L'assigner enregistre le contrôle comme écouteur tant qu'il est chargé ; l'effacer, ou libérer le graphique, empêche le contrôle de dessiner quoi que ce soit.
FrameCount	long (lecture seule)	Nombre d'images complètes rendues depuis la création du contrôle.
OverlayFrameCount	long (lecture seule)	Nombre d'images de superposition seule rendues depuis la création du contrôle, pour un changement qui ne fait que déplacer le réticule, l'infobulle ou une sélection en bande élastique.

Les propriétés habituelles de `Control` sont également publiques et inchangées : **Width**, **Height**, **HorizontalAlignment**, **VerticalAlignment**, **Background**, **IsEnabled**, **Visibility**, **Focusable**, ainsi que les événements routés de souris et de clavier.

Méthodes

Méthode	Valeur de retour	Description
SaveToPNG(string aFileName)	void	Effectue le rendu de **Chart** à la taille de rendu et à la résolution (DPI) actuelles du contrôle dans un nouveau fichier PNG. Ne fait rien lorsque **Chart** est null.
SaveToSVG(string aFileName)	void	Effectue le rendu de **Chart** à la taille de rendu et à la résolution (DPI) actuelles du contrôle dans un fichier SVG. Ne fait rien lorsque **Chart** est null.

Exemple : tracer un graphique dans une fenêtre WPF

Voir aussi

- Présentation .NET
- Démarrage rapide : WinForms
- TsgcVCLChart
- TsgcChartSeries_Line

Démarrage rapide : Avalonia

Package: esegece.sgcCharts.Avalonia (cible net8.0 et net9.0, dépend de **esegece.sgcCharts.Rendering**)
TsgcAvaloniaChart est la vue Avalonia d'un modèle **TsgcChart** : une classe descendante du `Control` Avalonia qui peint le graphique assigné à sa propriété **Chart**, sous Windows, Linux et macOS. Elle est l'équivalent .NET du TsgcFMXChart de Delphi. Le contrôle enregistre un écouteur sur le graphique tant qu'il est attaché à un arbre visuel et se redessine chaque fois que le modèle change, dessiné avec SkiaSharp dans un bitmap d'image persistant et affiché via le bail de canevas Skia du moteur de rendu Avalonia. Le modèle lui-même ne porte aucun code de dessin, si bien que le même graphique peut aussi être affiché par une vue WinForms ou WPF, ou être rendu sans interface graphique, en même temps.

Propriétés

Propriété	Type	Description
Chart	TsgcChart	Le modèle de graphique que ce contrôle peint. L'assigner enregistre le contrôle comme écouteur tant qu'il est attaché à une fenêtre ; l'effacer, ou libérer le graphique, empêche le contrôle de dessiner quoi que ce soit.
FrameCount	uint (lecture seule)	Nombre d'images complètes rendues depuis la création du contrôle.
OverlayFrameCount	uint (lecture seule)	Nombre d'images de superposition seule rendues depuis la création du contrôle (un changement de réticule, d'infobulle ou de sélection en bande élastique qui réutilise le dernier contenu mis en cache au lieu d'un redessin complet).

Les propriétés Avalonia habituelles de `Control` sont également disponibles sans changement : **Width**, **Height**, **Margin**, **HorizontalAlignment**, **VerticalAlignment**, **Focusable**, ainsi que les événements de pointeur, de molette et de clavier.

Méthodes

Méthode	Valeur de retour	Description
SaveToPNG(string aFileName)		Effectue le rendu de **Chart** à la taille et à la résolution (DPI) actuelles du contrôle dans un nouveau fichier PNG. Ne fait rien lorsque **Chart** est null.
SaveToSVG(string aFileName)		Effectue le rendu de **Chart** à la taille et à la résolution (DPI) actuelles du contrôle dans un fichier SVG. Ne fait rien lorsque **Chart** est null.

Le cadencement des images suit **Chart.MaxFPS** de la même manière que les autres vues visuelles : une notification de changement se contente de marquer le contrôle comme devant être redessiné et de programmer un redessin, au plus **Chart.MaxFPS** fois par seconde, si bien qu'un thread d'arrière-plan peut alimenter les séries du graphique sans inonder le thread d'interface utilisateur de dessins.

Exemple

Une application Avalonia minimale qui affiche un **TsgcAvaloniaChart** lié à un **TsgcChart** avec une série ligne, entièrement construite en code (aucun balisage XAML nécessaire). Outre **esegece.sgcCharts.Avalonia**, le projet a

aussi besoin des paquets NuGet **Avalonia.Desktop** et **Avalonia.Themes.Fluent**, comme toute autre application de bureau Avalonia :

Voir aussi

- Présentation .NET
- Démarrage rapide : WinForms
- Démarrage rapide : WPF
- TsgcFMXChart (Delphi)

Rendu sans interface graphique

Package: esegece.sgcCharts.Rendering (cible netstandard2.0, net472, net6.0, net8.0 et net9.0)
TsgcChartRenderer est la vue sans interface graphique d'un modèle **TsgcChart** : une simple classe sans dépendance à un framework d'interface utilisateur et sans fenêtre propre, qui effectue donc le rendu d'un graphique dans un fichier ou un flux PNG, un document SVG ou un tampon de pixels ARGB brut, depuis un programme console, un service ou une application ASP.NET. Elle fonctionne sous Windows, Linux et macOS, y compris dans un conteneur Linux ou une tâche d'intégration continue sans affichage, et elle est l'équivalent .NET du TsgcChartRenderer Delphi. N'ayant aucune surface visuelle, elle n'est pas affectée par **Chart.MaxFPS** : il n'y a ni boucle d'images ni minuteur, chaque appel de rendu peint le graphique de façon synchrone au moment même où il est appelé.

Propriétés

Propriété	Type	Description
Chart	TsgcChart	Le modèle de graphique que ce moteur de rendu peint. Chaque méthode Render* lève **EsgcChartRendererError** lorsque cette valeur est null.
Width	int	Largeur logique par défaut en pixels utilisée par les surcharges sans taille. Par défaut 800.
Height	int	Hauteur logique par défaut en pixels utilisée par les surcharges sans taille. Par défaut 500.
Scale	double	Pixels physiques par pixel logique. Par défaut 1. La sortie fait Round(Width * Scale) par Round(Height * Scale) pixels physiques ; **Chart.Scale** est réglé à cette valeur pour le rendu puis restauré à sa valeur précédente ensuite.
CompressPNG	bool	Indique si la sortie PNG est compressée par déflation via l'encodeur PNG de Skia. Par défaut true.

Méthodes

Méthode	Valeur de retour	Description
RenderToPNGFile(string aFileName) RenderToPNGFile(string aFileName, int aWidth, int aHeight)		Effectue le rendu dans un fichier PNG, à la taille **Width**/**Height** ou à la taille logique indiquée sans modifier **Width** et **Height**.
RenderToPNGStream(Stream aStream) RenderToPNGStream(Stream aStream, int aWidth, int aHeight)		Effectue le rendu PNG (RGBA, 8 bits) vers un flux.
RenderToSVG() RenderToSVG(int aWidth, int aHeight)	string	Effectue le rendu et retourne un document SVG 1.1 (ASCII pur) sous forme de chaîne.
RenderToSVGFile(string aFileName) RenderToSVGFile(string aFileName, int aWidth, int aHeight)		Effectue le rendu SVG vers un fichier.
RenderToSVGStream(Stream aStream) RenderToSVGStream(Stream aStream, int aWidth, int aHeight)		Effectue le rendu SVG vers un flux.

RenderToBuffer(ref uint[] aBuffer, out int aPixelWidth, out int aPixelHeight) RenderToBuffer(ref uint[] aBuffer, out int aPixelWidth, out int aPixelHeight, int aWidth, int aHeight)		Effectue le rendu en pixels 0xAARRGGBB droits (non prémultipliés), de haut en bas, avec un pas de ligne égal à **aPixelWidth** ; **aPixelWidth** et **aPixelHeight** retournent la taille physique.

Threads

Chaque méthode de rendu peut être appelée depuis n'importe quel thread. Chaque **TsgcChartRenderer** possède ses propres canevas, un raster et un SVG, protégés par un verrou propre à cette instance de moteur de rendu. En plus de cela, chaque rendu effectué par un **TsgcChartRenderer** quel qu'il soit dans le processus passe par un verrou supplémentaire partagé par tout le processus, car **TsgcChart.Render** conserve l'état de mise en page sur le graphique lui-même pendant son exécution. Pendant qu'un rendu est en cours, le graphique ne doit pas être modifié structurellement, c'est-à-dire qu'aucune série ni aucun axe ne doit être ajouté, supprimé ou libéré, et qu'aucune propriété affectant la mise en page ne doit être modifiée ; ajouter des données via les tampons des séries est sûr, car les séries dessinent à partir d'un instantané pris sous le verrou propre du tampon. Une vue visuelle (**TsgcAvaloniaChart**, une vue WinForms ou WPF) ne prend pas le verrou de rendu partagé ; il ne faut donc pas effectuer le rendu du même graphique avec un moteur de rendu dans un thread pendant qu'une vue visuelle le peint au même moment.

Exemple : rendu sans interface graphique vers PNG sous Linux

Une application console sans référence à un framework d'interface utilisateur, compilable et exécutable sous Linux, qui effectue le rendu d'un graphique dans un fichier PNG :
Ceci s'exécute sans modification dans un conteneur Linux ou une tâche d'intégration continue : **esegece.sgcCharts.Rendering** embarque ses propres ressources natives SkiaSharp pour Linux, si bien qu'aucun serveur X, GPU ni installation fontconfig n'est requis.

Exemple : pie et treemap rendus sans interface graphique

Les séries pie, radar, heatmap, treemap et funnel (voir Series Gallery) se rendent sans interface graphique comme n'importe quelle autre série. Un moteur de rendu ne dessine pas de superposition au survol, donc la sortie ne montre que l'état porté par le modèle : définissez-le dans le code, tel qu'un clic dans une vue le laisserait, avec **Exploded** sur une part de pie ou **DrillPath** sur un treemap. Un même moteur de rendu peint plusieurs graphiques l'un après l'autre quand **Chart** change :
Le SVG de ces séries est identique, octet pour octet, au SVG que le **TsgcChartRenderer** Delphi écrit pour le même graphique et la même taille. Un heatmap dont **ImagePath** est à true remplit ses cellules comme une seule image, que le SVG incorpore en PNG. La démo **SeriesHeadless** de `demos\Charts` rend ainsi chaque graphique de la galerie de séries : `SeriesHeadless [folder] [-dark] [-width N] [-height N] [-scale N].`

Voir aussi

- Présentation .NET
- Démarrage rapide : WinForms
- Series Gallery
- TsgcChartRenderer (Delphi)

Series Gallery

Package: esegece.sgcCharts (cible netstandard2.0, net472, net6.0, net8.0 et net9.0)
Les séries pie, radar, heatmap, treemap et funnel sont les équivalents .NET des séries Delphi de même nom. Elles se trouvent dans le paquet principal **esegece.sgcCharts**, donc elles se dessinent dans toutes les vues (**TsgcWinFormsChart**, **TsgcWpfChart** et **TsgcAvaloniaChart**) et sans interface graphique via **TsgcChartRenderer** (voir Headless Rendering), avec les mêmes propriétés, valeurs par défaut et mise en page que dans Delphi. La rubrique Delphi de chaque série, liée ci-dessous, décrit chaque propriété. Cette page montre comment les utiliser depuis C# et énumère les endroits où la forme .NET de l'API diffère.

Séries

Classe	Mode de tracé	Description
TsgcChartSeries_Pie	cpmCell	Un camembert, ou un donut quand **InnerRadiusPercent** est supérieur à 0. Les parts inférieures à **OthersThreshold** pour cent sont fusionnées en une part Others, **ExplodeOnClick** détache une part au clic, et les étiquettes se placent à l'intérieur ou à l'extérieur avec des lignes de repère. Éléments : **Slices**, un **TsgcChartPieSlice** chacun.
TsgcChartSeries_Radar	cpmPolar	Un polygone par série sur des rayons partagés par toutes les séries radar du graphique, **Filled** ou en simple contour, sur une grille polygonale ou circulaire (**GridShape**). Éléments : **Points**, un par rayon.
TsgcChartSeries_Heatmap	cpmXY	Une grille de cellules colorées selon une échelle de couleurs : un spectrogramme en direct alimenté ligne par ligne avec **AppendLine** (**Capacity** conserve les lignes les plus récentes), ou une matrice fixe avec lignes et colonnes étiquetées sur des axes de catégories. Données : **Data**, un **TsgcChartHeatmapData**.
TsgcChartSeries_Treemap	cpmCell	Des rectangles imbriqués construits à partir de chemins tels que `Europe/Spain/Madrid` (**PathDelimiter**), avec un en-tête par groupe (**Levels**), un forage au clic (**DrillOnClick**) et un fil d'Ariane pour remonter. Éléments : **Nodes**, un par feuille.
TsgcChartSeries_Funnel	cpmCell	Une étape par élément dans l'ordre de la collection, qui se rétrécit jusqu'à un goulot (**NeckWidthPercent**, **NeckHeightPercent**). Les étapes ont la même hauteur, ou une hauteur proportionnelle à la valeur avec **ProportionalHeight**. Éléments : **Stages**.

Le mode de tracé détermine quelles séries peuvent partager un graphique : voir Plot Areas. Un graphique peut contenir un pie et un funnel côte à côte, chacun dans sa propre zone de cellules, alors qu'un heatmap partage le tracé XY et ses axes avec les autres séries XY.

Éléments et valeurs en direct

Membre	Description
AddItem(string aText, double aValue, uint aColor = 0)	Pie, radar, treemap et funnel. Ajoute un élément et renvoie son index. Une couleur 0 prend la couleur de palette de l'index de l'élément, si bien qu'un élément garde sa couleur pendant que les valeurs changent.

SetValueAt(int aIndex, double aValue) DeleteItem(int aIndex)	Modifie ou supprime un élément. L'ordre des éléments ne change pas. Un index hors limites lève `ArgumentOutOfRangeException`.
ItemCount()	Le nombre d'éléments, une méthode comme toute fonction Delphi sans paramètre du portage.
Slices, Points, Nodes, Stages	La collection d'éléments. Son indexeur renvoie un **TsgcChartCategoryItem** (**Text**, **Value**, **Color**) ; convertissez-le en **TsgcChartPieSlice** pour atteindre **Exploded**.
BeginUpdate() EndUpdate()	Regroupe de nombreuses modifications en un seul repaint et une seule mise en page.
AppendLine(double[] aValues) AppendLine(double aKey, double[] aValues)	Heatmap. Ajoute une ligne de cellules, identifiée par **aKey** ou par le compteur de lignes courant du magasin. Les valeurs forment un tableau, pas une liste `params`.
SetLineLabels(string[] aLabels) SetCellLabels(string[] aLabels)	Heatmap. Les étiquettes des lignes et des cellules sur un axe de catégories. Une surcharge accepte un **TsgcChartStrings**, la forme .NET du `TStrings` de Delphi, donc un `null` nu est ambigu : castez-le.
Cells(int aLine, int aCell) SetCells(int aLine, int aCell, double aValue)	Heatmap. Lit et écrit une cellule, la forme méthode de la propriété indexée Delphi `Cells[Line, Cell]`.

Interaction

Les vues transmettent la souris à **TsgcChart.Interaction**, donc les info-bulles au survol, l'explode et le drill fonctionnent de la même façon dans WinForms, WPF et Avalonia. Un clic sur une série appelle d'abord **TsgcChart.OnSeriesClick**, puis la série le traite.

Membre	Description
TsgcChartSeries_Pie.ExplodeOnClick	Par défaut false. À true, un clic sur une part bascule son indicateur **Exploded**, ou **OthersExploded** pour la part Others.
TsgcChartSeries_Treemap.DrillOnClick	Par défaut true. Un clic sur une tuile descend d'un niveau, dans le groupe sous le clic, et un clic sur un élément du fil d'Ariane remonte jusqu'à lui. **DrillDown(string aPath)**, **DrillUp()**, **DrillReset()** et **DrillPath** font la même chose depuis le code.
TsgcChartSeries_Treemap.OnDrill	Un **TsgcChartNotifyEvent** (`object Sender`) appelé quand **DrillPath** change. Une propriété délégué : affecter un gestionnaire remplace le précédent, comme dans Delphi.
TsgcChart.OnSeriesClick	Un **TsgcChartHitEvent** (`object Sender, TsgcChartHitInfo aHit`). Pour ces séries, **aHit.SubIndex** est l'index de l'élément (-2 pour la part Others d'un pie), **aHit.Text** le texte de la part ou de l'étape ou le chemin du treemap, et **aHit.Percent** la part de l'élément dans un pie, un funnel ou un treemap.

La superposition au survol (la mise en évidence et l'info-bulle) n'est dessinée qu'à l'écran. **TsgcChartRenderer**, **SaveToPNG** et **SaveToSVG** ne rendent que le modèle, donc une part détachée ou un treemap exploré apparaît dans leur sortie, alors qu'une part survolée n'y apparaît pas.

Exemple : construire les cinq séries

Une classe utilitaire qui construit un graphique par série, les mêmes graphiques que la démo **SeriesGallery** avec moins de données :

Exemple : héberger les graphiques dans une fiche WinForms

Un **TsgcWinFormsChart** par onglet, une légende qui suit les clics et le chemin de drill, et des valeurs en direct. Avec WPF, utilisez **TsgcWpfChart**, avec Avalonia **TsgcAvaloniaChart** : le code du graphique ne change pas.

Démos

Démo	Framework	Description
SeriesGallery	WinForms, net8.0-windows	Le pie, le radar, un spectrogramme de 30 lignes par seconde, un heatmap matriciel, le treemap et le funnel, un **TsgcWinFormsChart** par onglet, avec un commutateur donut, des options de radar rempli ou circulaire et des boutons Up et Top pour le treemap.
SeriesGalleryAvalonia	Avalonia, net8.0	La même galerie dans des contrôles **TsgcAvaloniaChart**, sous Windows, Linux et macOS.
SeriesHeadless	Console, net8.0	Les graphiques de la galerie rendus avec **TsgcChartRenderer** vers des fichiers PNG et SVG, sans fenêtre : `SeriesHeadless [folder] [-dark] [-width N] [-height N] [-scale N]` .

Les démos se trouvent dans `demos\Charts`. Les deux galeries acceptent `-seconds N` comme autotest : elles affichent chaque onglet, cliquent sur une part du pie et sur une tuile du treemap par l'interaction du graphique, se ferment après N secondes et renvoient le code de sortie 0 quand toutes les vérifications ont réussi.

Voir aussi

- .NET Overview
- .NET Differences from Delphi
- Headless Rendering
- Quick Start: WinForms
- Plot Areas
- Colour Scale

Pont

Le paquet **esegece.sgcCharts.sgcWebSockets** (ciblant `netstandard2.0`, `net8.0`, `net9.0`) relie les sources de données sgcCharts à une connexion sgcWebSockets active : messages WebSocket, publications MQTT, Server-Sent Events et flux de marché Binance. C'est un paquet distinct du paquet de base **esegece.sgcCharts** et il nécessite sgcWebSockets pour compiler ; un projet qui n'utilise que les sources de données de base n'a besoin ni de ce paquet ni de sgcWebSockets.
Par défaut, le paquet référence directement le projet du portage .NET de sgcWebSockets ; une compilation sans cette arborescence source peut à la place référencer le paquet NuGet **esegece.sgcWebSockets**. Ce pont n'est pas fourni dans l'édition Community.

MQTT : TsgcChartDataSource_MQTT

Trace les publications MQTT reçues par un **TsgcWSProtocol_MQTT_Client**. Tant que la source est **Active** et que **Client** est assigné, elle abonne son propre gestionnaire à **Client.OnMQTTPublish**.

Propriété	Type	Description
Client	TsgcWSProtocol_MQTT_Client	Le client MQTT dont les publications sont tracées.
TopicFilter	string	Filtre de sujet MQTT, avec les jokers `+` (un niveau) et `#` (ce niveau et tous les niveaux inférieurs). Un filtre vide trace chaque publication.
PayloadFormat	TsgcChartMQTTPayloadFormat (cpfJSON, cpfNumber)	**cpfJSON** lit la charge utile via **Mapping** (un objet JSON ou un tableau d'enregistrements). **cpfNumber** interprète la charge utile comme un simple nombre pour Y et utilise l'heure locale actuelle pour X.
Active	bool (masque la propriété de base)	La source redéclare **Active** afin que sa modification mette aussi à jour le gestionnaire sur **Client**, de la même manière que l'assignation de **Client**.

WebSocket : TsgcChartDataSource_WebSocket

Trace les messages JSON reçus par un **TsgcWebSocketClient**. Tant que la source est **Active** et que **Client** est assigné, elle abonne son propre gestionnaire à **Client.OnMessage**. Chaque message texte est lu comme du JSON via **Mapping**.

Propriété	Type	Description
Client	TsgcWebSocketClient	Le client WebSocket dont les messages sont tracés.
MessageFilter	string	Un chemin JSON qui doit exister dans un message pour qu'il soit tracé, par exemple `"k"` pour ne garder que les messages kline de Binance. Un filtre vide trace chaque message qui s'interprète comme du JSON.
Active	bool (masque la propriété de base)	Redéclarée comme dans **TsgcChartDataSource_MQTT**, si bien que le rattachement suit la valeur.

Server-Sent Events : TsgcChartDataSource_SSE

Trace les Server-Sent Events provenant de deux sources indépendantes, qui peuvent être utilisées ensemble sur la même source de données.

- **Client** est un **TsgcHTTP1Client**, comme en Delphi. L'application démarre le flux avec la méthode bloquante `GetSSE(URL)` du client, généralement depuis un thread de travail. Tant que la source est **Active** et que **Client** est assigné, elle abonne son propre gestionnaire à **Client.OnSSEMessage**, qui délivre le texte brut du flux au fur et à mesure de son arrivée. La source met en mémoire tampon les blocs, les découpe en événements à chaque ligne vide, joint les lignes `data:` d'un événement, et lit le résultat comme du JSON via **Mapping**.
- **SSEClient** est un **TsgcSSEClient**, le client managé de style EventSource. Il n'a pas d'équivalent Delphi dans ce pont. Son **OnEvent** délivre des événements déjà interprétés, si bien que la source lit directement le **Data** de l'événement comme du JSON via **Mapping**, sans aucune analyse de blocs.

Propriété	Type	Description
Client	TsgcHTTP1Client	Client SSE bloquant. Voir ci-dessus.
SSE-Client	TsgcSSEClient	Client EventSource managé, un ajout propre au .NET. Voir ci-dessus.
Active	bool (masque la propriété de base)	Redéclarée comme dans **TsgcChartDataSource_MQTT**. **Client** et **SSE-Client** sont tous deux rattachés ou détachés indépendamment à mesure que la propriété change.

Une méthode **ResetStream** abandonne un événement mis en mémoire tampon, encore incomplet, de l'analyseur de blocs de **Client**, par exemple après une reconnexion.

Binance : TsgcChartDataSource_Binance

S'abonne à un flux de données de marché public Binance via un **TsgcWSAPI_Binance** et le trace. Les flux de marché Binance n'ont pas d'événement typé : ils arrivent en JSON brut sur l'événement **OnMessage** du client WebSocket auquel l'API est attachée (**API.Client**). Tant que la source est **Active** et que **API** est assigné, elle y abonne son propre gestionnaire ; le rattachement suit **API.Client** chaque fois que **API** ou **Active** est défini, ainsi qu'au moment de **Subscribe**.

Pro-priété	Type	Description
API	TsgcWSAPI_Binance	Le composant API Binance dont le flux est tracé.
Symbol	string	Le symbole Binance, par exemple `"BTCUSDT"`. Automatiquement mis en minuscules pour l'abonnement, car les noms de flux Binance sont exclusivement en minuscules. Les messages pour un autre symbole sont ignorés.
Stream	TsgcChartBinanceStream (cbsKLine, cbsTicker, cbsTrade)	Quel événement Binance est tracé : les klines (chandeliers), le ticker 24h (dernier prix), ou les transactions individuelles.
Interval	TsgcWSBinanceChartIn-tervals	Intervalle de kline, par exemple `bci1m`. Utilisé uniquement pour **cbsKLine** ; les messages pour un autre intervalle sont ignorés.
Close-dOnly	bool	Uniquement pour **cbsKLine**. Lorsque true, seuls les chandeliers clô-turés sont tracés ; sinon, le chandelier encore en formation est mis à jour sur place à chaque message.
Aggre-gator	TsgcChartAggregator (édition financière)	Uniquement pour **cbsTrade**. Lorsqu'il est assigné, chaque transac-tion va vers **Aggregator.AddTrade** au lieu du tampon de la série : la transaction elle-même n'est pas tracée, et la série OHLC de l'agrégateur dessine les chandeliers (volume inclus). Libérer

		l'agrégateur n'efface pas cette propriété ; un agrégateur libéré ignore les transactions qu'il continue de recevoir.
Active	bool (masque la propriété de base)	Redéclarée comme dans **TsgcChartDataSource_MQTT**.

Événements

Événement	Type	Description
OnTrade	TsgcChartBinanceTradeEvent(object Sender, double aTime, double aPrice, double aQty)	Uniquement pour **cbsTrade**. Se déclenche pour chaque transaction de **Symbol**, avec son heure UTC, son prix et sa quantité, avant que la transaction ne soit ajoutée au tampon ou transmise à **Aggregator**.

Méthodes

Méthode	Valeur de retour	Description
Subscribe()	int	S'abonne à **Stream** pour **Symbol** (et **Interval** pour les klines) en appelant `SubscribeKLine`, `SubscribeTicker` ou `SubscribeTrades` de **API**. Retourne l'identifiant que l'API retourne.
Unsubscribe()	int	L'appel de désabonnement correspondant.

Événements multicast

Chaque source de données du pont se rattache à un événement du client qui est un simple pointeur de méthode Delphi dans la bibliothèque Delphi (**OnMQTTPublish**, **OnMessage**, **OnSSEMessage**, **OnEvent**) mais un événement multicast .NET ici. La source Delphi sauvegarde le gestionnaire déjà assigné le cas échéant, installe le sien, et restaure celui sauvegardé lorsqu'elle se détache. Une source de données .NET ajoute au contraire son propre gestionnaire à la liste d'invocation avec `+=` lors du rattachement, et retire uniquement ce même gestionnaire avec `-=` lors du détachement ; chaque gestionnaire d'application déjà abonné, ou abonné par la suite, continue de se déclencher et n'est jamais touché par la source.
La différence pratique tient à l'ordre d'abonnement plutôt qu'à la présence : les gestionnaires s'exécutent dans l'ordre où ils ont été ajoutés, si bien qu'un gestionnaire d'application abonné au client avant que la source de données ne se rattache s'exécute avant que la source de données ne trace le message, alors qu'en Delphi la source trace toujours en premier. Le rattachement ne se produit qu'à l'exécution, jamais à la conception. Il n'existe pas de **FreeNotification** en .NET, donc libérer le client sous-jacent n'efface pas la propriété **Client** (ou **API**, ou **SSEClient**) de la source de données ; mettez-la à null, ou libérez la source de données, à la place. Sur **TsgcChartDataSource_SSE**, un gestionnaire d'application sur **OnSSEMessage** peut toujours définir **Cancel**, qui est partagé par toute la liste d'invocation, si bien que tout gestionnaire qui le met à true arrête le flux pour tous les autres gestionnaires également.
Les messages, publications et événements arrivent tous sur le thread depuis lequel le client sous-jacent notifie. Les ajouts passent par le verrou et la notification limitée en fréquence de la classe de base **TsgcChartDataSource**, si bien que tout mode de notification peut être utilisé sans risque avec une source de données du pont.

Exemple : graphique Binance en direct

Voir aussi

- Présentation du pont (Delphi)

- Présentation .NET

Différences .NET par rapport à Delphi

sgcCharts .NET conserve les noms de classes, les propriétés et le comportement de Delphi partout où les deux langages le permettent (voir Présentation .NET). Cette page recense les endroits où le portage .NET se comporte véritablement différemment de la bibliothèque Delphi, et non les simples cas où un nom ou un type a changé pour son équivalent naturel en C#.

Événements

Domaine	Delphi	.NET
Sources de données du pont en direct (**esegece.sgcCharts.sgcWebSockets**) : **OnMQTTPublish**, **OnMessage** (client WebSocket et client Binance), **OnSSEMessage** / **OnEvent**	Un simple pointeur de méthode. La source de données sauvegarde le gestionnaire actuel du client, installe le sien, et restaure celui sauvegardé lorsqu'elle s'arrête.	Un véritable `event` multicast C# (`+=` / `-=`). La source de données abonne et désabonne son propre gestionnaire au lieu d'en sauvegarder et restaurer un ; voir Présentation du pont pour le mécanisme complet.
Modèle de graphique de base : **TsgcChart.OnChange**, **TsgcChartDataSource.OnDataChanged** / **OnBeforeAppend**, **TsgcChart.OnCrosshairMove** / **OnSeriesClick** / **OnZoom**, ainsi que les membres similaires sur les axes, les séries et les dessins	Un simple pointeur de méthode, remplacé par affectation.	Aucune différence : ces éléments restent de simples propriétés de délégué `get; set;` , pas un `event` C#, si bien qu'assigner un nouveau gestionnaire remplace toujours le précédent exactement comme en Delphi. Mentionné ici uniquement pour contraster avec la ligne du pont ci-dessus, puisque les deux sont nommés **OnX**.

Propriétés indexées

Domaine	Delphi	.NET
TsgcChartAxes, **TsgcChartPanes**, **TsgcChartDrawings**, **TsgcChartPalette**, **TsgcChartJSONNode**	Une propriété tableau indexée, par exemple `Axes[Index]` .	Un véritable indexeur C# : `this[int Index]` .
TsgcChartBuffer (`X` , `Y`), **TsgcChartOHLCBuffer** (`X` , `Open` , `High` , `Low` , `Close` , `Volume`), les `XValues` et `Values` des séries financières	Une propriété tableau indexée par colonne, par exemple `Open[Index]` , et une propriété bidimensionnelle, `Values[Line, Index]` .	Une méthode ordinaire prenant l'index en paramètre : `Open(Index)` , `Values(Line, Index)` . Une classe C# ne peut déclarer qu'une seule signature d'indexeur (`this[...]`), et ces classes ont besoin de plusieurs accesseurs par index nommés indépendamment, ou à plusieurs paramètres, sur la même classe ; ils deviennent donc des méthodes à la place.

Les séries pie, radar, heatmap, treemap et funnel et leurs classes auxiliaires : **TsgcChartSeries_Radar.SpokeText**, **TsgcChartSeries_Funnel.DrawnStages**, **TsgcChartSeries_Heatmap.Cells**, **TsgcChartHeatmapData.Cells**, **TsgcChartColorScale.LUT**, **TsgcChartPieLayout.Slices**, **TsgcChartTreemapLayout.Nodes**, **Crumbs** et **DrawnNodes**	Une propriété tableau indexée, p. ex. `SpokeText[Index]`, `DrawnStages[Index]`, et la propriété en lecture et écriture `Cells[Line, Cell]`.	Une méthode : `SpokeText(Index)`, `DrawnStages(Index)` (une copie de l'enregistrement de l'étape), `Cells(Line, Cell)` pour lire et `SetCells(Line, Cell, Value)` pour écrire. `DrawnStages` lève `ArgumentOutOfRangeException` là où Delphi lève `EListError`. Les méthodes de mise en page `Slices`, `Nodes`, `Crumbs` et `DrawnNodes` la lèvent aussi pour un index hors de `0..Count - 1`, là où Delphi lit le tableau interne sans contrôle.

Séries pie, radar, heatmap, treemap et funnel

Domaine	Delphi	.NET
SetValueAt et **DeleteItem** des séries pie, radar, treemap et funnel	Un index hors limites lève `ERangeError`.	Un index hors limites lève `ArgumentOutOfRangeException`.
TsgcChartSeries_Heatmap: **AppendLine**, **SetLineLabels**, **SetCellLabels**	Un paramètre tableau ouvert (`AppendLine([1, 2, 3])`) et une surcharge `TStrings` pour les étiquettes.	Un paramètre `double[]` ou `string[]`, pas une liste `params`, et une surcharge **TsgcChartStrings** pour les étiquettes. Un `null` nu est ambigu entre les deux surcharges d'étiquettes, donc castez-le.
Les éléments (**Slices**, **Points**, **Nodes**, **Stages**) et les **Levels** du treemap	Modifiés dans l'inspecteur d'objets et stockés dans la fiche (DFM).	Il n'y a pas de streaming du concepteur de fiches : ajoutez les éléments dans le code avec **AddItem** et les niveaux avec **Levels.Add()**. Voir Series Gallery.
Sortie raster (**RenderToPNGFile**, **RenderToBuffer**, **SaveToPNG**)	Dessinée par le canvas raster de sgcUICore.	Dessinée par SkiaSharp, donc les pixels lissés des bords peuvent différer légèrement de Delphi. La mise en page est la même, et la sortie SVG est égale au SVG Delphi octet pour octet.

Chargement depuis un DataSet ou un DataReader

TsgcChartDataSource_DataSet charge les données depuis un `System.Data.DataTable` ou tout `IDataReader`, les équivalents .NET du `TDataSet` Delphi. Un `DataTable` n'a pas de curseur, il n'y a donc pas d'enregistrement courant à sauvegarder et restaurer après le chargement : la gestion des signets de la source de données Delphi, qui ramène le `TDataSet` source à l'enregistrement où il se trouvait avant l'exécution de **Refresh**, n'a pas d'équivalent en .NET.

Éditions

sgcCharts Delphi est fourni en éditions Community, Trial et Registered. sgcCharts .NET n'est fourni qu'en **Community** (compilée avec `-p:SgcCommunity=true`) et **Registered** (la compilation par défaut) : il n'existe pas de compilation d'essai .NET ni de constante d'essai.

Voir aussi

- Présentation .NET
- Series Gallery
- Présentation du pont
- TsgcChartDataSource_DataSet

Dépannage

Aucun problème connu n'est documenté pour l'instant.

Introduction

sgcMaps est une bibliothèque de composants Delphi pour les cartes interactives. Les contrôles de carte dessinent nativement les tuiles raster et vectorielles, sans contrôle de navigateur et sans JavaScript, et affichent par-dessus des marqueurs, des itinéraires, des formes, des étiquettes et des cartes de chaleur. Des composants non visuels ajoutent le géocodage, le calcul d'itinéraires, la recherche de lieux, l'altitude, les fuseaux horaires et la localisation de l'appareil, et le calculateur d'itinéraires hors ligne ainsi que le géocodeur hors ligne répondent sans aucune connexion réseau.
Un contrôle VCL, un contrôle FireMonkey et un contrôle Lazarus (LCL) partagent un même modèle de carte, si bien que le code qui ajoute une couche, un marqueur ou un itinéraire est identique sur chaque framework. Le contrôle Lazarus ne peut pas télécharger de tuiles, il n'affiche donc, sous Lazarus, que les tuiles d'un dossier. Un contrôle de carte peut aussi dessiner sans fenêtre, ce qui transforme une application console ou un service en moteur de rendu de cartes statiques.

- Présentation : ce que contient sgcMaps, et la version de Delphi requise par chaque fonctionnalité.
- Installation : l'installateur sgcUI, les packages et la configuration requise.
- Démarrage rapide VCL et Démarrage rapide FMX : une carte fonctionnelle en quelques lignes.
- Contrôles de carte : TsgcVCLMap et TsgcFMXMap, leurs couches et leurs surcouches.
- Services en ligne : géocodage, itinéraires, lieux, altitude et fuseaux horaires.
- Itinéraires : modification interactive d'un itinéraire et calculateur hors ligne.
- Géocodage hors ligne : recherche de lieux à partir d'un index GeoNames local.
- Localisation : la position de l'appareil, réelle ou simulée.
- .NET : sgcMaps for .NET, le portage C# avec un contrôle Windows Forms, le rendu sans interface graphique et les services hors ligne.
- Démos, Outils, Lazarus et Dépannage : les projets d'exemple, les outils de données, le contrôle LCL et les pièges courants.

Présentation

sgcMaps repose sur un **modèle de carte** : un ensemble de collections qui décrivent ce qu'une carte affiche. Layers contient les sources de tuiles, et Markers, Polylines, Polygons, Rectangles, Circles, Labels et Heatmaps contiennent les surcouches. Options et MarkerOptions règlent le comportement de la vue. Les classes du modèle se trouvent dans **sgcMaps_Classes** et sont identiques dans TsgcVCLMap et TsgcFMXMap.
Les composants non visuels travaillent avec ces collections plutôt qu'avec un contrôle particulier. TsgcMapRouteCalculator conserve ses points de passage dans les Markers d'une carte et son itinéraire dans ses Polylines, et TsgcMapLocation tient à jour un marqueur de position et un cercle de précision. Le même composant sert donc aussi bien un formulaire VCL qu'un formulaire FireMonkey.

Fonctionnalités

Tuiles et couches

- Tuiles raster en ligne issues de fournisseurs intégrés : OpenStreetMap, OpenTopoMap, Esri World Imagery, Esri World Street, CARTO Light et Dark, MapTiler Streets et Satellite, Thunderforest Cycle et Transport, ainsi que Stadia Alidade Smooth et Smooth Dark.
- Tout autre serveur de tuiles, par un modèle d'URL avec les paramètres {z}, {x}, {y}, sous-domaine, retina et clé d'API.
- Un cache disque par couche, si bien que les tuiles déjà vues s'affichent sans connexion.
- Tuiles hors ligne depuis un dossier z/x/y ou depuis un fichier MBTiles.
- Tuiles vectorielles (Mapbox Vector Tiles) depuis un serveur ou un fichier MBTiles local, dessinées avec un style MapLibre ou Mapbox GL.
- Plusieurs couches avec opacité, plage de zoom, rôle de couche de base ou de surcouche, et un sélecteur de couches facultatif sur la carte.
- Un téléchargeur de tuiles qui remplit un dossier avec toutes les tuiles d'une zone et d'une plage de zoom, pour créer un pack de carte hors ligne.

Surcouches

- Marqueurs avec texte de fenêtre contextuelle et d'infobulle, couleur, icônes bitmap personnalisées, ancrages et déplacement à la souris.
- Regroupement de marqueurs, pour que des milliers de marqueurs restent lisibles et rapides.
- Polylignes (pleines, en pointillés ou géodésiques), polygones avec trous, rectangles et cercles, tous modifiables à la souris.
- Étiquettes de texte et cartes de chaleur pondérées avec plusieurs dégradés de couleurs.

Interaction et sortie

- Zoom par glissement, molette et boutons, rotation de la carte (orientation), une échelle graphique et l'attribution des tuiles.
- Événements pour les clics, les déplacements de marqueurs, les modifications de formes, les clics sur les regroupements, les changements de couche, les changements de zoom et les erreurs de tuiles.
- Rendu vers un bitmap, un fichier PNG ou BMP, y compris depuis un contrôle sans fenêtre parente.

Données

- Import et export GeoJSON, GPX et KML des marqueurs, des polylignes et des polygones.
- Fonctions de géométrie : distance, orientation, point de destination, interpolation géodésique, point dans un polygone, simplification de ligne et point le plus proche sur une polyligne.
- Un lecteur en flux continu des fichiers OpenStreetMap .osm.pbf et .osm XML.

Services

- Géocodage et géocodage inverse avec Nominatim, Google, HERE, Azure Maps, Mapbox, OpenRouteService et Geoapify.
- Itinéraires avec OSRM, OpenRouteService, Google Routes, HERE, Azure Maps, Mapbox et Geoapify.
- Saisie semi-automatique de lieux avec Photon, Nominatim, Google, HERE, Mapbox et Geoapify.
- Altitude de points et de tracés avec Open-Elevation, OpenTopoData et Google.
- Fuseau horaire d'une coordonnée avec TimeAPI, GeoNames, Google et Azure Maps.

Itinéraires, données hors ligne et localisation

- Modification interactive d'un itinéraire avec des points de passage déplaçables et des itinéraires alternatifs.
- Calcul d'itinéraires hors ligne en voiture, à vélo et à pied, sur un graphe construit à partir des données OpenStreetMap.
- Géocodage hors ligne sur un index SQLite construit à partir des exports GeoNames.
- Localisation de l'appareil sous Windows, Android, iOS et macOS, avec en plus la simulation de position et de trace pour les tests.

Versions de Delphi

Le modèle de carte, le contrôle VCL, GPX, KML et les fonctions de géométrie se compilent de Delphi 7 à Delphi 13, et les packages couvrent aussi C++Builder 2007 à C++Builder 13. Certaines fonctionnalités dépendent de parties de la RTL apparues plus tard, si bien que leurs composants ne sont enregistrés que dans les versions de l'IDE qui les possèdent.

Fonctionnalité	Unité	Version minimale
Contrôle de carte VCL, surcouches, GPX, KML, géométrie	sgcMaps_VCL, sgcMaps_Classes	Delphi 7
Sortie PNG	sgcMaps_VCL	Delphi 2009
Tuiles depuis un serveur, téléchargeur de tuiles	sgcMaps_TileSource_HTTP, sgcMaps_TileDownloader	Delphi 2009 (Indy 10 ; THTTPClient à partir de Delphi XE8)
Calculateur d'itinéraires, calculateur hors ligne, tuiles vectorielles depuis un serveur	sgcMaps_RouteCalculator, sgcMaps_OfflineRouter, sgcMaps_TileSource_MVT	Delphi XE2
Contrôle de carte FireMonkey, localisation de l'appareil, GeoJSON	sgcMaps_FMX, sgcMaps_Location, sgcMaps_GeoJSON	Delphi XE6
MBTiles, MBTiles vectoriel, géocodeur hors ligne	sgcMaps_TileSource_MBTiles, sgcMaps_OfflineGeocoder	Delphi XE7 (FireDAC SQLite)
Services en ligne	sgcMaps_Geocoding, sgcMaps_Directions, sgcMaps_Places, sgcMaps_Elevation, sgcMaps_TimeZone	Delphi XE8 (THTTPClient, System.JSON)

Sous Lazarus, le package contient le contrôle de carte LCL, la source de tuiles depuis un dossier, les fournisseurs, le moteur raster et les fonctions de géométrie. Le contrôle LCL ne peut pas télécharger de tuiles, il n'affiche donc que les tuiles d'un dossier, et GeoJSON, GPX et KML ne sont pas compilés sous FPC. Voir Lazarus.

.NET

sgcMaps a aussi un portage .NET, sgcMaps for .NET, pour .NET 8 et .NET 9. Il a le même modèle de carte, les mêmes services en ligne et les mêmes fichiers de données hors ligne, un contrôle Windows Forms et un moteur de rendu sans interface graphique qui fonctionne aussi sous Linux et macOS. Voir la présentation .NET.

Installation

sgcMaps est fourni dans l'**installateur sgcUI** unifié, avec les autres produits visuels d'eSeGeCe. L'installateur connaît les produits inclus dans votre licence, il n'y a donc rien à choisir à la main en dehors des IDE.

Installation avec l'installateur sgcUI

1. Fermez toutes les instances de l'IDE en cours d'exécution et lancez l'installateur sgcUI.
2. Connectez-vous avec votre compte eSeGeCe.
3. L'installateur lit votre licence et liste les produits qu'elle inclut. sgcMaps est coché lorsque votre licence le couvre.
4. Sélectionnez les versions de Delphi et de C++Builder dans lesquelles installer, puis lancez l'installation.
5. Ouvrez l'IDE. Les composants se trouvent dans la page **SGC Maps** de la Palette d'outils.

Les contrôles de carte n'apparaissent que dans la palette de leur propre framework : TsgcVCLMap dans les projets VCL et TsgcFMXMap dans les projets FireMonkey. Les composants non visuels apparaissent dans les deux.

Configuration requise

- À partir de Delphi XE7, le package d'exécution requiert **FireDAC** avec son pilote **SQLite** (les packages FireDAC, FireDACCommon, FireDACCommonDriver et FireDACSqliteDriver). sgcMaps l'utilise pour lire les fichiers MBTiles et l'index du géocodeur hors ligne, votre édition de Delphi doit donc inclure FireDAC SQLite.
- Les services en ligne utilisent THTTPClient (System.Net.HttpClient) et System.JSON, disponibles à partir de Delphi XE8.
- TsgcMapLocation utilise System.Sensors, disponible à partir de Delphi XE6.

Packages

Le dossier **Packages** contient une paire de packages par version d'IDE : le package d'exécution **sgcMapsD*nn*.dpk** et le package de conception **dclsgcMapsD*nn*.dpk**, de Delphi 7 (D7) à Delphi 13 (D13). C++Builder utilise les projets correspondants **sgcMapsC*nn*.cbproj** et **dclsgcMapsC*nn*.cbproj**, de C++Builder 2007 à C++Builder 13. Pour installer à la main, compilez d'abord le package d'exécution, puis installez le package de conception, et ajoutez le dossier Source au chemin de bibliothèque de l'IDE.

Lazarus

Lazarus possède son propre package, **Packages\Lazarus\sgcMapsLazarus.lpk**. Ouvrez-le avec Package, Open Package File, compilez-le et choisissez Use, Install, ce qui reconstruit l'IDE. Il requiert les packages LCL, FCL et IDEIntf et contient le contrôle de carte (TsgcVCLMap sur la LCL), la source de tuiles depuis un dossier, les fournisseurs, le moteur raster et les fonctions de géométrie. Le contrôle LCL ne peut pas télécharger de tuiles, il n'affiche donc que les tuiles d'un dossier. GeoJSON, GPX et KML figurent dans le package mais ne sont pas compilés sous FPC, et les services en ligne, le calculateur d'itinéraires, le calculateur hors ligne, le géocodeur hors ligne et TsgcMapLocation ne sont disponibles que sous Delphi. La rubrique Lazarus donne les détails et un exemple.

Unités

Ajoutez à la clause uses l'unité de chaque composant que vous créez dans le code. Les composants déposés sur un formulaire ajoutent leurs unités automatiquement.

Unité	Contenu
sgcMaps_Classes	Le modèle de carte : couches, marqueurs, polylignes, polygones, rectangles, cercles, étiquettes, cartes de chaleur et options, ainsi que TsgcMapCoord.

sgcMaps_VCL	TsgcVCLMap.
sgcMaps_FMX	TsgcFMXMap.
sgcMaps_Providers	Les fournisseurs de tuiles intégrés et sgcMapApplyProvider.
sgcMaps_Services	TsgcMapService, la classe de base des services en ligne.
sgcMaps_Geocoding	TsgcMapGeocoder et les classes de résultat de géocodage.
sgcMaps_Directions	TsgcMapDirections, les classes de résultat d'itinéraire et sgcMapAddRouteTo-Polylines.
sgcMaps_Places	TsgcMapPlaces.
sgcMaps_Elevation	TsgcMapElevation.
sgcMaps_TimeZone	TsgcMapTimeZone.
sgcMaps_RouteCalculator	TsgcMapRouteCalculator.
sgcMaps_OfflineRouter	TsgcMapOfflineRouter et TsgcMapRouteGraphBuilder.
sgcMaps_OfflineGeocoder	TsgcMapOfflineGeocoder et TsgcMapGeoIndexBuilder.
sgcMaps_Location	TsgcMapLocation.

.NET

L'installateur sgcUI n'installe rien pour .NET. sgcMaps for .NET est distribué sous forme de paquets NuGet distincts (esegece.sgcMaps et ses paquets compagnons), pour l'édition Registered uniquement. Voir Installation .NET.

Démarrage rapide VCL

Ce pas à pas montre une carte sur un formulaire VCL, d'abord avec le concepteur de formulaires, puis entièrement dans le code.

1. À la conception

1. Créez une application VCL et déposez sur le formulaire un composant **TsgcVCLMap** de la page SGC Maps. Réglez sa propriété **Align** sur alClient.
2. Ajoutez la couche de base dans l'événement **OnCreate** du formulaire avec **Layers.Add** et renseignez sa propriété **Provider**. Une couche avec un fournisseur télécharge les tuiles de ce fournisseur sans autre réglage. L'appel facultatif à **sgcMapApplyProvider** copie aussi dans la couche le type de source, le modèle d'URL, le zoom maximal et l'attribution du fournisseur.
3. Définissez la vue initiale avec **SetCenter** et **Zoom**, et ajoutez des surcouches avec **AddMarker**, **AddPolyline** ou **AddCircle**.
4. Lancez l'application. Faites glisser la carte pour la déplacer, et utilisez la molette de la souris ou les boutons de zoom pour zoomer.

Le gestionnaire OnCreate complet correspondant à ces étapes est l'exemple de la rubrique TsgcVCLMap.

2. Dans le code

Le programme ci-dessous construit la même carte sans fichier .dfm. Le formulaire crée son propre contrôle, c'est pourquoi le constructeur appelle **CreateNew** au lieu du **Create** hérité. Un clic sur le marqueur ouvre sa fenêtre contextuelle, et un clic ailleurs ajoute un nouveau marqueur à cette position.

```
program QuickStartVCL;

uses
  Forms, Classes, SysUtils, Controls,
  sgcMaps_Classes, sgcMaps_Providers, sgcMaps_VCL;

type
  TfrmMain = class(TForm)
  private
    FMap: TsgcVCLMap;
    procedure DoMarkerClick(Sender: TObject; aMarker: TsgcMapMarker);
    procedure DoMapClick(Sender: TObject; aLat, aLon: Double);
  public
    constructor Create(aOwner: TComponent); override;
  end;

constructor TfrmMain.Create(aOwner: TComponent);
var
  oLayer: TsgcMapLayer;
begin
  inherited CreateNew(aOwner);
  Caption := 'sgcMaps Quick Start';
  Width := 900;
  Height := 600;

  // 1. the map control fills the form
  FMap := TsgcVCLMap.Create(Self);
  FMap.Parent := Self;
  FMap.Align := alClient;

  // 2. a base layer: OpenStreetMap raster tiles, cached on disk
  oLayer := FMap.Layers.Add;
  oLayer.Name := 'OpenStreetMap';
  oLayer.Provider := mtpOpenStreetMap;
  sgcMapApplyProvider(oLayer);
```

```
  oLayer.CacheFolder := ExtractFilePath(ParamStr(0)) + 'tile_cache';

  // 3. the initial view and a marker
  FMap.SetCenter(48.8584, 2.2945);
  FMap.Zoom := 14;
  FMap.AddMarker(48.8584, 2.2945, 'Eiffel Tower', 'Champ de Mars, Paris');

  // 4. events
  FMap.OnMarkerClick := DoMarkerClick;
  FMap.OnMapClick := DoMapClick;
end;

procedure TfrmMain.DoMarkerClick(Sender: TObject; aMarker: TsgcMapMarker);
begin
  FMap.ShowPopup(aMarker);
end;

procedure TfrmMain.DoMapClick(Sender: TObject; aLat, aLon: Double);
begin
  // every click on the map adds a marker at that position
  FMap.AddMarker(aLat, aLon, Format('%.5f, %.5f', [aLat, aLon]));
end;

var
  frmMain: TfrmMain;

begin
  Application.Initialize;
  Application.CreateForm(TfrmMain, frmMain);
  Application.Run;
end.
```

À propos des tuiles

La couche met en cache dans **CacheFolder** chaque tuile qu'elle télécharge, si bien qu'une vue déjà vue s'affiche aussitôt et sans connexion. Les serveurs de tuiles d'OpenStreetMap sont gérés par des bénévoles et ont une politique d'utilisation qui n'autorise pas un usage intensif. Pour une application en production, utilisez un fournisseur commercial tel que MapTiler, Thunderforest ou Stadia (renseignez l'**ApiKey** de la couche), votre propre serveur de tuiles, ou des tuiles hors ligne depuis un dossier ou un fichier MBTiles. Laissez **Options.ShowAttribution** activé, car les fournisseurs de tuiles exigent que l'attribution soit visible.

Étapes suivantes

- TsgcVCLMap : chaque propriété, méthode et événement du contrôle.
- TsgcMapGeocoder : rechercher une adresse et l'afficher sur la carte.
- TsgcMapRouteCalculator : laisser l'utilisateur construire un itinéraire en cliquant sur la carte.

Démarrage rapide FMX

TsgcFMXMap est le contrôle de carte FireMonkey. Il partage le modèle de carte avec le contrôle VCL, de sorte que les couches, les marqueurs et les formes se créent exactement comme dans le Démarrage rapide VCL. Il requiert Delphi XE6 ou une version ultérieure et fonctionne sur les plateformes de bureau et mobiles de FireMonkey.

1. À la conception

1. Créez une application multi-périphériques et déposez sur le formulaire un composant **TsgcFMXMap** de la page SGC Maps. Réglez sa propriété **Align** sur Client.
2. Ajoutez la couche de base dans l'événement **OnCreate** du formulaire avec **Layers.Add** et renseignez sa propriété **Provider**, ce qui suffit pour télécharger les tuiles de ce fournisseur. L'appel facultatif à **sgcMapApplyProvider** copie aussi dans la couche le type de source, le modèle d'URL, le zoom maximal et l'attribution du fournisseur. Sur un appareil mobile, le dossier de l'application est en lecture seule, placez donc le cache de tuiles dans un dossier accessible en écriture, tel que **TPath.GetCachePath**.
3. Définissez la vue initiale avec **SetCenter** et **Zoom**, et ajoutez des marqueurs.
4. Lancez l'application. Déplacez la carte avec un doigt, zoomez en pinçant, et faites-la pivoter avec deux doigts lorsque **Options.RotationEnabled** vaut True.

Le gestionnaire OnCreate complet correspondant à ces étapes est l'exemple de la rubrique TsgcFMXMap.

2. Dans le code

Le programme ci-dessous construit la carte dans le code. Il active le regroupement de marqueurs, si bien que les marqueurs proches fusionnent en un seul cercle numéroté lorsque la carte est dézoomée, et il affiche la fenêtre contextuelle d'un marqueur lorsqu'on appuie dessus.

```
program QuickStartFMX;

uses
  System.StartUpCopy, System.SysUtils, System.Classes, System.IOUtils,
  FMX.Forms, FMX.Types, FMX.Controls,
  sgcMaps_Classes, sgcMaps_Providers, sgcMaps_FMX;

type
  TfrmMain = class(TForm)
  private
    FMap: TsgcFMXMap;
    procedure DoMarkerClick(Sender: TObject; aMarker: TsgcMapMarker);
  public
    constructor Create(aOwner: TComponent); override;
  end;

constructor TfrmMain.Create(aOwner: TComponent);
var
  oLayer: TsgcMapLayer;
begin
  inherited CreateNew(aOwner);
  Caption := 'sgcMaps FMX Quick Start';
  Width := 900;
  Height := 600;

  // 1. the map control fills the form
  FMap := TsgcFMXMap.Create(Self);
  FMap.Parent := Self;
  FMap.Align := TAlignLayout.Client;

  // 2. a base layer; the cache folder is valid on desktop and mobile
  oLayer := FMap.Layers.Add;
  oLayer.Name := 'OpenStreetMap';
  oLayer.Provider := mtpOpenStreetMap;
```

```
  sgcMapApplyProvider(oLayer);
  oLayer.CacheFolder := TPath.Combine(TPath.GetCachePath, 'tile_cache');

  // 3. the initial view, touch options and a few markers
  FMap.SetCenter(41.3874, 2.1686);
  FMap.Zoom := 13;
  FMap.Options.RotationEnabled := True;
  FMap.MarkerOptions.Clustering.Enabled := True;
  FMap.AddMarker(41.4036, 2.1744, 'Sagrada Familia');
  FMap.AddMarker(41.4145, 2.1527, 'Park Guell');
  FMap.AddMarker(41.3809, 2.1228, 'Camp Nou');

  // 4. events
  FMap.OnMarkerClick := DoMarkerClick;
end;

procedure TfrmMain.DoMarkerClick(Sender: TObject; aMarker: TsgcMapMarker);
begin
  FMap.ShowPopup(aMarker);
end;

var
  frmMain: TfrmMain;

begin
  Application.Initialize;
  Application.CreateForm(TfrmMain, frmMain);
  Application.Run;
end.
```

Étapes suivantes

- TsgcFMXMap : ce qui diffère du contrôle VCL.
- TsgcMapLocation : afficher et suivre la position de l'appareil.
- TsgcMapPlaces : suggérer des lieux pendant la saisie de l'utilisateur.

Démos

Le dossier **Demos\Maps** de l'installation sgcUI contient sept projets Delphi. Chacun est une petite application complète qui montre un groupe de fonctionnalités, et ensemble ils utilisent tous les composants de sgcMaps.

Démo	Framework	Version minimale	Ce qu'elle montre
01 Carte hors ligne	VCL	Delphi 7	Tuiles depuis un dossier, MBTiles et tuiles en ligne, marqueurs, formes, import GeoJSON, GPX et KML.
02 Fonctions de la carte	VCL	Delphi 7	Fournisseurs et clés d'API, regroupement, modification, cartes de chaleur, rotation et export.
03 Services	VCL	Delphi XE8	Recherche de lieux, géocodage, itinéraires, altitude, fuseaux horaires et localisation.
04 Services hors ligne	VCL	Delphi XE7	Tuiles hors ligne, géocodeur hors ligne et calculateur d'itinéraires hors ligne.
05 Rendu statique	Console	Delphi 7	Une carte rendue dans un fichier PNG ou BMP sans fenêtre.
06 FireMonkey	FireMonkey	Delphi XE6	Le contrôle FireMonkey sur ordinateur et sur mobile, avec localisation, recherche et calcul d'itinéraires.
07 Tuiles vectorielles	VCL	Delphi XE2	Tuiles vectorielles depuis un serveur ou un fichier MBTiles local, avec styles, retina et rotation.

Les démos ne contiennent aucune clé d'API dans leur code source. Celles qui utilisent un fournisseur exigeant une clé ont un champ de saisie pour celle-ci, et chaque démo démarre avec des services et des tuiles qui n'exigent aucune clé.
La plupart des démos téléchargent des tuiles depuis les serveurs publics d'OpenStreetMap, ce qui convient pour essayer les démos mais pas pour une application en production. Voir Dépannage pour les politiques d'utilisation.

01 Carte hors ligne

Dossier	Projet	Frame-work	Version minimale
Demos\Maps\01.OfflineMap	sgcMapsDemo.dpr	VCL	Delphi 7 (tuiles en ligne à partir de Delphi 2009, GeoJSON à partir de Delphi XE6, MBTiles à partir de Delphi XE7)

Ce que la démo montre

Un TsgcVCLMap dont la couche de base peut provenir de trois sources, choisies avec les options **Folder**, **MBTiles** et **Online** : un dossier z/x/y de tuiles, un fichier MBTiles, ou le serveur de tuiles d'OpenStreetMap avec un cache disque. La carte démarre à Madrid au zoom 13.

- **Go To Coordinates** et une zone de zoom déplacent la vue.
- **Marker at Center**, **Sample Markers** et **Clear Markers** ajoutent et suppriment des marqueurs.
- **Sample Circle**, **Sample Polyline** et **Sample Polygon** dessinent des formes.
- Les cases à cocher **Zoom Controls** et **Scale Bar** activent et désactivent ces éléments de la carte.
- **GeoJSON...**, **GPX...** et **KML...** importent un fichier dans la carte.
- Le journal affiche les événements OnMapClick, OnMarkerClick et OnZoomChanged.

Composants et unités

TsgcVCLMap avec des couches de dossier, MBTiles et HTTP, et les fonctions sgcMapLoadGeoJSONFile, sgcMapLoadGPXFile et sgcMapLoadKMLFile. Unités : sgcMaps_Classes, sgcMaps_TileSource, sgcMaps_VCL, sgcMaps_GeoJSON, sgcMaps_GPX et sgcMaps_KML.

Comment l'exécuter

Ouvrez **sgcMapsDemo.dpr** dans l'IDE, puis compilez-le et exécutez-le. La démo trouve les unités sgcMaps par le chemin de bibliothèque que définit l'installateur sgcUI ; avec une installation manuelle, ajoutez d'abord le dossier Source au chemin de bibliothèque.

Données et clés

- Aucune clé d'API n'est nécessaire.
- Le dossier **data** à côté du projet contient sample.geojson, sample.gpx et sample.kml pour les boutons d'import. La démo le cherche d'abord à côté de l'exécutable, puis dans le dossier courant.
- Un chemin de tuiles relatif est résolu à partir du dossier de l'exécutable. Le chemin par défaut est **tiles**, un dossier de fichiers z\x\y.png, que vous pouvez remplir avec le téléchargeur de tuiles (voir Tuiles hors ligne et téléchargeur de tuiles).
- Les tuiles en ligne sont mises en cache dans le dossier **tile_cache** à côté de l'exécutable.
- Lorsque le dossier de tuiles ou le fichier MBTiles n'existe pas, la démo écrit une ligne dans le journal.
- Avec les versions de Delphi antérieures à XE6, le bouton GeoJSON est désactivé, et avec les versions antérieures à XE7, l'option MBTiles se contente d'indiquer dans le journal qu'elle requiert FireDAC.

02 Fonctions de la carte

Dossier	Projet	Frame-work	Version minimale
Demos\Maps\02.MapFeatures	sgcMapsFeaturesDemo.dpr	VCL	Delphi 7 (tuiles en ligne et PNG à partir de Delphi 2009, GeoJSON à partir de Delphi XE6)

Ce que la démo montre

Les fonctions de dessin et d'interaction de TsgcVCLMap, un onglet chacune :

Onglet	Ce que vous pouvez faire
Layers	Choisir une couche de base, saisir une clé d'API pour MapTiler, Thunderforest ou Stadia et appuyer sur **Apply key** pour ajouter leurs couches, et afficher le sélecteur de couches sur la carte.
Clusters	Ajouter 2000 marqueurs aléatoires autour de Madrid et activer ou désactiver le regroupement.
Heatmap	Ajouter une carte de chaleur de 5000 points pondérés et modifier son rayon, son dégradé et son opacité.
Drawing	Ajouter un marqueur déplaçable, une polyligne modifiable, un polygone modifiable avec un trou, un rectangle modifiable et une ligne géodésique de Madrid à New York.
Rotation	Activer la rotation, définir l'orientation, faire pivoter avec Ctrl+glisser ou un glisser avec le bouton droit, et remettre la carte nord en haut.
Export	Exporter les surcouches en GeoJSON, GPX ou KML, et enregistrer la carte en PNG (BMP avant Delphi 2009).

Le journal affiche OnMarkerDragEnd, OnShapeEdited, OnLayerChanged, OnClusterClick et OnBearingChanged.

Composants et unités

TsgcVCLMap, la liste des fournisseurs de sgcMaps_Providers (sgcMapGetProviderInfo), TsgcMapGPX, sgcMapSaveGeoJSONFile, sgcMapSaveKMLFile et sgcMapDistance. Unités : sgcMaps_Classes, sgcMaps_VCL, sgcMaps_Providers, sgcMaps_Geometry, sgcMaps_GPX, sgcMaps_KML et sgcMaps_GeoJSON.

Comment l'exécuter

Ouvrez **sgcMapsFeaturesDemo.dpr** dans l'IDE, puis compilez-le et exécutez-le. La démo trouve les unités sgcMaps par le chemin de bibliothèque que définit l'installateur sgcUI ; avec une installation manuelle, ajoutez d'abord le dossier Source au chemin de bibliothèque.

Données et clés

- Au démarrage, la démo ajoute une couche de base pour chaque fournisseur qui n'exige aucune clé : OpenStreetMap, OpenTopoMap, les couches Esri et CARTO Light et Dark.

- MapTiler, Thunderforest et Stadia exigent une clé d'API obtenue sur leur site web. Saisissez-la dans l'onglet Layers et appuyez sur Apply key.
- Chaque fournisseur a son propre cache de tuiles dans **%TEMP%\sgcMapsFeaturesDemo**.
- Les fichiers exportés sont écrits à l'endroit que vous choisissez dans la boîte de dialogue d'enregistrement.

03 Services

Dossier	Projet	Framework	Version minimale
Demos\Maps\03.Services	sgcMapsServicesDemo.dpr	VCL	Delphi XE8

Ce que la démo montre

Tous les services en ligne sur un seul formulaire. Chaque onglet a une liste de fournisseurs et un champ de clé d'API, et le résultat est dessiné sur la carte.

Onglet	Ce que vous pouvez faire
Search	Suggestions de lieux pendant la saisie (Photon, Nominatim, Google, HERE, Mapbox ou Geoapify), et géocodage d'une adresse.
Reverse	Cliquer sur la carte pour obtenir l'adresse de ce point.
Directions	Un itinéraire entre deux adresses ou entre des points de passage cliqués sur la carte (déplaçables), avec un mode de déplacement, des alternatives, et OSRM, OpenRouteService, Google Routes, HERE, Azure Maps, Mapbox ou Geoapify.
Elevation	Le profil altimétrique de l'itinéraire, avec Open-Elevation, OpenTopoData ou Google.
Time zone	Le fuseau horaire du centre de la carte, avec TimeAPI, GeoNames, Google ou Azure Maps.
Location	Simuler un trajet GPS le long de l'itinéraire, laisser la carte suivre la position, ou utiliser la localisation de l'appareil.

Composants et unités

TsgcMapPlaces, TsgcMapGeocoder, TsgcMapDirections, TsgcMapRouteCalculator, TsgcMapElevation, TsgcMapTimeZone et TsgcMapLocation, créés dans le code sur un formulaire avec un TsgcVCLMap.

Comment l'exécuter

Ouvrez **sgcMapsServicesDemo.dpr** dans l'IDE, puis compilez-le et exécutez-le. La démo trouve les unités sgcMaps par le chemin de bibliothèque que définit l'installateur sgcUI ; avec une installation manuelle, ajoutez d'abord le dossier Source au chemin de bibliothèque.

Données et clés

- Les fournisseurs par défaut n'exigent aucune clé : Photon, Nominatim, le serveur de démonstration d'OSRM, OpenTopoData et TimeAPI.
- Google, HERE, Azure Maps, Mapbox, OpenRouteService et Geoapify exigent une clé obtenue dans leurs consoles de développement. Pour GeoNames, saisissez votre nom d'utilisateur GeoNames comme clé.
- La démo définit son propre UserAgent sur chaque service, comme le demandent les serveurs publics, et affiche leurs limites sur le formulaire : une requête par seconde et aucune requête en masse. Utilisez votre propre serveur ou un fournisseur commercial en production.

- Les tuiles proviennent d'OpenStreetMap et sont mises en cache dans le dossier **tilecache** à côté de l'exécutable.
- Les erreurs des services sont écrites dans le journal. Lorsque la localisation de l'appareil n'est pas disponible, la démo l'indique et décoche l'option.

04 Services hors ligne

Dossier	Projet	Framework	Version minimale
Demos\Maps\04.OfflineServices	sgcMapsOfflineDemo.dpr	VCL	Delphi XE7 (FireDAC SQ-Lite)

Ce que la démo montre

Recherche et calcul d'itinéraires sans connexion réseau. La démo construit ses fichiers de données à partir de téléchargements gratuits, les rouvre au démarrage suivant et répond localement à chaque requête. La carte démarre en Andorre.

- **Offline tiles** avec **Folder...** ou **MBTiles...** définit une carte de base locale. Sans cela, ou lorsque le fichier est absent, la démo utilise les tuiles en ligne d'OpenStreetMap et l'indique dans le journal.
- **Build from GeoNames...** construit un index de géocodeur à partir d'un export GeoNames, et **Open index...** en ouvre un existant.
- **Build from .osm.pbf...** construit un graphe routier à partir d'un extrait OpenStreetMap, et **Open graph...** en ouvre un existant.
- **Search** trouve un lieu dans l'index, et un clic sur la carte effectue le géocodage inverse de ce point.
- Les clics sur la carte ajoutent des points de passage à l'itinéraire, qui peuvent être déplacés. L'itinéraire est calculé pour le profil Car, Bike ou Foot et ses étapes sont listées.

Composants et unités

TsgcMapOfflineGeocoder, TsgcMapGeoIndexBuilder, TsgcMapOfflineRouter, TsgcMapRouteGraphBuilder et TsgcMapRouteCalculator, qui appelle le calculateur hors ligne depuis son événement OnCalculateRoute.

Comment l'exécuter

Ouvrez **sgcMapsOfflineDemo.dpr** dans l'IDE, puis compilez-le et exécutez-le. La démo trouve les unités sgcMaps par le chemin de bibliothèque que définit l'installateur sgcUI ; avec une installation manuelle, ajoutez d'abord le dossier Source au chemin de bibliothèque.

Données et clés

- Aucune clé d'API n'est nécessaire.
- Un export GeoNames depuis https://download.geonames.org/export/dump/, tel que ES.zip ou cities15000.zip. Lorsque admin1CodesASCII.txt et countryInfo.txt se trouvent dans le même dossier, la démo les utilise aussi. L'index est écrit à côté de l'export avec l'extension .sqlite.
- Un extrait OpenStreetMap, tel que andorra-latest.osm.pbf depuis un service de téléchargement comme Geofabrik. Le graphe est écrit à côté avec l'extension .sgcgraph.
- Les constructions s'exécutent dans un thread avec une barre de progression, et le formulaire ne peut pas être fermé avant leur fin.
- Les chemins de l'index, du graphe et des tuiles sont enregistrés dans un fichier .ini à côté de l'exécutable et rouverts au démarrage.
- Les outils construisent les mêmes fichiers en ligne de commande.

05 Rendu statique

Dossier	Projet	Frame-work	Version minimale
Demos\Maps\05.StaticRender	sgcMapsStaticRender.dpr	Console	Delphi 7 (tuiles en ligne et PNG à partir de Delphi 2009, itinéraires à partir de Delphi XE2, GeoJSON à partir de Delphi XE6)

Ce que la démo montre

Une application console qui crée un TsgcVCLMap sans fenêtre parente, charge les tuiles d'une vue et l'enregistre sous forme d'image. C'est la base d'un service de cartes statiques ou d'images de cartes dans des rapports. Voir Rendu de cartes statiques.

Comment l'exécuter

Compilez le projet et exécutez-le depuis une invite de commandes :

```
sgcMapsStaticRender --center lat,lon --zoom z --size WxH
  --provider osm|opentopo|esri|carto --out file.png|file.bmp
  [--marker lat,lon]... [--geojson file]
  [--route lat,lon;lat,lon[;...] --graph file.sgcgraph]
```

Paramètre	Signification
--center lat,lon	Le centre de la vue. Obligatoire, sauf si --route est indiqué, auquel cas la vue est centrée sur l'itinéraire.
--zoom z	Le niveau de zoom. La valeur par défaut est 13.
--size WxH	La taille de l'image en pixels. La valeur par défaut est 800x600.
--provider	osm (par défaut), opentopo, esri ou carto. Ces fournisseurs n'exigent aucune clé.
--out file	L'image à écrire. Obligatoire. Une extension .bmp écrit un fichier BMP, toute autre extension un fichier PNG (un fichier BMP avant Delphi 2009).
--marker lat,lon	Ajoute un marqueur. Peut être répété.
--geojson file	Dessine les entités d'un fichier GeoJSON (Delphi XE6 et versions ultérieures).
--route and --graph	Calcule un itinéraire passant par les points de passage avec le calculateur hors ligne et le fichier .sgcgraph indiqué, et le dessine (Delphi XE2 et versions ultérieures).

Les coordonnées utilisent le point comme séparateur décimal. Le programme affiche le fournisseur, la vue, le fichier enregistré et l'attribution, et se termine avec le code 0 en cas de succès, 1 pour un paramètre erroné et 2 lorsqu'une exception se produit.

```
sgcMapsStaticRender --center 48.8584,2.2945 --zoom 15 --size 1024x768 --provider osm --out eiffel.png
```

Données et clés

- Aucune clé d'API n'est nécessaire.
- Les tuiles sont téléchargées lorsqu'elles ne se trouvent pas dans le dossier **tile_cache** à côté de l'exécutable, la première exécution requiert donc une connexion réseau.
- L'outil RouteGraphBuilder construit le fichier .sgcgraph pour --route.

06 FireMonkey

Dossier	Projet	Frame-work	Version minimale
Demos\Maps\06.FMX	sgcMapsFMXDemo.dpr	FireMon-key	Delphi XE6 (recherche et calcul d'itinéraires à partir de Delphi XE8)

Ce que la démo montre

Un TsgcFMXMap créé dans le code, avec quatre onglets :

Onglet	Ce que vous pouvez faire
Search	Suggestions de lieux pendant la saisie, et **Route from here** vers le lieu sélectionné.
Loca-tion	**Locate me** et **Follow my location**.
Over-lays	Regroupement de marqueurs avec 500 marqueurs, une carte de chaleur, un marqueur dépla-çable, un polygone modifiable, et **Clear overlays**.
View	La couche de base, le sélecteur de couches, la rotation à deux doigts, l'orientation, **Reset com-pass**, et **Snapshot to PNG**, qui enregistre sgcMapsFMX.png dans le dossier des documents.

Composants et unités

TsgcFMXMap, TsgcMapLocation, TsgcMapPlaces avec Photon et TsgcMapDirections avec OSRM. La carte est créée dans le code, la démo s'exécute donc sans le package de conception.

Comment l'exécuter

Ouvrez **sgcMapsFMXDemo.dpr** dans l'IDE, choisissez la plateforme cible, puis compilez-le et exécutez-le. Sous Android, l'autorisation de localisation est demandée à l'exécution (Delphi 10.3 et versions ultérieures).

Données et clés

- Aucune clé d'API n'est nécessaire. Les couches de base sont les fournisseurs qui n'exigent aucune clé, et Places et Directions définissent leur propre UserAgent.
- Le cache de tuiles se trouve dans le dossier temporaire de l'appareil, accessible en écriture sur toutes les plateformes.
- Lorsque l'appareil n'a pas de capteur de localisation, qu'aucune position n'arrive dans les cinq secondes, ou que l'autorisation est refusée, la démo utilise une position simulée à Madrid pour que le reste de la démo fonctionne quand même.

07 Tuiles vectorielles

Dossier	Projet	Frame-work	Version minimale
Demos\Maps\07.VectorTiles	sgcMapsVectorDemo.dpr	VCL	Delphi XE2 (MBTiles local à partir de Delphi XE7)

Ce que la démo montre

Des tuiles vectorielles dessinées par TsgcVCLMap à partir de trois sources et avec trois styles. Voir Tuiles vectorielles.

- Sources : **OpenFreeMap** (en ligne, sans clé), **MapTiler vector** (exige une clé) et un fichier **local .mbtiles** de tuiles vectorielles.
- Styles : le style intégré, le style d'exemple **Liberty**, ou n'importe quel fichier JSON de style MapLibre.
- **Retina** dessine les tuiles à 512 pixels, et le curseur d'orientation ou Ctrl+glisser fait pivoter la carte.
- La barre d'état affiche le zoom, le nombre d'étiquettes dessinées et les erreurs de tuiles signalées par OnTileError.

Comment l'exécuter

Ouvrez **sgcMapsVectorDemo.dpr** dans l'IDE, puis compilez-le et exécutez-le. La démo trouve les unités sgcMaps par le chemin de bibliothèque que définit l'installateur sgcUI ; avec une installation manuelle, ajoutez d'abord le dossier Source au chemin de bibliothèque.

Données et clés

- OpenFreeMap n'exige aucune clé. Au démarrage, la démo lit l'URL actuelle des tuiles dans le TileJSON d'OpenFreeMap (Delphi XE8 et versions ultérieures) et se rabat sur une URL fixe lorsqu'elle n'y parvient pas.
- MapTiler exige une clé obtenue sur le site web de MapTiler Cloud.
- Un fichier local doit contenir des tuiles vectorielles au schéma OpenMapTiles, celui que dessine le style intégré.
- Le style d'exemple Liberty est lu depuis les fixtures QA de l'installation. Lorsque le fichier est introuvable, cette option est désactivée.

TsgcVCLMap

Unité : sgcMaps_VCL
Le contrôle de carte VCL. Il descend de TCustomControl et dessine lui-même les tuiles et les surcouches, il n'a donc besoin ni de navigateur, ni de WebView, ni de JavaScript. Sous Lazarus, la même classe est le contrôle de carte LCL, qui n'affiche que les tuiles d'un dossier car il ne peut pas télécharger de tuiles (voir Lazarus).
Le contrôle affiche les collections du modèle de carte. Ajoutez une couche de tuiles à **Layers**, définissez la vue avec **SetCenter** et **Zoom**, et ajoutez des marqueurs et des formes avec les méthodes Add ou directement dans leurs collections. L'utilisateur déplace la carte à la souris, zoome avec la molette ou les boutons de zoom, et fait pivoter la carte avec Ctrl+glisser ou un glisser avec le bouton droit lorsque **Options.RotationEnabled** vaut True.

Propriétés

Propriété	Type	Par défaut	Description
CenterLatitude	Double		Latitude du centre de la vue, en degrés décimaux (WGS 84).
CenterLongitude	Double		Longitude du centre de la vue, en degrés décimaux (WGS 84).
Zoom	Integer	13	Niveau de zoom, limité par Options.MinZoom et Options.MaxZoom.
Layers	TsgcMapLayers		Couches de tuiles, dessinées dans l'ordre de la collection. Voir les propriétés de couche ci-dessous.
Markers	TsgcMapMarkers		Marqueurs : Latitude, Longitude, PopupText, TooltipText, Color, IconBitmap, AnchorX, AnchorY, Draggable, Visible, Tag et Data.
Polylines	TsgcMapPolylines		Lignes : AddPoint, Color, Width, Opacity, Dashed, Geodesic, Editable, Visible et Tag.
Polygons	TsgcMapPolygons		Polygones pleins, avec trous.
Rectangles	TsgcMapRectangles		Rectangles pleins définis par leurs bords nord, ouest, sud et est.
Circles	TsgcMapCircles		Cercles : CenterLat, CenterLon, RadiusMeters, FillColor, StrokeColor, StrokeWidth et FillOpacity.
Labels	TsgcMapLabels		Étiquettes de texte ancrées à une coordonnée.
Heatmaps	TsgcMapHeatmaps		Cartes de chaleur de points pondérés, avec les dégradés mhgDefault, mhgFire, mhgCool et mhgGrayscale.
Options	TsgcMapOptions		Options de la vue : MinZoom, MaxZoom, TileSize, ShowZoomControls, ShowScaleBar, ShowAttribution, AttributionText, BackgroundColor, DragEnabled, ZoomEnabled, Bearing, RotationEnabled et ShowLayerSwitcher.
MarkerOptions	TsgcMapMarkerOptions		Taille, couleur et police par défaut des marqueurs, et Clustering (Enabled, GridSize, MinClusterSize, Color, FontColor, ZoomOnClick, MaxZoom).

ActivePopupMarker	TsgcMapMarker		Le marqueur dont la fenêtre contextuelle est ouverte, ou nil. Lecture seule.
DrawnMarkerCount	Integer		Marqueurs dessinés lors du dernier affichage. Lecture seule.
DrawnClusterCount	Integer		Regroupements dessinés lors du dernier affichage. Lecture seule.
DrawnLabelCount	Integer		Étiquettes dessinées lors du dernier affichage. Lecture seule.
LastRenderMissingTiles	Integer		Tuiles qui n'étaient pas encore disponibles lors du dernier affichage. Lecture seule.

Propriétés de couche

Chaque élément de **Layers** est un TsgcMapLayer.

Propriété	Type	Par défaut	Description
Name	string		Nom affiché par le sélecteur de couches.
TileSourceType	TsgcMapTileSourceType	mtsFolder	mtsFolder (dossier z/x/y), mtsMBTiles, mtsHTTP, mtsVectorMBTiles ou mtsVectorHTTP.
TileSourcePath	string		Dossier ou fichier MBTiles d'une couche hors ligne.
URLTemplate	string		URL des tuiles d'une couche en ligne, avec les paramètres {z}, {x} et {y}.
CacheFolder	string		Cache disque d'une couche en ligne. Vide, il désactive le cache.
Provider	TsgcMapTileProvider	mtpCustom	Un fournisseur intégré. La couche télécharge ses tuiles sans autre réglage ; sgcMapApplyProvider copie aussi dans la couche le type de source, le modèle d'URL, le zoom maximal et l'attribution du fournisseur.
ApiKey	string		Clé des fournisseurs qui en exigent une (MapTiler, Thunderforest, Stadia).
Attribution	string		Texte d'attribution de la couche.
Opacity	Double		Opacité de 0 à 1.
Visible	Boolean	True	Affiche ou masque la couche.
MinZoom, MaxZoom	Integer	0, 19	Plage de zoom des tuiles de la couche.
IsBaseLayer	Boolean	True	True pour une couche de base, False pour une surcouche dessinée par-dessus. Afficher une couche de base avec SetLayerVisible

			ou dans le sélecteur de couches masque les autres couches de base.
Retina	Boolean	False	Demande des tuiles haute résolution.
Vector-StyleFile	string		Style MapLibre ou Mapbox GL d'une couche vectorielle.

Méthodes

Méthode	Description
SetCenter(aLat, aLon: Double)	Déplace le centre de la vue.
ZoomIn, ZoomOut	Modifie le zoom d'un niveau.
FitBounds(aNorthLat, aWestLon, aSouthLat, aEastLon: Double; aPaddingPx: Integer = 20)	Centre et zoome la vue pour afficher une zone.
FitToMarkers(aPaddingPx: Integer = 20)	Centre et zoome la vue pour afficher tous les marqueurs.
FitToPolyline(aPolyline: TsgcMapPolyline; aPaddingPx: Integer = 20)	Centre et zoome la vue pour afficher une polyligne, par exemple un itinéraire.
AddMarker(aLat, aLon: Double; const aPopup: string = ''; const aTooltip: string = '')	Ajoute un marqueur avec un texte de fenêtre contextuelle et une infobulle facultatifs.
ClearMarkers	Supprime tous les marqueurs.
AddPolyline: TsgcMapPolyline	Ajoute une polyligne vide ; ajoutez ses points avec AddPoint.
AddPolygon: TsgcMapPolygon	Ajoute un polygone vide.
AddCircle(aLat, aLon, aRadiusMeters: Double): TsgcMapCircle	Ajoute un cercle.
AddRectangle(aNorthLat, aWestLon, aSouthLat, aEastLon: Double): TsgcMapRectangle	Ajoute un rectangle.
AddLabel(aLat, aLon: Double; const aText: string): TsgcMapLabel	Ajoute une étiquette de texte.
ShowPopup(aMarker: TsgcMapMarker), HidePopup	Ouvre ou ferme la fenêtre contextuelle d'un marqueur.
MarkerAtScreen(aX, aY): TsgcMapMarker	Le marqueur situé en un point du contrôle, ou nil.
ClusterAtScreen(aX, aY; out aCount: Integer; out aLat, aLon: Double): Boolean	True lorsqu'un regroupement se trouve en un point du contrôle, avec sa taille et sa position.
SetLayerVisible(aIndex: Integer; aVisible: Boolean)	Affiche ou masque une couche. Afficher une couche de base masque les autres couches de base, ce que fait aussi le sélecteur de couches.
RefreshTiles	Recharge les tuiles de la vue.
AttributionText: string	L'attribution des couches visibles.

RenderToBitmap(aBitmap: TBitmap; aIncludeControls: Boolean = False)	Dessine la vue actuelle dans un bitmap.
SaveToPNG, SaveToBMP(const aFileName: string; aWidth, aHeight: Integer; aIncludeControls: Boolean = False)	Dessine la vue à la taille indiquée dans un fichier PNG ou BMP. Fonctionne sans fenêtre parente.

Événements

Événement	Description
OnMapClick	Un clic sur la carte, avec sa latitude et sa longitude.
OnMarkerClick	Un clic sur un marqueur.
OnMarkerDragStart, OnMarker-Drag, OnMarkerDragEnd	L'utilisateur déplace un marqueur dont la propriété Draggable vaut True.
OnShapeEdited	L'utilisateur a déplacé un sommet d'une polyligne, d'un polygone ou d'un rectangle modifiable.
OnClusterClick	Un clic sur un regroupement, avec sa taille et sa position. Définissez aHandled à True pour traiter le clic vous-même.
OnLayerChanged	L'utilisateur a choisi une autre couche dans le sélecteur de couches.
OnZoomChanged	Le niveau de zoom a changé.
OnBearingChanged	La carte a pivoté.
OnTileError	Une tuile n'a pas pu être chargée, avec la couche, les coordonnées de la tuile et l'erreur.

Exemple : configuration de la carte

Un gestionnaire OnCreate qui ajoute une couche OpenStreetMap, centre la vue et ajoute un marqueur, une ligne et un cercle.

```
procedure TForm1.FormCreate(Sender: TObject);
var
  oLayer: TsgcMapLayer;
  oLine: TsgcMapPolyline;
  oCircle: TsgcMapCircle;
begin
  // base layer: OpenStreetMap tiles, cached on disk
  oLayer := sgcVCLMap1.Layers.Add;
  oLayer.Name := 'OpenStreetMap';
  oLayer.Provider := mtpOpenStreetMap;
  sgcMapApplyProvider(oLayer);
  oLayer.CacheFolder := ExtractFilePath(ParamStr(0)) + 'tile_cache';

  // center on Madrid
  sgcVCLMap1.SetCenter(40.4168, -3.7038);
  sgcVCLMap1.Zoom := 12;

  // a marker with a popup text and a tooltip
  sgcVCLMap1.AddMarker(40.4153, -3.6844, 'Retiro Park', 'Click for details');

  // a polyline between two points
  oLine := sgcVCLMap1.AddPolyline;
```

```
  oLine.AddPoint(40.4168, -3.7038);
  oLine.AddPoint(40.4153, -3.6844);
  oLine.Width := 4;

  // a circle with a radius of 500 meters
  oCircle := sgcVCLMap1.AddCircle(40.4168, -3.7038, 500);
  oCircle.FillOpacity := 0.2;
end;
```

Exemple : événements et instantanés

```
procedure TForm1.sgcVCLMap1MarkerClick(Sender: TObject; aMarker: TsgcMapMarker);
begin
  sgcVCLMap1.ShowPopup(aMarker);
end;

procedure TForm1.sgcVCLMap1MapClick(Sender: TObject; aLat, aLon: Double);
begin
  memoLog.Lines.Add(Format('Clicked at %.5f, %.5f', [aLat, aLon]));
end;

procedure TForm1.btnSnapshotClick(Sender: TObject);
begin
  // renders the current view into an 800 x 600 PNG file
  sgcVCLMap1.SaveToPNG('map.png', 800, 600);
end;
```

Voir aussi

- TsgcFMXMap
- Démarrage rapide VCL
- TsgcMapRouteCalculator

TsgcFMXMap

Unité : sgcMaps_FMX
Le contrôle de carte FireMonkey, disponible à partir de Delphi XE6. Il descend du TControl de FireMonkey et partage le modèle de carte avec TsgcVCLMap : les mêmes Layers, Markers, Polylines, Polygons, Rectangles, Circles, Labels, Heatmaps, Options et MarkerOptions, les mêmes méthodes et les mêmes événements.

Différences avec TsgcVCLMap

- Saisie tactile : un doigt déplace la carte, un pincement zoome et, avec **Options.RotationEnabled**, deux doigts font pivoter la carte.
- **MarkerAtScreen** et **ClusterAtScreen** prennent des coordonnées Single, le type de coordonnée de FireMonkey.
- **RenderToBitmap** prend un TBitmap FMX, et le contrôle publie **HitTest** (True par défaut).
- Sur les plateformes mobiles, le dossier de l'application est en lecture seule : placez le **CacheFolder** de la couche dans un dossier accessible en écriture, tel que TPath.GetCachePath, et enregistrez les instantanés dans TPath.GetDocumentsPath.

Propriétés

Propriété	Type	Par défaut	Description
CenterLatitude	Double		Latitude du centre de la vue, en degrés décimaux (WGS 84).
CenterLongitude	Double		Longitude du centre de la vue, en degrés décimaux (WGS 84).
Zoom	Integer	13	Niveau de zoom, limité par Options.MinZoom et Options.MaxZoom.
Layers	TsgcMapLayers		Couches de tuiles, dessinées dans l'ordre de la collection. Voir les propriétés de couche ci-dessous.
Markers, Polylines, Polygons, Rectangles, Circles, Labels, Heatmaps	collections		Les surcouches, comme dans TsgcVCLMap.
Options, MarkerOptions	TsgcMapOptions, TsgcMapMarkerOptions		Options de la vue, des marqueurs et du regroupement, comme dans TsgcVCLMap.
Canvas	TCanvas		Le canevas FireMonkey du contrôle. Lecture seule.
ActivePopupMarker	TsgcMapMarker		Le marqueur dont la fenêtre contextuelle est ouverte, ou nil. Lecture seule.
DrawnMarkerCount	Integer		Marqueurs dessinés lors du dernier affichage. Lecture seule.
DrawnClusterCount	Integer		Regroupements dessinés lors du dernier affichage. Lecture seule.

DrawnLabelCount	Integer		Étiquettes dessinées lors du dernier affichage. Lecture seule.
LastRenderMissingTiles	Integer		Tuiles qui n'étaient pas encore disponibles lors du dernier affichage. Lecture seule.

Méthodes

Méthode	Description
SetCenter(aLat, aLon: Double)	Déplace le centre de la vue.
ZoomIn, ZoomOut	Modifie le zoom d'un niveau.
FitBounds(aNorthLat, aWestLon, aSouthLat, aEastLon: Double; aPaddingPx: Integer = 20)	Centre et zoome la vue pour afficher une zone.
FitToMarkers(aPaddingPx: Integer = 20)	Centre et zoome la vue pour afficher tous les marqueurs.
FitToPolyline(aPolyline: TsgcMapPolyline; aPaddingPx: Integer = 20)	Centre et zoome la vue pour afficher une polyligne, par exemple un itinéraire.
AddMarker(aLat, aLon: Double; const aPopup: string = ''; const aTooltip: string = '')	Ajoute un marqueur avec un texte de fenêtre contextuelle et une infobulle facultatifs.
ClearMarkers	Supprime tous les marqueurs.
AddPolyline: TsgcMapPolyline	Ajoute une polyligne vide ; ajoutez ses points avec AddPoint.
AddPolygon: TsgcMapPolygon	Ajoute un polygone vide.
AddCircle(aLat, aLon, aRadiusMeters: Double): TsgcMapCircle	Ajoute un cercle.
AddRectangle(aNorthLat, aWestLon, aSouthLat, aEastLon: Double): TsgcMapRectangle	Ajoute un rectangle.
AddLabel(aLat, aLon: Double; const aText: string): TsgcMapLabel	Ajoute une étiquette de texte.
ShowPopup(aMarker: TsgcMapMarker), HidePopup	Ouvre ou ferme la fenêtre contextuelle d'un marqueur.
MarkerAtScreen(aX, aY): TsgcMapMarker	Le marqueur situé en un point du contrôle, ou nil.
ClusterAtScreen(aX, aY; out aCount: Integer; out aLat, aLon: Double): Boolean	True lorsqu'un regroupement se trouve en un point du contrôle, avec sa taille et sa position.
SetLayerVisible(aIndex: Integer; aVisible: Boolean)	Affiche ou masque une couche. Afficher une couche de base masque les autres couches de base, ce que fait aussi le sélecteur de couches.
RefreshTiles	Recharge les tuiles de la vue.
AttributionText: string	L'attribution des couches visibles.

RenderToBitmap(aBitmap: TBitmap; aIncludeControls: Boolean = False)	Dessine la vue actuelle dans un bitmap.
SaveToPNG, SaveToBMP(const aFileName: string; aWidth, aHeight: Integer; aIncludeControls: Boolean = False)	Dessine la vue à la taille indiquée dans un fichier PNG ou BMP. Fonctionne sans fenêtre parente.

Événements

Événement	Description
OnMapClick	Un clic sur la carte, avec sa latitude et sa longitude.
OnMarkerClick	Un clic sur un marqueur.
OnMarkerDragStart, OnMarker-Drag, OnMarkerDragEnd	L'utilisateur déplace un marqueur dont la propriété Draggable vaut True.
OnShapeEdited	L'utilisateur a déplacé un sommet d'une polyligne, d'un polygone ou d'un rectangle modifiable.
OnClusterClick	Un clic sur un regroupement, avec sa taille et sa position. Définissez aHandled à True pour traiter le clic vous-même.
OnLayerChanged	L'utilisateur a choisi une autre couche dans le sélecteur de couches.
OnZoomChanged	Le niveau de zoom a changé.
OnBearingChanged	La carte a pivoté.
OnTileError	Une tuile n'a pas pu être chargée, avec la couche, les coordonnées de la tuile et l'erreur.

Exemple : configuration de la carte

Un gestionnaire OnCreate qui ajoute une couche OpenStreetMap dont le cache se trouve dans le dossier de cache de l'application, active la rotation et le regroupement, et enregistre un instantané dans le dossier des documents.

```
procedure TForm1.FormCreate(Sender: TObject);
var
  oLayer: TsgcMapLayer;
begin
  oLayer := sgcFMXMap1.Layers.Add;
  oLayer.Name := 'OpenStreetMap';
  oLayer.Provider := mtpOpenStreetMap;
  sgcMapApplyProvider(oLayer);
  // on mobile, keep the tile cache in the cache folder of the application
  oLayer.CacheFolder := TPath.Combine(TPath.GetCachePath, 'tile_cache');

  sgcFMXMap1.SetCenter(41.3874, 2.1686);
  sgcFMXMap1.Zoom := 13;
  // two finger rotation and marker clustering
  sgcFMXMap1.Options.RotationEnabled := True;
  sgcFMXMap1.MarkerOptions.Clustering.Enabled := True;
  sgcFMXMap1.AddMarker(41.4036, 2.1744, 'Sagrada Familia');
  sgcFMXMap1.OnMarkerClick := sgcFMXMap1MarkerClick;
end;

procedure TForm1.sgcFMXMap1MarkerClick(Sender: TObject; aMarker: TsgcMapMarker);
begin
```

```
  sgcFMXMap1.ShowPopup(aMarker);
end;

procedure TForm1.btnSnapshotClick(Sender: TObject);
begin
  sgcFMXMap1.SaveToPNG(TPath.Combine(TPath.GetDocumentsPath, 'map.png'),
    1024, 768);
end;
```

Voir aussi

- TsgcVCLMap
- Démarrage rapide FMX
- TsgcMapLocation

Tuiles, couches et fournisseurs

Une carte est dessinée à partir de **tuiles** : des images carrées, de 256 pixels de côté par défaut, qui couvrent le monde à chaque niveau de zoom. Au zoom 0, une seule tuile montre le monde entier, et chaque niveau supplémentaire double le nombre de tuiles dans chaque direction. Une tuile est adressée par son zoom, x et y, le schéma z/x/y utilisé par OpenStreetMap et par presque tous les serveurs de tuiles.
La collection **Layers** de TsgcVCLMap et de TsgcFMXMap indique d'où viennent les tuiles. Chaque élément est un **TsgcMapLayer** (unité sgcMaps_Classes), et son **TileSourceType** sélectionne la source :

TileSourceType	Origine des tuiles	Chemin ou URL
mtsFolder	Un dossier z/x/y de fichiers .png, .jpg ou .jpeg sur le disque.	TileSourcePath
mtsMBTiles	Un fichier MBTiles raster.	TileSourcePath
mtsHTTP	Un serveur de tuiles, par un modèle d'URL.	URLTemplate
mtsVectorHTTP	Un serveur de tuiles vectorielles. Voir Tuiles vectorielles.	URLTemplate
mtsVectorMBTiles	Un fichier MBTiles vectoriel. Voir Tuiles vectorielles.	TileSourcePath

Les sources de dossier et MBTiles fonctionnent sans connexion réseau ; Tuiles hors ligne et téléchargeur de tuiles explique comment les préparer.

Fournisseurs intégrés

La propriété **Provider** d'une couche désigne l'un des serveurs de tuiles intégrés de **TsgcMapTileProvider**. Appelez **sgcMapApplyProvider** (unité sgcMaps_Providers) après l'avoir définie : elle règle TileSourceType sur mtsHTTP, copie dans la couche le modèle d'URL et le zoom maximal du fournisseur, et renseigne Attribution lorsqu'elle est vide. Une couche dont le Provider n'est pas mtpCustom télécharge les tuiles de son fournisseur même sans cet appel, mais MaxZoom conserve alors sa valeur par défaut de 19 et les propriétés de la couche n'affichent pas les réglages du fournisseur.

Fournisseur	Clé d'API	Zoom max	Tuiles Retina
mtpOpenStreetMap	Non	19	Non
mtpOpenTopoMap	Non	17	Non
mtpEsriWorldImagery	Non	19	Non
mtpEsriWorldStreet	Non	19	Non
mtpCartoLight	Non	20	Oui
mtpCartoDark	Non	20	Oui
mtpMapTilerStreets	Oui	22	Oui
mtpMapTilerSatellite	Oui	20	Oui
mtpThunderforestCycle	Oui	22	Oui
mtpThunderforestTransport	Oui	22	Oui
mtpStadiaAlidadeSmooth	Oui	20	Oui

mtpStadiaAlidadeSmoothDark	Oui	20	Oui

Renseignez la clé des fournisseurs qui en exigent une dans la propriété **ApiKey** de la couche. Les serveurs publics ont des politiques d'utilisation : tile.openstreetmap.org, par exemple, interdit l'usage intensif et les téléchargements en masse, une application en production doit donc utiliser un fournisseur commercial ou son propre serveur de tuiles.

Modèles d'URL

Pour tout autre serveur, réglez TileSourceType sur mtsHTTP et écrivez son **URLTemplate**. Ces paramètres sont remplacés pour chaque tuile :

Para-mètre	Remplacé par
{z}, {x}, {y}	Le zoom, ainsi que la colonne et la ligne de la tuile.
{s}	Une lettre de sous-domaine, alternée par tuile pour répartir les requêtes entre les serveurs. Les lettres sont a, b et c sauf si le fournisseur en définit d'autres.
{r}	@2x lorsque la couche a Retina défini à True, vide sinon.
{apikey}	L'ApiKey de la couche.

La même expansion est disponible sous le nom **sgcMapExpandTileURL**. **sgcMapProviderNames** remplit une liste de chaînes avec les noms des fournisseurs, et **sgcMapGetProviderInfo** renvoie un enregistrement **TsgcMapProviderInfo** avec le nom, le modèle d'URL, les sous-domaines, l'attribution, le zoom maximal et l'indication que le fournisseur exige une clé d'API ou prend en charge les tuiles retina.

```
procedure TForm2.ListProviders;
var
  oInfo: TsgcMapProviderInfo;
begin
  // every provider name; Objects[i] holds Ord(provider)
  sgcMapProviderNames(cboProvider.Items);

  oInfo := sgcMapGetProviderInfo(mtpOpenTopoMap);
  memoLog.Lines.Add(Format('%s: max zoom %d, API key %s',
    [oInfo.Name, oInfo.MaxZoom, BoolToStr(oInfo.RequiresApiKey, True)]));
  // the URL of tile 10/523/380
  memoLog.Lines.Add(sgcMapExpandTileURL(oInfo.URLTemplate, oInfo.Subdomains,
    10, 523, 380, False, ''));
end;
```

Plusieurs couches

Une couche est soit une **couche de base** (IsBaseLayer True, la valeur par défaut), soit une **surcouche**. La carte dessine la première couche de base visible, puis chaque surcouche visible par-dessus, dans l'ordre de la collection. Opacity définit la transparence d'une couche, MinZoom et MaxZoom limitent les niveaux de zoom où elle est dessinée, et Retina demande au fournisseur des tuiles à deux fois la résolution.
Définissez **Options.ShowLayerSwitcher** à True pour laisser l'utilisateur choisir les couches sur la carte. Les couches de base se comportent comme des boutons radio : en afficher une avec **SetLayerVisible** masque les autres couches de base. L'événement **OnLayerChanged** indique la couche qui a été affichée ou masquée.

```
procedure TForm2.SetupLayers;
var
  oLayer: TsgcMapLayer;
begin
  // base layer: a built-in provider
  oLayer := sgcVCLMap1.Layers.Add;
  oLayer.Name := 'CARTO Light';
```

```
  oLayer.Provider := mtpCartoLight;
  sgcMapApplyProvider(oLayer);
  oLayer.Retina := True;
  oLayer.CacheFolder := ExtractFilePath(ParamStr(0)) + 'cache\carto';

  // second base layer, hidden until chosen in the layer switcher
  oLayer := sgcVCLMap1.Layers.Add;
  oLayer.Name := 'Satellite';
  oLayer.Provider := mtpMapTilerSatellite;
  sgcMapApplyProvider(oLayer);
  oLayer.ApiKey := 'YOUR_MAPTILER_KEY';
  oLayer.Visible := False;

  // overlay: any tile server through a URL template
  oLayer := sgcVCLMap1.Layers.Add;
  oLayer.Name := 'Hiking trails';
  oLayer.TileSourceType := mtsHTTP;
  oLayer.URLTemplate := 'https://tile.waymarkedtrails.org/hiking/{z}/{x}/{y}.png';
  oLayer.Attribution := '(c) waymarkedtrails.org';
  oLayer.IsBaseLayer := False;
  oLayer.Opacity := 0.8;

  sgcVCLMap1.Options.ShowLayerSwitcher := True;
end;
```

Cache de tuiles

Définissez **CacheFolder** pour conserver sur le disque chaque tuile téléchargée, dans une arborescence z/x/y. Les tuiles déjà présentes dans le cache sont dessinées sans aucune requête au serveur, si bien qu'une carte que l'utilisateur a vue une fois continue de fonctionner sans connexion. La carte conserve aussi les tuiles récentes en mémoire ; **RefreshTiles** recharge les tuiles de la vue. Une tuile qui ne peut pas être chargée déclenche **OnTileError** avec la couche, l'adresse de la tuile et le texte de l'erreur, et **LastRenderMissingTiles** compte les tuiles manquantes lors du dernier affichage.

Disponibilité

Framework	Version minimale
VCL, tuiles depuis un dossier	Delphi 7
VCL, tuiles depuis un serveur (fournisseurs, modèles d'URL, cache)	Delphi 2009
VCL et FireMonkey, MBTiles	Delphi XE7 (FireDAC SQLite)
FireMonkey	Delphi XE6
Lazarus (LCL)	Tuiles depuis un dossier uniquement

De Delphi 2009 à XE7, les tuiles sont téléchargées avec Indy 10, et https requiert les bibliothèques OpenSSL 1.0.2 (libeay32.dll et ssleay32.dll) à côté de l'exécutable. À partir de Delphi XE8, la carte utilise THTTPClient avec la prise en charge TLS du système d'exploitation, et n'a besoin d'aucune DLL. Sous Delphi 7, Delphi 2007 et Lazarus, il n'y a pas de téléchargement de tuiles, utilisez donc un dossier de tuiles.

Voir aussi

- TsgcVCLMap
- Tuiles vectorielles
- Tuiles hors ligne et téléchargeur de tuiles

Tuiles vectorielles

Une **tuile vectorielle** stocke les routes, les bâtiments, les eaux et les noms de lieux sous forme de géométrie plutôt que de pixels, au format Mapbox Vector Tile (MVT). sgcMaps décode les tuiles et les dessine lui-même, en suivant un **style** : un JSON de style MapLibre ou Mapbox GL qui indique quelles entités dessiner, avec quelles couleurs et quelles largeurs, à quels niveaux de zoom. Un même jeu de tuiles vectorielles peut donc paraître clair, sombre ou minimal selon le style, et les tuiles restent nettes quelle que soit la densité de pixels.

Une couche vectorielle

Ajoutez une couche à la carte avec l'un des types de source vectorielle :

- **mtsVectorHTTP** : des tuiles depuis un serveur. URLTemplate utilise les mêmes paramètres que les tuiles raster, par exemple https://api.maptiler.com/tiles/v3/{z}/{x}/{y}.pbf?key={apikey}, et ApiKey renseigne {apikey}.
- **mtsVectorMBTiles** : des tuiles depuis un fichier MBTiles vectoriel local défini dans TileSourcePath, tel qu'un fichier créé avec tilemaker ou planetiler.

VectorStyleFile est le chemin d'un fichier JSON de style enregistré en UTF-8. Lorsqu'il est vide, la carte utilise le style intégré, un style clair pour le schéma **OpenMapTiles** (eau, occupation du sol, usage du sol, parcs, bâtiments, routes, limites, cours d'eau et noms de lieux, de routes et d'eaux). Ce schéma est celui qu'utilisent MapTiler, OpenFreeMap et la plupart des tuiles vectorielles OpenStreetMap. Avec **Retina** défini à True, les tuiles sont rendues à 512 pixels au lieu de 256, pour les écrans à haute densité de pixels. Les étiquettes des couches de symboles du style sont collectées par tuile et dessinées à la verticale par la carte, même lorsque la carte pivote.

```
procedure TForm2.SetupVectorLayers;
var
  oLayer: TsgcMapLayer;
begin
  // vector tiles from a server, drawn with a MapLibre style file
  oLayer := sgcVCLMap1.Layers.Add;
  oLayer.Name := 'MapTiler vector';
  oLayer.TileSourceType := mtsVectorHTTP;
  oLayer.URLTemplate :=
    'https://api.maptiler.com/tiles/v3/{z}/{x}/{y}.pbf?key={apikey}';
  oLayer.ApiKey := 'YOUR_MAPTILER_KEY';
  oLayer.Attribution := '(c) MapTiler (c) OpenStreetMap contributors';
  oLayer.VectorStyleFile := 'styles\dark.json';
  // renders every tile at 512 pixels
  oLayer.Retina := True;

  // the same data offline, from a vector MBTiles file and the built-in style
  oLayer := sgcVCLMap1.Layers.Add;
  oLayer.Name := 'Offline vector';
  oLayer.TileSourceType := mtsVectorMBTiles;
  oLayer.TileSourcePath := 'C:\Maps\andorra.mbtiles';
  oLayer.VectorStyleFile := '';
  oLayer.Visible := False;
end;
```

La source de tuiles vectorielles

La couche crée un **TsgcMapTileSource_MVT** (unité sgcMaps_TileSource_MVT). Vous pouvez aussi en créer un vous-même pour rendre des tuiles en dehors d'une carte :

Membre	Description
SourceKind	mvsHTTP ou mvsMBTiles.

URLTemplate, ApiKey, UserAgent	Le serveur de mvsHTTP.
Path	Le fichier .mbtiles de mvsMBTiles.
StyleJSON, StyleFile	Le style. StyleJSON l'emporte sur StyleFile, et lorsque les deux sont vides, Open charge le style intégré.
TileSize	256 (par défaut) ou 512 pixels.
MaxDataZoom	Le zoom le plus élevé stocké dans les données, 14 par défaut. Les niveaux de zoom plus profonds sont dessinés à partir des tuiles de ce niveau.
CacheSize	Le nombre de tuiles rendues conservées en mémoire, 256 par défaut.
Open, Close	Open crée le lecteur et charge le style ; il lève une exception pour un style invalide.
GetTile	Renvoie la tuile rendue sous forme d'image PNG.
GetTileLabels	Renvoie les étiquettes d'une tuile, avec leur position.
ClearCache	Abandonne les tuiles rendues. Appelez-le après avoir modifié le style ou TileSize.
Style	Le TsgcMapVectorStyle utilisé, valide après Open.

Décodage et styles

L'unité **sgcMaps_MVT** contient les briques de base. **TsgcMapVectorTile** décode une tuile à partir d'octets, d'un flux ou d'un fichier, brute ou compressée avec gzip ou zlib, en couches (**TsgcMapVectorLayer**) et en entités (**TsgcMapVectorFeature**) avec leur type de géométrie, leurs étiquettes et leur géométrie. **TsgcMapVectorStyle** charge un style avec LoadFromJSON ou LoadFromFile, ou le style intégré avec LoadDefault.

```
procedure TForm2.ListRoadClasses(const aFileName: string);
var
  oTile: TsgcMapVectorTile;
  oLayer: TsgcMapVectorLayer;
  i: Integer;
begin
  oTile := TsgcMapVectorTile.Create;
  try
    // raw, gzip or zlib compressed protobuf
    oTile.LoadFromFile(aFileName);
    oLayer := oTile.LayerByName('transportation');
    if oLayer <> nil then
      for i := 0 to oLayer.FeatureCount - 1 do
        if oLayer.Features[i].GeomType = mvgLineString then
          memoLog.Lines.Add(oLayer.Features[i].GetTagAsString('class'));
  finally
    oTile.Free;
  end;
end;
```

La prise en charge des styles couvre les types de couche background, fill, line et symbol ; les couches raster, fill-extrusion, circle, heatmap et hillshade sont ignorées. Les filtres, les fonctions de zoom et les expressions courantes (get, has, match, case, step, interpolate, coalesce et les opérateurs de comparaison) sont évalués. Les icônes, les sprites, les motifs, les halos de texte et les glyphes ne sont pas utilisés.

Disponibilité

Framework	Version minimale
VCL et FireMonkey, tuiles vectorielles depuis un serveur	Delphi XE2 (FireMonkey à partir de XE6)
VCL et FireMonkey, MBTiles vectoriel	Delphi XE7 (FireDAC SQLite)
Lazarus (LCL)	Non disponible

Voir aussi

- Tuiles, couches et fournisseurs
- TsgcVCLMap
- TsgcFMXMap

Marqueurs et regroupement

Un **marqueur** épingle un point de la carte. La collection **Markers** de TsgcVCLMap et de TsgcFMXMap contient des éléments **TsgcMapMarker**, et **AddMarker** en ajoute un avec un texte de fenêtre contextuelle et une infobulle en un seul appel.

Propriétés des marqueurs

Propriété	Description
Latitude, Longitude	La position du marqueur.
PopupText	Le texte de la fenêtre contextuelle ouverte par ShowPopup, en général depuis OnMarkerClick. HidePopup la ferme et ActivePopupMarker est le marqueur dont la fenêtre contextuelle est ouverte.
TooltipText	Le texte affiché tant que la souris est sur le marqueur.
Color	La couleur de l'épingle par défaut. MarkerOptions.DefaultColor, DefaultWidth et DefaultHeight définissent l'épingle de chaque marqueur.
IconBitmap	Une icône personnalisée, chargée avec LoadFromFile ou LoadFromStream à partir d'une image PNG (Delphi 2009 et versions ultérieures, et Lazarus), JPEG ou BMP. Elle est stockée dans le formulaire, une icône définie à la conception n'a donc besoin d'aucun fichier à l'exécution.
AnchorX, AnchorY	Le point de l'icône qui touche la coordonnée, en fraction de sa largeur et de sa hauteur. La valeur par défaut, 0.5 et 1, est le bas au centre.
Draggable	Permet à l'utilisateur de déplacer le marqueur. OnMarkerDragStart, OnMarkerDrag et OnMarkerDragEnd suivent le déplacement.
Visible, Tag, Data	La visibilité, ainsi qu'un entier et une chaîne pour vos propres données.

MarkerAtScreen renvoie le marqueur situé en un point du contrôle, **FitToMarkers** zoome et centre la vue sur tous les marqueurs, et **ClearMarkers** les supprime tous.

```
procedure TForm2.AddStore;
var
  oMarker: TsgcMapMarker;
begin
  oMarker := sgcVCLMap1.Markers.Add;
  oMarker.Latitude := 41.3874;
  oMarker.Longitude := 2.1686;
  oMarker.PopupText := 'Barcelona store';
  oMarker.TooltipText := 'Open from 9:00 to 20:00';
  oMarker.Data := 'store-012';
  // a custom icon whose bottom center touches the coordinate
  oMarker.IconBitmap.LoadFromFile('icons\store.png');
  oMarker.AnchorX := 0.5;
  oMarker.AnchorY := 1.0;
  oMarker.Draggable := True;
  sgcVCLMap1.OnMarkerDragEnd := sgcVCLMap1MarkerDragEnd;
end;

procedure TForm2.sgcVCLMap1MarkerDragEnd(Sender: TObject;
  aMarker: TsgcMapMarker);
```

```
begin
  memoLog.Lines.Add(Format('%s moved to %.5f, %.5f',
    [aMarker.Data, aMarker.Latitude, aMarker.Longitude]));
end;
```

Regroupement

Des milliers de marqueurs rendent une carte lente et illisible lorsqu'elle est dézoomée. Avec **MarkerOptions.Clustering.Enabled** défini à True, les marqueurs proches sont fusionnés en un seul cercle qui indique combien de marqueurs il contient. L'écran est divisé en cellules carrées de **GridSize** pixels, et une cellule contenant au moins **MinClusterSize** marqueurs devient un regroupement dessiné au centre de ses marqueurs. Au-dessus de **MaxZoom** (18 par défaut), les marqueurs ne sont jamais regroupés. **Color** et **FontColor** définissent l'aspect des cercles.
Un clic sur un regroupement déclenche **OnClusterClick** avec le nombre de marqueurs et la position du regroupement. Lorsque le gestionnaire laisse aHandled à False et que **ZoomOnClick** vaut True, la valeur par défaut, la carte zoome alors jusqu'à ce que les marqueurs du regroupement se séparent. **ClusterAtScreen** trouve le regroupement situé en un point du contrôle, et **DrawnClusterCount** et **DrawnMarkerCount** indiquent ce que le dernier affichage a dessiné.

```
procedure TForm2.AddManyMarkers;
var
  i: Integer;
begin
  sgcVCLMap1.MarkerOptions.Clustering.Enabled := True;
  sgcVCLMap1.MarkerOptions.Clustering.GridSize := 60;
  // from zoom 17 on, every marker is drawn on its own
  sgcVCLMap1.MarkerOptions.Clustering.MaxZoom := 16;
  sgcVCLMap1.OnClusterClick := sgcVCLMap1ClusterClick;

  sgcVCLMap1.Markers.BeginUpdate;
  try
    for i := 1 to 2000 do
      sgcVCLMap1.AddMarker(40 + Random * 3, -4 + Random * 5,
        'Point ' + IntToStr(i));
  finally
    sgcVCLMap1.Markers.EndUpdate;
  end;
  sgcVCLMap1.FitToMarkers;
end;

procedure TForm2.sgcVCLMap1ClusterClick(Sender: TObject; aCount: Integer;
  aLatitude, aLongitude: Double; var aHandled: Boolean);
begin
  memoLog.Lines.Add(Format('%d markers near %.4f, %.4f',
    [aCount, aLatitude, aLongitude]));
end;
```

Encadrez les ajouts en masse par Markers.BeginUpdate et Markers.EndUpdate, pour que la carte se redessine une seule fois au lieu d'une fois par marqueur. La fonction de regroupement elle-même, **sgcMapClusterPoints**, est publique dans sgcMaps_Geometry pour vos propres points d'écran.

Disponibilité

Framework	Version minimale
VCL	Delphi 7
FireMonkey	Delphi XE6
Lazarus (LCL)	Oui

Voir aussi

- TsgcVCLMap
- Formes et étiquettes
- Cartes de chaleur

Formes et étiquettes

Outre les marqueurs, TsgcVCLMap et TsgcFMXMap dessinent des polylignes, des polygones, des rectangles, des cercles et des étiquettes de texte au-dessus des tuiles. Chaque type a sa propre collection et une méthode Add sur le contrôle qui crée un élément et le renvoie.

Collection	Méthode d'ajout	Élément	Propriétés principales
Polylines	AddPolyline	TsgcMapPolyline	AddPoint, ClearPoints, PointCount, Color, Width, Opacity, Dashed, Geodesic, Editable
Polygons	AddPolygon	TsgcMapPolygon	AddPoint, Holes, FillColor, FillOpacity, StrokeColor, StrokeWidth, Editable
Rectangles	AddRectangle	TsgcMapRectangle	NorthLatitude, WestLongitude, SouthLatitude, EastLongitude, FillColor, FillOpacity, StrokeColor, StrokeWidth, Editable
Circles	AddCircle	TsgcMapCircle	CenterLat, CenterLon, RadiusMeters, FillColor, FillOpacity, StrokeColor, StrokeWidth
Labels	AddLabel	TsgcMapLabel	Latitude, Longitude, Text, FontSize, FontColor, BackgroundColor, ShowBackground

Chaque élément possède aussi Visible, et les formes ont un Tag pour votre usage.

Polylignes

Une polyligne relie ses points par des lignes droites sur la carte. Avec **Dashed**, elle est dessinée en pointillés, et avec **Geodesic**, chaque segment suit le grand cercle, le plus court chemin sur le globe, ce qui convient aux vols et aux longues distances. **FitToPolyline** ajuste la vue à une polyligne, et les itinéraires des composants de calcul d'itinéraires sont eux aussi ajoutés sous forme de polylignes.

Polygones avec trous

Un polygone est rempli à l'intérieur de son anneau extérieur. **Holes** ajoute des anneaux intérieurs laissés vides : **AddHole** renvoie l'indice d'un nouveau trou, et **AddPoint** lui ajoute un sommet. Les polygones GeoJSON et KML avec trous sont importés de la même façon.

```
procedure TForm2.AddShapes;
var
  oLine: TsgcMapPolyline;
  oPolygon: TsgcMapPolygon;
  iHole: Integer;
begin
  // a dashed great circle from Madrid to New York
  oLine := sgcVCLMap1.AddPolyline;
  oLine.AddPoint(40.4722, -3.5608);
  oLine.AddPoint(40.6413, -73.7781);
  oLine.Geodesic := True;
  oLine.Dashed := True;
  oLine.Color := clNavy;
  oLine.Width := 3;

  // a polygon with a hole, editable with the mouse
  oPolygon := sgcVCLMap1.AddPolygon;
  oPolygon.AddPoint(40.425, -3.700);
  oPolygon.AddPoint(40.425, -3.675);
```

```
  oPolygon.AddPoint(40.405, -3.675);
  oPolygon.AddPoint(40.405, -3.700);
  iHole := oPolygon.Holes.AddHole;
  oPolygon.Holes.AddPoint(iHole, 40.418, -3.692);
  oPolygon.Holes.AddPoint(iHole, 40.418, -3.683);
  oPolygon.Holes.AddPoint(iHole, 40.412, -3.683);
  oPolygon.Holes.AddPoint(iHole, 40.412, -3.692);
  oPolygon.FillColor := clGreen;
  oPolygon.FillOpacity := 0.3;
  oPolygon.Editable := True;

  sgcVCLMap1.AddRectangle(40.43, -3.72, 40.42, -3.70).FillOpacity := 0.2;
  sgcVCLMap1.AddCircle(40.4168, -3.7038, 250).StrokeColor := clRed;
  sgcVCLMap1.AddLabel(40.4168, -3.7038, 'Puerta del Sol').FontSize := 12;
  sgcVCLMap1.OnShapeEdited := sgcVCLMap1ShapeEdited;
end;

procedure TForm2.sgcVCLMap1ShapeEdited(Sender: TObject; aItem: TCollectionItem);
begin
  if aItem is TsgcMapPolygon then
    memoLog.Lines.Add(Format('Polygon %d has %d points now',
      [aItem.Index, TsgcMapPolygon(aItem).PointCount]));
end;
```

Modification

Définissez **Editable** à True sur une polyligne, un polygone ou un rectangle pour laisser l'utilisateur le remodeler à la souris : un sommet se déplace en le faisant glisser, la poignée au milieu d'un segment insère un nouveau sommet, et un clic droit sur un sommet le supprime. Lorsqu'une modification se termine, la carte déclenche **OnShapeEdited** avec l'élément modifié. Les cercles et les étiquettes ne sont pas modifiables.

Disponibilité

Framework	Version minimale
VCL	Delphi 7
FireMonkey	Delphi XE6
Lazarus (LCL)	Oui

Voir aussi

- TsgcVCLMap
- Marqueurs et regroupement
- Fonctions de géométrie
- GeoJSON, GPX et KML

Cartes de chaleur

Une **carte de chaleur** montre où se concentrent des points pondérés : relevés de capteurs, ventes, incidents ou positions GPS. Chaque point répartit son poids sur un cercle, les poids s'additionnent, et le total est coloré du froid au chaud.
La collection **Heatmaps** de TsgcVCLMap et de TsgcFMXMap contient des éléments **TsgcMapHeatmap**. Ajoutez les points avec **AddPoint**(latitude, longitude, poids) ; le poids vaut 1 lorsqu'il est omis. Count, Points et Clear donnent accès aux points, et un enregistrement **TsgcMapHeatmapPoint** contient l'un d'eux.

Propriété	Par défaut	Description
Radius	25	Le rayon d'influence de chaque point, en pixels.
Opacity	0.6	L'opacité de la couleur la plus chaude.
MaxIntensity	0	L'intensité dessinée avec la couleur la plus chaude. 0 adapte les couleurs à l'intensité la plus élevée dessinée ; une valeur fixe garde les couleurs comparables d'une vue à l'autre.
Gradient	mhgDefault	mhgDefault (bleu, cyan, citron vert, jaune, rouge), mhgFire, mhgCool ou mhgGrayscale.
Visible	True	Affiche ou masque la carte de chaleur.

```
procedure TForm2.ShowReadings;
var
  oHeatmap: TsgcMapHeatmap;
  i: Integer;
begin
  // FReadings: array of TsgcMapHeatmapPoint, loaded elsewhere
  oHeatmap := sgcVCLMap1.Heatmaps.Add;
  oHeatmap.Radius := 30;
  oHeatmap.Gradient := mhgFire;
  oHeatmap.Opacity := 0.7;
  // 0 scales the colors to the hottest spot drawn
  oHeatmap.MaxIntensity := 0;
  for i := 0 to High(FReadings) do
    oHeatmap.AddPoint(FReadings[i].Latitude, FReadings[i].Longitude,
      FReadings[i].Weight);
end;
```

La carte de chaleur est dessinée en pixels d'écran, sa forme suit donc le zoom : zoomer sépare les taches. La fonction de rendu, **sgcMapRenderHeatmap** dans sgcMaps_Geometry, est elle aussi publique et remplit un tampon ARGB à partir de vos propres points d'écran.

Disponibilité

Framework	Version minimale
VCL	Delphi 7
FireMonkey	Delphi XE6
Lazarus (LCL)	Oui

Voir aussi

- TsgcVCLMap
- Marqueurs et regroupement

GeoJSON, GPX et KML

sgcMaps lit et écrit les trois formats d'échange courants de données géographiques. Les fonctions d'import ajoutent des éléments aux collections d'une carte, et les fonctions d'export écrivent les éléments d'une carte, si bien que n'importe quelle collection de TsgcVCLMap ou de TsgcFMXMap peut être transmise.

Format	Unité	Import	Export
GeoJSON (RFC 7946)	sgcMaps_GeoJSON	Point et MultiPoint vers des marqueurs, LineString et MultiLineString vers des polylignes, Polygon et MultiPolygon vers des polygones avec trous, GeometryCollection, Feature et FeatureCollection.	Une FeatureCollection avec des propriétés simplestyle. Les rectangles sont écrits sous forme de polygones.
GPX	sgcMaps_GPX	Points de passage vers des marqueurs, traces et itinéraires vers des polylignes.	Marqueurs vers des points de passage et polylignes vers des traces. Les polygones et les rectangles sont écrits sous forme de traces fermées lorsque ExportPolygonsAsTracks vaut True.
KML	sgcMaps_KML	Point vers des marqueurs, LineString vers des polylignes, Polygon avec ses limites intérieures vers des polygones, avec les couleurs des styles en ligne et partagés.	Un Placemark par élément avec un style en ligne. Les rectangles sont écrits sous forme de polygones.

Chaque format possède une classe (**TsgcMapGeoJSON**, **TsgcMapGPX**, **TsgcMapKML**) avec LoadFromString, LoadFromFile, AsText, SaveToStream et SaveToFile, et les mêmes opérations sous forme de fonctions simples : **sgcMapLoadGeoJSON**, **sgcMapLoadGeoJSONFile**, **sgcMapGeoJSONAsText**, **sgcMapSaveGeoJSONFile**, ainsi que les fonctions GPX et KML correspondantes. Une collection transmise à nil est ignorée.

```
procedure TForm2.ImportAndExport;
begin
  // GeoJSON: points, lines and polygons with holes (Delphi XE6 and later)
  sgcMapLoadGeoJSONFile('parks.geojson', sgcVCLMap1.Markers,
    sgcVCLMap1.Polylines, sgcVCLMap1.Polygons);
  // GPX: waypoints become markers, tracks and routes become polylines
  sgcMapLoadGPXFile('ride.gpx', sgcVCLMap1.Markers, sgcVCLMap1.Polylines);
  // KML: placemarks with their colors
  sgcMapLoadKMLFile('areas.kml', sgcVCLMap1.Markers, sgcVCLMap1.Polylines,
    sgcVCLMap1.Polygons);
  sgcVCLMap1.FitToMarkers;

  // everything the map shows, rectangles included
  sgcMapSaveGeoJSONFile('export.geojson', sgcVCLMap1.Markers,
    sgcVCLMap1.Polylines, sgcVCLMap1.Polygons, sgcVCLMap1.Rectangles);
  sgcMapSaveKMLFile('export.kml', sgcVCLMap1.Markers, sgcVCLMap1.Polylines,
    sgcVCLMap1.Polygons, sgcVCLMap1.Rectangles);
end;
```

Statistiques de traces GPX

TsgcMapGPX conserve des statistiques de chaque trace du dernier fichier qu'il a chargé. **TrackStatsCount** et **TrackStats** renvoient un enregistrement **TsgcMapGPXTrackStats** avec le nom, le nombre de points, la distance en mètres, l'altitude minimale et maximale, ainsi que le dénivelé positif et négatif total. Avec **ColorByElevation** dé-

fini à True, chaque trace est découpée en morceaux colorés selon l'altitude, de LowColor à HighColor en passant par MidColor de **ElevationGradient** (un TsgcMapElevationGradient).

```
procedure TForm2.ShowTrackStats;
var
  oGPX: TsgcMapGPX;
  oStats: TsgcMapGPXTrackStats;
  i: Integer;
begin
  oGPX := TsgcMapGPX.Create;
  try
    // splits every track into pieces colored by elevation
    oGPX.ColorByElevation := True;
    oGPX.LoadFromFile('ride.gpx', sgcVCLMap1.Markers, sgcVCLMap1.Polylines);
    for i := 0 to oGPX.TrackStatsCount - 1 do
    begin
      oStats := oGPX.TrackStats[i];
      memoLog.Lines.Add(Format('%s: %.1f km, up %.0f m, down %.0f m',
        [oStats.Name, oStats.Distance / 1000, oStats.TotalAscent,
        oStats.TotalDescent]));
    end;
  finally
    oGPX.Free;
  end;
end;
```

Disponibilité

Framework	Version minimale
GPX et KML, VCL et FireMonkey	Delphi 7 (FireMonkey à partir de XE6)
GeoJSON, VCL et FireMonkey	Delphi XE6 (System.JSON)
Lazarus (LCL)	Non disponible

GPX et KML sont analysés avec TXMLDocument : MSXML sous Windows, et OmniXML sur les autres plateformes à partir de Delphi XE7.

Voir aussi

- Formes et étiquettes
- Marqueurs et regroupement
- TsgcVCLMap

Fonctions de géométrie

L'unité **sgcMaps_Geometry** contient des fonctions qui travaillent sur des coordonnées, sans aucun contrôle de carte, si bien qu'elles servent aussi bien un service ou une application console qu'un formulaire. Les coordonnées sont en degrés WGS84, les distances sont en mètres sur une sphère de rayon terrestre moyen (SGC_MAP_EARTH_RADIUS, 6371008.8 m), et les orientations sont en degrés dans le sens horaire à partir du nord géographique, de 0 à 360.

Fonction	Renvoie
sgcMapDistance	La distance orthodromique entre deux points, en mètres.
sgcMapInitialBearing	L'orientation initiale d'un point vers un autre.
sgcMapDestinationPoint	Le point atteint depuis un point de départ avec une orientation et une distance.
sgcMapGeodesicInterpolate	Les points du grand cercle entre deux points, découpé en un nombre de segments.
sgcMapPointInPolygon	True lorsqu'un point est à l'intérieur d'un anneau.
sgcMapPointInPolygonWithHoles	True lorsqu'un point est à l'intérieur d'un anneau extérieur et à l'extérieur de tous ses trous.
sgcMapSimplify	Une ligne avec moins de points (Douglas-Peucker), dans une tolérance en mètres.
sgcMapNearestPointOnPolyline	La distance d'un point à une ligne, avec le point le plus proche et l'indice de son segment.

Les tableaux de points utilisent **TsgcMapCoordArray**, un tableau de l'enregistrement **TsgcMapCoord**, et **sgcMapCoord** construit une coordonnée.

```
procedure TForm2.GeometryExamples;
var
  dDistance, dBearing, dLat, dLon: Double;
  vArea, vTrack: TsgcMapCoordArray;
  iSegment: Integer;
begin
  // Madrid to Barcelona: meters and initial bearing
  dDistance := sgcMapDistance(40.4168, -3.7038, 41.3874, 2.1686);
  dBearing := sgcMapInitialBearing(40.4168, -3.7038, 41.3874, 2.1686);
  memoLog.Lines.Add(Format('%.1f km, bearing %.1f', [dDistance / 1000,
    dBearing]));

  // the point 10 km to the north east of Madrid, inside the area?
  sgcMapDestinationPoint(40.4168, -3.7038, 45, 10000, dLat, dLon);
  SetLength(vArea, 3);
  vArea[0] := sgcMapCoord(40.3, -3.9);
  vArea[1] := sgcMapCoord(40.7, -3.5);
  vArea[2] := sgcMapCoord(40.3, -3.3);
  if sgcMapPointInPolygon(dLat, dLon, vArea) then
    memoLog.Lines.Add('Inside the area');

  // a great circle track, simplified to 50 m, and the distance to it
  vTrack := sgcMapGeodesicInterpolate(40.4168, -3.7038, 41.3874, 2.1686, 64);
  vTrack := sgcMapSimplify(vTrack, 50);
  dDistance := sgcMapNearestPointOnPolyline(sgcVCLMap1.CenterLatitude,
    sgcVCLMap1.CenterLongitude, vTrack, iSegment, dLat, dLon);
```

```
  memoLog.Lines.Add(Format('%.0f m to segment %d', [dDistance, iSegment]));
end;
```

Limites et calcul des tuiles

FitBounds des contrôles de carte prend un rectangle (nord, ouest, sud, est) et l'affiche avec le zoom le plus grand qui convient, en conservant une marge en pixels ; une longitude ouest supérieure à la longitude est traverse l'antiméridien. FitToMarkers et FitToPolyline font de même pour les marqueurs et pour une polyligne. L'unité sgcMaps_Classes ajoute le calcul des tuiles de la projection Web Mercator : **sgcMapLatLonToTileF** et **sgcMapTileFToLatLon** convertissent entre coordonnées et numéros de tuiles fractionnaires, **sgcMapLatLonToPixel** et **sgcMapPixelToLatLon** entre coordonnées et pixels du monde, et **sgcMapTileCount** donne le nombre de tuiles par côté à un zoom donné.

```
procedure TForm2.ShowSpain;
var
  dTileX, dTileY: Double;
begin
  // the largest zoom that shows the whole box, with 20 pixels of margin
  sgcVCLMap1.FitBounds(43.8, -9.3, 36.0, 3.3, 20);

  // the tile under the center of the view
  if sgcMapLatLonToTileF(sgcVCLMap1.CenterLatitude,
    sgcVCLMap1.CenterLongitude, sgcVCLMap1.Zoom, dTileX, dTileY) then
    memoLog.Lines.Add(Format('Tile %d/%d/%d', [sgcVCLMap1.Zoom, Trunc(dTileX),
      Trunc(dTileY)]));
end;
```

Disponibilité

Framework	Version minimale
VCL, FireMonkey, console et services	Delphi 7
Lazarus	Oui

Voir aussi

- Formes et étiquettes
- Marqueurs et regroupement
- TsgcVCLMap

Rendu de cartes statiques

Un contrôle de carte peut dessiner sans fenêtre. Un TsgcVCLMap créé sans parent dessine dans un bitmap à la demande, ce qui transforme une application console, un serveur web ou un service Windows en moteur de rendu de cartes statiques : des miniatures pour un rapport, une carte jointe à un e-mail, ou une image servie par HTTP.

Méthodes de rendu

Méthode	Description
RenderToBitmap(aBitmap, aIncludeControls)	Dessine la carte dans un bitmap de sa propre taille. Un bitmap vide reçoit la taille du contrôle.
SaveToPNG(aFileName, aWidth, aHeight, aIncludeControls)	Écrit un fichier PNG de la taille indiquée.
SaveToBMP(aFileName, aWidth, aHeight, aIncludeControls)	Écrit un fichier BMP de la taille indiquée.

L'image montre les tuiles, les surcouches, l'échelle graphique et l'attribution. Les boutons de zoom, le sélecteur de couches et la boussole sont omis, sauf si aIncludeControls vaut True. Les tuiles sont lues pendant le rendu, et les tuiles d'un serveur sont alors téléchargées, l'appel bloque donc jusqu'à leur arrivée. Ensuite, **LastRenderMissingTiles** indique combien de tuiles n'ont pas pu être chargées, et **AttributionText** renvoie le texte de droit d'auteur à afficher à côté de l'image.

```
program StaticMap;

{$APPTYPE CONSOLE}

uses
  SysUtils,
  sgcMaps_Classes, sgcMaps_Providers, sgcMaps_VCL;

var
  oMap: TsgcVCLMap;
  oLayer: TsgcMapLayer;
begin
  // a map control with no parent: no window, it only renders
  oMap := TsgcVCLMap.Create(nil);
  try
    oMap.Options.ShowZoomControls := False;
    oLayer := oMap.Layers.Add;
    oLayer.Provider := mtpOpenStreetMap;
    sgcMapApplyProvider(oLayer);
    oLayer.CacheFolder := ExtractFilePath(ParamStr(0)) + 'tile_cache';

    oMap.AddMarker(48.8584, 2.2945, 'Eiffel Tower');
    oMap.SetCenter(48.8584, 2.2945);
    oMap.Zoom := 15;
    // the tiles are downloaded while the image is rendered
    oMap.SaveToPNG('eiffel.png', 1024, 768);
    Writeln('Missing tiles: ', oMap.LastRenderMissingTiles);
    Writeln(oMap.AttributionText);
  finally
    oMap.Free;
  end;
end.
```

Dans un formulaire, les mêmes méthodes exportent la vue que regarde l'utilisateur :

```
procedure TForm2.PreviewMap;
var
  oBitmap: TBitmap;
begin
  oBitmap := TBitmap.Create;
  try
    // the bitmap size is the size of the image; the controls are left out
    oBitmap.Width := 1200;
    oBitmap.Height := 900;
    sgcVCLMap1.RenderToBitmap(oBitmap);
    imgPreview.Picture.Assign(oBitmap);
  finally
    oBitmap.Free;
  end;
end;
```

Le rastériseur

Les tuiles vectorielles sont dessinées avec **TsgcMapRasterizer** (unité sgcMaps_Raster), un rastériseur logiciel avec anticrénelage qui ne dépend ni de la VCL ni de FireMonkey, et vous pouvez l'utiliser pour dessiner vos propres images. Il remplit et trace des chemins (BeginPath, MoveTo, LineTo, ClosePath, FillPath avec la règle non nulle ou pair-impair, StrokePath avec jonctions, extrémités et tirets), découpe selon un rectangle et enregistre le résultat avec SaveToBMPFile et SaveToPNGFile. Les couleurs sont des valeurs $AARRGGBB non prémultipliées.

```
procedure TForm2.DrawWarningIcon;
var
  oRaster: TsgcMapRasterizer;
begin
  oRaster := TsgcMapRasterizer.Create(64, 64);
  try
    oRaster.Clear($00000000);
    oRaster.BeginPath;
    oRaster.MoveTo(32, 4);
    oRaster.LineTo(60, 60);
    oRaster.LineTo(4, 60);
    oRaster.ClosePath;
    // colors are $AARRGGBB
    oRaster.FillPath($FFFFC107);
    oRaster.StrokePath($FF000000, 3, sjRound);
    oRaster.SaveToPNGFile('warning.png');
  finally
    oRaster.Free;
  end;
end;
```

Disponibilité

Framework	Version minimale
VCL, sortie BMP	Delphi 7
VCL, sortie PNG	Delphi 2009
FireMonkey	Delphi XE6
Lazarus (LCL)	Oui, PNG et BMP

Les tuiles d'un serveur requièrent Delphi 2009 ou une version ultérieure et ne sont pas disponibles sous Lazarus, où le contrôle ne dessine que les tuiles d'un dossier ; voir Tuiles, couches et fournisseurs. Le rastériseur compresse les fichiers PNG avec System.ZLib à partir de Delphi XE2, et les écrit non compressés sur les versions plus anciennes.

Voir aussi

- TsgcVCLMap
- TsgcFMXMap
- Tuiles, couches et fournisseurs

Tuiles hors ligne et téléchargeur de tuiles

Une carte qui doit fonctionner sans connexion réseau, dans un véhicule, une usine ou un navire, dessine ses tuiles à partir de fichiers locaux. sgcMaps lit deux types de packs de tuiles hors ligne :

- Un **dossier** (TileSourceType mtsFolder) : TileSourcePath contient un sous-dossier par zoom, un sous-dossier par colonne et un fichier par tuile, nommé z\x\y.png, .jpg ou .jpeg. C'est la disposition écrite par TsgcMapTileDownloader, par le cache disque d'une couche et par de nombreux outils de tuiles.
- Un fichier **MBTiles** (TileSourceType mtsMBTiles) : un seul fichier SQLite qui contient toutes les tuiles. Il est facile à copier et il est produit par des outils tels que MapTiler, TileMill et gdal. Les fichiers MBTiles vectoriels sont décrits dans Tuiles vectorielles.

```
procedure TForm2.UseOfflineTiles;
var
  oLayer: TsgcMapLayer;
begin
  sgcVCLMap1.Layers.Clear;
  // a z/x/y folder, for example the output of TsgcMapTileDownloader
  oLayer := sgcVCLMap1.Layers.Add;
  oLayer.Name := 'Tile folder';
  oLayer.TileSourceType := mtsFolder;
  oLayer.TileSourcePath := 'C:\Maps\andorra';
  oLayer.MaxZoom := 15;

  // a raster MBTiles file (Delphi XE7 and later)
  oLayer := sgcVCLMap1.Layers.Add;
  oLayer.Name := 'MBTiles';
  oLayer.TileSourceType := mtsMBTiles;
  oLayer.TileSourcePath := 'C:\Maps\andorra.mbtiles';
  oLayer.Visible := False;

  sgcVCLMap1.Options.ShowLayerSwitcher := True;
end;
```

Téléchargement d'un pack de tuiles

TsgcMapTileDownloader (unité sgcMaps_TileDownloader) remplit un dossier avec toutes les tuiles d'une zone et d'une plage de niveaux de zoom.

Membre	Description
URLTemplate	Le serveur de tuiles, avec les paramètres {z} {x} {y}. Il n'a volontairement aucune valeur par défaut.
OutputPath	Le dossier racine de l'arborescence z/x/y.
UserAgent	L'en-tête User-Agent envoyé avec chaque requête.
DelayMS	La pause entre deux téléchargements, 250 ms par défaut.
Timeout	Le délai d'expiration de chaque requête, 15000 ms par défaut.
CountTiles	Le nombre de tuiles d'une zone et d'une plage de zoom, pour vérifier la taille avant le téléchargement.
GetTileRange	La première et la dernière colonne et ligne de tuiles d'une zone à un zoom donné.
Download	Télécharge les tuiles. Les tuiles déjà présentes sur le disque sont ignorées, un téléchargement interrompu peut donc être repris.

OnProgress	Indique le total de tuiles, celles téléchargées, ignorées et en échec, ainsi que le zoom en cours. Définissez aCancel à True pour arrêter.

Download est synchrone : il bloque le thread appelant et déclenche OnProgress dans ce même thread. Appelez-le depuis un thread de travail pour garder une interface utilisateur réactive. Il lève une exception lorsque URLTemplate ou OutputPath est vide.

```
procedure TForm2.DownloadAndorra;
var
  oDownloader: TsgcMapTileDownloader;
begin
  oDownloader := TsgcMapTileDownloader.Create;
  try
    // your own tile server, or a provider whose terms allow bulk downloads
    oDownloader.URLTemplate := 'https://tiles.example.com/{z}/{x}/{y}.png';
    oDownloader.OutputPath := 'C:\Maps\andorra';
    oDownloader.UserAgent := 'MyMapApp/1.0 (contact@example.com)';
    oDownloader.DelayMS := 250;
    oDownloader.OnProgress := DownloadProgress;
    memoLog.Lines.Add(Format('%d tiles', [oDownloader.CountTiles(42.43, 1.41,
      42.66, 1.79, 10, 15)]));
    oDownloader.Download(42.43, 1.41, 42.66, 1.79, 10, 15);
  finally
    oDownloader.Free;
  end;
end;

procedure TForm2.DownloadProgress(Sender: TObject; aTotalTiles, aDownloaded,
  aSkipped, aFailed: Int64; aCurrentZoom: Integer; var aCancel: Boolean);
begin
  lblProgress.Caption := Format('Zoom %d: %d of %d tiles, %d failed',
    [aCurrentZoom, aDownloaded + aSkipped + aFailed, aTotalTiles, aFailed]);
  Application.ProcessMessages;
  aCancel := FCancelDownload;
end;
```

Respectez les politiques d'utilisation des tuiles. La plupart des serveurs de tuiles publics, tile.openstreetmap.org compris, interdisent les téléchargements en masse et le préchargement hors ligne. Faites pointer le téléchargeur vers votre propre serveur de tuiles ou vers un fournisseur dont les conditions l'autorisent, et gardez DelayMS supérieur à zéro. Le nombre de tuiles est multiplié par quatre à chaque niveau de zoom, vérifiez donc CountTiles avant de télécharger des niveaux profonds.

Disponibilité

Framework	Version minimale
Tuiles depuis un dossier	Delphi 7, et Lazarus
MBTiles	Delphi XE7 (FireDAC SQLite)
TsgcMapTileDownloader	Delphi 2009 (Indy 10 ; THTTPClient à partir de Delphi XE8)
Lazarus (LCL), MBTiles et le téléchargeur	Non disponible

Voir aussi

- Tuiles, couches et fournisseurs
- Tuiles vectorielles
- Installation

Lecteur OpenStreetMap

Les données OpenStreetMap sont distribuées sous forme de fichiers **.osm.pbf** (Protocol Buffers, compacts et rapides) et de fichiers XML **.osm**. Des extraits par pays ou par région sont publiés par Geofabrik et d'autres miroirs. **TsgcMapOSMReader** (unité sgcMaps_OSM) lit les deux formats en flux continu : il ne construit jamais le document entier en mémoire, il lit donc des fichiers de toute taille et fournit chaque élément par un événement pendant la lecture du fichier.

Membre	Description
ReadFile, ReadStream	Lit un fichier ou un flux. Le format est détecté à partir du contenu ; DetectFormat le renvoie sans lire.
ReadNodes, ReadWays, ReadRelations	Quels éléments fournir, tous à True par défaut. Désactiver ce dont vous n'avez pas besoin accélère la lecture.
SkipNodeTags	Ignore les étiquettes des nœuds, pour les lecteurs qui n'ont besoin que des coordonnées.
OnNode	Un nœud avec son identifiant, sa latitude, sa longitude et ses étiquettes.
OnWay	Un chemin avec son identifiant, les identifiants de ses nœuds et ses étiquettes.
OnRelation	Une relation avec son identifiant, ses membres (TsgcMapOSMMember : type, référence et rôle) et ses étiquettes.
OnFilterWay	Appelé avant qu'un chemin soit fourni. Définissez aAccept à False pour l'ignorer.
OnProgress	Les octets lus et la taille totale.

Les étiquettes arrivent dans un objet **TsgcMapOSMTags** avec Count, Keys, Values, Value, Has et IndexOf. L'objet d'étiquettes et les tableaux transmis aux événements sont réutilisés pour l'élément suivant, copiez donc ce dont vous avez besoin dans le gestionnaire.

```
procedure TForm2.ShowRestaurants(const aFileName: string);
var
  oReader: TsgcMapOSMReader;
begin
  FCount := 0;
  oReader := TsgcMapOSMReader.Create;
  try
    // only the nodes are needed
    oReader.ReadWays := False;
    oReader.ReadRelations := False;
    oReader.OnNode := OSMNode;
    // .osm.pbf or .osm XML, detected from the content
    oReader.ReadFile(aFileName);
  finally
    oReader.Free;
  end;
  memoLog.Lines.Add(Format('%d restaurants', [FCount]));
end;

procedure TForm2.OSMNode(Sender: TObject; aId: Int64; aLat, aLon: Double;
  aTags: TsgcMapOSMTags);
begin
  // aTags is reused for the next node: copy what you keep
  if aTags.Value('amenity') = 'restaurant' then
```

```
  begin
    Inc(FCount);
    sgcVCLMap1.AddMarker(aLat, aLon, aTags.Value('name'));
  end;
end;
```

Les blocs PBF compressés avec zlib ou stockés bruts sont pris en charge. Les blocs compressés avec lzma, lz4, zstd ou bzip2, ainsi que les fichiers qui requièrent des fonctionnalités autres que OsmSchema-V0.6 et DenseNodes, lèvent **EsgcMapOSMError**. Le lecteur est l'entrée de TsgcMapRouteGraphBuilder, voir Graphes routiers et calcul d'itinéraires hors ligne.

Disponibilité

Framework	Version minimale
VCL, FireMonkey, console et services	Delphi XE2 (System.ZLib)
Lazarus	Non disponible

Voir aussi

- Graphes routiers et calcul d'itinéraires hors ligne
- TsgcMapOfflineRouter

Index de géocodage hors ligne

Le géocodage sans connexion réseau requiert une base de données locale de noms de lieux. sgcMaps en construit une à partir des exports gratuits de **GeoNames** et l'interroge avec SQLite, en deux étapes :

1. TsgcMapGeoIndexBuilder lit un export GeoNames (cities500.zip, cities1000.zip, un fichier de pays tel que ES.zip, allCountries.zip ou le fichier .txt extrait), éventuellement avec admin1CodesASCII.txt et countryInfo.txt pour les noms de régions et de pays, et écrit une base SQLite indexée. FeatureClasses sélectionne les types de lieux à conserver, P pour les lieux habités et A pour les zones administratives.
2. TsgcMapOfflineGeocoder ouvre cette base et répond à Geocode (un nom de lieu, éventuellement suivi d'un pays), ReverseGeocode et ReverseGeocodeCountry.

L'index est construit une seule fois, sur une machine de développement ou à l'installation, et déployé avec l'application. Les résultats utilisent la même classe **TsgcMapGeocodingResults** que le service en ligne TsgcMapGeocoder, si bien que le code qui les affiche sert dans les deux cas.

Géocodage inverse sur un clic

ReverseGeocode renvoie les lieux situés à moins de 50 km d'une coordonnée, les lieux habités d'abord et chaque groupe trié par distance, et ReverseGeocodeCountry renvoie le code ISO du pays du lieu le plus proche.

```
procedure TForm2.MapClickNearestPlace(Sender: TObject; aLat, aLon: Double);
var
  oResults: TsgcMapGeocodingResults;
begin
  // DatabaseFile was built by TsgcMapGeoIndexBuilder
  if not sgcMapOfflineGeocoder1.Active then
  begin
    sgcMapOfflineGeocoder1.DatabaseFile := 'geonames.db';
    sgcMapOfflineGeocoder1.Open;
  end;
  oResults := TsgcMapGeocodingResults.Create;
  try
    if sgcMapOfflineGeocoder1.ReverseGeocode(aLat, aLon, oResults) and
      (oResults.Count > 0) then
      memoLog.Lines.Add('Nearest place: ' + oResults[0].DisplayName)
    else
      memoLog.Lines.Add('No place within 50 km');
  finally
    oResults.Free;
  end;
end;
```

Chaque composant conserve une connexion SQLite, utilisez donc un composant depuis un seul thread à la fois ; créez-en un par thread pour des requêtes en parallèle.

Disponibilité

Framework	Version minimale
VCL, FireMonkey, console et services	Delphi XE7 (FireDAC SQLite)
Lazarus	Non disponible

Voir aussi

- TsgcMapGeoIndexBuilder
- TsgcMapOfflineGeocoder
- TsgcMapGeocoder

Graphes routiers et calcul d'itinéraires hors ligne

Le calcul d'itinéraires sans connexion réseau requiert le réseau routier sur disque sous une forme qu'un calculateur peut parcourir rapidement. sgcMaps procède en deux étapes :

1. **TsgcMapRouteGraphBuilder** (unité sgcMaps_OfflineRouter) lit les données OpenStreetMap, un fichier .osm.pbf ou .osm, et écrit un **graphe routier** binaire compact, un fichier .sgcgraph avec les profils voiture, vélo et piéton.
2. TsgcMapOfflineRouter charge le graphe, aligne chaque point de passage sur la route la plus proche du profil sélectionné et trouve l'itinéraire avec une recherche A* bidirectionnelle.

Construction du graphe

Build(fichier d'entrée, fichier de sortie) s'exécute en trois phases, indiquées par **Phase** : 1 lit les chemins, 2 lit les coordonnées des nœuds que ces chemins utilisent, et 3 écrit le graphe. **OnProgress** indique les octets lus et la taille totale de l'entrée pendant sa lecture. Lorsque la construction se termine, WayCount, NodeCount, MissingNodes, VertexCount, EdgeCount, ShapeCount et NameCount décrivent le graphe. Le graphe stocke pour chaque route sa longueur, sa vitesse pour les voitures, les sens dans lesquels elle peut être parcourue pour chaque profil, ses points intermédiaires et son nom.

```
procedure TForm2.BuildGraph;
var
  oBuilder: TsgcMapRouteGraphBuilder;
begin
  oBuilder := TsgcMapRouteGraphBuilder.Create;
  try
    oBuilder.OnProgress := GraphProgress;
    oBuilder.Build('andorra-latest.osm.pbf', 'andorra.sgcgraph');
    memoLog.Lines.Add(Format('%d ways, %d vertices, %d edges',
      [oBuilder.WayCount, oBuilder.VertexCount, oBuilder.EdgeCount]));
  finally
    oBuilder.Free;
  end;
end;

procedure TForm2.GraphProgress(Sender: TObject; aBytesRead, aTotal: Int64);
begin
  if aTotal > 0 then
    lblProgress.Caption := Format('Phase %d of 3: %d%%',
      [TsgcMapRouteGraphBuilder(Sender).Phase, aBytesRead * 100 div aTotal]);
  Application.ProcessMessages;
end;
```

Construisez le graphe une seule fois, pour la région que couvre l'application, et déployez le fichier .sgcgraph. L'outil console RouteGraphBuilder fait la même chose en ligne de commande.

Calcul d'itinéraires

Renseignez GraphFile et Profile (morCar, morBike ou morFoot) et appelez Open. **GetRoute** prend au moins deux points de passage et renseigne un **TsgcMapRoute**, la classe utilisée par le service en ligne TsgcMapDirections, avec la géométrie, la distance, la durée, un tronçon par paire de points de passage et les étapes de guidage. **sgcMapAddRouteToPolylines** le dessine sur une carte.

```
procedure TForm2.RouteByBike;
var
  vWaypoints: array [0 .. 2] of TsgcMapCoord;
  oRoute: TsgcMapRoute;
begin
  sgcMapOfflineRouter1.GraphFile := 'andorra.sgcgraph';
```

```
  sgcMapOfflineRouter1.Profile := morBike;
  sgcMapOfflineRouter1.Open;

  vWaypoints[0] := sgcMapCoord(42.5063, 1.5218); // Andorra la Vella
  vWaypoints[1] := sgcMapCoord(42.5448, 1.5146); // La Massana
  vWaypoints[2] := sgcMapCoord(42.5347, 1.5800); // Encamp
  oRoute := TsgcMapRoute.Create;
  try
    if sgcMapOfflineRouter1.GetRoute(vWaypoints, oRoute) then
    begin
      sgcMapAddRouteToPolylines(oRoute, sgcVCLMap1.Polylines);
      memoLog.Lines.Add(Format('%.1f km in %d legs',
        [oRoute.Distance / 1000, oRoute.LegCount]));
    end
    else
      memoLog.Lines.Add(sgcMapOfflineRouter1.LastError);
  finally
    oRoute.Free;
  end;
end;
```

Le calculateur hors ligne peut aussi être le moteur de TsgcMapRouteCalculator, par son événement OnCalculate-Route, ce qui offre à l'utilisateur des points de passage déplaçables sans connexion réseau.

Disponibilité

Framework	Version minimale
VCL, FireMonkey, console et services	Delphi XE2
Lazarus	Non disponible

Voir aussi

- Lecteur OpenStreetMap
- TsgcMapOfflineRouter
- TsgcMapRouteCalculator

TsgcMapService

Unité : sgcMaps_Services
La classe de base de tous les services de cartes en ligne : TsgcMapGeocoder, TsgcMapDirections, TsgcMapPlaces, TsgcMapElevation et TsgcMapTimeZone. Elle n'est pas enregistrée dans la palette. Les services envoient des requêtes HTTPS avec THTTPClient et analysent les réponses avec System.JSON, ils requièrent donc Delphi XE8 ou une version ultérieure et ne sont pas disponibles sous Lazarus.

Appels synchrones et asynchrones

- Les méthodes **synchrones** (Geocode, GetRoute, Suggest...) bloquent le thread appelant jusqu'à l'arrivée de la réponse. Elles ne lèvent jamais d'exception : en cas d'échec, elles renvoient False et renseignent **LastError**.
- Les méthodes **asynchrones** (GeocodeAsync, GetRouteAsync, SuggestAsync...) exécutent la requête dans un thread d'arrière-plan et déclenchent l'événement de résultat, ou **OnError**, dans le thread principal par TThread.Queue. Les applications VCL et FireMonkey n'ont besoin de rien d'autre. Une application console ou un service doit appeler CheckSynchronize périodiquement, ou utiliser les méthodes synchrones.
- Le composant doit rester vivant jusqu'à ce qu'une requête asynchrone ait déclenché son événement. Les objets de résultat transmis aux événements appartiennent au composant et sont libérés au retour de l'événement, copiez donc ce que vous voulez conserver.

Serveurs publics et politiques d'utilisation

Les fournisseurs par défaut sont des services publics gratuits soumis à des politiques d'utilisation. Nominatim, le serveur de démonstration d'OSRM et l'API publique d'OpenTopoData autorisent au plus une requête par seconde, et les composants imposent cet intervalle. Ils attendent aussi un **UserAgent** qui identifie l'application. Utilisez votre propre serveur ou un fournisseur commercial pour la production et pour les requêtes en masse.

Propriétés

Propriété	Type	Par défaut	Description
ApiKey	string		Clé des fournisseurs qui en exigent une. Pour le fournisseur de fuseau horaire GeoNames, c'est le nom d'utilisateur GeoNames.
Language	string	en	Langue des résultats, pour les fournisseurs qui la prennent en charge.
Timeout	Integer	15000	Délai d'expiration de la requête en millisecondes.
UserAgent	string	sgcMaps/2.0	En-tête User-Agent de chaque requête. Identifiez ici votre application.
Referer	string		En-tête Referer, envoyé lorsqu'il n'est pas vide.
LastError	string		Erreur de la dernière requête en échec. Lecture seule.

Événements

Événement	Description

OnError	Une requête asynchrone a échoué. Déclenché dans le thread principal avec le texte de l'erreur.

Exemple : configuration commune des services

Une seule procédure configure tous les services d'un formulaire, puisque les propriétés sont déclarées dans la classe de base.

```
procedure TForm1.SetupService(aService: TsgcMapService);
begin
  // public servers such as Nominatim require an identifying User-Agent
  aService.UserAgent := 'MyMapApp/1.0 (contact@example.com)';
  aService.Language := 'en';
  aService.Timeout := 10000;
  aService.OnError := ServiceError;
end;

procedure TForm1.ServiceError(Sender: TObject; const aError: string);
begin
  memoLog.Lines.Add(Sender.ClassName + ': ' + aError);
end;
```

Voir aussi

- TsgcMapGeocoder
- TsgcMapDirections
- TsgcMapPlaces
- TsgcMapElevation
- TsgcMapTimeZone

TsgcMapGeocoder

Unité : sgcMaps_Geocoding
Transforme une adresse ou un nom de lieu en coordonnées (géocodage), et des coordonnées en adresse (géocodage inverse).
Le composant descend de TsgcMapService et hérite de **ApiKey**, **Language**, **Timeout**, **UserAgent**, **Referer**, **LastError** et **OnError**. Il requiert Delphi XE8 ou une version ultérieure. Les méthodes synchrones ne lèvent jamais d'exception : elles renvoient False et renseignent LastError. Les méthodes asynchrones déclenchent leurs événements dans le thread principal.

Fournisseurs

Fournisseur	Service	Clé
mgpNominatim	Nominatim (OpenStreetMap). Le fournisseur par défaut. NominatimURL pointe vers une instance auto-hébergée.	Non
mgpGoogle	Google Geocoding API	Oui
mgpHere	HERE Geocoding and Search	Oui
mgpAzure	Azure Maps Search	Oui
mgpMapBox	Mapbox Geocoding	Oui
mgpOpenRouteService	OpenRouteService Geocode	Oui
mgpGeoApify	Geoapify Geocoding	Oui

Propriétés

Propriété	Type	Par défaut	Description
Provider	TsgcMapGeocodingProvider	mgpNominatim	Le service de géocodage.
NominatimURL	string	https://nominatim.openstreetmap.org	URL de base du serveur Nominatim.
MaxResults	Integer	5	Nombre maximal de résultats.
CountryCodes	string		Filtre de pays séparé par des virgules, avec des codes ISO 3166-1 alpha-2 ('es,fr'). HERE attend des codes alpha-3 ('ESP,FRA') et Azure l'ignore.

Méthodes

Méthode	Description

Geocode(const aQuery: string; aResults: TsgcMapGeocodingResults): Boolean	Recherche une adresse ou un lieu. True lorsqu'au moins un résultat a été trouvé.
ReverseGeocode(aLat, aLon: Double; aResults: TsgcMapGeocodingResults): Boolean	Trouve l'adresse située à une coordonnée.
GeocodeAsync(const aQuery: string)	Version asynchrone de Geocode. Les résultats arrivent dans OnGeocodeResult.
ReverseGeocodeAsync(aLat, aLon: Double)	Version asynchrone de ReverseGeocode. Les résultats arrivent dans OnGeocodeResult.
BuildURL, BuildReverseURL	L'URL de la requête pour le fournisseur actuel, utile pour la journalisation et les tests.

Événements

Événement	Description
OnGeocodeResult	Les résultats d'une requête asynchrone, dans le thread principal. La liste est libérée au retour de l'événement.

Résultats

Chaque résultat est un TsgcMapGeocodingResult avec **Latitude**, **Longitude**, **DisplayName**, **Street**, **HouseNumber**, **PostalCode**, **City**, **State**, **Country**, **CountryCode** (ISO alpha-2 lorsque le fournisseur le renvoie, alpha-3 pour HERE), **Confidence** (de 0 à 1, 0 lorsqu'elle est inconnue) et le rectangle englobant **BoundsNorth**, **BoundsSouth**, **BoundsEast** et **BoundsWest** (voir HasBounds). Les mêmes classes de résultat sont utilisées par TsgcMapOfflineGeocoder.

Exemple : géocodage et géocodage inverse

```
procedure TForm1.btnGeocodeClick(Sender: TObject);
var
  oResults: TsgcMapGeocodingResults;
begin
  sgcMapGeocoder1.Provider := mgpNominatim;
  sgcMapGeocoder1.MaxResults := 3;
  oResults := TsgcMapGeocodingResults.Create;
  try
    if sgcMapGeocoder1.Geocode('Plaza Mayor, Madrid', oResults) then
    begin
      sgcVCLMap1.SetCenter(oResults[0].Latitude, oResults[0].Longitude);
      sgcVCLMap1.AddMarker(oResults[0].Latitude, oResults[0].Longitude,
        oResults[0].DisplayName);
    end
    else
      memoLog.Lines.Add('Geocode failed: ' + sgcMapGeocoder1.LastError);
  finally
    oResults.Free;
  end;
end;

procedure TForm1.btnReverseClick(Sender: TObject);
begin
  sgcMapGeocoder1.OnGeocodeResult := sgcMapGeocoder1GeocodeResult;
  sgcMapGeocoder1.ReverseGeocodeAsync(sgcVCLMap1.CenterLatitude,
    sgcVCLMap1.CenterLongitude);
end;
```

```
procedure TForm1.sgcMapGeocoder1GeocodeResult(Sender: TObject;
  aResults: TsgcMapGeocodingResults);
begin
  // aResults is owned by the component: copy what you need to keep
  if aResults.Count > 0 then
    memoLog.Lines.Add(aResults[0].DisplayName);
end;
```

Voir aussi

- TsgcMapService
- TsgcMapPlaces
- TsgcMapOfflineGeocoder

TsgcMapDirections

Unité : sgcMaps_Directions
Calcule un itinéraire passant par au moins deux points de passage avec un service de calcul d'itinéraires en ligne. Les distances sont en mètres et les durées en secondes.
Le composant descend de TsgcMapService et hérite de **ApiKey**, **Language**, **Timeout**, **UserAgent**, **Referer**, **LastError** et **OnError**. Il requiert Delphi XE8 ou une version ultérieure. Les méthodes synchrones ne lèvent jamais d'exception : elles renvoient False et renseignent LastError. Les méthodes asynchrones déclenchent leurs événements dans le thread principal.

Fournisseurs

Fournisseur	Service	Clé
mdpOSRM	OSRM. L'OSRMURL par défaut est le serveur de démonstration public du projet OSRM, qui n'autorise qu'un usage léger et ne fournit que le profil voiture. Utilisez votre propre osrm-routed pour la production, ainsi que pour la marche et le vélo.	Non
mdpOpenRouteService	OpenRouteService Directions	Oui
mdpGoogleRoutes	Google Routes API	Oui
mdpHere	HERE Routing	Oui
mdpAzure	Azure Maps Route	Oui
mdpMapBox	Mapbox Directions	Oui
mdpGeoApify	Geoapify Routing	Oui

Chaque fournisseur renvoie les instructions détaillées virage par virage dans la **Language** choisie, sauf OSRM, qui n'en renvoie aucune : pour OSRM, le composant construit de courtes instructions en anglais.

Propriétés

Propriété	Type	Par défaut	Description
Provider	TsgcMapDirectionsProvider	mdpOSRM	Le service de calcul d'itinéraires.
OSRMURL	string	https://router.project-osrm.org	URL de base du serveur OSRM.
TravelMode	TsgcMapTravelMode	mtmDriving	mtmDriving, mtmWalking, mtmCycling ou mtmTruck. IsTravelModeSupported indique quels modes un fournisseur accepte.
Alternatives	Boolean	False	Demande des itinéraires alternatifs.

AvoidTolls, AvoidHighways, AvoidFerries	Boolean	False	Restrictions d'itinéraire, pour les fournisseurs qui les prennent en charge.
OptimizeWaypoints	Boolean	False	Laisse le fournisseur réordonner les points de passage intermédiaires.

Méthodes

Méthode	Description
GetRoute(const aWaypoints: array of TsgcMapCoord; aRoutes: TsgcMapRoutes): Boolean	Calcule l'itinéraire. True lorsqu'au moins un itinéraire a été trouvé ; l'indice 0 est l'itinéraire principal.
GetRouteAsync(const aWaypoints: array of TsgcMapCoord)	Version asynchrone de GetRoute. Les itinéraires arrivent dans OnRouteResult.
ValidateRequest(const aWaypoints: array of TsgcMapCoord): string	Une chaîne vide lorsque la requête peut être envoyée, sinon la raison pour laquelle elle ne le peut pas.
IsTravelModeSupported(aProvider; aMode): Boolean	Fonction de classe : indique si un fournisseur prend en charge un mode de déplacement.

Événements

Événement	Description
OnRouteResult	Les itinéraires d'une requête asynchrone, dans le thread principal. La liste est libérée au retour de l'événement.

Résultats d'itinéraire

Classe	Membres
TsgcMapRoutes	Count et Items[Index] (propriété par défaut) de TsgcMapRoute.
TsgcMapRoute	Distance, Duration, Summary, la géométrie (Points, PointCount, GetPoints), les tronçons (Legs, LegCount) et le rectangle englobant (BoundsNorth, BoundsSouth, BoundsEast, BoundsWest, CalcBounds).
TsgcMapRouteLeg	La partie de l'itinéraire entre deux points de passage consécutifs : Distance, Duration, Summary, Steps et StepCount.
TsgcMapRouteStep	Une manœuvre : Instruction, Maneuver, StreetName, Distance, Duration, Latitude, Longitude et PointIndex (sa position dans la géométrie de l'itinéraire).

sgcMapAddRouteToPolylines(aRoute, aPolylines, aColor, aWidth) ajoute à une carte une polyligne ayant la géométrie d'un itinéraire et la renvoie.

Exemple : calcul d'un itinéraire

```
procedure TForm1.btnRouteClick(Sender: TObject);
var
  vWaypoints: array [0 .. 1] of TsgcMapCoord;
  oRoutes: TsgcMapRoutes;
  oLine: TsgcMapPolyline;
  i, j: Integer;
begin
  vWaypoints[0].Latitude := 40.4168;
  vWaypoints[0].Longitude := -3.7038;
  vWaypoints[1].Latitude := 40.4530;
  vWaypoints[1].Longitude := -3.6883;

  sgcMapDirections1.Provider := mdpOSRM;
  sgcMapDirections1.TravelMode := mtmDriving;
  oRoutes := TsgcMapRoutes.Create;
  try
    if sgcMapDirections1.GetRoute(vWaypoints, oRoutes) then
    begin
      oLine := sgcMapAddRouteToPolylines(oRoutes[0], sgcVCLMap1.Polylines);
      sgcVCLMap1.FitToPolyline(oLine);
      memoLog.Lines.Add(Format('%.1f km, %.0f min',
        [oRoutes[0].Distance / 1000, oRoutes[0].Duration / 60]));
      for i := 0 to oRoutes[0].LegCount - 1 do
        for j := 0 to oRoutes[0].Legs[i].StepCount - 1 do
          memoLog.Lines.Add(oRoutes[0].Legs[i].Steps[j].Instruction);
    end
    else
      memoLog.Lines.Add('Route failed: ' + sgcMapDirections1.LastError);
  finally
    oRoutes.Free;
  end;
end;
```

Voir aussi

- TsgcMapService
- TsgcMapRouteCalculator
- TsgcMapOfflineRouter

TsgcMapPlaces

Unité : sgcMaps_Places
Saisie semi-automatique de lieux : des suggestions pour le texte que l'utilisateur est en train de saisir, généralement affichées dans une liste sous un champ de recherche.
Le composant descend de TsgcMapService et hérite de **ApiKey**, **Language**, **Timeout**, **UserAgent**, **Referer**, **LastError** et **OnError**. Il requiert Delphi XE8 ou une version ultérieure. Les méthodes synchrones ne lèvent jamais d'exception : elles renvoient False et renseignent LastError. Les méthodes asynchrones déclenchent leurs événements dans le thread principal.

Fournisseurs

Fournisseur	Service	Clé
mplPhoton	Photon (komoot). Le fournisseur par défaut. Il n'accepte que les langues de, en et fr ; toute autre Language n'est pas envoyée.	Non
mplNominatim	Nominatim. Limité à une requête par seconde, n'utilisez donc pas le serveur public pour une saisie semi-automatique à chaque frappe.	Non
mplGoogle	Google Places API (New). Les suggestions n'ont pas de coordonnées : appelez Retrieve.	Oui
mplHere	HERE Autosuggest	Oui
mplMapBox	Mapbox Search Box API. Les suggestions n'ont pas de coordonnées : appelez Retrieve.	Oui
mplGeoApify	Geoapify Autocomplete	Oui

Propriétés

Propriété	Type	Par défaut	Description
Provider	TsgcMapPlacesProvider	mplPhoton	Le service de saisie semi-automatique.
NominatimURL	string	https://nominatim.openstreetmap.org	URL de base du serveur Nominatim.
MaxResults	Integer	5	Nombre maximal de suggestions.
BiasLatitude, BiasLongitude	Double	0	Les résultats proches de cette position sont favorisés et leur Distance est renseignée. Ignorées lorsque les deux valent 0.
BiasRadius	Double	0	Rayon autour de la position de biais, en mètres. 0 signifie aucun rayon. Utilisé par Google, HERE et Nominatim.
SessionToken	string		Jeton de la session de recherche en cours. Publique.

Méthodes

Méthode	Description
Suggest(const aText: string; aResults: TsgcMapPlaceResults): Boolean	Renvoie les suggestions pour un texte. True lorsqu'au moins une a été trouvée.
SuggestAsync(const aText: string)	Version asynchrone de Suggest. Les suggestions arrivent dans OnSuggestResult.
Retrieve(aPlace: TsgcMapPlace): Boolean	Renseigne les coordonnées, ainsi qu'un nom ou une adresse manquants, d'une suggestion. Seuls Google et Mapbox envoient une requête ; pour les autres fournisseurs, elle renvoie HasCoordinates.
NewSession	Démarre une nouvelle session de recherche avec un nouveau SessionToken.

Événements

Événement	Description
OnSuggestResult	Les suggestions d'une requête asynchrone, dans le thread principal. La liste est libérée au retour de l'événement.

Résultats

Chaque suggestion est un TsgcMapPlace avec **Id** (l'identifiant du fournisseur, utilisé par Retrieve), **Name**, **Address**, **Latitude**, **Longitude**, **HasCoordinates**, **Category** et **Distance** (en mètres depuis la position de biais, -1 lorsqu'elle est inconnue).

Exemple : suggestions pendant la saisie

```
procedure TForm1.edtSearchChange(Sender: TObject);
begin
  if Length(edtSearch.Text) < 3 then
    Exit;
  // prefer the places near the center of the map
  sgcMapPlaces1.BiasLatitude := sgcVCLMap1.CenterLatitude;
  sgcMapPlaces1.BiasLongitude := sgcVCLMap1.CenterLongitude;
  sgcMapPlaces1.SuggestAsync(edtSearch.Text);
end;

procedure TForm1.sgcMapPlaces1SuggestResult(Sender: TObject;
  aResults: TsgcMapPlaceResults);
var
  i: Integer;
begin
  lbPlaces.Items.Clear;
  for i := 0 to aResults.Count - 1 do
    lbPlaces.Items.Add(aResults[i].Name + ', ' + aResults[i].Address);
end;
```

Voir aussi

- TsgcMapService

- TsgcMapGeocoder

TsgcMapElevation

Unité : sgcMaps_Elevation
Renvoie l'altitude du terrain pour une liste de points, ou pour des échantillons régulièrement espacés le long d'un tracé, par exemple un profil altimétrique d'un itinéraire. Les longues listes sont découpées automatiquement en lots.
Le composant descend de TsgcMapService et hérite de **ApiKey**, **Language**, **Timeout**, **UserAgent**, **Referer**, **LastError** et **OnError**. Il requiert Delphi XE8 ou une version ultérieure. Les méthodes synchrones ne lèvent jamais d'exception : elles renvoient False et renseignent LastError. Les méthodes asynchrones déclenchent leurs événements dans le thread principal.

Fournisseurs

Fournisseur	Service	Points par requête	Clé
meOpenElevation	Open-Elevation	200	Non
meOpenTopoData	OpenTopoData. Le fournisseur par défaut. L'API publique est limitée à une requête par seconde.	100	Non
meGoogle	Google Elevation API	256	Oui

Propriétés

Propriété	Type	Par défaut	Description
Provider	TsgcMapElevationProvider	meOpenTopoData	Le service d'altitude.
Dataset	string	srtm90m	Jeu de données OpenTopoData : srtm90m, srtm30m, aster30m, eudem25m et d'autres.

Méthodes

Méthode	Description
GetElevations(const aPoints: array of TsgcMapCoord; aResults: TsgcMapElevationResults): Boolean	Un résultat par point, dans le même ordre.
GetElevationAlongPath(const aPoints: array of TsgcMapCoord; aSamples: Integer; aResults: TsgcMapElevationResults): Boolean	aSamples points (au moins 2, premier et dernier inclus) régulièrement espacés en distance le long de la polyligne aPoints.
GetElevationsAsync, GetElevationAlongPathAsync	Versions asynchrones. Les résultats arrivent dans OnElevationResult.
BatchSize: Integer	Nombre maximal de points par requête du fournisseur actuel.
SamplePath	Fonction de classe : les échantillons régulièrement espacés d'une polyligne et leurs distances depuis le premier point.

Événements

Événement	Description
OnElevationResult	Les résultats d'une requête asynchrone, dans le thread principal. La liste est libérée au retour de l'événement.

Résultats

Chaque résultat est un TsgcMapElevationResult avec **Latitude**, **Longitude**, **Elevation** (en mètres au-dessus du niveau de la mer), **HasElevation** (False lorsque le fournisseur n'a aucune donnée pour le point) et **Distance** (en mètres depuis le premier point, renseignée par GetElevationAlongPath).

Exemple : profil altimétrique

```
procedure TForm1.btnElevationClick(Sender: TObject);
var
  vPath: array [0 .. 1] of TsgcMapCoord;
  oResults: TsgcMapElevationResults;
  i: Integer;
begin
  vPath[0].Latitude := 40.4168; // Madrid
  vPath[0].Longitude := -3.7038;
  vPath[1].Latitude := 40.7934; // Navacerrada pass
  vPath[1].Longitude := -4.0102;

  sgcMapElevation1.Provider := meOpenTopoData;
  sgcMapElevation1.Dataset := 'srtm90m';
  oResults := TsgcMapElevationResults.Create;
  try
    if not sgcMapElevation1.GetElevationAlongPath(vPath, 20, oResults) then
      memoLog.Lines.Add('Elevation failed: ' + sgcMapElevation1.LastError)
    else
      for i := 0 to oResults.Count - 1 do
        if oResults[i].HasElevation then
          memoLog.Lines.Add(Format('%.0f m along the path: %.0f m',
            [oResults[i].Distance, oResults[i].Elevation]));
  finally
    oResults.Free;
  end;
end;
```

Voir aussi

- TsgcMapService
- TsgcMapDirections

TsgcMapTimeZone

Unité : sgcMaps_TimeZone
Renvoie le fuseau horaire d'une coordonnée : son identifiant IANA, le décalage par rapport à UTC et l'heure locale. Le composant descend de TsgcMapService et hérite de **ApiKey**, **Language**, **Timeout**, **UserAgent**, **Referer**, **LastError** et **OnError**. Il requiert Delphi XE8 ou une version ultérieure. Les méthodes synchrones ne lèvent jamais d'exception : elles renvoient False et renseignent LastError. Les méthodes asynchrones déclenchent leurs événements dans le thread principal.

Fournisseurs

Fournisseur	Service	Clé
mtzTimeAPI	TimeAPI (timeapi.io). Le fournisseur par défaut.	Non
mtzGeoNames	GeoNames. ApiKey est le nom d'utilisateur GeoNames.	Nom d'utilisateur
mtzGoogle	Google Time Zone API	Oui
mtzAzure	Azure Maps Timezone	Oui

Le paramètre facultatif **aUtcTime** (0 signifie maintenant) est pris en compte par Google et Azure. TimeAPI et GeoNames répondent toujours pour l'heure actuelle.

Propriétés

Propriété	Type	Par défaut	Description
Provider	TsgcMapTimeZoneProvider	mtzTimeAPI	Le service de fuseau horaire.

Méthodes

Méthode	Description
GetTimeZone(aLat, aLon: Double; aInfo: TsgcMapTimeZoneInfo; aUtcTime: TDateTime = 0): Boolean	Renseigne aInfo avec le fuseau horaire de la coordonnée.
GetTimeZoneAsync(aLat, aLon: Double; aUtcTime: TDateTime = 0)	Version asynchrone de GetTimeZone. La réponse arrive dans OnTimeZoneResult.
UtcNow: TDateTime	Fonction de classe : l'heure UTC actuelle.

Événements

Événement	Description
OnTimeZoneResult	La réponse d'une requête asynchrone, dans le thread principal. L'objet est libéré au retour de l'événement.

Résultats

TsgcMapTimeZoneInfo contient **TimeZoneId** ('Europe/Madrid'), **Name** (le nom d'affichage lorsque le fournisseur en renvoie un, sinon le TimeZoneId), **UtcOffsetSeconds** (heure standard plus heure d'été), **DstOffsetSeconds** (la part de l'heure d'été), **IsDst** et **LocalTime** (ISO 8601 sans décalage).

Exemple : fuseau horaire d'une coordonnée

```
procedure TForm1.btnTimeZoneClick(Sender: TObject);
var
  oInfo: TsgcMapTimeZoneInfo;
begin
  sgcMapTimeZone1.Provider := mtzTimeAPI;
  oInfo := TsgcMapTimeZoneInfo.Create;
  try
    if sgcMapTimeZone1.GetTimeZone(35.6762, 139.6503, oInfo) then
      memoLog.Lines.Add(Format('%s, UTC offset %d s, local time %s',
        [oInfo.TimeZoneId, oInfo.UtcOffsetSeconds, oInfo.LocalTime]))
    else
      memoLog.Lines.Add('Time zone failed: ' + sgcMapTimeZone1.LastError);
  finally
    oInfo.Free;
  end;
end;
```

Voir aussi

- TsgcMapService

TsgcMapRouteCalculator

Unité : sgcMaps_RouteCalculator
Modification interactive d'un itinéraire sur une carte. Les points de passage sont des marqueurs déplaçables dans **Markers** et l'itinéraire est une polyligne dans **Polylines** ; faites pointer les deux vers les collections d'un contrôle de carte. Déplacer un point de passage, en ajouter un ou en supprimer un recalcule l'itinéraire lorsque **AutoCalculate** vaut True. Requiert Delphi XE2 ou une version ultérieure ; la propriété Directions requiert Delphi XE8.
Le moteur de calcul est l'événement **OnCalculateRoute**, où n'importe quel moteur peut renseigner l'itinéraire (par exemple TsgcMapOfflineRouter) ou, lorsque l'événement ne le traite pas, le service en ligne TsgcMapDirections affecté à **Directions**.
Les marqueurs de points de passage reçoivent Tag = MarkerTagBase + indice et la polyligne de l'itinéraire reçoit Tag = MarkerTagBase. Avec ShowAlternatives, les itinéraires alternatifs sont dessinés sous forme de polylignes grises plus fines avec Tag = MarkerTagBase + 1.

Propriétés

Propriété	Type	Par défaut	Description
Markers	TsgcMap-Markers		Collection des marqueurs de points de passage, par exemple Map.Markers.
Polylines	TsgcMap-Polylines		Collection des polylignes d'itinéraire, par exemple Map.Polylines.
Directions	TsgcMap-Directions		Moteur de calcul en ligne, utilisé lorsque OnCalculate-Route ne traite pas la requête.
Active	Boolean	True	False détache le calculateur de Markers et de Polylines : plus de suivi des déplacements de marqueurs et plus de calcul automatique.
AutoCalculate	Boolean	True	Recalcule après chaque modification des points de passage.
ShowAlternatives	Boolean	False	Dessine aussi les itinéraires alternatifs.
RouteColor, Route-Width	TColor, Integer	bleu, 4	Couleur et largeur de la polyligne de l'itinéraire.
WaypointColor	TColor	rouge	Couleur des marqueurs de points de passage.
MarkerTagBase	Integer	SGC_MAP_ROUTE_TAG_BASE	Premier Tag utilisé pour les marqueurs et les polylignes du calculateur.
Waypoints[Index], WaypointMarkers[Index]	TsgcMap-Coord, TsgcMap-Marker		Les points de passage et leurs marqueurs. Lecture seule.

Route, RoutePolyline, Routes	TsgcMapRoute, TsgcMapPolyline, TsgcMapRoutes		L'itinéraire principal, sa polyligne et tous les itinéraires du dernier calcul (l'indice 0 est l'itinéraire principal). Lecture seule.

Méthodes

Méthode	Description
AddWaypoint(aLat, aLon: Double): Integer	Ajoute un point de passage à la fin et renvoie son indice.
InsertWaypoint(aIndex: Integer; aLat, aLon: Double)	Insère un point de passage.
AddWaypointNearestSegment(aLat, aLon: Double): Integer	Insère un point de passage entre les deux points de passage dont le tronçon est le plus proche du point, comme le fait le glissement de la ligne de l'itinéraire.
MoveWaypoint(aIndex: Integer; aLat, aLon: Double)	Déplace un point de passage.
RemoveWaypoint(aIndex: Integer), ClearWaypoints	Supprime un point de passage ou tous.
WaypointCount: Integer	Nombre de points de passage.
Calculate: Boolean	Calcul synchrone : OnCalculateRoute, puis Directions.GetRoute.
CalculateAsync	OnCalculateRoute, appliqué aussitôt lorsqu'il est traité, sinon Directions.GetRouteAsync. Les événements sont déclenchés dans le thread principal.
SelectRoute(aIndex: Integer)	Fait d'un itinéraire alternatif l'itinéraire principal, le redessine et déclenche OnRouteCalculated.

Événements

Événement	Description
OnCalculateRoute	Déclenché en premier à chaque calcul. Renseignez aRoute et définissez aHandled à True pour utiliser votre propre moteur.
OnRouteCalculated	Un nouvel itinéraire a été calculé et dessiné.
OnWaypointsChanged	Les points de passage ont changé.
OnError	Le calcul a échoué.

Exemple : construire un itinéraire en cliquant sur la carte

```
procedure TForm1.SetupRouteCalculator;
begin
  // the calculator draws on the collections of the map
  sgcMapRouteCalculator1.Markers := sgcVCLMap1.Markers;
  sgcMapRouteCalculator1.Polylines := sgcVCLMap1.Polylines;
  sgcMapRouteCalculator1.Directions := sgcMapDirections1;
  sgcMapRouteCalculator1.AutoCalculate := True;
  sgcMapRouteCalculator1.OnRouteCalculated :=
    sgcMapRouteCalculator1RouteCalculated;
  sgcVCLMap1.OnMapClick := MapClickAddWaypoint;
end;

procedure TForm1.MapClickAddWaypoint(Sender: TObject; aLat, aLon: Double);
begin
  sgcMapRouteCalculator1.AddWaypoint(aLat, aLon);
end;

procedure TForm1.sgcMapRouteCalculator1RouteCalculated(Sender: TObject;
  aRoute: TsgcMapRoute);
begin
  memoLog.Lines.Add(Format('Route: %.1f km, %.0f min',
    [aRoute.Distance / 1000, aRoute.Duration / 60]));
end;
```

Pour calculer l'itinéraire sans connexion réseau, traitez OnCalculateRoute avec le calculateur hors ligne, comme le montre la rubrique TsgcMapOfflineRouter.

Voir aussi

- TsgcMapDirections
- TsgcMapOfflineRouter
- TsgcVCLMap

TsgcMapOfflineRouter

Unité : sgcMaps_OfflineRouter
Calcule des itinéraires sans connexion réseau, à partir d'un fichier de graphe routier (**.sgcgraph**) construit à partir des données OpenStreetMap. Requiert Delphi XE2 ou une version ultérieure.
Construisez le graphe une fois à partir d'un extrait OpenStreetMap (.osm.pbf ou .osm XML) avec **TsgcMapRouteGraphBuilder**, déclaré dans la même unité, ou avec l'outil RouteGraphBuilder de Tools\Maps. Le calculateur renseigne la même classe TsgcMapRoute que TsgcMapDirections, si bien qu'un itinéraire issu de l'un ou l'autre moteur se dessine et s'inspecte de la même façon. Un itinéraire passant par plusieurs points de passage compte un tronçon par paire de points de passage consécutifs.

Propriétés

Propriété	Type	Par défaut	Description
GraphFile	string		Le fichier .sgcgraph.
Profile	TsgcMapOfflineRouterProfile	morCar	morCar, morBike ou morFoot.
Active	Boolean	False	Ouvre ou ferme le graphe.
LastError	string		Erreur du dernier appel en échec. Lecture seule.
VertexCount, EdgeCount	Integer		Taille du graphe chargé. Lecture seule.

Méthodes

Méthode	Description
Open, Close	Charge ou libère le graphe.
GetRoute(const aWaypoints: array of TsgcMapCoord; aRoute: TsgcMapRoute): Boolean	Calcule l'itinéraire. False lorsqu'aucun itinéraire n'a été trouvé, avec la raison dans LastError.

Exemple : itinéraire hors ligne et moteur hors ligne du calculateur d'itinéraires

```
procedure TForm1.btnOfflineRouteClick(Sender: TObject);
var
  vWaypoints: array [0 .. 1] of TsgcMapCoord;
  oRoute: TsgcMapRoute;
begin
  sgcMapOfflineRouter1.GraphFile := 'andorra.sgcgraph';
  sgcMapOfflineRouter1.Profile := morCar;
  sgcMapOfflineRouter1.Open;

  vWaypoints[0].Latitude := 42.5063; // Andorra la Vella
  vWaypoints[0].Longitude := 1.5218;
  vWaypoints[1].Latitude := 42.5347; // Encamp
  vWaypoints[1].Longitude := 1.5800;
  oRoute := TsgcMapRoute.Create;
  try
```

```
    if sgcMapOfflineRouter1.GetRoute(vWaypoints, oRoute) then
      sgcMapAddRouteToPolylines(oRoute, sgcVCLMap1.Polylines)
    else
      memoLog.Lines.Add('No route: ' + sgcMapOfflineRouter1.LastError);
  finally
    oRoute.Free;
  end;
end;

procedure TForm1.sgcMapRouteCalculator1CalculateRoute(Sender: TObject;
  const aWaypoints: array of TsgcMapCoord; aRoute: TsgcMapRoute;
  var aHandled: Boolean);
begin
  // the offline router as the engine of the interactive route calculator
  aHandled := sgcMapOfflineRouter1.GetRoute(aWaypoints, aRoute);
end;
```

Voir aussi

- TsgcMapRouteCalculator
- TsgcMapDirections

TsgcMapOfflineGeocoder

Unité : sgcMaps_OfflineGeocoder
Géocodage et géocodage inverse sans connexion réseau, à partir d'un index SQLite construit à partir des exports GeoNames par TsgcMapGeoIndexBuilder. Il utilise le pilote SQLite de FireDAC, il requiert donc Delphi XE7 ou une version ultérieure.
Les recherches directes acceptent un nom de lieu, un préfixe et, en option, un pays ou une région : 'Madrid', 'mostol', 'Toledo, US', 'Madrid ES' ou 'Toledo, Ohio'. Les résultats sont classés par correspondance exacte avant la correspondance de préfixe, puis par population, puis par code d'entité (capitale, puis siège administratif, puis autres lieux habités). Les recherches inverses renvoient les lieux les plus proches dans un rayon de 50 km, les lieux habités d'abord, chaque groupe étant trié par distance.
Chaque composant possède une connexion SQLite et ne doit être utilisé que depuis un seul thread à la fois. Pour interroger le même index depuis plusieurs threads, créez un composant par thread ; le fichier est ouvert en lecture seule.

Propriétés

Propriété	Type	Par défaut	Description
DatabaseFile	string		Le fichier d'index SQLite.
Active	Boolean	False	Ouvre ou ferme l'index.
MaxResults	Integer	5	Nombre maximal de résultats.
CountryCodes	string		Codes ISO alpha-2 séparés par des virgules qui restreignent chaque recherche.
Language	string		Indication de langue. Les exports GeoNames ne fournissent aucune langue par nom alternatif, la valeur est donc actuellement ignorée.

Méthodes

Méthode	Description
Open, Close	Ouvre ou ferme l'index.
Geocode(const aQuery: string; aResults: TsgcMapGeocodingResults): Boolean	Recherche directe. True lorsqu'au moins un lieu a été trouvé.
ReverseGeocode(aLatitude, aLongitude: Double; aResults: TsgcMapGeocodingResults): Boolean	Les lieux les plus proches dans un rayon de 50 km.
ReverseGeocodeCountry(aLatitude, aLongitude: Double): string	Code ISO du pays du lieu le plus proche, vide lorsqu'il n'y en a aucun dans un rayon de 50 km.

Les résultats utilisent la même classe TsgcMapGeocodingResult que TsgcMapGeocoder.

Exemple : recherche hors ligne

```
procedure TForm1.btnOfflineSearchClick(Sender: TObject);
var
  oResults: TsgcMapGeocodingResults;
  i: Integer;
begin
  sgcMapOfflineGeocoder1.DatabaseFile := 'geonames.db';
  sgcMapOfflineGeocoder1.MaxResults := 5;
  sgcMapOfflineGeocoder1.Open;
  oResults := TsgcMapGeocodingResults.Create;
  try
    if sgcMapOfflineGeocoder1.Geocode('Toledo, US', oResults) then
      for i := 0 to oResults.Count - 1 do
        memoLog.Lines.Add(Format('%s (%.4f, %.4f)', [oResults[i].DisplayName,
          oResults[i].Latitude, oResults[i].Longitude]));
    memoLog.Lines.Add('Country at the map center: ' +
      sgcMapOfflineGeocoder1.ReverseGeocodeCountry(sgcVCLMap1.CenterLatitude,
      sgcVCLMap1.CenterLongitude));
  finally
    oResults.Free;
  end;
end;
```

Voir aussi

- TsgcMapGeoIndexBuilder
- TsgcMapGeocoder

TsgcMapGeoIndexBuilder

Unité : sgcMaps_OfflineGeocoder
Construit l'index SQLite que parcourt TsgcMapOfflineGeocoder, à partir d'un export GeoNames téléchargé depuis https://download.geonames.org/export/dump/ : cities500.zip, cities1000.zip, allCountries.zip, un fichier de pays tel que ES.zip, ou le fichier .txt extrait. Requiert Delphi XE7 ou une version ultérieure (FireDAC SQLite).
La construction s'exécute dans le thread appelant. L'outil GeoIndexBuilder de Tools\Maps encapsule ce composant dans une application console.

Propriétés

Propriété	Type	Par défaut	Description
InputFile	string		L'export GeoNames : le fichier .txt séparé par des tabulations ou un .zip qui le contient.
DatabaseFile	string		Le fichier SQLite à créer. Un fichier existant est écrasé.
Admin1CodesFile	string		admin1CodesASCII.txt facultatif, pour les noms des régions de premier niveau.
CountryInfoFile	string		countryInfo.txt facultatif, pour les noms de pays.
FeatureClasses	string	PA	Classes d'entités GeoNames à importer : P lieux habités, A zones administratives, S sites, T relief, L zones, H hydrographie, R routes, U sous-marin, V végétation.
MinPopulation	Int64	0	Les lieux habités en dessous de cette population sont ignorés. Les autres classes ne sont pas filtrées.
IncludeAlternateNames	Boolean	True	Indexe les noms alternatifs jusqu'à 64 caractères, tels que les noms courts et les noms dans d'autres langues.
BatchSize	Integer	2000	Lignes écrites par lot.
RowsRead, PlacesWritten, NamesWritten, ElapsedMs	Int64		Statistiques de la dernière construction. Lecture seule.

Méthodes

Méthode	Description
Build	Construit l'index.
RowsPerSecond: Double	Vitesse de la dernière construction.

Événements

Événement	Description
OnProgress	Déclenché pendant la construction avec les lignes lues, les lieux écrits et les millisecondes écoulées.

Exemple : construire un index

```
procedure TForm1.btnBuildIndexClick(Sender: TObject);
begin
  sgcMapGeoIndexBuilder1.InputFile := 'cities500.zip';
  sgcMapGeoIndexBuilder1.Admin1CodesFile := 'admin1CodesASCII.txt';
  sgcMapGeoIndexBuilder1.CountryInfoFile := 'countryInfo.txt';
  sgcMapGeoIndexBuilder1.DatabaseFile := 'geonames.db';
  sgcMapGeoIndexBuilder1.FeatureClasses := 'PA';
  sgcMapGeoIndexBuilder1.OnProgress := sgcMapGeoIndexBuilder1Progress;
  sgcMapGeoIndexBuilder1.Build;
  memoLog.Lines.Add(Format('%d places indexed in %d ms',
    [sgcMapGeoIndexBuilder1.PlacesWritten, sgcMapGeoIndexBuilder1.ElapsedMs]));
end;

procedure TForm1.sgcMapGeoIndexBuilder1Progress(Sender: TObject;
  aRowsRead, aPlacesWritten, aElapsedMs: Int64);
begin
  Caption := Format('%d rows read, %d places', [aRowsRead, aPlacesWritten]);
  Update;
end;
```

Voir aussi

- TsgcMapOfflineGeocoder

TsgcMapLocation

Unité : sgcMaps_Location
La position de l'appareil. Il encapsule le TLocationSensor de System.Sensors, il requiert donc Delphi XE6 ou une version ultérieure et fonctionne dans les applications VCL et FireMonkey : Windows utilise l'API de localisation de Windows, et Android, iOS et macOS utilisent leurs services de localisation natifs.
Le composant ne connaît aucune carte. **OnLocationChanged** signale chaque mesure, et **UpdateMarker** et **UpdateAccuracyCircle** tiennent à jour un marqueur de position et un cercle de précision dans les collections d'une carte. Avec **FollowMode** défini à mfmCenter, chaque mesure déclenche aussi **OnFollow**, où l'application centre sa carte.

Autorisations

- Android : déclarez ACCESS_FINE_LOCATION dans les Uses Permissions des options du projet, et demandez-la à l'exécution avant de définir Active à True, avec **RequestPermission** (Delphi 10.3 ou version ultérieure) ou avec PermissionsService.RequestPermissions.
- iOS et macOS : ajoutez les clés de description de l'usage de la localisation aux informations de version du projet.

Propriétés

Propriété	Type	Par défaut	Description
Active	Boolean	False	Démarre ou arrête le capteur de localisation.
Accuracy	Double		Précision souhaitée du capteur, en mètres.
Distance	Double		Déplacement minimal, en mètres, entre deux mesures signalées.
FollowMode	TsgcMapLocationFollowMode	mfmNone	mfmCenter déclenche OnFollow après chaque mesure.
Available	Boolean		Indique si l'appareil possède un capteur de localisation. Lecture seule.
HasLocation	Boolean		Indique si une mesure est arrivée. Lecture seule.
LastLatitude, LastLongitude, LastAccuracy, LastHeading, LastSpeed, LastTimestamp			La dernière mesure. Lecture seule.
TrackCount, TrackIndex, TrackIntervalMs, TrackAccuracy			État de la trace simulée chargée avec LoadTrack. Lecture seule.

Méthodes

Méthode	Description

UpdateMarker(aMarkers: TsgcMapMarkers; aTag: Integer)	Ajoute ou déplace le marqueur de position, identifié par son Tag.
UpdateAccuracyCircle(aCircles: TsgcMapCircles; aTag: Integer)	Ajoute ou met à jour un cercle dont le rayon est la précision.
RequestPermission	Demande l'autorisation de localisation sous Android. La réponse arrive dans OnPermissionResult.
SimulateLocation(aLatitude, aLongitude: Double; aAccuracy: Double = 0; aHeading: Double = -1; aSpeed: Double = -1)	Signale une position comme si le capteur l'avait mesurée, pour les tests.
LoadTrack(const aPoints: array of TsgcMapCoord; aIntervalMs: Integer = 1000; aAccuracy: Double = 10)	Charge une liste de points, par exemple les points d'un itinéraire, pour simuler un trajet.
StepTrack: Boolean	Signale le point suivant de la trace chargée. False à la fin de la trace.
ResetTrack	Revient au premier point de la trace.

Événements

Événement	Description
OnLocationChanged	Une nouvelle mesure : latitude, longitude, précision, cap et vitesse.
OnFollow	Avec FollowMode = mfmCenter, déclenché après OnLocationChanged avec la position sur laquelle centrer.
OnStateChanged	Le capteur a été démarré ou arrêté.
OnError	Le capteur a signalé une erreur.
OnPermissionResult	La réponse à RequestPermission.

Exemple : afficher et suivre la position

```
procedure TForm1.chkFollowClick(Sender: TObject);
begin
  if chkFollow.Checked then
    sgcMapLocation1.FollowMode := mfmCenter
  else
    sgcMapLocation1.FollowMode := mfmNone;
  sgcMapLocation1.Active := True;
end;

procedure TForm1.sgcMapLocation1LocationChanged(Sender: TObject;
  aLatitude, aLongitude, aAccuracy, aHeading, aSpeed: Double);
begin
  // keeps a "my location" marker and its accuracy circle on the map
  sgcMapLocation1.UpdateMarker(sgcVCLMap1.Markers);
  sgcMapLocation1.UpdateAccuracyCircle(sgcVCLMap1.Circles);
end;

procedure TForm1.sgcMapLocation1Follow(Sender: TObject;
  aLatitude, aLongitude: Double);
```

```
begin
  sgcVCLMap1.SetCenter(aLatitude, aLongitude);
end;

procedure TForm1.btnSimulateClick(Sender: TObject);
begin
  // a fixed position with an accuracy of 15 meters, for testing
  sgcMapLocation1.SimulateLocation(40.4168, -3.7038, 15);
end;
```

Voir aussi

- TsgcVCLMap
- TsgcFMXMap

Outils

Le dossier **Tools\Maps** contient deux applications console qui préparent les données des composants hors ligne. Elles encapsulent les classes de construction de sgcMaps, le même travail peut donc être fait depuis votre propre code.

Outil	Entrée	Sortie	Utilisé par	Version minimale
GeoIndex-Builder	Un export Geo-Names	Un index SQLite	TsgcMapOffline-Geocoder	Delphi XE7 (FireDAC SQLite)
RouteGraph-Builder	Un extrait OpenS-treetMap	Un fichier .sgcgraph	TsgcMapOffline-Router	Delphi XE2

Compilation des outils

Chaque outil est un seul fichier .dpr. Ouvrez-le dans l'IDE et compilez-le, ou compilez-le depuis une invite de commandes avec le dossier Source dans les chemins d'unités et d'inclusion, par exemple :

```
dcc32 -NSSystem;Data;FireDAC -U..\..\..\Source -I..\..\..\Source sgcMapsGeoIndexBuilder.dpr
```

Construisez les données une seule fois, sur une machine de développement, et déployez les fichiers obtenus avec votre application.

GeoIndexBuilder

sgcMapsGeoIndexBuilder (Tools\Maps\GeoIndexBuilder) construit l'index SQLite du géocodeur hors ligne à partir d'un export GeoNames. C'est une enveloppe console de TsgcMapGeoIndexBuilder qui requiert Delphi XE7 ou une version ultérieure, puisqu'elle écrit l'index avec FireDAC SQLite.

Données d'entrée

Les exports GeoNames sont des téléchargements gratuits depuis https://download.geonames.org/export/dump/ :

- **cities500.zip**, **cities1000.zip**, **cities5000.zip** ou **cities15000.zip** : les lieux habités du monde dont la population dépasse ce seuil.
- **XX.zip**, tel que ES.zip : toutes les entités d'un pays.
- **allCountries.zip** : toutes les entités du monde.
- **admin1CodesASCII.txt** (facultatif) : les noms des zones administratives de premier niveau, telles que les États et les régions.
- **countryInfo.txt** (facultatif) : les noms de pays, utilisés pour qualifier des recherches telles que "Paris, France".

L'outil lit directement le fichier .zip, ou le fichier .txt extrait.

Ligne de commande

```
sgcMapsGeoIndexBuilder <input.zip|txt> <output.sqlite> [--minpop N]
  [--admin1 file] [--countryinfo file] [--classes PA] [--noalternates]
```

Paramètre	Propriété	Signification
input	InputFile	L'export GeoNames, .zip ou .txt.
output	DatabaseFile	Le fichier SQLite à créer. Un fichier existant est écrasé.
--minpop N	MinPopulation	Ignore les lieux habités (classe d'entités P) en dessous de cette population. Les autres classes d'entités ne sont pas filtrées.
--admin1 file	Admin1CodesFile	Le fichier admin1CodesASCII.txt.
--countryinfo file	CountryInfoFile	Le fichier countryInfo.txt.
--classes	FeatureClasses	Les classes d'entités GeoNames à importer. La valeur par défaut est PA : lieux habités et zones administratives. Les autres sont S (sites), T (relief), L (zones), H (hydrographie), R (routes), U (sous-marin) et V (végétation).
--noalternates	IncludeAlternateNames	Omet les noms alternatifs (noms courts et noms dans d'autres langues), ce qui réduit la taille de l'index.

```
sgcMapsGeoIndexBuilder cities500.zip world.sqlite --admin1 admin1CodesASCII.txt --countryinfo countryInfo.txt
```

Sortie

L'outil affiche sa progression, puis les lignes lues, les lieux et les noms écrits, la durée et la taille de l'index :

```
Building world.sqlite from cities500.zip
  ... rows read, ... places, ... s
Done: ... rows read, ... places, ... names, ... s, ... rows/s, ... MB
```

Il se termine avec le code 0 en cas de succès, 1 pour des paramètres erronés et 2 lorsque la construction échoue, après avoir affiché l'erreur.

Utilisation de l'index

Déployez le fichier .sqlite avec l'application et ouvrez-le avec TsgcMapOfflineGeocoder en renseignant son DatabaseFile. Le composant répond alors à Geocode, ReverseGeocode et ReverseGeocodeCountry sans connexion réseau. L'application a besoin du pilote FireDAC SQLite à l'exécution, comme décrit dans Dépannage. Index de géocodage hors ligne explique le contenu de l'index.

RouteGraphBuilder

sgcMapsRouteGraphBuilder (Tools\Maps\RouteGraphBuilder) transforme un extrait OpenStreetMap en graphe de calcul d'itinéraires du calculateur hors ligne. C'est une enveloppe console de TsgcMapRouteGraphBuilder (unité sgcMaps_OfflineRouter) qui requiert Delphi XE2 ou une version ultérieure.

Données d'entrée

Un extrait OpenStreetMap au format **.osm.pbf** ou en XML **.osm**. Des services de téléchargement tels que Geofabrik publient des extraits de chaque continent, pays et région. Choisissez le plus petit extrait qui couvre la zone de votre application, car la durée et la mémoire de la construction augmentent avec l'extrait.

Ligne de commande

```
sgcMapsRouteGraphBuilder <input.osm.pbf|input.osm> <output.sgcgraph>

sgcMapsRouteGraphBuilder andorra-latest.osm.pbf andorra.sgcgraph
```

Sortie

Le constructeur lit le fichier deux fois, d'abord les chemins, puis les coordonnées des nœuds que ces chemins utilisent, et affiche sa progression tous les 10 pour cent. À la fin, il affiche les statistiques du graphe :

```
Input : andorra-latest.osm.pbf
Output: andorra.sgcgraph
  reading ways   0%
  ...
  reading nodes 100%
Routable ways : ...
Nodes used    : ... (missing ...)
Vertices      : ...
Edges         : ...
Shape points  : ...
Street names  : ...
Graph size    : ... bytes
Time          : ... ms
```

Nodes used compte les nœuds que référencent les chemins routables, et missing compte ceux qui ne sont pas dans le fichier, ce qui arrive avec les chemins qui franchissent la limite d'un extrait. L'outil se termine avec le code 0 en cas de succès, 2 pour des paramètres erronés et 1 lorsque la construction échoue, après avoir affiché l'erreur.

Utilisation du graphe

Un seul fichier .sgcgraph contient les profils voiture, vélo et piéton. Déployez-le avec l'application, définissez-le comme GraphFile de TsgcMapOfflineRouter, choisissez le Profile (morCar, morBike ou morFoot), et appelez GetRoute. L'itinéraire revient dans la même classe TsgcMapRoute que celle qu'utilise TsgcMapDirections, il se dessine donc avec sgcMapAddRouteToPolylines ou se transmet à TsgcMapRouteCalculator. Graphes routiers et calcul d'itinéraires hors ligne décrit le graphe et la classe de construction.

Lazarus

sgcMaps possède un package pour Lazarus et Free Pascal : **Packages\Lazarus\sgcMapsLazarus.lpk**. Il installe le contrôle de carte sur la LCL, avec le même modèle de carte et le même code que le contrôle VCL, mais avec moins de fonctionnalités, car les parties qui requièrent des unités de la RTL propres à Delphi ne sont pas compilées sous FPC.
La différence la plus importante : **le contrôle Lazarus ne peut pas télécharger de tuiles, il n'affiche donc que les tuiles d'un dossier.**

Installation du package

1. Démarrez Lazarus et choisissez **Package**, **Open Package File (.lpk)**.
2. Ouvrez **Packages\Lazarus\sgcMapsLazarus.lpk** depuis l'installation de sgcUI. Le package trouve les unités du dossier Source par un chemin relatif, conservez donc la structure de dossiers de l'installation.
3. Appuyez sur **Compile**, puis sur **Use**, **Install**, et laissez Lazarus reconstruire et redémarrer l'IDE.
4. TsgcVCLMap se trouve maintenant dans la page **SGC Maps** de la palette de composants.

Le package requiert les packages LCL, FCL et IDEIntf, que possède toute installation de Lazarus. Un projet qui utilise la carte liste sgcMapsLazarus dans ses packages requis, ce que l'IDE ajoute lorsque vous déposez le contrôle sur un formulaire.

Ce qui est inclus

Inclus	Unité
Le contrôle de carte sur la LCL, TsgcVCLMap	sgcMaps_VCL
Le modèle de carte : couches, marqueurs, polylignes, polygones, rectangles, cercles, étiquettes, cartes de chaleur et options	sgcMaps_Classes
La source de tuiles depuis un dossier (fichiers z/x/y)	sgcMaps_TileSource
La liste des fournisseurs et sgcMapExpandTileURL	sgcMaps_Providers
Les fonctions de géométrie	sgcMaps_Geometry
Le rastériseur logiciel, TsgcMapRasterizer	sgcMaps_Raster

Le contrôle dessine les marqueurs (avec des icônes PNG personnalisées), les regroupements, les formes, les étiquettes et les cartes de chaleur, et il enregistre la vue avec RenderToBitmap, SaveToBMP et SaveToPNG.

Ce qui n'est pas inclus

- **Les tuiles depuis un serveur.** Une couche avec une source mtsHTTP ou un Provider n'affiche aucune tuile sous Lazarus. Préparez un dossier de tuiles sous Delphi avec TsgcMapTileDownloader, ou avec n'importe quel outil qui écrit des tuiles z/x/y, et déployez-le avec l'application.
- **MBTiles et les tuiles vectorielles.** Ces sources requièrent FireDAC et System.ZLib.
- **GeoJSON, GPX et KML.** Les unités figurent dans le package mais se compilent vides sous FPC.
- **Les composants non visuels.** Les services en ligne, le calculateur d'itinéraires, le calculateur hors ligne, le géocodeur hors ligne, le lecteur OSM et TsgcMapLocation ne sont disponibles que sous Delphi.
- **La rotation de la carte.** Options.Bearing est ignoré sur la LCL.
- **L'opacité.** L'opacité des couches, des polylignes et des remplissages n'est pas appliquée sur la LCL.

Exemple

Le programme ci-dessous affiche une carte à partir d'un dossier de tuiles situé à côté de l'exécutable. Créez un nouveau projet Application, ajoutez sgcMapsLazarus à ses packages requis, et remplacez le programme principal par ce code :

```
program LazarusMap;

{$mode delphi}{$H+}

uses
  Interfaces, Forms, Controls, SysUtils,
  sgcMaps_Classes, sgcMaps_VCL;

var
  oForm: TForm;
  oMap: TsgcVCLMap;
  oLayer: TsgcMapLayer;
begin
  Application.Initialize;
  Application.CreateForm(TForm, oForm);
  oForm.Caption := 'sgcMaps on Lazarus';
  oForm.SetBounds(100, 100, 900, 600);

  oMap := TsgcVCLMap.Create(oForm);
  oMap.Parent := oForm;
  oMap.Align := alClient;

  // the LCL control reads its tiles from a z\x\y folder
  oLayer := oMap.Layers.Add;
  oLayer.Name := 'Offline tiles';
  oLayer.TileSourceType := mtsFolder;
  oLayer.TileSourcePath := ExtractFilePath(ParamStr(0)) + 'tiles';
  oLayer.MaxZoom := 16;
  oLayer.Attribution := '(c) OpenStreetMap contributors';

  oMap.SetCenter(41.3874, 2.1686);
  oMap.Zoom := 12;
  oMap.AddMarker(41.3874, 2.1686, 'Barcelona');

  Application.Run;
end.
```

Le dossier **tiles** contient un sous-dossier par niveau de zoom, un sous-dossier par colonne et une image par tuile, par exemple tiles\12\2072\1529.png. Réglez MaxZoom sur le niveau de zoom le plus élevé du dossier, pour que le contrôle ne cherche pas des tuiles qui n'existent pas. Conservez l'attribution des tuiles, car les conditions du fournisseur de tuiles s'appliquent toujours à une copie sur disque.

Voir aussi

- TsgcVCLMap
- Tuiles hors ligne et téléchargeur de tuiles
- Installation

Présentation .NET

sgcMaps for .NET est un portage C# managé de la bibliothèque Delphi sgcMaps, et non une enveloppe autour d'elle. Il a le même modèle de carte, les mêmes fournisseurs de tuiles, les mêmes services en ligne et les mêmes routeur et géocodeur hors ligne, et il dessine la carte nativement avec SkiaSharp, sans contrôle de navigateur et sans JavaScript.
Les fichiers sont compatibles entre les deux bibliothèques : un graphe routier (.sgcgraph), un index GeoNames ou un dossier de tuiles construit avec Delphi s'ouvre dans .NET, et inversement. La suite de QA .NET le vérifie avec un graphe routier et un index GeoNames écrits par la bibliothèque Delphi.

Paquets

Le portage est réparti en quatre paquets NuGet, de sorte qu'un projet n'inclut que ce qu'il utilise. Les quatre ont la version **2.0.0** : sgcMaps for .NET a ses propres numéros de version, distincts de la version Delphi.

PackageId	Contient	Frameworks cibles	Dépend de
esegece.sgcMaps	Le noyau : le modèle de carte (couches, marqueurs, polylignes, polygones, rectangles, cercles, étiquettes, cartes de chaleur et options), les calculs de tuiles et la géométrie, les fournisseurs de tuiles, les sources de tuiles de dossier, MBTiles et HTTP, le téléchargeur de tuiles, l'import et l'export GeoJSON, GPX et KML, les lecteurs de tuiles vectorielles et de styles, et les services en ligne : Geocoder, Directions, Places, Elevation, MapTime-	net8.0, net9.0	Microsoft.Data.Sqlite

	Zone et RouteCalculator.		
esegece.sgcMaps.Rendering	**MapView**, le moteur de carte. Il dessine les tuiles raster et vectorielles, les surcouches, les regroupements, les cartes de chaleur et les popups avec SkiaSharp, gère la saisie du pointeur, de la molette et du clavier, et effectue le rendu vers une image ou un fichier PNG sans fenêtre.	net8.0, net9.0	esegece.sgcMaps, SkiaSharp, SkiaSharp.NativeAssets.Linux.NoDependencies
esegece.sgcMaps.WinForms	**SgcMapControl**, le contrôle de carte Windows Forms, qui héberge un MapView.	net8.0-windows, net9.0-windows	esegece.sgcMaps.Rendering
esegece.sgcMaps.Offline	Les services hors ligne : **OsmReader** (.osm.pbf et XML .osm), **RouteGraphBuilder** et **OfflineRouter**, **GeoIndexBuilder** et **OfflineGeocoder**.	net8.0, net9.0	esegece.sgcMaps

Édition Registered uniquement

sgcMaps for .NET est un produit Registered (commercial). Il n'existe pas d'édition Community ni de paquet **.Community** : les quatre paquets ci-dessus sont les seuls.

Hôtes

- **Windows Forms** : SgcMapControl, l'équivalent du TsgcVCLMap de Delphi.
- **Sans fenêtre** : MapView effectue le rendu dans un canevas SkiaSharp, une image ou un fichier PNG, sous Windows, Linux et macOS. Cela couvre les applications console, les services et les serveurs web.
- **Votre propre hôte** : MapView ne dépend d'aucun framework d'interface utilisateur. Il dessine sur n'importe quel SKCanvas et reçoit sa saisie par PointerDown, PointerMove, PointerUp, Wheel, Pinch, Rotate et KeyDown, exactement comme SgcMapControl le pilote.

Il n'existe pas de contrôle WPF, Avalonia ou MAUI, ni d'équivalent du contrôle FireMonkey TsgcFMXMap.

Espaces de noms

Espace de noms	Contenu
esegece.sgcMaps	Le modèle de carte, MapCoord, MapColors, TileMath, MapGeometry, MapProviders, les sources de tuiles, TileDownloader, GeoJsonFormat, GpxFormat et KmlFormat.
esegece.sgcMaps.Services	Les services en ligne (MapServiceBase, Geocoder, Directions, Places, Elevation, MapTimeZone), RouteCalculator, et les classes de résultat, comme GeocodingResult et Route.
esegece.sgcMaps.Vector	Le lecteur de Mapbox Vector Tile et le lecteur de style MapLibre : VectorTile, VectorStyle, StyleEvaluator.
esegece.sgcMaps.Rendering	MapView, VectorTileSource, VectorTileRenderer et les classes d'arguments d'événements.
esegece.sgcMaps.WinForms	SgcMapControl.
esegece.sgcMaps.Offline	OsmReader, RouteGraphBuilder, OfflineRouter, GeoIndexBuilder et OfflineGeocoder.

Les classes hors ligne partagent les classes de résultat des services en ligne : OfflineRouter remplit un objet **Route** et OfflineGeocoder renvoie des objets **GeocodingResult**, de sorte que le code qui affiche un résultat fonctionne avec les deux. C'est pourquoi les exemples hors ligne utilisent aussi esegece.sgcMaps.Services.

Classes à libérer

Ces classes détiennent des fichiers, des connexions, des threads ou des objets SkiaSharp natifs, et implémentent IDisposable. Libérez-les avec Dispose, ou créez-les avec une instruction using :

- **MapView** : ses chargeurs de tuiles et les images des icônes de marqueurs. **SgcMapControl** libère son propre MapView, son tampon d'arrière-plan et son minuteur.
- **Geocoder**, **Directions**, **Places**, **Elevation** et **MapTimeZone** : un HttpClient qui leur est propre lorsqu'ils ont été créés avec un HttpMessageHandler.
- **TileDownloader** : son HttpClient.
- **OfflineRouter** et **OfflineGeocoder** : Dispose appelle Close, qui libère le graphe routier ou la connexion SQLite.

Voir aussi

- Démarrage rapide : WinForms
- Rendu sans interface graphique
- Présentation (Delphi)
- Installation .NET
- Correspondance des classes
- Différences .NET par rapport à Delphi

Installation .NET

L'installateur sgcUI installe les packages Delphi, C++Builder et Lazarus. sgcMaps for .NET est distribué à la place sous forme de paquets NuGet distincts, pour l'édition Registered uniquement.

Ajouter les paquets

Ajoutez le paquet de l'hôte que vous utilisez. Chaque paquet apporte ceux dont il dépend, de sorte qu'une application Windows Forms n'a besoin que de esegece.sgcMaps.WinForms, et un moteur de rendu sans interface graphique que de esegece.sgcMaps.Rendering.

```
dotnet add package esegece.sgcMaps.WinForms --version 2.0.0
dotnet add package esegece.sgcMaps.Rendering --version 2.0.0
dotnet add package esegece.sgcMaps.Offline --version 2.0.0
dotnet add package esegece.sgcMaps --version 2.0.0
```

Paquet	À ajouter pour	Frameworks cibles
esegece.sgcMaps.WinForms	Un contrôle de carte sur un formulaire Windows Forms.	net8.0-windows, net9.0-windows
esegece.sgcMaps.Rendering	Des cartes rendues sans fenêtre, ou une carte dans un hôte de votre choix.	net8.0, net9.0
esegece.sgcMaps.Offline	Le calcul d'itinéraires hors ligne, le géocodage hors ligne et la lecture des fichiers OpenStreetMap.	net8.0, net9.0
esegece.sgcMaps	Le modèle de carte, les formats, le téléchargeur de tuiles et les services en ligne uniquement, sans rien dessiner.	net8.0, net9.0

Les mêmes lignes dans un fichier de projet :

```
<ItemGroup>
  <PackageReference Include="esegece.sgcMaps.WinForms" Version="2.0.0" />
  <PackageReference Include="esegece.sgcMaps.Offline" Version="2.0.0" />
</ItemGroup>
```

Lorsque les paquets arrivent sous forme de fichiers .nupkg plutôt que depuis un flux, ajoutez d'abord leur dossier comme source de paquets :

```
dotnet nuget add source C:\Packages\sgcMaps --name sgcMaps
```

Prérequis

- .NET 8 ou .NET 9. Les paquets ne ciblent ni .NET Framework ni .NET Standard.
- Le contrôle Windows Forms nécessite Windows. Les trois autres paquets fonctionnent sous Windows, Linux et macOS.
- esegece.sgcMaps utilise **Microsoft.Data.Sqlite** pour lire les fichiers MBTiles et l'index GeoNames. Rien ne doit être installé pour cela.
- esegece.sgcMaps.Rendering utilise **SkiaSharp** et apporte **SkiaSharp.NativeAssets.Linux.NoDependencies**, de sorte qu'il effectue aussi le rendu sur un serveur Linux ou dans un conteneur, sans bibliothèque native supplémentaire.

Pas d'édition Community

Il n'existe pas d'édition Community (gratuite) de sgcMaps for .NET, ni de paquet avec le suffixe **.Community**. Chaque paquet ci-dessus nécessite une licence Registered.

Voir aussi

- Démarrage rapide : WinForms
- Installation (Delphi)
- Présentation .NET
- Correspondance des classes
- Différences .NET par rapport à Delphi

Démarrage rapide : WinForms

SgcMapControl est le contrôle de carte Windows Forms, l'équivalent .NET du TsgcVCLMap de Delphi. Il est livré dans le paquet **esegece.sgcMaps.WinForms**, pour net8.0-windows et net9.0-windows, et apporte avec lui esegece.sgcMaps.Rendering et esegece.sgcMaps.
Le contrôle héberge un MapView, le moteur de carte, dans sa propriété **Map**. Il peint la vue avec SkiaSharp dans un tampon d'arrière-plan et lui transmet la souris, la molette et le clavier. Les tuiles se chargent dans des threads d'arrière-plan, de sorte que l'interface utilisateur n'attend jamais le réseau ; le contrôle se repeint sur le thread de l'interface à leur arrivée, et déclenche aussi TileError sur ce thread.

1. Créer le projet

```
dotnet new winforms -n MapQuickStart
cd MapQuickStart
dotnet add package esegece.sgcMaps.WinForms --version 2.0.0
```

2. Afficher une carte

Remplacez Program.cs par le programme ci-dessous, et supprimez Form1.cs et Form1.Designer.cs. Il ajoute une couche OpenStreetMap avec un cache disque, centre la carte sur Madrid, et ajoute un marqueur déplaçable et une ligne.
MapProviders.ApplyProvider copie les paramètres du fournisseur dans la couche : TileSourceType, UrlTemplate, MaxZoom et Attribution. Une couche dont le Provider n'est pas Custom télécharge les tuiles de son fournisseur même sans cet appel, il n'est donc nécessaire que si vous voulez lire ou modifier ces valeurs.

3. L'exécuter

```
dotnet run
```

Faites glisser la carte pour la déplacer, tournez la molette pour zoomer, et faites glisser le marqueur. Le titre du formulaire affiche le dernier clic, la position où le marqueur a été déposé, ou la tuile en échec.

Propriétés

Propriété	Type	Description
Map	MapView	Le moteur de carte. Des méthodes comme AddMarker, AddPolyline, FitToMarkers, SetLayerVisible et RenderToPng s'appellent sur lui.
CenterLatitude, CenterLongitude	double	Le centre de la vue, en degrés.
Zoom	int	Le niveau de zoom, limité par Options.MinZoom et Options.MaxZoom.
Bearing	double	La rotation de la carte en degrés dans le sens horaire à partir du nord. L'utilisateur peut faire pivoter la carte lorsque Options.RotationEnabled vaut true.
Options	MapOptions	Le comportement et les décorations de la vue : limites de zoom, TileSize, boutons de zoom, échelle graphique, attribu-

		tion, couleur d'arrière-plan, déplacement, zoom, rotation et sélecteur de couches.
MarkerOptions	MarkerOptions	La taille, la couleur et la police par défaut des marqueurs, et les options de regroupement dans MarkerOptions.Clustering.
Layers	MapItemCollection<Layer>	Les couches de tuiles.
Markers, Polylines, Polygons, Circles, Rectangles, Labels, Heatmaps	MapItemCollection<T>	Les surcouches. Ajoutez un nouvel élément avec Add(), ou avec les méthodes Add de Map.

Les propriétés habituelles de Control, comme Dock, Anchor, Enabled, Visible et TabStop, sont également disponibles.

Événements

Événement	Arguments	Déclenché lorsque
MapClick	MapClickEventArgs: Latitude, Longitude	La carte est cliquée.
MarkerClick	MapItemEventArgs<Marker>: Item	Un marqueur est cliqué.
MarkerDragStart, MarkerDrag, MarkerDragEnd	MapItemEventArgs<Marker>: Item	Un marqueur dont Draggable vaut true est déplacé.
ShapeEdited	MapItemEventArgs<MapItem>: Item	L'utilisateur a déplacé, ajouté ou supprimé un sommet d'une polyligne ou d'un polygone modifiable.
ClusterClick	ClusterClickEventArgs: Count, Latitude, Longitude, Handled	Un regroupement de marqueurs est cliqué. Définissez Handled sur true pour traiter le clic vous-même.
LayerChanged	MapItemEventArgs<Layer>: Item	Une couche est affichée ou masquée, par exemple depuis le sélecteur de couches.
ZoomChanged, BearingChanged	EventArgs	Le niveau de zoom ou la rotation change.
TileError	TileErrorEventArgs: Layer, Zoom, X, Y, Error	Une tuile ne peut pas être chargée, par exemple à cause d'une erreur HTTP du serveur de tuiles.

Ce sont des événements C# ordinaires : plusieurs gestionnaires peuvent être attachés avec +=.

Voir aussi

- Rendu sans interface graphique
- Démos .NET
- Tuiles, couches et fournisseurs
- Présentation .NET
- Installation .NET
- Correspondance des classes

- Différences .NET par rapport à Delphi

Rendu sans interface graphique

MapView, dans le paquet **esegece.sgcMaps.Rendering**, est le moteur de carte de sgcMaps for .NET. SgcMapControl l'affiche dans une fenêtre, mais il n'a besoin d'aucune fenêtre : il effectue le rendu dans un canevas SkiaSharp, une image ou un fichier PNG. Une application console, un service Windows, un démon Linux ou un serveur web peut donc produire des cartes statiques, avec les mêmes couches et surcouches que le contrôle. C'est l'équivalent .NET de l'usage sans interface graphique des contrôles Delphi décrit dans Rendu de cartes statiques.

Rendu vers un fichier PNG

Le programme ci-dessous effectue le rendu de la tour Eiffel au zoom 15 avec un marqueur et un cercle, et écrit un fichier PNG de 1024 x 768.
RenderToPng dessine la vue, attend les tuiles encore en cours de chargement, et la redessine jusqu'à ce que toutes les tuiles soient arrivées ou que **tileTimeout** soit écoulé (30 secondes s'il n'est pas indiqué). Après l'appel, **LastRenderMissingTiles** indique combien de tuiles manquaient dans l'image. La taille de la vue est restaurée ensuite, de sorte qu'un MapView peut produire des images de plusieurs tailles.

Rendu en mémoire

RenderToImage renvoie un SKImage au lieu d'écrire un fichier. Encodez-le dans le format dont vous avez besoin, par exemple pour répondre à une requête web avec des octets PNG :

Dessiner sur votre propre canevas

Render(SKCanvas canvas, int width, int height, bool includeControls) dessine la vue une fois sur n'importe quel canevas SkiaSharp, sans attendre. Les tuiles pas encore chargées sont mises en file d'attente en arrière-plan, et l'événement **Invalidated** demande le dessin suivant à leur arrivée. Invalidated peut être déclenché sur un thread de travail, de sorte qu'un hôte doit basculer vers son propre thread d'interface avant de dessiner, comme le fait SgcMapControl.

Ce qui est dessiné

- Les couches de tuiles, les étiquettes des tuiles vectorielles, les cartes de chaleur, les polygones, les rectangles, les polylignes, les cercles, les marqueurs et les étiquettes, dans cet ordre.
- Les boutons de zoom, le sélecteur de couches et la boussole uniquement lorsque **includeControls** vaut true. Il vaut false par défaut dans RenderToImage et RenderToPng.
- L'échelle graphique et l'attribution lorsque **Options.ShowScaleBar** et **Options.ShowAttribution** valent true, quelle que soit la valeur de includeControls. Gardez l'attribution visible : les fournisseurs de tuiles l'exigent.

Membres pour l'usage sans interface graphique

Membre	Description
SetSize(int width, int height)	Définit la taille de la vue, utilisée par FitBounds, FitToMarkers et les conversions de coordonnées avant un rendu.
SetCenter(double latitude, double longitude), Zoom, Bearing	La vue.
FitBounds, FitToMarkers(int paddingPx), FitToPolyline(Polyline polyline, int paddingPx)	Choisissent le centre et le zoom qui montrent une zone, les marqueurs ou une polyligne.

AddMarker, AddPolyline, AddPolygon, AddCircle, AddRectangle, AddLabel	Ajoutent des surcouches. AddMarker accepte le texte du popup et celui de l'info-bulle.
Render, RenderToImage, RenderToPng	Dessinent la vue, comme décrit ci-dessus.
WaitForTiles(TimeSpan timeout)	Bloque jusqu'à ce que les tuiles en file d'attente soient chargées ou que le délai expire, et renvoie true lorsque plus rien n'est en attente.
PendingTileCount	Le nombre de tuiles en cours de chargement en arrière-plan.
LastRenderMissingTiles	Le nombre de tuiles manquantes lors du dernier rendu.
DrawnMarkerCount, DrawnClusterCount, DrawnLabelCount	Ce que le dernier rendu a dessiné.
AttributionText()	L'attribution des couches visibles, à imprimer à côté de l'image lorsqu'elle n'y est pas dessinée.
LatLonToScreen, ScreenToLatLon	Convertissent entre les coordonnées et les pixels de la vue.

Linux et macOS

Le paquet Rendering cible net8.0 et net9.0 sans dépendance à Windows, et apporte la bibliothèque native SkiaSharp pour Linux (SkiaSharp.NativeAssets.Linux.NoDependencies), qui n'a besoin ni de fontconfig ni d'aucun autre paquet système. Un moteur de rendu sans interface graphique peut donc s'exécuter dans un conteneur Linux. Les tuiles d'un serveur nécessitent un accès HTTPS sortant ; les tuiles d'un dossier ou d'un fichier MBTiles n'en nécessitent aucun.

Voir aussi

- Démos .NET
- Données hors ligne .NET
- 05 Rendu statique (Delphi)
- Présentation .NET
- Installation .NET
- Correspondance des classes
- Différences .NET par rapport à Delphi

Données hors ligne .NET

sgcMaps for .NET peut afficher une carte, calculer des itinéraires et géocoder sans connexion réseau. Il utilise trois types de données locales : des tuiles sur disque, un graphe routier construit à partir d'OpenStreetMap, et un index GeoNames. Toutes sont compatibles au niveau fichier avec la bibliothèque Delphi, de sorte que les données construites avec les outils Delphi peuvent être utilisées ici, et inversement.

Données	Construit avec	Utilisé par	Paquet
Dossier de tuiles (z/x/y)	TileDownloader, ou n'importe quel outil de tuiles	Une couche avec TileSourceType.Folder	esegece.sgcMaps
Fichier MBTiles, raster ou vectoriel	N'importe quel outil MBTiles	Une couche avec TileSourceType.MBTiles ou TileSourceType.VectorMBTiles	esegece.sgcMaps, esegece.sgcMaps.Rendering pour les tuiles vectorielles
Graphe routier (.sgcgraph)	RouteGraphBuilder	OfflineRouter	esegece.sgcMaps.Offline
Index GeoNames (SQLite)	GeoIndexBuilder	OfflineGeocoder	esegece.sgcMaps.Offline

Tuiles hors ligne

Une couche lit ses tuiles depuis un dossier ou un fichier MBTiles lorsque **TileSourceType** vaut Folder ou MBTiles et que **TileSourcePath** y pointe. Les fichiers MBTiles vectoriels sont dessinés avec le style MapLibre indiqué dans **VectorStyleFile**, ou avec le style intégré lorsqu'il est vide. Voir Tuiles hors ligne et téléchargeur de tuiles et Tuiles vectorielles.

Téléchargement d'un pack de tuiles

TileDownloader télécharge les tuiles d'une zone et d'une plage de niveaux de zoom dans un dossier z/x/y, et ignore celles qui s'y trouvent déjà. Il n'a volontairement pas de UrlTemplate par défaut : tile.openstreetmap.org et la plupart des serveurs de tuiles publics interdisent les téléchargements en masse, pointez-le donc vers votre propre serveur ou vers un fournisseur dont les conditions les autorisent, et gardez DelayMS supérieur à zéro.

Calcul d'itinéraires hors ligne

RouteGraphBuilder lit un extrait OpenStreetMap, un fichier .osm.pbf ou un fichier XML .osm, et écrit le réseau routier sous forme de fichier .sgcgraph avec un profil voiture, un profil vélo et un profil piéton. **OfflineRouter** charge ce fichier et répond aux demandes d'itinéraire sans réseau. L'itinéraire est un objet **Route**, la même classe que renvoie le service en ligne Directions, avec la distance en mètres, la durée en secondes, les points, les tronçons et les étapes.

Membre	Description
RouteGraphBuilder.Build(string inputFile, string outputFile)	Construit le graphe. Ses statistiques sont dans WayCount, NodeCount, MissingNodes, VertexCount, EdgeCount, ShapeCount et NameCount, et Progress indique les octets lus.

OfflineRouter.GraphFile, Open(), Close(), Active	Le fichier du graphe, et son ouverture ou sa fermeture. Active est en lecture seule ; le constructeur qui prend un nom de fichier ne fait que définir GraphFile.
OfflineRouter.Profile	OfflineRouterProfile.Car (par défaut), Bike ou Foot.
OfflineRouter.GetRoute(Route route, params MapCoord[] waypoints)	Calcule un itinéraire passant par deux points de passage ou plus. Renvoie false et définit LastError lorsqu'il n'y a pas d'itinéraire. Une surcharge accepte un IReadOnlyList<MapCoord>.

Construire le graphe d'un pays entier demande du temps et de la mémoire ; construisez-le une fois et livrez le fichier .sgcgraph. Voir Graphes routiers et calcul d'itinéraires hors ligne pour les phases de la construction.

Géocodage hors ligne

GeoIndexBuilder transforme un export GeoNames en index SQLite, et **OfflineGeocoder** le parcourt par nom ou par position. Téléchargez un export comme cities1000.txt, ainsi que admin1CodesASCII.txt et countryInfo.txt, depuis le serveur d'export de GeoNames. Les résultats sont des objets **GeocodingResult**, comme avec le Geocoder en ligne.

Membre	Par défaut	Description
GeoIndexBuilder.InputFile, DatabaseFile		L'export GeoNames à lire et l'index à écrire.
GeoIndexBuilder.Admin1CodesFile, CountryInfoFile		Les fichiers qui transforment les codes en noms de région et de pays.
GeoIndexBuilder.FeatureClasses	PA	Les classes d'entités GeoNames à importer : P pour les lieux habités, A pour les zones administratives.
GeoIndexBuilder.MinPopulation	0	Ignore les lieux dont la population est inférieure.
GeoIndexBuilder.IncludeAlternateNames	true	Indexe aussi les noms alternatifs, comme les noms dans d'autres langues.
GeoIndexBuilder.BatchSize	2000	Lignes écrites par lot.
OfflineGeocoder.MaxResults	5	Le nombre maximal de résultats.
OfflineGeocoder.CountryCodes		Codes pays ISO alpha-2 séparés par des virgules, comme ES,PT, qui restreignent chaque recherche.
OfflineGeocoder.Geocode, ReverseGeocode, ReverseGeocodeCountry		Recherche par nom, recherche des lieux les plus proches d'une position, et renvoi du pays d'une position.

OfflineGeocoder, comme OfflineRouter, s'ouvre avec Open() et se ferme avec Close() ou Dispose().

Lecture des fichiers OpenStreetMap

OsmReader lit en flux un fichier .osm.pbf ou XML .osm et déclenche **Node**, **Way** et **Relation** pour chaque élément, avec ses tags, et **Progress** avec les octets lus. ReadNodes, ReadWays, ReadRelations, SkipNodeTags et WayFilter permettent d'ignorer ce dont vous n'avez pas besoin. RouteGraphBuilder repose sur lui, et il sert à tout autre import de données OpenStreetMap.

Voir aussi

- Démos .NET
- Index de géocodage hors ligne
- Présentation .NET
- Installation .NET
- Correspondance des classes
- Différences .NET par rapport à Delphi

Correspondance des classes

Le tableau fait correspondre chaque composant et chaque classe Delphi de sgcMaps à son équivalent .NET. Les classes .NET abandonnent le préfixe Tsgc et utilisent les noms .NET habituels ; leurs membres gardent les noms Delphi partout où le concept est le même.

Contrôles de carte et modèle

Delphi	.NET	Paquet
TsgcVCLMap	SgcMapControl (Windows Forms), avec son moteur dans SgcMapControl.Map	esegece.sgcMaps.WinForms
TsgcFMXMap	Pas d'équivalent. MapView peut être hébergé par tout framework qui dessine avec SkiaSharp.	esegece.sgcMaps.Rendering
Un contrôle de carte effectuant le rendu sans fenêtre, TsgcMapRasterizer	MapView: Render, RenderToImage, RenderToPng	esegece.sgcMaps.Rendering
TsgcMapLayer, TsgcMapMarker, TsgcMapPolyline, TsgcMapPolygon, TsgcMapRectangle, TsgcMapCircle, TsgcMapLabel, TsgcMapHeatmap	Layer, Marker, Polyline, Polygon (trous dans PolygonHoles), Rectangle, Circle, Label, Heatmap	esegece.sgcMaps
TsgcMapLayers, TsgcMapMarkers et les autres collections	MapItemCollection<T>	esegece.sgcMaps
TsgcMapOptions, TsgcMapMarkerOptions, TsgcMapClusterOptions	MapOptions, MarkerOptions, ClusterOptions	esegece.sgcMaps
TsgcMapCoord, TsgcMapHeatmapPoint	MapCoord, HeatmapPoint (readonly record structs)	esegece.sgcMaps

Tuiles, fournisseurs et formats

Delphi	.NET	Paquet
TsgcMapTileProvider, TsgcMapTileSourceType	TileProvider, TileSourceType	esegece.sgcMaps
sgcMapApplyProvider, sgcMapGetProviderInfo, sgcMapProviderNames, sgcMapExpandTileURL, TsgcMapProviderInfo	MapProviders.ApplyProvider, GetInfo, ProviderNames, ExpandTileUrl, ProviderInfo	esegece.sgcMaps
Les sources de tuiles de dossier, MBTiles et HTTP	FolderTileSource, MBTilesTileSource, HttpTileSource (ITileSource), TileSourceFactory	esegece.sgcMaps
TsgcMapTileSource_MVT	VectorTileSource	esegece.sgcMaps.Rendering

TsgcMapVectorTile, TsgcMapVectorLayer, TsgcMapVectorFeature, TsgcMapVectorStyle	VectorTile, VectorLayer, VectorFeature, VectorStyle (espace de noms esegece.sgcMaps.Vector)	esegece.sgcMaps
TsgcMapTileDownloader	TileDownloader	esegece.sgcMaps
TsgcMapGeoJSON, TsgcMapGPX, TsgcMapKML	GeoJsonFormat, GpxFormat, KmlFormat	esegece.sgcMaps
TsgcMapGPXTrackStats, TsgcMapElevationGradient	GpxTrackStats, ElevationGradient	esegece.sgcMaps
Les fonctions de sgcMaps_Geometry (sgcMapDistance, sgcMapInitialBearing, sgcMapDestinationPoint, sgcMapGeodesicInterpolate, sgcMapPointInPolygon, sgcMapSimplify, sgcMapNearestPointOnPolyline, sgcMapClusterPoints, sgcMapRenderHeatmap)	La classe statique MapGeometry : Distance, InitialBearing, DestinationPoint, GeodesicInterpolate, PointInPolygon, Simplify, NearestPointOnPolyline, ClusterPoints, RenderHeatmap	esegece.sgcMaps
Les calculs de tuiles de sgcMaps_Classes (sgcMapLatLonToTileF, sgcMapTileFToLatLon, sgcMapLatLonToPixel, sgcMapPixelToLatLon, sgcMapTileCount)	La classe statique TileMath : TryLatLonToTile, TryTileToLatLon, TryLatLonToPixel, TryPixelToLatLon, TileCount	esegece.sgcMaps

Services en ligne et calcul d'itinéraires

Delphi	.NET	Paquet
TsgcMapService	MapServiceBase	esegece.sgcMaps
TsgcMapGeocoder	Geocoder, avec GeocodingResult et GeocodingProvider	esegece.sgcMaps
TsgcMapDirections	Directions, avec Route, RouteLeg, RouteStep, DirectionsProvider, TravelMode et PolylineCodec	esegece.sgcMaps
sgcMapAddRouteToPolylines	Route.AddToPolylines	esegece.sgcMaps
TsgcMapPlaces	Places, avec Place et PlacesProvider	esegece.sgcMaps
TsgcMapElevation	Elevation, avec ElevationResult et ElevationProvider	esegece.sgcMaps
TsgcMapTimeZone	MapTimeZone, avec MapTimeZoneInfo et TimeZoneProvider	esegece.sgcMaps
TsgcMapRouteCalculator	RouteCalculator	esegece.sgcMaps

Hors ligne

Delphi	.NET	Paquet
TsgcMapOfflineRouter	OfflineRouter, avec OfflineRouterProfile	esegece.sgcMaps.Offline

TsgcMapRouteGraphBuilder, l'outil RouteGraphBuilder	RouteGraphBuilder	esegece.sgcMaps.Offline
TsgcMapOfflineGeocoder	OfflineGeocoder	esegece.sgcMaps.Offline
TsgcMapGeoIndexBuilder, l'outil GeoIndexBuilder	GeoIndexBuilder	esegece.sgcMaps.Offline
TsgcMapOSMReader, TsgcMapOSMTags, TsgcMapOSMMember	OsmReader, OsmTags, OsmMember	esegece.sgcMaps.Offline

Pas d'équivalent .NET

- TsgcMapLocation, la position de l'appareil.
- Les contrôles FireMonkey et Lazarus.
- Les outils en ligne de commande GeoIndexBuilder et RouteGraphBuilder. Appelez les classes du même nom, ou utilisez OfflineDemo, qui construit les deux fichiers.

Voir aussi

- Présentation .NET
- Installation .NET
- Différences .NET par rapport à Delphi

Différences .NET par rapport à Delphi

Le portage .NET conserve le modèle, le comportement et les formats de fichier de la bibliothèque Delphi. Ce qui change, c'est la forme de l'API, là où .NET a ses propres conventions.

Noms

- Les classes abandonnent le préfixe Tsgc : TsgcMapGeocoder devient Geocoder et TsgcVCLMap devient SgcMapControl. La Correspondance des classes les liste toutes.
- Les énumérations abandonnent aussi leurs préfixes : mtpOpenStreetMap devient TileProvider.OpenStreetMap, et mtsMBTiles devient TileSourceType.MBTiles.
- Certains membres suivent la casse .NET, comme UrlTemplate et NominatimUrl. La plupart gardent leurs noms Delphi.

Un moteur, un contrôle

Dans Delphi, les contrôles de carte détiennent la carte. Dans .NET, la carte est un **MapView**, qui fonctionne sans fenêtre, et **SgcMapControl** est un contrôle Windows Forms qui en héberge un dans sa propriété Map. Le contrôle expose directement la vue et les collections (Zoom, Layers, Markers...), et des méthodes comme AddMarker, FitToMarkers ou RenderToPng s'appellent sur SgcMapControl.Map.

Les services en ligne sont asynchrones

Dans Delphi, chaque requête d'un service a une méthode synchrone et une méthode Async qui répond dans un événement de résultat, comme OnGeocodeResult. Dans .NET, chaque requête est une méthode basée sur Task qui renvoie le résultat : GeocodeAsync et ReverseGeocodeAsync, GetRouteAsync, SuggestAsync et RetrieveAsync, GetElevationsAsync et GetElevationAlongPathAsync, GetTimeZoneAsync, et RouteCalculator.CalculateAsync. Chacune accepte un CancellationToken facultatif. Il n'y a pas d'événements de résultat, et aucun CheckSynchronize n'est nécessaire dans une application console.

- Une requête en échec lève une **ServiceException**, avec le statut HTTP dans StatusCode (0 lorsqu'il n'y a pas eu de réponse HTTP). Elle définit aussi LastError et déclenche l'événement Error, comme OnError dans Delphi.
- Une requête sans résultat n'est pas une erreur : la liste est vide, et Geocoder définit LastError sur "No results".
- Les services partagent un même HttpClient par défaut. Un constructeur qui prend un HttpMessageHandler ou un HttpClient permet d'utiliser un proxy ou un gestionnaire de test.
- L'espacement, à l'échelle du processus, des requêtes vers Nominatim, le serveur de démonstration OSRM et OpenTopoData, d'une par seconde, est le même que dans Delphi.

Événements

Les événements sont des événements C# de type EventHandler<T>, comme MapClick avec MapClickEventArgs. Plusieurs gestionnaires peuvent être attachés avec += et retirés avec -=, là où un événement Delphi ne contient qu'une méthode. Les données d'un paramètre d'événement Delphi sont une propriété des arguments : e.Item pour un marqueur, e.Latitude et e.Longitude pour un clic, e.Handled pour arrêter l'action par défaut.

Couleurs

Les couleurs sont des valeurs uint dans l'ordre ARGB, 0xAARRGGBB, comme MapColors.Blue (0xFF0000FF), là où le modèle Delphi utilise TColor. **MapColors.FromTColor** et **MapColors.ToTColor** convertissent entre les deux, et **MapColors.FromRgb** construit une couleur opaque.

Coordonnées et collections

- MapCoord et HeatmapPoint sont des readonly record structs, créés avec new MapCoord(latitude, longitude).
- Les collections de surcouches sont des MapItemCollection<T>, une Collection<T> avec Add(), BeginUpdate, EndUpdate et un événement Changed.
- Une icône de marqueur est le contenu d'un fichier image dans Marker.Icon, un tableau d'octets, chargé avec LoadIconFromFile ou LoadIconFromStream.

Routeur et géocodeur hors ligne

OfflineRouter et OfflineGeocoder s'ouvrent avec **Open()** et se ferment avec **Close()** ou Dispose(). Leur propriété Active est en lecture seule, alors que dans Delphi mettre Active à True les ouvre. OfflineRouter.GetRoute et OfflineGeocoder.Geocode sont synchrones, comme dans Delphi.

Rendu

Le moteur dessine avec SkiaSharp sur toutes les plateformes. RenderToPng écrit des fichiers PNG ; pour un autre format, encodez le SKImage de RenderToImage. Voir Rendu sans interface graphique.

Plateformes et versions

- Les paquets ciblent .NET 8 et .NET 9, et le contrôle cible net8.0-windows et net9.0-windows. Il n'existe pas de build pour .NET Framework.
- La version des paquets est 2.0.0, distincte de la version Delphi.
- Il n'existe pas d'édition Community. Voir Installation .NET.
- TsgcMapLocation, le contrôle FireMonkey et les outils en ligne de commande n'ont pas d'équivalent.

Voir aussi

- Présentation .NET
- Installation .NET
- Correspondance des classes

Démos .NET

Trois démos sont fournies avec sgcMaps for .NET, dans le dossier **.net\demos\Maps** de sgcUI. Chacune est un petit projet de style SDK pour net8.0 ou net8.0-windows. Elle référence les projets sgcMaps lorsque le code source de la bibliothèque se trouve à côté (.net\source\Maps), et les paquets NuGet de version 2.0.0 sinon.

Démo	Type	Paquets	Montre
Console-RenderDemo	Console, net8.0	esegece.sgcMaps, esegece.sgcMaps.Rendering	Une carte rendue vers un fichier PNG sans fenêtre, avec des marqueurs et un itinéraire du service en ligne Directions.
OfflineDemo	Console, net8.0	esegece.sgcMaps, esegece.sgcMaps.Offline	Calcul d'itinéraires et géocodage hors ligne, y compris la construction du graphe routier et de l'index GeoNames.
WinFormsDemo	Windows Forms, net8.0-windows	esegece.sgcMaps.WinForms	Le contrôle de carte avec deux couches de base, des marqueurs, des formes modifiables, un cercle, une carte de chaleur et les événements.

ConsoleRenderDemo

La version .NET de la démo Delphi 05 Rendu statique. Sans argument, elle effectue le rendu de Madrid au zoom 13 avec l'itinéraire en voiture de la Puerta del Sol au parc du Retiro, récupéré depuis le serveur OSRM public, et écrit **madrid.png** dans le dossier courant. Les tuiles sont mises en cache dans un dossier tile_cache à côté de l'exécutable.

```
dotnet run --project ConsoleRenderDemo
dotnet run --project ConsoleRenderDemo -- --center 48.8584,2.2945 --zoom 15 --size 800x600 --provider carto --marker 48.858
```

Argument	Description
--center lat,lon	Le centre de la carte.
--zoom z	Le niveau de zoom, 13 par défaut.
--size WxH	La taille de l'image, 1024x768 par défaut.
--provider osm|opentopo|esri|carto	Le fournisseur de tuiles, osm par défaut.
--marker lat,lon	Ajoute un marqueur. Peut être répété.
--route lat,lon;lat,lon[;...]	Demande au serveur OSRM un itinéraire passant par les points et le dessine.
--out file.png	Le fichier de sortie, madrid.png par défaut.

La démo affiche le fournisseur, la vue, l'itinéraire, le chemin complet du fichier et le texte d'attribution. Elle nécessite un accès à Internet pour les tuiles et l'itinéraire.

OfflineDemo

Calcule des itinéraires et géocode sans aucun appel réseau. Elle construit le graphe routier à partir d'un extrait OpenStreetMap, ou réutilise le fichier .sgcgraph s'il existe, et calcule un itinéraire en voiture entre deux points. Lorsque les arguments GeoNames sont fournis, elle construit aussi l'index GeoNames, ou le réutilise, et géocode une requête.

```
dotnet run --project OfflineDemo -- --pbf andorra-latest.osm.pbf --from 42.5063,1.5218 --to 42.5350,1.5807
dotnet run --project OfflineDemo -- --pbf andorra-latest.osm.pbf --from 42.5063,1.5218 --to 42.5350,1.5807 --geonames citie
```

Argument	Description
--pbf file.osm.pbf	L'extrait OpenStreetMap. Obligatoire.
--from lat,lon, --to lat,lon	Le départ et l'arrivée de l'itinéraire. Obligatoires.
--graph file.sgcgraph	Le graphe routier. Par défaut, le nom de l'extrait avec l'extension .sgcgraph.
--geonames, --admin1, --countryinfo	L'export GeoNames, admin1CodesASCII.txt et countryInfo.txt.
--geocode "query"	Le lieu à rechercher.
--db file.sqlite	L'index GeoNames. Par défaut, le nom de l'export avec l'extension .sqlite.

Des extraits sont disponibles sur les serveurs de téléchargement d'OpenStreetMap, comme Geofabrik, et les fichiers GeoNames sur le serveur d'export de GeoNames. Voir Données hors ligne .NET.

WinFormsDemo

Un formulaire avec un **SgcMapControl** et une barre d'état. Il montre :

- Deux couches de base, OpenStreetMap et OpenTopoMap, avec le sélecteur de couches, et la rotation activée.
- Des marqueurs déplaçables sur Madrid et Barcelone, et un marqueur sur Valence.
- Une polyligne modifiable de Madrid à Barcelone en passant par Saragosse, et un polygone modifiable.
- Un cercle de 40 km autour de Valence, et une carte de chaleur de 200 points pondérés autour de Séville.
- Les événements MarkerClick, MapClick, MarkerDragEnd, ShapeEdited, ZoomChanged et TileError, écrits dans la barre d'état.

Lorsque le formulaire s'affiche, la carte s'ajuste aux marqueurs. Les tuiles sont mises en cache dans le dossier sgcMaps_tiles du dossier temporaire.

```
dotnet run --project WinFormsDemo
```

Voir aussi

- Démarrage rapide : WinForms
- Rendu sans interface graphique
- Démos (Delphi)
- Présentation .NET
- Installation .NET
- Correspondance des classes
- Différences .NET par rapport à Delphi

Dépannage

Les problèmes ci-dessous sont ceux rencontrés le plus souvent avec sgcMaps, avec leur cause et leur solution.

La carte n'affiche aucune tuile

- Traitez **OnTileError**. Il est déclenché une fois pour chaque tuile en échec, avec la couche, les coordonnées de la tuile et l'erreur, telle que HTTP 403 Forbidden. Une tuile que le serveur n'a pas (HTTP 404) n'est pas une erreur : elle est seulement comptée dans **LastRenderMissingTiles**, comme une tuile absente d'un dossier.
- Lorsqu'une couche ne peut pas être ouverte du tout, la carte dessine la raison sur la vue, après le texte "Tile source not available". La plus courante est "Tile source directory not found" : la couche n'a pas de Provider et son TileSourceType est encore mtsFolder, la valeur par défaut, avec un TileSourcePath qui n'existe pas.
- Les tuiles depuis un serveur requièrent Delphi 2009 ou une version ultérieure, et le contrôle Lazarus n'affiche que les tuiles d'un dossier (voir Lazarus). MBTiles et MBTiles vectoriel requièrent Delphi XE7 ; sur les versions plus anciennes, la couche signale que le type de source de tuiles n'est pas disponible dans ce compilateur.
- De Delphi 2009 à XE7, les tuiles sont téléchargées avec Indy, et https requiert les bibliothèques OpenSSL 1.0.2 (libeay32.dll et ssleay32.dll) à côté de l'exécutable. Sans elles, l'erreur indique que les bibliothèques OpenSSL n'ont pas pu être chargées. À partir de Delphi XE8, la carte utilise THTTPClient et n'a besoin d'aucune DLL.
- Une tuile en échec n'est pas stockée, la carte la redemande donc au prochain affichage. OnTileError n'est pas déclenché deux fois pour la même tuile tant que les couches ne sont pas rouvertes.

```
procedure TForm3.sgcVCLMap1TileError(Sender: TObject; aLayer: TsgcMapLayer;
  aZoom, aX, aY: Integer; const aError: string);
begin
  // raised once per tile; a tile the server does not have (404) is no error
  memoLog.Lines.Add(Format('%s %d/%d/%d: %s', [aLayer.Name, aZoom, aX, aY,
    aError]));
end;

procedure TForm3.btnMissingClick(Sender: TObject);
begin
  // tiles that were not available yet in the last paint
  memoLog.Lines.Add(Format('Missing tiles: %d',
    [sgcVCLMap1.LastRenderMissingTiles]));
end;
```

Clés d'API

- Les fournisseurs de tuiles MapTiler, Thunderforest et Stadia exigent une clé d'API dans la propriété **ApiKey** de la couche. Sans clé, OnTileError signale "This tile provider requires an API key (ApiKey)" ; avec une mauvaise clé, le fournisseur répond HTTP 401 ou 403.
- Les services en ligne ne vérifient pas la clé avant d'envoyer une requête. L'erreur du fournisseur arrive dans **LastError** (méthodes synchrones) ou **OnError** (méthodes asynchrones), par exemple HTTP 403 avec l'explication du fournisseur.
- TsgcMapTimeZone avec GeoNames prend votre nom d'utilisateur GeoNames dans ApiKey, et non une clé.
- N'incluez jamais une clé dans une démo ou dans un code source public. Les démos de sgcMaps ont à la place un champ de saisie pour la clé.

Politiques d'utilisation et user agent

Les serveurs publics gratuits sont gérés par des bénévoles ou des sponsors, et chacun a une politique d'utilisation. Une application qui l'enfreint peut être bloquée sans préavis.

- **tile.openstreetmap.org** n'autorise que la consultation interactive : pas de téléchargement en masse, pas de préchargement de zones, un user agent qui identifie l'application, et une attribution visible. Les requêtes de

tuiles des contrôles de carte s'identifient comme sgcMaps/1.0. Pour une application en production, utilisez un fournisseur commercial, votre propre serveur de tuiles ou des tuiles hors ligne.
- **TsgcMapTileDownloader** n'a volontairement aucun URLTemplate par défaut. Faites-le pointer vers votre propre serveur ou vers un fournisseur dont les conditions autorisent les téléchargements en masse, jamais vers tile.openstreetmap.org, et gardez DelayMS supérieur à zéro (250 ms par défaut). Il s'identifie comme sgcMapsDownloader/1.0.
- Les services en ligne envoient sgcMaps/2.0 dans **UserAgent** par défaut. Définissez-le avec le nom de votre application : Nominatim exige un user agent qui identifie l'application.
- Les requêtes vers Nominatim et OpenTopoData sont espacées par le composant à une par seconde, pour l'ensemble du processus, comme le demandent leurs politiques. Un appel synchrone effectué depuis le thread principal peut donc attendre jusqu'à une seconde ; utilisez les méthodes Async dans une interface utilisateur.
- Le serveur OSRM par défaut de TsgcMapDirections est le serveur de démonstration public : usage léger uniquement, une requête par seconde, et le profil voiture seulement. Utilisez votre propre serveur OSRM ou un autre fournisseur pour la marche et le vélo, et appelez **ValidateRequest** ou **IsTravelModeSupported** pour vérifier un mode de déplacement auprès du fournisseur.

Les événements asynchrones n'arrivent jamais

Les méthodes Async des services exécutent la requête dans un thread et déclenchent l'événement de résultat, ou OnError, dans le thread principal avec TThread.Queue. Une application VCL ou FireMonkey livre ces événements depuis sa boucle de messages. Une application console, un service Windows ou un thread n'a pas de telle boucle, elle doit donc appeler **CheckSynchronize** jusqu'à l'arrivée de la réponse, et le composant doit rester vivant jusque-là. Les objets de résultat transmis à l'événement sont libérés au retour de l'événement, copiez donc ce dont vous avez besoin dans le gestionnaire.

```
program GeocodeConsole;

{$APPTYPE CONSOLE}

uses
  SysUtils, Classes,
  sgcMaps_Geocoding;

type
  TListener = class
  public
    Done: Boolean;
    procedure DoResult(Sender: TObject; aResults: TsgcMapGeocodingResults);
    procedure DoError(Sender: TObject; const aError: string);
  end;

procedure TListener.DoResult(Sender: TObject;
  aResults: TsgcMapGeocodingResults);
begin
  if aResults.Count > 0 then
    WriteLn(aResults[0].DisplayName)
  else
    WriteLn('No results');
  Done := True;
end;

procedure TListener.DoError(Sender: TObject; const aError: string);
begin
  WriteLn('Error: ', aError);
  Done := True;
end;

var
  oGeocoder: TsgcMapGeocoder;
  oListener: TListener;
begin
  oListener := TListener.Create;
  oGeocoder := TsgcMapGeocoder.Create(nil);
  try
    // identify your application, as the public servers ask
```

```
    oGeocoder.UserAgent := 'MyCompanyGeocoder/1.0';
    oGeocoder.OnGeocodeResult := oListener.DoResult;
    oGeocoder.OnError := oListener.DoError;
    oGeocoder.GeocodeAsync('Plaza Mayor, Madrid');
    // a console application has no message loop, so deliver the queued
    // events with CheckSynchronize until the answer arrives
    while not oListener.Done do
      CheckSynchronize(100);
  finally
    oGeocoder.Free;
    oListener.Free;
  end;
end.
```

Le **Timeout** des services et de la source de tuiles est appliqué avec THTTPClient à partir de Delphi 10.1 Berlin. Sous Delphi XE8 et Delphi 10 Seattle, il n'a aucun effet.

FireDAC SQLite

- Les fichiers MBTiles, les fichiers MBTiles vectoriels et l'index de TsgcMapOfflineGeocoder sont des bases SQLite lues avec FireDAC, à partir de Delphi XE7. Votre édition de Delphi doit inclure FireDAC avec le pilote SQLite, que requiert le package d'exécution de sgcMaps.
- Les unités créent elles-mêmes le lien vers le pilote FireDAC SQLite et ouvrent les fichiers en lecture seule, le formulaire n'a donc besoin d'aucun composant FireDAC. Lors du déploiement, incluez ce dont le pilote FireDAC SQLite a besoin sur chaque plateforme cible, comme le décrit la documentation FireDAC de votre version de Delphi.
- Un fichier manquant lève "MBTiles file not found" ou "Geocoder index not found" avec le chemin. Déployez les fichiers avec l'application et construisez les chemins à l'exécution, car un chemin relatif dépend du dossier courant.
- Un fichier MBTiles avec un autre schéma donne des tuiles vides et aucune erreur, vérifiez donc un nouveau fichier dans une démo d'abord.
- Chaque TsgcMapOfflineGeocoder a une connexion et ne doit être utilisé que par un seul thread à la fois. Donnez à chaque thread son propre composant. Contrairement au géocodeur en ligne, il lève une exception lorsque l'index est absent, au lieu de renvoyer False.

Dossiers de cache

- Une couche en ligne conserve ses tuiles dans **CacheFolder**. Lorsque CacheFolder est vide, elle utilise TileSourcePath, et lorsque les deux sont vides, il n'y a pas de cache disque, seulement un cache de 256 tuiles en mémoire.
- Les dossiers sont créés au besoin. Lorsque le dossier n'est pas accessible en écriture, les tuiles ne sont pas mises en cache et aucune erreur n'est levée : cela arrive sous Program Files, et dans le dossier de l'application sous Android et iOS. Utilisez un dossier de l'utilisateur, tel que TPath.GetCachePath sur mobile.
- Les fichiers du cache sont nommés uniquement par zoom, x et y. Donnez à chaque couche, fournisseur et réglage Retina un dossier distinct, sinon les tuiles d'une couche sont dessinées pour une autre.
- Les tuiles en cache n'expirent pas. Supprimez le dossier pour les télécharger à nouveau, par exemple après un changement de style chez le fournisseur.

Écrans à haute densité de pixels

Le contrôle VCL dessine ses tuiles à **Options.TileSize** pixels, 256 par défaut, et ne s'adapte pas de lui-même au DPI du moniteur. Sur un écran à haute densité de pixels, définissez TileSize à partir des pixels par pouce (de 64 à 1024) et définissez **Retina** sur les couches, pour que les fournisseurs qui le prennent en charge envoient des tuiles de 512 pixels qui restent nettes lorsqu'elles sont agrandies. Une couche vectorielle avec Retina dessine ses tuiles à 512 pixels. Les fournisseurs qui prennent en charge les tuiles retina sont listés dans Tuiles, couches et fournisseurs.

```
procedure TForm3.SetupHighDPI;
var
  oLayer: TsgcMapLayer;
begin
```

```
  oLayer := sgcVCLMap1.Layers.Add;
  oLayer.Name := 'CARTO Light';
  oLayer.Provider := mtpCartoLight;
  // request the 512 pixel (@2x) tiles of the provider
  oLayer.Retina := True;
  // a cache folder of its own: the cache file names hold only z, x and y
  oLayer.CacheFolder := ExtractFilePath(ParamStr(0)) + 'cache\carto@2x';
  // draw the map at the scale of the screen: 256 at 100%, 512 at 200%
  sgcVCLMap1.Options.TileSize := Round(256 * Screen.PixelsPerInch / 96);
end;
```

Localisation sur les appareils mobiles

- Sous Android, activez l'autorisation **Access fine location** dans les options du projet, et appelez **Request-Permission** de TsgcMapLocation (Delphi 10.3 et versions ultérieures) avant de définir Active à True. Définir Active à la conception démarre le capteur avant que l'utilisateur ait accordé l'autorisation.
- Sous iOS et macOS, ajoutez NSLocationWhenInUseUsageDescription aux informations de version du projet. Sous Windows, le service de localisation doit être activé dans les paramètres de confidentialité.
- Les erreurs sont "Location sensor not available", "Location sensor access denied or in error state" et "Location permission denied". Utilisez **SimulateLocation** ou **LoadTrack** pour tester sans capteur, comme le fait la démo FireMonkey.

Calcul d'itinéraires hors ligne

- **Open** de TsgcMapOfflineRouter lève "Invalid routing graph file", "Unsupported routing graph version" ou "Truncated routing graph file" lorsque GraphFile n'est pas un fichier .sgcgraph complet de cette version. Reconstruisez-le avec l'outil RouteGraphBuilder.
- "Waypoint n is not near a routable road" signifie que le point de passage est en dehors de la zone du graphe, ou loin de toute route du profil sélectionné. "No route found between waypoints" signifie que le graphe n'a aucune connexion entre eux, ce qui arrive à la limite d'un extrait.
- Le graphe entier est chargé en mémoire, choisissez donc le plus petit extrait qui couvre votre zone.
- Le lecteur OpenStreetMap prend en charge les fichiers .osm.pbf non compressés et compressés avec zlib, le format des services d'extraits habituels. Un fichier compressé avec LZMA, LZ4, Zstandard ou bzip2 lève une erreur.

Lazarus

Le contrôle Lazarus n'affiche que les tuiles d'un dossier, car il ne peut pas télécharger de tuiles, et il n'a ni rotation ni opacité. GeoJSON, GPX et KML ne sont pas compilés sous FPC. Voir Lazarus.

Introduction

sgcDashboard est un dashboard natif pour Delphi et C++Builder, pour VCL et FireMonkey. Un dashboard est une grille de tiles. Chaque tile est un contrôle enfant qui héberge tout autre contrôle déposé dans celui-ci, de sorte qu'un graphique, une carte, une visionneuse PDF, un aperçu de rapport, un mémo ou une liste peuvent cohabiter sur la même surface thématisée. Un contrôle carte KPI affiche un chiffre unique avec sa tendance et un sparkline.
sgcDashboard fait partie de la suite visuelle d'eSeGeCe. Ses tiles hébergent les visionneuses de **sgcCharts** (graphiques de données en direct), **sgcMaps** (cartes), **sgcPDF** (visionneuse PDF) et **sgcReports** (aperçu de rapport). Il est dessiné avec **sgcUICore**, le package d'exécution partagé qui fournit le canvas, le rasterizer, les polices, l'écrivain SVG et les thèmes clair et sombre. sgcDashboard requiert sgcUICore, qui s'installe avec lui. Les autres produits sont facultatifs : un tile ne les héberge que s'ils sont installés.
Une seule propriété **Theme** du dashboard fait basculer le dashboard, ses tiles et les contrôles hébergés entre clair et sombre. Une build Registered permet aussi à l'utilisateur de déplacer et de redimensionner les tiles à l'exécution et enregistre la disposition au format JSON, et la passerelle **sgcDashboardWS** alimente les cartes KPI depuis une connexion WebSocket, MQTT ou server-sent events de sgcWebSockets.

Éditions

sgcDashboard est livré en éditions Registered, Trial et Community. L'édition Community omet deux fonctionnalités : le déplacement et le redimensionnement des tiles à l'exécution avec SaveLayout et LoadLayout, et la passerelle de données en direct sgcDashboardWS. Voir Éditions.

Par où commencer

- Architecture : le moteur de disposition, les tiles, le canvas et le thème de sgcUICore, VCL et FMX.
- Plateformes : les versions de Delphi, C++Builder et les plateformes.
- Démarrage rapide VCL, Démarrage rapide FMX et Démarrage rapide C++Builder : un premier dashboard en quelques minutes.
- Composants : les composants dashboard, tile, carte KPI et source de la passerelle.
- Disposition, Thèmes, Persistance, Données en direct et Images KPI headless.
- Démos : les projets de démonstration pour Delphi et C++Builder.

Architecture

sgcDashboard est construit en couches. Chaque couche ne dépend que de celles situées en dessous, si bien que le moteur de disposition et le moteur de rendu KPI se compilent sans VCL ni FireMonkey.

Couche	Unités	Rôle
Moteur de disposition	sgcDashboard_Layout	TsgcDashboardLayout : une grille indépendante du framework qui transforme la cellule de chaque tile (Col, Row, ColSpan, RowSpan) en rectangle, avec Columns, RowHeight, Gap et Padding en pixels indépendants de l'appareil, l'auto-flow, la résolution des collisions et le calcul d'alignement pour le glisser et le redimensionnement. Voir Disposition.
Dashboard et tiles	sgcDashboard_VCL, sgcDashboard_FMX	TsgcVCLDashboard et TsgcFMXDashboard possèdent un moteur de disposition et y placent leurs tiles. TsgcVCLDashboardTile et TsgcFMXDashboardTile sont des contrôles enfants, comme un TTabSheet dans un TPageControl, et un contrôle déposé dans un tile est hébergé par celui-ci.
Carte KPI	sgcDashboard_KPI, sgcDashboard_KPI_VCL, sgcDashboard_KPI_FMX	TsgcDashboardKPI est le modèle et le moteur de rendu. TsgcVCLKPICard et TsgcFMXKPICard sont des contrôles légers qui possèdent un modèle et le dessinent.
sgcUICore	sgcUICore_Canvas, sgcUICore_VCL, sgcUICore_FMX, sgcUICore_Theme, sgcUICore_JSON	L'exécution partagée : le canvas vectoriel et le rasterizer sur lesquels chaque tile et chaque carte se dessinent, la base de contrôles VCL et FMX avec cadence d'images (MaxFPS) et demandes de redessin thread-safe, les jetons des thèmes clair et sombre et l'applicateur de thème, ainsi que le lecteur et l'écrivain JSON utilisés par la persistance de la disposition.
Passerelle de données en direct	sgcDashboardWS_Source, _WebSocket, _MQTT, _SSE	TsgcDashboardSource et ses descendants lisent une valeur dans chaque message JSON d'un client sgcWebSockets et l'envoient dans une carte KPI. La passerelle est un package distinct, sgcDashboardWS, qui requiert sgcDashboard et sgcWebSockets.

Les tiles sont des contrôles enfants

Un tile est un vrai contrôle à l'intérieur du dashboard. À la conception, le verbe "Add Tile" de l'éditeur de composant du dashboard crée un tile, et tout contrôle se dépose dans un tile depuis la palette. Déplacer ou redimensionner un tile dans le concepteur l'aligne sur la grille. À l'exécution, un tile est créé dans le code et son **Parent** est défini sur le dashboard. Le tile dessine une surface de carte avec une bordure, des coins arrondis et un en-tête facultatif avec son **Title**, et il laisse sa zone client aux contrôles qu'il héberge.

Dessin et thèmes

Les tiles et les cartes KPI se dessinent via le canvas sgcUICore, le même canvas que celui de sgcCharts, de sorte qu'un thème clair ou sombre a le même aspect sur les deux frameworks et dans la sortie PNG et SVG headless. Un tile ne dessine pas les contrôles qu'il héberge. Il leur transmet le thème par leurs propriétés publiées : voir Thèmes.

VCL et FMX

Les classes VCL et FMX ont les mêmes propriétés, méthodes et événements. Le dashboard VCL descend de TScrollingWinControl et le dashboard FMX de TVertScrollBox, si bien qu'une grille plus haute que la vue défile verticalement. Toutes les tailles de la disposition sont exprimées en pixels indépendants de l'appareil et sont mises à l'échelle par le PPI du moniteur sous VCL et par l'échelle de la fiche sous FMX.

Voir aussi

- Plateformes
- Disposition
- Thèmes

Éditions

sgcDashboard est construit à partir des mêmes sources pour chaque édition. L'édition ne change que les fonctionnalités qu'une build compile.

Édition	Contenu
Registered	Tout : les dashboards et tiles VCL et FMX, les cartes KPI et les images KPI headless, les thèmes, le déplacement et le redimensionnement des tiles à l'exécution avec SaveLayout et LoadLayout, et la passerelle de données en direct sgcDashboardWS.
Trial	Les mêmes fonctionnalités que Registered. Une build compilée avec SGC_TRIAL affiche un avis indiquant qu'il s'agit d'une version d'essai une fois la fenêtre de publication indiquée dans la build dépassée. Rien d'autre ne change.
Community	Les dashboards et tiles, la disposition en grille avec auto-flow, les cartes KPI avec sparklines et la sortie PNG et SVG headless, ainsi que les thèmes clair et sombre. Elle omet le déplacement et le redimensionnement à l'exécution avec SaveLayout et LoadLayout, ainsi que la passerelle sgcDashboardWS.

Ce que l'édition Community omet

L'édition Community se compile avec SGC_DASHBOARD_PERSIST non défini. Ces membres n'existent pas dans une build Community :

- **AllowDrag**, **AllowResize** et **OnLayoutChanged** sur le dashboard.
- **SaveLayout**, **LoadLayout**, **SaveLayoutToFile**, **LoadLayoutFromFile**, **SaveLayoutToStream** et **LoadLayoutFromStream** sur le dashboard.
- **Locked** sur le tile VCL. Sous FMX, Locked est la propriété FMX standard et reste disponible, mais il n'y a alors aucun glissement à bloquer.

L'édition Community est aussi livrée sans les unités et packages sgcDashboardWS, de sorte que TsgcDashboardSource_WebSocket, TsgcDashboardSource_MQTT et TsgcDashboardSource_SSE ne sont pas disponibles. Une carte KPI reste alimentable depuis le code avec **Value** et **AddValue**.
L'édition est un choix à la compilation dans sgcVer.inc (SGC_COMMUNITY, qui désactive aussi SGC_TRIAL), et non un commutateur d'exécution. Le code qui utilise l'un des membres ci-dessus ne se compile pas avec une build Community, voir Dépannage.

Voir aussi

- Persistance
- Données en direct
- Plateformes

Installation

sgcDashboard fait partie de la distribution sgcUI. Un seul installateur couvre les produits sgcUI (sgcUICore, sgcCharts, sgcPDF, sgcReports, sgcMaps, sgcDashboard, sgcGrid, sgcNavigation, sgcToolbar et sgcWizard) et n'installe que les produits inclus dans votre licence.

Installation avec l'installateur sgcUI

1. Lancez l'installateur sgcUI et connectez-vous avec le nom d'utilisateur et le mot de passe de votre compte eSeGeCe.
2. Choisissez les versions de Delphi et de C++Builder dans lesquelles installer.
3. Sur la page des composants, sélectionnez **sgcDashboard** et tout autre produit souhaité. Le serveur de licences renvoie les produits que vous possédez, et l'installateur affiche une case à cocher par produit. Les produits que votre licence n'inclut pas sont grisés. Une licence All-Access déverrouille tous les produits.
4. L'installateur copie les fichiers, puis compile les packages sur votre machine pour chaque IDE sélectionné, dans l'ordre des dépendances, et installe les packages de conception.

Ce qui est fourni avec sgcDashboard

- **sgcUICore**, la base commune des produits de mise en page et de sgcCharts. Il n'a pas de case à cocher propre. L'installateur l'installe automatiquement dès que sgcDashboard est sélectionné.
- **sgcDashboard pour sgcWebSockets**, listé sous **sgcWebSockets bridges**. Il installe les packages sgcDashboardWS avec les sources WebSocket, MQTT et SSE (voir Données en direct). L'installateur ne le propose que si sgcWebSockets est déjà installé sur la machine et que votre licence inclut sgcDashboard. Sinon, il est grisé.

Dossiers installés

Tous les produits s'installent dans un seul dossier, par défaut `Documents\esegece\sgcUI` (`sgcUI_Trial` pour la version d'essai et `sgcUI_Community` pour l'édition Community).

Dossier	Contenu
`Source`	Le code source de chaque produit installé. C'est l'unique chemin de bibliothèque de sgcUI.
`Packages`	Les packages de chaque produit installé, un jeu par produit et par version d'IDE.
`Lib<V>`	Les unités compilées pour chaque version d'IDE et chaque plateforme, par exemple `LibD13` .
`Demos\Dashboard`	Les démos de sgcDashboard. Voir Démos.

L'installateur ajoute `sgcUI\Source` au chemin de bibliothèque et au chemin de navigation de chaque IDE et plateforme sélectionnés, ainsi que le dossier des unités compilées. sgcUICore se trouve dans le même dossier `Source`, il n'a donc besoin d'aucun chemin propre.

Trial et Community

- **Trial** : l'installateur Trial de sgcUI installe tous les produits en mode d'essai, y compris sgcDashboard.
- **Community** : l'installateur Community gratuit de sgcUI inclut sgcDashboard, avec sgcCharts, sgcGrid, sgcNavigation, sgcToolbar et sgcWizard. Les passerelles sgcWebSockets ne font pas partie de l'installateur Community, et sgcPDF et sgcMaps ne font pas partie de Community. Voir Éditions pour ce que la version Community laisse de côté.

Installations autonomes précédentes

Si l'installateur trouve une ancienne installation autonome de sgcUICore, comme celle créée par l'installateur autonome de sgcCharts, il liste les chemins de bibliothèque et les packages de conception de cette installation sur la page **Previous installations**. Quand vous continuez, il les supprime de l'IDE avant de compiler les nouveaux packages, de sorte qu'il n'y ait jamais deux copies d'une unité dans le chemin.

Compiler les packages manuellement

Pour compiler vous-même les packages, ouvrez-les pour votre version d'IDE depuis `sgcUI\Packages`. `<V>` est le suffixe de version Delphi, par exemple `sgcDashboardD13.dpk`.

Package	Contenu
`sgcUICore<V>`	La base commune sgcUICore (exécution).
`sgcDashboard<V>` / `dclsgcDashboard<V>`	Les composants sgcDashboard (exécution / conception).
`sgcDashboardWS<V>` / `dclsgcDashboardWS<V>`	La passerelle sgcWebSockets (exécution / conception). Nécessite sgcWebSockets.

Compilez-les dans l'ordre des dépendances : `sgcUICore`, puis `sgcDashboard`, puis `dclsgcDashboard`, et enfin les packages de la passerelle. Installez ensuite les packages de conception `dcl*` et ajoutez `sgcUI\Source` au chemin de bibliothèque de votre version de Delphi (Tools > Options > Language > Delphi Options > Library > Library path). Pour C++Builder, ouvrez le projet `.cbproj` correspondant à votre version, par exemple `sgcDashboardC13.cbproj`, et compilez-le. Les en-têtes `.hpp` générés apparaissent à côté du package compilé.

.NET

Rien ne change pour .NET. sgcDashboard reste distribué sous forme de ses propres packages NuGet (`esegece.sgcDashboard` et ses packages compagnons). Voir l'installation .NET.

Plateformes

Cible	Prise en charge
Delphi	Delphi 7, 2007, 2009, 2010, XE à XE8, 10 à 10.4, 11, 12 et 13.
C++Builder	C++Builder 2007, 2010, XE à XE8, 10 à 10.4, 11, 12 et 13. Les packages génèrent les en-têtes .hpp et les fichiers .lib qu'un projet C++Builder inclut et lie.
VCL	TsgcVCLDashboard, TsgcVCLDashboardTile et TsgcVCLKPICard sous Windows, Win32 et Win64, toutes les versions de Delphi.
FireMonkey	TsgcFMXDashboard, TsgcFMXDashboardTile et TsgcFMXKPICard à partir de Delphi XE6 (SGC_DASHBOARD_FMX), sous Win32, Win64, macOS, iOS et Android.
Linux64	Les unités indépendantes du framework, le moteur de disposition (sgcDashboard_Layout) et le modèle KPI avec sa sortie PNG et SVG headless (sgcDashboard_KPI), à partir de Delphi 10.2 (SGC_DASHBOARD_LINUX). Les contrôles FMX sous Linux nécessitent FMX pour Linux (FMXLinux).
Passerelle de données en direct	Les packages sgcDashboardWS, là où les packages sgcWebSockets du même IDE sont installés.
.NET	Un port C# avec la même API pour WinForms, WPF et Avalonia, la sortie headless en PNG et SVG et la passerelle sgcWebSockets, sur .NET Framework 4.7.2 et .NET 8 et 9, voir Présentation de .NET.

Packages

sgcDashboard fournit un package d'exécution indépendant du framework et un package d'exécution par framework, de sorte qu'une application FireMonkey pour Android, iOS ou macOS ne lie jamais la VCL. Chaque nom de package se termine par le suffixe de l'IDE, par exemple sgcDashboardD13.dpk pour Delphi 13 et sgcDashboardC13.cbproj pour C++Builder 13.

Package	Type	Contenu	Requiert
sgcDashboard	Exécution, indépendant du framework	Chaque unité du produit qui n'utilise ni la VCL ni FireMonkey.	rtl, sgcUICore
sgcDashboardVCL	Exécution, VCL	sgcDashboard_VCL et sgcDashboard_KPI_VCL (TsgcVCLDashboard, TsgcVCLDashboardTile, TsgcVCLKPICard)	rtl, vcl, sgcUICore, sgcUICoreVCL, sgcDashboard
sgcDashboardFMX	Exécution, FireMonkey, à partir de XE6	sgcDashboard_FMX et sgcDashboard_KPI_FMX (TsgcFMXDashboard, TsgcFMXDashboardTile, TsgcFMXKPICard)	rtl, fmx, sgcUICore, sgcUICoreFMX, sgcDashboard
dclsgcDashboard	Conception, VCL	Enregistre les composants VCL dans la page de palette **SGC Dashboard** et ajoute le verbe "Add Tile" à l'éditeur de composant du dashboard.	designide, rtl, vcl, sgcDashboard, sgcDashboardVCL

dclsgcDashboardFMX	Conception, FireMonkey, à partir de XE6	Enregistre les composants FireMonkey dans la page de palette **SGC Dashboard**, avec le même verbe "Add Tile".	designide, rtl, fmx, sgcDashboard, sgcDashboardFMX, dclsgcDashboard
sgcDashboardWS, dclsgcDashboardWS	Exécution et conception, passerelle sgcWebSockets	La passerelle de données en direct, indépendante du framework.	rtl, sgcDashboard, sgcWebSockets

sgcUICore suit la même répartition : sgcUICore, sgcUICoreVCL et sgcUICoreFMX, sans package de conception. dclsgcDashboardFMX requiert dclsgcDashboard, installez donc d'abord dclsgcDashboard.

Versions de l'IDE et plateformes cibles

Delphi	sgcDashboard	sgcDashboardVCL	sgcDashboardFMX	dclsgcDashboard, dclsgcDashboardFMX
7, 2007, 2009, 2010, XE	Win32	Win32	Aucune	Win32
XE2 à XE5	Win32, Win64	Win32, Win64	Aucune	Win32
XE6 à 10.2	Win32, Win64	Win32, Win64	Win32, Win64	Win32
10.3	Win32, Win64, Linux64, OSX64	Win32, Win64	Win32, Win64, OSX64	Win32
10.4	Win32, Win64, Linux64, OSX64, Android, Android64, iOSDevice64	Win32, Win64	Win32, Win64, OSX64, Android, Android64, iOSDevice64	Win32
11	Win32, Win64, Linux64, OSX64, OSXARM64, Android, Android64, iOSDevice64, iOSSimARM64	Win32, Win64	Win32, Win64, OSX64, OSXARM64, Android, Android64, iOSDevice64, iOSSimARM64	Win32
12, 13	Comme 11	Win32, Win64	Comme 11	Win32, Win64

De XE6 à 10.2, le package FireMonkey est compilé pour Windows uniquement. Les packages de conception sont chargés par l'IDE, ils sont donc compilés pour la plateforme de l'IDE uniquement.

C++Builder	sgcDashboard, sgcDashboardVCL	sgcDashboardFMX	dclsgcDashboard, dclsgcDashboardFMX
2007, 2010, XE, XE2	Win32	Aucune	Win32 (dclsgcDashboard uniquement)
XE3 à XE5	Win32, Win64	Aucune	Win32 (dclsgcDashboard uniquement)
XE6 à 12	Win32, Win64	Win32, Win64	Win32

13	Win32, Win64, Win64x	Win32, Win64, Win64x	Win32, Win64x

Les projets C++Builder portent les mêmes noms avec le suffixe C, par exemple sgcDashboardC13.cbproj, sgcDashboardVCLC13.cbproj, sgcDashboardFMXC13.cbproj, dclsgcDashboardC13.cbproj et dclsgcDashboardFMXC13.cbproj. Les packages C++Builder sont compilés pour Windows uniquement. Les notes C++Builder qui s'appliquent à chaque produit sgcUI (packages d'exécution sous Win64, C++Builder 10.3 sous Win64, noms de fichiers des packages Win64x) se trouvent dans le sujet Plateformes et packages de l'aide de la suite sgcUI.

FireMonkey sur Android, iOS et macOS

Les contrôles FireMonkey (TsgcFMXDashboard, TsgcFMXDashboardTile et TsgcFMXKPICard) ciblent macOS à partir de Delphi 10.3, Android et iOS à partir de Delphi 10.4, et macOS ARM et le simulateur iOS à partir de Delphi 11. Dans un projet FireMonkey pour ces plateformes :

- Utilisez les contrôles FireMonkey. Les contrôles VCL et sgcDashboardVCL ne fonctionnent que sous Windows.
- Déposez le contrôle sur un formulaire, ou ajoutez son unité à la clause uses : sgcDashboard_FMX et sgcDashboard_KPI_FMX.
- Sélectionnez la plateforme cible dans la fenêtre Projets. L'installateur ajoute le dossier Source de sgcUI et les unités compilées de chaque plateforme (LibD<version>\<platform>) au chemin de bibliothèque de chaque plateforme qu'il compile. Sans l'installateur, ajoutez le dossier Source de sgcUI au chemin de bibliothèque de la plateforme cible (Outils, Options, Langage, Delphi, Bibliothèque).
- Les applications Android et iOS sont liées statiquement, aucun package d'exécution n'est donc déployé.

Linux64

Le package indépendant du framework sgcDashboard est compilé pour Linux64 à partir de Delphi 10.3. Une application Linux64, par exemple un programme console ou un serveur, utilise le moteur de disposition et le modèle KPI avec sa sortie PNG et SVG headless. Aucun package VCL ou FireMonkey n'est compilé pour Linux64, car FireMonkey sous Linux nécessite FMX pour Linux (FMXLinux), un module complémentaire tiers.

Capturer un formulaire VCL avec PaintTo

Pour capturer un formulaire VCL, ou un dashboard qu'il contient, dans un bitmap avec PaintTo, verrouillez le canvas du bitmap autour de l'appel. PaintTo dans un canvas non verrouillé peut perdre son contexte de périphérique lorsqu'un contrôle natif du formulaire traite un message pendant la capture, et le bitmap reste vide ou partiel. Un formulaire FireMonkey utilise MakeScreenshot à la place.

```
procedure CaptureForm(aForm: TForm; const aFileName: string);
var
  vBitmap: TBitmap;
begin
  vBitmap := TBitmap.Create;
  try
    vBitmap.SetSize(aForm.ClientWidth, aForm.ClientHeight);
    vBitmap.Canvas.Lock;
    try
      aForm.PaintTo(vBitmap.Canvas, 0, 0);
    finally
      vBitmap.Canvas.Unlock;
    end;
    vBitmap.SaveToFile(aFileName);
  finally
    vBitmap.Free;
  end;
end;
```

Voir aussi

- Éditions
- Architecture
- Dépannage

Démarrage rapide VCL

À la conception

- Déposez un **TsgcVCLDashboard** depuis la page de palette **SGC Dashboard** sur une fiche et définissez **Align** sur alClient.
- Faites un clic droit sur le dashboard et choisissez **Add Tile**. Répétez l'opération pour chaque tile. Un nouveau tile a Col et Row à -1, de sorte que l'auto-flow le place dans le premier emplacement libre de la grille. Définissez Col, Row, ColSpan et RowSpan dans l'inspecteur d'objets, ou déplacez et redimensionnez le tile dans le concepteur, pour lui donner une cellule fixe.
- Sélectionnez un tile et déposez-y un contrôle, par exemple un TsgcVCLChart de sgcCharts ou un TMemo, puis définissez son **Align** sur alClient.
- Déposez un **TsgcVCLKPICard** dans un autre tile, définissez son Align sur alClient et renseignez **Title**, **Value**, **Trend** et **TrendValue**. Définissez **ShowHeader** de ce tile sur False, la carte affichant déjà son propre titre.
- Définissez **Theme** du dashboard sur utDark. Le dashboard, les tiles, la carte KPI et les contrôles hébergés passent à la palette sombre.

Dans le code

Le même dashboard créé à l'exécution. Le programme se compile tel quel avec `sgcUI\Source` dans le chemin de recherche.

```
uses
  Forms, Classes, Controls, StdCtrls,
  sgcUICore_Theme, sgcDashboard_KPI, sgcDashboard_VCL, sgcDashboard_KPI_VCL;

constructor TfrmMain.Create(aOwner: TComponent);
var
  oTile: TsgcVCLDashboardTile;
  oCard: TsgcVCLKPICard;
  oMemo: TMemo;
begin
  inherited CreateNew(aOwner);
  Caption := 'sgcDashboard VCL Quick Start';
  Width := 900;
  Height := 560;

  // the dashboard: a 12 column grid that fills the form
  FDashboard := TsgcVCLDashboard.Create(Self);
  FDashboard.Parent := Self;
  FDashboard.Align := alClient;
  FDashboard.Columns := 12;
  FDashboard.RowHeight := 110;

  // a KPI tile at column 0, row 0, 3 columns wide and 2 rows high
  oTile := TsgcVCLDashboardTile.Create(Self);
  oTile.Name := 'tileRevenue';
  oTile.Parent := FDashboard;
  oTile.Col := 0;
  oTile.Row := 0;
  oTile.ColSpan := 3;
  oTile.RowSpan := 2;
  oTile.ShowHeader := False;

  oCard := TsgcVCLKPICard.Create(Self);
  oCard.Parent := oTile;
  oCard.Align := alClient;
  oCard.Title := 'Revenue';
  oCard.Value := '12,480 USD';
  oCard.Icon := '$';
```

```
  oCard.Trend := ktUp;
  oCard.TrendValue := '+4.2 %';
  oCard.FooterText := 'vs last week';
  oCard.AddValue(11200);
  oCard.AddValue(11900);
  oCard.AddValue(12480);

  // a log tile, Col and Row stay -1: auto-flow puts it in the first free slot
  oTile := TsgcVCLDashboardTile.Create(Self);
  oTile.Name := 'tileEvents';
  oTile.Parent := FDashboard;
  oTile.Title := 'Events';
  oTile.ColSpan := 6;
  oTile.RowSpan := 3;

  oMemo := TMemo.Create(Self);
  oMemo.Parent := oTile;
  oMemo.Align := alClient;
  oMemo.Lines.Add('Service started');

  // one property themes the dashboard, the tiles and the hosted controls
  FDashboard.Theme := utDark;
end;
```

Voir aussi

- TsgcVCLDashboard
- TsgcVCLDashboardTile
- TsgcVCLKPICard
- Disposition
- Thèmes

Démarrage rapide FMX

Les contrôles FireMonkey nécessitent Delphi XE6 ou une version ultérieure. Ils s'utilisent exactement comme ceux de la VCL : déposez un **TsgcFMXDashboard**, définissez **Align** sur Client, utilisez le verbe **Add Tile** et déposez des contrôles ou un **TsgcFMXKPICard** dans les tiles.

Dans le code

```
uses
  System.Classes, FMX.Forms, FMX.Types,
  sgcUICore_Theme, sgcDashboard_KPI, sgcDashboard_FMX, sgcDashboard_KPI_FMX;

constructor TfrmMain.Create(aOwner: TComponent);
var
  oTile: TsgcFMXDashboardTile;
  oCard: TsgcFMXKPICard;
begin
  inherited CreateNew(aOwner);
  Caption := 'sgcDashboard FMX Quick Start';
  Width := 900;
  Height := 560;

  FDashboard := TsgcFMXDashboard.Create(Self);
  FDashboard.Parent := Self;
  FDashboard.Align := TAlignLayout.Client;
  FDashboard.Columns := 12;
  FDashboard.RowHeight := 110;

  oTile := TsgcFMXDashboardTile.Create(Self);
  oTile.Parent := FDashboard;
  oTile.ColSpan := 4;
  oTile.RowSpan := 2;
  oTile.ShowHeader := False;

  oCard := TsgcFMXKPICard.Create(Self);
  oCard.Parent := oTile;
  oCard.Align := TAlignLayout.Client;
  oCard.Title := 'Active users';
  oCard.Value := '1,284';
  oCard.Trend := ktDown;
  oCard.TrendValue := '-2.1 %';
  oCard.AddValue(1310);
  oCard.AddValue(1284);

  FDashboard.Theme := utDark;
end;
```

Voir aussi

- TsgcFMXDashboard
- TsgcFMXDashboardTile
- TsgcFMXKPICard
- Plateformes

Démarrage rapide C++Builder

Installez les packages sgcDashboard pour votre version de C++Builder, ajoutez `sgcUI\Source` et le dossier des unités compilées, par exemple `sgcUI\LibD13\win32` (`sgcUI\LibD13\win64` pour le 64 bits), aux chemins d'inclusion et de bibliothèque du projet, puis utilisez les composants de la page de palette **SGC Dashboard** exactement comme dans Delphi. Le verbe **Add Tile** fonctionne de la même façon. L'installateur ajoute les deux chemins à l'IDE pour vous. Les en-têtes .hpp et les fichiers .lib générés de sgcDashboard et de sgcUICore se trouvent dans ce seul dossier, sgcUICore n'a donc besoin d'aucun chemin propre.

Dans le code

Voir aussi

- Démarrage rapide VCL
- Démos
- Plateformes

Démos

Les démos se trouvent dans le dossier **Demos** de l'installation, avec un projet Delphi sous Demos\Delphi et la même démo pour C++Builder sous Demos\CBuilder. Chaque dossier de démo contient un readme.html qui explique ce que fait la démo et comment elle fonctionne. Les démos sont des applications VCL pour Win32 et Win64.

Démo	Composants	Ce qu'elle montre
Basic	TsgcVCLDashboard, TsgcVCLDashboardTile, TsgcVCLKPICard	Un dashboard de six tiles sur une grille de 12 colonnes : quatre cartes KPI en direct animées par un timer, un journal d'événements et une liste de serveurs, avec un commutateur de thème clair et sombre qui thématise aussi le TMemo et le TListBox.
Suite	TsgcVCLDashboard, TsgcVCLKPICard, TsgcVCLChart, TsgcVCLMap, TsgcPDFViewerControl, TsgcReportPreviewVCL	Un dashboard dont les tiles hébergent les autres produits VCL d'eSeGeCe : un graphique linéaire sgcCharts en direct, une carte sgcMaps, une visionneuse sgcPDF et un aperçu sgcReports, à côté de deux cartes KPI, le tout suivant le même thème. Elle nécessite les sources de sgcCharts, sgcMaps, sgcPDF et sgcReports dans le chemin de recherche.
LiveKPI	TsgcVCLKPICard, TsgcDashboardSource_WebSocket, TsgcWebSocketClient, TsgcWebSocketServer	Trois cartes KPI liées à un flux WebSocket via TsgcDashboardSource_WebSocket. Un TsgcWebSocketServer intégré à la démo diffuse un message JSON toutes les 500 ms, et chaque source lit son propre ValuePath dans ce message. Elle nécessite les sources de sgcWebSockets et ne fait pas partie de l'édition Community.
SaveLayout	TsgcVCLDashboard, TsgcVCLDashboardTile, TsgcVCLKPICard	Un dashboard que l'utilisateur réorganise à l'exécution avec AllowDrag et AllowResize, une disposition enregistrée dans layout.json et rechargée depuis celui-ci avec SaveLayoutToFile et LoadLayoutFromFile, et OnLayoutChanged. Elle ne fait pas partie de l'édition Community.

Voir aussi

- Démarrage rapide VCL
- Persistance
- Données en direct

TsgcVCLDashboard

Unité : sgcDashboard_VCL
TsgcVCLDashboard est le dashboard VCL, uniquement pour Windows. Il dispose ses enfants TsgcVCLDashboardTile sur une grille de **Columns** colonnes et de lignes de **RowHeight** pixels, avec **Gap** entre les tiles et **Padding** autour d'eux, et défile verticalement lorsque les tiles sont plus hauts que le contrôle. Les tiles sont ajoutés à la conception avec le verbe **Add Tile** de l'éditeur de composant. Voir Disposition pour le placement des cellules.

Propriétés

Propriété	Type	Description
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom. Peint l'arrière-plan du dashboard et est suivi par chaque tile dont ParentTheme vaut True, qui le transmet aux contrôles qu'il héberge. Voir Thèmes.
Columns	Integer	Nombre de colonnes de la grille. 12 par défaut.
RowHeight	Integer	Hauteur d'une ligne de la grille, en pixels indépendants de l'appareil. 120 par défaut.
Gap	Integer	Espace entre deux tiles, en pixels indépendants de l'appareil. 12 par défaut.
Padding	Integer	Espace entre la bordure du dashboard et les tiles, en pixels indépendants de l'appareil. 12 par défaut.
AutoFlow	Boolean	Lorsque True (par défaut), un tile dont Col ou Row vaut -1 occupe le premier emplacement libre de la grille. Lorsque False, -1 est lu comme 0.
AllowDrag	Boolean	Déplacement d'un tile à l'exécution par son en-tête. False par défaut. Absent de l'édition Community.
AllowResize	Boolean	Redimensionnement d'un tile à l'exécution par la poignée de son coin inférieur droit. False par défaut. Absent de l'édition Community.
Layout	TsgcDashboardLayout (public, lecture seule)	Le moteur de disposition en grille du dashboard. Voir Disposition.

Les membres habituels de TScrollingWinControl sont également publiés : **Align**, **Anchors**, **Constraints**, **Enabled**, **ParentShowHint**, **PopupMenu**, **ShowHint**, **TabOrder**, **TabStop**, **Visible**, **OnClick**, **OnDblClick**, **OnMouseDown**, **OnMouseMove**, **OnMouseUp** et **OnResize**.

Méthodes

Méthode	Retourne	Description

Arrange		Place immédiatement les tiles sur la grille. Le dashboard se réorganise lorsqu'un tile ou une propriété de la grille change, un appel est donc rarement nécessaire.
ApplyTheme		Applique de nouveau le thème de chaque tile aux contrôles qu'il héberge, par exemple après l'ajout ou la modification de contrôles dans le code.
SaveLayout	string	Les métriques de la grille ainsi que la cellule et la visibilité de chaque tile nommé au format JSON. Absent de l'édition Community.
LoadLayout(const aJSON: string)		Applique un texte produit par SaveLayout. Les tiles sont associés par Name, les noms inconnus sont ignorés et les tiles absents du texte conservent leurs cellules. Une seule réorganisation, puis OnLayoutChanged. Absent de l'édition Community.
SaveLayoutToFile(const aFileName: string)		Écrit le résultat de SaveLayout dans un fichier, en UTF-8 sans BOM. Absent de l'édition Community.
LoadLayoutFromFile(const aFileName: string)		Lit un fichier de disposition écrit par SaveLayoutToFile. Un BOM est ignoré. Absent de l'édition Community.
SaveLayoutToStream(aStream: TStream)		Écrit le résultat de SaveLayout dans un flux, en UTF-8 sans BOM. Absent de l'édition Community.
LoadLayoutFromStream(aStream: TStream)		Lit une disposition depuis un flux. Un BOM est ignoré. Absent de l'édition Community.

Événements

Événement	Type	Description
OnLayoutChanged	TNotifyEvent	Un glissement ou un redimensionnement à l'exécution a modifié la cellule d'un tile, ou LoadLayout a été appliqué. Absent de l'édition Community.

Exemple

```
constructor TfrmMain.Create(aOwner: TComponent);
var
  oTile: TsgcVCLDashboardTile;
  oCard: TsgcVCLKPICard;
  oMemo: TMemo;
begin
  inherited CreateNew(aOwner);
  Caption := 'sgcDashboard VCL Quick Start';
  Width := 900;
  Height := 560;

  // the dashboard: a 12 column grid that fills the form
  FDashboard := TsgcVCLDashboard.Create(Self);
  FDashboard.Parent := Self;
  FDashboard.Align := alClient;
  FDashboard.Columns := 12;
  FDashboard.RowHeight := 110;

  // a KPI tile at column 0, row 0, 3 columns wide and 2 rows high
  oTile := TsgcVCLDashboardTile.Create(Self);
  oTile.Name := 'tileRevenue';
```

```
  oTile.Parent := FDashboard;
  oTile.Col := 0;
  oTile.Row := 0;
  oTile.ColSpan := 3;
  oTile.RowSpan := 2;
  oTile.ShowHeader := False;

  oCard := TsgcVCLKPICard.Create(Self);
  oCard.Parent := oTile;
  oCard.Align := alClient;
  oCard.Title := 'Revenue';
  oCard.Value := '12,480 USD';
  oCard.Icon := '$';
  oCard.Trend := ktUp;
  oCard.TrendValue := '+4.2 %';
  oCard.FooterText := 'vs last week';
  oCard.AddValue(11200);
  oCard.AddValue(11900);
  oCard.AddValue(12480);

  // a log tile, Col and Row stay -1: auto-flow puts it in the first free slot
  oTile := TsgcVCLDashboardTile.Create(Self);
  oTile.Name := 'tileEvents';
  oTile.Parent := FDashboard;
  oTile.Title := 'Events';
  oTile.ColSpan := 6;
  oTile.RowSpan := 3;

  oMemo := TMemo.Create(Self);
  oMemo.Parent := oTile;
  oMemo.Align := alClient;
  oMemo.Lines.Add('Service started');

  // one property themes the dashboard, the tiles and the hosted controls
  FDashboard.Theme := utDark;
end;
```

Voir aussi

- TsgcVCLDashboardTile
- TsgcFMXDashboard
- Disposition
- Persistance
- Thèmes

TsgcVCLDashboardTile

Unité : sgcDashboard_VCL
TsgcVCLDashboardTile est un tile d'un TsgcVCLDashboard. Il dessine une carte thématisée avec une bordure, des coins arrondis et un en-tête facultatif, et héberge tout contrôle VCL déposé dans celui-ci : un graphique, une carte, une visionneuse PDF, un aperçu de rapport, une carte KPI ou un contrôle standard. Le tile transmet son thème effectif aux contrôles qu'il héberge. Le tile ne dessine jamais lui-même les contrôles hébergés.

Propriétés

Propriété	Type	Description
Title	string	Texte de l'en-tête du tile.
ShowHeader	Boolean	Dessine la bande d'en-tête avec Title. True par défaut. La zone client du tile commence sous l'en-tête.
Col	Integer	Colonne de la grille du tile, à partir de zéro. -1 par défaut, placé par l'auto-flow.
Row	Integer	Ligne de la grille du tile, à partir de zéro. -1 par défaut, placé par l'auto-flow.
ColSpan	Integer	Nombre de colonnes couvertes par le tile. 3 par défaut, limité de 1 à Columns.
RowSpan	Integer	Nombre de lignes couvertes par le tile. 2 par défaut, au moins 1.
ParentTheme	Boolean	Le tile et les contrôles qu'il héberge suivent le Theme du dashboard. True par défaut.
Theme	TsgcUITheme	Thème du tile. Le définir désactive ParentTheme. En dehors d'un dashboard, un tile avec ParentTheme utilise utLight.
Locked	Boolean	Le tile n'est jamais déplacé ni redimensionné à l'exécution, même si le dashboard l'autorise. False par défaut. Absent de l'édition Community.
Dashboard	TsgcVCLDashboard (public, lecture seule)	Le dashboard auquel appartient le tile, nil lorsque son parent n'est pas un dashboard.

Les membres habituels des contrôles sont également publiés : **Constraints**, **Enabled**, **Hint**, **ParentShowHint**, **PopupMenu**, **ShowHint**, **Visible**, **OnClick**, **OnContextPopup**, **OnDblClick**, **OnMouseDown**, **OnMouseMove**, **OnMouseUp** et **OnResize**. Un tile masqué (Visible False) laisse sa cellule libre.

Méthodes

Méthode	Retourne	Description
GetCell	TsgcDashboardCell	La cellule demandée du tile, un enregistrement avec Col, Row, ColSpan et RowSpan.
SetCell(const aCell: TsgcDashboardCell)		Définit Col, Row, ColSpan et RowSpan en une fois, avec une seule réorganisation du dashboard.

Événements

Événement	Type	Description
OnResize	TNotifyEvent	Le tile a été redimensionné, par une réorganisation du dashboard ou par un redimensionnement à l'exécution.

Exemple

Les tiles suivent le thème du dashboard, sauf s'ils définissent le leur :

```
procedure ApplyThemes(aDashboard: TsgcVCLDashboard;
  aTileLight: TsgcVCLDashboardTile);
begin
  // every tile with ParentTheme = True follows the dashboard
  aDashboard.Theme := utDark;
  // this tile keeps its own theme: setting Theme clears ParentTheme
  aTileLight.Theme := utLight;
  // back to the dashboard theme
  aTileLight.ParentTheme := True;
  // pushes the theme again, after controls were changed in code
  aDashboard.ApplyTheme;
end;
```

Voir aussi

- TsgcVCLDashboard
- TsgcFMXDashboardTile
- Disposition
- Thèmes

TsgcFMXDashboard

Unité : sgcDashboard_FMX
TsgcFMXDashboard est le dashboard FireMonkey, disponible à partir de Delphi XE6. Il possède les mêmes membres de grille, de thème et de persistance que TsgcVCLDashboard. Il descend de TVertScrollBox, si bien que les tiles défilent verticalement, et son **Padding** est un Integer en pixels indépendants de l'appareil qui remplace le Padding de type TBounds des contrôles FMX. Le verbe **Add Tile** de l'éditeur de composant est disponible dans l'IDE Windows.

Propriétés

Propriété	Type	Description
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom. Peint l'arrière-plan du dashboard et est suivi par chaque tile dont ParentTheme vaut True, qui le transmet aux contrôles qu'il héberge. Voir Thèmes.
Columns	Integer	Nombre de colonnes de la grille. 12 par défaut.
RowHeight	Integer	Hauteur d'une ligne de la grille, en pixels indépendants de l'appareil. 120 par défaut.
Gap	Integer	Espace entre deux tiles, en pixels indépendants de l'appareil. 12 par défaut.
Padding	Integer	Espace entre la bordure du dashboard et les tiles, en pixels indépendants de l'appareil. 12 par défaut.
AutoFlow	Boolean	Lorsque True (par défaut), un tile dont Col ou Row vaut -1 occupe le premier emplacement libre de la grille. Lorsque False, -1 est lu comme 0.
AllowDrag	Boolean	Déplacement d'un tile à l'exécution par son en-tête. False par défaut. Absent de l'édition Community.
AllowResize	Boolean	Redimensionnement d'un tile à l'exécution par la poignée de son coin inférieur droit. False par défaut. Absent de l'édition Community.
Layout	TsgcDashboardLayout (public, lecture seule)	Le moteur de disposition en grille du dashboard. Voir Disposition.

Méthodes

Méthode	Retourne	Description
Arrange		Place immédiatement les tiles sur la grille. Le dashboard se réorganise lorsqu'un tile ou une propriété de la grille change, un appel est donc rarement nécessaire.

ApplyTheme		Applique de nouveau le thème de chaque tile aux contrôles qu'il héberge, par exemple après l'ajout ou la modification de contrôles dans le code.
SaveLayout	string	Les métriques de la grille ainsi que la cellule et la visibilité de chaque tile nommé au format JSON. Absent de l'édition Community.
LoadLayout(const aJSON: string)		Applique un texte produit par SaveLayout. Les tiles sont associés par Name, les noms inconnus sont ignorés et les tiles absents du texte conservent leurs cellules. Une seule réorganisation, puis OnLayoutChanged. Absent de l'édition Community.
SaveLayoutToFile(const aFileName: string)		Écrit le résultat de SaveLayout dans un fichier, en UTF-8 sans BOM. Absent de l'édition Community.
LoadLayoutFromFile(const aFileName: string)		Lit un fichier de disposition écrit par SaveLayoutToFile. Un BOM est ignoré. Absent de l'édition Community.
SaveLayoutToStream(aStream: TStream)		Écrit le résultat de SaveLayout dans un flux, en UTF-8 sans BOM. Absent de l'édition Community.
LoadLayoutFromStream(aStream: TStream)		Lit une disposition depuis un flux. Un BOM est ignoré. Absent de l'édition Community.

Événements

Événement	Type	Description
OnLayoutChanged	TNotifyEvent	Un glissement ou un redimensionnement à l'exécution a modifié la cellule d'un tile, ou LoadLayout a été appliqué. Absent de l'édition Community.

Exemple

```
constructor TfrmMain.Create(aOwner: TComponent);
var
  oTile: TsgcFMXDashboardTile;
  oCard: TsgcFMXKPICard;
begin
  inherited CreateNew(aOwner);
  Caption := 'sgcDashboard FMX Quick Start';
  Width := 900;
  Height := 560;

  FDashboard := TsgcFMXDashboard.Create(Self);
  FDashboard.Parent := Self;
  FDashboard.Align := TAlignLayout.Client;
  FDashboard.Columns := 12;
  FDashboard.RowHeight := 110;

  oTile := TsgcFMXDashboardTile.Create(Self);
  oTile.Parent := FDashboard;
  oTile.ColSpan := 4;
  oTile.RowSpan := 2;
  oTile.ShowHeader := False;

  oCard := TsgcFMXKPICard.Create(Self);
  oCard.Parent := oTile;
  oCard.Align := TAlignLayout.Client;
```

```
  oCard.Title := 'Active users';
  oCard.Value := '1,284';
  oCard.Trend := ktDown;
  oCard.TrendValue := '-2.1 %';
  oCard.AddValue(1310);
  oCard.AddValue(1284);

  FDashboard.Theme := utDark;
end;
```

Voir aussi

- TsgcFMXDashboardTile
- TsgcVCLDashboard
- Démarrage rapide FMX

TsgcFMXDashboardTile

Unité : sgcDashboard_FMX
TsgcFMXDashboardTile est un tile d'un TsgcFMXDashboard, avec les mêmes membres que TsgcVCLDashboard-Tile. Un tile dont la propriété FMX standard **Locked** vaut True ne démarre jamais de glissement ni de redimension-nement à l'exécution.

Propriétés

Propriété	Type	Description
Title	string	Texte de l'en-tête du tile.
ShowHea-der	Boolean	Dessine la bande d'en-tête avec Title. True par défaut. La zone client du tile commence sous l'en-tête.
Col	Integer	Colonne de la grille du tile, à partir de zéro. -1 par défaut, placé par l'auto-flow.
Row	Integer	Ligne de la grille du tile, à partir de zéro. -1 par défaut, placé par l'auto-flow.
ColSpan	Integer	Nombre de colonnes couvertes par le tile. 3 par défaut, limité de 1 à Columns.
RowSpan	Integer	Nombre de lignes couvertes par le tile. 2 par défaut, au moins 1.
Parent-Theme	Boolean	Le tile et les contrôles qu'il héberge suivent le Theme du dashboard. True par défaut.
Theme	TsgcUI-Theme	Thème du tile. Le définir désactive ParentTheme. En dehors d'un dashboard, un tile avec ParentTheme utilise utLight.
Locked	Boolean	La propriété FMX standard. Un tile verrouillé n'est jamais déplacé ni redimen-sionné à l'exécution. False par défaut.
Dashboard	TsgcFMX-Dashboard (public, lec-ture seule)	Le dashboard auquel appartient le tile, nil lorsque son parent n'est pas un da-shboard.

Les membres habituels des contrôles FMX sont également publiés, notamment **Anchors**, **Enabled**, **Hint**, **Margins**, **Opacity**, **PopupMenu**, **Visible**, ainsi que les événements de souris, de dessin et de redimensionnement.

Méthodes

Méthode	Retourne	Description
GetCell	TsgcDashboard-Cell	La cellule demandée du tile, un enregistrement avec Col, Row, ColSpan et RowSpan.
SetCell(const aCell: TsgcDashboard-Cell)		Définit Col, Row, ColSpan et RowSpan en une fois, avec une seule réorganisation du dashboard.

Événements

Événement	Type	Description
OnResize	TNotifyEvent	Le tile a été redimensionné.
OnResized	TNotifyEvent	Le redimensionnement du tile est terminé.

Exemple

```
constructor TfrmMain.Create(aOwner: TComponent);
var
  oTile: TsgcFMXDashboardTile;
  oCard: TsgcFMXKPICard;
begin
  inherited CreateNew(aOwner);
  Caption := 'sgcDashboard FMX Quick Start';
  Width := 900;
  Height := 560;

  FDashboard := TsgcFMXDashboard.Create(Self);
  FDashboard.Parent := Self;
  FDashboard.Align := TAlignLayout.Client;
  FDashboard.Columns := 12;
  FDashboard.RowHeight := 110;

  oTile := TsgcFMXDashboardTile.Create(Self);
  oTile.Parent := FDashboard;
  oTile.ColSpan := 4;
  oTile.RowSpan := 2;
  oTile.ShowHeader := False;

  oCard := TsgcFMXKPICard.Create(Self);
  oCard.Parent := oTile;
  oCard.Align := TAlignLayout.Client;
  oCard.Title := 'Active users';
  oCard.Value := '1,284';
  oCard.Trend := ktDown;
  oCard.TrendValue := '-2.1 %';
  oCard.AddValue(1310);
  oCard.AddValue(1284);

  FDashboard.Theme := utDark;
end;
```

Voir aussi

- TsgcFMXDashboard
- TsgcVCLDashboardTile
- Thèmes

TsgcDashboardKPI

Unité : sgcDashboard_KPI
TsgcDashboardKPI est le modèle et le moteur de rendu, indépendants du framework, d'une carte KPI : un titre, une grande valeur, une icône facultative, une flèche de tendance avec son texte, un texte de pied et un sparkline des dernières valeurs MaxPoints. **Render** dessine la carte sur n'importe quel canvas sgcUICore, de sorte que le même code dessine TsgcVCLKPICard, TsgcFMXKPICard et les images PNG et SVG headless. Il se compile sur toutes les plateformes, Linux64 compris.
AddValue et ClearValues peuvent être appelés depuis n'importe quel thread, les valeurs étant protégées par un verrou, et OnChange est alors appelé depuis ce thread. Toutes les autres propriétés sont destinées à être définies depuis le thread principal.

Propriétés

Propriété	Type	Description
Title	string	Légende en haut de la carte.
Value	string	Le grand chiffre de la carte, déjà mis en forme.
Icon	string	Un court glyphe de texte dessiné dans un cercle d'accent, par exemple '$' ou '%'. Vide, aucune icône n'est dessinée.
Trend	TsgcDash-boardKPITrend	ktNone (par défaut), ktUp, ktDown ou ktNeutral. Dessine la flèche de tendance, verte pour la hausse et rouge pour la baisse.
TrendValue	string	Texte à côté de la flèche de tendance, par exemple '+4.2 %'.
FooterText	string	Petit texte en bas de la carte.
AccentColor	TsgcUIColor	Couleur $AARRGGBB du cercle de l'icône et du sparkline. 0 (par défaut) utilise la couleur d'accent du thème.
Theme	TsgcUITheme	utLight (par défaut) ou utDark. Dans un tile de dashboard, c'est le tile qui le définit.
MaxPoints	Integer	Nombre de valeurs conservées par le sparkline. 50 par défaut. La valeur la plus ancienne est supprimée à l'arrivée d'une nouvelle valeur.
ValueCount	Integer (public, lecture seule)	Nombre de valeurs du sparkline.
Values[Index: Integer]	Double (public, lecture seule)	Une valeur du sparkline, la plus ancienne en premier.

Méthodes

Méthode	Retourne	Description
AddValue(aValue: Double)		Ajoute une valeur au sparkline. Thread-safe. Les valeurs NaN et infinies sont ignorées.
ClearValues		Supprime toutes les valeurs du sparkline.

GetValueArray	TsgcDashboardKPIValues	Une copie des valeurs du sparkline, la plus ancienne en premier.
Render(aCanvas: TsgcUICanvas; aWidth, aHeight: Integer; aScale: Double = 1)		Dessine la carte sur aCanvas, en aWidth par aHeight pixels de l'appareil. aScale est le facteur HiDPI appliqué à toutes les tailles.
SaveToPNG(const aFileName: string; aWidth, aHeight: Integer; aScale: Double = 1)		Rend la carte dans un fichier PNG. aWidth et aHeight sont en pixels logiques, l'image mesure Round(aWidth * aScale) par Round(aHeight * aScale) pixels. Une surcharge écrit dans un TStream.
SaveToSVG(const aFileName: string; aWidth, aHeight: Integer; aScale: Double = 1)		Rend la carte dans un fichier SVG. Une surcharge écrit dans un TStream.
Assign(Source: TPersistent)		Copie les propriétés et les valeurs du sparkline d'un autre TsgcDashboardKPI.

Événements

Événement	Type	Description
OnChange	TNotifyEvent	Une propriété ou le sparkline a changé. Appelé dans le thread qui a effectué la modification.

Exemple

```
uses
  SysUtils,
  sgcUICore_Theme, sgcDashboard_KPI;

procedure SaveKPIImages;
var
  oKPI: TsgcDashboardKPI;
begin
  oKPI := TsgcDashboardKPI.Create;
  try
    oKPI.Title := 'Orders';
    oKPI.Value := '842';
    oKPI.Icon := '#';
    oKPI.Trend := ktUp;
    oKPI.TrendValue := '+12';
    oKPI.FooterText := 'today';
    oKPI.Theme := utDark;
    oKPI.AddValue(790);
    oKPI.AddValue(815);
    oKPI.AddValue(842);
    // 320 x 160 logical pixels, a 640 x 320 image at scale 2
    oKPI.SaveToPNG('orders.png', 320, 160, 2);
    oKPI.SaveToSVG('orders.svg', 320, 160);
  finally
    oKPI.Free;
  end;
end;
```

Voir aussi

- Images KPI headless
- TsgcVCLKPICard
- TsgcFMXKPICard

TsgcVCLKPICard

Unité : sgcDashboard_KPI_VCL
TsgcVCLKPICard est la carte KPI VCL, uniquement pour Windows. Le contrôle possède un TsgcDashboardKPI, publie ses propriétés et le dessine via la base de contrôles VCL de sgcUICore. Une modification ne fait que planifier une nouvelle image, de sorte que AddValue peut être appelé depuis n'importe quel thread et que la carte se redessine dans le thread principal, au plus MaxFPS fois par seconde.

Propriétés

Propriété	Type	Description
Title	string	Légende en haut de la carte.
Value	string	Le grand chiffre de la carte, déjà mis en forme.
Icon	string	Un court glyphe de texte dessiné dans un cercle d'accent, par exemple '$' ou '%'. Vide, aucune icône n'est dessinée.
Trend	TsgcDashboardKPITrend	ktNone (par défaut), ktUp, ktDown ou ktNeutral. Dessine la flèche de tendance, verte pour la hausse et rouge pour la baisse.
TrendValue	string	Texte à côté de la flèche de tendance, par exemple '+4.2 %'.
FooterText	string	Petit texte en bas de la carte.
AccentColor	TsgcUIColor	Couleur $AARRGGBB du cercle de l'icône et du sparkline. 0 (par défaut) utilise la couleur d'accent du thème.
Theme	TsgcUITheme	utLight (par défaut) ou utDark. Dans un tile de dashboard, c'est le tile qui le définit.
MaxPoints	Integer	Nombre de valeurs conservées par le sparkline. 50 par défaut. La valeur la plus ancienne est supprimée à l'arrivée d'une nouvelle valeur.
KPI	TsgcDashboardKPI (public, lecture seule)	Le modèle de la carte, utilisable aussi pour une sortie PNG et SVG headless à la taille de votre choix.
MaxFPS	Integer (public)	Nombre maximal d'images par seconde du contrôle, hérité de la base de contrôles sgcUICore.
FrameCount	Cardinal (public, lecture seule)	Nombre d'images rendues depuis la création du contrôle.

Les membres habituels des contrôles sont également publiés : **Align**, **Anchors**, **Color**, **Constraints**, **Enabled**, **ParentColor**, **ParentShowHint**, **PopupMenu**, **ShowHint**, **Visible** et les événements de clic, de souris et de redimensionnement.

Méthodes

Méthode	Retourne	Description

AddValue(aValue: Double)		Ajoute une valeur au sparkline. Thread-safe. Les valeurs NaN et infinies sont ignorées.
ClearValues		Supprime toutes les valeurs du sparkline.
ContentChanged		Planifie une nouvelle image. Thread-safe. La carte l'appelle elle-même à chaque modification.

Événements

Événement	Type	Description
OnClick	TNotifyEvent	La carte a été cliquée.
OnResize	TNotifyEvent	La carte a été redimensionnée.

Exemple

Mise à jour d'une carte avec une nouvelle mesure :

```
procedure UpdateCard(aCard: TsgcVCLKPICard; aOld, aNew: Double);
begin
  aCard.Value := FormatFloat('#,##0', aNew);
  // AddValue is thread-safe, the card repaints on the main thread
  aCard.AddValue(aNew);
  if aNew > aOld then
    aCard.Trend := ktUp
  else if aNew < aOld then
    aCard.Trend := ktDown
  else
    aCard.Trend := ktNeutral;
  aCard.TrendValue := FormatFloat('+0.0;-0.0', aNew - aOld);
end;
```

Voir aussi

- TsgcDashboardKPI
- TsgcFMXKPICard
- Données en direct

TsgcFMXKPICard

Unité : sgcDashboard_KPI_FMX
TsgcFMXKPICard est la carte KPI FireMonkey, disponible à partir de Delphi XE6. Elle possède les mêmes membres que TsgcVCLKPICard et dessine le même modèle TsgcDashboardKPI.

Propriétés

Propriété	Type	Description
Title	string	Légende en haut de la carte.
Value	string	Le grand chiffre de la carte, déjà mis en forme.
Icon	string	Un court glyphe de texte dessiné dans un cercle d'accent, par exemple '$' ou '%'. Vide, aucune icône n'est dessinée.
Trend	TsgcDash-boardKPITrend	ktNone (par défaut), ktUp, ktDown ou ktNeutral. Dessine la flèche de tendance, verte pour la hausse et rouge pour la baisse.
TrendVa-lue	string	Texte à côté de la flèche de tendance, par exemple '+4.2 %'.
Footer-Text	string	Petit texte en bas de la carte.
Accent-Color	TsgcUIColor	Couleur $AARRGGBB du cercle de l'icône et du sparkline. 0 (par défaut) utilise la couleur d'accent du thème.
Theme	TsgcUITheme	utLight (par défaut) ou utDark. Dans un tile de dashboard, c'est le tile qui le définit.
Max-Points	Integer	Nombre de valeurs conservées par le sparkline. 50 par défaut. La valeur la plus ancienne est supprimée à l'arrivée d'une nouvelle valeur.
KPI	TsgcDash-boardKPI (public, lecture seule)	Le modèle de la carte, utilisable aussi pour une sortie PNG et SVG headless.
MaxFPS	Integer (public)	Nombre maximal d'images par seconde du contrôle, hérité de la base de contrôles sgcUICore.

Les membres habituels des contrôles FMX sont également publiés, notamment **Align**, **Anchors**, **Margins**, **Opacity**, **RotationAngle**, **Scale**, **Visible**, ainsi que les événements de souris, de dessin et de redimensionnement.

Méthodes

Méthode	Retourne	Description
AddValue(aValue: Double)		Ajoute une valeur au sparkline. Thread-safe. Les valeurs NaN et infinies sont ignorées.
ClearValues		Supprime toutes les valeurs du sparkline.

Événements

Événement	Type	Description
OnClick	TNotifyEvent	La carte a été cliquée.
OnResize	TNotifyEvent	La carte a été redimensionnée.

Exemple

```
procedure ConfigureCard(aCard: TsgcFMXKPICard);
begin
  aCard.Title := 'CPU';
  aCard.Icon := '%';
  aCard.FooterText := 'last 60 samples';
  // $AARRGGBB, 0 uses the accent colour of the theme
  aCard.AccentColor := $FF0078D4;
  aCard.MaxPoints := 60;
  aCard.ClearValues;
end;
```

Voir aussi

- TsgcDashboardKPI
- TsgcVCLKPICard
- Démarrage rapide FMX

TsgcDashboardSource

Unité : sgcDashboardWS_Source
TsgcDashboardSource est la classe de base des sources de la passerelle sgcWebSockets qui alimentent une carte KPI : TsgcDashboardSource_WebSocket, TsgcDashboardSource_MQTT et TsgcDashboardSource_SSE. Chaque message reçu par un descendant est lu comme du JSON : le nombre situé à **ValuePath** est ajouté au sparkline de la carte et devient sa Value, mise en forme avec **ValueFormat**. Le Trend de la carte est déduit de la valeur précédente de la même source : plus élevée, ktUp, plus basse, ktDown, égale, ktNeutral. La passerelle ne fait pas partie de l'édition Community. Elle nécessite les packages sgcDashboardWS et sgcWebSockets.
La gestion des threads est expliquée dans Données en direct.

Propriétés

Propriété	Type	Description
Active	Boolean	Lorsque False, la source ignore tous les messages et rétablit le gestionnaire précédent du client. True par défaut.
Card	TComponent	La carte KPI à alimenter : un TsgcVCLKPICard ou un TsgcFMXKPICard, tout composant qui implémente IsgcDashboardKPIHost. Les autres composants sont rejetés.
ValuePath	string	Chemin JSON du nombre dans chaque message, par exemple 'price' ou 'data.items[0].value'. Un nombre ou une chaîne numérique est accepté, avec '.' comme séparateur décimal quels que soient les paramètres régionaux. Vide, le message entier est lu comme le nombre.
ValueFormat	string	Chaîne de format du texte Value de la carte, transmise à Format avec le nombre. '%.2f' par défaut. La sortie utilise le séparateur décimal des paramètres régionaux de l'application.
TitlePath	string	Lorsqu'il est défini, copie cette valeur JSON dans le Title de la carte.
TrendValuePath	string	Lorsqu'il est défini, copie cette valeur JSON dans le TrendValue de la carte.

Événements

Événement	Type	Description
OnValue	TsgcDashboardSourceValueEvent	procedure(Sender: TObject; aValue: Double). Une valeur a été lue et la carte a été mise à jour. Thread principal sous Delphi 2010 et versions ultérieures.

Exemple

```
procedure TFeedHandler.DoValue(Sender: TObject; aValue: Double);
begin
  // main thread on Delphi 2010 and later, after the card was updated
  if aValue > 100 then
```

```
    WriteLn('price above 100: ', FormatFloat('0.00', aValue));
end;
```

Voir aussi

- Données en direct
- TsgcVCLKPICard

TsgcDashboardSource_WebSocket

Unité : sgcDashboardWS_Source_WebSocket
TsgcDashboardSource_WebSocket alimente une carte KPI à partir des messages texte d'un TsgcWebSocketClient. Tant que la source est Active et que Client est affecté, à l'exécution uniquement, la source installe son propre gestionnaire OnMessage sur le client et conserve le gestionnaire qui s'y trouvait auparavant : chaque message texte est d'abord lu dans la carte, puis transmis à ce gestionnaire précédent, si bien que le gestionnaire de l'application se déclenche toujours. Définir Active sur False, vider Client ou détruire la source rétablit le gestionnaire précédent, sauf si l'application l'a remplacé entre-temps. La passerelle ne fait pas partie de l'édition Community. Elle nécessite les packages sgcDashboardWS et sgcWebSockets.

Propriétés

Propriété	Type	Description
Client	TsgcWebSocketClient	Le client WebSocket dont les messages texte sont lus.
Active	Boolean	Lorsque False, la source ignore tous les messages et rétablit le gestionnaire précédent du client. True par défaut.
Card	TComponent	La carte KPI à alimenter : un TsgcVCLKPICard ou un TsgcFMXKPICard, tout composant qui implémente IsgcDashboardKPIHost. Les autres composants sont rejetés.
ValuePath	string	Chemin JSON du nombre dans chaque message, par exemple 'price' ou 'data.items[0].value'. Un nombre ou une chaîne numérique est accepté, avec '.' comme séparateur décimal quels que soient les paramètres régionaux. Vide, le message entier est lu comme le nombre.
ValueFormat	string	Chaîne de format du texte Value de la carte, transmise à Format avec le nombre. '%.2f' par défaut. La sortie utilise le séparateur décimal des paramètres régionaux de l'application.
TitlePath	string	Lorsqu'il est défini, copie cette valeur JSON dans le Title de la carte.
TrendValuePath	string	Lorsqu'il est défini, copie cette valeur JSON dans le TrendValue de la carte.

Méthodes

Méthode	Retourne	Description
ProcessMessage(const aText: string)	Boolean	Lit un message comme s'il provenait de Client. False lorsqu'il n'est pas du JSON valide ou que ValuePath n'est pas un nombre. Utile pour les tests et pour les messages qui arrivent par une autre voie.

Événements

Événement	Type	Description
OnValue	TsgcDashboardSourceValueEvent	procedure(Sender: TObject; aValue: Double). Une valeur a été lue et la carte a été mise à jour. Thread principal sous Delphi 2010 et versions ultérieures.

Exemple

```
procedure BindWebSocket;
begin
  WSClient := TsgcWebSocketClient.Create(nil);
  WSClient.URL := 'ws://127.0.0.1:5416';

  // messages look like {"price": 101.25, "change": "+0.4 %"}
  SourceWS := TsgcDashboardSource_WebSocket.Create(nil);
  SourceWS.Client := WSClient;
  SourceWS.Card := CardPrice;
  SourceWS.ValuePath := 'price';
  SourceWS.TrendValuePath := 'change';
  SourceWS.ValueFormat := '%.2f USD';

  WSClient.Active := True;
end;
```

Voir aussi

- TsgcDashboardSource
- TsgcDashboardSource_MQTT
- Données en direct

TsgcDashboardSource_MQTT

Unité : sgcDashboardWS_Source_MQTT
TsgcDashboardSource_MQTT alimente une carte KPI à partir des publications d'un TsgcWSPClient_MQTT. Tant que la source est Active et que Client est affecté, à l'exécution uniquement, la source installe son propre gestionnaire OnMQTTPublish sur le client et conserve le gestionnaire qui s'y trouvait auparavant : chaque publication est d'abord lue dans la carte, puis transmise à ce gestionnaire précédent, si bien que le gestionnaire de l'application se déclenche toujours. Définir Active sur False, vider Client ou détruire la source rétablit le gestionnaire précédent, sauf si l'application l'a remplacé entre-temps. Une application qui affecte Client.OnMQTTPublish après le rattachement de la source doit l'affecter avant de définir Client, ou basculer Active, pour conserver les deux.
Topic sélectionne les publications lues, avec les jokers MQTT + (un niveau) et # (ce niveau et tous les niveaux inférieurs). Un Topic vide lit toutes les publications. La source ne s'abonne pas, c'est l'application qui s'abonne via le client MQTT. Une charge utile qui est un simple nombre se lit avec un ValuePath vide. La passerelle ne fait pas partie de l'édition Community. Elle nécessite les packages sgcDashboardWS et sgcWebSockets.

Propriétés

Propriété	Type	Description
Client	TsgcWSPClient_MQTT	Le client MQTT dont les publications sont lues.
Topic	string	Filtre de topic MQTT des publications à lire. Vide, toutes les publications sont lues.
Active	Boolean	Lorsque False, la source ignore tous les messages et rétablit le gestionnaire précédent du client. True par défaut.
Card	TComponent	La carte KPI à alimenter : un TsgcVCLKPICard ou un TsgcFMXKPICard, tout composant qui implémente IsgcDashboardKPIHost. Les autres composants sont rejetés.
ValuePath	string	Chemin JSON du nombre dans chaque message, par exemple 'price' ou 'data.items[0].value'. Un nombre ou une chaîne numérique est accepté, avec '.' comme séparateur décimal quels que soient les paramètres régionaux. Vide, le message entier est lu comme le nombre.
ValueFormat	string	Chaîne de format du texte Value de la carte, transmise à Format avec le nombre. '%.2f' par défaut. La sortie utilise le séparateur décimal des paramètres régionaux de l'application.
TitlePath	string	Lorsqu'il est défini, copie cette valeur JSON dans le Title de la carte.
TrendValuePath	string	Lorsqu'il est défini, copie cette valeur JSON dans le TrendValue de la carte.

Méthodes

Méthode	Retourne	Description

ProcessPublish(const aTopic, aText: string)	Boolean	Lit une publication comme si elle provenait de Client. False lorsque le topic ne correspond pas à Topic ou que la charge utile ne peut pas être lue.

Événements

Événement	Type	Description
OnValue	TsgcDashboardSourceValueEvent	procedure(Sender: TObject; aValue: Double). Une valeur a été lue et la carte a été mise à jour. Thread principal sous Delphi 2010 et versions ultérieures.

Exemple

```
procedure BindMQTT;
begin
  MQTT := TsgcWSPClient_MQTT.Create(nil);
  MQTT.Client := WSClient;

  // payloads look like {"sensor": "hall", "value": 21.5}
  SourceMQTT := TsgcDashboardSource_MQTT.Create(nil);
  SourceMQTT.Client := MQTT;
  SourceMQTT.Card := CardTemp;
  SourceMQTT.Topic := 'sensors/+/temperature';
  SourceMQTT.ValuePath := 'value';
  SourceMQTT.ValueFormat := '%.1f C';
  // the source does not subscribe: call MQTT.Subscribe in OnMQTTConnect
end;
```

Voir aussi

- TsgcDashboardSource
- TsgcDashboardSource_WebSocket
- Données en direct

TsgcDashboardSource_SSE

Unité : sgcDashboardWS_Source_SSE
TsgcDashboardSource_SSE alimente une carte KPI à partir des server-sent events d'un TsgcHTTP1Client. L'application démarre le flux avec **Client.GetSSE(URL)**, qui bloque jusqu'à la fin du flux et qu'on appelle donc généralement depuis un thread de travail. Tant que la source est Active et que Client est affecté, à l'exécution uniquement, la source installe son propre gestionnaire OnSSEMessage sur le client et conserve le gestionnaire qui s'y trouvait auparavant : chaque fragment est d'abord lu dans la carte, puis transmis à ce gestionnaire précédent, si bien que le gestionnaire de l'application se déclenche toujours. Définir Active sur False, vider Client ou détruire la source rétablit le gestionnaire précédent, sauf si l'application l'a remplacé entre-temps. Le gestionnaire de l'application peut toujours annuler le flux.
OnSSEMessage fournit les octets bruts du flux à mesure qu'ils arrivent, si bien qu'un fragment peut contenir plusieurs événements ou une partie d'un seul. La source met les fragments en mémoire tampon, les découpe en événements à chaque ligne vide (fins de ligne LF, CRLF ou CR), joint avec LF les lignes data: d'un événement, ignore les commentaires et les autres champs, et lit les données de chaque événement via ValuePath. Un événement sans données est ignoré. La passerelle ne fait pas partie de l'édition Community. Elle nécessite les packages sgcDashboardWS et sgcWebSockets.

Propriétés

Propriété	Type	Description
Client	TsgcHTTP1Client	Le client HTTP dont les server-sent events sont lus.
Active	Boolean	Lorsque False, la source ignore tous les messages et rétablit le gestionnaire précédent du client. True par défaut.
Card	TComponent	La carte KPI à alimenter : un TsgcVCLKPICard ou un TsgcFMXKPICard, tout composant qui implémente IsgcDashboardKPIHost. Les autres composants sont rejetés.
ValuePath	string	Chemin JSON du nombre dans chaque message, par exemple 'price' ou 'data.items[0].value'. Un nombre ou une chaîne numérique est accepté, avec '.' comme séparateur décimal quels que soient les paramètres régionaux. Vide, le message entier est lu comme le nombre.
ValueFormat	string	Chaîne de format du texte Value de la carte, transmise à Format avec le nombre. '%.2f' par défaut. La sortie utilise le séparateur décimal des paramètres régionaux de l'application.
TitlePath	string	Lorsqu'il est défini, copie cette valeur JSON dans le Title de la carte.
TrendValuePath	string	Lorsqu'il est défini, copie cette valeur JSON dans le TrendValue de la carte.

Méthodes

Méthode	Retourne	Description
ProcessChunk(const aText: string)	Integer	Lit les événements complets d'un fragment de flux comme s'il provenait de Client. Un événement incomplet attend le fragment suivant. Re-

		tourne le nombre d'événements dont les données ont donné une valeur.
ResetStream		Abandonne un événement incomplet mis en mémoire tampon, par exemple avant le démarrage d'un nouveau flux.

Événements

Événement	Type	Description
OnValue	TsgcDashboardSourceValueEvent	procedure(Sender: TObject; aValue: Double). Une valeur a été lue et la carte a été mise à jour. Thread principal sous Delphi 2010 et versions ultérieures.

Exemple

```
procedure BindSSE;
begin
  HTTP := TsgcHTTP1Client.Create(nil);

  // events look like  data: {"cpu": {"load": 37.5}}
  SourceSSE := TsgcDashboardSource_SSE.Create(nil);
  SourceSSE.Client := HTTP;
  SourceSSE.Card := CardLoad;
  SourceSSE.ValuePath := 'cpu.load';
  SourceSSE.ValueFormat := '%.0f %%';
  // GetSSE blocks until the stream ends, call it from a worker thread:
  // HTTP.GetSSE('https://www.example.com/stream');
end;
```

Voir aussi

- TsgcDashboardSource
- TsgcDashboardSource_WebSocket
- Données en direct

Disposition

Le dashboard place ses tiles sur une grille. La grille et les cellules sont identiques sous VCL et FMX, et elles sont calculées par le moteur de disposition indépendant du framework TsgcDashboardLayout (unité sgcDashboard_Layout), exposé par la propriété **Layout** du dashboard.

Métriques de la grille

Propriété	Par défaut	Signification
Columns	12	Nombre de colonnes. La largeur d'une colonne est la largeur du dashboard, moins Padding des deux côtés et les espaces, divisée par Columns.
RowHeight	120	Hauteur d'une ligne en pixels indépendants de l'appareil. Un dashboard a autant de lignes que ses tiles en ont besoin, et défile verticalement.
Gap	12	Espace entre deux tiles, horizontalement et verticalement.
Padding	12	Espace entre la bordure du dashboard et la grille.
AutoFlow	True	Place les tiles dont Col ou Row vaut -1 dans le premier emplacement libre.

Toutes les tailles sont en pixels indépendants de l'appareil. Le dashboard VCL les met à l'échelle par le PPI du moniteur divisé par 96, le dashboard FMX par l'échelle de la fiche.

Cellules des tiles

Chaque tile demande une cellule avec **Col**, **Row**, **ColSpan** et **RowSpan**. Col et Row commencent à zéro. Un nouveau tile a Col et Row à -1 et une étendue de 3 colonnes sur 2 lignes. Avant de placer un tile, le moteur limite sa cellule : ColSpan entre 1 et Columns, RowSpan à 1 au minimum, et Col de sorte que le tile ne dépasse pas la dernière colonne.

Règle de placement et de collision

- **Passe 1, cellules explicites.** Les tiles dont Col et Row sont tous deux définis sont placés dans l'ordre des contrôles. Lorsque la cellule chevauche un tile déjà placé, le tile est poussé vers le bas d'une ligne à la fois jusqu'à ce que sa cellule soit libre. Ses colonnes ne changent jamais.
- **Passe 2, auto-flow.** Tant que AutoFlow vaut True, les tiles dont Col ou Row vaut -1 occupent le premier emplacement libre, en parcourant les lignes depuis le haut et les colonnes depuis la gauche. Une coordonnée donnée restreint le parcours : un tile avec Col 6 et Row -1 va à la première ligne libre de la colonne 6.
- Lorsque AutoFlow vaut False, un Col ou un Row de -1 est lu comme 0 et le tile est placé lors de la passe 1.
- Un tile masqué (Visible False) n'occupe aucune cellule.

La cellule demandée reste dans Col, Row, ColSpan et RowSpan. La cellule placée, après limitation et résolution des collisions, se trouve dans **Layout.PlacedCell[Index]** et son rectangle dans **Layout.Rect[Index]**.

Exemple

```
constructor TfrmMain.Create(aOwner: TComponent);
var
  oTile: TsgcVCLDashboardTile;
  oCard: TsgcVCLKPICard;
  oMemo: TMemo;
```

```
begin
  inherited CreateNew(aOwner);
  Caption := 'sgcDashboard VCL Quick Start';
  Width := 900;
  Height := 560;

  // the dashboard: a 12 column grid that fills the form
  FDashboard := TsgcVCLDashboard.Create(Self);
  FDashboard.Parent := Self;
  FDashboard.Align := alClient;
  FDashboard.Columns := 12;
  FDashboard.RowHeight := 110;

  // a KPI tile at column 0, row 0, 3 columns wide and 2 rows high
  oTile := TsgcVCLDashboardTile.Create(Self);
  oTile.Name := 'tileRevenue';
  oTile.Parent := FDashboard;
  oTile.Col := 0;
  oTile.Row := 0;
  oTile.ColSpan := 3;
  oTile.RowSpan := 2;
  oTile.ShowHeader := False;

  oCard := TsgcVCLKPICard.Create(Self);
  oCard.Parent := oTile;
  oCard.Align := alClient;
  oCard.Title := 'Revenue';
  oCard.Value := '12,480 USD';
  oCard.Icon := '$';
  oCard.Trend := ktUp;
  oCard.TrendValue := '+4.2 %';
  oCard.FooterText := 'vs last week';
  oCard.AddValue(11200);
  oCard.AddValue(11900);
  oCard.AddValue(12480);

  // a log tile, Col and Row stay -1: auto-flow puts it in the first free slot
  oTile := TsgcVCLDashboardTile.Create(Self);
  oTile.Name := 'tileEvents';
  oTile.Parent := FDashboard;
  oTile.Title := 'Events';
  oTile.ColSpan := 6;
  oTile.RowSpan := 3;

  oMemo := TMemo.Create(Self);
  oMemo.Parent := oTile;
  oMemo.Align := alClient;
  oMemo.Lines.Add('Service started');

  // one property themes the dashboard, the tiles and the hosted controls
  FDashboard.Theme := utDark;
end;
```

Voir aussi

- TsgcVCLDashboard
- TsgcVCLDashboardTile
- Persistance

Thèmes

La propriété **Theme** du dashboard, de type TsgcUITheme de sgcUICore_Theme, sélectionne **utLight** (par défaut), **utDark** ou **utCustom**. Le dashboard peint son arrière-plan avec le thème, et chaque tile dont **ParentTheme** vaut True le suit. Un tile peut définir son propre **Theme**, ce qui désactive ParentTheme. Le thème est appliqué lorsqu'il change et lorsqu'un contrôle est inséré dans un tile.

Ce que l'applicateur de thème définit sur les contrôles hébergés

Un tile transmet son thème à chaque contrôle qu'il héberge par les propriétés publiées que le contrôle expose, sans dépendance à la compilation envers le produit hébergé. La première règle qui correspond l'emporte :

1. Une propriété publiée modifiable **Theme** de type TsgcUITheme est définie. Cela couvre TsgcVCLKPICard et TsgcFMXKPICard.
2. Sinon, un objet publié **Chart** possédant un tel Theme est thématisé : **Chart.Theme** d'un TsgcVCLChart ou TsgcFMXChart de sgcCharts.
3. Sinon, une propriété publiée modifiable **BackgroundColor** reçoit la couleur de surface du thème, par exemple la visionneuse sgcPDF.
4. Sinon, une propriété publiée modifiable **Color** reçoit la couleur de surface, par exemple la carte sgcMaps, un TMemo ou un TListBox.

Lorsque la règle 3 ou 4 est atteinte, la couleur de texte du thème est également définie afin que le texte reste lisible sur une surface sombre : **Font.Color** sous VCL, et **TextSettings.FontColor** ou un **FontColor** publié sous FMX. Un objet publié **Viewer** d'un contrôle hébergé est thématisé selon les mêmes règles, sur un niveau de profondeur.
Avec **utCustom**, l'applicateur ne touche à aucun contrôle hébergé, de sorte que l'application garde la maîtrise complète de leurs couleurs. Appelez **ApplyTheme** sur le dashboard pour transmettre de nouveau le thème après la modification de contrôles dans le code.

Limites connues

- L'aperçu sgcReports ne publie aucune propriété de couleur, il conserve donc ses propres couleurs dans les deux thèmes.
- Un contrôle qui ne publie aucune des propriétés ci-dessus n'est pas modifié.
- Seuls les contrôles enfants directs d'un tile sont thématisés, ainsi que leur objet Viewer. Les contrôles imbriqués dans un panneau du tile conservent leurs couleurs.

Exemple

```
procedure ApplyThemes(aDashboard: TsgcVCLDashboard;
  aTileLight: TsgcVCLDashboardTile);
begin
  // every tile with ParentTheme = True follows the dashboard
  aDashboard.Theme := utDark;
  // this tile keeps its own theme: setting Theme clears ParentTheme
  aTileLight.Theme := utLight;
  // back to the dashboard theme
  aTileLight.ParentTheme := True;
  // pushes the theme again, after controls were changed in code
  aDashboard.ApplyTheme;
end;
```

Voir aussi

- TsgcVCLDashboardTile
- TsgcVCLKPICard
- Démos

Persistance

Une build Registered ou Trial permet à l'utilisateur de réorganiser les tiles à l'exécution et enregistre l'organisation au format JSON. Ces membres sont compilés avec SGC_DASHBOARD_PERSIST et n'existent pas dans l'édition Community.

Glisser et redimensionner à l'exécution

- **AllowDrag** : un appui sur le bouton gauche de la souris sur l'en-tête d'un tile le déplace. Un tile dont ShowHeader vaut False ne peut pas être déplacé.
- **AllowResize** : une poignée est dessinée dans le coin inférieur droit de chaque tile, et la faire glisser redimensionne le tile.
- Pendant le déplacement de la souris, la cellule cible est dessinée sous les tiles et le tile s'aligne sur la grille. Échap annule le glissement.
- **Locked** : un tile verrouillé n'est jamais déplacé ni redimensionné. Sous FMX, il s'agit de la propriété Locked standard.
- **OnLayoutChanged** se déclenche après qu'un glissement ou un redimensionnement a modifié une cellule, et après LoadLayout.

```
procedure SetupDashboard(aDashboard: TsgcVCLDashboard;
  aTileLocked: TsgcVCLDashboardTile; aOnChanged: TNotifyEvent);
begin
  // the user moves a tile by its header and resizes it by its grip
  aDashboard.AllowDrag := True;
  aDashboard.AllowResize := True;
  // this tile never moves
  aTileLocked.Locked := True;
  aDashboard.OnLayoutChanged := aOnChanged;
end;
```

SaveLayout et LoadLayout

SaveLayout retourne les métriques de la grille et, pour chaque tile ayant un Name, sa cellule et sa visibilité. **LoadLayout** applique un tel texte : les tiles sont associés par Name, les noms inconnus sont ignorés et les tiles absents du texte conservent leurs cellules. Les variantes fichier et flux utilisent l'UTF-8 sans BOM et ignorent un BOM au chargement.

```
procedure SaveUserLayout(aDashboard: TsgcVCLDashboard);
begin
  aDashboard.SaveLayoutToFile(ExtractFilePath(ParamStr(0)) + 'layout.json');
end;

procedure LoadUserLayout(aDashboard: TsgcVCLDashboard);
var
  vFile: string;
begin
  vFile := ExtractFilePath(ParamStr(0)) + 'layout.json';
  if FileExists(vFile) then
    aDashboard.LoadLayoutFromFile(vFile);
end;
```

Format JSON

Voici la sortie de SaveLayout pour un dashboard de deux tiles nommés, mise en forme ici pour la lecture. SaveLayout l'écrit sur une seule ligne.

```
{
  "version":1,
  "columns":12,
```

```
  "rowHeight":120,
  "gap":12,
  "padding":12,
  "tiles": [
    {"name":"tileSales","col":0,"row":0,"colSpan":6,"rowSpan":2,"visible":true},
    {"name":"tileLog","col":6,"row":0,"colSpan":6,"rowSpan":3,"visible":true}
  ]
}
```

Clé	Signification
version	Version du format, 1.
columns, rowHeight, gap, padding	Les métriques de la grille du dashboard.
tiles	Un objet par tile nommé : name, col, row, colSpan, rowSpan et visible.

Voir aussi

- TsgcVCLDashboard
- Disposition
- Démos
- Éditions

Données en direct

Une carte KPI est alimentée depuis le code avec **Value**, **Trend** et **AddValue**, ou depuis une connexion en direct via la passerelle sgcDashboardWS. Les sources de la passerelle lisent un nombre dans chaque message JSON d'un client sgcWebSockets et l'envoient dans une carte. La passerelle ne fait pas partie de l'édition Community.

Source	Client	Lit
TsgcDashboardSource_WebSocket	TsgcWebSocketClient	Chaque message texte.
TsgcDashboardSource_MQTT	TsgcWSPClient_MQTT	Chaque publication dont le topic correspond à Topic.
TsgcDashboardSource_SSE	TsgcHTTP1Client	Les données de chaque server-sent event.

Chemins et format

- **ValuePath** est le chemin JSON du nombre, par exemple price ou data.items[0].value. Un nombre ou une chaîne numérique est accepté, avec . comme séparateur décimal quels que soient les paramètres régionaux. Un ValuePath vide lit le message entier comme le nombre.
- **ValueFormat** met le nombre en forme pour le texte Value de la carte avec Format. La valeur par défaut est %.2f. Le texte utilise le séparateur décimal des paramètres régionaux de l'application.
- **TitlePath** et **TrendValuePath**, lorsqu'ils sont définis, copient cette valeur JSON dans le Title et le TrendValue de la carte.
- Le Trend de la carte suit la valeur précédente de la même source : plus élevée, ktUp, plus basse, ktDown, égale, ktNeutral.
- Un message qui n'est pas du JSON valide, ou dont le ValuePath n'est pas un nombre, est ignoré.

Gestion des threads

- Le nombre est ajouté immédiatement au sparkline, dans le thread où arrive le message. TsgcDashboardKPI.AddValue est thread-safe.
- Sous Delphi 2010 et versions ultérieures, les mises à jour de texte (Value, Title, TrendValue, Trend) sont appliquées dans le thread principal via TThread.Queue, suivies de OnValue. Le client peut utiliser n'importe quel mode NotifyEvents.
- Les applications console et les services n'ont pas de boucle de messages qui exécute la file, ils doivent donc appeler CheckSynchronize.
- Delphi 7 à 2009 n'ont pas de TThread.Queue. Dans ce cas, la mise à jour et OnValue s'exécutent dans le thread où le message est arrivé, et un gestionnaire OnValue doit se synchroniser lui-même avant de toucher à un contrôle.

Exemple

```
uses
  SysUtils, Classes,
  sgcWebSocket, sgcWebSocket_Protocols, sgcHTTP,
  sgcDashboard_KPI_VCL, sgcDashboardWS_Source_WebSocket,
  sgcDashboardWS_Source_MQTT, sgcDashboardWS_Source_SSE;

procedure BindWebSocket;
begin
  WSClient := TsgcWebSocketClient.Create(nil);
  WSClient.URL := 'ws://127.0.0.1:5416';

  // messages look like {"price": 101.25, "change": "+0.4 %"}
  SourceWS := TsgcDashboardSource_WebSocket.Create(nil);
```

```
  SourceWS.Client := WSClient;
  SourceWS.Card := CardPrice;
  SourceWS.ValuePath := 'price';
  SourceWS.TrendValuePath := 'change';
  SourceWS.ValueFormat := '%.2f USD';

  WSClient.Active := True;
end;
```

Voir aussi

- TsgcDashboardSource
- TsgcVCLKPICard
- Démos

Images KPI headless

TsgcDashboardKPI rend une carte KPI sans fenêtre et sans VCL ni FMX, de sorte qu'une application console, un service ou un serveur web peut produire la même carte sous forme d'image, par exemple pour un rapport, un e-mail ou une page web. L'unité se compile sur toutes les plateformes, Linux64 compris.

- **SaveToPNG** écrit un fichier ou un flux PNG. aWidth et aHeight sont des pixels logiques et aScale les multiplie : 320 par 160 à l'échelle 2 donne une image de 640 par 320.
- **SaveToSVG** écrit un fichier ou un flux SVG de la même carte.
- Le modèle de la carte d'un contrôle est disponible via sa propriété **KPI**, si bien que **Card.KPI.SaveToPNG** enregistre ce que le contrôle affiche, à n'importe quelle taille.

Exemple

```
uses
  SysUtils,
  sgcUICore_Theme, sgcDashboard_KPI;

procedure SaveKPIImages;
var
  oKPI: TsgcDashboardKPI;
begin
  oKPI := TsgcDashboardKPI.Create;
  try
    oKPI.Title := 'Orders';
    oKPI.Value := '842';
    oKPI.Icon := '#';
    oKPI.Trend := ktUp;
    oKPI.TrendValue := '+12';
    oKPI.FooterText := 'today';
    oKPI.Theme := utDark;
    oKPI.AddValue(790);
    oKPI.AddValue(815);
    oKPI.AddValue(842);
    // 320 x 160 logical pixels, a 640 x 320 image at scale 2
    oKPI.SaveToPNG('orders.png', 320, 160, 2);
    oKPI.SaveToSVG('orders.svg', 320, 160);
  finally
    oKPI.Free;
  end;
end;
```

Voir aussi

- TsgcDashboardKPI
- Plateformes

Présentation de .NET

sgcDashboard .NET est un port C# managé de la bibliothèque Delphi sgcDashboard, et non une enveloppe (wrapper) autour d'elle. Les classes C# se trouvent dans l'espace de noms **esegece.sgcDashboard** et gardent les noms Delphi, de sorte que **TsgcDashboardLayout**, **TsgcDashboardCell**, **TsgcDashboardKPI**, **TsgcDashboardKPITrend** et **IsgcDashboardKPIHost** se lisent de la même façon dans les deux langages, et le dashboard, le tile et la carte KPI de chaque hôte suivent le schéma de noms de Delphi. Ils sont dessinés avec le port .NET de **sgcUICore**, le même canevas, les mêmes polices et les mêmes thèmes clair et sombre que les autres produits de la suite visuelle eSeGeCe, si bien qu'un dashboard a le même aspect dans WinForms, WPF, Avalonia et dans la sortie headless en PNG et SVG.
Un dashboard dispose des tiles sur une grille de colonnes et de lignes. Chaque tile a une cellule (une colonne, une ligne et l'étendue des deux), un en-tête facultatif avec un titre, et héberge n'importe quel contrôle : une carte KPI, un graphique sgcCharts, une visionneuse sgcPDF, un aperçu sgcReports ou une zone de texte ordinaire. Le thème du dashboard atteint les tiles et les contrôles qu'ils hébergent. Dans l'édition Registered, l'utilisateur peut déplacer et redimensionner les tiles à l'exécution, la disposition est enregistrée et chargée en JSON, et la passerelle sgcWebSockets alimente les cartes KPI à partir d'une connexion en direct.

Packages

Le port est réparti en six packages NuGet, de sorte qu'un projet n'embarque que l'hôte qu'il utilise.

PackageId	Contient	Frameworks cibles
esegece.sgcDashboard	Le moteur de disposition **TsgcDashboardLayout** avec sa persistance JSON, le modèle KPI **TsgcDashboardKPI**, l'énumération de tendance et la sortie SVG headless. Aucune dépendance native.	netstandard2.0, net472, net6.0, net8.0, net9.0
esegece.sgcDashboard.Rendering	La sortie PNG headless d'une carte KPI (**SaveToPNG**), via SkiaSharp.	netstandard2.0, net472, net6.0, net8.0, net9.0
esegece.sgcDashboard.WinForms	**TsgcWinFormsDashboard**, **TsgcWinFormsDashboardTile** et **TsgcWinFormsKPICard**.	net472, net8.0-windows, net9.0-windows
esegece.sgcDashboard.Wpf	**TsgcWpfDashboard**, **TsgcWpfDashboardTile** et **TsgcWpfKPICard**.	net472, net8.0-windows, net9.0-windows
esegece.sgcDashboard.Avalonia	**TsgcAvaloniaDashboard**, **TsgcAvaloniaDashboardTile** et **TsgcAvaloniaKPICard**, sur Avalonia 11.3.	net8.0, net9.0
esegece.sgcDashboard.sgcWebSockets	La passerelle de données en direct : **TsgcDashboardSource**, **TsgcDashboardSource_WebSocket**, **TsgcDashboardSource_MQTT** et	netstandard2.0,

	TsgcDashboardSource_SSE. Elle dépend du package esegece.sgcWebSockets. Édition Registered uniquement.	net8.0, net9.0

Chaque package d'hôte dépend de esegece.sgcDashboard.Rendering et du package sgcUICore du même hôte, il suffit donc d'ajouter le package de l'hôte.

Éditions

Édition	Identifiants de package	Comprend
Registered	esegece.sgcDashboard, .Rendering, .WinForms, .Wpf, .Avalonia et .sgcWebSockets	Tout : la grille et les tiles, les cartes KPI, les thèmes, la sortie headless, le déplacement et le redimensionnement à l'exécution, **SaveLayout** et **LoadLayout**, **OnLayoutChanged** et la passerelle sgcWebSockets.
Community	Les mêmes ids avec le suffixe .Community, par exemple esegece.sgcDashboard.WinForms.Community. Il n'existe pas de package de passerelle Community.	La grille et les tiles, les cartes KPI, les thèmes et la sortie headless. Le déplacement et le redimensionnement à l'exécution, la persistance de la disposition (**SaveLayout**, **LoadLayout** et les **SaveToJSON** et **LoadFromJSON** du moteur) et la passerelle sont exclus.

C'est la même répartition que celle des éditions Delphi, voir Éditions.

Hôtes

Hôte	Équivalent Delphi	Plateformes
WinForms	TsgcVCLDashboard, TsgcVCLDashboardTile et TsgcVCLKPICard	Windows, sur .NET Framework 4.7.2 et .NET 8 et 9.
WPF	TsgcFMXDashboard, TsgcFMXDashboardTile et TsgcFMXKPICard	Windows, sur .NET Framework 4.7.2 et .NET 8 et 9.
Avalonia	TsgcFMXDashboard, TsgcFMXDashboardTile et TsgcFMXKPICard	Windows, Linux et macOS, sur .NET 8 et 9.
Headless	TsgcDashboardKPI et le moteur de disposition	Toute plateforme de .NET Standard 2.0, y compris les serveurs Linux.

Les tiles WPF et Avalonia suivent le tile FireMonkey : ils hébergent un élément et gèrent l'espace qui l'entoure par **ContentPadding**. Le tile WinForms suit le tile VCL et héberge autant de contrôles que l'on veut.

Voir aussi

- Installation depuis NuGet
- Démarrage rapide WinForms

- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Classes .NET
- Différences de .NET par rapport à Delphi
- Rendu headless en .NET
- Flux en direct en .NET
- Démos .NET

Installation depuis NuGet

Les packages sont sur nuget.org. Ajoutez le package de l'hôte, les packages core et de rendu arrivent avec lui comme dépendances.

Édition Registered

```
dotnet add package esegece.sgcDashboard.WinForms
dotnet add package esegece.sgcDashboard.Wpf
dotnet add package esegece.sgcDashboard.Avalonia
```

Ajoutez la passerelle pour alimenter des cartes KPI à partir de sgcWebSockets. Elle apporte le package esegece.sgcWebSockets :

```
dotnet add package esegece.sgcDashboard.sgcWebSockets
```

Édition Community

```
dotnet add package esegece.sgcDashboard.WinForms.Community
dotnet add package esegece.sgcDashboard.Wpf.Community
dotnet add package esegece.sgcDashboard.Avalonia.Community
```

Ou ajoutez la référence au fichier de projet à la main :

```
<ItemGroup>
  <PackageReference Include="esegece.sgcDashboard.Wpf" Version="2026.10.0" />
</ItemGroup>
```

Référencez dans un projet soit les packages Registered, soit les packages Community, jamais les deux : ils contiennent les mêmes types dans le même espace de noms. Le code qui utilise **AllowDrag**, **AllowResize**, **SaveLayout**, **LoadLayout**, **OnLayoutChanged** ou la propriété **Locked** d'un tile ne compile pas avec les packages Community ; protégez-le avec `#if !SGC_COMMUNITY` lorsqu'un projet compile les deux éditions.

Paramètres du projet

- WinForms : un projet `net8.0-windows` avec `<UseWindowsForms>true</UseWindowsForms>`, ou un projet .NET Framework 4.7.2.
- WPF : un projet `net8.0-windows` avec `<UseWPF>true</UseWPF>`, ou un projet .NET Framework 4.7.2. Associez l'espace de noms en XAML avec `xmlns:dash="clr-namespace:esegece.sgcDashboard;assembly=esegece.sgcDashboard.Wpf"`.
- Avalonia : un projet `net8.0` ou `net9.0` avec les packages **Avalonia.Desktop** et **Avalonia.Themes.Fluent** d'Avalonia 11.3, comme toute application de bureau Avalonia. Associez l'espace de noms en XAML avec `xmlns:dash="using:esegece.sgcDashboard"`.
- Headless : esegece.sgcDashboard seul suffit pour le moteur de disposition et pour SVG. Ajoutez esegece.sgcDashboard.Rendering pour PNG, il apporte SkiaSharp et ses ressources natives pour Windows et Linux.
- Passerelle : esegece.sgcDashboard.sgcWebSockets cible netstandard2.0, net8.0 et net9.0, elle fonctionne donc avec tous les hôtes, .NET Framework 4.7.2 compris.

En C#, chaque hôte a besoin de `using esegece.sgcDashboard;`, et de `using esegece.sgcUICore;` pour **TsgcUITheme** et les autres types des thèmes. La passerelle a aussi besoin de `using esegece.sgcWebSockets;`.

Voir aussi

- Présentation de .NET
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia

- Flux en direct en .NET
- Démos .NET

Démarrage rapide WinForms

Cet exemple construit un formulaire WinForms avec un dashboard de trois tiles : deux cartes KPI et un journal d'événements dans une zone de texte ordinaire. Un timer alimente la carte des revenus, et une case à cocher fait passer le dashboard du thème clair au thème sombre.

Étapes

- Créez un projet WinForms `net8.0-windows` et ajoutez le package **esegece.sgcDashboard.WinForms**, voir Installation depuis NuGet.
- Créez un **TsgcWinFormsDashboard**, ancrez-le (dock) et définissez **Columns** et **RowHeight**. La grille s'adapte toujours à la largeur du dashboard et défile verticalement.
- Pour chaque tile, créez un **TsgcWinFormsDashboardTile**, donnez-lui un **Name**, un **Title** et sa cellule (**Col**, **Row**, **ColSpan** et **RowSpan**), ajoutez le contenu aux **Controls** du tile et le tile aux **Controls** du dashboard. Un contrôle ancré remplit la zone située sous l'en-tête.
- Définissez **Value**, **Trend**, **TrendValue** et **FooterText** d'un **TsgcWinFormsKPICard** et appelez **AddValue** pour ajouter un point à son sparkline.
- Définissez **Theme** du dashboard : les tiles, les cartes et la zone de texte le suivent.

Exemple

AllowDrag et **AllowResize** permettent à l'utilisateur de déplacer un tile par son en-tête et de le redimensionner depuis la poignée en bas à droite, aligné sur la grille. Ils font partie de l'édition Registered ; supprimez les deux lignes pour les packages Community.

Enregistrer et restaurer la disposition

SaveLayout renvoie les dimensions de la grille ainsi que la cellule et la visibilité de chaque tile nommé sous forme de JSON, et **LoadLayout** applique à nouveau un tel texte. **SaveLayoutToFile** et **LoadLayoutFromFile** font de même avec un fichier, **SaveLayoutToStream** et **LoadLayoutFromStream** avec un flux UTF-8. **OnLayoutChanged** se déclenche après qu'un glissement a modifié une cellule et après un chargement. La persistance de la disposition fait partie de l'édition Registered.
Le programme écrit :

```
{"version":1,"columns":12,"rowHeight":120,"gap":12,"padding":12,"tiles":[{"name":"tileSales","col":0,"row":0,"colSpan":6,"r
tileSales at 0,0 size 6x2
tileOrders at 6,0 size 6x2
OnLayoutChanged: 2, round trip: ok
```

Le JSON est le texte qu'écrit le **SaveLayout** de Delphi, si bien qu'une disposition enregistrée par une application Delphi se charge dans une application .NET et inversement. Les tiles sont associés par **Name** sans tenir compte de la casse ; un tile sans nom n'est pas enregistré.

Voir aussi

- Installation depuis NuGet
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Classes .NET
- Persistance
- TsgcVCLDashboard
- Démos .NET

Démarrage rapide WPF

Le même dashboard en WPF. Les tiles et ce qu'ils hébergent sont déclarés en XAML : les tiles sont le contenu du dashboard, et chaque tile prend un élément comme contenu.

Étapes

- Créez un projet WPF `net8.0-windows` et ajoutez le package **esegece.sgcDashboard.Wpf**, voir Installation depuis NuGet.
- Associez l'espace de noms dans la fenêtre avec `xmlns:dash="clr-namespace:esegece.sgcDashboard;assembly=esegece.sgcDashboard.Wpf"`.
- Déclarez un **TsgcWpfDashboard** et, à l'intérieur, un **TsgcWpfDashboardTile** par tile avec son **Title**, **Col**, **Row**, **ColSpan** et **RowSpan**. Le contenu d'un tile est un **TsgcWpfKPICard** ou n'importe quel autre élément.
- Donnez un `x:Name` aux tiles : c'est le nom que **SaveLayout** enregistre.
- Alimentez les cartes et changez **Theme** depuis le code-behind.

Exemple

MainWindow.xaml:
MainWindow.xaml.cs:
App.xaml est celui d'un nouveau projet WPF, avec `StartupUri="MainWindow.xaml"`. **AllowDrag** et **AllowResize** font partie de l'édition Registered ; supprimez les deux attributs pour les packages Community.

Voir aussi

- Installation depuis NuGet
- Démarrage rapide WinForms
- Démarrage rapide Avalonia
- Classes .NET
- Différences de .NET par rapport à Delphi
- TsgcFMXDashboard
- Démos .NET

Démarrage rapide Avalonia

Le même dashboard en Avalonia, sous Windows, Linux et macOS. Comme en WPF, les tiles et leur contenu sont déclarés en XAML.

Étapes

- Créez un projet de bureau Avalonia `net8.0` avec **Avalonia.Desktop** et **Avalonia.Themes.Fluent** 11.3, et ajoutez le package **esegece.sgcDashboard.Avalonia**, voir Installation depuis NuGet.
- Associez l'espace de noms dans la fenêtre avec `xmlns:dash="using:esegece.sgcDashboard"`.
- Déclarez un **TsgcAvaloniaDashboard** et, à l'intérieur, un **TsgcAvaloniaDashboardTile** par tile. Le contenu d'un tile est un **TsgcAvaloniaKPICard** ou n'importe quel autre contrôle.
- Alimentez les cartes et changez **Theme** depuis le code-behind.

Exemple

MainWindow.axaml:
MainWindow.axaml.cs:
Program.cs:
Le tile gère son propre **Padding** à partir de **ContentPadding** et **ShowHeader**, un **Padding** défini en XAML est donc remplacé. Sur un écran tactile, un appui sur l'en-tête d'un tile fait glisser le tile et un appui n'importe où ailleurs fait défiler la grille, voir Différences de .NET par rapport à Delphi.

Voir aussi

- Installation depuis NuGet
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Classes .NET
- Différences de .NET par rapport à Delphi
- TsgcFMXDashboard
- Démos .NET

Classes .NET

Toutes les classes se trouvent dans l'espace de noms **esegece.sgcDashboard**. Les classes d'hôte sont des contrôles légers au-dessus du moteur et du modèle indépendants du framework, qui sont partagés par les trois hôtes et par la sortie headless.

Classes d'hôte

Hôte	Classe	Classe de base	Delphi
WinForms	TsgcWinFormsDashboard	ScrollableControl	TsgcVCLDashboard
WinForms	TsgcWinFormsDashboardTile	TsgcUIWinFormsControl	TsgcVCLDashboardTile
WinForms	TsgcWinFormsKPICard	TsgcUIWinFormsControl	TsgcVCLKPICard
WPF	TsgcWpfDashboard	FrameworkElement	TsgcFMXDashboard
WPF	TsgcWpfDashboardTile	TsgcUIWpfControl	TsgcFMXDashboardTile
WPF	TsgcWpfKPICard	TsgcUIWpfControl	TsgcFMXKPICard
Avalonia	TsgcAvaloniaDashboard	TsgcUIAvaloniaControl	TsgcFMXDashboard
Avalonia	TsgcAvaloniaDashboardTile	TsgcUIAvaloniaControl	TsgcFMXDashboardTile
Avalonia	TsgcAvaloniaKPICard	TsgcUIAvaloniaControl	TsgcFMXKPICard

Types indépendants du framework

Type	Description
TsgcDashboardLayout	Le moteur de grille : **Columns**, **RowHeight**, **Gap**, **Padding** et **AutoFlow**, cellules des tiles vers des rectangles (**Arrange**, **Rect**, **ContentHeight**), flux automatique, poussée vers le bas en cas de collision, le calcul d'alignement du glissement et du redimensionnement (**SnapMove**, **SnapResize**) et le JSON de la disposition (**SaveToJSON**, **LoadFromJSON**, édition Registered).
TsgcDashboardCell	Une struct avec **Col**, **Row**, **ColSpan** et **RowSpan**. `sgcDashboard_Layout.sgcDashboardCell` en construit une.
TsgcDashboardLayoutTile	Le nom, la cellule et la visibilité d'un tile dans le JSON de la disposition. Édition Registered.
TsgcDashboardKPI	Le modèle KPI et son moteur de rendu, voir TsgcDashboardKPI.
TsgcDashboardKPITrend	`ktNone` , `ktUp` , `ktDown` ou `ktNeutral` .

IsgcDashboardKPIHost	Implémentée par les trois contrôles de carte KPI ; la passerelle accepte toute carte qui l'implémente.

Dashboard

Les mêmes membres dans **TsgcWinFormsDashboard**, **TsgcWpfDashboard** et **TsgcAvaloniaDashboard**.

Membre	Type	Description
Columns	int	Nombre de colonnes de la grille, 12 par défaut.
RowHeight	int	Hauteur d'une ligne de la grille en pixels indépendants de l'appareil, 120 par défaut.
Gap	int	Espace entre les tiles, 12 par défaut.
Padding	int	Espace autour de la grille, 12 par défaut. Masque `Control.Padding` dans WinForms.
AutoFlow	bool	Place les tiles sans cellule (Col ou Row -1) dans les premières cellules libres. True par défaut.
Theme	TsgcUITheme	Thème du dashboard, de ses tiles et des contrôles qu'ils hébergent.
Layout	TsgcDashboardLayout	Le moteur de grille, en lecture seule. Masque l'événement `Control.Layout` dans WinForms.
Tiles	collection	WPF et Avalonia : les tiles, la propriété de contenu XAML.
VerticalOffset	double	Avalonia : la position de défilement de la grille.
AllowDrag	bool	Déplace un tile par son en-tête à l'exécution. Édition Registered.
AllowResize	bool	Redimensionne un tile depuis sa poignée en bas à droite à l'exécution. Édition Registered.
OnLayoutChanged	TsgcUINotifyEvent	Se déclenche après qu'un glissement a modifié une cellule et après un chargement. Édition Registered.
SaveLayout, LoadLayout	méthode	La disposition sous forme de chaîne JSON. Édition Registered.
SaveLayoutToStream, LoadLayoutFromStream	méthode	La disposition sous forme de flux UTF-8. Édition Registered.
SaveLayoutToFile, LoadLayoutFromFile	méthode	La disposition sous forme de fichier. Édition Registered.
Arrange	méthode	Réorganise les tiles.
ApplyTheme	méthode	Applique à nouveau le thème aux tiles et à leur contenu, par exemple après que le contenu a modifié ses propres couleurs.

Tile

Les mêmes membres dans **TsgcWinFormsDashboardTile**, **TsgcWpfDashboardTile** et **TsgcAvaloniaDashboardTile**.

Membre	Type	Description
Dashboard	dashboard	Le dashboard qui contient le tile, ou null.
Title	string	Texte de l'en-tête.
ShowHeader	bool	Affiche la bande d'en-tête, true par défaut.
Col, Row	int	La cellule demandée, -1 par défaut pour que le flux automatique place le tile.
ColSpan, RowSpan	int	Taille du tile en colonnes et en lignes, 3 et 2 par défaut.
GetCell, SetCell	méthode	Les quatre valeurs sous la forme d'un seul **TsgcDashboardCell**.
ParentTheme	bool	Prend le thème du dashboard, true par défaut.
Theme	TsgcUI-Theme	Thème du tile lorsque **ParentTheme** vaut false.
Locked	bool	Le tile ne démarre jamais un glissement ni un redimensionnement. Édition Registered.
Content	élément	WPF et Avalonia : l'élément hébergé, la propriété de contenu XAML. Dans WinForms, le tile héberge les contrôles de sa collection **Controls**.
ContentPadding	int	WPF et Avalonia : espace autour du contenu, 8 par défaut ; la hauteur de l'en-tête s'ajoute en haut tant que **ShowHeader** vaut true.

Carte KPI

TsgcWinFormsKPICard, **TsgcWpfKPICard** et **TsgcAvaloniaKPICard** encapsulent un TsgcDashboardKPI, disponible dans la propriété **KPI**, et reprennent ses membres : **Title**, **Value**, **Icon**, **Trend**, **TrendValue**, **FooterText**, **AccentColor**, **Theme**, **MaxPoints** et **AddValue**.

Voir aussi

- Présentation de .NET
- Différences de .NET par rapport à Delphi
- Rendu headless en .NET
- Flux en direct en .NET
- Disposition
- Thèmes

Différences de .NET par rapport à Delphi

Le port .NET conserve les noms, les valeurs par défaut et les règles de disposition de Delphi. Les différences ci-dessous viennent des frameworks d'hôte.

Les tiles sont des conteneurs enfants

Hôte	Comment un tile héberge son contenu
WinForms	Un tile est un conteneur `Control`, comme le tile VCL : ajoutez autant de contrôles que vous voulez à ses **Controls**. Ils sont placés sous l'en-tête avec un retrait de 8 pixels, si bien qu'un contrôle avec `Dock = DockStyle.Fill` remplit la zone de contenu. Le tile porte le ParentControlDesigner, on peut donc y déposer un contrôle dans le concepteur.
WPF	Un tile prend un élément par **Content**, la propriété de contenu XAML (`[ContentProperty("Content")]`), comme le tile FireMonkey. Le dashboard est le parent logique de ses tiles, et **Tiles** est sa propriété de contenu, les tiles sont donc déclarés comme enfants de l'élément dashboard.
Avalonia	Le même modèle qu'en WPF, par l'attribut `[Content]` sur **Content** du tile et sur **Tiles** du dashboard. Le dashboard est le parent logique et visuel de chaque tile, et le tile celui de son contenu.

En WPF et Avalonia, le tile calcule son **Padding** à partir de **ContentPadding** et **ShowHeader**, un **Padding** défini en XAML est donc remplacé. Un tile masqué : en WPF, un tile `Hidden` garde sa cellule et un tile `Collapsed` quitte la grille ; en Avalonia, un tile dont `IsVisible` vaut false quitte la grille.

Propagation du thème aux contrôles hébergés

Lorsque le **Theme** du dashboard change, lorsqu'un tile entre dans un dashboard, lorsqu'un contenu est ajouté à un tile et lors de **ApplyTheme**, les contrôles hébergés prennent le thème du tile par les helpers de sgcUICore `sgcUIApplyThemeWinForms`, `sgcUIApplyThemeWpf` et `sgcUIApplyThemeAvalonia`, les équivalents des `sgcUIApplyThemeVCL` et `sgcUIApplyThemeFMX` de Delphi. Ainsi, un graphique sgcCharts, une visionneuse sgcPDF ou un aperçu sgcReports dans un tile passe du clair au sombre avec le dashboard. `utCustom` laisse les contrôles hébergés intacts.

Le JSON de la disposition vit dans le moteur

SaveToJSON et **LoadFromJSON** de **TsgcDashboardLayout** écrivent et lisent le texte de la disposition ; les méthodes **SaveLayout** et **LoadLayout** des trois hôtes ne font que rassembler leurs tiles et les appeler. Le texte est celui qu'écrit le dashboard Delphi, les dispositions circulent donc librement entre Delphi, WinForms, WPF et Avalonia. La clé d'un tile est `Control.Name` dans WinForms, `FrameworkElement.Name` (`x:Name`) en WPF et `StyledElement.Name` dans Avalonia. Voir Rendu headless en .NET pour un exemple sans hôte.

Padding et Layout dans WinForms

TsgcWinFormsDashboard déclare deux membres avec `new` : son **Padding** est l'espace autour de la grille, un `int`, et masque `Control.Padding`, et sa propriété **Layout** renvoie le moteur de grille et masque l'événement `Control.Layout`. Convertissez le dashboard en `Control` pour atteindre les membres masqués.

Theme dans Avalonia

Chaque contrôle Avalonia possède déjà une propriété **Theme**, le `ControlTheme` de `StyledElement`. Le dashboard, le tile et la carte KPI déclarent leur propre **Theme** de type **TsgcUITheme** avec `new`. Par une référence du type sgc, c'est le thème sgc ; par une référence typée `Control` ou `StyledElement`, c'est le thème de contrôle d'Avalonia.

Glissement et défilement tactiles dans Avalonia

Un contact tactile qui se déplace de plus de 8 unités à la verticale fait défiler la grille. Avec **AllowDrag** ou **AllowResize**, un appui tactile sur l'en-tête ou la poignée d'un tile démarre aussitôt le glissement du tile et ne fait jamais défiler, comme dans FireMonkey ; un appui tactile n'importe où ailleurs fait toujours défiler la grille. Un glissement à la souris ne fait jamais défiler. En WPF, le tactile et le stylet sont promus en entrée souris, et WinForms n'utilise que la souris.

Le contour du glissement

Pendant qu'un tile est glissé ou redimensionné, la cellule cible est marquée avec la couleur d'accent et un remplissage de 25 pour cent. WinForms la dessine sous les tiles, comme le dashboard VCL. WPF la dessine dans une couche d'adorner et Avalonia dans un visuel de superposition, tous deux au-dessus des tiles, alors que le marqueur FireMonkey est en dessous. **Escape** annule le glissement dans tous les hôtes, et en WPF et Avalonia une capture perdue l'annule aussi.

Défilement

La grille s'adapte toujours à la largeur et défile verticalement, d'une demi-ligne par cran de molette dans tous les hôtes. WinForms défile par `ScrollableControl`. WPF place la grille dans un `ScrollViewer`. Avalonia affiche sa propre `ScrollBar` et expose la position sous la forme de **VerticalOffset**, qui remonte les tiles et les rogne au dashboard.

Événements

OnLayoutChanged est une propriété du type délégué nommé de Delphi **TsgcUINotifyEvent**, comme dans les autres produits sgc .NET : affectez-la avec `=`, elle ne contient qu'un gestionnaire. Les sources de la passerelle s'abonnent aux événements de sgcWebSockets .NET avec `+=` et ne retirent que leur propre gestionnaire, si bien que les gestionnaires de l'application continuent de se déclencher, voir Flux en direct en .NET.

Pas de verbes de concepteur et pas de tile de carte

Le verbe d'éditeur de composant Delphi "Add Tile" n'a pas d'équivalent dans WinForms : ajoutez les tiles dans le code, ou déclarez-les en XAML en WPF et Avalonia. L'alignement à la conception d'un tile déplacé ne fait pas non plus partie du port. La démo Suite de Delphi a un tile sgcMaps ; sgcMaps n'a pas de port .NET, la démo Suite .NET a donc un graphique, une visionneuse PDF et un aperçu de rapport, mais pas de carte.

Éditions

Les packages Community sont compilés avec `SGC_COMMUNITY`. Le déplacement et le redimensionnement à l'exécution, **Locked**, la persistance de la disposition et **OnLayoutChanged** sont exclus de la compilation, et le package de la passerelle n'est pas produit, comme le bloc `SGC_DASHBOARD_PERSIST` de Delphi. Protégez le code qui les utilise avec `#if !SGC_COMMUNITY`.

Liaison de données

Sous WPF et Avalonia, les propriétés visibles par l'utilisateur sont des propriétés de dépendance et des propriétés Avalonia, si bien qu'elles peuvent être liées, stylées et animées, voir Liaison de données .NET.

Voir aussi

- Présentation de .NET
- Classes .NET
- Rendu headless en .NET
- Flux en direct en .NET
- Disposition
- Thèmes
- Persistance

Liaison de données .NET

Dans les hôtes WPF et Avalonia, les propriétés visibles par l'utilisateur du dashboard, de ses tiles et de la carte KPI sont des propriétés de dépendance (WPF) et des propriétés Avalonia. Elles peuvent donc être liées à un modèle de vue, définies par un Style et animées. Cette rubrique s'applique uniquement à WPF et Avalonia. Les contrôles WinForms ne changent pas : leurs propriétés sont de simples propriétés CLR, comme avant.

Propriétés pouvant être liées

Chaque propriété du tableau possède un champ statique avec le suffixe Property, par exemple `TsgcWpfDashboardTile.ColProperty`. Sous WPF, les champs sont de type DependencyProperty. Sous Avalonia, ce sont des StyledProperty, sauf Col, Row, ColSpan et RowSpan, qui sont des DirectProperty. Les propriétés marquées **TwoWay** sont liées en TwoWay par défaut, car un glissement ou un redimensionnement à l'exécution et LoadLayout les modifient, de sorte qu'une liaison sans Mode maintient le modèle de vue à jour. Les autres propriétés sont liées en OneWay par défaut.

Classe WPF	Classe Avalonia	Propriétés pouvant être liées
TsgcWpfDashboard	TsgcAvaloniaDashboard	Columns, RowHeight, Gap, Padding, AutoFlow, Theme, AllowDrag, AllowResize
TsgcWpfDashboardTile	TsgcAvaloniaDashboardTile	Title, **Col (TwoWay)**, **Row (TwoWay)**, **ColSpan (TwoWay)**, **RowSpan (TwoWay)**, ShowHeader, ContentPadding, ParentTheme, Locked
TsgcWpfKPICard	TsgcAvaloniaKPICard	Title, Value, Icon, Trend, TrendValue, FooterText, AccentColor, MaxPoints

AllowDrag et AllowResize ne sont pas dans l'édition Community. Padding est l'espace autour de la grille en unités indépendantes de l'appareil, un int. Sous Avalonia, Theme est le thème sgcUICore (TsgcUITheme) et masque le thème de contrôle Avalonia de StyledElement, qui reste accessible via `StyledElement.ThemeProperty`.

Comportement

- **Un glissement réécrit la cellule.** Un glissement ou un redimensionnement à l'exécution, ainsi que LoadLayout, modifient la cellule d'un tile, et le tile copie la cellule alignée dans Col, Row, ColSpan et RowSpan. Avec la liaison TwoWay par défaut, le modèle de vue reçoit la cellule où le tile a atterri.
- **Une cellule liée est un changement de disposition.** Une nouvelle cellule qui arrive par une liaison de données est traitée comme un glissement, si bien que OnLayoutChanged du dashboard se déclenche (pas dans l'édition Community). Définies depuis le code (les setters CLR ou SetCell), les propriétés de cellule se comportent comme avant.
- **Une valeur rejetée est réécrite.** Une valeur que le dashboard ou le tile n'accepte pas (Columns ou RowHeight inférieur à 1, un Gap ou un Padding négatif, une étendue inférieure à 1, un ContentPadding négatif) n'est pas appliquée, et la valeur utilisée est réécrite dans la propriété et dans une source TwoWay, sous WPF et sous Avalonia. Il en va de même pour une valeur que la carte KPI corrige (MaxPoints inférieur à 1, un texte null). Sous WPF, la source reçoit la valeur utilisée une fois que le dispatcher a exécuté le travail en attente. Sous Avalonia, la réécriture couvre une valeur que la source envoie lorsque la liaison est déjà active. Une valeur que la source détient déjà au démarrage de la liaison est corrigée dans le contrôle, mais n'est pas réécrite dans la source.
- **LoadLayout met à jour les propriétés.** Après LoadLayout, les métriques de la grille et les cellules des tiles sont recopiées dans les propriétés, si bien que les liaisons voient la disposition restaurée.
- **Styles.** Un setter de Style peut définir n'importe quelle propriété pouvant être liée. Un Theme du dashboard défini par un Style atteint chaque tile dont ParentTheme est true.

Propriétés qui restent des propriétés CLR

Propriété	Raison
Tiles (dashboard)	La collection de tiles, remplie par les tiles déclarés en XAML, et non une valeur à lier.
Layout (dashboard)	L'objet du moteur de disposition, en lecture seule. Ses métriques de grille sont liées via Columns, RowHeight, Gap, Padding et AutoFlow.
VerticalOffset (dashboard Avalonia)	La position de défilement de la dernière passe de disposition, et non un paramètre.
Theme (tile)	Tant que ParentTheme est true, le tile lit le thème du dashboard, et définir Theme efface ParentTheme. Liez plutôt le Theme du dashboard, ou ParentTheme.
Theme (carte KPI)	Défini par le tile qui héberge la carte.

Exemple

Le modèle de vue ci-dessous est partagé par les deux exemples. Revenue est lié au Value d'une carte KPI, et RevenueCol et RevenueRow sont liés à Col et Row de son tile sans Mode, ils sont donc liés en TwoWay : lorsque l'utilisateur fait glisser le tile, le modèle de vue reçoit la nouvelle cellule. Définissez AllowDrag à True sur le dashboard pour que l'utilisateur puisse faire glisser les tiles.

WPF

MainWindow.xaml crée le modèle de vue comme DataContext de la fenêtre. Le code-behind et App.xaml sont ceux du modèle WPF.

Avalonia

MainWindow.axaml définit `x:DataType` sur le modèle de vue, si bien que les liaisons sont des liaisons compilées. Le code-behind appelle InitializeComponent, et la classe App est celle de Démarrage rapide Avalonia.

Voir aussi

- Différences de .NET par rapport à Delphi
- Classes .NET
- Démarrage rapide WPF
- Démarrage rapide Avalonia

Rendu headless en .NET

Le moteur de disposition et la carte KPI n'ont besoin d'aucun framework d'interface, si bien qu'un serveur, une application console ou un test peut placer des tiles et rendre des cartes dans des fichiers image. C'est l'équivalent .NET de Images KPI headless.

Méthodes

Méthode	Package	Description
TsgcDashboardKPI.SaveToSVG(stream or file, width, height, scale)	esegece.sgcDashboard	Écrit la carte sous forme de texte SVG, octet pour octet le SVG de la carte Delphi.
TsgcDashboardKPI.SaveToPNG(stream or file, width, height, scale)	esegece.sgcDashboard.Rendering	Une méthode d'extension qui écrit un PNG à fond transparent. L'image fait largeur fois échelle par hauteur fois échelle pixels.
TsgcDashboardKPI.Render(canvas, width, height, scale)	esegece.sgcDashboard	Dessine la carte sur n'importe quel canevas sgcUICore.
TsgcDashboardLayout.Arrange(width, scale)	esegece.sgcDashboard	Place les éléments ; **PlacedCell**, **Rect** et **ContentHeight** renvoient le résultat.
TsgcDashboardLayout.SaveToJSON, LoadFromJSON	esegece.sgcDashboard	Le texte de disposition des dashboards. Édition Registered.

Exemple

Le programme écrit revenue.svg et revenue.png, de 560 par 400 pixels, et affiche :

```
revenue: column 0, row 0
orders: column 2, row 2
{"version":1,"columns":12,"rowHeight":100,"gap":12,"padding":12,"tiles":[{"name":"revenue","col":0,"row":0,"colSpan":4,"row
orders back at column 2, row 0
```

Le tile orders demande la colonne 2, ligne 0, qui chevauche le tile revenue, le moteur le pousse donc vers le bas à la ligne 2. **Cell** renvoie la cellule qu'un tile a demandée, **PlacedCell** celle qu'il a obtenue ; le JSON enregistre les cellules demandées, comme Delphi.
Les packages headless fonctionnent sous Windows, Linux et macOS. Sous Linux, esegece.sgcDashboard.Rendering apporte les ressources natives de SkiaSharp pour Linux.

Voir aussi

- Présentation de .NET
- Classes .NET
- Différences de .NET par rapport à Delphi
- Images KPI headless

- TsgcDashboardKPI
- Disposition

Flux en direct en .NET

Le package esegece.sgcDashboard.sgcWebSockets alimente des cartes KPI à partir d'une connexion en direct de sgcWebSockets .NET, comme la passerelle sgcDashboardWS de Delphi, voir Données en direct. Une source lit un nombre dans chaque message JSON et le pousse dans une carte : le sparkline reçoit le point, et **Value**, **Title**, **TrendValue** et **Trend** sont mis à jour. **Édition Registered uniquement** : le package de la passerelle n'est pas livré dans l'édition Community.

Sources

Source	Client	Lit
TsgcDashboardSource_WebSocket	TsgcWebSocketClient	Chaque message texte, par **OnMessage**.
TsgcDashboardSource_MQTT	TsgcWSProtocol_MQTT_Client	Chaque publish dont le topic correspond à **Topic**, avec les jokers MQTT `+` et `#`, par **OnMQTTPublish**. La source ne s'abonne pas ; l'application s'abonne par le client.
TsgcDashboardSource_SSE	TsgcHTTP1Client dans **Client**, ou TsgcSSEClient dans **SSEClient**	Les données de chaque server sent event, à partir des fragments de **GetSSE** et **OnSSEMessage**, ou à partir des événements de **OnEvent**. **GetSSE** bloque jusqu'à la fin du flux, appelez-le donc depuis un thread de travail.

Toutes les sources partagent les membres de TsgcDashboardSource :

Membre	Description
Card	Un contrôle de carte KPI (tout **IsgcDashboardKPIHost**) ou un **TsgcDashboardKPI** headless. Tout autre objet déclenche **EsgcDashboardSource**.
ValuePath	Chemin JSON du nombre, par exemple `price` ou `data.items[0].value`. Un nombre ou une chaîne numérique est lu, avec le point comme séparateur décimal. Vide lit le message entier comme le nombre.
TitlePath, TrendValuePath	Lorsqu'ils sont définis, copient cette valeur JSON dans **Title** et **TrendValue** de la carte.
ValueFormat	Une chaîne de format Delphi avec un argument float, comme `%.2f` ou `%.1f ms`, formatée avec la culture actuelle. Vide écrit le nombre comme le FloatToStr de Delphi.
Active	True par défaut. False ignore tous les messages.
OnValue	Appelé avec chaque nombre lu, après la mise à jour de la carte.

Threads

Le point est ajouté au sparkline aussitôt, dans le thread où le message arrive. Les textes et la tendance (à la hausse, à la baisse ou neutre par rapport à la valeur précédente de la source) sont appliqués par le `Synchroniza-`

`tionContext` qui était courant à la création de la source, le thread d'interface d'une application WinForms, WPF ou Avalonia, si bien que chaque mode **NotifyEvents** du client est sûr. Une source créée sans contexte de synchronisation, dans une application console ou un service, applique la mise à jour dans le thread du message.

Exemple

Trois cartes KPI sur un dashboard WinForms, alimentées par un flux de cotations WebSocket, un topic de capteur MQTT et un flux de server sent events. Les noms d'hôte sont des paramètres fictifs pour vos propres serveurs.

Essai sur l'interface de bouclage

Ce programme console démarre un **TsgcWebSocketServer** dans le même processus sur l'interface de bouclage, y connecte un **TsgcWebSocketClient** et alimente une carte headless avec trois messages de diffusion, il n'a donc besoin d'aucun réseau.
Il affiche la valeur formatée dans la culture actuelle, ici avec une virgule comme séparateur décimal, la tendance par rapport à la valeur précédente et le nombre de points du sparkline :

```
EUR/USD 1,0839, trend ktDown, 3 points
```

Différences par rapport à Delphi

- **OnMessage**, **OnMQTTPublish**, **OnSSEMessage** et **OnEvent** sont des événements .NET. Une source ajoute son gestionnaire avec `+=` et ne retire que son propre gestionnaire, là où Delphi sauvegarde, remplace et restaure le gestionnaire de l'application. Les gestionnaires de l'application se déclenchent toujours, dans l'ordre d'abonnement.
- Il n'y a pas de FreeNotification en .NET. Lorsque la carte est un `IComponent`, comme un contrôle WinForms, la source efface **Card** quand la carte est libérée. Pour toute autre carte, mettez **Card** à null avant que la carte ne disparaisse.
- **SSEClient** de la source SSE, qui accepte un **TsgcSSEClient**, est un ajout de .NET.

Voir aussi

- Présentation de .NET
- Installation depuis NuGet
- Différences de .NET par rapport à Delphi
- Données en direct
- TsgcDashboardSource
- TsgcDashboardSource_WebSocket
- TsgcDashboardSource_MQTT
- TsgcDashboardSource_SSE
- Démos .NET

Démos .NET

Les démos .NET se trouvent dans le dossier `demos` du produit et dans le zip des démos, avec la solution `sgcDashboard_Demos.sln`. Chaque démo a un README.md.

Démo	Hôte	Description
Basic	WinForms	Six tiles sur une grille de 12 colonnes : quatre **TsgcWinFormsKPICard** alimentées par un timer avec une marche aléatoire, un journal d'événements et une liste de serveurs dans des contrôles ordinaires, et une liste Theme qui fait passer le dashboard du clair au sombre.
BasicWpf	WPF	Le même dashboard déclaré en XAML sur un **TsgcWpfDashboard**.
BasicAvalonia	Avalonia	Le même dashboard sur un **TsgcAvaloniaDashboard**, sous Windows, Linux et macOS.
Suite	WinForms	Les produits eSeGeCe sur un seul dashboard : deux cartes KPI, un graphique sgcCharts avec une courbe en direct, une visionneuse sgcPDF et un aperçu sgcReports dans des tiles, et un commutateur clair et sombre. Il n'y a pas de tile de carte, car sgcMaps n'a pas de port .NET.
LiveKPI	WinForms	Trois cartes KPI alimentées par **TsgcDashboardSource_WebSocket** à partir d'un **TsgcWebSocketServer** dans le même processus sur un port de bouclage libre. Édition Registered uniquement.
SaveLayout	WinForms	Déplacez et redimensionnez les tiles, puis enregistrez la disposition dans layout.json, rechargez-la ou réinitialisez-la. Édition Registered uniquement.

Avec les packages Community, LiveKPI et SaveLayout compilent toujours : ils excluent le code Registered de la compilation et expliquent la limite dans la barre d'état.

Compiler et exécuter

Les démos nécessitent le SDK .NET 8 ou ultérieur. Lancez une démo depuis son dossier :

```
dotnet run
```

Dans l'arborescence du produit, les démos se compilent contre les projets de la bibliothèque. Depuis le zip des démos, elles restaurent les packages NuGet, l'édition Community par défaut. Définissez **SgcDashboardEdition** pour utiliser les packages Registered :

```
dotnet build -p:SgcDashboardEdition=Registered
```

Auto-vérification

Chaque démo a une option `--test` qui ouvre la fenêtre hors écran et la pilote : Basic vérifie les tiles et leurs cellules, alimente les cartes et change le thème, Suite vérifie le graphique, la visionneuse PDF et l'aperçu de rapport dans leurs tiles, LiveKPI vérifie que chaque carte a reçu des valeurs, et SaveLayout enregistre, déplace un tile, charge et réinitialise. La démo écrit une ligne PASS ou FAIL par vérification dans le fichier et se termine avec le nombre de vérifications échouées, 0 signifie donc que toutes les vérifications ont réussi.

```
dotnet run -- --test result.txt
```

Voir aussi

- Présentation de .NET
- Installation depuis NuGet
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Flux en direct en .NET
- Démos Delphi

Dépannage

Identificateur non déclaré SaveLayout, AllowDrag ou OnLayoutChanged

L'édition Community se compile sans SGC_DASHBOARD_PERSIST, si bien que AllowDrag, AllowResize, OnLayoutChanged, les méthodes SaveLayout et LoadLayout et la propriété VCL Locked n'existent pas, et le code qui les utilise échoue avec E2003 Undeclared identifier. Une fiche enregistrée par une build Registered qui contient AllowDrag ou AllowResize dans son .dfm déclenche une exception "Error reading" avec "Property AllowDrag does not exist". Supprimez ces membres, ou utilisez une build Registered ou Trial. Voir Éditions.

Unité sgcDashboardWS_Source_WebSocket ou sgcWebSocket introuvable

La passerelle de données en direct est un package distinct, sgcDashboardWS, qui requiert sgcDashboard et sgcWebSockets. Installez d'abord sgcWebSockets pour le même IDE, puis les packages sgcDashboardWS, et ajoutez le dossier des sources ou des bibliothèques de sgcWebSockets au chemin de recherche. La passerelle n'est pas livrée dans l'édition Community.

Une carte KPI alimentée par une source de la passerelle ne change jamais dans une application console ou un service

Sous Delphi 2010 et versions ultérieures, les textes de la carte sont appliqués dans le thread principal via TThread.Queue. Une application console ou un service n'a pas de boucle de messages qui exécute la file, il doit donc appeler CheckSynchronize régulièrement. Les valeurs du sparkline sont dans tous les cas ajoutées immédiatement.

La Value d'une carte affiche une virgule comme séparateur décimal

ValueFormat est appliqué avec Format, qui utilise le séparateur décimal des paramètres régionaux de l'application. Le nombre JSON lui-même est toujours lu avec un point. Modifiez ValueFormat, ou définissez la valeur depuis OnValue avec votre propre mise en forme.

Le dashboard FMX ne se compile pas pour Linux64

Les contrôles FMX sous Linux nécessitent FMX pour Linux (FMXLinux) installé dans l'IDE. Sans lui, seules les unités indépendantes du framework se compilent sous Linux64 : le moteur de disposition et le modèle KPI avec sa sortie PNG et SVG headless. Voir Plateformes.

Un contrôle hébergé conserve ses couleurs quand le thème change

Le thème n'atteint un contrôle hébergé que par une propriété publiée Theme, Chart.Theme, BackgroundColor ou Color. L'aperçu sgcReports n'en publie aucune et conserve ses propres couleurs, et les contrôles imbriqués dans un panneau d'un tile ne sont pas thématisés. Theme est aussi ignoré avec utCustom. Voir Thèmes.

La démo Suite ou LiveKPI ne se compile pas

La démo Suite nécessite les sources de sgcCharts, sgcMaps, sgcPDF et sgcReports dans le chemin de recherche, et LiveKPI nécessite les sources de sgcWebSockets. Voir Démos.

Introduction

sgcGrid est une grille de données en direct pour Delphi et C++Builder, pour VCL et FireMonkey. Elle est conçue pour des données qui changent en permanence : les lignes sont mises à jour par clé depuis n'importe quel thread, les cellules modifiées clignotent en vert ou en rouge puis s'estompent, les rafales de mises à jour sont regroupées en repeints cadencés, et le mode queue garde les lignes les plus récentes d'un journal ou d'un flux à l'écran. **TsgcVCLGrid** et **TsgcFMXGrid** trient sur plusieurs colonnes, filtrent, sélectionnent des lignes ou des cellules, redimensionnent les colonnes, appliquent des règles de mise en forme conditionnelle et modifient les cellules sur place.
sgcGrid n'est pas une grille de base de données généraliste avec bandes, regroupements et tableaux croisés. C'est une grille ciblée, vendue sur son débit : elle ne dessine que les lignes visibles, et une image 1080p complète d'une grille de 100 000 lignes prend quelques millisecondes. Voir Performances.
sgcGrid fait partie de la suite visuelle eSeGeCe. Elle est dessinée avec **sgcUICore**, le runtime partagé qui fournit le canevas, le rasteriseur et les thèmes clair et sombre. Ses données implémentent l'interface de table de sgcUICore, IsgcUITable, de sorte qu'un graphique **sgcCharts** peut tracer les lignes d'une grille avec un TsgcChartDataSource_Table, et qu'une grille placée dans une tuile **sgcDashboard** suit le thème du tableau de bord.

Symbol	Price ▼	Change %	Volume	Halted
MSFT	412.50	+1.25	1,200,500	
AAPL	228.10	-0.84	3,400,200	
AMZN	186.40	-2.10	2,250,000	✓
GOOG	171.90	+0.37	980,100	
NVDA	121.70	+3.02	**5,120,750**	

Éditions

sgcGrid est livré en éditions Registered, Trial et Community. L'édition Community n'inclut pas la modification sur place (SGC_GRID_EDIT), ni le pont sgcGridWS avec les flux en direct et la requête IA. Voir Éditions.

Par où commencer

- Architecture : le modèle indépendant du framework, le moteur de rendu sur le canevas sgcUICore, les contrôles VCL et FMX et le modèle de threads.
- Plateformes et Performances : les versions de Delphi, C++Builder, les plateformes et le budget par image.
- Démarrage rapide VCL, Démarrage rapide FMX et Démarrage rapide C++Builder : une première grille en quelques minutes.
- Composants : les grilles, les classes du modèle, les sources de données et la requête IA.
- Mises à jour en direct et threads, Tri, Filtrage, Sélection et clavier et Redimensionnement des colonnes.
- Clignotement à la modification, Mise en forme conditionnelle, Modification sur place et Mode virtuel.
- JSON et CSV, Liaison à un DataSet, Flux en direct, Interroger la grille et Graphiques à partir de la grille.
- Thèmes et Rendu sans fenêtre.
- Démos : les projets de démonstration Delphi et C++Builder.

Architecture

sgcGrid est construite en couches. Chaque couche ne dépend que de celles situées en dessous, de sorte que le modèle, le moteur de rendu et les unités d'import et d'export se compilent sans VCL ni FireMonkey et sont aussi testés sous Linux64.

Couche	Unité	Rôle
Modèle	sgcGrid_Columns, sgcGrid_Data	TsgcGridData contient les colonnes et les lignes : un Variant par colonne, un horodatage et une direction de modification par cellule, indexés par clé de ligne. Il trie, filtre et prend en charge un mode virtuel. Il est thread-safe et implémente IsgcUITable.
Vue de lignes	sgcGrid_Lines, sgcGrid_Group, sgcGrid_Summary	TsgcGridLineView met en correspondance les lignes d'affichage d'une grille (lignes de groupe et lignes de la vue) avec les lignes de la vue des données, TsgcGridGrouper construit les groupes et sgcGrid_Summary calcule les résumés de groupe et de pied. Dans l'édition Registered, sgcGrid_Tree et sgcGrid_Detail ajoutent les lignes d'arbre et les bandes de détail.
Mise en forme	sgcGrid_Format	Le clignotement à la modification et les règles de mise en forme conditionnelle, partagés par tous les moteurs de rendu.
Rende-rer	sgcGrid_Render	TsgcGridRenderer dessine un TsgcGridData sur n'importe quel canevas sgcUICore : en-tête, lignes, lignes de grille, sélection, glyphes de tri, coches et styles de cellule. Seules les lignes visibles sont dessinées. Il écrit aussi des fichiers PNG et SVG sans fenêtre.
Import et export	sgcGrid_IO, sgcGrid_Source_DataSet	Le chargement et l'enregistrement JSON et CSV, et le TsgcGridSource_DataSet qui remplit une grille à partir de n'importe quel TDataSet.
Export Excel	sgcGrid_Zip, sgcGrid_XLSX	L'écrivain XLSX derrière SaveToXLSX, édition Registered uniquement. Voir Export Excel.
Contrôle VCL	sgcGrid_VCL	TsgcVCLGrid possède un TsgcGridData et un moteur de rendu, et ajoute les barres de défilement, la souris, le clavier, le presse-papiers et les éditeurs sur place.
Contrôle FMX	sgcGrid_FMX	TsgcFMXGrid est le contrôle FireMonkey avec les mêmes propriétés, méthodes et événements, des barres de défilement dessinées et le panoramique tactile.
Pont	sgcGridWS_*	Les sources de flux WebSocket, MQTT et SSE et TsgcGridAIQuery. Elles nécessitent sgcWebSockets.
sgcUI-Core	sgcUICore_Canvas, sgcUICore_VCL, sgcUICore_FMX, sgcUICore_Theme, sgcUICore_Table	Le runtime partagé : le canevas vectoriel et le rasteriseur, la base des contrôles VCL et FMX avec cadencement des images (MaxFPS), les jetons de thème et l'interface IsgcUITable.

Modèle de threads

Chaque lecteur et chaque écrivain public de TsgcGridData prend une section critique interne réentrante, de sorte qu'un thread de flux peut appeler directement UpdateRow, AppendRow, UpdateCell et DeleteRow, sans Synchronize. Les données déclenchent leur OnChange dans le thread d'écriture après la libération du verrou. La grille n'y

dessine pas : elle marque son contenu comme à redessiner et la base des contrôles sgcUICore planifie un repeint dans le thread principal, au plus **MaxFPS** fois par seconde (60 par défaut). Les changements de vue, comme une nouvelle ligne ou un nouveau tri, sont postés une seule fois vers le thread principal, où la plage de défilement, la position de la queue et la ligne sélectionnée, retrouvée par sa clé, sont mises à jour.
BeginUpdate et EndUpdate regroupent un nombre quelconque de modifications en un seul OnChange. Les colonnes sont modifiées depuis le thread principal ; tant que des threads d'écriture s'exécutent, encadrez les modifications de colonnes par Lock et Unlock. Voir Mises à jour en direct et threads.

Vue et magasin

Les lignes sont stockées dans l'ordre d'insertion. Sans colonnes de tri et sans filtre, la grille affiche le magasin tel quel. Sinon, une vue est reconstruite paresseusement à la première lecture après une modification : le filtre, puis un tri multicolonne stable. Le changement d'une colonne non triée, sans filtre actif, conserve la vue courante, de sorte qu'une colonne de prix qui varie ne coûte aucun nouveau tri.

Vue de lignes

Les grilles dessinent et font défiler des lignes d'affichage, pas des lignes de la vue. Chaque grille possède un TsgcGridLineView sur ses données. Sans colonne regroupée, la vue de lignes est l'identité, la ligne i est la ligne de la vue i, et rien n'est construit. Le regroupement et, dans l'édition Registered, les lignes d'arbre et les bandes de détail sont un état de la vue de lignes : les données, leurs lignes de la vue et IsgcUITable restent à plat, de sorte que les graphiques, les grilles pivot et le pont voient les mêmes lignes qu'avant. La vue de lignes se reconstruit paresseusement lorsque le LayoutVersion des données bouge, et n'actualise que les résumés de groupe lorsque seules des valeurs ont changé. Voir Regroupement.

Voir aussi

- Plateformes
- Performances
- TsgcGridData
- TsgcGridRenderer
- TsgcGridLineView

Éditions

sgcGrid est construite à partir des mêmes sources pour chaque édition. L'édition ne change que les fonctionnalités compilées dans un build et les unités livrées.

Édition	Inclut
Registered	Tout : les grilles VCL et FMX, le modèle, le tri, le filtrage, le regroupement avec le panneau de regroupement, les résumés de groupe et de pied, les colonnes figées, les lignes d'arbre, les lignes maître-détail, l'export Excel, la sélection, le clignotement, les règles de format, l'édition sur place, le mode virtuel, JSON et CSV, la liaison à un DataSet, les thèmes, la sortie PNG et SVG sans fenêtre, et le pont sgcGridWS avec les sources de flux WebSocket, MQTT et SSE et la requête IA.
Trial	Les mêmes fonctionnalités que Registered. Un build compilé avec SGC_TRIAL affiche un avis indiquant qu'il s'agit d'une version d'essai une fois passée la période de publication indiquée dans le build. Rien d'autre ne change.
Community	Tout sauf l'édition sur place, les lignes d'arbre, les lignes maître-détail, l'export Excel et le pont sgcGridWS.

Ce que l'édition Community n'inclut pas

- La modification sur place. L'édition Community se compile avec SGC_GRID_EDIT non défini, de sorte que StartEdit, CommitEdit, CancelEdit, EditorActive et les événements OnBeforeEdit, OnValidate, OnEdited et OnEditError n'existent pas, et ReadOnly à False n'a pas d'éditeur. Voir Modification sur place.
- Lignes d'arbre, lignes maître-détail et export Excel. L'édition Community se compile avec SGC_GRID_ADVANCED non défini, de sorte que les propriétés Tree et Detail, ExpandNode, CollapseNode, ExpandDetail, CollapseDetail, DetailExpanded, DetailData, SaveToXLSX, les événements OnTreeExpandChange, OnDetailExpandChange, OnGetDetailData et OnDetailCellClick et le composant TsgcGridDataStore n'existent pas. Voir Lignes d'arbre, Maître-détail et Export Excel.
- Le pont sgcGridWS : les unités sgcGridWS_*.pas et leurs packages ne sont pas livrés, de sorte que les sources de flux et TsgcGridAIQuery ne sont pas disponibles. Voir Flux en direct et Interroger la grille.

Le regroupement, le panneau de regroupement, les résumés de groupe et de pied et les colonnes figées font partie de toutes les éditions, Community comprise. Voir Regroupement, Résumés et Colonnes figées.
L'édition est un choix à la compilation dans sgcVer.inc (SGC_COMMUNITY, qui désactive aussi SGC_TRIAL), pas un interrupteur à l'exécution. Le code qui utilise l'API d'édition, les membres d'arbre, de détail et d'Excel ou le pont ne se compile pas avec une build Community, voir Dépannage.

Installation

sgcGrid fait partie de la distribution sgcUI. Un seul installateur couvre les produits sgcUI (sgcUICore, sgcCharts, sgcPDF, sgcReports, sgcMaps, sgcDashboard, sgcGrid, sgcNavigation, sgcToolbar et sgcWizard) et n'installe que les produits inclus dans votre licence.

Installation avec l'installateur sgcUI

1. Lancez l'installateur sgcUI et connectez-vous avec le nom d'utilisateur et le mot de passe de votre compte eSeGeCe.
2. Choisissez les versions de Delphi et de C++Builder dans lesquelles installer.
3. Sur la page des composants, sélectionnez **sgcGrid** et tout autre produit souhaité. Le serveur de licences renvoie les produits que vous possédez, et l'installateur affiche une case à cocher par produit. Les produits que votre licence n'inclut pas sont grisés. Une licence All-Access déverrouille tous les produits.
4. L'installateur copie les fichiers, puis compile les packages sur votre machine pour chaque IDE sélectionné, dans l'ordre des dépendances, et installe les packages de conception.

Ce qui est fourni avec sgcGrid

- **sgcUICore**, la base commune des produits de mise en page et de sgcCharts. Il n'a pas de case à cocher propre. L'installateur l'installe automatiquement dès que sgcGrid est sélectionné.
- **sgcGrid pour sgcWebSockets**, listé sous **sgcWebSockets bridges**. Il installe les packages sgcGridWS avec les sources de flux et TsgcGridAIQuery (voir Flux en direct). L'installateur ne le propose que si sgcWebSockets est déjà installé sur la machine et que votre licence inclut sgcGrid. Sinon, il est grisé.

Dossiers installés

Tous les produits s'installent dans un seul dossier, par défaut `Documents\esegece\sgcUI` (`sgcUI_Trial` pour la version d'essai et `sgcUI_Community` pour l'édition Community).

Dossier	Contenu
`Source`	Le code source de chaque produit installé. C'est l'unique chemin de bibliothèque de sgcUI.
`Packages`	Les packages de chaque produit installé, un jeu par produit et par version d'IDE.
`Lib<V>`	Les unités compilées pour chaque version d'IDE et chaque plateforme, par exemple `LibD13` .
`Demos\Grid`	Les démos de sgcGrid. Voir Démos.

L'installateur ajoute `sgcUI\Source` au chemin de bibliothèque et au chemin de navigation de chaque IDE et plateforme sélectionnés, ainsi que le dossier des unités compilées. sgcUICore se trouve dans le même dossier `Source`, il n'a donc besoin d'aucun chemin propre.

Trial et Community

- **Trial** : l'installateur Trial de sgcUI installe tous les produits en mode d'essai, y compris sgcGrid.
- **Community** : l'installateur Community gratuit de sgcUI inclut sgcGrid, avec sgcCharts, sgcDashboard, sgcNavigation, sgcToolbar et sgcWizard. Les ponts sgcWebSockets ne font pas partie de l'installateur Community, et sgcPDF et sgcMaps ne font pas partie de Community. Voir Éditions pour ce que la version Community laisse de côté.

Installations autonomes précédentes

Si l'installateur trouve une ancienne installation autonome de sgcUICore, comme celle créée par l'installateur autonome de sgcCharts, il liste les chemins de bibliothèque et les packages de conception de cette installation sur la page **Previous installations**. Quand vous continuez, il les supprime de l'IDE avant de compiler les nouveaux packages, de sorte qu'il n'y ait jamais deux copies d'une unité dans le chemin.

Compiler les packages manuellement

Pour compiler vous-même les packages, ouvrez-les pour votre version d'IDE depuis `sgcUI\Packages`. `<V>` est le suffixe de version Delphi, par exemple `sgcGridD13.dpk`.

Package	Contenu
`sgcUICore<V>`	La base commune sgcUICore (exécution).
`sgcGrid<V>` / `dclsgcGrid<V>`	Les composants sgcGrid (exécution / conception).
`sgcGridWS<V>` / `dclsgcGridWS<V>`	Le pont sgcWebSockets (exécution / conception). Nécessite sgcWebSockets.

Compilez-les dans l'ordre des dépendances : `sgcUICore`, puis `sgcGrid`, puis `dclsgcGrid`, et enfin les packages du pont. Installez ensuite les packages de conception `dcl*` et ajoutez `sgcUI\Source` au chemin de bibliothèque de votre version de Delphi (Tools > Options > Language > Delphi Options > Library > Library path). Pour C++Builder, ouvrez le projet `.cbproj` correspondant à votre version, par exemple `sgcGridC13.cbproj`, et compilez-le. Les en-têtes `.hpp` générés apparaissent à côté du package compilé.

.NET

Rien ne change pour .NET. sgcGrid reste distribué sous forme de ses propres packages NuGet (`esegece.sgcGrid` et ses packages compagnons). Voir l'installation .NET.

Plateformes

Cible	Prise en charge
Delphi	Delphi 7, 2007, 2009, 2010, XE à XE8, 10 à 10.4, 11, 12 et 13, avec des packages d'exécution indépendants du framework, VCL et FireMonkey et des packages de conception VCL et FireMonkey, voir Packages ci-dessous.
C++Builder	C++Builder 2007 à C++Builder 13, avec les en-têtes .hpp générés et les bibliothèques statiques. Voir Démarrage rapide C++Builder.
VCL	TsgcVCLGrid sous Windows, Win32 et, à partir des versions qui la prennent en charge, Win64.
FireMonkey	TsgcFMXGrid à partir de Delphi XE6 (SGC_GRID_FMX), sous Windows, macOS, iOS et Android.
Linux64	Les unités indépendantes du framework (le modèle, la mise en forme, le moteur de rendu et l'import et l'export JSON et CSV) à partir de Delphi 10.2 (SGC_GRID_LINUX), par exemple pour rendre des images de grille sur un serveur.
Pont	Les packages sgcGridWS nécessitent le package sgcGrid et le package sgcWebSockets de la même version de Delphi ou de C++Builder.
.NET	Un portage C# avec la même API pour WinForms, WPF et Avalonia, la sortie PNG et SVG sans fenêtre, la source DataTable et les ponts sgcWebSockets et sgcHTML, sur .NET Framework 4.7.2 et .NET 8 et 9, voir Vue d'ensemble de .NET.

Packages

sgcGrid fournit un package d'exécution indépendant du framework et un package d'exécution par framework, de sorte qu'une application FireMonkey pour Android, iOS ou macOS ne lie jamais la VCL. Chaque nom de package se termine par le suffixe de l'IDE, par exemple sgcGridD13.dpk pour Delphi 13 et sgcGridC13.cbproj pour C++Builder 13.

Package	Type	Contenu	Requiert
sgcGrid	Exécution, indépendant du framework	Chaque unité du produit qui n'utilise ni la VCL ni FireMonkey : le modèle, les colonnes, le formatage, le moteur de rendu, le regroupement, les synthèses, les lignes d'arborescence et de détail, l'import et l'export JSON, CSV et XLSX, TsgcGridDataStore et TsgcGridSource_DataSet.	rtl, dbrtl, sgcUICore
sgcGridVCL	Exécution, VCL	sgcGrid_VCL (TsgcVCLGrid)	rtl, vcl, sgcUICore, sgcUICoreVCL, sgcGrid
sgcGridFMX	Exécution, FireMonkey, à partir de XE6	sgcGrid_FMX (TsgcFMXGrid)	rtl, fmx, sgcUICore, sgcUICoreFMX, sgcGrid
dclsgcGrid	Conception, VCL	Enregistre TsgcGridSource_DataSet, TsgcGridDataStore (pas dans l'édition Community) et TsgcVCLGrid dans la page de pa-	designide, rtl, vcl, sgc-

		lette **SGC Grid**, ainsi qu'un éditeur de propriété pour la Source des options de détail, qui ne liste que les composants contenant des données de grille.	Grid, sgcGridVCL
dclsgcGridFMX	Conception, FireMonkey, à partir de XE6	Enregistre TsgcFMXGrid dans la page de palette **SGC Grid**.	designide, rtl, fmx, sgcGrid, sgcGridFMX, dclsgcGrid
sgcGridWS, dclsgcGridWS	Exécution et conception, passerelle sgcWebSockets	Les sources de flux en direct (TsgcGridFeedSource_WebSocket, TsgcGridFeedSource_MQTT, TsgcGridFeedSource_SSE) et TsgcGridAIQuery, indépendants du framework, dans la page de palette **SGC Grid**.	rtl, sgcGrid, sgcWebSockets

sgcUICore suit la même répartition : sgcUICore, sgcUICoreVCL et sgcUICoreFMX, sans package de conception. dclsgcGridFMX requiert dclsgcGrid, installez donc d'abord dclsgcGrid. L'installateur installe sgcUICore avec sgcGrid. Si vous compilez les packages vous-même, compilez d'abord sgcUICore.

Versions de l'IDE et plateformes cibles

Delphi	sgcGrid	sgcGridVCL	sgcGridFMX	dclsgcGrid, dclsgcGridFMX
7, 2007, 2009, 2010, XE	Win32	Win32	Aucune	Win32
XE2 à XE5	Win32, Win64	Win32, Win64	Aucune	Win32
XE6 à 10.2	Win32, Win64	Win32, Win64	Win32, Win64	Win32
10.3	Win32, Win64, Linux64, OSX64	Win32, Win64	Win32, Win64, OSX64	Win32
10.4	Win32, Win64, Linux64, OSX64, Android, Android64, iOSDevice64	Win32, Win64	Win32, Win64, OSX64, Android, Android64, iOSDevice64	Win32
11	Win32, Win64, Linux64, OSX64, OSXARM64, Android, Android64, iOSDevice64, iOSSimARM64	Win32, Win64	Win32, Win64, OSX64, OSXARM64, Android, Android64, iOSDevice64, iOSSimARM64	Win32
12, 13	Comme 11	Win32, Win64	Comme 11	Win32, Win64

De XE6 à 10.2, le package FireMonkey est compilé pour Windows uniquement. Les packages de conception sont chargés par l'IDE, ils sont donc compilés pour la plateforme de l'IDE uniquement. sgcGridWS est compilé pour les mêmes plateformes que sgcGrid, et dclsgcGridWS pour les mêmes plateformes que dclsgcGrid.

C++Builder	sgcGrid, sgcGridVCL	sgcGridFMX	dclsgcGrid, dclsgcGridFMX
2007, 2010, XE, XE2	Win32	Aucune	Win32 (dclsgcGrid uniquement)

XE3 à XE5	Win32, Win64	Aucune	Win32 (dclsgcGrid uniquement)
XE6 à 12	Win32, Win64	Win32, Win64	Win32
13	Win32, Win64, Win64x	Win32, Win64, Win64x	Win32, Win64x

Les projets C++Builder portent les mêmes noms avec le suffixe C, par exemple sgcGridC13.cbproj, sgcGridVCLC13.cbproj, sgcGridFMXC13.cbproj, dclsgcGridC13.cbproj et dclsgcGridFMXC13.cbproj. Les packages C++Builder sont compilés pour Windows uniquement. sgcGridWS et dclsgcGridWS sont compilés pour les mêmes versions de C++Builder et les mêmes plateformes que sgcGrid et dclsgcGrid, sauf Win64x, car ils nécessitent une version Win64x du package sgcWebSockets. Les notes C++Builder qui s'appliquent à chaque produit sgcUI (packages d'exécution sous Win64, C++Builder 10.3 sous Win64, noms de fichiers des packages Win64x) se trouvent dans le sujet Plateformes et packages de l'aide de la suite sgcUI.

FireMonkey sur Android, iOS et macOS

Les contrôles FireMonkey (TsgcFMXGrid) ciblent macOS à partir de Delphi 10.3, Android et iOS à partir de Delphi 10.4, et macOS ARM et le simulateur iOS à partir de Delphi 11. Dans un projet FireMonkey pour ces plateformes :

- Utilisez les contrôles FireMonkey. Les contrôles VCL et sgcGridVCL ne fonctionnent que sous Windows.
- Déposez le contrôle sur un formulaire, ou ajoutez son unité à la clause uses : sgcGrid_FMX.
- Sélectionnez la plateforme cible dans la fenêtre Projets. L'installateur ajoute le dossier Source de sgcUI et les unités compilées de chaque plateforme (LibD<version>\<platform>) au chemin de bibliothèque de chaque plateforme qu'il compile. Sans l'installateur, ajoutez le dossier Source de sgcUI au chemin de bibliothèque de la plateforme cible (Outils, Options, Langage, Delphi, Bibliothèque).
- Les applications Android et iOS sont liées statiquement, aucun package d'exécution n'est donc déployé.

Linux64

Le package indépendant du framework sgcGrid est compilé pour Linux64 à partir de Delphi 10.3. Une application Linux64, par exemple un programme console ou un serveur, utilise le modèle, le formatage, le moteur de rendu et l'import et l'export JSON et CSV, par exemple pour rendre des images de grille. Aucun package VCL ou FireMonkey n'est compilé pour Linux64, car FireMonkey sous Linux nécessite FMX pour Linux (FMXLinux), un module complémentaire tiers.

Voir aussi

- Éditions
- Architecture

Performances

La grille ne dessine que les lignes et les colonnes visibles, prend le verrou des données une seule fois par image, met en cache la géométrie des colonnes et dessine le texte des cellules à partir d'un cache d'images de glyphes sur le canevas raster. Le coût d'une image dépend de la taille de la fenêtre, pas du nombre de lignes.

Budget par image

La suite QA comporte un seuil de performance : une image de 1920 x 1080 avec 12 colonnes, 100 000 lignes et 45 lignes visibles, rendue 60 fois sur le canevas raster. Le seuil échoue au-delà de 16 ms par image sous Win64 et 25 ms sous Win32. Les dernières moyennes mesurées étaient :

Plateforme	Image simple	Clignotement et 3 règles de mise en forme
Win32	3.0 ms	3.2 ms
Win64	4.6 ms	6.8 ms
Linux64	3.7 ms	

Une grille 1080p est donc rendue en quelques millisecondes, bien en deçà des 16,7 ms d'une image à 60 ips.

Cadence des mises à jour

Les mises à jour et les images sont découplées. Les threads d'écriture peuvent appeler UpdateRow aussi souvent que le flux livre des données, et la grille se repeint au plus MaxFPS fois par seconde, quel que soit le nombre de lignes modifiées entre-temps. La démo LiveTicker écrit environ 1 000 mises à jour de prix par seconde depuis un thread de travail dans une grille triée qui clignote à 60 ips.

Conseils

- Encadrez un lot de modifications par BeginUpdate et EndUpdate, afin que la grille reçoive une seule notification.
- Trier sur une colonne qui varie retrie la vue à chaque changement de cette colonne. Les changements de colonnes non triées, sans filtre, conservent la vue courante.
- Utilisez MaxRows pour les journaux et les flux qui grandissent sans fin, et le mode virtuel pour de très grandes données qui vivent déjà ailleurs.
- Réduisez MaxFPS lorsque la grille doit seulement sembler en direct, pas fluide.

Voir aussi

- Mises à jour en direct et threads
- Mode virtuel
- Architecture

Démarrage rapide VCL

Au moment de la conception

- Déposez un **TsgcVCLGrid** depuis la page de palette sgcGrid sur une fiche et définissez Align sur alClient.
- Ouvrez l'éditeur **Columns** et ajoutez une colonne par champ. Définissez FieldName, Caption et DataType, et DisplayFormat pour les nombres et les dates.
- Remplissez les lignes depuis le code avec Data.AppendRow et gardez-les à jour avec Data.UpdateRow, depuis n'importe quel thread.

Depuis le code

```
uses
  Forms, Classes, Controls,
  sgcGrid_Columns, sgcGrid_Data, sgcGrid_VCL;

procedure CreateGrid(aForm: TForm);
var
  oGrid: TsgcVCLGrid;
begin
  oGrid := TsgcVCLGrid.Create(aForm);
  oGrid.Parent := aForm;
  oGrid.Align := alClient;

  // the columns: FieldName, Caption and DataType
  oGrid.Columns.Add('Symbol', 'Symbol');
  oGrid.Columns.Add('Price', 'Price', gdtFloat).DisplayFormat := '0.00';
  oGrid.Columns.Add('Volume', 'Volume', gdtInteger);

  // one row per key, the values in column order
  oGrid.Data.AppendRow('MSFT', ['MSFT', 412.5, 1200]);
  oGrid.Data.AppendRow('AAPL', ['AAPL', 228.1, 3400]);

  // later, from any thread: update the row of a key by field name
  oGrid.Data.UpdateRow('MSFT', ['Price', 'Volume'], [413.1, 1350]);
end;
```

AppendRow prend les valeurs dans l'ordre des colonnes ; UpdateRow prend des noms de champs, de sorte qu'un message qui ne porte qu'un prix ne met à jour que le prix. Une clé qui n'existe pas encore est aussi insérée par UpdateRow, sauf si Data.AutoInsert vaut False.

Voir aussi

- Démarrage rapide FMX
- Mises à jour en direct et threads
- TsgcVCLGrid

Démarrage rapide FMX

TsgcFMXGrid nécessite Delphi XE6 ou version ultérieure. Déposez-le sur une fiche FireMonkey, ou créez-le depuis le code. Il a les mêmes colonnes, données et événements que la grille VCL.

```
uses
  System.Classes, FMX.Forms, FMX.Types,
  sgcUICore_Theme, sgcGrid_Columns, sgcGrid_Data, sgcGrid_FMX;

procedure CreateGrid(aForm: TForm);
var
  oGrid: TsgcFMXGrid;
begin
  oGrid := TsgcFMXGrid.Create(aForm);
  oGrid.Parent := aForm;
  oGrid.Align := TAlignLayout.Client;
  oGrid.Theme := utDark;

  oGrid.Columns.Add('Symbol', 'Symbol');
  oGrid.Columns.Add('Price', 'Price', gdtFloat).DisplayFormat := '0.00';
  oGrid.Columns.Add('Volume', 'Volume', gdtInteger);

  oGrid.Data.AppendRow('MSFT', ['MSFT', 412.5, 1200]);
  oGrid.Data.AppendRow('AAPL', ['AAPL', 228.1, 3400]);
  oGrid.Data.UpdateRow('MSFT', ['Price'], [413.1]);
end;
```

FireMonkey n'a pas de barres de défilement natives sur un contrôle raster, la grille dessine donc les siennes. La molette de la souris défile de trois lignes, Maj défile horizontalement, et un geste de panoramique défile avec le doigt sur les écrans tactiles.

Voir aussi

- Démarrage rapide VCL
- TsgcFMXGrid

Démarrage rapide C++Builder

C++Builder utilise les mêmes composants grâce aux en-têtes et aux bibliothèques statiques générés par les packages C++Builder. Ajoutez `sgcUI\Source` et `sgcUI\LibD13\win32` (`sgcUI\LibD13\win64` pour le 64 bits) aux chemins d'inclusion et de bibliothèques, ce que l'installateur fait déjà pour l'IDE. Les en-têtes et les bibliothèques de sgcGrid et de sgcUICore se trouvent dans ce seul dossier. Incluez les en-têtes sgcGrid_*.hpp et liez sgcGridC13.lib et sgcUICoreC13.lib, ou installez les packages de conception et déposez un TsgcVCLGrid sur une fiche.
Les tableaux ouverts de Delphi se passent avec la macro OPENARRAY. Les appels AppendRow et UpdateRow sont thread-safe depuis C++ également.

Voir aussi

- Démarrage rapide VCL
- Démos
- Dépannage

Démos

Les démos se trouvent dans le dossier Demos de l'installation : les projets Delphi sous Demos\Delphi et les projets C++Builder sous Demos\CBuilder. Chaque dossier de démo contient un readme.html. Chaque démo accepte aussi deux options utilisées par la vérification des builds : /test exécute ses contrôles et se termine avec le nombre d'échecs, et /snapshot enregistre la fenêtre dans un fichier PNG.

Démo	Delphi	C++Builder	Montre
LiveTicker	Oui	Oui	Environ 1 000 mises à jour de prix par seconde depuis un thread de travail avec UpdateRow, une vue triée par prix, des modifications qui clignotent, une règle de mise en forme pour les variations négatives, des commutateurs TailMode et thème sombre, et les mises à jour et images par seconde.
LiveTickerFMX	Oui	Non	La démo LiveTicker avec un TsgcFMXGrid.
DataSetGrid	Oui	Oui	Un TsgcGridSource_DataSet lié à un TClientDataSet de 200 enregistrements avec KeyField, AutoColumns et LiveUpdates, une sélection qui suit l'enregistrement courant dans les deux sens, et l'export et l'import JSON et CSV.
Editing	Oui	Oui	Modification sur place avec une colonne à liste de choix, une colonne booléenne, OnValidate, OnEditError et OnEdited. Absente de l'édition Community.
GridAndChart	Oui	Oui	Un tableau de bord avec une tuile grille et une tuile graphique ; un TsgcChartDataSource_Table lié à la grille trace ses lignes. Nécessite sgcCharts et sgcDashboard.
AskYourGrid	Oui	Oui	TsgcGridAIQuery filtre, trie et met en évidence 200 actions à partir de questions en langage courant, avec l'analyseur à base de règles. Nécessite sgcWebSockets.
LiveFeed	Oui	Non	Un TsgcGridFeedSource_WebSocket alimenté par un serveur WebSocket dans le même processus. Nécessite sgcWebSockets.

Notes C++Builder

- Les démos C++Builder se compilent avec le compilateur classique de C++Builder 13 pour Win32 et Win64 et se lient à sgcGridC13.lib et sgcUICoreC13.lib.
- DataSetGrid lie MIDAS statiquement sous Win32. C++Builder n'a pas de bibliothèque MIDAS statique pour Win64, le build 64 bits nécessite donc midas.dll à côté de l'exécutable.
- GridAndChart se lie aussi à sgcChartsC13.lib et sgcDashboardC13.lib, et AskYourGrid se lie à sgcGridWSC13.lib et sgcWebSocketsC13.lib.

Voir aussi

- Démarrage rapide VCL
- Démarrage rapide C++Builder
- Dépannage

TsgcVCLGrid

La grille de données en direct VCL, dans l'unité sgcGrid_VCL. Elle possède un TsgcGridData et un TsgcGridRenderer, dessine avec le canevas raster de sgcUICore et implémente IsgcUITable (délégué à Data) et IsgcGridDataHost, de sorte qu'une source de données de graphique, une source de flux ou une source DataSet peut se lier au composant grille lui-même.
Il utilise les barres de défilement natives de Windows : la barre verticale compte des lignes (des lignes d'affichage tant que les lignes sont regroupées) et la barre horizontale des pixels logiques, et la ligne d'en-tête reste fixe. Les membres d'édition n'existent que lorsque SGC_GRID_EDIT est défini.

Gestionnaires d'événements des exemples de fonctionnalités

Les exemples des pages Fonctionnalités affectent les méthodes de cette petite classe aux événements.

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

type
  TGridHandlers = class
  public
    procedure FilterRow(Sender: TObject; aRow: Integer; var aAccept: Boolean);
    procedure SelectionChange(Sender: TObject);
    procedure CellClick(Sender: TObject; aRow, aCol: Integer);
    procedure ColumnResized(Sender: TObject; aCol: Integer);
    procedure GetCellStyle(Sender: TObject; aRow, aCol: Integer;
      var aBack, aFore: TsgcUIColor; var aBold: Boolean);
    procedure GetValue(Sender: TObject; aRow, aCol: Integer;
      var aValue: Variant);
    procedure Validate(Sender: TObject; aRow, aCol: Integer;
      var aValue: Variant; var aAccept: Boolean);
    procedure Edited(Sender: TObject; aRow, aCol: Integer;
      const aOldValue, aNewValue: Variant);
    procedure EditError(Sender: TObject; aRow, aCol: Integer;
      const aText: string; var aMessage: string);
  end;
```

Propriétés

Propriété	Type	Description
Columns	TsgcGridColumns	Les colonnes de la grille. Voir TsgcGridColumns.
Data	TsgcGridData	Les lignes, le tri, le filtre et le suivi des modifications. Publique. Elle implémente aussi IsgcUITable pour la grille. Voir TsgcGridData.
Renderer	TsgcGridRenderer	Le moteur de rendu avec lequel la grille dessine. Publique, en lecture seule.
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom.
AccentColor	TsgcUIColor	L'accent de la sélection et du clignotement des modifications. 0 utilise l'accent du thème.

FontSize	Double	La taille du texte des cellules.
HeaderHeight, RowHeight	Integer	La hauteur de l'en-tête (32) et de chaque ligne (28), en pixels logiques.
GridLinesH, GridLinesV	Boolean	Dessinent les lignes de grille horizontales et verticales. True par défaut.
AlternateRows	Boolean	Teinte une ligne sur deux. True par défaut.
SelectionMode	TsgcGridSelectionMode	gsmRow (par défaut) sélectionne des lignes entières, gsmCell sélectionne des cellules individuelles.
BorderStyle	TBorderStyle (VCL), TsgcGridBorderStyle (FMX)	Un cadre simple dans la couleur Border du thème, ou aucun.
MaxRows	Integer	Mode queue : 0 conserve toutes les lignes, une valeur supérieure à 0 supprime les plus anciennes lors d'une insertion.
TailMode	Boolean	Garde la ligne la plus récente visible tant que la grille est défilée jusqu'en bas.
FilterText	string	Affiche uniquement les lignes dont le texte visible le contient, sans tenir compte de la casse.
ReadOnly	Boolean	True par défaut. False active la modification sur place des colonnes Editable (SGC_GRID_EDIT).
FlashChanges, FlashDuration, FlashMode	Boolean, Integer, TsgcGridFlashMode	Fait clignoter les cellules modifiées, pendant combien de millisecondes (800), sur l'arrière-plan ou sur le texte.
FlashUpColor, FlashDownColor, FlashChangedColor	TsgcUIColor	Les couleurs de clignotement ; 0 utilise les valeurs par défaut du thème.
FormatRules	TsgcGridFormatRules	Les règles de mise en forme conditionnelle. Voir Mise en forme conditionnelle.
MaxFPS	Integer	Le plus grand nombre de repeints par seconde (60).
FrameCount	Cardinal	Le nombre d'images dessinées jusqu'à présent. Publique, en lecture seule.
TopRow, ScrollX	Integer	La première ligne visible (la première ligne d'affichage tant que les lignes sont regroupées) et la position de défilement horizontale. Publique.
SelectedRow, SelectedCol	Integer	La ligne de vue et l'index de colonne sélectionnés, -1 si aucun. Publics.
SelectedKey	string	La clé de la ligne sélectionnée. La sélection suit sa clé lorsque la vue change. Publique, en lecture seule.
FrozenColumns	Integer	Le nombre de colonnes visibles qui ne défilent pas horizontalement, 0 par défaut. Voir Colonnes figées.
ShowFooter	Boolean	Affiche la ligne de pied avec les résumés des colonnes, False par défaut. Voir Résumés.

ShowGroupPanel, GroupPanelHeight	Boolean, Integer	Affiche le panneau de regroupement au-dessus de l'en-tête (False par défaut) et sa hauteur en DIP (36). Voir Regroupement.
GroupIndent	Integer	Le décalage de chaque niveau de groupe en DIP, 20 par défaut.
GroupsExpanded	Boolean	L'état des nouveaux groupes, True par défaut. L'affecter conserve l'état des groupes actuels.
GroupSummaries	Boolean	Affiche les résumés des colonnes dans les lignes de groupe, True par défaut.
Lines	TsgcGridLineView	Les lignes d'affichage : lignes de groupe et lignes de la vue. Publique, en lecture seule. Voir TsgcGridLineView.
FocusedLine	Integer	La ligne d'affichage ayant le focus, -1 si aucune. Sur une ligne de groupe, SelectedRow vaut -1. Publique.
Tree	TsgcGridTreeOptions	Lignes d'arbre : KeyField, ParentField, TreeField, Indent et Expanded. Le mode arbre est actif lorsque KeyField et ParentField nomment des colonnes de la grille. Édition Registered uniquement. Voir Lignes d'arbre.
Detail	TsgcGridDetailOptions	Lignes maître-détail : Active, Source, MasterKeyField, DetailKeyField, Lines et Columns. Édition Registered uniquement. Voir Maître-détail.

Méthodes

Méthode	Retourne	Description
ScrollToRow(aRow)		Fait défiler la vue jusqu'à la ligne de vue indiquée.
RowAt(X, Y), ColAt(X, Y)	Integer	La ligne de vue et l'index de colonne situés sous un point, -1 si aucun.
SaveToPNG(aFileName, aWidth, aHeight)		Écrit la vue courante dans un fichier PNG.
SaveToSVG(aFileName, aWidth, aHeight)		Écrit la vue courante dans un fichier SVG.
StartEdit(aRow, aCol)	Boolean	Ouvre l'éditeur sur une cellule, ou bascule une cellule booléenne (SGC_GRID_EDIT).
CommitEdit	Boolean	Convertit, valide et écrit le texte de l'éditeur ; False garde l'éditeur ouvert (SGC_GRID_EDIT).
CancelEdit		Ferme l'éditeur sans écrire (SGC_GRID_EDIT).
EditorActive, Editor, EditRow, EditCol		L'état de l'éditeur ouvert (SGC_GRID_EDIT). Propriétés.
GroupBy(aFieldNames), ClearGrouping		Regroupe selon les champs indiqués (FieldName, niveau 0 d'abord ; les champs inconnus sont ignorés), ou supprime le regroupement.

ExpandAll, CollapseAll		Déploie ou replie chaque groupe, et les nouveaux groupes à partir de maintenant. En mode arbre, ils agissent sur les nœuds de l'arbre (édition Registered).
ExpandGroup(aGroup), CollapseGroup(aGroup)		Déploie ou replie un seul groupe.
GroupCount	Integer	Le nombre de groupes.
GroupInfo(aGroup)	TsgcGridGroupInfo	Niveau, parent, colonne, membre, nombre de lignes, état et ligne d'un groupe.
FooterSummary(aCol), FooterSummaryText(aCol)	Variant, string	La valeur de pied d'une colonne (index de Columns) et son texte ; Null et une chaîne vide lorsque la colonne n'a pas de SummaryType.
ExpandNode(aRow), CollapseNode(aRow)		Déploie ou replie le nœud d'arbre de la ligne de la vue aRow et déclenche OnTreeExpandChange. Édition Registered uniquement.
ExpandDetail(aRow), CollapseDetail(aRow)		Ouvre ou ferme la bande de détail de la ligne maître de la vue aRow et déclenche OnDetailExpandChange. Édition Registered uniquement.
DetailExpanded(aRow)	Boolean	True lorsque la bande de la ligne de la vue aRow est ouverte. Édition Registered uniquement.
DetailData(aRow)	TsgcGridData	Les lignes de la bande de la ligne de la vue aRow, nil lorsqu'elle est repliée. Lecture seule. Édition Registered uniquement.
SaveToXLSX(aStream; aSheetName = 'Sheet1'), SaveToXLSX(aFileName; aSheetName = 'Sheet1')		Écrit la grille sous forme de classeur Excel avec les largeurs de colonne actuelles et FrozenColumns. Édition Registered uniquement. Voir Export Excel.

Événements

Événement	Type	Description
OnCellClick, OnCellDblClick	TsgcGridCellEvent	Une cellule a été cliquée ou double-cliquée : ligne de vue et index de colonne.
OnSelectionChange	TNotifyEvent	La ligne ou la cellule sélectionnée a changé.
OnSortChange	TNotifyEvent	Un clic sur l'en-tête a changé le tri.
OnColumnResized	TsgcGridColumnEvent	Une colonne a été redimensionnée à la souris.
OnGetCellStyle	TsgcGridCellStyleEvent	Dernier mot sur les couleurs et le style gras de chaque cellule dessinée. Voir Mise en forme conditionnelle.
OnBeforeEdit	TsgcGridBeforeEditEvent	Un éditeur est sur le point de s'ouvrir ; définissez aAllow sur False pour le refuser (SGC_GRID_EDIT).

OnValidate	TsgcGridValidateEvent	La valeur convertie avant son écriture ; modifiez-la ou rejetez-la (SGC_GRID_EDIT).
OnEdited	TsgcGridEditedEvent	Une valeur a été écrite, avec l'ancienne et la nouvelle valeur (SGC_GRID_EDIT).
OnEditError	TsgcGridEditErrorEvent	Le texte n'a pas pu être converti vers le DataType de la colonne ; aMessage peut être modifié (SGC_GRID_EDIT).
OnGroupingChange	TNotifyEvent	Les colonnes regroupées ou leurs niveaux ont changé, depuis le panneau de regroupement ou depuis le code.
OnGroupExpandChange	TsgcGridGroupExpandEvent	Un groupe a été déployé ou replié ; aGroup vaut -1 après ExpandAll et CollapseAll.
OnGetGroupText	TsgcGridGetGroupTextEvent	Modifie le libellé aText de la ligne de groupe aGroup. Déclenché avec le verrou des données détenu.
OnTreeExpandChange	TsgcGridRowExpandEvent	Un nœud d'arbre a été déployé ou replié ; aRow vaut -1 après ExpandAll et CollapseAll en mode arbre. Édition Registered uniquement.
OnDetailExpandChange	TsgcGridRowExpandEvent	Une bande de détail a été ouverte ou fermée ; aRow est la ligne maître de la vue. Édition Registered uniquement.
OnGetDetailData	TsgcGridGetDetailDataEvent	Remplit les lignes d'une bande (aData) pour la clé maître aMasterKey ; affectez True à aHandled pour les conserver. Sender est la grille. Déclenché sans verrou détenu. Édition Registered uniquement.
OnDetailCellClick	TsgcGridDetailCellEvent	Une cellule d'une bande a reçu un clic : aRow est la ligne maître de la vue, aDetailRow et aDetailCol la ligne de la vue et l'index de Columns des données de la bande. Édition Registered uniquement.

Exemple

```
uses
  Forms, Classes, Controls,
  sgcGrid_Columns, sgcGrid_Data, sgcGrid_VCL;

procedure CreateGrid(aForm: TForm);
var
  oGrid: TsgcVCLGrid;
begin
  oGrid := TsgcVCLGrid.Create(aForm);
  oGrid.Parent := aForm;
  oGrid.Align := alClient;

  // the columns: FieldName, Caption and DataType
  oGrid.Columns.Add('Symbol', 'Symbol');
  oGrid.Columns.Add('Price', 'Price', gdtFloat).DisplayFormat := '0.00';
  oGrid.Columns.Add('Volume', 'Volume', gdtInteger);

  // one row per key, the values in column order
  oGrid.Data.AppendRow('MSFT', ['MSFT', 412.5, 1200]);
  oGrid.Data.AppendRow('AAPL', ['AAPL', 228.1, 3400]);

  // later, from any thread: update the row of a key by field name
  oGrid.Data.UpdateRow('MSFT', ['Price', 'Volume'], [413.1, 1350]);
end;
```

Voir aussi

- TsgcGridColumns
- TsgcGridData
- TsgcGridRenderer
- Sélection et clavier
- Modification sur place
- Regroupement
- Résumés
- Colonnes figées
- TsgcGridLineView
- Lignes d'arbre
- Maître-détail
- Export Excel

TsgcFMXGrid

La grille de données en direct FireMonkey, dans l'unité sgcGrid_FMX (Delphi XE6 et versions ultérieures, Windows, macOS, iOS et Android). Elle publie les propriétés, méthodes et événements de TsgcVCLGrid, plus les membres FireMonkey habituels tels que Position, Size, Margins, Opacity et les événements de souris et de clavier. La grille dessine ses propres barres de défilement sur les bords droit et inférieur tant que le contenu dépasse la vue : le curseur peut être glissé et un clic sur la piste défile page par page. La molette de la souris défile de trois lignes (Maj : horizontalement) et un geste de panoramique défile avec le doigt. Ctrl+C copie la sélection, Cmd+C sur macOS. BorderStyle est un TsgcGridBorderStyle, gbsNone ou gbsSingle. Depuis un thread d'écriture, la grille met en file d'attente une mise à jour de la vue vers le thread principal avec TThread.Queue.

Propriétés

Propriété	Type	Description
Columns	TsgcGridColumns	Les colonnes de la grille. Voir TsgcGridColumns.
Data	TsgcGridData	Les lignes, le tri, le filtre et le suivi des modifications. Publique. Elle implémente aussi IsgcUITable pour la grille. Voir TsgcGrid-Data.
Renderer	TsgcGridRenderer	Le moteur de rendu avec lequel la grille dessine. Publique, en lecture seule.
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom.
AccentColor	TsgcUIColor	L'accent de la sélection et du clignotement des modifications. 0 utilise l'accent du thème.
FontSize	Double	La taille du texte des cellules.
HeaderHeight, RowHeight	Integer	La hauteur de l'en-tête (32) et de chaque ligne (28), en pixels logiques.
GridLinesH, GridLi-nesV	Boolean	Dessinent les lignes de grille horizontales et verticales. True par défaut.
AlternateRows	Boolean	Teinte une ligne sur deux. True par défaut.
SelectionMode	TsgcGridSelection-Mode	gsmRow (par défaut) sélectionne des lignes entières, gsmCell sélectionne des cellules individuelles.
BorderStyle	TBorderStyle (VCL), TsgcGrid-BorderStyle (FMX)	Un cadre simple dans la couleur Border du thème, ou aucun.
MaxRows	Integer	Mode queue : 0 conserve toutes les lignes, une valeur supérieure à 0 supprime les plus anciennes lors d'une insertion.
TailMode	Boolean	Garde la ligne la plus récente visible tant que la grille est défilée jusqu'en bas.
FilterText	string	Affiche uniquement les lignes dont le texte visible le contient, sans tenir compte de la casse.
ReadOnly	Boolean	True par défaut. False active la modification sur place des colonnes Editable (SGC_GRID_EDIT).

FlashChanges, FlashDuration, FlashMode	Boolean, Integer, TsgcGridFlashMode	Fait clignoter les cellules modifiées, pendant combien de millisecondes (800), sur l'arrière-plan ou sur le texte.
FlashUpColor, FlashDownColor, FlashChangedColor	TsgcUIColor	Les couleurs de clignotement ; 0 utilise les valeurs par défaut du thème.
FormatRules	TsgcGridFormatRules	Les règles de mise en forme conditionnelle. Voir Mise en forme conditionnelle.
MaxFPS	Integer	Le plus grand nombre de repeints par seconde (60).
FrameCount	Cardinal	Le nombre d'images dessinées jusqu'à présent. Publique, en lecture seule.
TopRow, ScrollX	Integer	La première ligne visible (la première ligne d'affichage tant que les lignes sont regroupées) et la position de défilement horizontale. Publique.
SelectedRow, SelectedCol	Integer	La ligne de vue et l'index de colonne sélectionnés, -1 si aucun. Publics.
SelectedKey	string	La clé de la ligne sélectionnée. La sélection suit sa clé lorsque la vue change. Publique, en lecture seule.
FrozenColumns	Integer	Le nombre de colonnes visibles qui ne défilent pas horizontalement, 0 par défaut. Voir Colonnes figées.
ShowFooter	Boolean	Affiche la ligne de pied avec les résumés des colonnes, False par défaut. Voir Résumés.
ShowGroupPanel, GroupPanelHeight	Boolean, Integer	Affiche le panneau de regroupement au-dessus de l'en-tête (False par défaut) et sa hauteur en DIP (36). Voir Regroupement.
GroupIndent	Integer	Le décalage de chaque niveau de groupe en DIP, 20 par défaut.
GroupsExpanded	Boolean	L'état des nouveaux groupes, True par défaut. L'affecter conserve l'état des groupes actuels.
GroupSummaries	Boolean	Affiche les résumés des colonnes dans les lignes de groupe, True par défaut.
Lines	TsgcGridLineView	Les lignes d'affichage : lignes de groupe et lignes de la vue. Publique, en lecture seule. Voir TsgcGridLineView.
FocusedLine	Integer	La ligne d'affichage ayant le focus, -1 si aucune. Sur une ligne de groupe, SelectedRow vaut -1. Publique.
Tree	TsgcGridTreeOptions	Lignes d'arbre : KeyField, ParentField, TreeField, Indent et Expanded. Le mode arbre est actif lorsque KeyField et ParentField nomment des colonnes de la grille. Édition Registered uniquement. Voir Lignes d'arbre.
Detail	TsgcGridDetailOptions	Lignes maître-détail : Active, Source, MasterKeyField, DetailKeyField, Lines et Columns. Édition Registered uniquement. Voir Maître-détail.

Méthodes

Méthode	Retourne	Description
ScrollToRow(aRow)		Fait défiler la vue jusqu'à la ligne de vue indiquée.
RowAt(X, Y), ColAt(X, Y)	Integer	La ligne de vue et l'index de colonne situés sous un point, -1 si aucun.
SaveToPNG(aFileName, aWidth, aHeight)		Écrit la vue courante dans un fichier PNG.
SaveToSVG(aFileName, aWidth, aHeight)		Écrit la vue courante dans un fichier SVG.
StartEdit(aRow, aCol)	Boolean	Ouvre l'éditeur sur une cellule, ou bascule une cellule booléenne (SGC_GRID_EDIT).
CommitEdit	Boolean	Convertit, valide et écrit le texte de l'éditeur ; False garde l'éditeur ouvert (SGC_GRID_EDIT).
CancelEdit		Ferme l'éditeur sans écrire (SGC_GRID_EDIT).
EditorActive, Editor, EditRow, EditCol		L'état de l'éditeur ouvert (SGC_GRID_EDIT). Propriétés.
GroupBy(aFieldNames), ClearGrouping		Regroupe selon les champs indiqués (FieldName, niveau 0 d'abord ; les champs inconnus sont ignorés), ou supprime le regroupement.
ExpandAll, CollapseAll		Déploie ou replie chaque groupe, et les nouveaux groupes à partir de maintenant. En mode arbre, ils agissent sur les nœuds de l'arbre (édition Registered).
ExpandGroup(aGroup), CollapseGroup(aGroup)		Déploie ou replie un seul groupe.
GroupCount	Integer	Le nombre de groupes.
GroupInfo(aGroup)	TsgcGridGroupInfo	Niveau, parent, colonne, membre, nombre de lignes, état et ligne d'un groupe.
FooterSummary(aCol), FooterSummaryText(aCol)	Variant, string	La valeur de pied d'une colonne (index de Columns) et son texte ; Null et une chaîne vide lorsque la colonne n'a pas de SummaryType.
ExpandNode(aRow), CollapseNode(aRow)		Déploie ou replie le nœud d'arbre de la ligne de la vue aRow et déclenche OnTreeExpandChange. Édition Registered uniquement.
ExpandDetail(aRow), CollapseDetail(aRow)		Ouvre ou ferme la bande de détail de la ligne maître de la vue aRow et déclenche OnDetailExpandChange. Édition Registered uniquement.
DetailExpanded(aRow)	Boolean	True lorsque la bande de la ligne de la vue aRow est ouverte. Édition Registered uniquement.

DetailData(aRow)	TsgcGridData	Les lignes de la bande de la ligne de la vue aRow, nil lorsqu'elle est repliée. Lecture seule. Édition Registered uniquement.
SaveToXLSX(aStream; aSheetName = 'Sheet1'), SaveToXLSX(aFileName; aSheetName = 'Sheet1')		Écrit la grille sous forme de classeur Excel avec les largeurs de colonne actuelles et FrozenColumns. Édition Registered uniquement. Voir Export Excel.

Événements

Événement	Type	Description
OnCellClick, OnCellDblClick	TsgcGridCellEvent	Une cellule a été cliquée ou double-cliquée : ligne de vue et index de colonne.
OnSelectionChange	TNotifyEvent	La ligne ou la cellule sélectionnée a changé.
OnSortChange	TNotifyEvent	Un clic sur l'en-tête a changé le tri.
OnColumnResized	TsgcGridColumnEvent	Une colonne a été redimensionnée à la souris.
OnGetCellStyle	TsgcGridCellStyleEvent	Dernier mot sur les couleurs et le style gras de chaque cellule dessinée. Voir Mise en forme conditionnelle.
OnBeforeEdit	TsgcGridBeforeEditEvent	Un éditeur est sur le point de s'ouvrir ; définissez aAllow sur False pour le refuser (SGC_GRID_EDIT).
OnValidate	TsgcGridValidateEvent	La valeur convertie avant son écriture ; modifiez-la ou rejetez-la (SGC_GRID_EDIT).
OnEdited	TsgcGridEditedEvent	Une valeur a été écrite, avec l'ancienne et la nouvelle valeur (SGC_GRID_EDIT).
OnEditError	TsgcGridEditErrorEvent	Le texte n'a pas pu être converti vers le DataType de la colonne ; aMessage peut être modifié (SGC_GRID_EDIT).
OnGroupingChange	TNotifyEvent	Les colonnes regroupées ou leurs niveaux ont changé, depuis le panneau de regroupement ou depuis le code.
OnGroupExpandChange	TsgcFMXGridGroupExpandEvent	Un groupe a été déployé ou replié ; aGroup vaut -1 après ExpandAll et CollapseAll.
OnGetGroupText	TsgcGridGetGroupTextEvent	Modifie le libellé aText de la ligne de groupe aGroup. Déclenché avec le verrou des données détenu.
OnTreeExpandChange	TsgcFMXGridRowExpandEvent	Un nœud d'arbre a été déployé ou replié ; aRow vaut -1 après ExpandAll et CollapseAll en mode arbre. Édition Registered uniquement.
OnDetailExpandChange	TsgcFMXGridRowExpandEvent	Une bande de détail a été ouverte ou fermée ; aRow est la ligne maître de la vue. Édition Registered uniquement.
OnGetDetailData	TsgcGridGetDetailDataEvent	Remplit les lignes d'une bande (aData) pour la clé maître aMasterKey ; affectez True à aHandled pour les conserver. Sender est la grille. Déclenché sans verrou détenu. Édition Registered uniquement.

OnDetailCellClick	TsgcFMXGridDetailCellEvent	Une cellule d'une bande a reçu un clic : aRow est la ligne maître de la vue, aDetailRow et aDetailCol la ligne de la vue et l'index de Columns des données de la bande. Édition Registered uniquement.

Exemple

```
uses
  System.Classes, FMX.Forms, FMX.Types,
  sgcUICore_Theme, sgcGrid_Columns, sgcGrid_Data, sgcGrid_FMX;

procedure CreateGrid(aForm: TForm);
var
  oGrid: TsgcFMXGrid;
begin
  oGrid := TsgcFMXGrid.Create(aForm);
  oGrid.Parent := aForm;
  oGrid.Align := TAlignLayout.Client;
  oGrid.Theme := utDark;

  oGrid.Columns.Add('Symbol', 'Symbol');
  oGrid.Columns.Add('Price', 'Price', gdtFloat).DisplayFormat := '0.00';
  oGrid.Columns.Add('Volume', 'Volume', gdtInteger);

  oGrid.Data.AppendRow('MSFT', ['MSFT', 412.5, 1200]);
  oGrid.Data.AppendRow('AAPL', ['AAPL', 228.1, 3400]);
  oGrid.Data.UpdateRow('MSFT', ['Price'], [413.1]);
end;
```

Voir aussi

- TsgcGridColumns
- TsgcGridData
- TsgcGridRenderer
- Sélection et clavier
- Modification sur place
- Regroupement
- Résumés
- Colonnes figées
- TsgcGridLineView
- Lignes d'arbre
- Maître-détail
- Export Excel

TsgcGridColumn

Une colonne d'une grille, un élément de TsgcGridColumns, dans l'unité sgcGrid_Columns. Les données adressent les valeurs par l'ID de la colonne, de sorte que déplacer ou supprimer une colonne ne décale jamais les valeurs des autres.

Propriétés

Propriété	Type	Description
FieldName	string	Le nom que les données, les flux, les imports et les règles de mise en forme utilisent pour la colonne.
Caption	string	Le texte de l'en-tête.
DataType	TsgcGrid-DataType	gdtText (par défaut), gdtInteger, gdtFloat, gdtCurrency, gdtDateTime ou gdt-Boolean. Il pilote le tri, la mise en forme, l'alignement, la modification et les comparaisons des règles.
Width, MinWidth	Integer	La largeur (100) et la plus petite largeur qu'autorise un redimensionnement à la souris (20), en pixels logiques.
Alignment	TsgcGridA-lignment	gaLeft, gaCenter, gaRight ou gaAuto (par défaut) : nombres à droite, texte à gauche, booléens centrés.
Display-Format	string	Un format FormatFloat pour les nombres ou un format FormatDateTime pour les dates. Une colonne booléenne sans DisplayFormat est dessinée sous forme de coche.
Visible	Boolean	Affiche la colonne. Les colonnes masquées conservent leurs valeurs et leur in-dex.
Editable	Boolean	La colonne peut être modifiée sur place lorsque la grille n'est pas ReadOnly. True par défaut.
SortOrder, SortIndex	TsgcGrid-SortOrder, Integer	gsoNone, gsoAscending ou gsoDescending, et le rang de la colonne dans un tri multicolonne.
PickList	TStrings	Les valeurs de l'éditeur à liste déroulante de la colonne.
Tag	Integer	Libre pour l'application.
GroupIn-dex	Integer	0 ou plus regroupe la grille selon cette colonne à ce niveau (niveaux classés par GroupIndex, puis par Index) ; -1 (par défaut) ne regroupe pas. Voir Regrou-pement.
Summary-Type	TsgcGrid-Summary-Type	gsuNone (par défaut), gsuSum, gsuCount, gsuAverage, gsuMin ou gsuMax : le résumé de la colonne dans le pied et dans les lignes de groupe. Voir Résumés.
Summary-Format	string	Le masque FormatFloat ou FormatDateTime du résumé, texte littéral entre guillemets doubles ; vide utilise DisplayFormat, et 0 pour gsuCount.

Méthodes

Méthode	Retourne	Description
IsNumeric	Boolean	True pour gdtInteger, gdtFloat et gdtCurrency.
EffectiveAlignment	TsgcGridAlignment	L'alignement en lequel gaAuto se résout.

Exemple

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure SetupColumns(Grid: TsgcVCLGrid);
var
  oColumn: TsgcGridColumn;
begin
  Grid.Columns.Clear;
  Grid.Columns.Add('Symbol', 'Symbol');
  oColumn := Grid.Columns.Add('Price', 'Price', gdtFloat);
  oColumn.DisplayFormat := '#,##0.00';
  oColumn.Width := 90;
  Grid.Columns.Add('Change', 'Change %', gdtFloat).DisplayFormat :=
    '+0.00;-0.00;0.00';
  Grid.Columns.Add('Volume', 'Volume', gdtInteger).DisplayFormat := '#,##0';
  Grid.Columns.Add('Time', 'Updated', gdtDateTime).DisplayFormat := 'hh:nn:ss';
  Grid.Columns.Add('Halted', 'Halted', gdtBoolean);
end;
```

Voir aussi

- TsgcGridColumns
- Tri
- Modification sur place
- Regroupement
- Résumés

TsgcGridColumns

La collection d'éléments TsgcGridColumn d'une grille ou d'un TsgcGridData autonome, dans l'unité sgcGrid_Columns.

Propriétés

Propriété	Type	Description
Items[Index]	TsgcGridColumn	Les colonnes, propriété par défaut.
VisibleColumn[Index]	TsgcGridColumn	Les colonnes visibles dans l'ordre.
Version	Cardinal	Incrémenté à chaque modification des colonnes.

Méthodes

Méthode	Retourne	Description
Add	TsgcGridColumn	Ajoute une colonne vide.
Add(aFieldName, aCaption, aDataType)	TsgcGridColumn	Ajoute une colonne avec un nom de champ, un titre et un type de données (gdtText par défaut).
FindByField(aFieldName)	TsgcGridColumn	La colonne d'un nom de champ, nil s'il n'y en a pas.
IndexOfField(aFieldName)	Integer	L'index de la colonne d'un nom de champ, -1 s'il n'y en a pas.
VisibleCount	Integer	Le nombre de colonnes visibles.

Événements

Événement	Type	Description
OnChange	TNotifyEvent	Une propriété d'une colonne a changé.
OnModelChange	TNotifyEvent	Une modification qui affecte les données ou la vue, comme un tri ou une nouvelle colonne.

Exemple

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;
```

```
procedure SortByChange(Grid: TsgcVCLGrid);
begin
  Grid.Data.BeginUpdate;
  try
    // first by Change, highest first, then by Symbol
    Grid.Columns.FindByField('Change').SortIndex := 0;
    Grid.Columns.FindByField('Change').SortOrder := gsoDescending;
    Grid.Columns.FindByField('Symbol').SortIndex := 1;
    Grid.Columns.FindByField('Symbol').SortOrder := gsoAscending;
  finally
    Grid.Data.EndUpdate;
  end;
end;
```

Voir aussi

- TsgcGridColumn
- TsgcGridData

TsgcGridData

Le magasin de lignes de sgcGrid, indépendant du framework et thread-safe, dans l'unité sgcGrid_Data. Chaque grille en possède un dans sa propriété Data, et il peut aussi s'utiliser seul, par exemple avec TsgcGridRenderer. Create(nil) crée et possède ses propres colonnes.
Les lignes sont conservées dans l'ordre d'insertion et retrouvées par clé en O(log n). Les index de ligne de Cells, CellText, KeyOf, RowOfKey et CellChange sont des lignes de vue (triées et filtrées) ; les index de colonne sont des index de Columns, colonnes masquées comprises. Une lecture hors limites renvoie Null, une chaîne vide ou cdNone et ne lève jamais d'exception, car un thread d'écriture peut supprimer une ligne entre un comptage et une lecture.

Propriétés

Propriété	Type	Description
Columns	TsgcGridColumns	Les colonnes.
Cells[aRow, aCol]	Variant	La valeur d'une ligne de vue.
CellText[aRow, aCol]	string	La valeur mise en forme avec le DisplayFormat de la colonne.
StoreCells[aStoreRow, aCol]	Variant	La valeur d'une ligne du magasin, pour OnFilterRow.
FormatSettings	TFormatSettings	Les paramètres utilisés pour formater et analyser les valeurs.
AutoInsert	Boolean	UpdateRow et UpdateCell insèrent une clé manquante. True par défaut.
MaxRows	Integer	0 : illimité, sinon les lignes les plus anciennes sont supprimées à l'insertion.
FilterText	string	Conserve les lignes dont le texte visible le contient, sans tenir compte de la casse.
Virtual, VirtualRowCount	Boolean, Integer	Le mode virtuel et son nombre de lignes. Voir Mode virtuel.
LayoutVersion	Cardinal	Incrémenté lorsque la disposition de regroupement ou d'arbre des lignes de la vue a pu changer : insertion, suppression, Clear, tout changement de vue ou de colonne, un changement des colonnes regroupées, et un changement de valeur dans une colonne avec GroupIndex 0 ou plus ou dans un champ passé à SetLayoutFields. Un changement de valeur dans toute autre colonne le laisse inchangé. La vue de lignes des grilles le lit. Lecture seule.

Méthodes

Méthode	Retourne	Description

AppendRow(aKey, aValues)	Integer	Ajoute une ligne avec les valeurs dans l'ordre des colonnes et retourne sa ligne du magasin. Une clé vide génère une clé unique #<n> ; une clé existante met à jour cette ligne.
UpdateRow(aKey, aFields, aValues)		Définit les champs indiqués d'une ligne. Les champs inconnus sont ignorés.
UpdateCell(aKey, aField, aValue)		Définit un champ d'une ligne.
DeleteRow(aKey)	Boolean	Supprime la ligne d'une clé.
Clear		Supprime toutes les lignes.
BeginUpdate, EndUpdate, IsUpdating		Regroupent les modifications en un seul OnChange.
Lock, Unlock		Conservent le verrou interne pendant plusieurs lectures.
RowCount, ViewCount	Integer	Les lignes stockées (VirtualRowCount en mode virtuel) et les lignes qui passent le filtre.
ViewRow(aRow)	Integer	La ligne du magasin d'une ligne de vue, -1 si hors limites.
KeyOf(aRow), RowOfKey(aKey), StoreKeyOf(aStoreRow)		Convertissent entre lignes de vue, lignes du magasin et clés. RowOfKey retourne -1 lorsque la clé est absente ou filtrée.
CellChange(aRow, aCol)	TsgcGridCellChange	La Direction (cdNone, cdUp, cdDown, cdChanged) et l'Age en millisecondes de la dernière modification d'une cellule.
GetTick	Int64	L'horloge monotone en millisecondes des horodatages de modification.
InvalidateView		Reconstruit la vue, par exemple après un changement de la logique de OnFilterRow.
UseInvariantFormat		Définit FormatSettings avec un séparateur décimal point, un séparateur de milliers virgule, yyyy-mm-dd et hh:nn:ss.
SetLayoutFields(aFields)		Champs (FieldName, sans tenir compte de la casse) dont les changements de valeur font aussi bouger LayoutVersion, comme les colonnes regroupées ; un tableau vide vide la liste.

Événements

Événement	Type	Description
OnChange	TsgcGridChangeEvent	gckData, gckView ou gckStructure, déclenché après la libération du verrou, dans le thread d'écriture.
OnFilterRow	TsgcGridFilterRowEvent	Accepte ou rejette une ligne du magasin. Appelé avec le verrou détenu.
OnGetValue	TsgcGridGetValueEvent	Les valeurs du mode virtuel.

OnGetTick	TsgcGridGetTickEvent	Remplace l'horloge des horodatages de modification, par exemple dans les tests.

TsgcGridData implémente aussi IsgcUITable, voir IsgcGridDataHost et IsgcUITable.

Exemple

```
uses
  SysUtils,
  sgcGrid_Columns, sgcGrid_Data;

    oData.Columns[1].SortOrder := gsoAscending;
    oData.Lock;
    try
      // view rows: sorted and filtered
      for i := 0 to oData.ViewCount - 1 do
        WriteLn(oData.KeyOf(i), ' = ', oData.CellText[i, 1]);
    finally
      oData.Unlock;
    end;
```

Voir aussi

- Mises à jour en direct et threads
- Filtrage
- Tri
- TsgcGridLineView

TsgcGridLineView

La vue de lignes de sgcGrid, indépendante du framework, dans l'unité sgcGrid_Lines. Elle met en correspondance les lignes d'affichage d'une grille avec les lignes de la vue de son TsgcGridData. Chaque grille en possède une dans sa propriété Lines, et elle peut s'utiliser seule avec un TsgcGridData et le TsgcGridRenderer. Elle ne possède pas les données : libérez-la avant les données.
Une vue de lignes est un état de la vue. Les données, leurs lignes de la vue et IsgcUITable ne changent pas. Chaque ligne fait une ligne de haut, de sorte qu'un index de ligne est une position de défilement. Sans colonne re-groupée (et sans arbre ni bande de détail déployée dans l'édition Registered), ou avec des données virtuelles, rien n'est construit et la ligne i est la ligne de la vue i : c'est l'identité, la grille à plat.

Types de ligne

TsgcGridLineKind	Line
glkRow	Une ligne de la vue. Row la retourne.
glkGroup	Une ligne de groupe. Group retourne le groupe, Row retourne -1.
glkDetail	Une ligne d'une bande de détail (édition Registered, voir Maître-détail). Row retourne la ligne maître.

Les lignes regroupées sont les lignes de groupe en ordre de profondeur, chacune suivie de ses groupes enfants, et un groupe du dernier niveau de ses lignes de la vue. Les lignes de groupe ont Level 0 au niveau le plus extérieur, les lignes de données Level = GroupLevels.

Méthodes

Méthode	Returns	Description
Create(aData)		Crée une vue de lignes sur aData.
Update		Met les lignes à jour par rapport aux données. Chaque lecteur ci-dessous l'appelle, de sorte qu'elle ne coûte rien quand rien n'a changé.
Invalidate		Force une reconstruction à la prochaine lecture.
Identity	Boolean	True lorsque la ligne i est la ligne de la vue i.
Count	Integer	Le nombre de lignes d'affichage.
Kind(aLine)	TsgcGrid-LineKind	Le type d'une ligne, glkRow hors limites.
Row(aLine)	Integer	La ligne de la vue d'une ligne de données, la ligne maître d'une ligne de détail, -1 pour une ligne de groupe ou hors limites.
Level(aLine)	Integer	Le niveau de décalage d'une ligne.
Group(aLine)	Integer	Le groupe d'une ligne de groupe, le groupe le plus interne d'une ligne de données, -1 en identité ou hors limites.
Expandable(aLine), Expanded(aLine)	Boolean	Si la ligne peut être déployée ou repliée, et son état.

ToggleLine(aLine)	Boolean	Bascule la ligne ; False lorsqu'elle n'a rien à basculer.
LineOfRow(aRow)	Integer	La ligne d'une ligne de la vue, -1 dans un groupe replié ou hors limites.
GroupLevels, GroupCount	Integer	Le nombre de niveaux regroupés et de groupes.
GroupInfo(aGroup)	TsgcGrid-GroupIn-fo	Un groupe, de `0` à `GroupCount - 1` en ordre de profondeur.
GroupSummary(aGroup, aCol)	Variant	Le résumé d'une colonne (index de Columns) sur les lignes du groupe ; Null lorsque la colonne n'a pas de SummaryType ou que aGroup est hors limites.
GroupBy(aFieldNames), ClearGrouping		Affectent GroupIndex aux colonnes nommées (niveau 0 d'abord) et -1 aux autres, ou -1 à chaque colonne.
SetGroupExpanded(aGroup, aExpanded)		Déploie ou replie un seul groupe.
ExpandAll, CollapseAll		Chaque groupe, et les nouveaux groupes à partir de maintenant.
FlattenAll	TsgcGrid-FlatLines	Chaque ligne de groupe et chaque ligne de la vue dans l'ordre des lignes, quoi qu'il soit replié ; Hidden marque les lignes situées dans un groupe replié. Les lignes de détail ne sont pas listées.

Propriétés

Propriété	Type	Description
Data	TsgcGridData	Les données de la vue de lignes. Lecture seule.
GroupsExpanded	Boolean	L'état des groupes qui n'ont pas été basculés et des nouveaux groupes. L'affecter conserve l'état des groupes actuels.

Les membres d'arbre et de détail (TreeMode, NodeExpanded, SetNodeExpanded, DetailExpandable, DetailExpanded, SetDetailExpanded, ToggleDetail, DetailView, DetailIndex, TreeOptions, DetailOptions, Tree et Detail) n'existent que dans l'édition Registered, voir Lignes d'arbre et Maître-détail.

TsgcGridGroupInfo

Champ	Type	Description
Level	Integer	0 pour le niveau le plus extérieur.
Parent	Integer	Le groupe parent, -1 au niveau 0.
Column	Integer	L'index de Columns de la colonne regroupée.
RowCount	Integer	Les lignes de la vue du groupe, ses groupes enfants compris.
FirstRow	Integer	La ligne de la vue de la première ligne du groupe, dans l'ordre des lignes.

Text	string	Le texte du membre, (blank) pour le membre vide.
Value	Variant	La valeur typée de la première ligne, Null pour le membre vide.
Blank, Expanded	Boolean	Si le groupe est le membre vide, et son état.
Line	Integer	La ligne de la ligne de groupe, -1 dans un groupe replié.
Path	string	Les textes des membres à partir du niveau 0, joints par #1.

TsgcGridFlatLines est un enregistrement avec Count et les tableaux Kind, Ref (la ligne de la vue ou le groupe), Level et Hidden.

Threads

Appelez la vue de lignes depuis le thread qui possède la grille, le thread principal pour un contrôle. Les threads d'écriture continuent d'écrire dans les données ; la vue de lignes se reconstruit à sa prochaine lecture lorsque le LayoutVersion des données a bougé, et n'actualise que les résumés de groupe lorsque seules des valeurs ont changé. TsgcGridGrouper (unité sgcGrid_Group) effectue le regroupement pour elle.

Exemple

Regroupe trois lignes par Region sans contrôle et affiche chaque groupe avec son nombre de lignes et la somme de Amount, puis les produits qu'il contient.

```
uses
  SysUtils, Classes, Variants,
  sgcGrid_Columns, sgcGrid_Data, sgcGrid_Summary, sgcGrid_Lines;

procedure PrintGroups(aOut: TStrings);
var
  oData: TsgcGridData;
  oLines: TsgcGridLineView;
  vAmount: TsgcGridColumn;
  vInfo: TsgcGridGroupInfo;
  i: Integer;
begin
  oData := TsgcGridData.Create(nil);
  try
    oData.Columns.Add('Region', 'Region');
    oData.Columns.Add('Product', 'Product');
    vAmount := oData.Columns.Add('Amount', 'Amount', gdtFloat);
    vAmount.SummaryType := gsuSum;
    oData.AppendRow('1', ['North', 'Apples', 120.0]);
    oData.AppendRow('2', ['South', 'Pears', 80.0]);
    oData.AppendRow('3', ['North', 'Pears', 45.5]);

    oLines := TsgcGridLineView.Create(oData);
    try
      oLines.GroupBy(['Region']);
      for i := 0 to oLines.Count - 1 do
        case oLines.Kind(i) of
          glkGroup:
            begin
              vInfo := oLines.GroupInfo(oLines.Group(i));
              aOut.Add(Format('%s: %d rows, %s', [vInfo.Text, vInfo.RowCount,
                sgcGridSummaryText(vAmount, gsuSum,
                oLines.GroupSummary(oLines.Group(i), vAmount.Index),
                oData.FormatSettings)]));
            end;
          glkRow:
            aOut.Add('  ' + oData.CellText[oLines.Row(i), 1]);
```

```
        end;
    finally
      oLines.Free;
    end;
  finally
    oData.Free;
  end;
end;
```

Voir aussi

- Regroupement
- Résumés
- TsgcGridData
- TsgcGridRenderer
- Regroupement et résumés .NET

TsgcGridDataStore

Un composant non visuel, dans l'unité sgcGrid_Detail, qui conserve des lignes sans les afficher. Son rôle est d'être la Source des bandes de détail d'une grille, voir Maître-détail. Il est rempli par un TsgcGridSource_DataSet dont la Grid est le store, ou par code, de sorte qu'un détail TDataSet fonctionne depuis le concepteur de fiches sans code. Il se trouve sur la page SGC Grid de la palette de composants.
Cette fonctionnalité fait partie de l'édition Registered. Les packages Community ne la contiennent pas.
TsgcGridDataStore dérive de TComponent et implémente IsgcGridDataHost et IsgcUITable, cette dernière déléguée à Data.

Propriétés

Propriété	Type	Description
Columns	TsgcGridColumns	Les colonnes des lignes. Publiée, de sorte qu'elle se définit dans le concepteur. Avec un TsgcGridSource_DataSet et AutoColumns True, elles sont créées à partir des champs.
Data	TsgcGridData	Les lignes, lues et écrites comme le Data d'une grille : AppendRow, UpdateRow, DeleteRow, BeginUpdate et EndUpdate. Publique, en lecture seule.

Utilisation

- Affectez le store à Detail.Source d'une grille, et à DetailKeyField la colonne du store qui contient la clé maître.
- Les modifications des lignes du store, par code ou par la source TDataSet, remplissent de nouveau les bandes de détail ouvertes à l'image suivante.
- Libérer le store affecte nil à Detail.Source de la grille.

Exemple

Un TDataSet de commandes chargé dans un store, affiché comme le détail d'une grille de clients :

```
uses
  SysUtils, Classes, DB,
  sgcGrid_Detail, sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure BindOrders(Grid: TsgcVCLGrid; Orders: TDataSet;
  Store: TsgcGridDataStore; Source: TsgcGridSource_DataSet);
begin
  // the TDataSet rows go into the store, not into a visible grid
  Source.DataSet := Orders;
  Source.Grid := Store;
  Source.KeyField := 'OrderID';
  // AutoColumns (True by default) creates the store columns from the fields
  Source.Active := True;
  // every expanded customer shows its orders
  Grid.Detail.Source := Store;
  Grid.Detail.MasterKeyField := 'CustomerID';
  Grid.Detail.DetailKeyField := 'CustomerID';
  Grid.Detail.Active := True;
end;
```

La même chose depuis le concepteur : affectez DataSet, Grid (le store), KeyField et Active sur le TsgcGridSource_DataSet, puis Detail.Source, Detail.MasterKeyField, Detail.DetailKeyField et Detail.Active sur la grille. La démo TreeAndDetail fonctionne ainsi.

Voir aussi

- Maître-détail
- TsgcGridData
- TsgcGridColumns
- TsgcGridSource_DataSet
- Éditions
- Lignes d'arbre et maître-détail .NET

TsgcGridRenderer

Le moteur de rendu de sgcGrid, indépendant du framework, dans l'unité sgcGrid_Render. Il dessine un TsgcGridData sur n'importe quel canevas sgcUICore, raster ou SVG, de sorte que la grille VCL, la grille FMX et la sortie sans fenêtre ont le même aspect. La géométrie est en pixels logiques ; Render prend la taille du canevas en pixels d'appareil et l'échelle, et aligne chaque bord sur des pixels d'appareil entiers.

TsgcGridRenderOptions

L'apparence publiée d'une grille : HeaderHeight, RowHeight, CellPadding (8), GridLinesH, GridLinesV, AlternateRows, SelectionMode, Theme, AccentColor, FontSize, HeaderFontSize, ShowSortGlyphs, FlashChanges, FlashDuration, FlashUpColor, FlashDownColor, FlashChangedColor, FlashMode et FormatRules, et la disposition de la vue de lignes FrozenColumns, ShowFooter, ShowGroupPanel, GroupPanelHeight, GroupIndent et GroupSummaries. Les contrôles de grille publient les mêmes propriétés. Colors retourne les couleurs résolues du thème.

TsgcGridRenderState

Un enregistrement avec l'état de vue d'une image : ScrollX, ScrollY, SelectedRow, SelectedCol, HotRow, HotCol, FocusRow, FocusCol, Focused, FocusLine, HotLine et ColWidths. sgcGridRenderState retourne un état vide sans rien de sélectionné. FocusLine et HotLine sont des lignes d'affichage, -1 par défaut ; FocusLine -1 donne le focus à la ligne de FocusRow.

Propriétés

Propriété	Type	Description
Options	TsgcGridRenderOptions	L'aspect.
CurrentData, CurrentNow	TsgcGridData, Int64	Les données et le tick de l'image en cours de dessin, valides dans OnGetCellStyle.
Fading	Boolean	True après une image qui a dessiné un clignotement encore en cours.
FooterSummaries	TsgcGridSummaryCache	Les valeurs de pied, actualisées à chaque image avec ShowFooter.

Méthodes

Méthode	Retourne	Description
Render(aData, aState, aCanvas, aWidth, aHeight, aScale, aNow)		Dessine une image.
SaveToPNG, SaveToSVG		Écrivent une image dans un flux ou un fichier, avec une échelle facultative.
HitTest(aData, aState, aX, aY)	TsgcGridHitInfo	La partie (ghpNone, ghpHeader, ghpHeaderResize, ghpCell), la ligne et la colonne situées sous un point.

ColumnWidth, ColumnLeft, ColumnRight, ContentWidth, ContentHeight, VisibleRowRange		Fonctions d'aide de géométrie.
Render(aData, aLines, aState, aCanvas, aWidth, aHeight, aScale, aNow)		Dessine une image de lignes d'affichage : le panneau de regroupement, l'en-tête, les lignes et le pied. aLines nil est la grille à plat. Voir TsgcGridLineView.
SaveToPNG(aData, aLines, ...), SaveToSVG(aData, aLines, ...)		Écrivent une image de lignes d'affichage dans un flux ou un fichier.
HitTest(aData, aLines, aState, aX, aY, aHeight)	TsgcGridHitInfo	Test de collision sur les lignes d'affichage. Passez la hauteur de la zone d'affichage dans aHeight pour atteindre le pied ; avec 0 le pied n'est jamais atteint.
VisibleLineRange, GutterWidth, FrozenWidth, ContentWidth(aColumns, aLines, aState)		Géométrie de la vue de lignes : les lignes visibles sans le panneau de regroupement ni le pied, la largeur des chevrons de groupe, la largeur qui ne défile pas et la largeur totale.
GroupPanelDropIndex, GroupChipBounds, PaintDragChip		L'emplacement de dépôt sur le panneau de regroupement, les limites d'une pastille et le dessin de la pastille glissée, pour les hôtes qui font glisser des en-têtes vers le panneau de regroupement.
GroupText(aLines, aGroup)	string	Le libellé d'une ligne de groupe, OnGetGroupText appliqué.

Sur les lignes, TsgcGridHitPart ajoute ghpGroupPanel, ghpGroupChip, ghpGroupRow, ghpExpandButton (le chevron), ghpFooter et ghpDetail, et TsgcGridHitInfo ajoute Line (la ligne d'affichage, -1 hors des lignes), Group (le groupe de la ligne, la pastille sur ghpGroupChip, l'emplacement de dépôt sur ghpGroupPanel), DetailRow et DetailCol.

Événements

Événement	Type	Description
OnGetCellStyle	TsgcGridCellStyleEvent	Déclenché en dernier pour chaque cellule dessinée, avec le verrou des données détenu ; Sender est le moteur de rendu.
OnGetGroupText	TsgcGridGetGroupTextEvent	Modifie le libellé d'une ligne de groupe. Déclenché avec le verrou des données détenu.

Les procédures sgcGridRender, sgcGridSaveToPNG et sgcGridSaveToSVG font la même chose avec un moteur de rendu temporaire et un TsgcGridRenderOptions.
sgcGridColumnCaption retourne le libellé d'une colonne dans le panneau de regroupement et dans les lignes de groupe : Caption, sinon FieldName.

Exemple

```
uses
  SysUtils,
  sgcUICore_Theme, sgcGrid_Columns, sgcGrid_Data, sgcGrid_Format,
  sgcGrid_Render;

var
  oData: TsgcGridData;
```

```
  oOptions: TsgcGridRenderOptions;
  vState: TsgcGridRenderState;
begin
  // nil: the data creates and owns its own columns
  oData := TsgcGridData.Create(nil);
  oOptions := TsgcGridRenderOptions.Create;
  try
    // '.' decimals and ',' thousands whatever the locale
    oData.UseInvariantFormat;
    oData.Columns.Add('Symbol', 'Symbol');
    oData.Columns.Add('Price', 'Price', gdtFloat).DisplayFormat := '#,##0.00';
    oData.Columns.Add('Change', 'Change %', gdtFloat).DisplayFormat :=
      '+0.00;-0.00;0.00';
    oData.Columns.Add('Volume', 'Volume', gdtInteger).DisplayFormat := '#,##0';
    oData.Columns.Add('Halted', 'Halted', gdtBoolean).Width := 70;
    oData.AppendRow('MSFT', ['MSFT', 412.5, 1.25, 1200500, False]);
    oData.AppendRow('AAPL', ['AAPL', 228.1, -0.84, 3400200, False]);
    oData.AppendRow('GOOG', ['GOOG', 171.9, 0.37, 980100, False]);
    oData.AppendRow('AMZN', ['AMZN', 186.4, -2.10, 2250000, True]);
    oData.AppendRow('NVDA', ['NVDA', 121.7, 3.02, 5120750, False]);
    oData.Columns[1].SortOrder := gsoDescending;

    oOptions.Theme := utDark;
    oOptions.FormatRules.Add('Change', gfoLess, '0', 0,
      sgcUIARGB($FF, $F0, $60, $60));
    oOptions.FormatRules.Add('Volume', gfoGreater, '5000000', 0, 0, True);

    vState := sgcGridRenderState;
    vState.SelectedRow := 1;
    // 470 x 200 logical pixels, written at scale 2 as 940 x 400 pixels
    sgcGridSaveToPNG(oData, oOptions, vState, 'grid.png', 470, 200, 2);
    sgcGridSaveToSVG(oData, oOptions, vState, 'grid.svg', 470, 200);
  finally
    oOptions.Free;
    oData.Free;
  end;
end;
```

Voir aussi

- Rendu sans fenêtre
- Thèmes
- Mise en forme conditionnelle
- Regroupement
- Colonnes figées
- TsgcGridLineView

TsgcGridFormatRule

Une règle de mise en forme conditionnelle, un élément de TsgcGridFormatRules, dans l'unité sgcGrid_Format.

Propriétés

Propriété	Type	Description
FieldName	string	La colonne dont la valeur est testée.
Operator	TsgcGridFormatOperator	gfoEqual, gfoNotEqual, gfoGreater, gfoGreaterEqual, gfoLess, gfoLessEqual, gfoBetween, gfoContains, gfoStartsWith or gfoIsEmpty.
Value, Value2	string	La valeur de comparaison, et la borne supérieure de gfoBetween (incluse).
BackColor, ForeColor	TsgcUIColor	Les couleurs d'une cellule qui correspond ; 0 les laisse inchangées.
Bold	Boolean	Texte en gras pour une cellule qui correspond.
ApplyToRow	Boolean	Applique le style à chaque cellule de la ligne lorsque la valeur de FieldName correspond.
Enabled	Boolean	True par défaut.
StopOnMatch	Boolean	Une correspondance met fin à l'évaluation pour cette cellule.

Méthodes

Méthode	Retourne	Description
Matches(aValue, aDataType)	Boolean	Teste une valeur.

Exemple

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure SetupRules(Grid: TsgcVCLGrid; Handlers: TGridHandlers);
var
  oRule: TsgcGridFormatRule;
begin
  // red text for a negative change
  Grid.FormatRules.Add('Change', gfoLess, '0', 0,
    sgcUIARGB($FF, $D0, $30, $30));
  // the whole row tinted when a symbol is halted
  oRule := Grid.FormatRules.Add('Halted', gfoEqual, 'true',
    sgcUIARGB($40, $F0, $A0, $20));
  oRule.ApplyToRow := True;
  // bold prices between 100 and 200, and no later rule for that cell
```

```
  oRule := Grid.FormatRules.Add;
  oRule.FieldName := 'Price';
  oRule.Operator := gfoBetween;
  oRule.Value := '100';
  oRule.Value2 := '200';
  oRule.Bold := True;
  oRule.StopOnMatch := True;
  // the last word, per painted cell
  Grid.OnGetCellStyle := Handlers.GetCellStyle;
end;
```

Voir aussi

- Mise en forme conditionnelle
- TsgcGridFormatRules

TsgcGridFormatRules

La collection d'éléments TsgcGridFormatRule, publiée sous le nom FormatRules par les grilles et par TsgcGridRenderOptions.

Méthodes

Méthode	Retourne	Description
Add	TsgcGridFormatRule	Ajoute une règle vide.
Add(aFieldName, aOperator, aValue, aBackColor, aForeColor, aBold)	TsgcGridFormatRule	Ajoute une règle en un seul appel ; aForeColor et aBold sont facultatifs.
Prepare, EvaluateRow, ApplyToCell		Utilisées par le moteur de rendu pendant le dessin.

Événements

Événement	Type	Description
OnChange	TNotifyEvent	Une règle a changé.

Voir aussi

- Mise en forme conditionnelle
- TsgcGridFormatRule

IsgcGridDataHost et IsgcUITable

IsgcGridDataHost

Une interface de l'unité sgcGrid_Data avec une seule méthode, GetGridData, implémentée par chaque composant qui possède un TsgcGridData : TsgcVCLGrid et TsgcFMXGrid. Les sources de données et la requête IA retrouvent les données de leur propriété **Grid** avec sgcGridResolveData : d'abord via IsgcGridDataHost, qui fonctionne sur toutes les versions de Delphi, Delphi 7 compris, puis via une propriété publiée Data et, à partir de Delphi 2010, une propriété publique Data. Leur propriété GridData, lorsqu'elle est affectée, l'emporte toujours sur Grid.

IsgcUITable

L'interface de table de sgcUICore (unité sgcUICore_Table), implémentée par TsgcGridData et déléguée par les grilles. Elle permet à un autre produit de lire la grille sans dépendre de sgcGrid, comme le fait le TsgcChartDataSource_Table de sgcCharts. Les lignes sont des lignes de vue et les colonnes sont des index de Columns.

Méthode	Description
TableRowCount, TableColCount	Le nombre de lignes de vue et de colonnes.
TableColName(aCol)	Le FieldName d'une colonne.
TableValue(aRow, aCol), TableText(aRow, aCol)	Cells et CellText.
TableLock, TableUnlock	Lock et Unlock.
TableVersion	Incrémenté à chaque modification.
TableAddListener, TableRemoveListener	Notify : événements déclenchés après OnChange, dans le thread de la modification, avec les données comme Sender.

Le comptage de références est désactivé. Les données appartiennent à leur grille, il faut donc abandonner la référence d'interface avant de libérer la grille.

Exemple

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure ReadAsTable(Grid: TsgcVCLGrid);
var
  oTable: IsgcUITable;
  i: Integer;
begin
  // no reference counting: the grid owns the data
  oTable := Grid.Data;
  oTable.TableLock;
  try
    for i := 0 to oTable.TableRowCount - 1 do
      OutputDebugString(PChar(oTable.TableText(i, 0)));
  finally
```

```
    oTable.TableUnlock;
  end;
end;
```

Voir aussi

- Graphiques à partir de la grille
- TsgcGridData

TsgcGridSource_DataSet

Remplit une grille à partir de n'importe quel TDataSet, dans l'unité sgcGrid_Source_DataSet (unité DB uniquement, sans dépendance VCL ni FMX).

Propriétés

Propriété	Type	Description
Grid	TComponent	La grille à remplir. Voir IsgcGridDataHost.
GridData	TsgcGridData	Des données à remplir à la place d'une grille. Publique ; l'emporte sur Grid.
DataSet	TDataSet	Le dataset source.
KeyField	string	Le champ dont le texte est la clé de ligne. Avec une clé, les lignes sont mises à jour sur place et conservent leur clignotement, les nouvelles clés sont ajoutées et les clés manquantes supprimées. Sans clé, chaque chargement remplace les lignes.
AutoColumns	Boolean	Crée les colonnes à partir des champs lorsque la grille n'en a aucune. True par défaut.
LiveUpdates	Boolean	Suit le dataset : un Post, une insertion, une suppression, une annulation ou un rafraîchissement resynchronise la grille. Les modifications non validées et le défilement sont ignorés. True par défaut.
Active	Boolean	Charge le dataset lorsqu'il est ouvert et continue de le suivre.

Méthodes

Méthode	Retourne	Description
Load		Charge le dataset maintenant.
Refresh		Recharge lorsque Active.
FieldTypeToDataType(aFieldType)	TsgcGridDataType	Le type de colonne d'un type de champ. Fonction de classe.

Exemple

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

function BindDataSet(Grid: TsgcVCLGrid; DataSet: TDataSet)
  : TsgcGridSource_DataSet;
begin
  Result := TsgcGridSource_DataSet.Create(Grid.Owner);
```

```
  Result.Grid := Grid;
  Result.DataSet := DataSet;
  Result.KeyField := 'ID'; // rows are updated in place by ID
  Result.AutoColumns := True; // columns from the fields when the grid has none
  Result.LiveUpdates := True; // follow Post, Delete and Refresh
  Result.Active := True;
end;
```

Voir aussi

- Liaison à un DataSet
- JSON et CSV

TsgcGridFeedSource

La classe de base des sources de flux sgcGridWS, dans l'unité sgcGridWS_Source. Elle lit chaque message comme du JSON et écrit ses lignes dans une grille par clé : seuls les chemins présents dans une ligne sont écrits, un message peut donc contenir une mise à jour partielle. Les nombres JSON sont écrits en Double, les booléens en Boolean, les chaînes en string et null en Null.
Cette fonctionnalité fait partie du pont sgcGridWS : les unités nécessitent les sources ou les packages de sgcWebSockets, et l'édition Community ne l'inclut pas.

Propriétés

Propriété	Type	Description
Grid	TComponent	La grille dans laquelle écrire.
GridData	TsgcGridData	Des données dans lesquelles écrire à la place d'une grille. Publique ; l'emporte sur Grid.
RowsPath	string	Le chemin des lignes dans un message : un tableau d'objets ligne ou un seul objet ligne. Vide : le message lui-même est une ligne ou un tableau de lignes.
KeyPath	string	Le chemin de la clé de ligne dans un objet ligne, par exemple symbol ou id.value. Une ligne sans clé est ignorée et signalée par OnError.
Fields	TStrings	Un FieldName=chemin.json par ligne, tel que Price=quote.last ou Bid=book.bid[0]. Une ligne sans = lit le chemin de même nom. Vide : chaque colonne est lue depuis le chemin nommé comme son FieldName.
DeletePath	string	Un chemin booléen dans l'objet ligne : true, un nombre non nul ou la chaîne true supprime la ligne de la clé.
MaxRowsPerMessage	Integer	Le nombre maximal de lignes appliquées depuis un message (10000) ; 0 sans limite. Les lignes en trop sont ignorées et signalées.
Active	Boolean	False ignore tous les messages. True par défaut.

Événements

Événement	Type	Description
OnRow	TsgcGridFeedRowEvent	Accepte ou ignore la ligne d'une clé.
OnError	TsgcGridFeedErrorEvent	Un message ou une ligne n'a pas pu être appliqué.

Threads

Toutes les lignes d'un message sont appliquées dans un seul Data.BeginUpdate et EndUpdate, dans le thread où arrive le message. OnRow et OnError s'exécutent aussi dans ce thread, il faut donc synchroniser avant de toucher un contrôle à cet endroit.

Exemple

```
uses
  Windows, SysUtils, Classes, Forms,
  sgcWebSocket, sgcWebSocket_Protocols, sgcHTTP, sgcHTML_AI,
  sgcGrid_Columns, sgcGrid_VCL, sgcGridWS_Source, sgcGridWS_Source_WebSocket,
  sgcGridWS_Source_MQTT, sgcGridWS_Source_SSE, sgcGridWS_AI;

procedure TFeedHandlers.FeedError(Sender: TObject; const aError: string);
begin
  // runs in the thread the message arrived in
  OutputDebugString(PChar('feed: ' + aError));
end;

procedure TFeedHandlers.FeedRow(Sender: TObject; const aKey: string;
  var aAccept: Boolean);
begin
  // skip the rows of test symbols
  aAccept := Pos('TEST', aKey) <> 1;
end;
```

Voir aussi

- TsgcGridFeedSource_WebSocket
- TsgcGridFeedSource_MQTT
- TsgcGridFeedSource_SSE
- Flux en direct

TsgcGridFeedSource_WebSocket

Un TsgcGridFeedSource qui lit les messages d'un TsgcWebSocketClient de sgcWebSockets, dans l'unité sgcGridWS_Source_WebSocket.
Tant que la source est Active et que Client est affecté, à l'exécution, elle installe son propre gestionnaire OnMessage sur le client et conserve le gestionnaire qui s'y trouvait : chaque message texte est d'abord appliqué à la grille puis transmis au gestionnaire précédent, de sorte que le gestionnaire de l'application se déclenche toujours. Définir Active sur False, effacer Client ou détruire la source restaure le gestionnaire précédent.
Cette fonctionnalité fait partie du pont sgcGridWS : les unités nécessitent les sources ou les packages de sgcWebSockets, et l'édition Community ne l'inclut pas.

Propriétés

Propriété	Type	Description
Client	TsgcWebSocketClient	Le client WebSocket.
Grid	TComponent	La grille dans laquelle écrire.
GridData	TsgcGridData	Des données dans lesquelles écrire à la place d'une grille. Publique ; l'emporte sur Grid.
RowsPath	string	Le chemin des lignes dans un message : un tableau d'objets ligne ou un seul objet ligne. Vide : le message lui-même est une ligne ou un tableau de lignes.
KeyPath	string	Le chemin de la clé de ligne dans un objet ligne, par exemple symbol ou id.value. Une ligne sans clé est ignorée et signalée par OnError.
Fields	TStrings	Un FieldName=chemin.json par ligne, tel que Price=quote.last ou Bid=book.bid[0]. Une ligne sans = lit le chemin de même nom. Vide : chaque colonne est lue depuis le chemin nommé comme son FieldName.
DeletePath	string	Un chemin booléen dans l'objet ligne : true, un nombre non nul ou la chaîne true supprime la ligne de la clé.
MaxRowsPerMessage	Integer	Le nombre maximal de lignes appliquées depuis un message (10000) ; 0 sans limite. Les lignes en trop sont ignorées et signalées.
Active	Boolean	False ignore tous les messages. True par défaut.

Méthodes

Méthode	Retourne	Description
ProcessMessage(aText)	Integer	Applique manuellement un message et retourne le nombre de lignes appliquées.

Événements

Événement	Type	Description

OnRow	TsgcGridFeedRowEvent	Accepte ou ignore la ligne d'une clé.
OnError	TsgcGridFeedErrorEvent	Un message ou une ligne n'a pas pu être appliqué.

Exemple

```
uses
  Windows, SysUtils, Classes, Forms,
  sgcWebSocket, sgcWebSocket_Protocols, sgcHTTP, sgcHTML_AI,
  sgcGrid_Columns, sgcGrid_VCL, sgcGridWS_Source, sgcGridWS_Source_WebSocket,
  sgcGridWS_Source_MQTT, sgcGridWS_Source_SSE, sgcGridWS_AI;

function FeedFromWebSocket(Grid: TsgcVCLGrid; Client: TsgcWebSocketClient)
  : TsgcGridFeedSource_WebSocket;
begin
  // messages such as {"quotes":[{"symbol":"MSFT","quote":{"last":412.5}}]}
  Result := TsgcGridFeedSource_WebSocket.Create(Grid.Owner);
  Result.Grid := Grid;
  Result.RowsPath := 'quotes';
  Result.KeyPath := 'symbol';
  Result.Fields.Add('Symbol=symbol');
  Result.Fields.Add('Price=quote.last');
  Result.Fields.Add('Volume'); // same-named path
  Result.DeletePath := 'removed';
  Result.Client := Client;
  Client.URL := 'wss://quotes.example.com/stream';
  Client.Active := True;
end;
```

Voir aussi

- TsgcGridFeedSource
- Flux en direct

TsgcGridFeedSource_MQTT

Un TsgcGridFeedSource qui lit les messages d'un TsgcWSPClient_MQTT de sgcWebSockets, dans l'unité sgcGridWS_Source_MQTT.
Tant que la source est Active et que Client est affecté, à l'exécution, elle installe son propre gestionnaire OnMQTTPublish sur le client et conserve le gestionnaire qui s'y trouvait : chaque publication dont le topic correspond à Topic est d'abord appliquée à la grille puis transmise au gestionnaire précédent, de sorte que le gestionnaire de l'application se déclenche toujours. Définir Active sur False, effacer Client ou détruire la source restaure le gestionnaire précédent. La source ne s'abonne pas : c'est l'application qui s'abonne sur le client.
Cette fonctionnalité fait partie du pont sgcGridWS : les unités nécessitent les sources ou les packages de sgcWebSockets, et l'édition Community ne l'inclut pas.

Propriétés

Propriété	Type	Description
Client	TsgcWSPClient_MQTT	Le client MQTT.
Topic	string	Un filtre de topic MQTT : + correspond à un niveau, # à ce niveau et à tous les niveaux inférieurs. Vide : accepte toutes les publications.
Grid	TComponent	La grille dans laquelle écrire.
GridData	TsgcGridData	Des données dans lesquelles écrire à la place d'une grille. Publique ; l'emporte sur Grid.
RowsPath	string	Le chemin des lignes dans un message : un tableau d'objets ligne ou un seul objet ligne. Vide : le message lui-même est une ligne ou un tableau de lignes.
KeyPath	string	Le chemin de la clé de ligne dans un objet ligne, par exemple symbol ou id.value. Une ligne sans clé est ignorée et signalée par OnError.
Fields	TStrings	Un FieldName=chemin.json par ligne, tel que Price=quote.last ou Bid=book.bid[0]. Une ligne sans = lit le chemin de même nom. Vide : chaque colonne est lue depuis le chemin nommé comme son FieldName.
DeletePath	string	Un chemin booléen dans l'objet ligne : true, un nombre non nul ou la chaîne true supprime la ligne de la clé.
MaxRowsPerMessage	Integer	Le nombre maximal de lignes appliquées depuis un message (10000) ; 0 sans limite. Les lignes en trop sont ignorées et signalées.
Active	Boolean	False ignore tous les messages. True par défaut.

Méthodes

Méthode	Retourne	Description
ProcessPublish(aTopic, aText)	Integer	Applique une publication manuellement.

Événements

Événement	Type	Description
OnRow	TsgcGridFeedRowEvent	Accepte ou ignore la ligne d'une clé.
OnError	TsgcGridFeedErrorEvent	Un message ou une ligne n'a pas pu être appliqué.

Exemple

```
uses
  Windows, SysUtils, Classes, Forms,
  sgcWebSocket, sgcWebSocket_Protocols, sgcHTTP, sgcHTML_AI,
  sgcGrid_Columns, sgcGrid_VCL, sgcGridWS_Source, sgcGridWS_Source_WebSocket,
  sgcGridWS_Source_MQTT, sgcGridWS_Source_SSE, sgcGridWS_AI;

function FeedFromMQTT(Grid: TsgcVCLGrid; MQTT: TsgcWSPClient_MQTT)
  : TsgcGridFeedSource_MQTT;
begin
  Result := TsgcGridFeedSource_MQTT.Create(Grid.Owner);
  Result.Grid := Grid;
  Result.KeyPath := 'id';
  Result.Topic := 'plant/+/sensors';
  Result.Client := MQTT;
  // the source does not subscribe: subscribe on the MQTT client, for
  // example in its OnMQTTConnect event
end;
```

Voir aussi

- TsgcGridFeedSource
- Flux en direct

TsgcGridFeedSource_SSE

Un TsgcGridFeedSource qui lit les messages des événements envoyés par le serveur (server-sent events) d'un TsgcHTTP1Client de sgcWebSockets, dans l'unité sgcGridWS_Source_SSE.
Tant que la source est Active et que Client est affecté, à l'exécution, elle installe son propre gestionnaire OnSSE-Message sur le client et conserve le gestionnaire qui s'y trouvait : chaque fragment du flux est d'abord appliqué à la grille puis transmis au gestionnaire précédent, de sorte que le gestionnaire de l'application se déclenche toujours. Définir Active sur False, effacer Client ou détruire la source restaure le gestionnaire précédent. Les fragments sont découpés en événements : les lignes data sont jointes, une ligne vide termine l'événement, et un événement peut s'étendre sur plusieurs fragments. L'application démarre le flux avec Client.GetSSE.
Cette fonctionnalité fait partie du pont sgcGridWS : les unités nécessitent les sources ou les packages de sgcWebSockets, et l'édition Community ne l'inclut pas.

Propriétés

Propriété	Type	Description
Client	TsgcHTTP1Client	Le client HTTP.
Grid	TComponent	La grille dans laquelle écrire.
GridData	TsgcGridData	Des données dans lesquelles écrire à la place d'une grille. Publique ; l'emporte sur Grid.
RowsPath	string	Le chemin des lignes dans un message : un tableau d'objets ligne ou un seul objet ligne. Vide : le message lui-même est une ligne ou un tableau de lignes.
KeyPath	string	Le chemin de la clé de ligne dans un objet ligne, par exemple symbol ou id.value. Une ligne sans clé est ignorée et signalée par OnError.
Fields	TStrings	Un FieldName=chemin.json par ligne, tel que Price=quote.last ou Bid=book.bid[0]. Une ligne sans = lit le chemin de même nom. Vide : chaque colonne est lue depuis le chemin nommé comme son FieldName.
DeletePath	string	Un chemin booléen dans l'objet ligne : true, un nombre non nul ou la chaîne true supprime la ligne de la clé.
MaxRowsPerMessage	Integer	Le nombre maximal de lignes appliquées depuis un message (10000) ; 0 sans limite. Les lignes en trop sont ignorées et signalées.
Active	Boolean	False ignore tous les messages. True par défaut.

Méthodes

Méthode	Retourne	Description
ProcessChunk(aText)	Integer	Applique manuellement un fragment du flux.
ResetStream		Abandonne un événement partiellement reçu.

Événements

Événement	Type	Description
OnRow	TsgcGridFeedRowEvent	Accepte ou ignore la ligne d'une clé.
OnError	TsgcGridFeedErrorEvent	Un message ou une ligne n'a pas pu être appliqué.

Exemple

```
uses
  Windows, SysUtils, Classes, Forms,
  sgcWebSocket, sgcWebSocket_Protocols, sgcHTTP, sgcHTML_AI,
  sgcGrid_Columns, sgcGrid_VCL, sgcGridWS_Source, sgcGridWS_Source_WebSocket,
  sgcGridWS_Source_MQTT, sgcGridWS_Source_SSE, sgcGridWS_AI;

function FeedFromSSE(Grid: TsgcVCLGrid; HTTP: TsgcHTTP1Client)
  : TsgcGridFeedSource_SSE;
begin
  Result := TsgcGridFeedSource_SSE.Create(Grid.Owner);
  Result.Grid := Grid;
  Result.KeyPath := 'orderId';
  Result.Client := HTTP;
  // the application starts the stream
  HTTP.GetSSE('https://api.example.com/orders/stream');
end;
```

Voir aussi

- TsgcGridFeedSource
- Flux en direct

TsgcGridAIQuery

Transforme une question en langage courant en filtres, tris et mises en évidence d'une grille, dans l'unité sgcGridWS_AI. La question est résolue par le TsgcHTMLAIAssistant de sgcWebSockets : un LLM lorsque l'Assistant a un fournisseur, sinon son analyseur à base de règles, qui comprend des questions telles que price greater than 100, sort by volume descending et highlight change less than 0. La réponse est vérifiée par rapport aux colonnes de la grille avant que quoi que ce soit ne soit appliqué.
Cette fonctionnalité fait partie du pont sgcGridWS : les unités nécessitent les sources ou les packages de sgcWebSockets, et l'édition Community ne l'inclut pas.

Propriétés

Propriété	Type	Description
Grid	TComponent	La grille à interroger. Voir IsgcGridDataHost.
GridData	TsgcGridData	Des données à interroger à la place d'une grille. Publique ; l'emporte sur Grid.
Assistant	TsgcHTMLAIAssistant	L'assistant qui répond. Nil utilise l'analyseur à base de règles, sans réseau ni clé d'API.
Active	Boolean	True par défaut.

Méthodes

Méthode	Retourne	Description
Ask(aQuestion)	Boolean	Supprime ce que le Ask précédent a appliqué, puis applique la réponse. Synchrone ; appelez-le depuis le thread principal.
Reset		Restaure le OnFilterRow et le tri précédents, et supprime uniquement les règles de mise en forme ajoutées par ce composant.
LastExplanation	string	Ce qui a été appliqué, et ce qui a été ignoré.

Événements

Événement	Type	Description
OnApplied	TNotifyEvent	Une réponse a été appliquée.
OnError	TsgcGridAIErrorEvent	La question n'a pas pu être résolue ou appliquée.

Exemple

```
uses
  Windows, SysUtils, Classes, Forms,
  sgcWebSocket, sgcWebSocket_Protocols, sgcHTTP, sgcHTML_AI,
```

```
  sgcGrid_Columns, sgcGrid_VCL, sgcGridWS_Source, sgcGridWS_Source_WebSocket,
  sgcGridWS_Source_MQTT, sgcGridWS_Source_SSE, sgcGridWS_AI;

function AskGrid(Grid: TsgcVCLGrid): string;
var
  oQuery: TsgcGridAIQuery;
begin
  oQuery := TsgcGridAIQuery.Create(Grid.Owner);
  oQuery.Grid := Grid;
  // no Assistant: the rule-based parser answers, no network, no API key
  if oQuery.Ask('price greater than 100 sort by volume descending ' +
    'highlight change less than 0') then
    Result := oQuery.LastExplanation
  else
    Result := 'not understood';
  // later: remove the filter, the sort and the highlight
  oQuery.Reset;
end;
```

Voir aussi

- Interroger la grille
- Filtrage
- Mise en forme conditionnelle

Mises à jour en direct et threads

sgcGrid est conçue pour des lignes qui changent en permanence. Chaque ligne a une clé, et un thread d'écriture met à jour la ligne d'une clé par nom de champ, depuis n'importe quel thread, sans Synchronize.
Les exemples sont de simples procédures qui prennent un **TsgcVCLGrid**. Les gestionnaires d'événements sont des méthodes d'une petite classe, TGridHandlers, déclarée dans TsgcVCLGrid. Le même code fonctionne avec TsgcFMXGrid.

Mise à jour par clé

UpdateRow définit les champs indiqués de la ligne d'une clé et insère la ligne lorsque la clé est nouvelle (sauf si AutoInsert vaut False). UpdateCell définit un seul champ, AppendRow ajoute une ligne avec les valeurs dans l'ordre des colonnes, et DeleteRow supprime la ligne d'une clé. Chaque cellule modifiée enregistre l'heure et la direction de sa modification, hausse, baisse ou changement, ce qui pilote le clignotement à la modification.

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

type
  TFeedThread = class(TThread)
  private
    FData: TsgcGridData;
  protected
    procedure Execute; override;
  public
    constructor Create(aData: TsgcGridData);
  end;

constructor TFeedThread.Create(aData: TsgcGridData);
begin
  FData := aData;
  inherited Create(False);
end;

procedure TFeedThread.Execute;
const
  CS_SYMBOLS: array [0 .. 2] of string = ('MSFT', 'AAPL', 'GOOG');
begin
  while not Terminated do
  begin
    // thread-safe: no Synchronize, the grid repaints at most MaxFPS
    // times per second in the main thread
    FData.UpdateRow(CS_SYMBOLS[Random(3)], ['Price', 'Time'],
      [100 + Random * 50, Now]);
    Sleep(1);
  end;
end;
```

Repeints

Les données déclenchent OnChange dans le thread d'écriture. La grille s'y contente de se marquer comme à redessiner et se repeint dans le thread principal au plus MaxFPS fois par seconde, de sorte que 1 000 mises à jour par seconde coûtent au plus 60 images. La ligne sélectionnée est retrouvée par sa clé après chaque changement de la vue.

BeginUpdate et EndUpdate

Encadrez un lot de modifications par BeginUpdate et EndUpdate pour n'obtenir qu'une seule notification pour l'ensemble.

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure LoadSnapshot(Grid: TsgcVCLGrid);
var
  i: Integer;
begin
  Grid.Data.BeginUpdate;
  try
    Grid.Data.Clear;
    for i := 1 to 1000 do
      Grid.Data.AppendRow('S' + IntToStr(i), ['S' + IntToStr(i), 100.0, 0.0,
        0, Now, False]);
  finally
    // one change notification and one repaint for the whole batch
    Grid.Data.EndUpdate;
  end;
end;
```

Mode queue

Pour les journaux et les flux qui grandissent, **MaxRows** supérieur à 0 supprime les lignes les plus anciennes lors de l'insertion de nouvelles, et **TailMode** garde la ligne la plus récente visible tant que la grille est défilée jusqu'en bas. Remonter arrête le suivi, redescendre jusqu'en bas le reprend.

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure SetupLog(Grid: TsgcVCLGrid);
begin
  Grid.Columns.Clear;
  Grid.Columns.Add('Level', 'Level').Width := 60;
  Grid.Columns.Add('Text', 'Message').Width := 400;
  Grid.Columns.Add('Time', 'Time', gdtDateTime).DisplayFormat := 'hh:nn:ss.zzz';
  Grid.MaxRows := 5000; // keep the newest 5,000 rows
  Grid.TailMode := True; // follow the newest row while at the bottom
  // an empty key generates a unique key '#<n>'
  Grid.Data.AppendRow('', ['info', 'Service started', Now]);
end;
```

Voir aussi

- TsgcGridData
- Performances
- Flux en direct

Tri

Chaque colonne a un SortOrder (gsoNone, gsoAscending ou gsoDescending) et un SortIndex, son rang dans un tri multicolonne. Le tri est stable. Le texte se compare sans tenir compte de la casse et les autres types numériquement ; les valeurs nulles, vides ou non convertibles sont toujours triées en premier.

À la souris

Un clic sur un en-tête fait passer la colonne par croissant, décroissant puis non trié. Avec Ctrl enfoncé, la colonne est ajoutée au tri courant au lieu de le remplacer, et l'en-tête affiche le rang de chaque colonne triée. OnSortChange se déclenche après le changement.

Depuis le code

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure SortByChange(Grid: TsgcVCLGrid);
begin
  Grid.Data.BeginUpdate;
  try
    // first by Change, highest first, then by Symbol
    Grid.Columns.FindByField('Change').SortIndex := 0;
    Grid.Columns.FindByField('Change').SortOrder := gsoDescending;
    Grid.Columns.FindByField('Symbol').SortIndex := 1;
    Grid.Columns.FindByField('Symbol').SortOrder := gsoAscending;
  finally
    Grid.Data.EndUpdate;
  end;
end;
```

Règle de mise à jour en direct

Un changement de valeur ne retrie la vue que si sa colonne est une colonne de tri ou si un filtre est actif. Une colonne qui varie sans être triée conserve l'ordre courant et ne coûte aucun tri. Les insertions, les suppressions et les changements de tri ou de filtre reconstruisent toujours la vue.

Voir aussi

- Filtrage
- TsgcGridColumn

Filtrage

Deux filtres s'appliquent, l'un après l'autre. **FilterText** conserve les lignes dont le texte formaté, dans n'importe quelle colonne visible, contient le texte, sans tenir compte de la casse. **OnFilterRow** des données est ensuite interrogé pour chaque ligne qui a passé le premier filtre.
Les exemples sont de simples procédures qui prennent un **TsgcVCLGrid**. Les gestionnaires d'événements sont des méthodes d'une petite classe, TGridHandlers, déclarée dans TsgcVCLGrid. Le même code fonctionne avec TsgcFMXGrid.

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure FilterGrid(Grid: TsgcVCLGrid; Handlers: TGridHandlers);
begin
  // rows whose visible text contains 'ms', case-insensitive
  Grid.FilterText := 'ms';
  // and a predicate on top of it
  Grid.Data.OnFilterRow := Handlers.FilterRow;
  // after the logic of the handler changed, rebuild the view
  Grid.Data.InvalidateView;
end;
```

OnFilterRow

Le gestionnaire reçoit une ligne du magasin, pas une ligne de vue : lisez ses valeurs avec StoreCells ou sa clé avec StoreKeyOf. Il est appelé avec le verrou des données détenu, il ne doit donc pas attendre un autre thread qui utilise la grille. Lorsque la logique du gestionnaire change, appelez InvalidateView.

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure TGridHandlers.FilterRow(Sender: TObject; aRow: Integer;
  var aAccept: Boolean);
var
  vPrice: Variant;
begin
  // Sender is the TsgcGridData and aRow a store row: read it with
  // StoreCells, the lock is already held
  vPrice := TsgcGridData(Sender).StoreCells[aRow, 1];
  aAccept := not VarIsNull(vPrice) and (vPrice > 100);
end;
```

Avec un filtre actif, le changement d'une valeur peut faire entrer une ligne dans la vue ou l'en faire sortir. RowOfKey retourne -1 pour une ligne filtrée.

Voir aussi

- Tri
- Interroger la grille
- TsgcGridData

Regroupement

Le regroupement range les lignes d'une grille sous des lignes de groupe, un niveau par colonne regroupée. Une ligne de groupe affiche le membre de son niveau, le nombre de lignes qu'elle contient et, avec les résumés, leurs totaux. Les lignes de groupe se déploient et se replient, et les niveaux imbriqués sont décalés de GroupIndent DIP (20 par défaut). Le regroupement fait partie de toutes les éditions, Community comprise, et fonctionne de la même façon dans TsgcVCLGrid et TsgcFMXGrid.
Le regroupement est un état de la vue. La grille garde ses lignes de groupe dans un TsgcGridLineView (Lines), tandis que TsgcGridData, ses lignes de la vue et IsgcUITable restent inchangés, de sorte qu'un graphique ou tout autre lecteur lié à la grille voit toujours les lignes à plat, triées et filtrées. Sans colonne regroupée, la grille se comporte exactement comme une grille non regroupée.

Regrouper une colonne

Chaque colonne a un GroupIndex, -1 par défaut. Une valeur de 0 ou plus regroupe la grille selon cette colonne à ce niveau : les niveaux sont classés par GroupIndex, puis par l'Index de la colonne. Une colonne masquée regroupe aussi. GroupBy affecte GroupIndex aux colonnes nommées (FieldName, sans tenir compte de la casse, niveau 0 d'abord) et -1 à toutes les autres ; les champs inconnus sont ignorés. ClearGrouping affecte GroupIndex -1 à chaque colonne. Modifier GroupIndex ne reconstruit pas la vue des données.

- Les membres d'un niveau sont classés selon la valeur typée de leur première ligne, avec la règle du tri de la grille : le texte sans tenir compte de la casse, les autres types numériquement, par ordre croissant, ou décroissant lorsque le SortOrder de la colonne est gsoDescending.
- Null, le texte vide et les valeurs qui ne se convertissent pas forment le membre vide, affiché comme (blank) et toujours en premier.
- Lorsque la colonne a un DisplayFormat, le membre est le texte formaté de la cellule, de sorte qu'un DisplayFormat de yyyy-mm regroupe les dates par mois.
- Dans un groupe, les lignes gardent l'ordre du tri de la grille et le filtre s'applique toujours.

Le panneau de regroupement

ShowGroupPanel affiche au-dessus de l'en-tête un panneau de GroupPanelHeight DIP (36 par défaut), avec une pastille par niveau regroupé, ou une invite quand rien n'est regroupé. Avec la souris :

- Faites glisser l'en-tête d'une colonne de plus de 4 pixels sur le panneau pour regrouper selon cette colonne à l'emplacement du dépôt. Les en-têtes ne se déplacent que tant que le panneau est affiché, et Échap annule le glissement.
- Faites glisser une pastille dans le panneau pour déplacer son niveau, ou hors du panneau pour dissocier sa colonne.
- Cliquez sur une pastille pour inverser l'ordre de ses groupes. Cela inverse le SortOrder de la colonne et déclenche OnSortChange.

Dans TsgcFMXGrid, un balayage tactile continue de faire défiler la grille et annule le glissement d'un en-tête, de sorte que le regroupement tactile passe par le code ou la souris.

Lignes de groupe

Une ligne de groupe affiche un chevron, le libellé et les résumés de groupe des colonnes ayant un SummaryType (GroupSummaries, True par défaut). Le libellé est le titre de la colonne, le membre et le nombre de lignes, comme dans Region: North (12). OnGetGroupText peut le remplacer ; il est déclenché avec le verrou des données détenu, donc lisez la grille à cet endroit mais n'écrivez jamais dans les données. GroupInfo(aGroup) retourne le niveau, le groupe parent, la colonne regroupée, le texte et la valeur du membre, le nombre de lignes, l'état déployé et la ligne d'affichage d'un groupe, voir TsgcGridLineView.

Déployer et replier

GroupsExpanded (True par défaut) est l'état des nouveaux groupes ; l'affecter conserve l'état des groupes actuels. ExpandAll et CollapseAll modifient chaque groupe ainsi que les groupes créés ensuite. ExpandGroup et CollapseGroup modifient un seul groupe. OnGroupExpandChange indique le groupe et son nouvel état, avec aGroup -1

après ExpandAll et CollapseAll. L'état déployé d'un groupe est conservé lors des mises à jour en direct et des nouveaux tris.
Un clic sur le chevron ou un double clic sur une ligne de groupe la bascule. Touches sur une ligne de groupe :

Touche	Action
Right	Déploie le groupe, ou passe à sa première ligne enfant lorsqu'il est déployé.
Left	Replie le groupe, ou passe à son groupe parent lorsqu'il est replié.
+ et -	Déploient et replient le groupe.
*	Déploie le groupe et chaque groupe situé en dessous.
Enter	Bascule le groupe.
Ctrl+C	Copie le libellé de la ligne de groupe.

Lorsqu'un repli masque la ligne sélectionnée, sa ligne de groupe visible reçoit le focus.

Lignes et sélection

Tant que les lignes sont regroupées, la grille fait défiler des lignes d'affichage : TopRow et la barre de défilement verticale comptent des lignes, et FocusedLine est la ligne ayant le focus, -1 si aucune. Les index de ligne restent des lignes de la vue partout ailleurs (SelectedRow, Cells, OnCellClick). Sur une ligne de groupe, SelectedRow vaut -1. La ligne sélectionnée est conservée par sa clé et retrouvée après un nouveau regroupement. GroupCount retourne le nombre de groupes et Lines donne la vue de lignes complète.
Un changement de valeur en direct dans une colonne regroupée déplace la ligne vers son nouveau groupe, et OnGroupingChange est déclenché chaque fois que les colonnes regroupées ou leurs niveaux changent, depuis le panneau ou depuis le code.

Exemple

Les gestionnaires se trouvent dans une petite classe, comme ceux des autres exemples de fonctionnalités.

```
uses
  SysUtils, Classes, Forms, Controls,
  sgcGrid_Columns, sgcGrid_Data, sgcGrid_Lines, sgcGrid_VCL;

type
  TGroupHandlers = class
  public
    procedure GetGroupText(Sender: TObject; aGroup: Integer;
      var aText: string);
    procedure GroupExpandChange(Sender: TObject; aGroup: Integer;
      aExpanded: Boolean);
    procedure GroupingChange(Sender: TObject);
  end;

procedure TGroupHandlers.GetGroupText(Sender: TObject; aGroup: Integer;
  var aText: string);
var
  vInfo: TsgcGridGroupInfo;
begin
  // raised with the data lock held: read, never write the data here
  vInfo := TsgcVCLGrid(Sender).GroupInfo(aGroup);
  aText := Format('%s (%d orders)', [vInfo.Text, vInfo.RowCount]);
end;

procedure TGroupHandlers.GroupExpandChange(Sender: TObject; aGroup: Integer;
  aExpanded: Boolean);
begin
```

```
  // aGroup is -1 after ExpandAll and CollapseAll
end;

procedure TGroupHandlers.GroupingChange(Sender: TObject);
begin
  // the grouped columns or their order changed
end;

procedure GroupOrders(Grid: TsgcVCLGrid; Handlers: TGroupHandlers);
var
  i: Integer;
begin
  Grid.OnGetGroupText := Handlers.GetGroupText;
  Grid.OnGroupExpandChange := Handlers.GroupExpandChange;
  Grid.OnGroupingChange := Handlers.GroupingChange;
  Grid.ShowGroupPanel := True;
  // new groups start collapsed
  Grid.GroupsExpanded := False;
  // level 0 by Region, level 1 by Product
  Grid.GroupBy(['Region', 'Product']);
  // open the outer level only
  for i := 0 to Grid.GroupCount - 1 do
    if Grid.GroupInfo(i).Level = 0 then
      Grid.ExpandGroup(i);
end;

procedure GroupByColumn(Grid: TsgcVCLGrid);
begin
  // the same through the columns: GroupIndex is the level, -1 ungroups
  Grid.Columns.FindByField('Region').GroupIndex := 0;
  Grid.Columns.FindByField('Product').GroupIndex := 1;
end;
```

Voir aussi

- Résumés
- Colonnes figées
- Tri
- TsgcGridLineView
- TsgcGridColumn
- TsgcVCLGrid
- Regroupement et résumés .NET

Résumés

Une colonne avec un SummaryType affiche un résumé de ses valeurs dans la ligne de pied et dans chaque ligne de groupe. Les résumés font partie de toutes les éditions, Community comprise.

SummaryType	Valeur
gsuNone	Aucun résumé (par défaut).
gsuSum	La somme des valeurs numériques.
gsuCount	Le nombre de valeurs qui ne sont ni Null ni du texte vide.
gsuAverage	La somme divisée par le nombre de valeurs numériques.
gsuMin, gsuMax	La plus petite et la plus grande valeur.

- Null, Unassigned et le texte vide sont ignorés. Le texte est converti avec le FormatSettings des données, comme le modèle de données le convertit.
- Les colonnes de texte et booléennes ne prennent en charge que gsuCount.
- Les colonnes de date gardent Min et Max sous forme de dates et n'ont ni Sum ni Average.
- Un résumé sans valeur numérique vaut Null, et gsuCount vaut alors 0.

Format

SummaryFormat est un masque FormatFloat ou FormatDateTime. Le texte littéral se met entre guillemets doubles, par exemple `"Total: "#,##0.00`. Lorsqu'il est vide, le DisplayFormat de la colonne est utilisé, sinon la valeur par défaut du type de données ; gsuCount utilise `0`. Modifier SummaryType ou SummaryFormat actualise les résumés sans reconstruire la vue des données.

Lignes de pied et de groupe

ShowFooter (False par défaut) affiche une ligne de pied de hauteur RowHeight fixée en bas de la grille, avec le résumé de chaque colonne ayant un SummaryType sur les lignes de la vue, de sorte qu'un filtre s'applique. Le pied suit le défilement horizontal et les colonnes figées.
GroupSummaries (True par défaut) affiche les résumés de chaque groupe, sur les lignes du groupe, alignés à droite dans sa ligne de groupe. La colonne visible la plus à gauche contient toujours le libellé : un résumé à cet endroit est dessiné après le libellé s'il tient, sinon seul le pied l'affiche. Voir Regroupement.
FooterSummary(aCol) retourne la valeur de pied d'une colonne (un index de Columns), Null si elle n'en a pas, et FooterSummaryText(aCol) son texte formaté. Les valeurs de groupe proviennent de Lines.GroupSummary(aGroup, aCol). Les résumés sont calculés une fois et recalculés seulement lorsque les données changent, de sorte qu'une image qui les dessine ne coûte rien de plus. En mode virtuel, les valeurs sont lues par OnGetValue pour chaque ligne.

Exemple

```
uses
  SysUtils, Classes, Variants, Forms, Controls,
  sgcGrid_Columns, sgcGrid_Data, sgcGrid_VCL;

procedure ShowTotals(Grid: TsgcVCLGrid);
var
  vQty, vAmount: TsgcGridColumn;
begin
  vQty := Grid.Columns.FindByField('Qty');
  vQty.SummaryType := gsuSum;
```

```
  vAmount := Grid.Columns.FindByField('Amount');
  vAmount.SummaryType := gsuSum;
  // literal text goes in double quotes
  vAmount.SummaryFormat := '"Total: "#,##0.00';

  Grid.Columns.FindByField('Price').SummaryType := gsuAverage;
  Grid.Columns.FindByField('Order').SummaryType := gsuCount;

  // the footer row and the summaries in the group rows
  Grid.ShowFooter := True;
  Grid.GroupSummaries := True;
end;

function AmountTotal(Grid: TsgcVCLGrid): Double;
var
  vValue: Variant;
begin
  // FooterSummary takes a Columns index and returns Null without a value
  vValue := Grid.FooterSummary(Grid.Columns.FindByField('Amount').Index);
  if VarIsNull(vValue) then
    Result := 0
  else
    Result := vValue;
end;

function AmountTotalText(Grid: TsgcVCLGrid): string;
begin
  // the text of the footer cell, formatted with SummaryFormat
  Result := Grid.FooterSummaryText(Grid.Columns.FindByField('Amount').Index);
end;
```

Sans grille, TsgcGridSummaryAccumulator (unité sgcGrid_Summary) additionne des valeurs et retourne tout type de résumé avec Value, TsgcGridSummaryCache garde les valeurs de pied d'un TsgcGridData, et sgcGridSummaryText formate une valeur selon les règles ci-dessus. L'exemple de TsgcGridLineView utilise sgcGridSummaryText.

Voir aussi

- Regroupement
- TsgcGridColumn
- TsgcGridLineView
- Regroupement et résumés .NET

Lignes d'arbre

Les lignes d'arbre affichent les lignes d'une grille sous forme de hiérarchie. Chaque ligne désigne son parent par une colonne clé, et la grille dispose les lignes en profondeur d'abord sous des nœuds d'arbre dépliables, décalées dans une colonne. L'arbre fonctionne avec des lignes en mémoire, des lignes chargées depuis un TDataSet et des lignes modifiées par des mises à jour en direct, aussi bien dans TsgcVCLGrid que dans TsgcFMXGrid.
Cette fonctionnalité fait partie de l'édition Registered. Les packages Community ne la contiennent pas.

Activer l'arbre

La propriété Tree de la grille (TsgcGridTreeOptions, unité sgcGrid_Tree) contient les réglages. Le mode arbre est actif lorsque KeyField et ParentField nomment tous deux des colonnes de la grille ; videz l'une des deux pour le désactiver. Les colonnes de clé et de parent peuvent être masquées.

Propriété	Par défaut	Description
KeyField		La colonne qui contient la clé de chaque ligne.
Parent-Field		La colonne qui contient la clé de la ligne parente. Une valeur vide ou Null fait de la ligne une racine.
TreeField		La colonne qui affiche le décalage et les boutons de l'arbre. Vide utilise la première colonne visible.
Indent	16	Le décalage d'un niveau, en DIP.
Expanded	False	L'état des nœuds qui n'ont jamais été déployés ni repliés. Le modifier déploie ou replie tous les nœuds.

Comment les lignes sont reliées

- Les clés sont comparées comme du texte, en tenant compte de la casse. Null et le texte vide sont la clé vide. Les entiers et les flottants sans partie fractionnaire utilisent leur texte entier, de sorte que 5, 5.0 et '5' sont la même clé. Les autres flottants et les dates utilisent le texte du FormatSettings des données.
- Les parents sont recherchés parmi toutes les lignes stockées. Lorsque plusieurs lignes partagent une clé, la première ligne stockée possède les enfants.
- Une ligne dont le parent est vide, inconnu (orpheline) ou la ligne elle-même devient une racine. Un cycle est rompu à la première ligne qui se répète, qui devient une racine.
- Les frères gardent l'ordre du tri de la grille, de sorte que trier une colonne trie les enfants de chaque nœud et les racines.
- Avec un filtre, une ligne visible dont le parent est filtré se rattache à son plus proche ancêtre visible, ou devient une racine s'il n'y en a pas.
- Des données virtuelles (Data.Virtual) n'affichent pas d'arbre.

Déployer et replier

ExpandNode(aRow) et CollapseNode(aRow) modifient le nœud de la ligne de la vue aRow ; une ligne sans enfants reste telle quelle. En mode arbre, ExpandAll et CollapseAll agissent sur les nœuds de l'arbre, et sur les nœuds créés ensuite. OnTreeExpandChange indique la ligne de la vue et son nouvel état, avec aRow -1 après ExpandAll et CollapseAll. L'état est conservé par clé, de sorte qu'il survit à un tri, à un filtre, à un rechargement des lignes et aux mises à jour en direct. Lorsqu'un repli masque la ligne sélectionnée, son plus proche parent visible est sélectionné.
Un clic sur un bouton de l'arbre bascule le nœud. Touches sur une ligne sélectionnée en mode arbre :

Touche	Action

Right	Déploie le nœud, ou passe à son premier enfant lorsqu'il est déployé.
Left	Replie le nœud, ou passe à son parent lorsqu'il est replié.
+ et -	Déploient et replient le nœud.
*	Déploie le nœud et chaque nœud situé en dessous.
Ctrl+Right et Ctrl+Left	Déploient et replient la bande de détail de la ligne, voir Maître-détail.

Dans TsgcFMXGrid, un appui sur un bouton de l'arbre le bascule, et Cmd fonctionne comme Ctrl.

L'arbre et les autres fonctionnalités

- Le regroupement est ignoré tant que l'arbre est actif. Le GroupIndex des colonnes est conservé et s'applique de nouveau lorsque le mode arbre est désactivé.
- Les bandes de détail fonctionnent en mode arbre : chaque ligne maître déployée affiche sa bande en dessous.
- SaveToXLSX écrit l'arbre sous forme de plan Excel, voir Export Excel.
- Les index de ligne restent des lignes de la vue (SelectedRow, Cells, OnCellClick). Lines.TreeMode indique si l'arbre est actif, et Lines.Tree (TsgcGridTreeBuilder) répond Depth, ParentRow, HasChildren, ChildCount et DescendantCount pour une ligne de la vue, voir TsgcGridLineView.

Exemple

Un organigramme en mémoire. Les gestionnaires se trouvent dans une petite classe, comme ceux des autres exemples de fonctionnalités.

```
uses
  SysUtils, Classes, Variants,
  sgcGrid_Columns, sgcGrid_Data, sgcGrid_VCL;

type
  TTreeHandlers = class
  public
    procedure TreeExpandChange(Sender: TObject; aRow: Integer;
      aExpanded: Boolean);
  end;

procedure TTreeHandlers.TreeExpandChange(Sender: TObject; aRow: Integer;
  aExpanded: Boolean);
begin
  // aRow is the view row of the node, -1 after ExpandAll and CollapseAll
end;

procedure ShowOrgChart(Grid: TsgcVCLGrid; Handlers: TTreeHandlers);
begin
  Grid.Columns.Clear;
  // the key and parent columns may be hidden
  Grid.Columns.Add('id', 'Id', gdtInteger).Visible := False;
  Grid.Columns.Add('manager', 'Manager', gdtInteger).Visible := False;
  Grid.Columns.Add('name', 'Name');
  Grid.Columns.Add('title', 'Title');
  Grid.Data.BeginUpdate;
  try
    Grid.Data.Clear;
    // values by column index: id, manager, name, title
    Grid.Data.AppendRow('1', [1, Null, 'Ana Ruiz', 'CEO']);
    Grid.Data.AppendRow('2', [2, 1, 'Marc Puig', 'CTO']);
    Grid.Data.AppendRow('3', [3, 1, 'Laura Vidal', 'CFO']);
    Grid.Data.AppendRow('4', [4, 2, 'Jordi Mas', 'Developer']);
    Grid.Data.AppendRow('5', [5, 2, 'Eva Soler', 'Developer']);
```

```
  finally
    Grid.Data.EndUpdate;
  end;
  Grid.OnTreeExpandChange := Handlers.TreeExpandChange;
  // KeyField and ParentField switch the tree mode on
  Grid.Tree.KeyField := 'id';
  Grid.Tree.ParentField := 'manager';
  // the column that shows the indentation and the tree buttons
  Grid.Tree.TreeField := 'name';
  Grid.Tree.Indent := 20;
  // collapse every node, then open the root
  Grid.CollapseAll;
  Grid.ExpandNode(Grid.Data.RowOfKey('1'));
end;

function NodeDepth(Grid: TsgcVCLGrid; const aKey: string): Integer;
begin
  // 0 for a root, -1 when the grid is not in tree mode
  Result := -1;
  if Grid.Lines.TreeMode then
    Result := Grid.Lines.Tree.Depth(Grid.Data.RowOfKey(aKey));
end;
```

Voir aussi

- Maître-détail
- Regroupement
- Export Excel
- TsgcVCLGrid
- TsgcGridLineView
- Éditions
- Lignes d'arbre et maître-détail .NET

Maître-détail

Les lignes maître-détail affichent une bande de détail sous chaque ligne maître déployée. La bande est une petite grille à part avec les lignes associées : les commandes d'un client, les lignes d'une facture. Elle a un en-tête, défile et se trie toute seule. Les bandes fonctionnent en mode plat, regroupé et arbre, aussi bien dans TsgcVCLGrid que dans TsgcFMXGrid.
Cette fonctionnalité fait partie de l'édition Registered. Les packages Community ne la contiennent pas.

Options de détail

La propriété Detail de la grille (TsgcGridDetailOptions, unité sgcGrid_Detail) contient les réglages.

Propriété	Par défaut	Description
Active	False	Active les bandes de détail.
Source	nil	Le composant qui contient les lignes de détail : un TsgcGridDataStore ou une grille. La grille le surveille par une notification de libération, de sorte qu'une source libérée affecte nil à Source.
MasterKeyField		La colonne maître dont la valeur est recherchée dans les lignes de détail.
DetailKeyField		La colonne de la source qui contient la clé maître.
Lines	8	La hauteur d'une bande en lignes de grille, en-tête imbriqué compris. La plus petite valeur est 2.
Columns		Les colonnes des grilles de détail. Vide utilise une copie des colonnes de la source. Stocké seulement lorsqu'il contient des colonnes.

Une ligne affiche un expanseur de détail lorsque Active vaut True, que MasterKeyField nomme une colonne maître et qu'une Source ou un gestionnaire OnGetDetailData est défini. Une ligne dont la clé maître est vide ne se déploie jamais, et des données virtuelles n'affichent pas de bandes.

D'où viennent les lignes

- OnGetDetailData est déclenché en premier, sans verrou détenu. Sender est la grille, aRow la ligne maître de la vue, aMasterKey le texte de MasterKeyField et aData les données de la bande. Remplissez aData et affectez True à aHandled pour conserver ce que vous avez écrit ; la Source n'est alors pas lue. Sans Source, les colonnes de la bande viennent de Detail.Columns.
- Sinon la bande est remplie depuis la Source : les lignes stockées de la source dont le DetailKeyField correspond à la clé maître, quels que soient le filtre et le tri de la source. Les lignes sont copiées par FieldName avec leurs clés de source. Une colonne de détail sans champ de source vaut Null, et une source virtuelle ne donne aucune ligne.
- Les clés sont comparées comme du texte, en tenant compte de la casse. Les entiers et les flottants sans partie fractionnaire utilisent leur texte entier, de sorte que 5, 5.0 et '5' sont la même clé.
- Les données de la bande reprennent le FormatSettings de la source, ou celui de la grille maître lorsque les lignes viennent seulement de OnGetDetailData.

Lorsque la source change, chaque bande ouverte est remplie de nouveau à l'image suivante. Les lignes sont mises à jour sur place, de sorte qu'une cellule modifiée clignote comme dans la grille maître. OnGetDetailData est déclenché de nouveau à chaque remplissage.

Déployer et replier

ExpandDetail(aRow) et CollapseDetail(aRow) ouvrent et ferment la bande de la ligne maître de la vue aRow, DetailExpanded(aRow) indique son état, et OnDetailExpandChange indique la ligne et son nouvel état. DetailData(aRow) retourne les données d'une bande ouverte, nil lorsqu'elle est repliée. Elles sont en lecture seule ; verrouillez-les pendant que vous les lisez. L'état est conservé par clé maître, de sorte qu'il survit à un tri, à un filtre, à un rechargement des lignes maître et aux mises à jour en direct.
Un clic sur l'expanseur bascule la bande. Down saute les lignes d'une bande. Touches sur une ligne sélectionnée :

Mode	Touches
Plat ou regroupé, sélection de ligne	Right et Left déploient et replient la bande.
Plat ou regroupé, SelectionMode gsmCell	Ctrl+Right et Ctrl+Left déploient et replient la bande ; Right et Left continuent de se déplacer entre les cellules.
Mode arbre	Ctrl+Right et Ctrl+Left déploient et replient la bande ; Right et Left agissent sur le nœud de l'arbre.

Dans une bande

- La molette de la souris sur une bande fait défiler la bande. À la fin de la bande, c'est la grille qui défile.
- Un clic sur l'en-tête imbriqué trie les lignes de la bande. OnSortChange n'est pas déclenché.
- Un clic sur une cellule imbriquée la sélectionne et déclenche OnDetailCellClick avec la ligne maître, ainsi que la ligne de la vue et l'index de Columns des données de détail. La sélection maître ne bouge pas.
- Les grilles de détail sont en lecture seule et n'ont qu'un niveau. Les bandes ne sont pas écrites par SaveToXLSX.

Dans TsgcFMXGrid, la molette ne fait défiler une bande que tant que le pointeur de la souris est dessus, un balayage tactile continue de faire défiler la grille, un appui sur l'expanseur le bascule, et Cmd fonctionne comme Ctrl.

Exemple

Des clients avec une bande de leurs commandes, conservés dans un TsgcGridDataStore rempli par code :

```
uses
  SysUtils, Classes,
  sgcGrid_Columns, sgcGrid_Data, sgcGrid_Detail, sgcGrid_VCL;

type
  TDetailHandlers = class
  public
    procedure DetailExpandChange(Sender: TObject; aRow: Integer;
      aExpanded: Boolean);
    procedure DetailCellClick(Sender: TObject;
      aRow, aDetailRow, aDetailCol: Integer);
  end;

procedure TDetailHandlers.DetailExpandChange(Sender: TObject; aRow: Integer;
  aExpanded: Boolean);
begin
  // aRow is the master view row whose band was opened or closed
end;

procedure TDetailHandlers.DetailCellClick(Sender: TObject;
  aRow, aDetailRow, aDetailCol: Integer);
var
  oData: TsgcGridData;
  vText: string;
begin
  // aDetailRow and aDetailCol address the data of the band of aRow
```

```
  oData := TsgcVCLGrid(Sender).DetailData(aRow);
  if oData = nil then
    Exit;
  oData.Lock;
  try
    vText := oData.CellText[aDetailRow, aDetailCol];
  finally
    oData.Unlock;
  end;
end;

procedure ShowOrders(Grid: TsgcVCLGrid; Orders: TsgcGridDataStore;
  Handlers: TDetailHandlers);
begin
  // the detail rows: every order names its customer
  Orders.Columns.Add('OrderID', 'Order', gdtInteger);
  Orders.Columns.Add('CustomerID', 'Customer', gdtInteger).Visible := False;
  Orders.Columns.Add('Total', 'Total', gdtCurrency);
  Orders.Data.BeginUpdate;
  try
    Orders.Data.AppendRow('1001', [1001, 1, 250.5]);
    Orders.Data.AppendRow('1002', [1002, 1, 99.9]);
    Orders.Data.AppendRow('1003', [1003, 2, 1200]);
  finally
    Orders.Data.EndUpdate;
  end;
  Grid.OnDetailExpandChange := Handlers.DetailExpandChange;
  Grid.OnDetailCellClick := Handlers.DetailCellClick;
  // the master grid has a CustomerID column
  Grid.Detail.Source := Orders;
  Grid.Detail.MasterKeyField := 'CustomerID';
  Grid.Detail.DetailKeyField := 'CustomerID';
  // band height in grid rows, nested header included
  Grid.Detail.Lines := 6;
  Grid.Detail.Active := True;
  // open the band of the first view row
  Grid.ExpandDetail(0);
end;
```

Les mêmes bandes avec les lignes écrites par OnGetDetailData, par exemple depuis une requête par client :

```
uses
  SysUtils, Classes,
  sgcGrid_Columns, sgcGrid_Data, sgcGrid_VCL;

type
  TOrderLoader = class
  public
    procedure GetDetailData(Sender: TObject; aRow: Integer;
      const aMasterKey: string; aData: TsgcGridData; var aHandled: Boolean);
  end;

procedure TOrderLoader.GetDetailData(Sender: TObject; aRow: Integer;
  const aMasterKey: string; aData: TsgcGridData; var aHandled: Boolean);
begin
  // raised with no lock held; aMasterKey is the MasterKeyField text
  aData.BeginUpdate;
  try
    aData.Clear;
    aData.AppendRow(aMasterKey + '-1', [aMasterKey + '-1', 'Shipped']);
    aData.AppendRow(aMasterKey + '-2', [aMasterKey + '-2', 'Open']);
  finally
    aData.EndUpdate;
  end;
  // the band keeps these rows, the Source is not read
  aHandled := True;
end;
```

```
procedure ShowLoadedOrders(Grid: TsgcVCLGrid; Loader: TOrderLoader);
begin
  // with no Source the bands take their columns from Detail.Columns
  Grid.Detail.Columns.Clear;
  Grid.Detail.Columns.Add('Order', 'Order');
  Grid.Detail.Columns.Add('Status', 'Status');
  Grid.Detail.MasterKeyField := 'CustomerID';
  Grid.Detail.Lines := 5;
  Grid.OnGetDetailData := Loader.GetDetailData;
  Grid.Detail.Active := True;
end;
```

Pour afficher les commandes d'un TDataSet, chargez-le dans un TsgcGridDataStore avec un TsgcGridSource_DataSet, voir TsgcGridDataStore.

Voir aussi

- TsgcGridDataStore
- Lignes d'arbre
- Liaison à un DataSet
- TsgcVCLGrid
- Éditions
- Lignes d'arbre et maître-détail .NET

Sélection et clavier

SelectionMode gsmRow sélectionne des lignes entières et gsmCell des cellules individuelles. SelectedRow et SelectedCol sont la ligne de vue et l'index de colonne, et SelectedKey la clé de la ligne sélectionnée : lorsqu'une mise à jour en direct déplace la ligne, la sélection la suit.
Les exemples sont de simples procédures qui prennent un **TsgcVCLGrid**. Les gestionnaires d'événements sont des méthodes d'une petite classe, TGridHandlers, déclarée dans TsgcVCLGrid. Le même code fonctionne avec TsgcFMXGrid.

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure SetupSelection(Grid: TsgcVCLGrid; Handlers: TGridHandlers);
begin
  Grid.SelectionMode := gsmCell;
  Grid.OnSelectionChange := Handlers.SelectionChange;
  Grid.OnCellClick := Handlers.CellClick;
  // select a row by its key and scroll it into view
  Grid.SelectedRow := Grid.Data.RowOfKey('MSFT');
  if Grid.SelectedRow >= 0 then
    Grid.ScrollToRow(Grid.SelectedRow);
end;
```

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure TGridHandlers.SelectionChange(Sender: TObject);
var
  oGrid: TsgcVCLGrid;
begin
  oGrid := TsgcVCLGrid(Sender);
  if oGrid.SelectedRow >= 0 then
    TForm(oGrid.Owner).Caption := oGrid.SelectedKey + ' ' +
      oGrid.Data.CellText[oGrid.SelectedRow, 1];
end;
```

Clavier

Touche	Action
Haut, Bas	Ligne précédente ou suivante.
Page précédente, Page suivante	Une page vers le haut ou vers le bas.
Début, Fin	Première ou dernière ligne ; en mode cellule, première ou dernière colonne, et Ctrl+Début ou Ctrl+Fin pour la première ou la dernière ligne.
Gauche, Droite	Colonne précédente ou suivante en mode cellule.
Ctrl+C, Ctrl+Insert	Copie la ligne sélectionnée sous forme de texte séparé par des tabulations, ou la cellule sélectionnée en mode cellule. Cmd+C sur macOS.

F2, Entrée, un caractère, Espace	Démarrent la modification lorsque la grille n'est pas ReadOnly. Voir Modification sur place.

Souris

Un clic sélectionne, OnCellClick et OnCellDblClick signalent la cellule, la molette défile, et RowAt et ColAt trouvent la cellule située sous un point.

Voir aussi

- TsgcVCLGrid
- Redimensionnement des colonnes

Redimensionnement des colonnes

Faites glisser le bord droit d'une cellule d'en-tête pour redimensionner la colonne ; le curseur change au-dessus du bord. La largeur ne descend jamais sous le MinWidth de la colonne, 20 par défaut. La nouvelle largeur est stockée dans le Width de la colonne et OnColumnResized se déclenche.
Les exemples sont de simples procédures qui prennent un **TsgcVCLGrid**. Les gestionnaires d'événements sont des méthodes d'une petite classe, TGridHandlers, déclarée dans TsgcVCLGrid. Le même code fonctionne avec TsgcFMXGrid.

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure SetupResizing(Grid: TsgcVCLGrid; Handlers: TGridHandlers);
begin
  Grid.Columns.FindByField('Symbol').MinWidth := 60;
  Grid.OnColumnResized := Handlers.ColumnResized;
end;
```

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure TGridHandlers.ColumnResized(Sender: TObject; aCol: Integer);
begin
  // the new width is already in the column
  OutputDebugString(PChar(Format('%s is now %d wide',
    [TsgcVCLGrid(Sender).Columns[aCol].FieldName,
    TsgcVCLGrid(Sender).Columns[aCol].Width])));
end;
```

Voir aussi

- TsgcGridColumn
- Sélection et clavier

Colonnes figées

FrozenColumns (0 par défaut) garde les premières colonnes visibles en place pendant que les autres colonnes défilent horizontalement, de sorte qu'une colonne clé, comme un symbole ou un nom, reste visible. L'en-tête, le pied et les résumés de groupe suivent la même séparation, et un trait de bordure d'un pixel marque le bord figé. Les colonnes figées font partie de toutes les éditions, Community comprise.

- FrozenColumns compte les colonnes visibles : une colonne masquée est ignorée.
- Les chevrons de groupe à gauche (la gouttière, un GroupIndent par niveau regroupé) ne défilent jamais non plus.
- Dans TsgcVCLGrid, la barre de défilement horizontale native ne couvre que la partie qui défile, de sorte que le plus grand ScrollX ne change pas. TsgcFMXGrid dessine son curseur horizontal sur la partie qui défile de la même façon.
- Les déplacements au clavier amènent dans la vue une colonne défilée située à droite du bord figé.

Le moteur de rendu expose le même réglage dans ses Options, de sorte que la sortie PNG et SVG sans fenêtre correspond au contrôle, et FrozenWidth retourne la largeur qui ne défile pas.

Exemple

```
uses
  SysUtils, Classes, Forms, Controls,
  sgcGrid_Columns, sgcGrid_Data, sgcGrid_Render, sgcGrid_Lines, sgcGrid_VCL;

procedure FreezeKeyColumns(Grid: TsgcVCLGrid);
begin
  // the first two visible columns stay in place while the others scroll
  Grid.FrozenColumns := 2;
end;

procedure SaveFrozenPNG(Data: TsgcGridData; const aFileName: string);
var
  oRenderer: TsgcGridRenderer;
  oLines: TsgcGridLineView;
begin
  oRenderer := TsgcGridRenderer.Create;
  oLines := TsgcGridLineView.Create(Data);
  try
    oRenderer.Options.FrozenColumns := 1;
    oRenderer.Options.ShowFooter := True;
    oRenderer.SaveToPNG(Data, oLines, sgcGridRenderState, aFileName, 800, 400);
  finally
    oLines.Free;
    oRenderer.Free;
  end;
end;
```

Voir aussi

- Redimensionnement des colonnes
- Regroupement
- Rendu sans fenêtre
- TsgcGridRenderer

Clignotement à la modification

Avec **FlashChanges** à True (la valeur par défaut), une cellule dont la valeur a changé est teintée de la couleur de sa direction : FlashUpColor lorsque la valeur a augmenté, FlashDownColor lorsqu'elle a baissé et FlashChanged-Color pour tout autre changement. La teinte s'estompe linéairement pendant **FlashDuration** millisecondes, 800 par défaut.
FlashMode fmBackground teinte l'arrière-plan de la cellule et fmText le texte. Une couleur à 0 utilise la valeur par défaut du thème : vert, rouge et l'accent, translucides pour l'arrière-plan et des couleurs de texte lisibles pour fm-Text, plus claires sur un thème sombre.
Les exemples sont de simples procédures qui prennent un **TsgcVCLGrid**. Les gestionnaires d'événements sont des méthodes d'une petite classe, TGridHandlers, déclarée dans TsgcVCLGrid. Le même code fonctionne avec TsgcFMXGrid.

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure SetupFlash(Grid: TsgcVCLGrid);
begin
  Grid.FlashChanges := True;
  Grid.FlashDuration := 1200; // ms until the tint has faded out
  Grid.FlashMode := fmBackground;
  // 0 keeps the theme colours; an explicit colour is ARGB
  Grid.FlashUpColor := sgcUIARGB($60, $00, $C8, $50);
  Grid.FlashDownColor := sgcUIARGB($60, $E0, $30, $30);
end;
```

Pendant qu'un clignotement s'estompe, la grille continue de demander des images, jusqu'à MaxFPS. La sélection est prioritaire sur le clignotement, et le clignotement sur les règles de mise en forme.

Voir aussi

- Mise en forme conditionnelle
- Thèmes

Mise en forme conditionnelle

Les **FormatRules** d'une grille colorent les cellules dont la valeur remplit une condition. Une règle compare la valeur de la colonne FieldName avec Value, et Value2 pour gfoBetween, selon le DataType de cette colonne : les nombres numériquement, les dates comme des dates (Value en ISO 8601 ou un numéro de série de date), les booléens comme true ou false, le texte sans tenir compte de la casse. gfoContains et gfoStartsWith comparent toujours du texte, gfoIsEmpty correspond à une valeur nulle ou vide, et une cellule vide ne correspond à aucun opérateur d'ordre.
Les exemples sont de simples procédures qui prennent un **TsgcVCLGrid**. Les gestionnaires d'événements sont des méthodes d'une petite classe, TGridHandlers, déclarée dans TsgcVCLGrid. Le même code fonctionne avec TsgcFMXGrid.

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure SetupRules(Grid: TsgcVCLGrid; Handlers: TGridHandlers);
var
  oRule: TsgcGridFormatRule;
begin
  // red text for a negative change
  Grid.FormatRules.Add('Change', gfoLess, '0', 0,
    sgcUIARGB($FF, $D0, $30, $30));
  // the whole row tinted when a symbol is halted
  oRule := Grid.FormatRules.Add('Halted', gfoEqual, 'true',
    sgcUIARGB($40, $F0, $A0, $20));
  oRule.ApplyToRow := True;
  // bold prices between 100 and 200, and no later rule for that cell
  oRule := Grid.FormatRules.Add;
  oRule.FieldName := 'Price';
  oRule.Operator := gfoBetween;
  oRule.Value := '100';
  oRule.Value2 := '200';
  oRule.Bold := True;
  oRule.StopOnMatch := True;
  // the last word, per painted cell
  Grid.OnGetCellStyle := Handlers.GetCellStyle;
end;
```

Priorité

- Les règles sont évaluées dans l'ordre de la collection. Chaque règle activée qui correspond applique ses BackColor et ForeColor non nuls, une correspondance ultérieure l'emportant sur une précédente, et Bold s'accumule.
- Une règle qui correspond avec StopOnMatch met fin à l'évaluation pour cette cellule.
- Une règle ApplyToRow teste sa propre colonne et, en cas de correspondance, applique le style à chaque cellule de la ligne.
- Lorsque la grille dessine, la sélection l'emporte sur le clignotement, le clignotement sur les règles et les règles sur la teinte des lignes alternées.
- OnGetCellStyle est déclenché en dernier pour chaque cellule dessinée : aFore et aBold portent le résultat obtenu jusque-là, et aBack est un remplissage supplémentaire par-dessus.

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure TGridHandlers.GetCellStyle(Sender: TObject; aRow, aCol: Integer;
  var aBack, aFore: TsgcUIColor; var aBold: Boolean);
```

```
var
  oRenderer: TsgcGridRenderer;
begin
  // Sender is the renderer; aBack is an extra fill over the rule and flash
  oRenderer := TsgcGridRenderer(Sender);
  if (aCol = 0) and (oRenderer.CurrentData.KeyOf(aRow) = 'MSFT') then
    aBold := True;
end;
```

Voir aussi

- TsgcGridFormatRule
- Clignotement à la modification
- Interroger la grille

Modification sur place

La modification sur place nécessite SGC_GRID_EDIT, que l'édition Community ne définit pas. Voir Éditions.

Démarrer une modification

Définissez **ReadOnly** sur False. Une colonne est modifiable tant que son Editable vaut True. F2, Entrée, un caractère saisi (qui remplace le texte) ou un double-clic ouvre un TEdit sur la cellule, ou un TComboBox lorsque la colonne a une PickList. StartEdit l'ouvre depuis le code, et OnBeforeEdit peut le refuser.

Valider et annuler

Entrée valide, Tab et Maj+Tab valident et passent à la cellule modifiable suivante ou précédente, et Échap annule. Perdre le focus valide, ou annule lorsque le texte est invalide. CommitEdit et CancelEdit font de même depuis le code.

Conversion et validation

Le texte est converti vers le DataType de la colonne avec Data.FormatSettings. Un texte qui ne se convertit pas garde l'éditeur ouvert et déclenche OnEditError, dont aMessage peut être modifié. OnValidate reçoit la valeur convertie et peut la modifier ou la rejeter. La valeur est ensuite écrite avec Data.UpdateCell, de sorte que le suivi des modifications, le clignotement et le tri s'appliquent, et OnEdited signale l'ancienne et la nouvelle valeur.

Booléens

Une colonne gdtBoolean n'a pas d'éditeur : Espace, un clic, F2, Entrée ou StartEdit basculent la valeur.

Mises à jour en direct pendant la modification

Lorsqu'une mise à jour en direct retrie ou filtre la vue, l'éditeur suit sa ligne, retrouvée par sa clé. Lorsque la ligne est supprimée ou filtrée, la modification est annulée.
Les exemples sont de simples procédures qui prennent un **TsgcVCLGrid**. Les gestionnaires d'événements sont des méthodes d'une petite classe, TGridHandlers, déclarée dans TsgcVCLGrid. Le même code fonctionne avec TsgcFMXGrid.

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure SetupEditing(Grid: TsgcVCLGrid; Handlers: TGridHandlers);
begin
  Grid.Columns.Clear;
  Grid.Columns.Add('Item', 'Item').Editable := False;
  Grid.Columns.Add('Qty', 'Qty', gdtInteger);
  Grid.Columns.Add('Side', 'Side').PickList.CommaText := 'Buy,Sell';
  Grid.Columns.Add('Urgent', 'Urgent', gdtBoolean);
  Grid.Data.AppendRow('1', ['Widget', 10, 'Buy', False]);

  Grid.ReadOnly := False;
  Grid.OnValidate := Handlers.Validate;
  Grid.OnEdited := Handlers.Edited;
  Grid.OnEditError := Handlers.EditError;
  // open the editor on the Qty cell of the first row from code
```

```
  Grid.StartEdit(0, 1);
end;

uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure TGridHandlers.Validate(Sender: TObject; aRow, aCol: Integer;
  var aValue: Variant; var aAccept: Boolean);
begin
  // aValue is already converted to the DataType of the column
  if TsgcVCLGrid(Sender).Columns[aCol].FieldName = 'Qty' then
    aAccept := aValue >= 0;
end;

procedure TGridHandlers.Edited(Sender: TObject; aRow, aCol: Integer;
  const aOldValue, aNewValue: Variant);
begin
  OutputDebugString(PChar(VarToStr(aOldValue) + ' -> ' + VarToStr(aNewValue)));
end;

procedure TGridHandlers.EditError(Sender: TObject; aRow, aCol: Integer;
  const aText: string; var aMessage: string);
begin
  aMessage := 'Not a valid quantity: ' + aText;
end;
```

Voir aussi

- TsgcGridColumn
- TsgcVCLGrid

Mode virtuel

En mode virtuel, la grille ne stocke aucune ligne. **VirtualRowCount** définit le nombre de lignes, et **OnGetValue** des données retourne la valeur d'une ligne et d'une colonne lorsque le moteur de rendu la dessine. Comme seules les lignes visibles sont dessinées, une grille d'un million de lignes coûte autant qu'une grille de cinquante.
Les lignes virtuelles n'ont pas de clé, la vue est donc constituée des lignes dans l'ordre : il n'y a ni tri, ni filtrage, ni suivi des modifications, et KeyOf et RowOfKey ne retournent rien. Modifiez VirtualRowCount lorsque le nombre de lignes change.
Les exemples sont de simples procédures qui prennent un **TsgcVCLGrid**. Les gestionnaires d'événements sont des méthodes d'une petite classe, TGridHandlers, déclarée dans TsgcVCLGrid. Le même code fonctionne avec TsgcFMXGrid.

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure ShowVirtual(Grid: TsgcVCLGrid; Handlers: TGridHandlers);
begin
  Grid.Columns.Clear;
  Grid.Columns.Add('Name', 'Name');
  Grid.Columns.Add('Value', 'Value', gdtFloat);
  Grid.Data.OnGetValue := Handlers.GetValue;
  Grid.Data.VirtualRowCount := 1000000;
  Grid.Data.Virtual := True;
end;
```

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure TGridHandlers.GetValue(Sender: TObject; aRow, aCol: Integer;
  var aValue: Variant);
begin
  case aCol of
    0: aValue := 'Row ' + IntToStr(aRow);
    1: aValue := aRow * 1.5;
  end;
end;
```

Voir aussi

- TsgcGridData
- Performances

JSON et CSV

L'unité sgcGrid_IO charge et enregistre un TsgcGridData en JSON ou en CSV, depuis et vers des chaînes, des flux et des fichiers. Elle est indépendante du framework et se compile aussi sous Linux64.

Chargement

- Un chargement remplace les lignes et s'exécute dans un seul BeginUpdate et EndUpdate. Avec un champ clé, la clé d'une ligne est le texte de ce champ ; une clé répétée met à jour la ligne précédente.
- Avec aAutoColumns à True et sans colonnes existantes, les colonnes sont créées à partir de l'entrée et leur DataType est deviné à partir des valeurs. Les colonnes existantes sont conservées et associées par FieldName.
- L'entrée JSON est un tableau d'objets. Un FieldName avec des points, tel que price.last, lit des membres imbriqués.
- Le CSV suit la RFC 4180, le premier enregistrement contient les FieldName. aInvariant à True (par défaut) lit et écrit les nombres avec un point et les dates au format ISO 8601 ; False utilise les FormatSettings des données.

Enregistrement

GridSaveToJSON et GridSaveToCSV écrivent les lignes dans l'ordre de la vue, triées et filtrées, avec les colonnes dans l'ordre, uniquement les visibles lorsque aVisibleOnly est à True. Le texte est en UTF-8 sans BOM.
Les exemples sont de simples procédures qui prennent un **TsgcVCLGrid**. Les gestionnaires d'événements sont des méthodes d'une petite classe, TGridHandlers, déclarée dans TsgcVCLGrid. Le même code fonctionne avec TsgcFMXGrid.

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure ExportImport(Grid: TsgcVCLGrid);
var
  vJSON: string;
begin
  // the rows in view order, as an array of objects
  vJSON := GridSaveToJSON(Grid.Data);
  // the visible columns only, invariant numbers and ISO dates
  GridSaveToCSVFile(Grid.Data, 'quotes.csv', True);

  // a load replaces the rows; the key of a row is its Symbol
  GridLoadFromJSON(Grid.Data,
    '[{"Symbol":"MSFT","Price":412.5},{"Symbol":"AAPL","Price":228.1}]',
    'Symbol');
  // semicolon delimited, numbers in the FormatSettings of the data
  GridLoadFromCSVFile(Grid.Data, 'quotes.csv', 'Symbol', True, ';', False);
end;
```

Voir aussi

- Liaison à un DataSet
- TsgcGridData

Export Excel

SaveToXLSX écrit la grille sous forme de classeur Excel (Office Open XML, une feuille de calcul) dans un flux ou un fichier, sans bibliothèque externe. Le classeur garde ce que la grille affiche : les colonnes visibles dans l'ordre d'affichage avec leurs largeurs et leurs formats numériques actuels, l'en-tête figé et FrozenColumns, les groupes ou l'arbre sous forme de plan repliable, et les résumés sous forme de formules.
Cette fonctionnalité fait partie de l'édition Registered. Les packages Community ne la contiennent pas.

Exporter une grille

Méthode	Description
SaveToXLSX(aStream; aSheetName = 'Sheet1')	Écrit le classeur dans un flux.
SaveToXLSX(aFileName; aSheetName = 'Sheet1')	Écrit le classeur dans un fichier.

Appelez-la dans le thread de l'interface. Le nom de la feuille perd les caractères que Excel refuse (ils deviennent _) et est coupé à 31 caractères. Une feuille qui a besoin de plus de 1 048 576 lignes ou de 16 384 colonnes déclenche EsgcGridXLSXError avant que quoi que ce soit ne soit écrit.

Ce que contient la feuille

- La ligne 1 est l'en-tête en gras, figé avec les premières FrozenColumns colonnes. Les largeurs de colonne sont les largeurs à l'écran.
- Chaque ligne de groupe et chaque ligne de la vue est écrite, quoi qu'il soit replié. Les lignes d'un groupe replié sont masquées et la ligne de groupe est marquée comme repliée, de sorte qu'Excel ouvre le plan tel que la grille le montre. Les lignes de groupe se trouvent au-dessus de leurs lignes.
- Une ligne de groupe contient son libellé (Region: North (12)) dans la première colonne, ou dans la première colonne sans résumé. OnGetGroupText n'est pas utilisé.
- Les lignes d'arbre reçoivent le niveau de plan de leur profondeur et un décalage sur la colonne de l'arbre. Les niveaux de plan s'arrêtent à 7, le maximum d'Excel.
- Les bandes de détail ne sont pas exportées.
- Les nombres sont écrits comme des nombres, les dates comme des dates sérielles Excel, les booléens comme des booléens et le texte comme des chaînes partagées sans caractères de contrôle.
- Le DisplayFormat d'une colonne devient le format numérique Excel. Les masques FormatFloat sont repris tels quels ; dans les masques de date, nn devient mm, z devient 000 et am/pm devient AM/PM. Sans DisplayFormat, une colonne d'entiers utilise 0, une colonne monétaire #,##0.00 et une colonne de date yyyy-mm-dd hh:mm:ss.

Résumés

Chaque colonne de résumé d'une ligne de groupe contient une formule SUBTOTAL sur les lignes du groupe, avec la valeur de la grille mise en cache dans la cellule, de sorte que le fichier affiche les totaux avant qu'Excel ne recalcule. Lorsqu'une colonne visible a un résumé, une ligne de pied contient SUBTOTAL sur toutes les lignes de données. La ligne de pied est écrite quelle que soit la valeur de ShowFooter.

SummaryType	SUBTOTAL	Types de colonne
gsuSum	9	Nombres
gsuAverage	1	Nombres
gsuMin	5	Nombres et dates

gsuMax	4	Nombres et dates
gsuCount	3	Tout type (cellules non vides)

Ces codes comptent les lignes masquées, de sorte qu'un groupe replié garde son total, et SUBTOTAL ignore les cellules SUBTOTAL imbriquées des groupes intérieurs.

TsgcGridXLSXWriter

L'écrivain derrière SaveToXLSX se trouve dans l'unité sgcGrid_XLSX et n'a pas besoin de grille : il écrit n'importe quel TsgcGridData.

Membre	Description
SaveToStream(aData, aLines, aStream), SaveToFile(aData, aLines, aFileName)	Écrit le classeur de aData. aLines donne le regroupement, l'arbre et l'état déployé ; nil utilise une vue de lignes propre.
SheetName	Le nom de la feuille de calcul, 'Sheet1' par défaut.
FrozenColumns	Les premières colonnes visibles gardées dans la vue.
SetColumnWidth(aCol, aDIPs)	La largeur de la colonne aCol (index de Columns) en DIP, utilisée à la place de sa Width. Une valeur négative rétablit Width.
Summaries	Écrit les cellules SUBTOTAL de groupe et de pied, True par défaut.
FileTime	La date et l'heure des entrées du zip, Now à la création de l'écrivain. Les mêmes données et le même FileTime donnent toujours les mêmes octets.
sgcGridExcelNumberFormat(aDataType, aDisplayFormat)	Le code de format numérique Excel d'un type de colonne et d'un DisplayFormat.

Exemple

```
uses
  SysUtils, Classes,
  sgcGrid_Data, sgcGrid_XLSX, sgcGrid_VCL;

function ExportGrid(Grid: TsgcVCLGrid; const aFileName: string): Boolean;
begin
  // the visible columns with their current widths, FrozenColumns, the
  // groups or the tree as an outline and the summaries
  Result := True;
  try
    Grid.SaveToXLSX(aFileName, 'Orders');
  except
    on E: EsgcGridXLSXError do
      // too many rows or columns, nothing was written
      Result := False;
  end;
end;

procedure ExportToStream(Grid: TsgcVCLGrid; aStream: TStream);
begin
  // the sheet is named Sheet1
```

```
  Grid.SaveToXLSX(aStream);
end;

procedure ExportData(aData: TsgcGridData; aStream: TStream);
var
  oWriter: TsgcGridXLSXWriter;
begin
  // headless: no grid needed
  oWriter := TsgcGridXLSXWriter.Create;
  try
    oWriter.SheetName := 'Prices';
    oWriter.FrozenColumns := 1;
    // no SUBTOTAL rows
    oWriter.Summaries := False;
    // a fixed time gives the same bytes for the same rows
    oWriter.FileTime := EncodeDate(2026, 1, 1);
    // first column 160 DIPs wide instead of its Width
    oWriter.SetColumnWidth(0, 160);
    // nil: the writer uses a line view of its own
    oWriter.SaveToStream(aData, nil, aStream);
  finally
    oWriter.Free;
  end;
end;
```

Voir aussi

- JSON et CSV
- Résumés
- Regroupement
- Lignes d'arbre
- Colonnes figées
- Éditions
- Export Excel .NET

Liaison à un DataSet

Un TsgcGridSource_DataSet remplit une grille à partir de n'importe quel TDataSet. Avec un KeyField, la grille est synchronisée plutôt que rechargée : les lignes modifiées sont mises à jour sur place et clignotent, les nouvelles clés sont ajoutées et les clés manquantes supprimées. Avec LiveUpdates, la grille suit chaque modification validée du dataset.
Les exemples sont de simples procédures qui prennent un **TsgcVCLGrid**. Les gestionnaires d'événements sont des méthodes d'une petite classe, TGridHandlers, déclarée dans TsgcVCLGrid. Le même code fonctionne avec TsgcFMXGrid.

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

function BindDataSet(Grid: TsgcVCLGrid; DataSet: TDataSet)
  : TsgcGridSource_DataSet;
begin
  Result := TsgcGridSource_DataSet.Create(Grid.Owner);
  Result.Grid := Grid;
  Result.DataSet := DataSet;
  Result.KeyField := 'ID'; // rows are updated in place by ID
  Result.AutoColumns := True; // columns from the fields when the grid has none
  Result.LiveUpdates := True; // follow Post, Delete and Refresh
  Result.Active := True;
end;
```

Le dataset est parcouru avec DisableControls et son enregistrement courant est restauré. Fermer le dataset conserve les lignes. La démo DataSetGrid montre la liaison avec un TClientDataSet.

Voir aussi

- TsgcGridSource_DataSet
- JSON et CSV
- Démos

Flux en direct

Les sources de flux sgcGridWS écrivent directement dans une grille les messages JSON des clients sgcWebSockets : TsgcGridFeedSource_WebSocket, TsgcGridFeedSource_MQTT et TsgcGridFeedSource_SSE.
Cette fonctionnalité fait partie du pont sgcGridWS : les unités nécessitent les sources ou les packages de sgcWebSockets, et l'édition Community ne l'inclut pas.

Règles de correspondance

Propriété	Règle
RowsPath	L'emplacement des lignes dans un message. Vide : le message est un objet ligne ou un tableau de lignes.
KeyPath	Le chemin de la clé dans une ligne, tel que symbol ou id.value. Une chaîne, un nombre ou un booléen est accepté.
Fields	Lignes FieldName=chemin.json ; une ligne sans = lit le chemin de même nom ; vide, lit chaque colonne depuis le chemin nommé comme son FieldName. Seuls les chemins présents sont écrits, les mises à jour partielles fonctionnent donc.
DeletePath	Un chemin booléen qui supprime la ligne de la clé lorsqu'il vaut true.

WebSocket

```
uses
  Windows, SysUtils, Classes, Forms,
  sgcWebSocket, sgcWebSocket_Protocols, sgcHTTP, sgcHTML_AI,
  sgcGrid_Columns, sgcGrid_VCL, sgcGridWS_Source, sgcGridWS_Source_WebSocket,
  sgcGridWS_Source_MQTT, sgcGridWS_Source_SSE, sgcGridWS_AI;

function FeedFromWebSocket(Grid: TsgcVCLGrid; Client: TsgcWebSocketClient)
  : TsgcGridFeedSource_WebSocket;
begin
  // messages such as {"quotes":[{"symbol":"MSFT","quote":{"last":412.5}}]}
  Result := TsgcGridFeedSource_WebSocket.Create(Grid.Owner);
  Result.Grid := Grid;
  Result.RowsPath := 'quotes';
  Result.KeyPath := 'symbol';
  Result.Fields.Add('Symbol=symbol');
  Result.Fields.Add('Price=quote.last');
  Result.Fields.Add('Volume'); // same-named path
  Result.DeletePath := 'removed';
  Result.Client := Client;
  Client.URL := 'wss://quotes.example.com/stream';
  Client.Active := True;
end;
```

MQTT

```
uses
  Windows, SysUtils, Classes, Forms,
  sgcWebSocket, sgcWebSocket_Protocols, sgcHTTP, sgcHTML_AI,
  sgcGrid_Columns, sgcGrid_VCL, sgcGridWS_Source, sgcGridWS_Source_WebSocket,
  sgcGridWS_Source_MQTT, sgcGridWS_Source_SSE, sgcGridWS_AI;
```

```
function FeedFromMQTT(Grid: TsgcVCLGrid; MQTT: TsgcWSPClient_MQTT)
  : TsgcGridFeedSource_MQTT;
begin
  Result := TsgcGridFeedSource_MQTT.Create(Grid.Owner);
  Result.Grid := Grid;
  Result.KeyPath := 'id';
  Result.Topic := 'plant/+/sensors';
  Result.Client := MQTT;
  // the source does not subscribe: subscribe on the MQTT client, for
  // example in its OnMQTTConnect event
end;
```

Événements envoyés par le serveur (SSE)

```
uses
  Windows, SysUtils, Classes, Forms,
  sgcWebSocket, sgcWebSocket_Protocols, sgcHTTP, sgcHTML_AI,
  sgcGrid_Columns, sgcGrid_VCL, sgcGridWS_Source, sgcGridWS_Source_WebSocket,
  sgcGridWS_Source_MQTT, sgcGridWS_Source_SSE, sgcGridWS_AI;

function FeedFromSSE(Grid: TsgcVCLGrid; HTTP: TsgcHTTP1Client)
  : TsgcGridFeedSource_SSE;
begin
  Result := TsgcGridFeedSource_SSE.Create(Grid.Owner);
  Result.Grid := Grid;
  Result.KeyPath := 'orderId';
  Result.Client := HTTP;
  // the application starts the stream
  HTTP.GetSSE('https://api.example.com/orders/stream');
end;
```

Messages provenant d'ailleurs

```
uses
  Windows, SysUtils, Classes, Forms,
  sgcWebSocket, sgcWebSocket_Protocols, sgcHTTP, sgcHTML_AI,
  sgcGrid_Columns, sgcGrid_VCL, sgcGridWS_Source, sgcGridWS_Source_WebSocket,
  sgcGridWS_Source_MQTT, sgcGridWS_Source_SSE, sgcGridWS_AI;

procedure FeedByHand(Source: TsgcGridFeedSource_WebSocket);
begin
  // apply a message that did not come from the client, returns the rows
  Source.ProcessMessage('[{"symbol":"MSFT","Price":413.2},' +
    '{"symbol":"IBM","removed":true}]');
end;
```

Les lignes d'un message sont appliquées dans un seul BeginUpdate et EndUpdate, dans le thread du client, ce qui est sûr car les données sont thread-safe. La démo LiveFeed montre la source WebSocket avec un serveur dans le même processus.

Voir aussi

- TsgcGridFeedSource
- Mises à jour en direct et threads

Interroger la grille

TsgcGridAIQuery permet à un utilisateur de filtrer, trier et mettre en évidence une grille avec une question en langage courant, telle que price greater than 100 sort by volume descending highlight change less than 0.
Cette fonctionnalité fait partie du pont sgcGridWS : les unités nécessitent les sources ou les packages de sgcWebSockets, et l'édition Community ne l'inclut pas.

Comment la réponse est appliquée

- Les filtres sont combinés avec AND et appliqués par le OnFilterRow des données. Un gestionnaire déjà affecté est conservé et appelé en premier.
- Les tris remplacent le SortOrder et le SortIndex des colonnes.
- Les mises en évidence sont ajoutées aux FormatRules de la grille avec ApplyToRow activé.
- Le regroupement (group by) et les limites de lignes ne sont pas pris en charge par la grille ; ils sont ignorés et signalés dans LastExplanation.
- Chaque Ask appelle d'abord Reset, de sorte que les questions ne s'accumulent pas. Reset restaure le gestionnaire de filtre et le tri précédents et ne supprime que les règles ajoutées par la requête.

```
uses
  Windows, SysUtils, Classes, Forms,
  sgcWebSocket, sgcWebSocket_Protocols, sgcHTTP, sgcHTML_AI,
  sgcGrid_Columns, sgcGrid_VCL, sgcGridWS_Source, sgcGridWS_Source_WebSocket,
  sgcGridWS_Source_MQTT, sgcGridWS_Source_SSE, sgcGridWS_AI;

function AskGrid(Grid: TsgcVCLGrid): string;
var
  oQuery: TsgcGridAIQuery;
begin
  oQuery := TsgcGridAIQuery.Create(Grid.Owner);
  oQuery.Grid := Grid;
  // no Assistant: the rule-based parser answers, no network, no API key
  if oQuery.Ask('price greater than 100 sort by volume descending ' +
    'highlight change less than 0') then
    Result := oQuery.LastExplanation
  else
    Result := 'not understood';
  // later: remove the filter, the sort and the highlight
  oQuery.Reset;
end;
```

Avec un LLM

Affectez un TsgcHTMLAIAssistant avec un fournisseur pour répondre avec un LLM. Ask est synchrone et bloque pendant que le fournisseur répond, appelez-le donc depuis le thread principal. Sans Assistant, c'est l'analyseur à base de règles qui répond, sans réseau ni clé d'API, comme dans la démo AskYourGrid.

```
uses
  Windows, SysUtils, Classes, Forms,
  sgcWebSocket, sgcWebSocket_Protocols, sgcHTTP, sgcHTML_AI,
  sgcGrid_Columns, sgcGrid_VCL, sgcGridWS_Source, sgcGridWS_Source_WebSocket,
  sgcGridWS_Source_MQTT, sgcGridWS_Source_SSE, sgcGridWS_AI;

procedure UseAssistant(Query: TsgcGridAIQuery; Assistant: TsgcHTMLAIAssistant);
begin
  // an Assistant with a provider answers with an LLM; Ask blocks meanwhile
  Query.Assistant := Assistant;
  Query.Ask('show the halted symbols with the biggest loss first');
end;
```

Voir aussi

- TsgcGridAIQuery
- Filtrage
- Mise en forme conditionnelle

Graphiques à partir de la grille

Les données d'une grille implémentent IsgcUITable, l'interface de table de sgcUICore, et les grilles la délèguent. La source de données sgcCharts **TsgcChartDataSource_Table** lit une telle table : affectez sa propriété Table à la grille, et XColumn et YColumn à des FieldName de colonnes. Chaque modification des lignes de la grille, une mise à jour d'un flux ou une modification sur place, atteint le graphique sans code intermédiaire, et le graphique suit le tri et le filtre de la grille.

```
uses
  Forms, Classes, Controls,
  sgcGrid_Columns, sgcGrid_VCL,
  sgcCharts_Chart, sgcCharts_Series, sgcCharts_DataSource_Table, sgcCharts_VCL;

procedure ChartFromGrid(Form: TForm; Grid: TsgcVCLGrid);
var
  oChart: TsgcChart;
  oSeries: TsgcChartSeries_Line;
  oSource: TsgcChartDataSource_Table;
  oView: TsgcVCLChart;
begin
  Grid.Columns.Add('x', 'x', gdtFloat);
  Grid.Columns.Add('y', 'y', gdtFloat);
  Grid.Data.AppendRow('1', [1, 10]);
  Grid.Data.AppendRow('2', [2, 14]);

  oChart := TsgcChart.Create(Form);
  oSeries := TsgcChartSeries_Line.Create(Form);
  oSeries.Chart := oChart;
  oSource := TsgcChartDataSource_Table.Create(Form);
  oSource.Series := oSeries;
  oSource.XColumn := 'x'; // a column FieldName
  oSource.YColumn := 'y';
  oSource.Table := Grid; // the grid implements IsgcUITable

  oView := TsgcVCLChart.Create(Form);
  oView.Parent := Form;
  oView.Align := alRight;
  oView.Chart := oChart;

  // every change of the grid rows now reaches the chart
  Grid.Data.UpdateRow('2', ['y'], [18]);
end;
```

L'exemple nécessite les unités sgcCharts. La démo GridAndChart place la grille et le graphique dans deux tuiles sgcDashboard.

Voir aussi

- IsgcGridDataHost et IsgcUITable
- Démos

Thèmes

La grille utilise les jetons de thème clair et sombre de sgcUICore : Theme utLight (par défaut) ou utDark, AccentColor pour la sélection et le clignotement des modifications, et FontSize, RowHeight, HeaderHeight, AlternateRows et les lignes de grille pour la densité. Les couleurs sont des valeurs ARGB TsgcUIColor ; sgcUIARGB en construit une.
Un TsgcVCLGrid placé dans une tuile sgcDashboard suit le thème du tableau de bord grâce à sa propriété publiée Theme.
Les exemples sont de simples procédures qui prennent un **TsgcVCLGrid**. Les gestionnaires d'événements sont des méthodes d'une petite classe, TGridHandlers, déclarée dans TsgcVCLGrid. Le même code fonctionne avec TsgcFMXGrid.

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure SetupTheme(Grid: TsgcVCLGrid);
begin
  Grid.Theme := utDark;
  Grid.AccentColor := sgcUIARGB($FF, $E0, $80, $20);
  Grid.RowHeight := 24;
  Grid.HeaderHeight := 30;
  Grid.FontSize := 12;
  Grid.AlternateRows := False;
  Grid.GridLinesV := False;
  Grid.MaxFPS := 30; // repaint at most 30 times per second
end;
```

Voir aussi

- Clignotement à la modification
- Rendu sans fenêtre

Rendu sans fenêtre

Le moteur de rendu n'a pas besoin de fenêtre. SaveToPNG et SaveToSVG des grilles écrivent la vue courante ; les procédures sgcGridSaveToPNG et sgcGridSaveToSVG, ou un TsgcGridRenderer, rendent un TsgcGridData autonome, y compris sous Linux64, pour des rapports, des e-mails ou des images générées côté serveur. L'image de l'Introduction a été produite par l'exemple ci-dessous.

```
uses
  SysUtils,
  sgcUICore_Theme, sgcGrid_Columns, sgcGrid_Data, sgcGrid_Format,
  sgcGrid_Render;

var
  oData: TsgcGridData;
  oOptions: TsgcGridRenderOptions;
  vState: TsgcGridRenderState;
begin
  // nil: the data creates and owns its own columns
  oData := TsgcGridData.Create(nil);
  oOptions := TsgcGridRenderOptions.Create;
  try
    // '.' decimals and ',' thousands whatever the locale
    oData.UseInvariantFormat;
    oData.Columns.Add('Symbol', 'Symbol');
    oData.Columns.Add('Price', 'Price', gdtFloat).DisplayFormat := '#,##0.00';
    oData.Columns.Add('Change', 'Change %', gdtFloat).DisplayFormat :=
      '+0.00;-0.00;0.00';
    oData.Columns.Add('Volume', 'Volume', gdtInteger).DisplayFormat := '#,##0';
    oData.Columns.Add('Halted', 'Halted', gdtBoolean).Width := 70;
    oData.AppendRow('MSFT', ['MSFT', 412.5, 1.25, 1200500, False]);
    oData.AppendRow('AAPL', ['AAPL', 228.1, -0.84, 3400200, False]);
    oData.AppendRow('GOOG', ['GOOG', 171.9, 0.37, 980100, False]);
    oData.AppendRow('AMZN', ['AMZN', 186.4, -2.10, 2250000, True]);
    oData.AppendRow('NVDA', ['NVDA', 121.7, 3.02, 5120750, False]);
    oData.Columns[1].SortOrder := gsoDescending;

    oOptions.Theme := utDark;
    oOptions.FormatRules.Add('Change', gfoLess, '0', 0,
      sgcUIARGB($FF, $F0, $60, $60));
    oOptions.FormatRules.Add('Volume', gfoGreater, '5000000', 0, 0, True);

    vState := sgcGridRenderState;
    vState.SelectedRow := 1;
    // 470 x 200 logical pixels, written at scale 2 as 940 x 400 pixels
    sgcGridSaveToPNG(oData, oOptions, vState, 'grid.png', 470, 200, 2);
    sgcGridSaveToSVG(oData, oOptions, vState, 'grid.svg', 470, 200);
  finally
    oOptions.Free;
    oData.Free;
  end;
end;
```

À partir d'une grille

```
uses
  Windows, SysUtils, Classes, Variants, Forms, Controls, DB,
  sgcUICore_Canvas, sgcUICore_Theme, sgcUICore_Table, sgcGrid_Columns,
  sgcGrid_Data, sgcGrid_Format, sgcGrid_Render, sgcGrid_IO,
  sgcGrid_Source_DataSet, sgcGrid_VCL;

procedure SaveGridImage(Grid: TsgcVCLGrid);
begin
```

```
  // the rows of the current view, without a window, 800 x 400 pixels
  Grid.SaveToPNG('grid.png', 800, 400);
  Grid.SaveToSVG('grid.svg', 800, 400);
end;
```

Voir aussi

- TsgcGridRenderer
- Thèmes

Vue d'ensemble de .NET

sgcGrid .NET est un portage managé en C# de la bibliothèque sgcGrid de Delphi, et non une enveloppe autour d'elle. Les classes C# se trouvent dans l'espace de noms **esegece.sgcGrid** et gardent les noms Delphi, si bien que **TsgcGridData**, **TsgcGridColumns**, **TsgcGridColumn**, **TsgcGridFormatRules**, **TsgcGridRenderer** et **TsgcGridSource_DataSet** se lisent de la même façon dans les deux langages. Les contrôles de grille suivent le schéma de nommage de Delphi : **TsgcWinFormsGrid** pour WinForms, **TsgcWpfGrid** pour WPF et **TsgcAvaloniaGrid** pour Avalonia. Tous les trois dessinent avec le portage .NET de **sgcUICore**, avec le même canevas, les mêmes polices et les mêmes thèmes clair et sombre que les autres produits de la suite visuelle d'eSeGeCe, si bien qu'une grille a le même aspect sur chaque hôte et dans la sortie sans fenêtre en PNG et SVG.
La grille est conçue pour les données en direct. Le stockage des lignes est indexé par clé et sûr pour les threads : un thread de travail peut donc mettre à jour, ajouter et supprimer des lignes pendant que la grille ne dessine que les lignes visibles, à une cadence d'images contrôlée. Les cellules modifiées clignotent, les règles de mise en forme colorent des cellules ou des lignes entières, et la grille trie, filtre, fonctionne en mode virtuel et exporte en JSON et CSV. Dans l'édition Registered, l'utilisateur modifie les cellules sur place, le pont sgcWebSockets alimente la grille à partir de connexions WebSocket, MQTT et SSE, et le pont sgcHTML répond en langage naturel à des questions sur les lignes.

Packages

Le portage est réparti en packages NuGet, de sorte qu'un projet n'inclut que ce qu'il utilise.

PackageId	Contient	Frameworks cibles
esegece.sgcGrid	Le modèle (**TsgcGridColumns**, **TsgcGridData**, **TsgcGridFormatSettings**), le tri et le filtrage, le mode virtuel, le clignotement et les règles de mise en forme (**TsgcGridFormatRules**), le moteur de rendu **TsgcGridRenderer** avec la sortie SVG sans fenêtre, l'import et l'export JSON et CSV (**sgcGrid_IO**) et la source DataTable **TsgcGridSource_DataSet**. Aucune dépendance native.	netstandard2.0, net472, net6.0, net8.0, net9.0
esegece.sgcGrid.Rendering	La sortie PNG sans fenêtre (**SaveToPNG**), via SkiaSharp.	netstandard2.0, net472, net6.0, net8.0, net9.0
esegece.sgcGrid.WinForms	**TsgcWinFormsGrid**.	net472, net8.0-windows, net9.0-windows
esegece.sgcGrid.Wpf	**TsgcWpfGrid**.	net472, net8.0-windows, net9.0-windows
esegece.sgcGrid.Avalonia	**TsgcAvaloniaGrid**, sur Avalonia 11.3.	net8.0, net9.0
esegece.sgcGrid.sgcWebSockets	Le pont des flux en direct : **TsgcGridFeedSource**, **TsgcGridFeedSource_WebSocket**, **TsgcGridFeedSource_MQTT** et	netstandard2.0,

	TsgcGridFeedSource_SSE. Il dépend du package esegece.sgcWebSockets. Édition Registered uniquement.	net8.0, net9.0
esegece.sgcGrid.sgcHTML	Le pont des requêtes IA : **TsgcGridAIQuery** et **TsgcGridAIFilterItem**, au-dessus du **TsgcHTMLAIAssistant** du package esegece.sgcHTML. Édition Registered uniquement.	netstandard2.0, net8.0, net9.0

Chaque package d'hôte dépend d'esegece.sgcGrid.Rendering et du package sgcUICore du même hôte, il suffit donc d'ajouter le package de l'hôte.

Éditions

Édition	Ids des packages	Comprend
Registered	esegece.sgcGrid, .Rendering, .WinForms, .Wpf, .Avalonia, .sgcWebSockets et .sgcHTML	Tout : les trois grilles, les mises à jour en direct, le clignotement et les règles de mise en forme, le tri, le filtrage, le mode virtuel, JSON et CSV, la source DataTable, la sortie sans fenêtre, la modification sur place et les deux ponts.
Community	Les mêmes ids avec le suffixe .Community, par exemple esegece.sgcGrid.WinForms.Community. Il n'existe pas de packages de pont Community.	Tout sauf la modification sur place (**StartEdit**, **CommitEdit**, **CancelEdit**, les événements **OnBeforeEdit**, **OnValidate**, **OnEdited** et **OnEditError**) et les deux ponts.

C'est la même répartition que celle des éditions Delphi, voir Éditions.

Hôtes

Hôte	Équivalent Delphi	Plateformes
WinForms	TsgcVCLGrid	Windows, sur .NET Framework 4.7.2 et .NET 8 et 9.
WPF	TsgcFMXGrid	Windows, sur .NET Framework 4.7.2 et .NET 8 et 9.
Avalonia	TsgcFMXGrid	Windows, Linux et macOS, sur .NET 8 et 9.
Headless	TsgcGridRenderer	Toute plateforme de .NET Standard 2.0, serveurs Linux compris.

Voir aussi

- Installation depuis NuGet
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Classes .NET
- Différences entre .NET et Delphi
- Rendu sans fenêtre .NET
- Liaison DataTable, JSON et CSV en .NET
- Modification sur place en .NET
- Flux en direct .NET
- Interroger la grille en .NET
- Performances .NET
- Démos .NET

Installation depuis NuGet

Les packages se trouvent sur nuget.org. Ajoutez le package de l'hôte, les packages core et de rendu arrivent avec lui comme dépendances.

Édition Registered

```
dotnet add package esegece.sgcGrid.WinForms
dotnet add package esegece.sgcGrid.Wpf
dotnet add package esegece.sgcGrid.Avalonia
```

Ajoutez les ponts pour les flux en direct et les requêtes en langage naturel. Ils apportent avec eux les packages esegece.sgcWebSockets et esegece.sgcHTML :

```
dotnet add package esegece.sgcGrid.sgcWebSockets
dotnet add package esegece.sgcGrid.sgcHTML
```

Édition Community

```
dotnet add package esegece.sgcGrid.WinForms.Community
dotnet add package esegece.sgcGrid.Wpf.Community
dotnet add package esegece.sgcGrid.Avalonia.Community
```

Ou ajoutez la référence à la main dans le fichier de projet :

```
<ItemGroup>
  <PackageReference Include="esegece.sgcGrid.WinForms" Version="2026.10.0" />
</ItemGroup>
```

Dans un projet, référencez soit les packages Registered, soit les packages Community, jamais les deux : ils contiennent les mêmes types dans le même espace de noms. Le code qui utilise l'API de modification ne se compile pas avec les packages Community ; protégez-le avec `#if !SGC_COMMUNITY` lorsqu'un même projet compile les deux éditions.

Paramètres du projet

- WinForms : un projet `net8.0-windows` avec `<UseWindowsForms>true</UseWindowsForms>`, ou un projet .NET Framework 4.7.2.
- WPF : un projet `net8.0-windows` avec `<UseWPF>true</UseWPF>`, ou un projet .NET Framework 4.7.2. Associez l'espace de noms dans le XAML avec `xmlns:grid="clr-namespace:esegece.sgcGrid;assembly=esegece.sgcGrid.Wpf"`.
- Avalonia : un projet `net8.0` ou `net9.0` avec les packages **Avalonia.Desktop** et **Avalonia.Themes.Fluent** d'Avalonia 11.3, comme pour toute application de bureau Avalonia. Associez l'espace de noms dans le XAML avec `xmlns:grid="using:esegece.sgcGrid"`.
- Sans fenêtre : esegece.sgcGrid seul suffit pour le modèle, la source DataTable, JSON, CSV et SVG. Ajoutez esegece.sgcGrid.Rendering pour le PNG, il apporte SkiaSharp et ses ressources natives pour Windows et Linux.
- Ponts : esegece.sgcGrid.sgcWebSockets et esegece.sgcGrid.sgcHTML ciblent netstandard2.0, net8.0 et net9.0, ils fonctionnent donc avec tous les hôtes, .NET Framework 4.7.2 compris.

En C#, chaque hôte a besoin de `using esegece.sgcGrid;`, et de `using esegece.sgcUICore;` pour **TsgcUITheme** et les autres types des thèmes. Les sources de flux en direct ont aussi besoin de `using esegece.sgcWebSockets;` pour les clients de socket.

Voir aussi

- Vue d'ensemble de .NET
- Démarrage rapide WinForms

- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Flux en direct .NET
- Interroger la grille en .NET
- Démos .NET

Démarrage rapide WinForms

Cet exemple construit un formulaire WinForms avec une grille de cotations. Un thread de travail met à jour les lignes quatre fois par seconde, les cellules modifiées clignotent, une règle de mise en forme peint en rouge clair les lignes dont la variation est négative, et une case à cocher fait passer la grille du thème clair au thème sombre.

Étapes

- Créez un projet WinForms `net8.0-windows` et ajoutez le package **esegece.sgcGrid.WinForms**, voir Installation depuis NuGet.
- Créez un **TsgcWinFormsGrid** et ancrez-le. Ajoutez les colonnes avec **Columns.Add** : le nom du champ, le titre et le **TsgcGridDataType**, et définissez **DisplayFormat** pour les nombres et les dates.
- Ajoutez les lignes avec **Data.AppendRow** : le premier argument est la clé de la ligne et les valeurs suivent l'ordre des colonnes.
- Mettez à jour les lignes par clé avec **Data.UpdateRow** ou **Data.UpdateCell**, depuis n'importe quel thread. **BeginUpdate** et **EndUpdate** regroupent de nombreuses modifications en un seul redessin.
- Ajoutez une règle à **FormatRules** pour colorer une cellule, ou la ligne entière avec **ApplyToRow**.

Exemple

Les couleurs sont des valeurs `$AARRGGBB` ; un alpha inférieur à `FF` teinte la cellule au-dessus de son arrière-plan. La grille se redessine au plus **MaxFPS** fois par seconde, quelle que soit la vitesse à laquelle les données changent, voir Mises à jour en direct et threads et Performances .NET.

Voir aussi

- Installation depuis NuGet
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Classes .NET
- TsgcVCLGrid
- Clignotement à la modification
- Mise en forme conditionnelle
- Démos .NET

Démarrage rapide WPF

La même grille de cotations en WPF. La grille et ses options d'affichage sont déclarées en XAML ; les colonnes et les lignes sont définies dans le code-behind.

Étapes

- Créez un projet WPF `net8.0-windows` et ajoutez le package **esegece.sgcGrid.Wpf**, voir Installation depuis NuGet.
- Associez l'espace de noms dans la fenêtre avec `xmlns:grid="clr-namespace:esegece.sgcGrid;assembly=esegece.sgcGrid.Wpf"`.
- Déclarez un **TsgcWpfGrid** et définissez **RowHeight**, **FlashChanges**, **TailMode** et les autres options d'affichage sous forme d'attributs.
- Ajoutez les colonnes, les règles de mise en forme et les lignes dans le constructeur de la fenêtre, après `InitializeComponent`.

Exemple

MainWindow.xaml :
MainWindow.xaml.cs :
App.xaml est celui d'un nouveau projet WPF, avec `StartupUri="MainWindow.xaml"`. La grille WPF dessine ses propres barres de défilement, comme le fait la grille FireMonkey, voir Différences entre .NET et Delphi.

Voir aussi

- Installation depuis NuGet
- Démarrage rapide WinForms
- Démarrage rapide Avalonia
- Classes .NET
- Différences entre .NET et Delphi
- TsgcFMXGrid
- Démos .NET

Démarrage rapide Avalonia

La même grille de cotations en Avalonia, sous Windows, Linux et macOS.

Étapes

- Créez un projet de bureau Avalonia `net8.0` avec **Avalonia.Desktop** et **Avalonia.Themes.Fluent** 11.3, et ajoutez le package **esegece.sgcGrid.Avalonia**, voir Installation depuis NuGet.
- Associez l'espace de noms dans la fenêtre avec `xmlns:grid="using:esegece.sgcGrid"`.
- Déclarez un **TsgcAvaloniaGrid** avec un `Name`, retrouvez-le après `AvaloniaXamlLoader.Load` et ajoutez les colonnes et les lignes.
- Changez **Theme** depuis le code-behind.

Exemple

MainWindow.axaml :
MainWindow.axaml.cs :
Program.cs :
Sur un écran tactile, le contenu suit le doigt et un appui sélectionne la cellule. Par une référence de type `Control`, **Theme** est le thème de contrôle d'Avalonia, voir Différences entre .NET et Delphi.

Voir aussi

- Installation depuis NuGet
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Classes .NET
- Différences entre .NET et Delphi
- TsgcFMXGrid
- Démos .NET

Classes .NET

Toutes les classes se trouvent dans l'espace de noms **esegece.sgcGrid**. Les trois contrôles de grille sont de fins hôtes au-dessus du modèle et du moteur de rendu indépendants du framework, qui sont partagés par les hôtes, la sortie sans fenêtre et les ponts.

Classes d'hôte

Hôte	Classe	Classe de base	Delphi
WinForms	TsgcWinFormsGrid	TsgcUIWinFormsControl	TsgcVCLGrid
WPF	TsgcWpfGrid	TsgcUIWpfControl	TsgcFMXGrid
Avalonia	TsgcAvaloniaGrid	TsgcUIAvaloniaControl	TsgcFMXGrid

Les trois grilles implémentent **IsgcUITable**, si bien qu'un graphique sgcCharts peut les lire par **TsgcChartDataSource_Table**.

Types indépendants du framework

Type	Description
TsgcGridColumns, TsgcGridColumn	Les colonnes : **FieldName**, **Caption**, **DataType**, **Width**, **MinWidth**, **Alignment**, **DisplayFormat**, **Visible**, **Editable**, **SortOrder**, **SortIndex**, **PickList**, ainsi que **GroupIndex**, **SummaryType** et **SummaryFormat** pour le regroupement et les résumés. Voir TsgcGridColumn.
TsgcGridData	Le stockage des lignes, indexé par clé et sûr pour les threads : **AppendRow**, **UpdateRow**, **UpdateCell**, **DeleteRow**, **Clear**, **BeginUpdate** et **EndUpdate**, estampilles de modification, mode tail avec **MaxRows**, tri, **FilterText** et **OnFilterRow**, mode virtuel (**Virtual**, **VirtualRowCount**, **OnGetValue**) et **FormatSettings**. **LayoutVersion** avance à chaque modification pouvant changer le regroupement ou l'arbre, et **SetLayoutFields** ajoute des champs dont les changements de valeur la font aussi avancer. Voir TsgcGridData.
TsgcGridDataType	`gdtText` , `gdtInteger` , `gdtFloat` , `gdtCurrency` , `gdtDateTime` et `gdtBoolean` .
TsgcGridFormatRules, TsgcGridFormatRule	Mise en forme conditionnelle : **FieldName**, **Operator** (de `gfoEqual` à `gfoIsEmpty`), **Value**, **Value2**, **BackColor**, **ForeColor**, **Bold**, **ApplyToRow**, **Enabled** et **StopOnMatch**. Voir TsgcGridFormatRule.
TsgcGridSummaryType	`gsuNone` , `gsuSum` , `gsuCount` , `gsuAverage` , `gsuMin` et `gsuMax` .
TsgcGridLineView, TsgcGridGroupInfo	Les lignes d'affichage d'un TsgcGridData : lignes de groupe, lignes de données, lignes d'arbre et bandes de détail, avec **GroupBy**, **GroupCount**, **GroupInfo**, **GroupSummary** et l'état déployé. Voir Regroupement et résumés .NET.
TsgcGridTreeOptions	Lignes d'arbre : **KeyField**, **ParentField**, **TreeField**, **Indent** et **Expanded**. Voir Lignes d'arbre et maître-détail .NET. Édition Registered uniquement.
TsgcGridDetailOptions	Bandes de détail : **Active**, **Source**, **MasterKeyField**, **DetailKeyField**, **Lines** et **Columns**. Voir Lignes d'arbre et maître-détail .NET. Édition Registered uniquement.

TsgcGridDataStore	Un conteneur non visuel de lignes de détail, rempli depuis le code ou par un **TsgcGridSource_DataSet**. Édition Registered uniquement.
TsgcGridXLSXWriter	Écrit un TsgcGridData et sa vue en lignes sous forme de classeur Excel. Voir Export Excel .NET. Édition Registered uniquement.
TsgcGridRenderer, TsgcGridRenderOptions	Dessine une image d'un **TsgcGridData** sur un **TsgcUICanvas** et l'écrit en SVG ou, avec esegece.sgcGrid.Rendering, en PNG. Voir Rendu sans fenêtre .NET.
sgcGrid_IO	Import et export JSON et CSV, voir Liaison DataTable, JSON et CSV en .NET.
TsgcGridSource_DataSet	Lie un `System.Data.DataTable` à une grille, voir Liaison DataTable, JSON et CSV en .NET.
TsgcGridFeedSource et ses descendants	Les sources de flux en direct d'esegece.sgcGrid.sgcWebSockets, voir Flux en direct .NET. Édition Registered uniquement.
TsgcGridAIQuery	Les questions en langage naturel d'esegece.sgcGrid.sgcHTML, voir Interroger la grille en .NET. Édition Registered uniquement.

Grille

Les mêmes membres dans **TsgcWinFormsGrid**, **TsgcWpfGrid** et **TsgcAvaloniaGrid**.

Membre	Type	Description
Columns	TsgcGridColumns	Les colonnes.
Data	TsgcGridData	Le stockage des lignes, en lecture seule. Il peut être écrit depuis n'importe quel thread.
FormatRules	TsgcGridFormatRules	Les règles de mise en forme conditionnelle.
Renderer	TsgcGridRenderer	Le moteur de rendu de la grille, en lecture seule.
HeaderHeight, RowHeight	int	Hauteur de l'en-tête et d'une ligne, 32 et 28 par défaut.
GridLinesH, GridLinesV, AlternateRows	bool	Lignes de quadrillage horizontales et verticales et teinte des lignes alternées, true par défaut.
SelectionMode	TsgcGridSelectionMode	`gsmRow` (par défaut) ou `gsmCell`.
Theme, AccentColor	TsgcUITheme, uint	Thème de la grille et couleur d'accent facultative `$AARRGGBB`, 0 pour l'accent du thème.
FontSize	double	Taille de police des cellules et de l'en-tête, 13 par défaut.
MaxRows, TailMode	int, bool	Mode tail : les lignes les plus anciennes sont supprimées au-delà de **MaxRows**, et **TailMode** garde la dernière ligne en vue.
FilterText	string	N'affiche que les lignes dont le texte mis en forme le contient.
FlashChanges, FlashDuration, FlashMode	bool, int, TsgcGridFlashMode	Clignotement des modifications, 800 ms par défaut, sur l'arrière-plan ou sur le texte.

FlashUpColor, FlashDownColor, FlashChangedColor	uint	Couleurs du clignotement, 0 pour les couleurs du thème.
ReadOnly	bool	True par défaut. False active la modification sur place des colonnes **Editable**. Édition Registered uniquement.
SelectedRow, SelectedCol, SelectedKey	int, int, string	La sélection. La sélection suit sa ligne par la clé lorsque les lignes sont retriées.
TopRow, ScrollX, ScrollToRow	int, int, method	La position de défilement. **TopRow** compte des lignes d'affichage, qui sont les lignes de la vue quand rien n'est regroupé.
RowAt, ColAt	method	La ligne de la vue et la colonne situées sous un point.
StartEdit, CommitEdit, CancelEdit, EditorActive	method, bool	Modification sur place depuis le code. Édition Registered uniquement.
SaveToSVG, SaveToPNG	method	Écrit la vue actuelle sous forme d'image.
OnCellClick, OnCellDblClick	TsgcGridCellEvent	Une cellule a été cliquée, avec la ligne de la vue et l'index de la colonne.
OnSelectionChange, OnSortChange	EventHandler	La sélection ou le tri a changé.
OnColumnResized	TsgcGridColumnEvent	L'utilisateur a redimensionné une colonne.
OnGetCellStyle	TsgcGridCellStyleEvent	Modifie les couleurs d'une cellule avant qu'elle ne soit dessinée.
OnBeforeEdit, OnValidate, OnEdited, OnEditError	events	Modification sur place, voir Modification sur place en .NET. Édition Registered uniquement.
ShowGroupPanel, GroupPanelHeight, GroupIndent, GroupsExpanded, GroupSummaries	bool, int, int, bool, bool	Options d'affichage du regroupement, voir Regroupement et résumés .NET.
ShowFooter, FrozenColumns	bool, int	Le pied de résumés et les colonnes qui restent visibles pendant que les autres défilent.
Lines, FocusedLine	TsgcGridLineView, int	Les lignes d'affichage et la ligne ayant le focus.
GroupBy, ClearGrouping, ExpandAll, CollapseAll, ExpandGroup, CollapseGroup, GroupCount, GroupInfo, FooterSummary, FooterSummaryText	method	Regroupement et résumés depuis le code.
OnGroupingChange, OnGroupExpandChange, OnGetGroupText	événements	Le regroupement ou l'état d'un groupe a changé, et le libellé d'une ligne de groupe.
Tree, Detail	TsgcGridTreeOptions, TsgcGridDetailOptions	Lignes d'arbre et bandes de détail, voir Lignes d'arbre et maître-détail .NET. Édition Registered uniquement.

ExpandNode, CollapseNode, ExpandDetail, CollapseDetail, DetailExpanded, DetailData	method	Nœuds d'arbre et bandes de détail depuis le code. Édition Registered uniquement.
OnTreeExpandChange, OnDetailExpandChange, OnDetailCellClick, OnGetDetailData	événements	Événements d'arbre et de détail. Édition Registered uniquement.
SaveToXLSX	method	Écrit la vue sous forme de classeur Excel, voir Export Excel .NET. Édition Registered uniquement.
VertBarVisible, HorzBarVisible, BorderStyle	bool, bool, TsgcGridBorderStyle	WPF et Avalonia : les barres de défilement dessinées et la bordure. WinForms utilise le **BorderStyle** de `System.Windows.Forms` et des barres de défilement natives.

Voir aussi

- Vue d'ensemble de .NET
- Différences entre .NET et Delphi
- Rendu sans fenêtre .NET
- Liaison DataTable, JSON et CSV en .NET
- Flux en direct .NET
- Interroger la grille en .NET
- Regroupement et résumés .NET
- Lignes d'arbre et maître-détail .NET
- Export Excel .NET
- TsgcGridData
- TsgcGridRenderer

Différences entre .NET et Delphi

Le portage .NET conserve les noms, les valeurs par défaut et les règles de rendu de Delphi ; sa sortie SVG est identique octet pour octet à celle de Delphi. Les différences ci-dessous viennent des frameworks des hôtes et de la bibliothèque de classes .NET.

Data et FormatRules au lieu de IsgcGridDataHost

En Delphi, une source trouve les données d'une grille par l'interface IsgcGridDataHost et IsgcUITable. Le portage .NET n'a pas cette interface. **TsgcWinFormsGrid**, **TsgcWpfGrid** et **TsgcAvaloniaGrid** exposent **Data** et **FormatRules** comme propriétés publiques, et **TsgcGridSource_DataSet**, les sources de flux et **TsgcGridAIQuery** les trouvent par réflexion : **Grid** accepte tout objet doté d'une propriété publique **Data** de type **TsgcGridData**, et la requête IA lit aussi, pour ses surlignages, une propriété publique **FormatRules** de type **TsgcGridFormatRules**. Votre propre classe peut donc être le **Grid** d'une source, voir l'exemple dans Interroger la grille en .NET. **GridData**, lorsqu'il est assigné, l'emporte toujours sur **Grid**.
Il n'existe pas de notification de libération en .NET. Lorsque **Grid** est un `IComponent`, comme un contrôle WinForms, la source écoute son événement `Disposed` et efface alors **Grid**. Pour tout autre objet, mettez **Grid** à null avant que la grille ne disparaisse. **GridData** n'est pas surveillé.

Barres de défilement

TsgcWinFormsGrid utilise les barres de défilement natives de Windows, comme la grille VCL. **TsgcWpfGrid** et **TsgcAvaloniaGrid** dessinent leurs propres barres de défilement, comme la grille FireMonkey, avec glissement du curseur et défilement par page sur la piste ; **VertBarVisible** et **HorzBarVisible** les affichent ou les masquent.

FontSize sous WPF, Theme sous Avalonia

La grille dessine son propre texte, donc **TsgcWpfGrid** déclare son propre **FontSize** avec `new`, ce qui masque `Control.FontSize`. Par une référence de type `Control`, c'est la propriété WPF, que la grille ne lit pas. De la même façon, chaque contrôle Avalonia possède déjà une propriété **Theme**, le `ControlTheme` de `StyledElement` : **TsgcAvaloniaGrid** déclare son propre **Theme** de type **TsgcUITheme** avec `new`. Par une référence du type de la grille, c'est le thème sgc ; par une référence de type `Control` ou `StyledElement`, c'est le thème de contrôle d'Avalonia.
Sous WPF et Avalonia, les deux propriétés acceptent les liaisons et les styles, comme la sélection et les autres options d'affichage, voir Liaison de données .NET.

Tactile

Sous Avalonia, un contact tactile fait défiler la grille, le contenu suit le doigt sans défilement cinétique, comme le fait le geste de panoramique de FireMonkey, et un appui sélectionne la cellule. Là, un contact tactile ne fait jamais glisser un en-tête vers le panneau de regroupement ; un appui sur un chevron, un expanseur ou une bande de détail agit comme un clic. WPF n'a pas de panoramique tactile : le toucher et le stylet sont promus en entrée souris, si bien qu'un appui sélectionne et que les barres de défilement dessinées font défiler. WinForms utilise la souris et la molette.

Types de données

Les types CLR sont associés aux types de données de la grille comme les types de champ Delphi : `byte`, `short`, `int`, `long` et les autres types entiers sont associés à `gdtInteger` ; `float`, `double` et `decimal` sont associés à `gdtFloat`, comme les champs Delphi `ftBCD` et `ftFMTBcd` ; `DateTime`, `DateTimeOffset` et `TimeSpan` sont associés à `gdtDateTime` ; `bool` est associé à `gdtBoolean` ; tout le reste est `gdtText`. Le CLR n'a pas de type currency, donc `gdtCurrency` ne provient que d'une colonne que vous ajoutez vous-même.

DataTable au lieu de TDataSet

TsgcGridSource_DataSet lie un `System.Data.DataTable`, l'équivalent .NET d'un `TDataSet` Delphi. Un DataTable est toujours ouvert et n'a pas de curseur : il n'y a donc ni enregistrement courant ni signet à restaurer après un chargement. Les événements de **LiveUpdates** (`RowChanged`, `RowDeleted`, `TableCleared`) remplacent le lien de données Delphi, et les modifications inachevées ainsi que `AcceptChanges` sont ignorés. Voir Liaison DataTable, JSON et CSV en .NET.

Règles de mise en forme sans ID d'élément

Les règles de mise en forme .NET n'ont pas d'ID d'élément de collection. **TsgcGridAIQuery.Reset** supprime les règles qu'il a ajoutées par référence plutôt que par ID ; les colonnes sont toujours retrouvées par leur **ID**.

Édition des lignes d'arbre, des bandes de détail et de l'export Excel

Dans Delphi, les lignes d'arbre, les bandes de détail et **SaveToXLSX** sont compilées sous la définition `SGC_GRID_ADVANCED`. Les packages Community de .NET ne contiennent ni **TsgcGridTreeOptions**, ni **TsgcGridDetailOptions**, ni **TsgcGridDataStore**, ni **TsgcGridXLSXWriter**, ni les membres de la grille qui les utilisent ; le regroupement, les résumés, le pied et les colonnes figées sont dans les deux éditions. Voir Lignes d'arbre et maître-détail .NET et Export Excel .NET.

Événements de regroupement, d'arbre et de détail

Les nouveaux événements de la grille sont des événements C#, comme les autres événements des grilles. Leurs délégués, **TsgcGridGroupExpandEvent**, **TsgcGridRowExpandEvent** et **TsgcGridDetailCellEvent**, sont déclarés une seule fois dans esegece.sgcGrid et partagés par les trois hôtes. Les méthodes sans argument, comme **GroupCount**, restent des méthodes.

Source du détail

Detail.Source est un `object` : un **TsgcGridDataStore**, un **TsgcGridData** ou tout objet ayant une propriété publique **Data** de type **TsgcGridData**, trouvé comme les sources trouvent une grille. Sans notification de libération, une Source qui est un `IComponent` est effacée quand son événement `Disposed` se déclenche.

Fichiers Excel

Le classeur est compressé avec `System.IO.Compression.DeflateStream`. Ses parties XML sont les mêmes octets que la sortie de Delphi pour les mêmes données ; seuls les octets compressés, et donc la taille du fichier, diffèrent.

Sérialisation par le concepteur des nouvelles options

Il n'y a pas de DFM. Sous WinForms, les sept options de regroupement ont des valeurs par défaut et ne sont sérialisées que lorsqu'elles diffèrent, et **Tree** et **Detail** sont sérialisées comme contenu. Sous WPF et Avalonia, les options sont des propriétés de dépendance et des propriétés stylées.

Pas de verbes de concepteur

Les grilles Delphi ajoutent des verbes de conception à l'IDE. Les grilles .NET n'en ont aucun : ajoutez les **Columns** et les **FormatRules** dans le code, comme le font les exemples de démarrage rapide.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Liaison DataTable, JSON et CSV en .NET
- Interroger la grille en .NET
- Lignes d'arbre et maître-détail .NET
- Export Excel .NET
- IsgcGridDataHost et IsgcUITable

Liaison de données .NET

Cette rubrique traite uniquement de **TsgcWpfGrid** et **TsgcAvaloniaGrid**. Leurs propriétés scalaires visibles par l'utilisateur sont des propriétés de dépendance WPF et des propriétés stylées Avalonia, si bien qu'elles acceptent les liaisons, les styles et les setters XAML, et qu'un modèle de vue peut piloter la grille sans code-behind. **TsgcWinFormsGrid** ne change pas : ses propriétés sont de simples propriétés CLR, comme avant.

Propriétés pouvant être liées

Les propriétés ci-dessous acceptent une liaison sur les deux grilles. Celles marquées TwoWay sont liées en TwoWay par défaut, si bien qu'une liaison sans `Mode` renvoie aussi au modèle de vue les modifications de la grille. Toutes les autres sont liées en OneWay par défaut.

Propriété	Type	Mode par défaut
SelectedRow	int	TwoWay
SelectedCol	int	TwoWay
Theme	TsgcUITheme	OneWay
FontSize	double	OneWay
BorderStyle	TsgcGridBorderStyle	OneWay
ReadOnly	bool	OneWay
TailMode	bool	OneWay
MaxRows	int	OneWay
FilterText	string	OneWay
HeaderHeight	int	OneWay
RowHeight	int	OneWay
GridLinesH	bool	OneWay
GridLinesV	bool	OneWay
AlternateRows	bool	OneWay
SelectionMode	TsgcGridSelectionMode	OneWay
AccentColor	uint	OneWay
FlashChanges	bool	OneWay
FlashDuration	int	OneWay
FlashUpColor	uint	OneWay
FlashDownColor	uint	OneWay
FlashChangedColor	uint	OneWay

FlashMode	TsgcGridFlashMode	OneWay

SelectedRow et **SelectedCol** suivent un clic, une touche ou le code. Sous Avalonia, ce sont des propriétés directes. Sous WPF, **FontSize** est déclarée avec `new` et masque `Control.FontSize`, et sous Avalonia **Theme** masque le thème de contrôle, voir Différences de .NET par rapport à Delphi. Liez-les sur l'élément grille lui-même, comme le font les exemples.

Comportement

- Le modèle de la grille conserve sa validation. Lorsqu'il rejette ou corrige une valeur, par exemple une **FlashDuration** négative ou un **SelectedRow** au-delà de la dernière ligne, la grille réécrit dans la propriété la valeur qu'elle a conservée. Sous WPF, lorsque la valeur provenait d'une source TwoWay, la valeur conservée retourne aussi à cette source un tick du dispatcher plus tard, une fois que la liaison a terminé son transfert.
- Lorsque le code modifie directement le modèle via **Renderer.Options**, la nouvelle valeur est recopiée dans la propriété pouvant être liée uniquement sur le thread de l'interface, si bien qu'un modèle de vue lié n'est jamais mis à jour depuis un autre thread.
- **OnSelectionChange** et les autres événements se déclenchent pour chaque modification, qu'elle provienne d'une liaison, d'un style, du code, de la souris ou du clavier.
- La sélection suit sa clé de ligne lorsque **FilterText** déplace la ligne, et le **SelectedRow** lié la suit aussi.

Propriétés qui restent CLR

- **Data**, **Columns** et **FormatRules** restent des propriétés CLR, car **TsgcGridSource_DataSet**, les sources de flux et **TsgcGridAlQuery** les trouvent par réflexion. Remplissez-les depuis le code ou via une source, voir Liaison DataTable, JSON et CSV en .NET et Flux en direct .NET.
- **TopRow** et **ScrollX** restent des propriétés CLR, car ce sont des états de la vue qui changent à chaque pas de défilement.
- Les propriétés d'état en lecture seule, comme **SelectedKey**, **VertBarVisible**, **HorzBarVisible** et **Renderer**, restent des propriétés CLR, car il n'y a rien à leur lier.

Modèle de vue

Les deux exemples utilisent le même modèle de vue. **SelectedRow** est lié en TwoWay, **FilterText** est partagé avec une zone de texte, et **Theme** est calculé à partir d'une case à cocher.
MainViewModel.cs :

Exemple WPF

MainWindow.xaml :
MainWindow.xaml.cs :

Exemple Avalonia

La fenêtre Avalonia utilise des liaisons compilées, elle définit donc `x:DataType` sur le modèle de vue.
MainWindow.axaml :
MainWindow.axaml.cs :
Définir `vm.SelectedRow = 3` sélectionne la ligne AMZN, et un clic sur une autre ligne réécrit son index dans `vm.SelectedRow`. Saisir du texte dans la zone de texte filtre la grille, et la case à cocher bascule le thème.

Voir aussi

- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Classes .NET
- Différences de .NET par rapport à Delphi

- Liaison DataTable, JSON et CSV en .NET
- Flux en direct .NET

Rendu sans fenêtre .NET

Le moteur de rendu n'a besoin ni d'un contrôle ni d'une fenêtre : un service, une application web ou un serveur Linux peut donc écrire l'image d'une grille. **TsgcGridRenderer.SaveToSVG** écrit du SVG depuis le package core. **SaveToPNG**, une méthode d'extension d'esegece.sgcGrid.Rendering, écrit du PNG via SkiaSharp.

Exemple

Ce programme console construit le modèle, le trie, ajoute une règle de mise en forme et écrit la même image en SVG et en PNG à deux fois la taille :
Le programme écrit :

```
North America: 318,950.00
Europe: 245,300.50
Asia Pacific: 162,420.75
Latin America: 71,880.20
regions.svg 6846 bytes, regions.png 49632 bytes
```

Remarques

- La largeur et la hauteur sont en pixels indépendants de l'appareil ; le dernier argument de **SaveToSVG** et de **SaveToPNG** est l'échelle, si bien que `520 x 170` avec une échelle de 2 écrit une image de 1040 x 340 pixels.
- `sgcGrid_Render.sgcGridRenderState()` renvoie un état sans sélection et sans défilement. Définissez ses champs pour rendre une image défilée ou avec une sélection.
- Les options de rendu (**Theme**, **FontSize**, **RowHeight**, les paramètres de clignotement et **FormatRules**) sont celles qu'exposent les contrôles de grille, et un contrôle de grille écrit sa propre vue avec **SaveToSVG** et **SaveToPNG**.
- Le texte SVG est identique octet pour octet à celui que le moteur de rendu Delphi écrit pour les mêmes données. La sortie PNG utilise le lissage de Skia, elle peut donc différer de la trame Delphi sur quelques pixels.
- **UseInvariantFormat** rend **CellText** indépendant de la culture de la machine.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- Performances .NET
- Rendu sans fenêtre
- TsgcGridRenderer

Liaison DataTable, JSON et CSV en .NET

TsgcGridSource_DataSet lie un `System.Data.DataTable` à une grille, et la classe **sgcGrid_IO** charge et enregistre les lignes en JSON et CSV. Les deux se trouvent dans le package core et fonctionnent avec tous les hôtes et sans fenêtre.

Liaison DataTable

- Définissez **Grid** sur le contrôle de grille, ou **GridData** sur son **Data**.
- Avec **AutoColumns** à true (la valeur par défaut) et une grille sans colonnes, les colonnes sont créées à partir du DataTable : le nom du champ est `ColumnName`, le titre est `Caption`, une colonne dont le `ColumnMapping` est `Hidden` est masquée, et le type de données suit le type CLR (`decimal` est associé à `gdtFloat`).
- **KeyField** nomme la colonne dont la valeur est la clé de la ligne. Avec une clé, un rechargement met à jour les lignes sur place et conserve leur clignotement de modification, ajoute les nouvelles clés et supprime celles qui ont disparu. Sans clé, chaque chargement remplace les lignes.
- Avec **LiveUpdates** à true (la valeur par défaut), une modification validée, une suppression ou un vidage du DataTable parvient aussitôt à la grille. **Load** charge à tout moment et **Refresh** recharge.

Le programme écrit :

```
sku "sku" gdtText
name "Product" gdtText
stock "Stock" gdtInteger
price "Price" gdtFloat
3 rows
A-200: Mouse, stock 95
B-300: Monitor, stock 8
B-400: Webcam, stock 30
```

Affecter null à **DataSet** détache la table et conserve les lignes. Libérer le DataTable a le même effet.

JSON et CSV

GridLoadFromJSON lit un tableau d'objets ; un nom de champ à points tel que `weather.temp` lit un objet imbriqué, et **GridSaveToJSON** le réécrit sous cette forme. **GridLoadFromCSV** et **GridSaveToCSV** utilisent les guillemets de la RFC 4180, n'importe quel séparateur et une ligne d'en-tête. L'argument du champ clé nomme la clé de la ligne et, avec `aAutoColumns`, les colonnes sont créées à partir du premier objet ou de la ligne d'en-tête, avec des types déduits des valeurs. Les surcharges pour chaîne, flux et fichier lisent de l'UTF-8 en ignorant un BOM, et écrivent de l'UTF-8 sans BOM.
Le programme écrit :

```
2 rows from JSON, key of row 0: 1
id,city,weather.temp,weather.sky
1,Barcelona,21.5,Sunny
2,Oslo,8,"Rain, light"
4 columns, 2 rows from CSV, Temp is gdtFloat
[{"id":1,"city":"Barcelona","weather":{"temp":21.5,"sky":"Sunny"}},{"id":2,"city":"Oslo","weather":{"temp":8,"sky":"Rain, l
```

Les nombres CSV utilisent la culture invariante, sauf si `aInvariant` vaut false ; ce sont alors les **FormatSettings** des données qui sont utilisés. Les lignes sont écrites dans l'ordre de la vue, et `aVisibleOnly` n'écrit que les colonnes visibles.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- Différences entre .NET et Delphi
- TsgcGridSource_DataSet
- Liaison à un DataSet

- JSON et CSV

Modification sur place en .NET

Cette fonctionnalité fait partie de l'édition Registered. Les packages Community ne la contiennent pas.
Les trois grilles modifient une cellule sur place lorsque **ReadOnly** vaut false et que la colonne est **Editable**. Une zone de texte s'ouvre sur la cellule, ou une liste déroulante lorsque la colonne a une **PickList**. Une cellule booléenne bascule avec la barre d'espace ou par un clic. La modification démarre avec F2, Entrée, un caractère saisi, un double-clic ou **StartEdit**, elle est validée par Entrée, Tab ou lorsque l'éditeur perd le focus, et annulée par Échap ou **CancelEdit**.

Événements

Événement	Description
OnBeforeEdit	Avant l'ouverture de l'éditeur. Mettez `aAllow` à false pour interdire la modification.
OnValidate	Avant l'écriture de la valeur, déjà convertie au **DataType** de la colonne. Modifiez `aValue`, ou mettez `aAccept` à false pour la rejeter ; l'éditeur reste alors ouvert.
OnEdited	Après l'écriture de la valeur avec **Data.UpdateCell**, avec l'ancienne et la nouvelle valeur.
OnEditError	Le texte ne se convertit pas au **DataType** de la colonne. Modifiez `aMessage` pour définir le message.

L'argument de colonne des événements de modification est l'index dans **Columns**, et non la position visible.

Exemple

Un flux en direct n'écrase jamais un éditeur ouvert : l'éditeur suit sa ligne par la clé lorsque les lignes sont retriées et se ferme lorsque la ligne est supprimée. Sous WinForms, les éditeurs sont les contrôles enfants **TsgcWinFormsGridEdit** et **TsgcWinFormsGridCombo** ; sous WPF et Avalonia, ce sont un `TextBox` et un `ComboBox` hébergés. Lorsqu'un même projet compile les deux éditions, protégez le code de modification avec `#if !SGC_COMMUNITY`.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- Démarrage rapide WinForms
- Modification sur place
- Sélection et clavier
- Éditions

Flux en direct .NET

Cette fonctionnalité fait partie de l'édition Registered. Les packages Community ne la contiennent pas.
Le package esegece.sgcGrid.sgcWebSockets alimente une grille à partir de connexions sgcWebSockets .NET. **TsgcGridFeedSource** est la classe de base ; **TsgcGridFeedSource_WebSocket** lit un **TsgcWebSocketClient**, **TsgcGridFeedSource_MQTT** lit les publications d'un client MQTT et **TsgcGridFeedSource_SSE** lit des server-sent events. Chaque message est lu comme du JSON et associé à des lignes de la grille.

Correspondance

Propriété	Description
Grid, GridData	La cible : un contrôle de grille, trouvé par sa propriété publique **Data**, ou un **TsgcGridData**. **GridData** l'emporte.
Rows-Path	Chemin des lignes dans le message : un tableau d'objets ligne ou un seul objet ligne. Vide lorsque le message est lui-même la ligne ou le tableau.
KeyPath	Chemin de la clé de la ligne dans un objet ligne, par exemple `symbol` ou `id.value` . Une ligne sans clé est ignorée et signalée par **OnError**.
Fields	Un `FieldName=json.path` par ligne, par exemple `price=quote.last` ou `bid=book.bid[0]` . Une ligne sans `=` lit le chemin du même nom. Lorsque **Fields** est vide, chaque colonne est lue depuis le chemin de son **FieldName**. Seuls les chemins présents dans une ligne sont écrits : un message peut donc porter une mise à jour partielle.
Delete-Path	Un chemin booléen dans l'objet ligne : true supprime la ligne de la clé.
MaxRow-sPerMes-sage	Limite les lignes appliquées à partir d'un message, 0 pour aucune limite. Les lignes en excès sont signalées par **OnError**.
Active	False ignore tous les messages.
OnRow, OnError	Une ligne est sur le point d'être appliquée (mettez `aAccept` à false pour l'ignorer), ou un message n'a pas pu être appliqué.

Les nombres JSON sont écrits en double, les booléens en bool, les chaînes en string et null en null de la grille.

Sources

Classe	Propriété du client	Lit
TsgcGridFeedSource_WebSocket	**Client** (`TsgcWebSocketClient`)	Chaque message texte.
TsgcGridFeedSource_MQTT	**Client** (`TsgcWSProtocol_MQTT_Client`)	Les publications dont le topic correspond à **Topic**, avec les jokers `+` et `#` .
TsgcGridFeedSource_SSE	**Client** (`TsgcHTTP1Client`) ou **SSEClient** (`TsgcSSEClient`)	Les données de chaque server-sent event. **SSEClient** est un ajout de .NET.

Les sources s'abonnent aux événements du client à côté des gestionnaires de l'application et ne les remplacent jamais.

Exemple

Ce programme console démarre un **TsgcWebSocketServer** dans le même processus, sur l'interface de bouclage, et alimente un modèle de grille avec trois messages : un lot de lignes, une mise à jour partielle avec une suppression, et un seul objet ligne.
Le programme écrit :

```
AAPL 228.05 x 300
NVDA 121.40 x 900
JPM 212.80 x 50
```

Threads

Toutes les lignes d'un message sont appliquées dans un seul **BeginUpdate** et **EndUpdate**, dans le thread où arrive le message. **TsgcGridData** est sûr pour les threads et les grilles transfèrent leur redessin vers le thread de l'interface : aucune synchronisation n'est donc nécessaire. **OnRow** et **OnError** s'exécutent aussi dans ce thread : basculez vers le thread de l'interface avant d'y toucher un contrôle.

Voir aussi

- Vue d'ensemble de .NET
- Installation depuis NuGet
- Classes .NET
- Démos .NET
- TsgcGridFeedSource
- TsgcGridFeedSource_WebSocket
- TsgcGridFeedSource_MQTT
- TsgcGridFeedSource_SSE
- Flux en direct

Interroger la grille en .NET

Cette fonctionnalité fait partie de l'édition Registered. Les packages Community ne la contiennent pas.
TsgcGridAIQuery, du package esegece.sgcGrid.sgcHTML, transforme une question en langage naturel en filtres, tris et surlignages d'une grille. La question est traitée par le **TsgcHTMLAIAssistant** d'esegece.sgcHTML : par un modèle de langage lorsque l'assistant a un fournisseur, et par son analyseur à base de règles lorsque **Assistant** est null ou n'a pas de fournisseur. L'analyseur à base de règles n'a pas besoin de réseau et comprend des questions telles que `price greater than 100`, `sort by volume descending` et `highlight change less than 0`.

Comment la réponse est appliquée

- La réponse est vérifiée par rapport aux colonnes de la grille (**FieldName**, **Caption** et **DataType**) avant que quoi que ce soit ne soit appliqué.
- Les filtres sont combinés avec AND par **Data.OnFilterRow**. Un gestionnaire que vous avez assigné est conservé et appelé en premier, et une ligne qu'il rejette reste rejetée.
- Les tris remplacent le **SortOrder** et le **SortIndex** des colonnes.
- Les surlignages sont ajoutés aux **FormatRules** de la grille avec **ApplyToRow**, dans les couleurs du style de l'assistant.
- Le regroupement et les limites de lignes ne sont pas pris en charge par la grille ; ils sont signalés dans **LastExplanation**.
- Chaque **Ask** annule d'abord le précédent. **Reset** rétablit le gestionnaire de filtre et le tri de colonnes précédents et ne supprime que les règles ajoutées par la requête.

Ask est synchrone et doit être appelé depuis le thread de l'interface, puisqu'il modifie les colonnes et les règles de mise en forme. Avec un modèle de langage, il bloque pendant que le fournisseur répond. **OnApplied** et **OnError** signalent le résultat.

Exemple

Le **Grid** de la requête est tout objet doté d'une propriété publique **Data** ; pour les surlignages, il lui faut aussi une propriété publique **FormatRules**. Les contrôles de grille ont les deux. Ce programme console utilise une petite classe à lui et aucun assistant, si bien que l'analyseur à base de règles répond :
Le programme écrit :

```
Filter: Price > 120
Sort: Volume descending
Highlight (warning): Change < 0
NVDA 98000
AAPL 51000
MSFT 22000
JPM 9000
1 highlight rule
5 rows after Reset
```

Pour utiliser un modèle de langage, assignez à **Assistant** un **TsgcHTMLAIAssistant** doté d'un fournisseur configuré.

Voir aussi

- Vue d'ensemble de .NET
- Installation depuis NuGet
- Classes .NET
- Différences entre .NET et Delphi
- Démos .NET
- TsgcGridAIQuery
- Interroger la grille

Performances .NET

La grille ne dessine que les lignes et les colonnes visibles, et se redessine au plus **MaxFPS** fois par seconde, quelle que soit la vitesse à laquelle les données changent. L'hôte WinForms est mesuré par une porte de performance de la suite de tests, un portage du test de performance Delphi auquel ont été ajoutés le chemin de dessin de l'hôte et un flux en direct.

La scène

- Une zone cliente de `1920 x 1080` d'un **TsgcWinFormsGrid**, 12 colonnes de 160 pixels indépendants de l'appareil (texte, entier, float, currency, heure et booléen) et une hauteur de ligne de 23.
- 100 000 lignes avec clé.
- Un thread de travail envoie 1000 mises à jour par seconde, la moitié vers des lignes proches du haut de la vue.

Temps d'image

Le temps d'image est le dessin complet du contrôle, rendu et blit, sur 60 images pendant que le flux tourne. Le budget est de 16 ms par image dans un processus 64 bits, soit une image à 60 fps, et de 25 ms dans un processus 32 bits.

Runtime	Temps d'image
.NET 8, avant les masques de couverture des glyphes de sgcUICore	44 ms
.NET 8	11,4 à 13,1 ms
.NET Framework 4.7.2	7,3 à 7,7 ms

La première ligne est la mesure qui a conduit aux masques de couverture des glyphes de sgcUICore : chaque texte de cellule remplissait tout son contour via Skia.

Cadencement

La seconde phase de la porte mesure la fréquence de redessin soutenue avec **MaxFPS** à 60 sur une vraie boucle de messages pendant 3 secondes, et vérifie que le flux conserve au moins 90 pour cent de sa fréquence et que le redessin ne dépasse jamais **MaxFPS**. Les plages ci-dessous proviennent de trois exécutions de la porte.

Runtime	Fréquence de redessin
.NET 8	59,5 à 59,7 fps
.NET Framework 4.7.2	59,8 à 60 fps

Sous .NET 8, le 95e centile de l'intervalle entre deux images est de 21,4 à 22,2 ms.

Exécuter la porte

La porte est sensible au temps et nécessite une machine par ailleurs inactive : elle ne s'exécute donc que sur demande. Définissez la variable d'environnement `SGC_GRID_RUN_PERF=1` et exécutez la catégorie `Perf` des tests WinForms.

Les chiffres sont ajoutés à `sgcGridQA_perf.txt`, à côté de l'assembly de test. La porte écrit aussi, à titre de référence, les temps de **SaveToPNG** et de **Render** sans fenêtre de la même image. Le **Render** sans fenêtre de cette image prend 5,0 ms.

Voir aussi

- Vue d'ensemble de .NET
- Démarrage rapide WinForms
- Rendu sans fenêtre .NET
- Performances
- Mises à jour en direct et threads

Regroupement et résumés .NET

Les lignes peuvent être regroupées selon une ou plusieurs colonnes. Chaque groupe reçoit une ligne de groupe avec un chevron de déploiement, le texte du membre, le nombre de lignes et les résumés des colonnes ; un pied affiche les mêmes résumés sur toutes les lignes, et les colonnes figées restent visibles pendant que les autres défilent horizontalement. Le regroupement, les résumés, le pied et les colonnes figées font partie des deux éditions, sur **TsgcWinFormsGrid**, **TsgcWpfGrid** et **TsgcAvaloniaGrid**.
Le regroupement est un état de la vue. **TsgcGridData** conserve ses lignes à plat et son tri, et **IsgcUITable** continue de signaler les lignes de la vue non regroupées, si bien qu'un graphique sgcCharts lié à la grille ne voit aucun changement. La grille met les lignes en page à travers un **TsgcGridLineView**, disponible dans sa propriété **Lines**.

Propriétés de colonne

Propriété	Par défaut	Description
GroupIndex	-1	Une valeur de 0 ou plus regroupe les lignes selon la colonne à ce niveau. Les niveaux sont ordonnés par **GroupIndex**, puis par l'index de la colonne. **GroupBy**, **ClearGrouping** et le panneau de regroupement le définissent pour vous.
SummaryType	`gsuNone`	`gsuSum`, `gsuCount`, `gsuAverage`, `gsuMin` ou `gsuMax`. Le résumé est affiché sur chaque ligne de groupe et dans le pied.
SummaryFormat	vide	Le format du résumé, comme pour **DisplayFormat** : par exemple `#,##0.00`, avec du texte littéral entre guillemets doubles, `"Total: "#,##0.00`. Vide utilise **DisplayFormat** ; un comptage utilise `0`.

Modifier ces propriétés ne déclenche que le **OnChange** des colonnes et ne recharge jamais les données. Les valeurs du résumé sont `null` quand il n'y a pas de valeur, un `int` pour un comptage, un `double` pour une somme, une moyenne, un minimum ou un maximum, et un `DateTime` pour le minimum ou le maximum d'une colonne de dates.

Membres de la grille

Les mêmes membres existent dans **TsgcWinFormsGrid**, **TsgcWpfGrid** et **TsgcAvaloniaGrid**. Sous WPF et Avalonia, les sept options sont des propriétés de dépendance et des propriétés stylées, si bien qu'elles peuvent être définies en XAML, liées et stylées.

Membre	Type	Description
ShowGroupPanel, GroupPanelHeight	bool, int	Le panneau au-dessus de l'en-tête : faites-y glisser l'en-tête d'une colonne pour regrouper selon elle. Masqué par défaut, haut de 36 pixels.
GroupIndent	int	Retrait de chaque niveau de groupe, 20 par défaut.
GroupsExpanded	bool	L'état des nouveaux groupes, true par défaut.
GroupSummaries	bool	Affiche les résumés sur les lignes de groupe, true par défaut.
ShowFooter	bool	Affiche le pied avec les résumés sur toutes les lignes, false par défaut.
FrozenColumns	int	Les premières colonnes visibles qui restent visibles pendant que les autres défilent, 0 par défaut.

GroupBy, ClearGrouping	method	`GroupBy("region", "category")` regroupe selon ces champs dans cet ordre et efface tous les autres **GroupIndex** ; **ClearGrouping** affiche de nouveau les lignes à plat.
ExpandAll, CollapseAll, ExpandGroup, CollapseGroup	method	Déploient ou réduisent tous les groupes, ou un groupe selon son index.
GroupCount, GroupInfo	method	Le nombre de groupes et le **TsgcGridGroupInfo** d'un groupe.
FooterSummary, FooterSummaryText	method	La valeur du pied d'une colonne (null s'il n'y en a pas) et son texte tel que le pied l'affiche.
Lines	TsgcGridLineView	La vue en lignes de la grille : lignes de groupe, lignes de données et, dans l'édition Registered, lignes d'arbre et bandes de détail.
FocusedLine	int	La ligne ayant le focus. Sur une ligne de groupe, **SelectedRow** vaut -1 et **FocusedLine** indique la ligne.
OnGroupingChange	EventHandler	Les colonnes regroupées ou leur ordre ont changé, depuis le code ou depuis le panneau de regroupement.
OnGroupExpandChange	TsgcGridGroupExpandEvent	Un groupe a été déployé ou réduit. `aGroup` vaut -1 après **ExpandAll** et **CollapseAll**.
OnGetGroupText	TsgcGridGetGroupTextEvent	Modifie le libellé d'une ligne de groupe avant qu'elle soit dessinée.

Les index de ligne restent des lignes de la vue : **SelectedRow**, **OnCellClick** et **Data** utilisent les mêmes lignes que sans regroupement. **TopRow** compte des lignes d'affichage, qui sont les lignes de la vue quand rien n'est regroupé.

Informations de groupe

Field	Description
Level	0 pour le niveau le plus externe.
Parent	Le groupe parent, -1 au niveau 0.
Column	L'index dans **Columns** de la colonne regroupée.
RowCount	Les lignes de la vue du groupe, ses groupes enfants compris.
FirstRow	La ligne de la vue de la première ligne du groupe.
Text, Value, Blank	Le texte du membre, sa valeur typée, et true pour le membre vide, affiché comme `(blank)`.
Expanded	Le groupe est déployé.
Line	La ligne de la ligne de groupe, -1 dans un groupe réduit.

Path	Les textes des membres depuis le niveau 0.

Souris et clavier

- Faites glisser l'en-tête d'une colonne vers le panneau de regroupement pour regrouper selon elle ; un repère montre où elle sera déposée. Faites glisser une puce le long du panneau pour changer l'ordre, faites-la glisser hors du panneau pour la retirer, cliquez dessus pour inverser le tri du groupe. Échap annule un glissement.
- Cliquez sur le chevron d'une ligne de groupe, ou double-cliquez sur la ligne, pour la déployer ou la réduire.
- Sur une ligne de groupe, Right et Left déploient et réduisent, + et la touche moins font de même, * déploie tous les groupes et Entrée bascule le groupe. Ctrl+C sur une ligne de groupe copie son libellé.
- Sous Avalonia, un contact tactile fait toujours défiler la grille ; un appui sur un chevron agit comme un clic.

Exemple

Un formulaire WinForms qui regroupe des commandes par région et par catégorie, avec une somme et une moyenne sur chaque ligne de groupe et dans le pied, deux colonnes figées et un libellé personnalisé pour les groupes de région :
Les données peuvent être écrites depuis n'importe quel thread pendant qu'elles sont regroupées : la grille resynchronise les groupes et les résumés dans le thread de l'interface et cadence le travail, si bien que l'interface reste réactive.

Regroupement sans contrôle

TsgcGridLineView fonctionne avec un **TsgcGridData** seul, par exemple dans un service ou un programme console. **GroupSummary** renvoie la valeur du résumé d'un groupe et d'une colonne, et **SetGroupExpanded** déploie ou réduit un groupe :
Sortie :

```
2 levels, 6 groups, 11 lines
North: 3 rows, total 142.5
  Books: 2 rows, total 82.5
  Games: 1 rows, total 60
South: 2 rows, total 140.9
  Books: 1 rows, total 41
  Games: 1 rows, total 99.9
6 lines with North collapsed
```

Des données et une vue en lignes sont aussi ce qu'écrit **TsgcGridXLSXWriter**, voir Export Excel .NET.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- Lignes d'arbre et maître-détail .NET
- Export Excel .NET
- Démos .NET
- TsgcGridColumn
- Regroupement
- Résumés
- Colonnes figées
- TsgcGridLineView

Lignes d'arbre et maître-détail .NET

Cette fonctionnalité fait partie de l'édition Registered. Les packages Community ne la contiennent pas. **TsgcWinFormsGrid**, **TsgcWpfGrid** et **TsgcAvaloniaGrid** affichent les lignes sous forme d'arbre, construit à partir d'un champ clé et d'un champ parent, et ouvrent une bande de détail sous une ligne maître avec ses lignes enfants issues d'une seconde source. Le mode arbre et le regroupement ne sont pas actifs en même temps : en mode arbre, le regroupement est ignoré, le **GroupIndex** des colonnes est conservé et le panneau de regroupement est masqué.

Lignes d'arbre

La propriété **Tree** est un **TsgcGridTreeOptions**. L'arbre est actif quand **KeyField** et **ParentField** sont tous deux définis.

Propriété	Par défaut	Description
KeyField	vide	La colonne qui contient la clé de la ligne.
ParentField	vide	La colonne qui contient la clé de la ligne parent. Les lignes dont la clé parent est vide, inconnue ou égale à la leur sont des racines.
TreeField	vide	La colonne qui affiche le retrait et le bouton de déploiement. Vide utilise la première colonne visible.
Indent	16	Retrait de chaque niveau.
Expanded	false	L'état des nouveaux nœuds.

Les champs doivent être des colonnes de la grille ; elles peuvent être masquées. Les clés se comparent comme du texte : les entiers et les flottants sans partie fractionnaire utilisent leur texte entier, si bien qu'une clé `int` et un parent `double` correspondent. Quand des lignes partagent une clé, la première ligne possède les enfants, et un cycle est rompu au premier nœud répété. Les frères gardent l'ordre de la vue, si bien que trier trie les frères, et avec un filtre une ligne visible se rattache à son plus proche ancêtre visible. L'état déployé est conservé par clé, il survit donc à un rechargement.

Membre	Description
ExpandNode, CollapseNode	Déploient ou réduisent le nœud d'une ligne de la vue.
ExpandAll, CollapseAll	En mode arbre, ils déploient ou réduisent tous les nœuds.
OnTreeExpandChange	Un nœud a été déployé ou réduit ; `aRow` vaut -1 après **ExpandAll** et **CollapseAll**.

Cliquez sur le bouton d'un nœud pour le déployer ou le réduire. Sur un nœud, Right le déploie ou passe à son premier enfant, et Left le réduit ou passe à son parent.

Maître-détail

La propriété **Detail** est un **TsgcGridDetailOptions**. Une colonne d'expansion ouvre une bande sous une ligne maître ; la bande est une grille imbriquée avec son propre en-tête, sa propre position de défilement et sa propre sélection. Les détails s'imbriquent sur un seul niveau et sont en lecture seule.

Propriété	Par défaut	Description
Active	false	Affiche la colonne d'expansion.
Source	null	Les lignes de détail : un **TsgcGridDataStore**, un **TsgcGridData**, ou tout objet ayant une propriété publique **Data** de type **TsgcGridData**, comme une autre grille.
MasterKeyField	vide	La colonne maître dont la valeur est recherchée dans les lignes de détail.
DetailKeyField	vide	La colonne de la source qui contient la clé maître.
Lines	8	Hauteur d'une bande en lignes de grille, en-tête imbriqué compris.
Columns	vide	Les colonnes des bandes. Vide utilise une copie des colonnes de la source.

Membre	Description
ExpandDetail, CollapseDetail, DetailExpanded	Ouvrent, ferment et testent la bande d'une ligne de la vue.
DetailData	Le **TsgcGridData** d'une bande ouverte, null quand la bande est fermée.
OnDetailExpandChange	Une bande a été ouverte ou fermée.
OnDetailCellClick	Une cellule d'une bande a été cliquée : ligne maître, ligne de détail et colonne de détail.
OnGetDetailData	Déclenché avant le remplissage d'une bande. Écrivez les lignes dans `aData` et définissez `aHandled` à true pour remplir la bande depuis le code ; sinon la bande est remplie depuis **Source**.

L'état déployé est conservé par clé maître. Quand les lignes de la source changent, chaque bande ouverte est remplie de nouveau dans le thread de l'interface, sur place. Un clic sur un en-tête d'une bande la trie, la molette de la souris au-dessus d'une bande la fait défiler, et Down passe au-delà d'une bande. L'image de la bande est dessinée par esegece.sgcGrid.Rendering, que chaque package de grille référence.
Quand **Source** est un composant, sa libération efface **Source**. Pour tout autre objet, définissez **Source** à null avant que l'objet disparaisse.

TsgcGridDataStore

TsgcGridDataStore est un composant non visuel qui contient des lignes de détail : **Columns**, **Data** et **IsgcUITable**. Remplissez-le depuis le code, ou définissez-le comme **Grid** d'un **TsgcGridSource_DataSet** pour y charger un `DataTable`. Définissez **Active** à true après **Grid** et **DataSet**, et **LiveUpdates** garde le stockage synchronisé, voir Liaison DataTable, JSON et CSV en .NET.

Exemple

Un organigramme sous forme d'arbre, et des clients avec une bande de leurs commandes issues d'un **TsgcGridDataStore** :
La deuxième classe remplit les bandes uniquement depuis le code : sans **Source**, les bandes utilisent les **Columns** des options, et chaque ligne vient de **OnGetDetailData**.

Sans contrôle

TsgcGridLineView accepte les mêmes options : affectez un **TsgcGridTreeOptions** à **TreeOptions** et un **TsgcGridDetailOptions** à **DetailOptions** ; **SetNodeExpanded** et **SetDetailExpanded** changent l'état. L'export Excel s'en sert pour écrire un arbre sans contrôle, voir Export Excel .NET.

Voir aussi

- Classes .NET
- Regroupement et résumés .NET
- Export Excel .NET
- Liaison DataTable, JSON et CSV en .NET
- Démos .NET
- TsgcGridSource_DataSet
- Lignes d'arbre
- Maître-détail
- TsgcGridDataStore

Export Excel .NET

Cette fonctionnalité fait partie de l'édition Registered. Les packages Community ne la contiennent pas. **TsgcWinFormsGrid**, **TsgcWpfGrid** et **TsgcAvaloniaGrid** enregistrent leur vue sous forme de classeur Excel (.xlsx) sans installation d'Excel et sans bibliothèque externe. Le fichier est écrit par **TsgcGridXLSXWriter** du package esegece.sgcGrid, qui fonctionne aussi sans contrôle.

Ce qui est écrit

- Les colonnes visibles dans leur ordre, les libellés dans une ligne d'en-tête en gras, et les largeurs de colonne.
- Les nombres, dates et booléens sous forme de cellules typées, avec le format numérique Excel du **DisplayFormat** de la colonne ; le texte sous forme de chaînes partagées.
- Les groupes sous forme de lignes de groupe avec leur libellé et des formules `SUBTOTAL` pour les résumés des colonnes, au-dessus de leurs lignes et dans des niveaux de plan Excel. Les groupes réduits sont écrits comme des lignes masquées, si bien que le classeur s'ouvre tel que la grille l'affiche.
- Quand une colonne a un résumé, une ligne de pied sous les données avec `SUBTOTAL` sur toute la plage de données.
- Les lignes d'arbre en retrait dans la colonne d'arbre et dans des niveaux de plan ; les lignes des nœuds réduits sont des lignes masquées.
- Les colonnes figées et la ligne d'en-tête sous forme de volet figé.

Les bandes de détail ne sont pas écrites. Le tri et le filtrage s'appliquent : le classeur contient les lignes de la vue.

SaveToXLSX

`SaveToXLSX(Stream aStream, string aSheetName = "Sheet1")` et `SaveToXLSX(string aFileName, string aSheetName = "Sheet1")` écrivent la vue avec les largeurs de colonne à l'écran et **FrozenColumns**. Appelez-les dans le thread de l'interface.

TsgcGridXLSXWriter

Membre	Description
SheetName	Nom de la feuille de calcul, `Sheet1` par défaut. Les caractères qu'Excel refuse deviennent `_` et le nom est coupé à 31 caractères.
FrozenColumns	Les premières colonnes visibles qui restent visibles, 0 par défaut.
Summaries	Écrit les lignes `SUBTOTAL` de groupe et de pied, true par défaut.
FileTime	Date et heure des entrées du fichier, l'heure de création par défaut.
SetColumnWidth	`SetColumnWidth(aCol, aDIPs)` : largeur d'une colonne (index dans **Columns**) en pixels indépendants de l'appareil, utilisée à la place de sa **Width**. Une valeur négative rétablit **Width**.
SaveToStream, SaveToFile	`SaveToStream(aData, aLines, aStream)` et `SaveToFile(aData, aLines, aFileName)` écrivent le classeur de `aData`. `aLines` fournit le regroupement, l'arbre et l'état déployé ; null utilise une vue en lignes propre, construite à partir du **GroupIndex** des colonnes.

Les erreurs lèvent **EsgcGridXLSXError**. `sgcGrid_XLSX.sgcGridExcelNumberFormat` renvoie le format numérique Excel utilisé pour un type de données et un **DisplayFormat**.

Exemple sans contrôle

Un programme console qui écrit un classeur regroupé avec le premier groupe réduit, et un organigramme sous forme d'arbre. Il utilise la méthode `CreateSales` de l'exemple de Regroupement et résumés .NET :
100 000 lignes sont écrites en environ 1,4 seconde sous .NET 8.

Voir aussi

- Regroupement et résumés .NET
- Lignes d'arbre et maître-détail .NET
- Classes .NET
- Différences entre .NET et Delphi
- Démos .NET
- Export Excel

Démos .NET

Les démos se trouvent dans `.net\demos`, dans la solution `sgcGrid_Demos.sln`. Chaque démo possède un README.md et une option `--test` qui exécute un auto-contrôle hors écran et se termine avec le code 0 en cas de succès, comme l'option `/test` des démos Delphi.

Démo	Hôte	Montre
LiveTicker	WinForms	1000 symboles mis à jour depuis un thread de travail, avec clignotement des modifications, couleurs de hausse et de baisse et **TailMode**.
LiveTickerWpf	WPF	Le même ticker sur **TsgcWpfGrid**.
LiveTickerAvalonia	Avalonia	Le même ticker sur **TsgcAvaloniaGrid**.
DataSetGrid	WinForms	Un DataTable lié par **TsgcGridSource_DataSet**, avec import et export JSON et CSV.
Editing	WinForms	Modification sur place avec **OnValidate**. Édition Registered uniquement.
GridAndChart	WinForms	Les lignes et la sélection de la grille pilotent un graphique sgcCharts par **TsgcChartDataSource_Table**, dans des tuiles sgcDashboard.
AskYourGrid	WinForms	**TsgcGridAIQuery** avec l'analyseur à base de règles. Édition Registered uniquement.
LiveFeed	WinForms	**TsgcGridFeedSource_WebSocket** alimenté par un **TsgcWebSocketServer** du même processus. Édition Registered uniquement.
Grouping	WinForms	100 000 commandes regroupées par région et par catégorie, avec le panneau de regroupement, des résumés de groupe et de pied, deux colonnes figées et des mises à jour de prix en direct depuis un thread de travail.
GroupingWpf	WPF	La même démo de regroupement sur **TsgcWpfGrid**.
GroupingAvalonia	Avalonia	La même démo de regroupement sur **TsgcAvaloniaGrid**.
TreeAndDetail	WinForms	Un organigramme sous forme de lignes d'arbre, et des clients issus d'un DataTable avec une bande de détail de leurs commandes issues d'un **TsgcGridDataStore**. Édition Registered uniquement.
ExcelExport	Console	Écrit un classeur de ventes regroupées et un organigramme sous forme d'arbre avec **TsgcGridXLSXWriter**, sans contrôle. Fonctionne sous Windows et Linux. Édition Registered uniquement.

Compiler les démos

Le zip des démos utilise par défaut les packages NuGet de l'édition Community. Compilez les démos avec les packages Registered en définissant **SgcGridEdition** :

```
dotnet build sgcGrid_Demos.sln -p:SgcGridEdition=Registered
```

Dans l'édition Community, les démos réservées à Registered se compilent toujours et expliquent la limite lorsqu'elles s'exécutent.

Voir aussi

- Vue d'ensemble de .NET
- Installation depuis NuGet
- Démarrage rapide WinForms
- Flux en direct .NET
- Interroger la grille en .NET
- Regroupement et résumés .NET
- Lignes d'arbre et maître-détail .NET
- Export Excel .NET
- Démos Delphi

Dépannage

Undeclared identifier StartEdit, OnValidate or OnEdited

L'édition Community se compile sans SGC_GRID_EDIT, de sorte que les membres de modification des grilles n'existent pas et que le code qui les utilise échoue avec E2003 Undeclared identifier. Une fiche enregistrée par un build Registered avec des événements de modification déclenche une exception Error reading au chargement. Supprimez le code de modification, ou utilisez un build Registered ou Trial. Voir Éditions.

File not found sgcGridWS_Source ou sgcWebSocket

Les sources de flux et la requête IA se trouvent dans le pont sgcGridWS. L'édition Community ne le livre pas. Dans les autres éditions, le pont nécessite sgcWebSockets : ajoutez ses sources au chemin de recherche, ou installez le package sgcWebSockets avant le package sgcGridWS de la même version. Voir Plateformes.

Le pont ne se compile pas sous Delphi 7

Sous Delphi 7, les unités sgcGridWS se compilent avec sgcWebSockets, qui nécessite la version d'Indy pour laquelle il a été construit. Installez la version d'Indy attendue par votre build sgcWebSockets avant de construire le pont.

La démo C++Builder DataSetGrid échoue sous Win64

Win32 lie MIDAS statiquement, mais C++Builder n'a pas de bibliothèque MIDAS statique pour Win64. Copiez midas.dll à côté de l'exécutable 64 bits.

Les valeurs ne changent pas ou des lignes sont dupliquées

UpdateRow trouve les lignes par clé et les clés sont sensibles à la casse. Utilisez la même clé pour AppendRow et UpdateRow. Avec AutoInsert à False, la mise à jour d'une clé absente est ignorée. Les noms de champs qui ne correspondent à aucune colonne sont également ignorés.

OnFilterRow lit la mauvaise ligne

OnFilterRow reçoit une ligne du magasin. Lisez-la avec StoreCells ou StoreKeyOf, pas avec Cells, qui prend des lignes de vue. Voir Filtrage.

La grille se fige lorsqu'un gestionnaire de flux touche un contrôle

OnFilterRow et OnGetValue s'exécutent avec le verrou des données détenu, et OnChange, OnRow et OnError des flux s'exécutent dans le thread d'écriture. N'attendez jamais le thread principal à cet endroit et ne touchez jamais un contrôle ; utilisez plutôt TThread.Queue.

Les nombres affichent une virgule ou un point de façon inattendue

Les valeurs sont formatées avec Data.FormatSettings, qui suit les paramètres régionaux. Appelez Data.UseInvariantFormat pour un séparateur décimal point, ou définissez FormatSettings.

Identificateur non déclaré Tree, Detail ou SaveToXLSX

Les lignes d'arbre, les lignes maître-détail, TsgcGridDataStore et l'export Excel ne sont compilés qu'avec SGC_GRID_ADVANCED, que l'édition Community laisse de côté. Le code qui les utilise échoue avec E2003 Undeclared identifier, et une fiche enregistrée avec des propriétés Tree ou Detail déclenche une exception Error reading au chargement. Voir Éditions.

L'arbre affiche une liste à plat

Tree.KeyField et Tree.ParentField doivent tous deux nommer des colonnes de la grille (elles peuvent être masquées), et les données ne doivent pas être virtuelles. Les valeurs de parent sont comparées aux textes de clé en tenant compte de la casse ; une ligne dont le parent est introuvable devient une racine. Voir Lignes d'arbre.

Le regroupement n'a aucun effet

Le regroupement est ignoré tant que le mode arbre est actif. Videz Tree.KeyField ou Tree.ParentField pour regrouper de nouveau.

Aucun expanseur de détail n'est affiché

Une ligne n'obtient un expanseur que lorsque Detail.Active vaut True, que Detail.MasterKeyField nomme une colonne de la grille, qu'une Detail.Source ou un gestionnaire OnGetDetailData est défini et que la clé maître de la ligne n'est pas vide. Des données virtuelles n'affichent pas de bandes. Voir Maître-détail.

Une bande de détail reste vide

Detail.DetailKeyField doit nommer une colonne de la source, et ses valeurs doivent correspondre aux textes de la clé maître (en tenant compte de la casse ; 5, 5.0 et '5' sont la même clé). Une source virtuelle ne donne aucune ligne. Avec OnGetDetailData et sans Source, définissez Detail.Columns, car la bande y prend ses colonnes.

Les lignes de détail manquent dans le fichier Excel

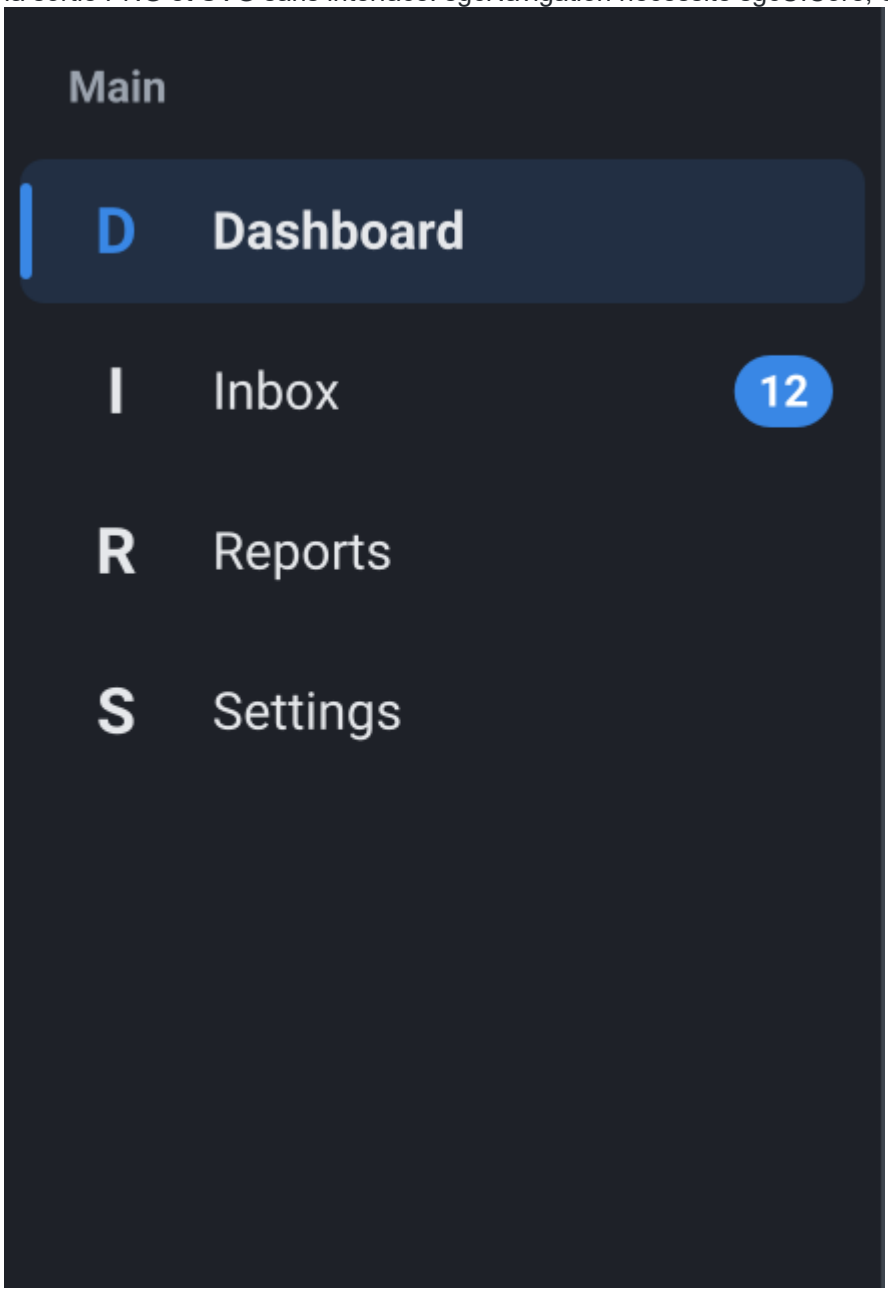
SaveToXLSX n'écrit pas les bandes de détail. Exportez la source de détail à part, par exemple avec un TsgcGridXLSXWriter sur le Data du TsgcGridDataStore. Voir Export Excel.

SaveToXLSX déclenche EsgcGridXLSXError

Une feuille XLSX contient au plus 1 048 576 lignes et 16 384 colonnes, lignes de groupe et en-tête compris. Filtrez les lignes ou masquez des colonnes avant l'export. Rien n'est écrit lorsque l'exception est déclenchée.

Introduction

sgcNavigation est une barre de navigation native pour Delphi et C++Builder, pour VCL et FireMonkey. Un seul composant, **TsgcVCLNavigationBar** ou **TsgcFMXNavigationBar**, présente les pages d'une application sous la forme d'une liste verticale d'éléments avec des icônes, des libellés, des badges et des en-têtes de groupe. En mode sidebar, c'est une liste plate qui se réduit en un rail d'icônes avec une animation. En mode accordéon, les en-têtes de groupe déplient et replient les éléments situés en dessous.
sgcNavigation fait partie de la suite visuelle eSeGeCe et sert de cadre aux autres produits : une barre de navigation à gauche et, à droite, un dashboard **sgcDashboard**, une grille **sgcGrid** ou une page avec un **sgcToolbar**. Elle est dessinée avec **sgcUICore**, le package d'exécution partagé qui fournit le canvas, le rastériseur, les polices, l'écrivain SVG et les thèmes clair et sombre. La barre a donc le même aspect sous VCL, sous FireMonkey et dans la sortie PNG et SVG sans interface. sgcNavigation nécessite sgcUICore, qui s'installe avec lui.



Éditions

sgcNavigation est proposé en éditions Registered, Trial et Community. L'édition Community offre toutes les fonctionnalités, sauf les liens d'action (Action links) des éléments. Voir Éditions.

Par où commencer

- Architecture : le modèle indépendant du framework, le moteur de rendu sur le canvas sgcUICore et les contrôles VCL et FMX.
- Plateformes : les versions de Delphi, C++Builder et les plateformes.
- Démarrage rapide VCL, Démarrage rapide FMX et Démarrage rapide C++Builder : une première barre de navigation en quelques minutes.
- Composants : les barres de navigation, les éléments et le modèle.
- Mode sidebar, Mode accordéon, Réduction et animation, Images et glyphes, Badges, Clavier et focus, Liens d'action, Thèmes et Rendu sans interface.
- Démos : les projets de démonstration Delphi et C++Builder.

Architecture

sgcNavigation est construit en couches. Chaque couche ne dépend que de celles situées en dessous, de sorte que le modèle et le moteur de rendu se compilent sans VCL ni FireMonkey.

Couche	Unité	Rôle
Modèle	sgcNavigation_Model	TsgcNavigationModel contient les Items (une collection de TsgcNavigationItem), le mode, l'état réduit, les tailles, le thème et l'état d'interaction (élément sélectionné, survolé, enfoncé et ayant le focus). Sa méthode Layout calcule la ligne de chaque élément affiché et HitTest trouve l'élément situé sous un point.
Moteur de rendu	sgcNavigation_Render	sgcNavigationRender dessine un modèle sur n'importe quel canvas sgcUICore. sgcNavigationSaveToPNG et sgcNavigationSaveToSVG l'utilisent pour écrire une image sans fenêtre. Voir Rendu sans interface.
Contrôle VCL	sgcNavigation_VCL	TsgcVCLNavigationBar possède un modèle, publie ses propriétés et le dessine grâce à la classe de base de contrôle VCL de sgcUICore. Il ajoute la souris, le clavier, les info-bulles, la liste d'images et l'animation de réduction.
Contrôle FMX	sgcNavigation_FMX	TsgcFMXNavigationBar est le contrôle FireMonkey, avec les mêmes propriétés, méthodes et événements, plus la saisie tactile.
sgcUICore	sgcUICore_Canvas, sgcUICore_VCL, sgcUICore_FMX, sgcUICore_Theme	Le runtime partagé : le canvas vectoriel et le rastériseur, la classe de base de contrôle VCL et FMX avec régulation des images (MaxFPS), et les jetons des thèmes clair et sombre.

Modèle et contrôle

Les propriétés publiées d'une barre de navigation sont celles de son modèle, et la propriété **Model** de la barre donne accès au reste : ToggleGroup, HitTest, NextVisible et l'état d'interaction. Une modification du modèle planifie une nouvelle image et ne dessine jamais directement, si bien que de nombreuses modifications consécutives ne coûtent qu'un seul rafraîchissement.

Dessin

Chaque ligne est dessinée par le moteur de rendu : l'emplacement de l'icône avec une image ou un glyphe, le libellé ajusté à la largeur disponible, le badge, ainsi que les états survolé, enfoncé, sélectionné, avec focus et désactivé. Toutes les tailles sont des pixels logiques multipliés par l'échelle du moniteur, ce qui rend la barre nette quel que soit le DPI. La barre VCL utilise PPI / 96 comme échelle et la barre FMX l'échelle de la fiche.

Voir aussi

- Plateformes
- TsgcNavigationModel
- Thèmes

Éditions

sgcNavigation est compilé à partir des mêmes sources pour chaque édition. L'édition ne change que les fonctionnalités incluses dans une compilation.

Édition	Contenu
Registered	Tout : les barres de navigation VCL et FMX, les modes sidebar et accordéon, le rail d'icônes réduit avec son animation, les images, les glyphes et les badges, la navigation au clavier, les thèmes, les liens d'action (Action links) des éléments et la sortie PNG et SVG sans interface.
Trial	Les mêmes fonctionnalités que Registered. Une compilation avec SGC_TRIAL affiche un avis indiquant qu'il s'agit d'une version d'essai une fois la fenêtre de publication indiquée dans la compilation dépassée. Rien d'autre ne change.
Community	Tout, sauf les liens d'action (Action links) des éléments.

Ce que Community ne contient pas

L'édition Community se compile avec SGC_NAVIGATION_ACTIONS non défini. Dans une compilation Community, la propriété **Action** de TsgcNavigationItem n'existe pas, un élément ne peut donc pas être lié à un TAction. Affectez plutôt l'événement **OnClick** de l'élément, ou l'événement **OnItemClick** de la barre.
L'édition est un choix fait à la compilation dans sgcVer.inc (SGC_COMMUNITY, qui désactive aussi SGC_TRIAL), et non un interrupteur d'exécution. Le code ou le fichier de fiche qui utilise Action ne se compile pas et ne se charge pas avec une compilation Community, voir Dépannage.

Voir aussi

- Liens d'action
- Plateformes

Installation

sgcNavigation fait partie de la distribution sgcUI. Un seul installateur couvre les produits sgcUI (sgcUICore, sgcCharts, sgcPDF, sgcReports, sgcMaps, sgcDashboard, sgcGrid, sgcNavigation, sgcToolbar et sgcWizard) et n'installe que les produits inclus dans votre licence.

Installation avec l'installateur sgcUI

1. Lancez l'installateur sgcUI et connectez-vous avec le nom d'utilisateur et le mot de passe de votre compte eSeGeCe.
2. Choisissez les versions de Delphi et de C++Builder dans lesquelles installer.
3. Sur la page des composants, sélectionnez **sgcNavigation** et tout autre produit souhaité. Le serveur de licences renvoie les produits que vous possédez, et l'installateur affiche une case à cocher par produit. Les produits que votre licence n'inclut pas sont grisés. Une licence All-Access déverrouille tous les produits.
4. L'installateur copie les fichiers, puis compile les packages sur votre machine pour chaque IDE sélectionné, dans l'ordre des dépendances, et installe les packages de conception.

Ce qui est fourni avec sgcNavigation

sgcUICore, la base commune des produits de mise en page et de sgcCharts, est fourni avec sgcNavigation. Il n'a pas de case à cocher propre. L'installateur l'installe automatiquement dès que sgcNavigation est sélectionné. sgcNavigation n'a pas de package de passerelle sgcWebSockets.

Dossiers installés

Tous les produits s'installent dans un seul dossier, par défaut `Documents\esegece\sgcUI` (`sgcUI_Trial` pour la version d'essai et `sgcUI_Community` pour l'édition Community).

Dossier	Contenu
`Source`	Le code source de chaque produit installé. C'est l'unique chemin de bibliothèque de sgcUI.
`Packages`	Les packages de chaque produit installé, un jeu par produit et par version d'IDE.
`Lib<V>`	Les unités compilées pour chaque version d'IDE et chaque plateforme, par exemple `LibD13` .
`Demos\Navigation`	Les démos de sgcNavigation. Voir Démos.

L'installateur ajoute `sgcUI\Source` au chemin de bibliothèque et au chemin de navigation de chaque IDE et plateforme sélectionnés, ainsi que le dossier des unités compilées. sgcUICore se trouve dans le même dossier `Source`, il n'a donc besoin d'aucun chemin propre.

Trial et Community

- **Trial** : l'installateur Trial de sgcUI installe tous les produits en mode d'essai, y compris sgcNavigation.
- **Community** : l'installateur Community gratuit de sgcUI inclut sgcNavigation. Ses produits sont sgcCharts, sgcDashboard, sgcGrid, sgcNavigation, sgcToolbar et sgcWizard, tandis que sgcPDF et sgcMaps ne font pas partie de Community. Voir Éditions pour ce que la version Community de sgcNavigation laisse de côté.

Installations autonomes précédentes

Si l'installateur trouve une ancienne installation autonome de sgcUICore, comme celle créée par l'installateur autonome de sgcCharts, il liste les chemins de bibliothèque et les packages de conception de cette installation sur la page **Previous installations**. Quand vous continuez, il les supprime de l'IDE avant de compiler les nouveaux packages, de sorte qu'il n'y ait jamais deux copies d'une unité dans le chemin.

Compiler les packages manuellement

Pour compiler vous-même les packages, ouvrez-les pour votre version d'IDE depuis `sgcUI\Packages`. `<V>` est le suffixe de version Delphi, par exemple `sgcNavigationD13.dpk`.

Package	Contenu
`sgcUICore<V>`	La base commune sgcUICore (exécution).
`sgcNavigation<V>` / `dclsgcNavigation<V>`	Les composants sgcNavigation (exécution / conception).

Compilez-les dans l'ordre des dépendances : `sgcUICore`, puis `sgcNavigation`, puis `dclsgcNavigation`. Installez ensuite le package de conception `dclsgcNavigation` et ajoutez `sgcUI\Source` au chemin de bibliothèque de votre version de Delphi (Tools > Options > Language > Delphi Options > Library > Library path). Pour C++Builder, ouvrez le projet `.cbproj` correspondant à votre version, par exemple `sgcNavigationC13.cbproj`, et compilez-le. Les en-têtes `.hpp` générés apparaissent à côté du package compilé.

.NET

Rien ne change pour .NET. sgcNavigation reste distribué sous forme de ses propres packages NuGet (`esegece.sgcNavigation` et ses packages compagnons). Voir l'installation .NET.

Plateformes

Cible	Prise en charge
Delphi	Delphi 7, 2007, 2009, 2010, XE à XE8, 10 à 10.4, 11, 12 et 13.
C++Builder	C++Builder 2007, 2010, XE à XE8, 10 à 10.4, 11, 12 et 13. Les packages génèrent les en-têtes .hpp et les fichiers .lib qu'un projet C++Builder inclut et lie.
VCL	TsgcVCLNavigationBar sous Windows, Win32 et Win64, toutes les versions de Delphi.
FireMonkey	TsgcFMXNavigationBar à partir de Delphi XE6 (SGC_NAVIGATION_FMX), sous Win32, Win64, macOS, iOS et Android, et sous Linux avec FMX for Linux (FMXLinux). La propriété Images nécessite XE8 ou une version ultérieure (FMX.ImgList), et les info-bulles des éléments nécessitent 10.1 Berlin ou une version ultérieure.
Linux64	Les unités indépendantes du framework, le modèle et le moteur de rendu avec la sortie PNG et SVG sans interface, à partir de Delphi 10.2 (SGC_NAVIGATION_LINUX).
.NET	Un portage C# avec la même API pour WinForms, WPF et Avalonia, et la sortie headless en PNG et SVG, sous .NET Framework 4.7.2 et .NET 8 et 9, voir Vue d'ensemble de .NET.

Packages

sgcNavigation fournit un package d'exécution indépendant du framework et un package d'exécution par framework, de sorte qu'une application FireMonkey pour Android, iOS ou macOS ne lie jamais la VCL. Chaque nom de package se termine par le suffixe de l'IDE, par exemple sgcNavigationD13.dpk pour Delphi 13 et sgcNavigationC13.cbproj pour C++Builder 13.

Package	Type	Contenu	Requiert
sgcNavigation	Exécution, indépendant du framework	Chaque unité du produit qui n'utilise ni la VCL ni FireMonkey.	rtl, sgcUICore
sgcNavigationVCL	Exécution, VCL	sgcNavigation_VCL (TsgcVCLNavigationBar)	rtl, vcl, sgcUICore, sgcUICoreVCL, sgcNavigation
sgcNavigationFMX	Exécution, FireMonkey, à partir de XE6	sgcNavigation_FMX (TsgcFMXNavigationBar)	rtl, fmx, sgcUICore, sgcUICoreFMX, sgcNavigation
dclsgcNavigation	Conception, VCL	Enregistre TsgcVCLNavigationBar dans la page de palette **SGC Navigation**.	designide, rtl, vcl, sgcNavigation, sgcNavigationVCL
dclsgcNavigationFMX	Conception, FireMonkey, à partir de XE6	Enregistre TsgcFMXNavigationBar dans la page de palette **SGC Navigation**.	designide, rtl, fmx, sgcNavigation, sgcNavigationFMX, dclsgcNavigation

sgcUICore suit la même répartition : sgcUICore, sgcUICoreVCL et sgcUICoreFMX, sans package de conception. dclsgcNavigationFMX requiert dclsgcNavigation, installez donc d'abord dclsgcNavigation.

Versions de l'IDE et plateformes cibles

Delphi	sgcNavigation	sgcNavigationVCL	sgcNavigationFMX	dclsgcNavigation, dclsgcNavigationFMX
7, 2007, 2009, 2010, XE	Win32	Win32	Aucune	Win32
XE2 à XE5	Win32, Win64	Win32, Win64	Aucune	Win32
XE6 à 10.2	Win32, Win64	Win32, Win64	Win32, Win64	Win32
10.3	Win32, Win64, Linux64, OSX64	Win32, Win64	Win32, Win64, OSX64	Win32
10.4	Win32, Win64, Linux64, OSX64, Android, Android64, iOSDevice64	Win32, Win64	Win32, Win64, OSX64, Android, Android64, iOSDevice64	Win32
11	Win32, Win64, Linux64, OSX64, OSXARM64, Android, Android64, iOSDevice64, iOSSimARM64	Win32, Win64	Win32, Win64, OSX64, OSXARM64, Android, Android64, iOSDevice64, iOSSimARM64	Win32
12, 13	Comme 11	Win32, Win64	Comme 11	Win32, Win64

De XE6 à 10.2, le package FireMonkey est compilé pour Windows uniquement. Les packages de conception sont chargés par l'IDE, ils sont donc compilés pour la plateforme de l'IDE uniquement.

C++Builder	sgcNavigation, sgcNavigationVCL	sgcNavigationFMX	dclsgcNavigation, dclsgcNavigationFMX
2007, 2010, XE, XE2	Win32	Aucune	Win32 (dclsgcNavigation uniquement)
XE3 à XE5	Win32, Win64	Aucune	Win32 (dclsgcNavigation uniquement)
XE6 à 12	Win32, Win64	Win32, Win64	Win32
13	Win32, Win64, Win64x	Win32, Win64, Win64x	Win32, Win64x

Les projets C++Builder portent les mêmes noms avec le suffixe C, par exemple sgcNavigationC13.cbproj, sgcNavigationVCLC13.cbproj, sgcNavigationFMXC13.cbproj, dclsgcNavigationC13.cbproj et dclsgcNavigationFMXC13.cbproj. Les packages C++Builder sont compilés pour Windows uniquement. Les notes C++Builder qui s'appliquent à chaque produit sgcUI (packages d'exécution sous Win64, C++Builder 10.3 sous Win64, noms de fichiers des packages Win64x) se trouvent dans le sujet Plateformes et packages de l'aide de la suite sgcUI.

FireMonkey sur Android, iOS et macOS

Les contrôles FireMonkey (TsgcFMXNavigationBar) ciblent macOS à partir de Delphi 10.3, Android et iOS à partir de Delphi 10.4, et macOS ARM et le simulateur iOS à partir de Delphi 11. Dans un projet FireMonkey pour ces plateformes :

- Utilisez les contrôles FireMonkey. Les contrôles VCL et sgcNavigationVCL ne fonctionnent que sous Windows.
- Déposez le contrôle sur un formulaire, ou ajoutez son unité à la clause uses : sgcNavigation_FMX.
- Sélectionnez la plateforme cible dans la fenêtre Projets. L'installateur ajoute le dossier Source de sgcUI et les unités compilées de chaque plateforme (LibD<version>\<platform>) au chemin de bibliothèque de chaque plateforme qu'il compile. Sans l'installateur, ajoutez le dossier Source de sgcUI au chemin de bibliothèque de la plateforme cible (Outils, Options, Langage, Delphi, Bibliothèque).
- Les applications Android et iOS sont liées statiquement, aucun package d'exécution n'est donc déployé.

Linux64

Le package indépendant du framework sgcNavigation est compilé pour Linux64 à partir de Delphi 10.3. Une application Linux64, par exemple un programme console ou un serveur, utilise les unités indépendantes du framework. Aucun package VCL ou FireMonkey n'est compilé pour Linux64, car FireMonkey sous Linux nécessite FMX pour Linux (FMXLinux), un module complémentaire tiers.
La barre VCL n'apparaît que dans la palette d'une fiche VCL et la barre FMX que dans la palette d'une fiche FireMonkey. La propriété **Items** s'édite avec l'éditeur de collection standard.

Voir aussi

- Éditions
- Architecture
- Dépannage

Démarrage rapide VCL

À la conception

- Déposez un **TsgcVCLNavigationBar** de la page de palette **SGC Navigation** sur une fiche. Il est aligné à gauche par défaut.
- Double-cliquez sur **Items** dans l'inspecteur d'objets et ajoutez les éléments. Définissez **Caption** et **Glyph** (une lettre ou un court symbole) pour chaque élément, et mettez **IsGroup** à True sur un élément qui sert d'en-tête de groupe.
- Déposez un TPanel ou un TPageControl sur le reste de la fiche avec **Align** défini sur alClient.
- Écrivez un gestionnaire **OnItemClick** qui affiche la page de l'élément cliqué, ou affectez l'événement **OnClick** de chaque élément.
- Définissez **Theme** sur utDark pour la palette sombre et **Collapsed** sur True pour voir le rail d'icônes.

Dans le code

La même barre créée à l'exécution. Le programme se compile tel quel avec `sgcUI\Source` dans le chemin de recherche.

```
uses
  Forms, Classes, Controls, StdCtrls, ExtCtrls,
  sgcUICore_Theme, sgcNavigation_Model, sgcNavigation_VCL;

constructor TfrmMain.Create(aOwner: TComponent);
var
  oPanel: TPanel;
begin
  inherited CreateNew(aOwner);
  Caption := 'sgcNavigation VCL Quick Start';
  Width := 900;
  Height := 560;

  // the navigation bar, aligned to the left of the form
  FNav := TsgcVCLNavigationBar.Create(Self);
  FNav.Parent := Self;
  FNav.Align := alLeft;
  FNav.Mode := nbmSidebar;

  // a group header, then the items: Caption, Glyph (drawn in the icon slot)
  FNav.Items.AddItem('Main', '', True);
  FNav.Items.AddItem('Dashboard', 'D');
  FNav.Items.AddItem('Inbox', 'I').Badge := '12';
  FNav.Items.AddItem('Reports', 'R');
  FNav.Items.AddItem('System', '', True);
  FNav.Items.AddItem('Settings', 'S').Hint := 'Application settings';
  FNav.SelectedIndex := 1;
  FNav.OnItemClick := OnNavItemClick;

  // the content area on the right
  oPanel := TPanel.Create(Self);
  oPanel.Parent := Self;
  oPanel.Align := alClient;
  FTitle := TLabel.Create(Self);
  FTitle.Parent := oPanel;
  FTitle.Left := 16;
  FTitle.Top := 16;
  FTitle.Caption := 'Dashboard';

  FNav.Theme := utDark;
end;
```

```
procedure TfrmMain.OnNavItemClick(Sender: TObject; aItem: TsgcNavigationItem);
begin
  FTitle.Caption := aItem.Caption;
end;
```

Voir aussi

- TsgcVCLNavigationBar
- TsgcNavigationItem
- Mode sidebar
- Réduction et animation
- Thèmes

Démarrage rapide FMX

Le contrôle FireMonkey nécessite Delphi XE6 ou une version ultérieure. Il s'utilise exactement comme celui de la VCL : déposez un **TsgcFMXNavigationBar**, aligné à gauche par défaut, remplissez **Items** avec l'éditeur de collection et gérez **OnItemClick**. Un appui sur une ligne fonctionne comme un clic.

Dans le code

```
uses
  System.Classes, FMX.Forms, FMX.Types, FMX.StdCtrls,
  sgcUICore_Theme, sgcNavigation_Model, sgcNavigation_FMX;

constructor TfrmMain.Create(aOwner: TComponent);
begin
  inherited CreateNew(aOwner);
  Caption := 'sgcNavigation FMX Quick Start';
  Width := 900;
  Height := 560;

  FNav := TsgcFMXNavigationBar.Create(Self);
  FNav.Parent := Self;
  FNav.Align := TAlignLayout.Left;
  FNav.Items.AddItem('Main', '', True);
  FNav.Items.AddItem('Dashboard', 'D');
  FNav.Items.AddItem('Inbox', 'I').Badge := '12';
  FNav.Items.AddItem('Settings', 'S');
  FNav.SelectedIndex := 1;
  FNav.OnItemClick := OnNavItemClick;

  FTitle := TLabel.Create(Self);
  FTitle.Parent := Self;
  FTitle.Align := TAlignLayout.Client;
  FTitle.Text := 'Dashboard';

  FNav.Theme := utDark;
end;

procedure TfrmMain.OnNavItemClick(Sender: TObject; aItem: TsgcNavigationItem);
begin
  FTitle.Text := aItem.Caption;
end;
```

Voir aussi

- TsgcFMXNavigationBar
- TsgcNavigationItem
- Plateformes

Démarrage rapide C++Builder

Installez les packages sgcNavigation pour votre version de C++Builder, ajoutez `sgcUI\Source` et le dossier des unités compilées, par exemple `sgcUI\LibD13\win32` (`sgcUI\LibD13\win64` pour le 64 bits), aux chemins d'inclusion et de bibliothèque du projet, puis utilisez les composants de la page de palette **SGC Navigation** exactement comme dans Delphi. L'installateur ajoute les deux chemins à l'IDE pour vous. Les en-têtes .hpp et les fichiers .lib générés de sgcNavigation et de sgcUICore se trouvent dans ce seul dossier, sgcUICore n'a donc besoin d'aucun chemin propre.

Dans le code

OnItemClick est un événement de type closure, son gestionnaire est donc une fonction membre __fastcall d'une classe.

Voir aussi

- Démarrage rapide VCL
- Démos
- Plateformes

Démos

Les démos se trouvent dans le dossier **Demos** de l'installation, avec les projets Delphi sous Demos\Delphi et les portages C++Builder des démos VCL sous Demos\CBuilder. Chaque dossier de démo contient un readme.html qui explique ce que fait la démo et comment elle fonctionne.

Démo	Composants	Ce qu'elle montre
SidebarApp	TsgcVCLNavigationBar, TsgcVCLDashboard, TsgcVCLKPICard	Une barre de navigation en mode sidebar avec des icônes, des badges, un élément désactivé et une réduction animée, qui pilote les pages d'un TPageControl, dont l'une contient un dashboard sgcDashboard. Delphi et C++Builder, Win32 et Win64. Elle nécessite sgcDashboard.
Accordion	TsgcVCLNavigationBar, TActionList	Une barre de navigation en mode accordéon avec trois groupes, un commutateur d'expansion simple ou multiple, et des éléments liés aux actions d'un TActionList. Delphi et C++Builder, Win32 et Win64. Les liens d'action (Action links) ne font pas partie de l'édition Community, cette démo ne se compile donc pas avec elle.
SidebarFMX	TsgcFMXNavigationBar, TsgcFMXDashboard, TsgcFMXKPICard	La version FireMonkey de SidebarApp : glyphes, badges, un élément désactivé et une réduction animée, qui pilote les onglets d'un TTabControl, dont l'un contient un dashboard sgcDashboard. Delphi, Win32 et Win64. Elle nécessite sgcDashboard.

Voir aussi

- Démarrage rapide VCL
- Mode accordéon
- Liens d'action

TsgcVCLNavigationBar

Unité : sgcNavigation_VCL
TsgcVCLNavigationBar est la barre de navigation VCL, uniquement pour Windows. Elle est alignée à gauche par défaut et affiche ses Items sous forme de lignes, en mode sidebar ou en mode accordéon. Elle gère la souris, la molette, le clavier et les info-bulles, et elle défile lorsque les lignes sont plus hautes que le contrôle. Toutes les tailles sont des pixels logiques multipliés par PPI / 96.

Propriétés

Propriété	Type	Description
Items	TsgcNavigationItems	Les éléments et les en-têtes de groupe, dans l'ordre d'affichage. Voir TsgcNavigationItems.
Mode	TsgcNavigationMode	nbmSidebar (par défaut), une liste plate, ou nbmAccordion, des groupes qui se déplient et se replient.
Collapsed	Boolean	Affiche la barre sous la forme d'un rail d'icônes de largeur CollapsedWidth au lieu de ExpandedWidth. Par défaut False.
ExpandedWidth	Integer	Largeur de la barre dépliée en pixels logiques. Par défaut 220.
CollapsedWidth	Integer	Largeur du rail d'icônes en pixels logiques. Par défaut 56.
ItemHeight	Integer	Hauteur d'une ligne d'élément en pixels logiques. Par défaut 40.
GroupHeight	Integer	Hauteur d'une ligne d'en-tête de groupe en pixels logiques. Par défaut 32.
Indent	Integer	Retrait des éléments d'un groupe en mode accordéon, en pixels logiques. Par défaut 16.
AllowMultiExpand	Boolean	En mode accordéon, permet de déplier plusieurs groupes. Lorsque la valeur est False (par défaut), déplier un groupe replie les autres.
SelectedIndex	Integer	Index de l'élément sélectionné, -1 (par défaut) si aucun. Un index hors limites devient -1.
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom, qui utilise les couleurs claires.
AccentColor	TsgcUIColor	Couleur de la sélection, du focus et des badges, au format $AARRGGBB. 0 (par défaut) utilise la couleur d'accent du thème.
Images	TCustomImageList	La liste d'images de l'ImageIndex des éléments. Les images sont dessinées à leur propre taille.
AnimateCollapse	Boolean	Anime la largeur pendant environ 150 ms lorsque Collapsed change à l'exécution. Par défaut True.
Model	TsgcNavigationModel (public, lecture seule)	Le modèle de la barre. Voir TsgcNavigationModel.

ScrollY	Integer (public, lecture seule)	Le décalage de défilement vertical en pixels de l'appareil.

Les membres habituels d'un contrôle sont également publiés : **Align** (par défaut alLeft), **Anchors**, **Constraints**, **Enabled**, **ParentShowHint**, **PopupMenu**, **ShowHint**, **TabOrder**, **TabStop** (par défaut True), **Visible**, **OnClick**, **OnContextPopup**, **OnEnter**, **OnExit**, **OnKeyDown**, **OnKeyUp**, **OnMouseDown**, **OnMouseMove**, **OnMouseUp** and **OnResize**.

Méthodes

Méthode	Retour	Description
ItemAtPos(X, Y: Integer)	Integer	L'élément situé à un point du contrôle, -1 si aucun. Une surcharge renvoie aussi la partie située sous le point, npItem ou npExpander.
ItemRect(aIndex: Integer)	TRect	La ligne d'un élément en coordonnées du contrôle, après défilement, ou un rectangle vide lorsque l'élément n'est pas affiché.
ScrollIntoView(aIndex: Integer)		Fait défiler la barre pour que la ligne de l'élément soit visible.
InitiateAction		Permet à chaque élément doté d'une Action de se mettre à jour. Le framework l'appelle en période d'inactivité.

Événements

Événement	Type	Description
OnItemClick	TsgcNavigationItemEvent	Un élément activé a été cliqué, touché ou activé avec Entrée ou Espace. L'élément est déjà sélectionné. L'événement OnClick de l'élément ou son Action s'exécute après cet événement.
OnSelectionChange	TNotifyEvent	SelectedIndex a changé, par l'utilisateur ou depuis le code.
OnCollapsedChange	TNotifyEvent	Collapsed a changé.

Exemple

```
uses
  Forms, Classes, Controls, StdCtrls, ExtCtrls,
  sgcUICore_Theme, sgcNavigation_Model, sgcNavigation_VCL;

constructor TfrmMain.Create(aOwner: TComponent);
var
  oPanel: TPanel;
begin
  inherited CreateNew(aOwner);
  Caption := 'sgcNavigation VCL Quick Start';
  Width := 900;
  Height := 560;

  // the navigation bar, aligned to the left of the form
  FNav := TsgcVCLNavigationBar.Create(Self);
  FNav.Parent := Self;
```

```
  FNav.Align := alLeft;
  FNav.Mode := nbmSidebar;

  // a group header, then the items: Caption, Glyph (drawn in the icon slot)
  FNav.Items.AddItem('Main', '', True);
  FNav.Items.AddItem('Dashboard', 'D');
  FNav.Items.AddItem('Inbox', 'I').Badge := '12';
  FNav.Items.AddItem('Reports', 'R');
  FNav.Items.AddItem('System', '', True);
  FNav.Items.AddItem('Settings', 'S').Hint := 'Application settings';
  FNav.SelectedIndex := 1;
  FNav.OnItemClick := OnNavItemClick;

  // the content area on the right
  oPanel := TPanel.Create(Self);
  oPanel.Parent := Self;
  oPanel.Align := alClient;
  FTitle := TLabel.Create(Self);
  FTitle.Parent := oPanel;
  FTitle.Left := 16;
  FTitle.Top := 16;
  FTitle.Caption := 'Dashboard';

  FNav.Theme := utDark;
end;

procedure TfrmMain.OnNavItemClick(Sender: TObject; aItem: TsgcNavigationItem);
begin
  FTitle.Caption := aItem.Caption;
end;
```

Voir aussi

- TsgcFMXNavigationBar
- TsgcNavigationItem
- TsgcNavigationModel
- Clavier et focus

TsgcFMXNavigationBar

Unité : sgcNavigation_FMX
TsgcFMXNavigationBar est la barre de navigation FireMonkey, à partir de Delphi XE6, pour Windows, macOS, iOS, Android et Linux avec FMXLinux. Elle possède les propriétés, méthodes et événements de TsgcVCLNavigationBar et ajoute la saisie tactile : un appui sur une ligne fonctionne comme un clic. Les tailles sont des unités logiques multipliées par l'échelle de la fiche. À partir de 10.1 Berlin, le Hint du contrôle suit l'élément situé sous le curseur.

Propriétés

Propriété	Type	Description
Items	TsgcNavigationItems	Les éléments et les en-têtes de groupe, dans l'ordre d'affichage. Voir TsgcNavigationItems.
Mode	TsgcNavigationMode	nbmSidebar (par défaut), une liste plate, ou nbmAccordion, des groupes qui se déplient et se replient.
Collapsed	Boolean	Affiche la barre sous la forme d'un rail d'icônes de largeur CollapsedWidth au lieu de ExpandedWidth. Par défaut False.
ExpandedWidth	Integer	Largeur de la barre dépliée en pixels logiques. Par défaut 220.
CollapsedWidth	Integer	Largeur du rail d'icônes en pixels logiques. Par défaut 56.
ItemHeight	Integer	Hauteur d'une ligne d'élément en pixels logiques. Par défaut 40.
GroupHeight	Integer	Hauteur d'une ligne d'en-tête de groupe en pixels logiques. Par défaut 32.
Indent	Integer	Retrait des éléments d'un groupe en mode accordéon, en pixels logiques. Par défaut 16.
AllowMultiExpand	Boolean	En mode accordéon, permet de déplier plusieurs groupes. Lorsque la valeur est False (par défaut), déplier un groupe replie les autres.
SelectedIndex	Integer	Index de l'élément sélectionné, -1 (par défaut) si aucun. Un index hors limites devient -1.
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom, qui utilise les couleurs claires.
AccentColor	TsgcUIColor	Couleur de la sélection, du focus et des badges, au format $AARRGGBB. 0 (par défaut) utilise la couleur d'accent du thème.
Images	TCustomImageList	La liste d'images de l'ImageIndex des éléments. Sous FMX, elle nécessite XE8 ou une version ultérieure et les images sont dessinées en 16 x 16 pixels logiques.
AnimateCollapse	Boolean	Anime la largeur pendant environ 150 ms lorsque Collapsed change à l'exécution. Par défaut True.
Model	TsgcNavigationModel (pu-	Le modèle de la barre. Voir TsgcNavigationModel.

	blic, lecture seule)	
ScrollY	Integer (public, lecture seule)	Le décalage de défilement vertical en pixels de l'appareil.

Les membres habituels d'un contrôle FMX sont également publiés, notamment **Align** (par défaut Left), **Anchors**, **CanFocus** (par défaut True), **Enabled**, **HitTest**, **Margins**, **Padding**, **Opacity**, **PopupMenu**, **Position**, **Width**, **Height**, **TabOrder**, **Visible**, **ShowHint** et **TabStop** (10.1 et versions ultérieures), ainsi que les événements OnClick, OnEnter, OnExit, OnKey, OnMouse, OnPaint et OnResize.

Méthodes

Méthode	Retour	Description
ItemAtPos(X, Y: Single)	Integer	L'élément situé à un point du contrôle, -1 si aucun. Une surcharge renvoie aussi la partie située sous le point, npItem ou npExpander.
ItemRect(aIndex: Integer)	TRectF	La ligne d'un élément en coordonnées du contrôle, après défilement, ou un rectangle vide lorsque l'élément n'est pas affiché.
ScrollIntoView(aIndex: Integer)		Fait défiler la barre pour que la ligne de l'élément soit visible.
InitiateAction		Permet à chaque élément doté d'une Action de se mettre à jour. Le framework l'appelle en période d'inactivité.

Événements

Événement	Type	Description
OnItemClick	TsgcNavigationItemEvent	Un élément activé a été cliqué, touché ou activé avec Entrée ou Espace. L'élément est déjà sélectionné. L'événement OnClick de l'élément ou son Action s'exécute après cet événement.
OnSelectionChange	TNotifyEvent	SelectedIndex a changé, par l'utilisateur ou depuis le code.
OnCollapsedChange	TNotifyEvent	Collapsed a changé.

Exemple

```
uses
  System.Classes, FMX.Forms, FMX.Types, FMX.StdCtrls,
  sgcUICore_Theme, sgcNavigation_Model, sgcNavigation_FMX;

constructor TfrmMain.Create(aOwner: TComponent);
begin
  inherited CreateNew(aOwner);
  Caption := 'sgcNavigation FMX Quick Start';
  Width := 900;
  Height := 560;

  FNav := TsgcFMXNavigationBar.Create(Self);
  FNav.Parent := Self;
  FNav.Align := TAlignLayout.Left;
  FNav.Items.AddItem('Main', '', True);
```

```
  FNav.Items.AddItem('Dashboard', 'D');
  FNav.Items.AddItem('Inbox', 'I').Badge := '12';
  FNav.Items.AddItem('Settings', 'S');
  FNav.SelectedIndex := 1;
  FNav.OnItemClick := OnNavItemClick;

  FTitle := TLabel.Create(Self);
  FTitle.Parent := Self;
  FTitle.Align := TAlignLayout.Client;
  FTitle.Text := 'Dashboard';

  FNav.Theme := utDark;
end;

procedure TfrmMain.OnNavItemClick(Sender: TObject; aItem: TsgcNavigationItem);
begin
  FTitle.Text := aItem.Caption;
end;
```

Voir aussi

- TsgcVCLNavigationBar
- TsgcNavigationItem
- Plateformes

TsgcNavigationItem

Unité : sgcNavigation_Model
TsgcNavigationItem est une ligne d'une barre de navigation : une page de l'application, ou un en-tête de groupe lorsque IsGroup vaut True. Les éléments se trouvent dans la collection Items de la barre.

Propriétés

Propriété	Type	Description
Caption	string	Le texte de la ligne. Il est masqué dans le rail d'icônes réduit.
Glyph	string	Un court texte ou symbole, par exemple une lettre, dessiné au centre de l'emplacement de l'icône lorsque l'élément n'a pas d'image.
ImageIndex	Integer	Index de l'image dans les Images de la barre. Par défaut -1, aucune image.
Badge	string	Un court texte dessiné sous forme de pastille d'accent à droite de la ligne, un point sur l'icône dans le rail. Vide (par défaut), aucun badge n'est dessiné. Voir Badges.
IsGroup	Boolean	Fait de l'élément un en-tête de groupe. Un en-tête n'est jamais sélectionné. En mode accordéon, il déplie et replie les éléments situés en dessous jusqu'au prochain en-tête. Par défaut False.
Expanded	Boolean	Indique si les éléments d'un en-tête de groupe sont affichés en mode accordéon. Par défaut True.
Enabled	Boolean	Un élément désactivé est dessiné atténué et ne peut être ni sélectionné ni cliqué. Par défaut True.
Visible	Boolean	Un élément masqué n'a pas de ligne. Par défaut True.
Hint	string	L'info-bulle de la ligne. Dans le rail réduit, un élément sans Hint affiche son Caption comme info-bulle.
Tag	NativeInt	Une valeur libre pour l'application, Integer avant XE2.
Action	TBasicAction	Une action liée. Caption, Enabled, Visible, Hint et ImageIndex la suivent et un clic l'exécute, sauf si OnClick est affecté. Absente de l'édition Community. Voir Liens d'action.
BoundsRect	TsgcUIRect (public, lecture seule)	La ligne de la dernière mise en page en pixels de l'appareil, en coordonnées du contenu avant défilement.
Shown	Boolean (public, lecture seule)	Indique si la dernière mise en page a attribué une ligne à l'élément.
Separator	Boolean (public,	Indique si la ligne est une ligne de séparation d'un groupe dans le rail.

	lecture seule)	
Level	Integer (public, lecture seule)	Le niveau de retrait de la ligne, 1 pour les éléments d'un groupe accordéon.

Méthodes

Méthode	Re-tour	Description
Click(aComponent: TComponent = nil)		Exécute l'élément : son OnClick s'il est affecté, sinon son Action, exécutée avec aComponent comme ActionComponent. La barre l'appelle après OnItemClick.
InitiateAction		Permet à l'Action liée de se mettre à jour. Ne fait rien sans Action.

Événements

Événement	Type	Description
OnClick	TNotifyEvent	L'élément a été activé. Sender est l'élément.

Exemple

```
uses
  Forms, Classes, Controls, StdCtrls, ActnList, ImgList, Graphics,
  sgcUICore_Theme, sgcNavigation_Model, sgcNavigation_VCL;

procedure TfrmMain.SetupItems;
var
  oItem: TsgcNavigationItem;
begin
  FNav.Items.Clear;
  FNav.Items.AddItem('Workspace', '', True);
  FNav.Items.AddItem('Dashboard', 'D');

  oItem := FNav.Items.Add;
  oItem.Caption := 'Inbox';
  oItem.Glyph := 'I';
  oItem.Badge := '12';
  oItem.Hint := '12 unread messages';

  oItem := FNav.Items.AddItem('Billing', 'B');
  oItem.Enabled := False;
  oItem.Hint := 'Requires an administrator account';

  oItem := FNav.Items.AddItem('Settings', 'S');
  oItem.Tag := 100;
  oItem.OnClick := OnSettingsClick;
end;

procedure TfrmMain.OnSettingsClick(Sender: TObject);
begin
  // Sender is the item
  FStatus.Caption := 'Settings, tag ' +
```

```
    IntToStr((Sender as TsgcNavigationItem).Tag);
end;
```

Voir aussi

- TsgcNavigationItems
- Images et glyphes
- Badges
- Liens d'action

TsgcNavigationItems

Unité : sgcNavigation_Model
TsgcNavigationItems est la collection de TsgcNavigationItem d'une barre de navigation, une TOwnedCollection : les éléments s'éditent donc à la conception avec l'éditeur de collection standard et sont enregistrés dans le fichier de fiche. L'ordre de la collection est l'ordre des lignes. Un en-tête de groupe possède les éléments qui le suivent jusqu'au prochain en-tête.

Méthodes

Méthode	Retour	Description
Add	TsgcNavigationItem	Ajoute un élément vide à la fin.
AddItem(const aCaption: string; const aGlyph: string = ''; aIsGroup: Boolean = False)	TsgcNavigationItem	Ajoute un élément, ou un en-tête de groupe lorsque aIsGroup vaut True, avec son Caption et son Glyph, et le renvoie.
Items[Index: Integer]	TsgcNavigationItem	L'élément situé à un index. C'est la propriété par défaut, donc Nav.Items[2] est le troisième élément.

Les membres hérités de TCollection sont également disponibles : **Count**, **Clear**, **Delete**, **Insert**, **BeginUpdate** et **EndUpdate**. Encadrez de nombreuses modifications par BeginUpdate et EndUpdate pour ne calculer la mise en page de la barre qu'une seule fois.

Exemple

```
uses
  Forms, Classes, Controls, StdCtrls, ActnList, ImgList, Graphics,
  sgcUICore_Theme, sgcNavigation_Model, sgcNavigation_VCL;

procedure TfrmMain.SetupAccordion;
begin
  FNav.Items.BeginUpdate;
  try
    FNav.Items.Clear;
    // a group header owns the items that follow it until the next header
    FNav.Items.AddItem('Sales', '', True);
    FNav.Items.AddItem('Orders', 'O');
    FNav.Items.AddItem('Customers', 'C').Badge := '3';
    FNav.Items.AddItem('Inventory', '', True).Expanded := False;
    FNav.Items.AddItem('Products', 'P');
    FNav.Items.AddItem('Stock', 'S');
    FNav.Items.AddItem('Admin', '', True).Expanded := False;
    FNav.Items.AddItem('Users', 'U');
  finally
    FNav.Items.EndUpdate;
  end;
  FNav.Mode := nbmAccordion;
  // False: expanding a group closes the others
  FNav.AllowMultiExpand := False;
  // expand the Inventory group (index 3) from code
  FNav.Model.ToggleGroup(3);
end;
```

Voir aussi

- TsgcNavigationItem
- Mode accordéon

TsgcNavigationModel

Unité : sgcNavigation_Model
TsgcNavigationModel est le modèle indépendant du framework d'une barre de navigation. Chaque barre en possède un, exposé par sa propriété **Model**, et un modèle peut aussi être créé de façon autonome, sans contrôle, pour rendre une barre sous forme d'image ou pour tester la mise en page. Il n'a aucune dépendance à VCL ou FMX et se compile sur toutes les plateformes. Il est prévu pour être utilisé depuis le thread principal.

Propriétés

Les propriétés publiées sont celles de la barre : Items, Mode, Collapsed, ExpandedWidth, CollapsedWidth, ItemHeight, GroupHeight, Indent, AllowMultiExpand, SelectedIndex, Theme et AccentColor, voir TsgcVCLNavigationBar. Le modèle ajoute ces propriétés publiques :

Propriété	Type	Description
HotIndex	Integer	L'élément situé sous la souris, -1 si aucun.
PressedIndex	Integer	L'élément en cours d'appui, -1 si aucun.
FocusedIndex	Integer	L'élément ayant le focus clavier, -1 si aucun.
ShowFocus	Boolean	Dessine le rectangle de focus. Le contrôle l'active pendant l'utilisation du clavier.
LayoutWidth	Integer (lecture seule)	La largeur du dernier Layout, en pixels de l'appareil.
LayoutScale	Double (lecture seule)	L'échelle du dernier Layout.

Méthodes

Méthode	Retour	Description
Layout(aWidth: Integer; aScale: Double = 1)		Calcule les lignes pour une barre large de aWidth pixels de l'appareil. aScale multiplie chaque taille logique.
ContentHeight	Double	Hauteur des lignes plus la marge intérieure du dernier Layout, en pixels de l'appareil.
HitTest(aX, aY: Double; out aPart: TsgcNavigationPart)	Integer	L'élément situé à un point, en coordonnées du contenu du dernier Layout, -1 si aucun. aPart vaut npItem, ou npExpander sur l'indicateur de dépliage d'un groupe accordéon. Une surcharge n'a pas de aPart. Les éléments désactivés sont signalés, un séparateur du rail ne l'est pas.
IsItemShown(aIndex: Integer)	Boolean	Indique si l'élément obtient une ligne avec le Mode, l'état Collapsed et l'état de groupe actuels.
IsItemFocusable(aIndex: Integer)	Boolean	Indique si le clavier peut s'arrêter sur l'élément : affiché, activé et n'étant pas un en-tête de groupe, sauf dans l'accordéon déplié.
NextVisible(aIndex: Integer)	Integer	L'élément suivant pouvant recevoir le focus, -1 si aucun. NextVisible(-1) est le premier.

PrevVisible(aIndex: Integer)	Integer	L'élément précédent pouvant recevoir le focus, -1 si aucun. PrevVisible(Items.Count) est le dernier.
GroupOf(aIndex: Integer)	Integer	L'en-tête de groupe auquel appartient un élément, -1 lorsque l'élément est un groupe ou qu'aucun groupe ne le précède.
ToggleGroup(aIndex: Integer)		Déplie ou replie un en-tête de groupe. Le dépliage replie les autres groupes lorsque AllowMultiExpand vaut False.
CurrentWidth	Integer	La largeur logique pour l'état Collapsed actuel, ExpandedWidth ou CollapsedWidth.
GetAccent	TsgcUIColor	La couleur d'accent utilisée : AccentColor, ou l'accent du thème lorsqu'elle vaut 0.

Événements

Événement	Type	Description
OnChange	TNotifyEvent	Le modèle a changé et doit être redessiné. La barre l'affecte, ne l'utilisez donc que sur un modèle créé de façon autonome.

Exemple

```
uses
  SysUtils,
  sgcUICore_Theme, sgcNavigation_Model, sgcNavigation_Render;

var
  oModel: TsgcNavigationModel;
  vIndex: Integer;
  vPart: TsgcNavigationPart;
begin
  oModel := TsgcNavigationModel.Create;
  try
    oModel.Items.AddItem('Main', '', True);
    oModel.Items.AddItem('Dashboard', 'D');
    oModel.Items.AddItem('Inbox', 'I').Badge := '12';
    oModel.Items.AddItem('Reports', 'R');
    oModel.Items.AddItem('Settings', 'S');
    oModel.SelectedIndex := 1;
    oModel.Theme := utDark;

    // 220 x 320 logical pixels, written at scale 2 as 440 x 640 pixels
    sgcNavigationSaveToPNG(oModel, 'navigation.png', 220, 320, 2);
    sgcNavigationSaveToSVG(oModel, 'navigation.svg', 220, 320);

    // the icon rail of the same bar
    oModel.Collapsed := True;
    sgcNavigationSaveToPNG(oModel, 'navigation_rail.png',
      oModel.CurrentWidth, 320);

    // the row under a point, after a Layout at the same width and scale
    oModel.Collapsed := False;
    oModel.Layout(220, 1);
    vIndex := oModel.HitTest(40, 60, vPart);
    if vIndex >= 0 then
      WriteLn('Hit: ', oModel.Items[vIndex].Caption);
    WriteLn('Content height: ', Round(oModel.ContentHeight));
  finally
```

```
    oModel.Free;
  end;
end;
```

Voir aussi

- Rendu sans interface
- Architecture
- TsgcVCLNavigationBar

Mode sidebar

Le mode sidebar, **Mode** = nbmSidebar, est le mode par défaut. La barre affiche chaque élément visible sous forme de ligne, empilées à partir du haut. Un en-tête de groupe est un libellé statique de couleur atténuée qui intitule les éléments situés en dessous, il n'est jamais sélectionné et un clic dessus ne fait rien.

- Une ligne d'élément mesure ItemHeight pixels de haut et une ligne d'en-tête de groupe GroupHeight pixels.
- Un clic ou un appui sur un élément activé le sélectionne, déclenche OnItemClick puis exécute l'OnClick de l'élément ou son Action.
- La ligne sélectionnée reçoit un arrière-plan teinté de la couleur d'accent et une barre d'accent de 3 pixels sur son bord gauche.
- Lorsque les lignes sont plus hautes que le contrôle, la molette de la souris fait défiler la barre, et ScrollInto-View affiche un élément donné.
- Collapsed transforme la sidebar en un rail d'icônes, voir Réduction et animation.

Exemple

```
uses
  Forms, Classes, Controls, StdCtrls, ExtCtrls,
  sgcUICore_Theme, sgcNavigation_Model, sgcNavigation_VCL;

constructor TfrmMain.Create(aOwner: TComponent);
var
  oPanel: TPanel;
begin
  inherited CreateNew(aOwner);
  Caption := 'sgcNavigation VCL Quick Start';
  Width := 900;
  Height := 560;

  // the navigation bar, aligned to the left of the form
  FNav := TsgcVCLNavigationBar.Create(Self);
  FNav.Parent := Self;
  FNav.Align := alLeft;
  FNav.Mode := nbmSidebar;

  // a group header, then the items: Caption, Glyph (drawn in the icon slot)
  FNav.Items.AddItem('Main', '', True);
  FNav.Items.AddItem('Dashboard', 'D');
  FNav.Items.AddItem('Inbox', 'I').Badge := '12';
  FNav.Items.AddItem('Reports', 'R');
  FNav.Items.AddItem('System', '', True);
  FNav.Items.AddItem('Settings', 'S').Hint := 'Application settings';
  FNav.SelectedIndex := 1;
  FNav.OnItemClick := OnNavItemClick;

  // the content area on the right
  oPanel := TPanel.Create(Self);
  oPanel.Parent := Self;
  oPanel.Align := alClient;
  FTitle := TLabel.Create(Self);
  FTitle.Parent := oPanel;
  FTitle.Left := 16;
  FTitle.Top := 16;
  FTitle.Caption := 'Dashboard';

  FNav.Theme := utDark;
end;

procedure TfrmMain.OnNavItemClick(Sender: TObject; aItem: TsgcNavigationItem);
begin
```

```
  FTitle.Caption := aItem.Caption;
end;
```

Voir aussi

- Mode accordéon
- Réduction et animation
- TsgcVCLNavigationBar

Mode accordéon

En mode accordéon, **Mode** = nbmAccordion, un en-tête de groupe déplie et replie les éléments qui le suivent jusqu'au prochain en-tête. Les éléments d'un groupe ne sont affichés que tant que le groupe est visible et que sa propriété **Expanded** vaut True, et ils sont indentés de **Indent**. L'en-tête possède un chevron de dépliage sur son bord droit.

- Un clic sur la ligne d'en-tête ou sur son chevron bascule le groupe. Le clavier fait de même avec Entrée ou Espace, Gauche replie et Droite déplie le groupe ayant le focus.
- Avec **AllowMultiExpand** à False (par défaut), déplier un groupe replie les autres, un seul groupe est donc ouvert à la fois. Mettez-le à True pour laisser plusieurs groupes ouverts.
- Dans le code, **Model.ToggleGroup** bascule un groupe avec la même règle, et définir **Expanded** sur un en-tête l'ouvre ou le ferme sans toucher aux autres.
- Les éléments situés avant le premier en-tête de groupe n'appartiennent à aucun groupe et sont toujours affichés.
- Dans le rail réduit, les groupes ne sont pas basculés : chaque élément visible qui n'est pas un en-tête est affiché, et chaque groupe devient une ligne de séparation.

Exemple

```
uses
  Forms, Classes, Controls, StdCtrls, ActnList, ImgList, Graphics,
  sgcUICore_Theme, sgcNavigation_Model, sgcNavigation_VCL;

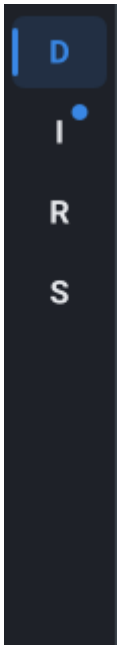
procedure TfrmMain.SetupAccordion;
begin
  FNav.Items.BeginUpdate;
  try
    FNav.Items.Clear;
    // a group header owns the items that follow it until the next header
    FNav.Items.AddItem('Sales', '', True);
    FNav.Items.AddItem('Orders', 'O');
    FNav.Items.AddItem('Customers', 'C').Badge := '3';
    FNav.Items.AddItem('Inventory', '', True).Expanded := False;
    FNav.Items.AddItem('Products', 'P');
    FNav.Items.AddItem('Stock', 'S');
    FNav.Items.AddItem('Admin', '', True).Expanded := False;
    FNav.Items.AddItem('Users', 'U');
  finally
    FNav.Items.EndUpdate;
  end;
  FNav.Mode := nbmAccordion;
  // False: expanding a group closes the others
  FNav.AllowMultiExpand := False;
  // expand the Inventory group (index 3) from code
  FNav.Model.ToggleGroup(3);
end;
```

Voir aussi

- Mode sidebar
- TsgcNavigationItems
- Démos

Réduction et animation

Collapsed fait passer la largeur de la barre entre **ExpandedWidth** (par défaut 220) et **CollapsedWidth** (par défaut 56), toutes deux en pixels logiques. La barre réduite est un rail d'icônes, en mode sidebar comme en mode accordéon :



- Seul l'emplacement de l'icône de chaque élément est dessiné. Les libellés sont masqués, et l'info-bulle d'un élément sans Hint est son Caption.
- Un en-tête de groupe devient une ligne de séparation, sauf s'il serait la première ligne ou s'il suivrait un autre séparateur.
- Un badge devient un point sur l'icône.
- Avec **AnimateCollapse** à True (par défaut), la largeur est animée pendant environ 150 ms à l'exécution. À la conception, ou avec AnimateCollapse à False, elle change immédiatement.
- **OnCollapsedChange** se déclenche lorsque Collapsed change.

La barre définit sa propre largeur Width, laissez donc Align sur alLeft (VCL) ou Left (FMX) et faites utiliser alClient au contenu situé à sa droite. Le modèle indique la largeur avec **CurrentWidth**.

Exemple

```
uses
  Forms, Classes, Controls, StdCtrls, ActnList, ImgList, Graphics,
  sgcUICore_Theme, sgcNavigation_Model, sgcNavigation_VCL;

procedure TfrmMain.SetupCollapse;
var
  oButton: TButton;
begin
  FNav.ExpandedWidth := 220;
  FNav.CollapsedWidth := 56;
  FNav.AnimateCollapse := True;
  FNav.OnCollapsedChange := OnNavCollapsedChange;

  oButton := TButton.Create(Self);
  oButton.Parent := Self;
  oButton.Caption := 'Collapse';
  oButton.OnClick := OnToggleClick;
end;

procedure TfrmMain.OnToggleClick(Sender: TObject);
begin
  // animates the width to CollapsedWidth, the bar becomes an icon rail
```

```
  FNav.Collapsed := not FNav.Collapsed;
end;

procedure TfrmMain.OnNavCollapsedChange(Sender: TObject);
begin
  if FNav.Collapsed then
    FStatus.Caption := 'Icon rail'
  else
    FStatus.Caption := 'Expanded';
end;
```

Voir aussi

- Mode sidebar
- Badges
- TsgcVCLNavigationBar

Images et glyphes

Chaque ligne possède un emplacement d'icône sur sa gauche. Il affiche, par ordre de préférence :

- l'image **ImageIndex** de la liste d'images **Images** de la barre, lorsque les deux sont définis ;
- sinon le texte **Glyph** de l'élément, centré, par exemple une lettre ou un court symbole ;
- sinon rien.

Un glyphe n'a pas besoin de liste d'images et suit les couleurs du thème, c'est pourquoi les démos utilisent des lettres. Un élément désactivé dessine son image avec 40 % d'opacité.

VCL

Images est un TCustomImageList, et les images sont dessinées à leur propre taille. Les listes d'images masquées et 32 bits fonctionnent toutes deux : chaque image est convertie une fois en ARGB et mise en cache jusqu'à ce que la liste d'images change.

FireMonkey

Images est le TCustomImageList FMX de FMX.ImgList et nécessite Delphi XE8 ou une version ultérieure. Les images sont dessinées en 16 x 16 pixels logiques, prises dans la liste d'images à la taille de l'appareil de l'image affichée, et mises en cache jusqu'à ce que la liste d'images change.

Exemple

```
uses
  Forms, Classes, Controls, StdCtrls, ActnList, ImgList, Graphics,
  sgcUICore_Theme, sgcNavigation_Model, sgcNavigation_VCL;

procedure TfrmMain.SetupImages;
var
  oImages: TImageList;
  oBitmap: TBitmap;
  i: Integer;
begin
  oImages := TImageList.Create(Self);
  oImages.Width := 16;
  oImages.Height := 16;
  oBitmap := TBitmap.Create;
  try
    oBitmap.Width := 16;
    oBitmap.Height := 16;
    oBitmap.Canvas.Brush.Color := clTeal;
    oBitmap.Canvas.Ellipse(1, 1, 15, 15);
    oImages.AddMasked(oBitmap, clWhite);
  finally
    oBitmap.Free;
  end;

  FNav.Images := oImages;
  // an item with an ImageIndex draws the image, the others their Glyph
  for i := 0 to FNav.Items.Count - 1 do
    if not FNav.Items[i].IsGroup then
      FNav.Items[i].ImageIndex := 0;
end;
```

Voir aussi

- TsgcNavigationItem

- Plateformes

Badges

Badge est un court texte d'un élément, par exemple un nombre de non lus ou le mot New. Il est dessiné sous forme de pastille de la couleur d'accent, aligné à droite dans la ligne, avec un texte noir ou blanc, selon ce qui se lit le mieux sur l'accent. Un Badge vide ne dessine rien. Dans le rail d'icônes réduit, le badge devient un point sur l'icône.
Définissez Badge depuis le code chaque fois que la valeur change, la barre se redessine d'elle-même.

Exemple

```
uses
  Forms, Classes, Controls, StdCtrls, ActnList, ImgList, Graphics,
  sgcUICore_Theme, sgcNavigation_Model, sgcNavigation_VCL;

procedure TfrmMain.SetupItems;
var
  oItem: TsgcNavigationItem;
begin
  FNav.Items.Clear;
  FNav.Items.AddItem('Workspace', '', True);
  FNav.Items.AddItem('Dashboard', 'D');

  oItem := FNav.Items.Add;
  oItem.Caption := 'Inbox';
  oItem.Glyph := 'I';
  oItem.Badge := '12';
  oItem.Hint := '12 unread messages';

  oItem := FNav.Items.AddItem('Billing', 'B');
  oItem.Enabled := False;
  oItem.Hint := 'Requires an administrator account';

  oItem := FNav.Items.AddItem('Settings', 'S');
  oItem.Tag := 100;
  oItem.OnClick := OnSettingsClick;
end;

procedure TfrmMain.OnSettingsClick(Sender: TObject);
begin
  // Sender is the item
  FStatus.Caption := 'Settings, tag ' +
    IntToStr((Sender as TsgcNavigationItem).Tag);
end;
```

Voir aussi

- TsgcNavigationItem
- Thèmes
- Réduction et animation

Clavier et focus

La barre prend le focus clavier (**TabStop** sous VCL, **CanFocus** sous FMX, tous deux à True par défaut) et déplace une ligne ayant le focus sur les éléments pouvant le recevoir : les éléments affichés et activés, et, dans l'accordéon déplié, également les en-têtes de groupe.

Touche	Action
Haut, Bas	Déplace le focus vers la ligne précédente ou suivante pouvant le recevoir, et la fait défiler en vue.
Début, Fin	Déplace le focus vers la première ou la dernière ligne pouvant le recevoir.
Entrée, Espace	Active la ligne ayant le focus comme un clic : sélectionne un élément et déclenche OnItemClick, ou bascule un groupe accordéon.
Gauche, Droite	Replie ou déplie le groupe accordéon ayant le focus. Gauche sur un élément d'un groupe déplace le focus vers son en-tête de groupe.

Le rectangle de focus en pointillés n'est affiché que pendant l'utilisation du clavier. La sélection et le focus sont distincts : **SelectedIndex** est la page choisie par l'utilisateur, **Model.FocusedIndex** la ligne sur laquelle se trouve le clavier.

Info-bulles

Avec **ShowHint** à True, l'info-bulle est le **Hint** de l'élément situé sous le curseur, ou son Caption dans le rail réduit, de sorte que passer à une autre ligne affiche son info-bulle. Sous FMX, les info-bulles nécessitent 10.1 Berlin ou une version ultérieure.

Exemple

```
uses
  Forms, Classes, Controls, StdCtrls, ActnList, ImgList, Graphics,
  sgcUICore_Theme, sgcNavigation_Model, sgcNavigation_VCL;

procedure TfrmMain.SelectByCode;
begin
  FNav.TabStop := True;
  FNav.OnSelectionChange := OnNavSelectionChange;
  FNav.OnKeyDown := OnNavKeyDown;
  // select the second item and scroll it into view
  FNav.SelectedIndex := 1;
  FNav.ScrollIntoView(FNav.SelectedIndex);
end;

procedure TfrmMain.OnNavSelectionChange(Sender: TObject);
begin
  if FNav.SelectedIndex >= 0 then
    FStatus.Caption := FNav.Items[FNav.SelectedIndex].Caption;
end;

procedure TfrmMain.OnNavKeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  // F2 collapses and expands the bar; the arrows, Home, End, Enter and
  // Space are handled by the bar itself
  if Key = $71 then
```

```
    FNav.Collapsed := not FNav.Collapsed;
end;
```

Voir aussi

- TsgcVCLNavigationBar
- TsgcNavigationModel
- Mode accordéon

Liens d'action

Un élément se lie à n'importe quel TBasicAction, un TAction VCL ou FireMonkey, par sa propriété **Action**. Les liens d'action (Action links) ne font pas partie de l'édition Community, voir Éditions.

- **Caption**, **Enabled**, **Visible**, **Hint** et **ImageIndex** de l'élément suivent l'action.
- Un clic sur l'élément déclenche OnItemClick de la barre, puis exécute l'action, avec la barre comme Action-Component. Lorsque l'élément possède un gestionnaire OnClick, OnClick s'exécute à la place de l'action.
- L'OnUpdate des actions s'exécute en période d'inactivité, car la barre appelle InitiateAction sur chaque élément.
- Sous Delphi XE3 et versions ultérieures, l'action transmet chaque modification à l'élément. Sur les versions plus anciennes, l'élément copie les propriétés de l'action lorsqu'elle est affectée, puis à chaque InitiateAction.

Exemple

```
uses
  Forms, Classes, Controls, StdCtrls, ActnList, ImgList, Graphics,
  sgcUICore_Theme, sgcNavigation_Model, sgcNavigation_VCL;

procedure TfrmMain.SetupActions;
var
  oActions: TActionList;
  oSave: TAction;
  oItem: TsgcNavigationItem;
begin
  oActions := TActionList.Create(Self);
  oSave := TAction.Create(Self);
  oSave.ActionList := oActions;
  oSave.Caption := 'Save';
  oSave.Hint := 'Save the current document';
  oSave.OnExecute := OnSaveExecute;
  oSave.OnUpdate := OnSaveUpdate;

  // Caption, Hint, Enabled, Visible and ImageIndex now follow the action,
  // and a click on the item executes it
  oItem := FNav.Items.Add;
  oItem.Glyph := 'S';
  oItem.Action := oSave;
end;

procedure TfrmMain.OnSaveExecute(Sender: TObject);
begin
  FStatus.Caption := 'Saved';
end;

procedure TfrmMain.OnSaveUpdate(Sender: TObject);
begin
  (Sender as TAction).Enabled := FStatus.Caption <> 'Saved';
end;
```

Voir aussi

- TsgcNavigationItem
- Démos
- Éditions

Thèmes

Theme définit les couleurs de la barre à partir des jetons de thème de sgcUICore : utLight (par défaut), utDark, ou utCustom, qui utilise les couleurs claires. La surface, la bordure du bord droit, le texte, les en-têtes de groupe atténués, les superpositions de survol et d'appui, ainsi que l'accent proviennent tous du thème.
AccentColor remplace l'accent du thème par une couleur $AARRGGBB, utilisée par la barre et l'arrière-plan de la sélection, le rectangle de focus et les badges. 0 (par défaut) conserve l'accent du thème.
Les jetons de thème sont les mêmes que ceux de sgcDashboard et des autres produits sgcUICore, une barre et un dashboard ayant le même Theme s'accordent donc. Une barre hébergée dans un tile sgcDashboard suit le thème du dashboard, car le tile transmet son thème à la propriété publiée Theme des contrôles qu'il héberge.

Exemple

```
uses
  Forms, Classes, Controls, StdCtrls, ActnList, ImgList, Graphics,
  sgcUICore_Theme, sgcNavigation_Model, sgcNavigation_VCL;

procedure TfrmMain.SetupTheme;
begin
  FNav.Theme := utDark;
  // $AARRGGBB, 0 uses the accent colour of the theme
  FNav.AccentColor := $FF7C3AED;
end;
```

Voir aussi

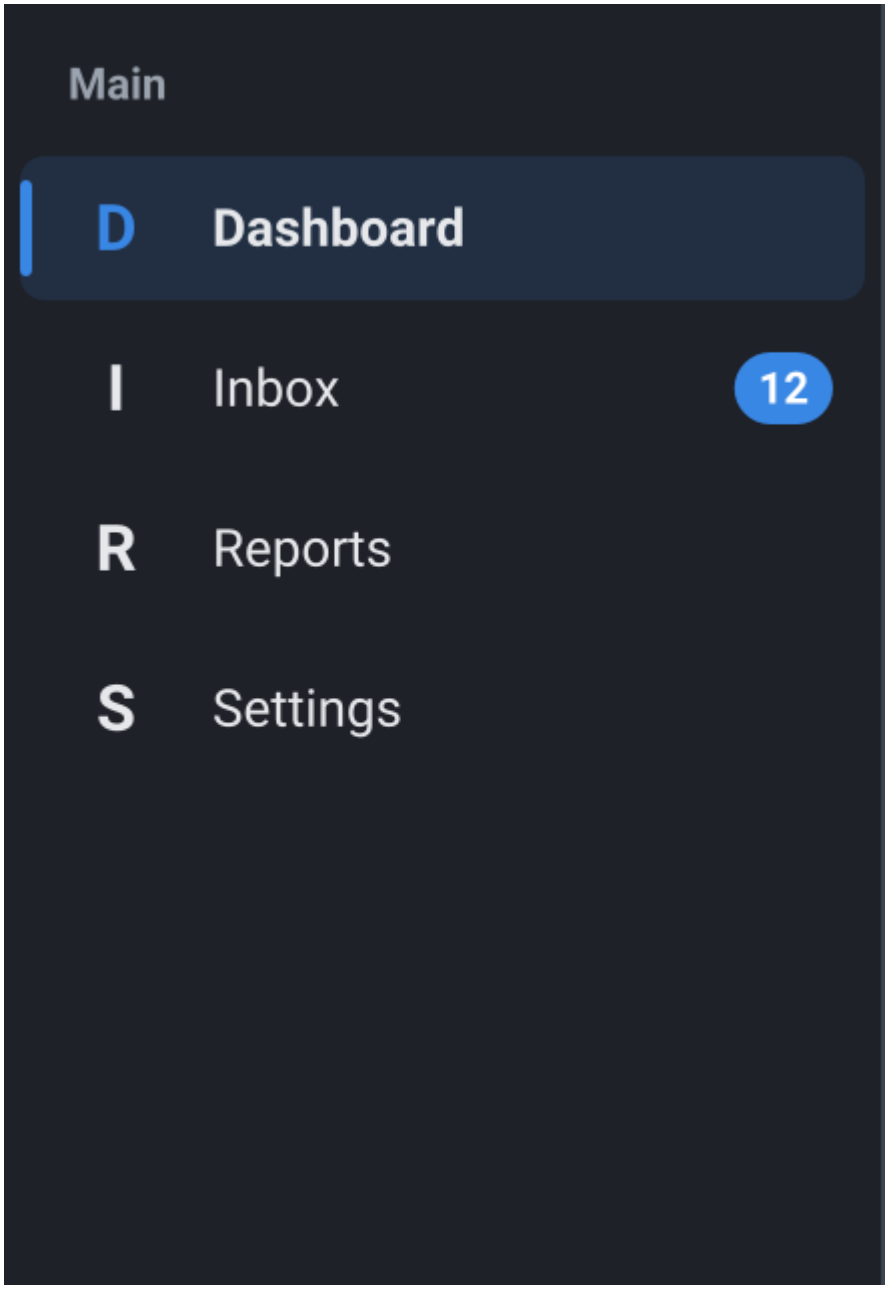
- Architecture
- Badges
- Rendu sans interface

Rendu sans interface

Le moteur de rendu de sgcNavigation_Render dessine un TsgcNavigationModel sans fenêtre et sans VCL ni FMX, de sorte qu'une application console, un service, un test ou un serveur web peut produire la même barre sous forme d'image, par exemple pour la documentation d'une application ou un test visuel. L'unité se compile sur toutes les plateformes, Linux64 compris.

Routine	Description
sgcNavigationSaveToPNG(aModel, aFileName or aStream, aWidth, aHeight, aScale = 1, aImages = nil)	Écrit un fichier ou un flux PNG. aWidth et aHeight sont des pixels logiques et aScale les multiplie : 220 par 320 à l'échelle 2 donne une image de 440 par 640.
sgcNavigationSaveToSVG(aModel, aFileName or aStream, aWidth, aHeight, aScale = 1, aImages = nil)	Écrit un fichier ou un flux SVG de la même barre.
sgcNavigationRender(aModel, aCanvas, aWidth, aHeight, aScale = 1, aScrollY = 0, aImages = nil)	Dessine la barre sur n'importe quel canvas sgcUI-Core, en pixels de l'appareil.

aImages est un rappel optionnel de type TsgcNavigationImageEvent qui renvoie l'image d'un ImageIndex. Sans lui, les éléments dessinent leur Glyph. Le modèle d'un contrôle est disponible par sa propriété **Model**, si bien que **sgcNavigationSaveToPNG(Nav.Model, ...)** enregistre ce que le contrôle affiche, à n'importe quelle taille.



Exemple

```
uses
  SysUtils,
  sgcUICore_Theme, sgcNavigation_Model, sgcNavigation_Render;

var
  oModel: TsgcNavigationModel;
  vIndex: Integer;
  vPart: TsgcNavigationPart;
begin
  oModel := TsgcNavigationModel.Create;
  try
    oModel.Items.AddItem('Main', '', True);
    oModel.Items.AddItem('Dashboard', 'D');
    oModel.Items.AddItem('Inbox', 'I').Badge := '12';
    oModel.Items.AddItem('Reports', 'R');
    oModel.Items.AddItem('Settings', 'S');
    oModel.SelectedIndex := 1;
    oModel.Theme := utDark;

    // 220 x 320 logical pixels, written at scale 2 as 440 x 640 pixels
```

```
    sgcNavigationSaveToPNG(oModel, 'navigation.png', 220, 320, 2);
    sgcNavigationSaveToSVG(oModel, 'navigation.svg', 220, 320);

    // the icon rail of the same bar
    oModel.Collapsed := True;
    sgcNavigationSaveToPNG(oModel, 'navigation_rail.png',
      oModel.CurrentWidth, 320);

    // the row under a point, after a Layout at the same width and scale
    oModel.Collapsed := False;
    oModel.Layout(220, 1);
    vIndex := oModel.HitTest(40, 60, vPart);
    if vIndex >= 0 then
      WriteLn('Hit: ', oModel.Items[vIndex].Caption);
    WriteLn('Content height: ', Round(oModel.ContentHeight));
  finally
    oModel.Free;
  end;
end;
```

Voir aussi

- TsgcNavigationModel
- Plateformes

Vue d'ensemble de .NET

sgcNavigation .NET est un portage C# managé de la bibliothèque Delphi sgcNavigation, et non un wrapper autour d'elle. Les classes C# se trouvent dans l'espace de noms **esegece.sgcNavigation** et gardent les noms Delphi, de sorte que **TsgcNavigationModel**, **TsgcNavigationItems**, **TsgcNavigationItem** et **TsgcNavigationMode** se lisent de la même façon dans les deux langages, et la barre de navigation de chaque hôte suit le modèle de nommage Delphi. Elle est dessinée avec le portage .NET de **sgcUICore**, le même canevas, les mêmes polices et les mêmes thèmes clair et sombre que les autres produits de la suite visuelle eSeGeCe, de sorte qu'une barre de navigation a le même aspect dans WinForms, WPF, Avalonia et dans la sortie headless en PNG et SVG.
Une barre de navigation affiche une liste d'éléments avec une icône, un libellé et un badge facultatif. En mode side-bar, tous les éléments sont affichés et les en-têtes de groupe sont des libellés statiques. En mode accordéon, un clic sur l'en-tête d'un groupe développe ou réduit ses éléments. Dans les deux modes, la barre se replie en un rail d'icônes, animé entre ExpandedWidth et CollapsedWidth.

Packages

Le portage est réparti en cinq packages NuGet, de sorte qu'un projet ne récupère que l'hôte qu'il utilise.

PackageId	Contient	Frame-works cibles
esegece.sgcNavigation	Le modèle (**TsgcNavigationModel**, **TsgcNavigationI-tems**, **TsgcNavigationItem**), le lien **Command** de l'élément, le moteur de rendu **sgcNavigation_Render** et la sortie headless en SVG. Aucune dépendance na-tive. Dépend de esegece.sgcUICore.	netstan-dard2.0, net472, net6.0, net8.0, net9.0
esegece.sgcNavigation.Rendering	La sortie headless en PNG, **sgcNavigation_Rendering**, dessinée sur le canevas SkiaSharp de esegece.sgcUICore.Rendering, sous Windows, Linux et macOS.	netstan-dard2.0, net472, net6.0, net8.0, net9.0
esegece.sgcNavigation.WinForms	**TsgcWinFormsNavigationBar**.	net472, net8.0-win-dows, net9.0-win-dows
esegece.sgcNavigation.Wpf	**TsgcWpfNavigationBar**.	net472, net8.0-win-dows, net9.0-win-dows
esegece.sgcNavigation.Avalonia	**TsgcAvaloniaNavigationBar**, sous Windows, Linux et macOS avec Avalonia 11.	net8.0, net9.0

Les trois packages d'hôte dépendent de esegece.sgcNavigation.Rendering, qui dépend de esegece.sgcNavigation, de sorte qu'une seule ligne `PackageReference` apporte tout ce dont un hôte a besoin. Voir Installation depuis NuGet.

Éditions

Édition	Ids des packages	Inclut
Registered	esegece.sgcNavigation, esegece.sgcNavigation.Rendering, esegece.sgcNavigation.WinForms, esegece.sgcNavigation.Wpf, esegece.sgcNavigation.Avalonia	Tout : les barres de navigation des trois hôtes, les modes sidebar et accordéon, le rail d'icônes, les images, les badges, la navigation au clavier, le lien Command de l'élément, les thèmes et la sortie headless en PNG et SVG.
Community	Les mêmes ids avec le suffixe .Community, par exemple esegece.sgcNavigation.WinForms.Community. Ils dépendent des packages Community de sgcUICore.	Le même contenu que Registered sans les propriétés **Command** et **CommandParameter** de l'élément, car l'édition Community de Delphi n'a pas de liens d'action. Les éléments s'exécutent par leur OnClick.

L'édition Community est gratuite pour un usage commercial et non commercial par des particuliers et des organisations dont le chiffre d'affaires annuel brut est inférieur à 1 000 000 EUR et qui comptent au plus cinq développeurs. Une organisation qui atteint ou dépasse l'un des deux seuils a besoin d'une licence Registered. Le portage .NET n'a pas d'édition Trial.

Hôtes

Hôte	Équivalent Delphi	Plateformes
WinForms	VCL (TsgcVCLNavigationBar)	Windows, .NET Framework 4.7.2 et .NET 8 et 9.
WPF	FMX (TsgcFMXNavigationBar)	Windows, .NET Framework 4.7.2 et .NET 8 et 9. La barre est déclarée en XAML et les éléments sont ajoutés dans le code.
Avalonia	FMX (TsgcFMXNavigationBar)	Windows, Linux et macOS, .NET 8 et 9, avec souris, tactile et stylet. La barre est déclarée en XAML et les éléments sont ajoutés dans le code.
Headless	sgcNavigation_Render	Toute plateforme des frameworks cibles : SVG avec esegece.sgcNavigation, PNG avec esegece.sgcNavigation.Rendering.

Chaque hôte a la même API : Items, Mode, Collapsed, ExpandedWidth, CollapsedWidth, ItemHeight, GroupHeight, Indent, AllowMultiExpand, SelectedIndex, Theme, AccentColor, Images, AnimateCollapse, ScrollY, ItemAtPos, ItemRect, ScrollIntoView et les événements OnItemClick, OnSelectionChange et OnCollapsedChange. Voir Classes .NET pour les noms des classes et Différences de .NET avec Delphi pour ce qui change par rapport à Delphi.

Voir aussi

- Installation depuis NuGet
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Classes .NET
- Différences de .NET avec Delphi
- Rendu sans interface en .NET

- Démos .NET
- Plateformes

Installation depuis NuGet

Les packages sont sur nuget.org. Ajoutez le package de l'hôte, les packages core et de rendu arrivent avec lui comme dépendances.

Édition Registered

```
dotnet add package esegece.sgcNavigation.WinForms
dotnet add package esegece.sgcNavigation.Wpf
dotnet add package esegece.sgcNavigation.Avalonia
```

Édition Community

```
dotnet add package esegece.sgcNavigation.WinForms.Community
dotnet add package esegece.sgcNavigation.Wpf.Community
dotnet add package esegece.sgcNavigation.Avalonia.Community
```

Ou ajoutez la référence au fichier de projet à la main :

```
<ItemGroup>
  <PackageReference Include="esegece.sgcNavigation.Wpf" Version="2026.10.0" />
</ItemGroup>
```

Référencez soit les packages Registered, soit les packages Community dans un projet, jamais les deux : ils contiennent les mêmes types dans le même espace de noms. Le code qui utilise le **Command** de l'élément ne compile pas avec les packages Community ; protégez-le avec `#if !SGC_COMMUNITY` lorsqu'un projet compile les deux éditions.

Paramètres du projet

- WinForms : un projet `net8.0-windows` avec `<UseWindowsForms>true</UseWindowsForms>`, ou un projet .NET Framework 4.7.2.
- WPF : un projet `net8.0-windows` avec `<UseWPF>true</UseWPF>`, ou un projet .NET Framework 4.7.2. Associez l'espace de noms en XAML avec `xmlns:sgc="clr-namespace:esegece.sgcNavigation;assembly=esegece.sgcNavigation.Wpf"`.
- Avalonia : un projet `net8.0` ou `net9.0` avec les packages **Avalonia.Desktop** et **Avalonia.Themes.Fluent** d'Avalonia 11.3, comme toute application de bureau Avalonia. Associez l'espace de noms en XAML avec `xmlns:sgc="using:esegece.sgcNavigation"`.
- Headless : esegece.sgcNavigation seul suffit pour le SVG. Ajoutez esegece.sgcNavigation.Rendering pour le PNG, il apporte SkiaSharp et ses ressources natives pour Windows et Linux.

En C#, chaque hôte a besoin de `using esegece.sgcNavigation;`, et de `using esegece.sgcUICore;` pour **TsgcUITheme** et les autres types des thèmes.

Voir aussi

- Vue d'ensemble de .NET
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Démos .NET

Démarrage rapide WinForms

TsgcWinFormsNavigationBar est la barre de navigation WinForms, la contrepartie du TsgcVCLNavigationBar de Delphi. Elle possède un **TsgcNavigationModel**, expose ses propriétés et le dessine par l'intermédiaire de sgcUI-Core. Le package est **esegece.sgcNavigation.WinForms**.

Étapes

- Créez un projet WinForms et ajoutez le package esegece.sgcNavigation.WinForms.
- Créez un **TsgcWinFormsNavigationBar**. Il s'ancre à gauche par défaut et prend sa largeur de **ExpandedWidth**.
- Affectez une **ImageList** à **Images** et ajoutez les éléments avec `Items.AddItem(aCaption)`. Définissez **ImageIndex**, **Badge** et **Hint** de chaque élément. `AddItem(aCaption, "", true)` ajoute un en-tête de groupe.
- Affectez **OnSelectionChange** pour afficher la page de l'élément sélectionné, et **OnItemClick** pour réagir à chaque clic.
- Exécutez : un clic ou Entrée sélectionne un élément, les touches fléchées, Début et Fin déplacent le focus, et **Collapsed** replie la barre en rail d'icônes.

Exemple

Le Program.cs complet d'une application WinForms avec une sidebar, un badge, un en-tête de groupe et un bouton Collapse. Il compile tel quel dans un projet `net8.0-windows` qui référence esegece.sgcNavigation.WinForms.
La barre figure aussi dans la boîte à outils. Les éléments ne sont pas sérialisés par le concepteur, ils sont donc toujours ajoutés dans le code, voir Différences de .NET avec Delphi.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- Différences de .NET avec Delphi
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Images et glyphes

Démarrage rapide WPF

TsgcWpfNavigationBar est la barre de navigation WPF. Elle est déclarée en XAML comme n'importe quel contrôle, et ses éléments sont ajoutés dans le code. Le package est **esegece.sgcNavigation.Wpf**.

Étapes

- Créez un projet WPF (`dotnet new wpf`) et ajoutez le package esegece.sgcNavigation.Wpf.
- Associez l'espace de noms `esegece.sgcNavigation` et déclarez un **TsgcWpfNavigationBar**. **Mode**, **AnimateCollapse**, **AllowMultiExpand** et les autres propriétés simples peuvent être définies comme attributs XAML.
- Dans le code-behind, affectez **Images**, une liste de **ImageSource**, ajoutez les éléments et affectez les événements : ce sont des délégués, pas des événements WPF, ils ne peuvent donc pas être définis comme attributs XAML.

Exemple

MainWindow.xaml déclare la barre, un bouton Collapse et un texte qui affiche la page sélectionnée. Le App.xaml du projet est celui du modèle WPF, avec StartupUri défini sur MainWindow.xaml.
MainWindow.xaml.cs construit trois icônes vectorielles **DrawingImage**, ajoute les éléments et affecte les événements.
Images accepte n'importe quelle **ImageSource**, aussi bien une **BitmapImage** qu'une **DrawingImage**. Lorsque la liste est une **ObservableCollection<ImageSource>**, la barre se redessine lorsqu'une image est ajoutée, supprimée ou remplacée.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- Différences de .NET avec Delphi
- Démarrage rapide WinForms
- Démarrage rapide Avalonia
- Images et glyphes

Démarrage rapide Avalonia

TsgcAvaloniaNavigationBar est la barre de navigation Avalonia, la contrepartie du TsgcFMXNavigationBar de Delphi, sous Windows, Linux et macOS. La souris, le tactile et le stylet passent par le même chemin de pointeur. Le package est **esegece.sgcNavigation.Avalonia**.

Étapes

- Créez un projet .NET 8 et ajoutez les packages esegece.sgcNavigation.Avalonia, Avalonia.Desktop et Avalonia.Themes.Fluent.
- Associez l'espace de noms avec `xmlns:sgc="using:esegece.sgcNavigation"` et déclarez un **TsgcAvaloniaNavigationBar** avec son **Mode**.
- Dans le code-behind, définissez le **Theme**, ajoutez les groupes et les éléments, et affectez les événements : ce sont des délégués, pas des événements Avalonia.

Exemple

MainWindow.axaml déclare un accordéon dans lequel un seul groupe est ouvert à la fois.
MainWindow.axaml.cs définit le thème sombre, ajoute trois groupes avec leurs éléments et affiche l'élément cliqué.
Program.cs démarre l'application avec le thème Fluent.
Theme masque la propriété `StyledElement.Theme` des contrôles Avalonia, définissez-le donc dans le code sur une variable de type TsgcAvaloniaNavigationBar, et non comme attribut XAML. Images est une liste de **IImage**, un **Bitmap** ou une **DrawingImage**.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- Différences de .NET avec Delphi
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Mode accordéon

Classes .NET

Chaque hôte a une classe de barre de navigation avec les mêmes membres. Les types indépendants du framework sont partagés par les trois hôtes et gardent leurs noms Delphi.

Classes d'hôte

Hôte	Classe	Classe de base	Delphi
WinForms	TsgcWinFormsNavigationBar	TsgcUIWinFormsControl	TsgcVCLNavigationBar
WPF	TsgcWpfNavigationBar	TsgcUIWpfControl	TsgcFMXNavigationBar
Avalonia	TsgcAvaloniaNavigationBar	TsgcUIAvaloniaControl	TsgcFMXNavigationBar

Types indépendants du framework

Type	Description
TsgcNavigationModel	Les éléments, le mode, le rail replié, les éléments sélectionné, survolé, enfoncé et ayant le focus, la disposition des lignes et le test de positionnement. Chaque barre en possède un, voir TsgcNavigationModel.
TsgcNavigationItems	La collection d'éléments : Count, l'indexeur, Add, AddItem(aCaption, aGlyph = "", aIsGroup = false), Insert, Delete, Clear, IndexOf, BeginUpdate et EndUpdate.
TsgcNavigationItem	Un élément : Caption, Glyph, ImageIndex, Badge, IsGroup, Expanded, Enabled, Visible, Hint, Tag (long), OnClick, Click() et, dans l'édition Registered, Command et CommandParameter. Voir TsgcNavigationItem.
TsgcNavigationMode	nbmSidebar, nbmAccordion.
TsgcNavigationPart	npNone, npItem, npExpander : la partie d'une ligne renvoyée par ItemAtPos.
TsgcNavigationItemEvent	`void (object Sender, TsgcNavigationItem aItem)`, le type de OnItemClick.
TsgcUINotifyEvent	`void (object Sender)`, de esegece.sgcUICore.
sgcNavigation_Render, sgcNavigation_Rendering	Le moteur de rendu et la sortie headless en SVG et PNG, voir Rendu sans interface en .NET.

Barre de navigation

Membre	Type	Description
Model	TsgcNavigationModel	Le modèle de la barre.
Items	TsgcNavigationItems	Les éléments et les en-têtes de groupe, dans l'ordre.
Mode	TsgcNavigationMode	Sidebar ou accordéon.

Collapsed	bool	Replie la barre en rail d'icônes. La largeur passe de ExpandedWidth à CollapsedWidth, avec animation lorsque AnimateCollapse vaut true.
ExpandedWidth, CollapsedWidth	int	La largeur de la barre développée et du rail d'icônes, 220 et 56 par défaut.
ItemHeight, GroupHeight, Indent	int	La hauteur d'une ligne d'élément, d'une ligne d'en-tête de groupe, et le retrait des éléments d'un groupe en mode accordéon.
AllowMultiExpand	bool	En mode accordéon, false ferme les autres groupes lorsqu'un groupe s'ouvre.
SelectedIndex	int	L'élément sélectionné, ou -1.
Theme, AccentColor	TsgcUITheme, uint	Le thème de la barre. AccentColor 0 utilise la couleur d'accentuation du thème.
Images	ImageList, IList<ImageSource>, IList<IImage>	Les images de ImageIndex, voir Différences de .NET avec Delphi.
AnimateCollapse	bool	Anime Collapsed sur environ 150 ms.
ScrollY	int	Le décalage de défilement vertical en pixels de l'appareil.
ItemAtPos(X, Y), ItemAtPos(X, Y, out aPart)	int	L'index de l'élément à un point du contrôle, ou -1, et la partie de la ligne.
ItemRect(aIndex)	Rectangle, Rect	La ligne d'un élément dans les coordonnées du contrôle.
ScrollIntoView(aIndex)	void	Fait défiler la barre pour que la ligne de l'élément soit visible.
OnItemClick	TsgcNavigationItemEvent	Un clic, Entrée ou Espace sur un élément activé, avant que son OnClick ou son Command ne s'exécute. Un en-tête de groupe ne le déclenche pas.
OnSelectionChange	TsgcUINotifyEvent	SelectedIndex a changé, par l'utilisateur ou dans le code.
OnCollapsedChange	TsgcUINotifyEvent	Collapsed a changé.

Les positions et les tailles sont des pixels dans WinForms (Rectangle) et des unités indépendantes de l'appareil dans WPF et Avalonia (Rect).

Voir aussi

- Vue d'ensemble de .NET
- Différences de .NET avec Delphi
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia

Différences de .NET avec Delphi

L'API .NET suit l'API Delphi nom pour nom. Voici les endroits où elle diffère.

Command au lieu des liens d'action

Delphi relie un élément à un TBasicAction par sa propriété Action, voir Liens d'action. .NET n'a pas de TBasicAction, l'élément a donc **Command**, un `System.Windows.Input.ICommand`, et **CommandParameter**. Les deux font partie de l'édition Registered uniquement, comme les liens d'action de Delphi.

- `CanExecute(CommandParameter)` pilote le **Enabled** de l'élément : lorsque la commande ou le paramètre est affecté et à chaque `CanExecuteChanged`.
- Un clic, Entrée ou Espace déclenche OnItemClick de la barre puis exécute la commande, lorsqu'elle peut s'exécuter. Lorsque l'élément a un gestionnaire OnClick, OnClick s'exécute à la place de la commande, comme dans Delphi.
- Caption, Hint et ImageIndex ne sont pas synchronisés depuis la commande, car ICommand n'en porte aucun. Ils restent des propriétés de l'élément. Visible n'est pas synchronisé non plus.
- Il n'y a pas de OnUpdate au repos. Une commande qui change d'état déclenche CanExecuteChanged, et l'élément la suit immédiatement.

Un programme console qui relie un élément à une commande minimale, l'exécute et la désactive par CanExecuteChanged. Il n'a besoin que de esegece.sgcNavigation, les barres des trois hôtes se comportent de la même façon.

Images

Delphi utilise une TCustomImageList sous VCL et une TCustomImageList de FMX.ImgList sous FireMonkey, voir Images et glyphes. En .NET, la propriété **Images** a le type d'image de chaque hôte, et le **ImageIndex** de l'élément est un index dans celle-ci.

Hôte	Images	Dessinée à
WinForms	System.Windows.Forms.ImageList	Le ImageSize de la liste d'images, en pixels.
WPF	IList<ImageSource> : bitmaps et icônes vectorielles DrawingImage. Une ObservableCollection se redessine lorsqu'elle change.	16 x 16 unités indépendantes de l'appareil, nette à tout DPI pour les icônes vectorielles.
Avalonia	IList<IImage> : un Bitmap ou une DrawingImage. Une liste avec INotifyCollectionChanged se redessine lorsqu'elle change.	16 x 16 unités indépendantes de l'appareil.

Chaque image est convertie une fois en ARGB et mise en cache jusqu'à ce que la liste change et, dans WPF et Avalonia, aussi jusqu'à ce que l'échelle DPI change. Lorsqu'un élément n'a pas d'image, le texte de son **Glyph** est dessiné dans l'emplacement de l'icône.

Info-bulles et tooltips

Dans Delphi, la barre affiche le Hint de l'élément sous le curseur lorsque ShowHint vaut True, et FMX affiche les info-bulles à partir de 10.1 Berlin. En .NET, la barre affiche toujours un tooltip, sans propriété ShowHint : le Hint de l'élément sous le curseur, ou son Caption dans le rail replié où le libellé n'est pas visible. WinForms utilise un composant ToolTip interne, WPF définit la propriété ToolTip de la barre et Avalonia définit la propriété attachée ToolTip.Tip. Ne définissez pas vous-même le tooltip de la barre, la barre le remplace ; définissez plutôt le Hint de l'élément.

Unités

- WinForms : ExpandedWidth, CollapsedWidth, ItemHeight, GroupHeight et Indent sont des pixels logiques multipliés par DPI / 96, comme la barre VCL avec PPI / 96. ItemAtPos, ItemRect et ScrollY sont des pixels client.
- WPF et Avalonia : Width, ExpandedWidth, CollapsedWidth, ItemAtPos et ItemRect sont des unités indépendantes de l'appareil, comme dans FMX. Le cadre, la disposition du modèle et **ScrollY** sont des pixels de l'appareil, les unités multipliées par l'échelle DPI du contrôle (WPF) ou l'échelle de rendu de la fenêtre (Avalonia). Le BoundsRect d'un élément est aussi en pixels de l'appareil.
- Headless : aWidth et aHeight sont des pixels logiques et aScale les multiplie, voir Rendu sans interface en .NET.

Événements

OnItemClick, OnSelectionChange, OnCollapsedChange et le OnClick de l'élément sont des propriétés délégué avec les noms Delphi, pas des membres `event` C#. Affecter un gestionnaire remplace le précédent, comme un pointeur de méthode Delphi, et ils sont affectés dans le code, y compris dans WPF et Avalonia où XAML ne peut pas les définir.

Pas de verbes de concepteur

Dans Delphi, la propriété Items est modifiée à la conception avec l'éditeur de collection standard. Le portage .NET n'a ni verbes de concepteur ni éditeur de collection : les Items d'une barre ne sont pas sérialisés, ils sont donc toujours ajoutés dans le code, généralement dans le constructeur du formulaire ou de la fenêtre. Dans WinForms, la barre figure dans la boîte à outils et ses propriétés simples se définissent dans la fenêtre Propriétés. Dans WPF et Avalonia, la barre est déclarée en XAML.

Éditions

Le portage .NET est livré en packages Registered et Community, voir Vue d'ensemble de .NET. Les packages Community n'ont ni Command ni CommandParameter. Il n'y a pas d'édition Trial.

Liaison de données

Sous WPF et Avalonia, les propriétés visibles par l'utilisateur sont des propriétés de dépendance et des propriétés Avalonia, si bien qu'elles peuvent être liées, stylées et animées, voir Liaison de données .NET.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Liens d'action
- Images et glyphes

Liaison de données .NET

Dans les hôtes WPF et Avalonia, les propriétés visibles par l'utilisateur de la barre de navigation sont des propriétés de dépendance (WPF) et des propriétés Avalonia. Elles peuvent donc être liées à un modèle de vue, définies par un Style et animées. Cette rubrique s'applique uniquement à WPF et Avalonia. Le contrôle WinForms ne change pas : ses propriétés sont de simples propriétés CLR, comme avant.

Propriétés pouvant être liées

Chaque propriété du tableau possède un champ statique avec le suffixe Property, par exemple `TsgcWpfNavigationBar.SelectedIndexProperty`. Sous WPF, les champs sont de type DependencyProperty. Sous Avalonia, ce sont des StyledProperty, sauf SelectedIndex, qui est une DirectProperty. Les propriétés marquées **TwoWay** sont liées en TwoWay par défaut, car la barre les modifie aussi (SelectedIndex après un clic ou une touche, les deux après une modification du modèle), de sorte qu'une liaison sans Mode maintient le modèle de vue à jour. Les autres propriétés sont liées en OneWay par défaut.

Classe WPF	Classe Avalonia	Propriétés pouvant être liées
TsgcWpfNavigationBar	TsgcAvaloniaNavigationBar	Mode, **Collapsed (TwoWay)**, ExpandedWidth, CollapsedWidth, ItemHeight, GroupHeight, Indent, AllowMultiExpand, **SelectedIndex (TwoWay)**, Theme, AccentColor, AnimateCollapse

Sous Avalonia, Theme est le thème sgcUICore (TsgcUITheme) et masque le thème de contrôle Avalonia de StyledElement, qui reste accessible via `StyledElement.ThemeProperty`.

Comportement

- **Le modèle conserve les valeurs.** Chaque propriété est stockée dans le modèle de navigation. Une valeur liée est transmise au modèle, et la propriété reprend la valeur que le modèle conserve : le modèle rejette une largeur ou une hauteur inférieure à 1 et maintient SelectedIndex dans les éléments, puis la propriété affiche la valeur utilisée. Une source TwoWay reçoit aussi la valeur corrigée, sous WPF et sous Avalonia. Sous WPF, elle arrive une fois que le dispatcher a exécuté le travail en attente. Sous Avalonia, la réécriture couvre une valeur que la source envoie lorsque la liaison est déjà active. Une valeur que la source détient déjà au démarrage de la liaison est corrigée dans le contrôle, mais n'est pas réécrite dans la source.
- **Les modifications faites par la barre atteignent le modèle de vue.** Un clic sur un élément, une touche, une modification du modèle et une modification depuis le code mettent à jour SelectedIndex, et une modification du modèle ou depuis le code met à jour Collapsed. Avec la liaison TwoWay par défaut, le modèle de vue suit.
- **Les événements se déclenchent pour toute source.** OnSelectionChange et OnCollapsedChange se déclenchent pour une modification par saisie, par le code ou par une liaison.
- **D'abord les éléments, ensuite la liaison.** SelectedIndex fait référence à un élément, ajoutez donc les éléments avant l'arrivée de la valeur liée, par exemple en définissant le DataContext après les éléments, comme le font les exemples.
- **Styles.** Un setter de Style peut définir n'importe quelle propriété pouvant être liée, par exemple Mode et Theme de toutes les barres de navigation d'une fenêtre.

Propriétés qui restent des propriétés CLR

Propriété	Raison
Items	L'arborescence des éléments du modèle, construite avec AddItem et les autres méthodes de la collection, et non une valeur unique à lier.

Images	La liste d'images des éléments, un objet partagé par référence.
Model	Le modèle indépendant de la plateforme derrière le contrôle, en lecture seule.
ScrollY	La position de défilement, qui change à chaque cran de la molette et n'est pas un paramètre.
OnSelectionChange, OnCollapsedChange et les autres événements	Propriétés de type délégué portant les noms Delphi, affectées dans le code, voir Différences de .NET par rapport à Delphi.

Exemple

Le modèle de vue ci-dessous est partagé par les deux exemples. SelectedIndex et Collapsed sont liés sans Mode, ils sont donc liés en TwoWay : un clic sur un élément modifie le modèle de vue, et la case à cocher réduit la barre via le modèle de vue.

WPF

MainWindow.xaml lie la barre. App.xaml est celui du modèle WPF.
MainWindow.xaml.cs ajoute les éléments puis définit le modèle de vue comme DataContext.

Avalonia

MainWindow.axaml définit `x:DataType` sur le modèle de vue, si bien que les liaisons sont des liaisons compilées. La classe App est celle de Démarrage rapide Avalonia.
MainWindow.axaml.cs ajoute les éléments puis définit le modèle de vue comme DataContext.

Voir aussi

- Différences de .NET par rapport à Delphi
- Classes .NET
- Démarrage rapide WPF
- Démarrage rapide Avalonia

Rendu sans interface en .NET

Une barre de navigation peut être dessinée sans fenêtre, sur un serveur, dans un test ou dans un rapport. **sgcNavigation_Render.sgcNavigationSaveToSVG** écrit un fichier ou un flux SVG avec le seul package esegece.sgcNavigation, et **sgcNavigation_Rendering.sgcNavigationSaveToPNG** écrit un PNG avec esegece.sgcNavigation.Rendering, qui dessine sur SkiaSharp sous Windows, Linux et macOS. La sortie SVG est identique octet pour octet à celle du rendu sans interface de Delphi.

Méthodes

Méthode	Package	Description
sgcNavigation_Render.sgcNavigationSaveToSVG(aModel, aFileName or aStream, aWidth, aHeight, aScale = 1, aImages = null)	esegece.sgcNavigation	Écrit la barre de aModel en SVG.
sgcNavigation_Rendering.sgcNavigationSaveToPNG(aModel, aFileName or aStream, aWidth, aHeight, aScale = 1, aImages = null)	esegece.sgcNavigation.Rendering	Écrit la barre de aModel en PNG sur fond transparent.
sgcNavigation_Render.sgcNavigationRender(aModel, aCanvas, aWidth, aHeight, aScale = 1, aScrollY = 0, aImages = null)	esegece.sgcNavigation	Dessine la barre sur n'importe quel canevas sgcUICore.

aWidth et aHeight sont des pixels logiques. L'image fait aWidth x aScale par aHeight x aScale pixels, de sorte qu'une échelle de 2 donne une image nette pour un écran haute résolution ou une impression. aImages est un **TsgcNavigationImageEvent** facultatif, `bool (int aImageIndex, out TsgcUIARGBBuffer aImage)`, qui renvoie l'image d'un ImageIndex sous forme de **TsgcUIARGBBuffer** ; sans lui, le Glyph de l'élément est dessiné dans l'emplacement de l'icône.

Exemple

Un programme console (`net8.0`) qui référence esegece.sgcNavigation.Rendering et écrit en PNG et SVG une sidebar avec un badge, un en-tête de groupe et un élément désactivé, le rail d'icônes replié, et les mêmes éléments sous forme d'accordéon sombre avec un groupe fermé.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- Rendu sans interface
- TsgcNavigationModel

Démos .NET

Les démos .NET se trouvent dans le dossier `demos` du produit et dans le zip des démos, avec la solution `sgcNavigation_Demos.sln`. Chaque démo a un README.md.

Démo	Hôte	Description
SidebarApp	WinForms	Une sidebar à gauche change les pages d'une application. Les cinq éléments (Dashboard, Inbox, Reports, Billing et Settings) prennent leurs icônes dans une ImageList, Inbox et Reports affichent un badge et Billing est désactivé. Le bouton Collapse replie la barre en rail d'icônes avec une animation. OnSelectionChange affiche la page stockée dans le Tag de l'élément sélectionné, OnItemClick écrit l'élément cliqué dans la ligne d'état et OnCollapsedChange met à jour le libellé du bouton. La première page héberge un tableau de bord sgcDashboard avec deux cartes KPI en direct, et la liste Theme fait passer la barre, le tableau de bord et les pages du clair au sombre.
SidebarAppWpf	WPF	La même application sur un TsgcWpfNavigationBar avec des icônes vectorielles DrawingImage et un TsgcWpfDashboard déclaré en XAML.
SidebarAppAvalonia	Avalonia	La même application sur un TsgcAvaloniaNavigationBar avec des icônes vectorielles IImage, sous Windows, Linux et macOS.
Accordion	WinForms	Un accordéon avec les groupes Sales, Inventory et Admin. Avec "Allow multi expand" décoché, l'ouverture d'un groupe ferme les autres. New order, Export et Audit log sont reliés à des commandes par le Command de l'élément : un clic exécute la commande et l'élément Export suit CanExecute, que modifie la case à cocher "Export command enabled". Le bouton Rename renomme l'élément lui-même, car ICommand ne porte pas de libellé. Dans l'édition Community, les éléments s'exécutent par leur OnClick.
AccordionWpf	WPF	Le même accordéon sur un TsgcWpfNavigationBar.
AccordionAvalonia	Avalonia	Le même accordéon sur un TsgcAvaloniaNavigationBar.

Compiler et exécuter

Les démos nécessitent le SDK .NET 8 ou ultérieur. Exécutez une démo depuis son dossier :

```
dotnet run
```

Dans l'arborescence du produit, les démos se compilent contre les projets de la bibliothèque. Depuis le zip des démos, elles restaurent les packages NuGet, l'édition Community par défaut. Définissez **SgcNavigationEdition** pour utiliser les packages Registered :

```
dotnet build -p:SgcNavigationEdition=Registered
```

Auto-vérification

Chaque démo a un commutateur `--test` qui ouvre la fenêtre hors écran, clique sur les éléments, ouvre et ferme les groupes, replie la barre et change de thème, écrit une ligne PASS ou FAIL par vérification dans le fichier et se termine avec le nombre de vérifications en échec, de sorte que 0 signifie que toutes ont réussi.

```
dotnet run -- --test result.txt
```

Voir aussi

- Vue d'ensemble de .NET
- Installation depuis NuGet
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Démos

Dépannage

Undeclared identifier Action, ou "Property Action does not exist"

L'édition Community se compile sans SGC_NAVIGATION_ACTIONS, la propriété Action des éléments n'existe donc pas. Le code qui l'utilise échoue avec E2003 Undeclared identifier, et une fiche enregistrée par une compilation Registered avec un élément lié à une action déclenche une exception "Error reading" au chargement. Utilisez l'OnClick de l'élément ou l'événement OnItemClick de la barre, ou une compilation Registered ou Trial. Voir Éditions.

Un clic sur un en-tête de groupe ne le sélectionne pas

Les en-têtes de groupe ne sont jamais sélectionnés. En mode sidebar, ce sont des libellés statiques, et dans l'accordéon déplié un clic bascule le groupe. Utilisez un élément avec IsGroup à False pour une page. Voir Mode accordéon.

Un élément ne réagit pas à un clic

Un élément désactivé (Enabled à False) est dessiné atténué et ignore les clics et le clavier. Un élément doté d'une Action suit le Enabled de l'action, vérifiez donc aussi l'OnUpdate de l'action.

Les images ne s'affichent pas sous FireMonkey

La propriété Images de la barre FMX nécessite Delphi XE8 ou une version ultérieure. Sur les versions plus anciennes, les éléments dessinent leur Glyph. Voir Images et glyphes.

Pas d'info-bulles sur la barre FMX

Les info-bulles des éléments de la barre FMX nécessitent Delphi 10.1 Berlin ou une version ultérieure, et ShowHint doit être à True.

La barre passe à sa nouvelle largeur au lieu de s'animer

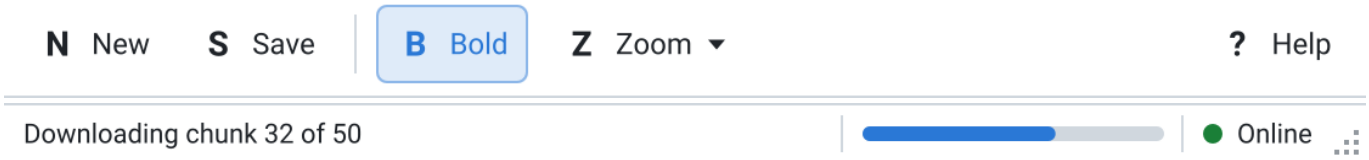
L'animation de réduction ne s'exécute qu'à l'exécution et seulement tant que AnimateCollapse vaut True. À la conception, la largeur change immédiatement.

La démo SidebarApp, SidebarFMX ou Accordion ne se compile pas

SidebarApp et SidebarFMX hébergent un dashboard et nécessitent que sgcDashboard soit installé. Accordion utilise les liens d'action (Action links), qui ne font pas partie de l'édition Community. Voir Démos.

Introduction

sgcToolbar est une barre d'outils et une barre d'état natives pour Delphi et C++Builder, pour VCL et FireMonkey. **TsgcVCLToolbar** et **TsgcFMXToolbar** affichent une rangée de boutons, de boutons à bascule, de groupes d'options, de boutons déroulants, de libellés, de séparateurs et d'espaceurs, et déplacent les éléments qui ne tiennent pas dans un menu de débordement. **TsgcVCLStatusBar** et **TsgcFMXStatusBar** affichent des panneaux de texte, des barres de progression et des indicateurs d'état colorés, et un thread de travail peut les mettre à jour directement grâce à des setters thread-safe.
sgcToolbar fait partie de la suite visuelle eSeGeCe et donne aux autres produits leur cadre : une barre d'outils en haut et une barre d'état en bas d'une fenêtre qui héberge un tableau de bord **sgcDashboard** , une grille **sgcGrid** ou une barre de navigation **sgcNavigation** . Elle est dessinée avec **sgcUICore**, le package d'exécution partagé avec le canvas, le rastériseur, les polices, l'écrivain SVG et les thèmes clair et sombre, de sorte que les barres ont le même aspect sous VCL, sous FireMonkey et dans la sortie PNG et SVG sans interface. sgcToolbar nécessite sgcUICore, qui s'installe avec lui.



Éditions

sgcToolbar est livré en éditions Registered, Trial et Community. L'édition Community a toutes les fonctionnalités sauf le menu de débordement et les liens d'action des éléments. Voir Éditions.

Par où commencer

- Architecture: les modèles indépendants du framework, le moteur de rendu sur le canvas sgcUICore, et les contrôles VCL et FMX.
- Plateformes: les versions de Delphi, C++Builder et les plateformes.
- Démarrage rapide VCL, Démarrage rapide FMX et Démarrage rapide C++Builder: une première barre d'outils et barre d'état en quelques minutes.
- Composants: les barres d'outils, les barres d'état, les éléments, les panneaux et les modèles.
- Types d'éléments, Boutons à bascule et groupes d'options, Menus déroulants, Débordement, Liens d'action, Images et glyphes et Clavier.
- Panneaux d'état, Progression et indicateurs, Mises à jour thread-safe et Point de redimensionnement.
- Thèmes et Rendu sans interface.
- Démos: les projets de démonstration Delphi et C++Builder.

Architecture

sgcToolbar est construit en couches. Chaque couche ne dépend que de celles qui sont en dessous, donc les modèles et le moteur de rendu se compilent sans VCL ni FireMonkey.

Couche	Unit	Rôle
Modèles	sgcToolbar_Model	TsgcToolbarModel contient les Items (une collection de TsgcToolbarItem), ShowCaptions, les tailles, le thème et l'état d'interaction (élément survolé, enfoncé et ciblé). TsgcStatusBarModel contient les Panels (une collection de TsgcStatusBarPanel), le thème et le point de redimensionnement. Leurs méthodes Layout calculent le rectangle de chaque élément ou panneau, et HitTest trouve celui qui se trouve sous un point.
Moteur de rendu	sgcToolbar_Render	RenderToolbar et RenderStatusBar dessinent un modèle sur n'importe quel canvas sgcUICore. Les méthodes SaveToPNG et SaveToSVG des modèles les utilisent pour écrire une image sans fenêtre. Voir Rendu sans interface.
Contrôles VCL	sgcToolbar_VCL	TsgcVCLToolbar et TsgcVCLStatusBar possèdent un modèle, publient ses propriétés et le dessinent par la base de contrôle VCL de sgcUICore. Ils ajoutent la souris, le clavier, les info-bulles, la liste d'images, les menus contextuels et le point de redimensionnement.
Contrôles FMX	sgcToolbar_FMX	TsgcFMXToolbar et TsgcFMXStatusBar sont les contrôles FireMonkey avec les mêmes propriétés, méthodes et événements, plus la saisie tactile.
sgcUICore	sgcUICore_Canvas, sgcUICore_VCL, sgcUICore_FMX, sgcUICore_Theme	Le runtime partagé : le canvas vectoriel et le rastériseur, la base de contrôle VCL et FMX avec cadence d'images (MaxFPS), et les jetons des thèmes clair et sombre.

Modèle et contrôle

Les propriétés publiées d'une barre sont les propriétés de son modèle, et la propriété **Model** de la barre donne accès au reste : Layout, HitTest, la liste de débordement et l'état d'interaction. Une modification du modèle planifie une nouvelle image, elle ne dessine jamais directement, donc de nombreuses modifications à la suite ne coûtent qu'un seul repaint.

Dessin

Chaque élément et chaque panneau est dessiné par le moteur de rendu : l'emplacement de l'icône avec une image ou un glyphe, la légende, la flèche déroulante, les états survolé, enfoncé, coché, ciblé et désactivé, les séparateurs, le bouton de débordement, les barres de progression, les points d'indicateur et le point de redimensionnement. Toutes les tailles sont des pixels logiques multipliés par l'échelle du moniteur, donc les barres sont nettes quel que soit le DPI. Les contrôles VCL utilisent PPI / 96 comme échelle et les contrôles FMX l'échelle de la fiche.

Height

La hauteur d'une barre d'outils suit ses boutons : ButtonSize plus deux fois Padding, en pixels logiques, et toute autre hauteur est ignorée. Une barre d'état conserve la Height que vous lui donnez, 24 pixels logiques par défaut.

Voir aussi

- Plateformes
- TsgcToolbarModel
- TsgcStatusBarModel
- Thèmes

Éditions

sgcToolbar est construit à partir des mêmes sources pour chaque édition. L'édition ne change que les fonctionnalités qu'une compilation inclut.

Édition	Contient
Registered	Tout : les barres d'outils et barres d'état VCL et FMX, tous les types d'éléments, les boutons à bascule et groupes d'options, les menus déroulants, le menu de débordement, les liens d'action des éléments, les images et glyphes, la navigation au clavier, les panneaux de texte, de progression et d'indicateur, les setters thread-safe, le point de redimensionnement, les thèmes et la sortie PNG et SVG sans interface.
Trial	Les mêmes fonctionnalités que Registered. Une compilation avec SGC_TRIAL affiche un avis indiquant qu'il s'agit d'une version d'évaluation une fois la fenêtre de publication indiquée dans la compilation dépassée. Rien d'autre ne change.
Community	Tout sauf le menu de débordement et les liens d'action des éléments.

Ce que l'édition Community omet

L'édition Community se compile avec SGC_TOOLBAR_ADVANCED non défini. Dans une compilation Community :

- La barre d'outils n'a pas de bouton de débordement. Les éléments qui ne tiennent pas sont tronqués au bord droit, et la propriété **OverflowMenu** des barres d'outils n'existe pas. Voir Débordement.
- La propriété **Action** de TsgcToolbarItem n'existe pas, donc un élément ne peut pas être lié à une TAction. Affectez plutôt à l'élément **OnClick**, ou l'événement **OnItemClick** de la barre d'outils. Voir Liens d'action.

L'édition est un choix à la compilation dans sgcVer.inc (SGC_COMMUNITY, qui désactive aussi SGC_TRIAL), et non un interrupteur d'exécution. Le code ou le fichier de fiche qui utilise Action ou OverflowMenu ne se compile ou ne se charge pas avec une compilation Community, voir Dépannage.

Voir aussi

- Débordement
- Liens d'action
- Plateformes

Installation

sgcToolbar fait partie de la distribution sgcUI. Un seul installateur couvre les produits sgcUI (sgcUICore, sgcCharts, sgcPDF, sgcReports, sgcMaps, sgcDashboard, sgcGrid, sgcNavigation, sgcToolbar et sgcWizard) et n'installe que les produits inclus dans votre licence.

Installation avec l'installateur sgcUI

1. Lancez l'installateur sgcUI et connectez-vous avec le nom d'utilisateur et le mot de passe de votre compte eSeGeCe.
2. Choisissez les versions de Delphi et de C++Builder dans lesquelles installer.
3. Sur la page des composants, sélectionnez **sgcToolbar** et tout autre produit souhaité. Le serveur de licences renvoie les produits que vous possédez, et l'installateur affiche une case à cocher par produit. Les produits que votre licence n'inclut pas sont grisés. Une licence All-Access déverrouille tous les produits.
4. L'installateur copie les fichiers, puis compile les packages sur votre machine pour chaque IDE sélectionné, dans l'ordre des dépendances, et installe les packages de conception.

Ce qui est fourni avec sgcToolbar

sgcUICore, la base commune des produits de mise en page et de sgcCharts, est fourni avec sgcToolbar. Il n'a pas de case à cocher propre. L'installateur l'installe automatiquement dès que sgcToolbar est sélectionné. sgcToolbar n'a pas de package de passerelle sgcWebSockets.

Dossiers installés

Tous les produits s'installent dans un seul dossier, par défaut `Documents\esegece\sgcUI` (`sgcUI_Trial` pour la version d'essai et `sgcUI_Community` pour l'édition Community).

Dossier	Contenu
`Source`	Le code source de chaque produit installé. C'est l'unique chemin de bibliothèque de sgcUI.
`Packages`	Les packages de chaque produit installé, un jeu par produit et par version d'IDE.
`Lib<V>`	Les unités compilées pour chaque version d'IDE et chaque plateforme, par exemple `LibD13` .
`Demos\Toolbar`	Les démos de sgcToolbar. Voir Démos.

L'installateur ajoute `sgcUI\Source` au chemin de bibliothèque et au chemin de navigation de chaque IDE et plateforme sélectionnés, ainsi que le dossier des unités compilées. sgcUICore se trouve dans le même dossier `Source`, il n'a donc besoin d'aucun chemin propre.

Trial et Community

- **Trial** : l'installateur Trial de sgcUI installe tous les produits en mode d'essai, y compris sgcToolbar.
- **Community** : l'installateur Community gratuit de sgcUI inclut sgcToolbar. Ses produits sont sgcCharts, sgcDashboard, sgcGrid, sgcNavigation, sgcToolbar et sgcWizard, tandis que sgcPDF et sgcMaps ne font pas partie de Community. Voir Éditions pour ce que la version Community de sgcToolbar laisse de côté.

Installations autonomes précédentes

Si l'installateur trouve une ancienne installation autonome de sgcUICore, comme celle créée par l'installateur autonome de sgcCharts, il liste les chemins de bibliothèque et les packages de conception de cette installation sur la page **Previous installations**. Quand vous continuez, il les supprime de l'IDE avant de compiler les nouveaux packages, de sorte qu'il n'y ait jamais deux copies d'une unité dans le chemin.

Compiler les packages manuellement

Pour compiler vous-même les packages, ouvrez-les pour votre version d'IDE depuis `sgcUI\Packages`. `<V>` est le suffixe de version Delphi, par exemple `sgcToolbarD13.dpk`.

Package	Contenu
`sgcUICore<V>`	La base commune sgcUICore (exécution).
`sgcToolbar<V>` / `dclsgcToolbar<V>`	Les composants sgcToolbar (exécution / conception).

Compilez-les dans l'ordre des dépendances : `sgcUICore`, puis `sgcToolbar`, puis `dclsgcToolbar`. Installez ensuite le package de conception `dclsgcToolbar` et ajoutez `sgcUI\Source` au chemin de bibliothèque de votre version de Delphi (Tools > Options > Language > Delphi Options > Library > Library path). Pour C++Builder, ouvrez le projet `.cbproj` correspondant à votre version, par exemple `sgcToolbarC13.cbproj`, et compilez-le. Les en-têtes `.hpp` générés apparaissent à côté du package compilé.

.NET

Rien ne change pour .NET. sgcToolbar reste distribué sous forme de ses propres packages NuGet (`esegece.sgcToolbar` et ses packages compagnons). Voir l'installation .NET.

Plateformes

Cible	Prise en charge
Delphi	Delphi 7, 2007, 2009, 2010, XE à XE8, 10 à 10.4, 11, 12 et 13.
C++Builder	C++Builder 2007, 2010, XE à XE8, 10 à 10.4, 11, 12 et 13. Les packages génèrent les en-têtes .hpp et les fichiers .lib qu'un projet C++Builder inclut et lie.
VCL	TsgcVCLToolbar et TsgcVCLStatusBar sous Windows, Win32 et Win64, toutes les versions de Delphi.
FireMonkey	TsgcFMXToolbar et TsgcFMXStatusBar à partir de Delphi XE6 (SGC_TOOLBAR_FMX), sous Win32, Win64, macOS, iOS et Android, et sous Linux avec FMX pour Linux (FMXLinux). La propriété Images nécessite XE8 ou ultérieur (FMX.ImgList), et les info-bulles nécessitent 10.1 Berlin ou ultérieur.
Linux64	Les unités indépendantes du framework, les modèles et le moteur de rendu avec la sortie PNG et SVG sans interface, à partir de Delphi 10.2 (SGC_TOOLBAR_LINUX).
.NET	Un portage en C# avec la même API : TsgcWinFormsToolbar et TsgcWinFormsStatusBar, TsgcWpfToolbar et TsgcWpfStatusBar, TsgcAvaloniaToolbar et TsgcAvaloniaStatusBar, et la sortie sans interface en PNG et SVG, sur .NET Framework 4.7.2 et .NET 8 et 9, voir Présentation de .NET.

Packages

sgcToolbar fournit un package d'exécution indépendant du framework et un package d'exécution par framework, de sorte qu'une application FireMonkey pour Android, iOS ou macOS ne lie jamais la VCL. Chaque nom de package se termine par le suffixe de l'IDE, par exemple sgcToolbarD13.dpk pour Delphi 13 et sgcToolbarC13.cbproj pour C++Builder 13.

Package	Type	Contenu	Requiert
sgcToolbar	Exécution, indépendant du framework	Chaque unité du produit qui n'utilise ni la VCL ni FireMonkey.	rtl, sgcUICore
sgcToolbarVCL	Exécution, VCL	sgcToolbar_VCL (TsgcVCLToolbar, TsgcVCLStatusBar)	rtl, vcl, sgcUICore, sgcUICoreVCL, sgcToolbar
sgcToolbarFMX	Exécution, FireMonkey, à partir de XE6	sgcToolbar_FMX (TsgcFMXToolbar, TsgcFMXStatusBar)	rtl, fmx, sgcUICore, sgcUICoreFMX, sgcToolbar
dclsgcToolbar	Conception, VCL	Enregistre TsgcVCLToolbar et TsgcVCLStatusBar dans la page de palette **SGC Toolbar**.	designide, rtl, vcl, sgcToolbar, sgcToolbarVCL
dclsgcToolbarFMX	Conception, FireMonkey, à partir de XE6	Enregistre TsgcFMXToolbar et TsgcFMXStatusBar dans la page de palette **SGC Toolbar**.	designide, rtl, fmx, sgcToolbar, sgcToolbarFMX, dclsgcToolbar

sgcUICore suit la même répartition : sgcUICore, sgcUICoreVCL et sgcUICoreFMX, sans package de conception. dclsgcToolbarFMX requiert dclsgcToolbar, installez donc d'abord dclsgcToolbar.

Versions de l'IDE et plateformes cibles

Delphi	sgcToolbar	sgcToolbarVCL	sgcToolbarFMX	dclsgcToolbar, dclsgcToolbarFMX
7, 2007, 2009, 2010, XE	Win32	Win32	Aucune	Win32
XE2 à XE5	Win32, Win64	Win32, Win64	Aucune	Win32
XE6 à 10.2	Win32, Win64	Win32, Win64	Win32, Win64	Win32
10.3	Win32, Win64, Linux64, OSX64	Win32, Win64	Win32, Win64, OSX64	Win32
10.4	Win32, Win64, Linux64, OSX64, Android, Android64, iOSDevice64	Win32, Win64	Win32, Win64, OSX64, Android, Android64, iOSDevice64	Win32
11	Win32, Win64, Linux64, OSX64, OSXARM64, Android, Android64, iOSDevice64, iOSSimARM64	Win32, Win64	Win32, Win64, OSX64, OSXARM64, Android, Android64, iOSDevice64, iOSSimARM64	Win32
12, 13	Comme 11	Win32, Win64	Comme 11	Win32, Win64

De XE6 à 10.2, le package FireMonkey est compilé pour Windows uniquement. Les packages de conception sont chargés par l'IDE, ils sont donc compilés pour la plateforme de l'IDE uniquement.

C++Builder	sgcToolbar, sgcToolbarVCL	sgcToolbarFMX	dclsgcToolbar, dclsgcToolbarFMX
2007, 2010, XE, XE2	Win32	Aucune	Win32 (dclsgcToolbar uniquement)
XE3 à XE5	Win32, Win64	Aucune	Win32 (dclsgcToolbar uniquement)
XE6 à 12	Win32, Win64	Win32, Win64	Win32
13	Win32, Win64, Win64x	Win32, Win64, Win64x	Win32, Win64x

Les projets C++Builder portent les mêmes noms avec le suffixe C, par exemple sgcToolbarC13.cbproj, sgcToolbarVCLC13.cbproj, sgcToolbarFMXC13.cbproj, dclsgcToolbarC13.cbproj et dclsgcToolbarFMXC13.cbproj. Les packages C++Builder sont compilés pour Windows uniquement. Les notes C++Builder qui s'appliquent à chaque produit sgcUI (packages d'exécution sous Win64, C++Builder 10.3 sous Win64, noms de fichiers des packages Win64x) se trouvent dans le sujet Plateformes et packages de l'aide de la suite sgcUI.

FireMonkey sur Android, iOS et macOS

Les contrôles FireMonkey (TsgcFMXToolbar et TsgcFMXStatusBar) ciblent macOS à partir de Delphi 10.3, Android et iOS à partir de Delphi 10.4, et macOS ARM et le simulateur iOS à partir de Delphi 11. Dans un projet FireMonkey pour ces plateformes :

- Utilisez les contrôles FireMonkey. Les contrôles VCL et sgcToolbarVCL ne fonctionnent que sous Windows.

- Déposez le contrôle sur un formulaire, ou ajoutez son unité à la clause uses : sgcToolbar_FMX.
- Sélectionnez la plateforme cible dans la fenêtre Projets. L'installateur ajoute le dossier Source de sgcUI et les unités compilées de chaque plateforme (LibD<version>\<platform>) au chemin de bibliothèque de chaque plateforme qu'il compile. Sans l'installateur, ajoutez le dossier Source de sgcUI au chemin de bibliothèque de la plateforme cible (Outils, Options, Langage, Delphi, Bibliothèque).
- Les applications Android et iOS sont liées statiquement, aucun package d'exécution n'est donc déployé.

Linux64

Le package indépendant du framework sgcToolbar est compilé pour Linux64 à partir de Delphi 10.3. Une application Linux64, par exemple un programme console ou un serveur, utilise les unités indépendantes du framework. Aucun package VCL ou FireMonkey n'est compilé pour Linux64, car FireMonkey sous Linux nécessite FMX pour Linux (FMXLinux), un module complémentaire tiers.
Les barres VCL n'apparaissent que dans la palette d'une fiche VCL et les barres FMX que dans la palette d'une fiche FireMonkey. Les propriétés **Items** et **Panels** se modifient avec l'éditeur de collection standard.

Voir aussi

- Éditions
- Architecture
- Dépannage

Démarrage rapide VCL

À la conception

- Déposez un **TsgcVCLToolbar** depuis la **SGC Toolbar** page de la palette sur une fiche. Il est aligné en haut par défaut.
- Double-cliquez sur **Items** dans l'inspecteur d'objets et ajoutez les éléments. Définissez **Caption** et **Glyph** (une lettre ou un court symbole) de chaque bouton, et définissez **Kind** à tikSeparator, tikToggle, tikDrop-down, tikLabel ou tikSpacer là où vous en avez besoin.
- Déposez un **TsgcVCLStatusBar**. Il est aligné en bas par défaut. Double-cliquez sur **Panels** et ajoutez les panneaux, avec **Spring** défini à True sur celui qui prend la largeur libre.
- Écrivez un gestionnaire **OnItemClick** pour la barre d'outils, ou affectez le **OnClick** de chaque élément.
- Définissez **Theme** à utDark sur les deux barres pour la palette sombre.

Dans le code

Les mêmes barres créées à l'exécution. Le programme se compile tel quel avec `sgcUI\Source` dans le chemin de recherche.

```
uses
  Forms, Classes, Controls, StdCtrls,
  sgcUICore_Theme, sgcToolbar_Model, sgcToolbar_VCL;

procedure TMainForm.CreateBars;
var
  oItem: TsgcToolbarItem;
  oPanel: TsgcStatusBarPanel;
begin
  // the toolbar, aligned to the top of the form by default
  Toolbar := TsgcVCLToolbar.Create(Self);
  Toolbar.Parent := Self;
  Toolbar.ShowHint := True;
  Toolbar.OnItemClick := ToolbarItemClick;

  oItem := Toolbar.Items.Add;
  oItem.Caption := 'New';
  oItem.Glyph := 'N';
  oItem.Hint := 'Create a new document';
  oItem := Toolbar.Items.Add;
  oItem.Caption := 'Save';
  oItem.Glyph := 'S';
  Toolbar.Items.Add.Kind := tikSeparator;
  oItem := Toolbar.Items.Add;
  oItem.Kind := tikToggle;
  oItem.Caption := 'Bold';
  oItem.Glyph := 'B';
  // a spacer takes the free width and pushes the next items to the right
  Toolbar.Items.Add.Kind := tikSpacer;
  oItem := Toolbar.Items.Add;
  oItem.Caption := 'Help';
  oItem.Glyph := '?';

  // the status bar, aligned to the bottom of the form by default
  StatusBar := TsgcVCLStatusBar.Create(Self);
  StatusBar.Parent := Self;
  oPanel := StatusBar.Panels.Add;
  oPanel.Spring := True;
  oPanel.Text := 'Ready';
  oPanel := StatusBar.Panels.Add;
  oPanel.Kind := spkIndicator;
  oPanel.AutoSize := True;
```

```
  oPanel.Text := 'Saved';
  oPanel.IndicatorState := sisOk;

  Memo := TMemo.Create(Self);
  Memo.Parent := Self;
  Memo.Align := alClient;
end;

procedure TMainForm.ToolbarItemClick(Sender: TObject; aItem: TsgcToolbarItem);
begin
  StatusBar.SetPanelText(0, aItem.Caption + ' clicked');
end;
```

Voir aussi

- TsgcVCLToolbar
- TsgcVCLStatusBar
- TsgcToolbarItem
- Types d'éléments
- Panneaux d'état

Démarrage rapide FMX

Les contrôles FireMonkey nécessitent Delphi XE6 ou ultérieur. Ils s'utilisent exactement comme ceux de VCL : déposez un **TsgcFMXToolbar**, aligné en haut par défaut, et un **TsgcFMXStatusBar**, aligné en bas, remplissez **Items** et **Panels** avec l'éditeur de collection et gérez **OnItemClick**. Une pression sur un bouton fonctionne comme un clic.

Dans le code

```
uses
  System.Classes, FMX.Forms, FMX.Types, FMX.Memo,
  sgcUICore_Theme, sgcToolbar_Model, sgcToolbar_FMX;

procedure TMainForm.CreateBars;
var
  oItem: TsgcToolbarItem;
  oPanel: TsgcStatusBarPanel;
begin
  // the toolbar, aligned to the top by default; its height follows ButtonSize
  Toolbar := TsgcFMXToolbar.Create(Self);
  Toolbar.Parent := Self;
  Toolbar.Theme := utDark;
  Toolbar.OnItemClick := ToolbarItemClick;
  oItem := Toolbar.Items.Add;
  oItem.Caption := 'Open';
  oItem.Glyph := 'O';
  oItem := Toolbar.Items.Add;
  oItem.Kind := tikToggle;
  oItem.Caption := 'Wrap';
  oItem.Glyph := 'W';

  // the status bar, aligned to the bottom by default
  StatusBar := TsgcFMXStatusBar.Create(Self);
  StatusBar.Parent := Self;
  StatusBar.Theme := utDark;
  oPanel := StatusBar.Panels.Add;
  oPanel.Spring := True;
  oPanel.Text := 'Ready';
  oPanel := StatusBar.Panels.Add;
  oPanel.Kind := spkProgress;
  oPanel.Width := 120;

  Memo := TMemo.Create(Self);
  Memo.Parent := Self;
  Memo.Align := TAlignLayout.Client;
end;

procedure TMainForm.ToolbarItemClick(Sender: TObject; aItem: TsgcToolbarItem);
begin
  if aItem.Kind = tikToggle then
    Memo.WordWrap := aItem.Down
  else
    StatusBar.SetPanelText(0, aItem.Caption + ' clicked');
end;
```

Voir aussi

- TsgcFMXToolbar
- TsgcFMXStatusBar
- Plateformes

Démarrage rapide C++Builder

Installez les packages sgcToolbar pour votre version de C++Builder, ajoutez `sgcUI\Source` et le dossier des unités compilées, par exemple `sgcUI\LibD13\win32` (`sgcUI\LibD13\win64` pour le 64 bits), aux chemins d'inclusion et de bibliothèque du projet, puis utilisez les composants de la page de palette **SGC Toolbar** exactement comme dans Delphi. L'installateur ajoute les deux chemins à l'IDE pour vous. Les en-têtes .hpp et les fichiers .lib générés de sgcToolbar et de sgcUICore se trouvent dans ce seul dossier, sgcUICore n'a donc besoin d'aucun chemin propre.

Dans le code

OnItemClick est un événement closure, donc son gestionnaire est une fonction membre __fastcall d'une classe.

Voir aussi

- Démarrage rapide VCL
- Démos
- Plateformes

Démos

Les démos se trouvent dans le dossier **Démos** de l'installation, avec les projets Delphi sous Demos\Delphi et les portages C++Builder des démos VCL sous Demos\CBuilder. Chaque dossier de démo contient un readme.html qui explique ce que fait la démo et comment elle fonctionne.

Démo	Composants	Ce qu'elle montre
EditorShell	TsgcVCLToolbar, TsgcVCLStatusBar, TActionList	L'ossature d'un petit éditeur de texte : des boutons New, Open et Save liés à des actions, les boutons à bascule Bold et Italic, un groupe d'options Left, Center et Right, un bouton déroulant Zoom, un espaceur, un bouton à bascule de thème sombre et un menu de débordement, et une barre d'état avec la position du curseur et un indicateur d'état enregistré. Delphi et C++Builder, Win32 et Win64. Les liens d'action et le menu de débordement ne font pas partie de l'édition Community, donc cette démo ne se compile pas avec elle.
EditorShellFMX	TsgcFMXToolbar, TsgcFMXStatusBar, TActionList	La démo EditorShell pour FireMonkey, avec les mêmes éléments, actions et panneaux. Delphi, Win32 et Win64. Elle nécessite aussi une compilation Registered ou Trial.
LiveStatus	TsgcVCLStatusBar, TsgcVCLToolbar, TThread	Un thread de travail simule un téléchargement et met à jour directement un panneau de texte Spring, un panneau de progression, un panneau d'indicateur et un compteur AutoSize avec SetPanelText, SetProgress et SetIndicator, sans Synchronize. Delphi et C++Builder, Win32 et Win64.

Voir aussi

- Démarrage rapide VCL
- Mises à jour thread-safe
- Liens d'action

TsgcVCLToolbar

Unité : sgcToolbar_VCL
TsgcVCLToolbar est la barre d'outils VCL, Windows uniquement. Elle est alignée en haut par défaut et affiche ses Items de gauche à droite : voir Types d'éléments. Elle gère la souris, le clavier, les info-bulles des éléments, les menus déroulants et le menu de débordement. Sa hauteur suit ButtonSize et Padding, multipliés par PPI / 96.

Propriétés

Propriété	Type	Description
Items	TsgcToolbarItems	Les éléments, dans l'ordre d'affichage. Voir TsgcToolbarItems.
ShowCaptions	Boolean	Dessine la légende à côté de l'icône. Lorsque la valeur est False, les boutons avec icône n'affichent que l'icône et la légende devient l'info-bulle. Un élément sans Glyph ni ImageIndex affiche toujours sa légende. Par défaut True.
ButtonSize	Integer	Hauteur d'un bouton, et largeur d'un bouton sans légende, en pixels logiques. Par défaut 32.
Spacing	Integer	Espace entre deux éléments en pixels logiques. Par défaut 4.
Padding	Integer	Espace autour des boutons en pixels logiques. La hauteur de la barre vaut ButtonSize plus deux fois Padding. Par défaut 6.
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom, qui utilise les couleurs claires. Voir Thèmes.
AccentColor	TsgcUIColor	Couleur de l'anneau de focus et d'un bouton à bascule enfoncé, sous la forme $AARRGGBB. 0 (par défaut) utilise la couleur d'accent du thème.
Images	TCustomImageList	La liste d'images de l'ImageIndex des éléments. Chaque image est convertie une fois en ARGB direct et mise en cache jusqu'à ce que la liste change. Voir Images et glyphes.
Model	TsgcToolbarModel (public, lecture seule)	Le modèle de la barre. Voir TsgcToolbarModel.
OverflowMenu	TPopupMenu (public, lecture seule)	Le menu du bouton de débordement, reconstruit à chaque affichage. Absent de l'édition Community. Voir Débordement.
MaxFPS	Integer	La limite de cadence d'images de la base de contrôle sgcUICore.

Les membres habituels d'un contrôle sont également publiés : **Align** (alTop par défaut), **Anchors**, **Constraints**, **Enabled**, **ParentShowHint**, **PopupMenu**, **ShowHint**, **TabOrder**, **TabStop** (False par défaut), **Visible**, OnClick, OnContextPopup, OnEnter, OnExit, OnMouseDown, OnMouseMove, OnMouseUp et OnResize.

Méthodes

Méthode	Re-tourne	Description
ItemAtPos(X, Y: Integer)	Integer	L'index de l'élément à un point du contrôle, CS_TOOLBAR_HIT_OVERFLOW (-2) pour le bouton de débordement ou CS_TOOLBAR_HIT_NONE (-1). Les séparateurs et les espaceurs ne sont jamais atteints.
ItemRect(aIndex: Integer)	TRect	Le rectangle d'un élément en coordonnées du contrôle, ou un rectangle vide lorsque l'élément n'est pas sur la barre.
OverflowRect	TRect	Le rectangle du bouton de débordement, vide lorsqu'il n'y a pas de débordement.
ClickItem(aIndex: Integer)		Active un élément comme le ferait un clic : bascule un bouton à bascule, déclenche OnItemClick, exécute l'OnClick de l'élément ou son Action et affiche le menu d'un élément déroulant.
InitiateAction		Permet à chaque élément ayant une Action de se mettre à jour. Le framework l'appelle au repos.

Événements

Événement	Type	Description
OnItemClick	TsgcToolbarItemEvent	Un bouton, un bouton à bascule ou un élément déroulant activé a reçu un clic ou une pression, a été choisi dans le menu de débordement ou activé avec Entrée ou Espace. Down d'un bouton à bascule a déjà changé. L'OnClick de l'élément ou son Action s'exécute après cet événement.

Exemple

L'exemple est une méthode d'une fiche VCL avec un **Toolbar** (TsgcVCLToolbar), un **StatusBar** (TsgcVCLStatusBar) et un **Memo** (TMemo).

```
var
  vIndex: Integer;
  vRect: TRect;
  vPoint: TPoint;
begin
  Toolbar.ShowCaptions := False;       // icons only; the caption becomes the hint
  Toolbar.ButtonSize := 28;            // the height follows: 28 + 2 x Padding
  vPoint := Toolbar.ScreenToClient(Mouse.CursorPos);
  vIndex := Toolbar.ItemAtPos(vPoint.X, vPoint.Y);
  if vIndex >= 0 then
  begin
    vRect := Toolbar.ItemRect(vIndex);
    Caption := Format('%s at x=%d', [Toolbar.Items[vIndex].Caption, vRect.Left]);
  end
  else if vIndex = CS_TOOLBAR_HIT_OVERFLOW then
    Caption := 'The overflow button';
end;
```

Voir aussi

- TsgcFMXToolbar
- TsgcToolbarItem
- TsgcToolbarModel
- Démarrage rapide VCL

TsgcFMXToolbar

Unité : sgcToolbar_FMX (Delphi XE6 ou ultérieur)
TsgcFMXToolbar est la barre d'outils FireMonkey avec les mêmes propriétés, méthodes et événements que TsgcVCLToolbar. Elle est alignée en haut par défaut et sa hauteur vaut ButtonSize plus deux fois Padding, en unités logiques. Elle gère la souris et le tactile, le clavier lorsqu'elle peut recevoir le focus, et affiche le TPopupMenu FireMonkey d'un élément déroulant et du bouton de débordement.

Propriétés

Propriété	Type	Description
Items	TsgcToolbarItems	Les éléments, dans l'ordre d'affichage. Voir TsgcToolbarItems.
ShowCaptions	Boolean	Dessine la légende à côté de l'icône. Lorsque la valeur est False, les boutons avec icône n'affichent que l'icône et la légende devient l'info-bulle. Un élément sans Glyph ni ImageIndex affiche toujours sa légende. Par défaut True.
ButtonSize	Integer	Hauteur d'un bouton, et largeur d'un bouton sans légende, en pixels logiques. Par défaut 32.
Spacing	Integer	Espace entre deux éléments en pixels logiques. Par défaut 4.
Padding	Integer	Espace autour des boutons en pixels logiques. La hauteur de la barre vaut ButtonSize plus deux fois Padding. Par défaut 6.
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom, qui utilise les couleurs claires. Voir Thèmes.
AccentColor	TsgcUIColor	Couleur de l'anneau de focus et d'un bouton à bascule enfoncé, sous la forme $AARRGGBB. 0 (par défaut) utilise la couleur d'accent du thème.
Images	TCustomImageList	La liste d'images de l'ImageIndex des éléments. Elle nécessite XE8 ou ultérieur, et les images sont dessinées en 16 x 16 pixels logiques. Voir Images et glyphes.
Model	TsgcToolbarModel (public, lecture seule)	Le modèle de la barre. Voir TsgcToolbarModel.
OverflowMenu	TPopupMenu (public, lecture seule)	Le menu du bouton de débordement, reconstruit à chaque affichage. Absent de l'édition Community. Voir Débordement.
MaxFPS	Integer	La limite de cadence d'images de la base de contrôle sgcUICore.

Les membres habituels d'un contrôle FireMonkey sont également publiés : **Align** (Top par défaut), **Anchors**, **CanFocus** (False par défaut), **Cursor**, **Enabled**, **HitTest**, **Locked**, **Margins**, **Opacity**, **PopupMenu**, **Position**, **TabOrder**, **Visible**, **Width**, et à partir de 10.1 Berlin **TabStop**, **ParentShowHint** et **ShowHint**. Les événements comprennent OnClick, OnEnter, OnExit, OnKeyDown, OnKeyUp, les événements de souris, OnPainting, OnPaint et OnResize.

Méthodes

Méthode	Re-tourne	Description
ItemAtPos(X, Y: Single)	Integer	L'index de l'élément à un point du contrôle, CS_TOOLBAR_HIT_OVERFLOW (-2) pour le bouton de débordement ou CS_TOOLBAR_HIT_NONE (-1). Les séparateurs et les espaceurs ne sont jamais atteints.
ItemRect(aIndex: Integer)	TRectF	Le rectangle d'un élément en coordonnées du contrôle, ou un rectangle vide lorsque l'élément n'est pas sur la barre.
OverflowRect	TRectF	Le rectangle du bouton de débordement, vide lorsqu'il n'y a pas de débordement.
ClickItem(aIndex: Integer)		Active un élément comme le ferait un clic : bascule un bouton à bascule, déclenche OnItemClick, exécute l'OnClick de l'élément ou son Action et affiche le menu d'un élément déroulant.
InitiateAction		Permet à chaque élément ayant une Action de se mettre à jour. Le framework l'appelle au repos.

Événements

Événe-ment	Type	Description
OnItem-Click	TsgcTool-barIte-mEvent	Un bouton, un bouton à bascule ou un élément déroulant activé a reçu un clic ou une pression, a été choisi dans le menu de débordement ou activé avec Entrée ou Espace. Down d'un bouton à bascule a déjà changé. L'OnClick de l'élément ou son Action s'exécute après cet événement.

Exemple

```
uses
  System.Classes, FMX.Forms, FMX.Types, FMX.Memo,
  sgcUICore_Theme, sgcToolbar_Model, sgcToolbar_FMX;

procedure TMainForm.CreateBars;
var
  oItem: TsgcToolbarItem;
  oPanel: TsgcStatusBarPanel;
begin
  // the toolbar, aligned to the top by default; its height follows ButtonSize
  Toolbar := TsgcFMXToolbar.Create(Self);
  Toolbar.Parent := Self;
  Toolbar.Theme := utDark;
  Toolbar.OnItemClick := ToolbarItemClick;
  oItem := Toolbar.Items.Add;
  oItem.Caption := 'Open';
  oItem.Glyph := 'O';
  oItem := Toolbar.Items.Add;
  oItem.Kind := tikToggle;
  oItem.Caption := 'Wrap';
  oItem.Glyph := 'W';

  // the status bar, aligned to the bottom by default
  StatusBar := TsgcFMXStatusBar.Create(Self);
```

```
  StatusBar.Parent := Self;
  StatusBar.Theme := utDark;
  oPanel := StatusBar.Panels.Add;
  oPanel.Spring := True;
  oPanel.Text := 'Ready';
  oPanel := StatusBar.Panels.Add;
  oPanel.Kind := spkProgress;
  oPanel.Width := 120;

  Memo := TMemo.Create(Self);
  Memo.Parent := Self;
  Memo.Align := TAlignLayout.Client;
end;

procedure TMainForm.ToolbarItemClick(Sender: TObject; aItem: TsgcToolbarItem);
begin
  if aItem.Kind = tikToggle then
    Memo.WordWrap := aItem.Down
  else
    StatusBar.SetPanelText(0, aItem.Caption + ' clicked');
end;
```

Voir aussi

- TsgcVCLToolbar
- Démarrage rapide FMX
- Plateformes

TsgcVCLStatusBar

Unité : sgcToolbar_VCL
TsgcVCLStatusBar est la barre d'état VCL, Windows uniquement. Elle est alignée en bas par défaut et affiche ses Panels: panneaux de texte, panneaux de progression et panneaux d'indicateur, fixes, AutoSize ou Spring. Elle affiche le Hint du panneau sous la souris et son point de redimensionnement redimensionne la fiche. N'importe quel thread peut mettre à jour les panneaux avec les trois setters thread-safe.

Propriétés

Propriété	Type	Description
Panels	TsgcStatusBar-Panels	Les panneaux, de gauche à droite. Voir TsgcStatusBarPanels.
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom, qui utilise les couleurs claires. Les couleurs des indicateurs suivent le thème.
Accent-Color	TsgcUIColor	Couleur des barres de progression, sous la forme $AARRGGBB. 0 (par défaut) utilise la couleur d'accent du thème.
SizeGrip	Boolean	Dessine le point de redimensionnement à l'extrémité droite. Il redimensionne la fiche parente lorsque celle-ci a une bordure redimensionnable et n'est pas agrandie. Par défaut True. Voir Point de redimensionnement.
Height	Integer	Hauteur de la barre, 24 pixels logiques par défaut.
Model	TsgcStatusBar-Model (public, lecture seule)	Le modèle de la barre. Voir TsgcStatusBarModel.
MaxFPS	Integer	La limite de cadence d'images de la base de contrôle sgcUICore.

Les membres habituels d'un contrôle sont également publiés : **Align** (alBottom par défaut), **Anchors**, **Constraints**, **Enabled**, **ParentShowHint**, **PopupMenu**, **ShowHint**, **Visible**, OnClick, OnContextPopup, OnMouseDown, OnMouseMove, OnMouseUp et OnResize.

Méthodes

Méthode	Retourne	Description
SetPanelText(aIndex: Integer; const aText: string)		Définit le Text d'un panneau. Thread-safe, un index invalide est ignoré. Voir Mises à jour thread-safe.
SetProgress(aIndex, aValue: Integer)		Définit le Progress (0 à 100) d'un panneau. Thread-safe.
SetIndicator(aIndex: Integer; aState: TsgcStatusIndicatorState)		Définit le IndicatorState d'un panneau. Thread-safe.
PanelAtPos(X, Y: Integer)	Integer	L'index du panneau à un point du contrôle, ou -1.
PanelRect(aIndex: Integer)	TRect	Le rectangle d'un panneau en coordonnées du contrôle.

GripRect	TRect	Le rectangle du point de redimensionnement, vide lorsque SizeGrip vaut False.

Événements

Événement	Type	Description
OnPanelClick	TsgcStatusBarPanelEvent	Un panneau a reçu un clic ou une pression. aPanel est le panneau.

Exemple

L'exemple est une méthode d'une fiche VCL avec un **Toolbar** (TsgcVCLToolbar), un **StatusBar** (TsgcVCLStatusBar) et un **Memo** (TMemo).

```
procedure TFeatureForm.StatusBarEvents;
begin
  StatusBar.ShowHint := True;          // shows the Hint of the panel under the mouse
  StatusBar.OnPanelClick := StatusPanelClick;
end;

procedure TFeatureForm.StatusPanelClick(Sender: TObject;
  aPanel: TsgcStatusBarPanel);
begin
  if aPanel.Kind = spkIndicator then
    ShowMessage(aPanel.Text);
end;
```

Voir aussi

- TsgcFMXStatusBar
- TsgcStatusBarPanel
- Mises à jour thread-safe
- TsgcStatusBarModel

TsgcFMXStatusBar

Unité : sgcToolbar_FMX (Delphi XE6 ou ultérieur)
TsgcFMXStatusBar est la barre d'état FireMonkey avec les mêmes propriétés, méthodes et événements que TsgcVCLStatusBar. Elle est alignée en bas par défaut. Les setters thread-safe mettent le repaint en file d'attente vers le thread principal. Le point de redimensionnement est dessiné, mais il ne redimensionne pas la fiche.

Propriétés

Propriété	Type	Description
Panels	TsgcStatusBar-Panels	Les panneaux, de gauche à droite. Voir TsgcStatusBarPanels.
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom, qui utilise les couleurs claires. Les couleurs des indicateurs suivent le thème.
Accent-Color	TsgcUIColor	Couleur des barres de progression, sous la forme $AARRGGBB. 0 (par défaut) utilise la couleur d'accent du thème.
SizeGrip	Boolean	Dessine le point de redimensionnement à l'extrémité droite. FireMonkey n'a pas de test de position hors zone client, donc le point ne redimensionne pas la fiche. Par défaut True. Voir Point de redimensionnement.
Height	Single	Hauteur de la barre, 24 pixels logiques par défaut.
Model	TsgcStatusBar-Model (public, lecture seule)	Le modèle de la barre. Voir TsgcStatusBarModel.
MaxFPS	Integer	La limite de cadence d'images de la base de contrôle sgcUICore.

Les membres habituels d'un contrôle FireMonkey sont également publiés : **Align** (Bottom par défaut), **Anchors**, **Cursor**, **Enabled**, **HitTest**, **Locked**, **Margins**, **Opacity**, **PopupMenu**, **Position**, **Visible**, **Width**, et à partir de 10.1 Berlin **Size**, **ParentShowHint** et **ShowHint**. Les événements comprennent OnClick, les événements de souris, OnPainting, OnPaint et OnResize.

Méthodes

Méthode	Retourne	Description
SetPanelText(aIndex: Integer; const aText: string)		Définit le Text d'un panneau. Thread-safe, un index invalide est ignoré. Voir Mises à jour thread-safe.
SetProgress(aIndex, aValue: Integer)		Définit le Progress (0 à 100) d'un panneau. Thread-safe.
SetIndicator(aIndex: Integer; aState: TsgcStatusIndicatorState)		Définit le IndicatorState d'un panneau. Thread-safe.
PanelAtPos(X, Y: Single)	Integer	L'index du panneau à un point du contrôle, ou -1.
PanelRect(aIndex: Integer)	TRectF	Le rectangle d'un panneau en coordonnées du contrôle.

GripRect	TRectF	Le rectangle du point de redimensionnement, vide lorsque SizeGrip vaut False.

Événements

Événement	Type	Description
OnPanelClick	TsgcStatusBarPanelEvent	Un panneau a reçu un clic ou une pression. aPanel est le panneau.

Exemple

La partie barre d'état du démarrage rapide FireMonkey : un panneau de texte Spring et un panneau de progression.

```
uses
  System.Classes, FMX.Forms, FMX.Types, FMX.Memo,
  sgcUICore_Theme, sgcToolbar_Model, sgcToolbar_FMX;

procedure TMainForm.CreateBars;
var
  oItem: TsgcToolbarItem;
  oPanel: TsgcStatusBarPanel;
begin
  // the toolbar, aligned to the top by default; its height follows ButtonSize
  Toolbar := TsgcFMXToolbar.Create(Self);
  Toolbar.Parent := Self;
  Toolbar.Theme := utDark;
  Toolbar.OnItemClick := ToolbarItemClick;
  oItem := Toolbar.Items.Add;
  oItem.Caption := 'Open';
  oItem.Glyph := 'O';
  oItem := Toolbar.Items.Add;
  oItem.Kind := tikToggle;
  oItem.Caption := 'Wrap';
  oItem.Glyph := 'W';

  // the status bar, aligned to the bottom by default
  StatusBar := TsgcFMXStatusBar.Create(Self);
  StatusBar.Parent := Self;
  StatusBar.Theme := utDark;
  oPanel := StatusBar.Panels.Add;
  oPanel.Spring := True;
  oPanel.Text := 'Ready';
  oPanel := StatusBar.Panels.Add;
  oPanel.Kind := spkProgress;
  oPanel.Width := 120;

  Memo := TMemo.Create(Self);
  Memo.Parent := Self;
  Memo.Align := TAlignLayout.Client;
end;

procedure TMainForm.ToolbarItemClick(Sender: TObject; aItem: TsgcToolbarItem);
begin
  if aItem.Kind = tikToggle then
    Memo.WordWrap := aItem.Down
  else
    StatusBar.SetPanelText(0, aItem.Caption + ' clicked');
end;
```

Voir aussi

- TsgcVCLStatusBar
- Démarrage rapide FMX
- Point de redimensionnement

TsgcToolbarItem

Unité : sgcToolbar_Model
TsgcToolbarItem est un élément d'une barre d'outils, un TCollectionItem de TsgcToolbarItems. La même classe est utilisée par les barres d'outils VCL et FMX.

Propriétés

Propriété	Type	Description
Kind	TsgcToolbarItemKind	tikButton (par défaut), tikToggle, tikDropdown, tikSeparator, tikSpacer ou tikLabel. Voir Types d'éléments.
Caption	string	Le texte de l'élément.
Glyph	string	Un court texte dessiné dans l'emplacement de l'icône lorsque l'élément n'a pas d'image, par exemple B pour Bold.
ImageIndex	Integer	Index dans les Images de la barre, -1 (par défaut) pour aucune.
Hint	string	L'info-bulle de l'élément. Lorsqu'elle est vide et que les légendes sont masquées, la Caption sert d'info-bulle.
Enabled	Boolean	Un élément désactivé est dessiné en teinte atténuée et ignore la souris et le clavier. Par défaut True.
Visible	Boolean	Un élément masqué n'occupe aucun espace. Par défaut True.
Down	Boolean	L'état d'un élément tikToggle. Avec GroupIndex supérieur à 0, définir Down réinitialise les autres éléments du même groupe. Par défaut False.
GroupIndex	Integer	Les boutons à bascule ayant le même GroupIndex supérieur à 0 forment un groupe d'options. Par défaut 0. Voir Boutons à bascule et groupes d'options.
Width	Integer	Largeur en pixels logiques, 0 (par défaut) pour automatique. Pour un espaceur, c'est la largeur minimale.
Tag	Integer	Une valeur libre pour votre code.
DropDownMenu	TComponent	Le menu contextuel d'un élément tikDropdown : un TPopupMenu VCL sur TsgcVCLToolbar, un TPopupMenu FireMonkey sur TsgcFMXToolbar. La barre efface la référence lorsque le menu est libéré. Voir Menus déroulants.
Action	TBasicAction	Une action liée. Caption, Enabled, Visible, Hint, ImageIndex et Checked (Down d'un bouton à bascule) la suivent, et un clic l'exécute sauf si OnClick est affecté. Absent de l'édition Community. Voir Liens d'action.
BoundsRect	TsgcUIRect (public, lecture seule)	Le rectangle de la dernière mise en page, en pixels de périphérique.
Shown	Boolean (public, lecture seule)	True lorsque la dernière mise en page a placé l'élément sur la barre.

InOverflow	Boolean (public, lecture seule)	True lorsque la dernière mise en page a déplacé l'élément dans la liste de débordement.

Méthodes

Méthode	Retourne	Description
Click(aComponent: TComponent = nil)		Exécute OnClick s'il est affecté, sinon l'Action liée avec aComponent comme ActionComponent. Elle ne bascule pas un bouton à bascule et ne déclenche pas OnItemClick, utilisez pour cela ClickItem de la barre.
InitiateAction		Permet à l'Action liée de se mettre à jour. Sans effet sans Action.
IsButton	Boolean	True pour les types sur lesquels on peut cliquer : tikButton, tikToggle et tikDropdown.

Événements

Événement	Type	Description
OnClick	TNotifyEvent	L'élément a été activé. Sender est l'élément. Il s'exécute après OnItemClick de la barre.

Exemple

L'exemple est une méthode d'une fiche VCL avec un **Toolbar** (TsgcVCLToolbar), un **StatusBar** (TsgcVCLStatusBar) et un **Memo** (TMemo).

```
var
  oItem: TsgcToolbarItem;
begin
  oItem := Toolbar.Items.Add;          // tikButton is the default kind
  oItem.Caption := 'Save';
  oItem.Glyph := 'S';

  oItem := Toolbar.Items.Add;
  oItem.Kind := tikToggle;             // stays down until clicked again
  oItem.Caption := 'Bold';
  oItem.Glyph := 'B';

  oItem := Toolbar.Items.Add;
  oItem.Kind := tikDropdown;           // shows its DropdownMenu
  oItem.Caption := 'Zoom';

  Toolbar.Items.Add.Kind := tikSeparator;  // a vertical line

  oItem := Toolbar.Items.Add;
  oItem.Kind := tikLabel;              // text only, never clicked
  oItem.Caption := 'Find:';

  oItem := Toolbar.Items.Add;
  oItem.Kind := tikSpacer;             // shares the free width
  oItem.Width := 16;                   // minimum width, logical pixels
end;
```

Voir aussi

- TsgcToolbarItems
- Types d'éléments
- Boutons à bascule et groupes d'options

TsgcToolbarItems

Unité : sgcToolbar_Model
TsgcToolbarItems est la collection des TsgcToolbarItem d'une barre d'outils et de TsgcToolbarModel. Une modification d'un élément met à jour la barre.

Propriétés

Propriété	Type	Description
Items[Index: Integer]	TsgcToolbarItem	L'élément à Index. C'est la propriété par défaut, donc Toolbar.Items[0] est le premier élément.
Count	Integer	Le nombre d'éléments.

Méthodes

Méthode	Retourne	Description
Add	TsgcToolbarItem	Ajoute un élément tikButton à la fin et le retourne.
Delete(Index: Integer)		Supprime un élément.
Clear		Supprime tous les éléments.

Pour modifier de nombreux éléments avec un seul repaint, encadrez les modifications par BeginUpdate et EndUpdate du modèle.

Exemple

L'exemple est une méthode d'une fiche VCL avec un **Toolbar** (TsgcVCLToolbar), un **StatusBar** (TsgcVCLStatusBar) et un **Memo** (TMemo).

```
var
  i: Integer;
begin
  // disables every clickable item with Tag 1, in one repaint
  Toolbar.Model.BeginUpdate;
  try
    for i := 0 to Toolbar.Items.Count - 1 do
      if Toolbar.Items[i].IsButton and (Toolbar.Items[i].Tag = 1) then
        Toolbar.Items[i].Enabled := False;
  finally
    Toolbar.Model.EndUpdate;
  end;
  Toolbar.Items.Delete(Toolbar.Items.Count - 1);
end;
```

Voir aussi

- TsgcToolbarItem
- TsgcToolbarModel

TsgcStatusBarPanel

Unité : sgcToolbar_Model
TsgcStatusBarPanel est un panneau d'une barre d'état, un TCollectionItem de TsgcStatusBarPanels. Text, Progress et IndicatorState sont protégés par le verrou du modèle.

Propriétés

Propriété	Type	Description
Kind	TsgcStatusPanelKind	spkText (par défaut), spkProgress ou spkIndicator. Voir Progression et indicateurs.
Text	string	Le texte d'un panneau de texte ou d'indicateur.
Width	Integer	Largeur en pixels logiques lorsque ni AutoSize ni Spring n'est défini. Par défaut 100.
AutoSize	Boolean	La largeur suit le texte, ainsi que le point d'un indicateur. Par défaut False.
Spring	Boolean	Les panneaux Spring se partagent la largeur laissée par les autres. Par défaut False. Voir Panneaux d'état.
Alignment	TAlignment	taLeftJustify (par défaut), taCenter ou taRightJustify.
Progress	Integer	La valeur d'un panneau spkProgress, de 0 à 100. Par défaut 0.
IndicatorState	TsgcStatusIndicatorState	L'état d'un panneau spkIndicator : sisOff (par défaut), sisOk, sisWarning ou sisError.
Hint	string	L'info-bulle affichée au-dessus du panneau.
Tag	Integer	Une valeur libre pour votre code.
BoundsRect	TsgcUIRect (public, lecture seule)	Le rectangle de la dernière mise en page, en pixels de périphérique.

Exemple

L'exemple est une méthode d'une fiche VCL avec un **Toolbar** (TsgcVCLToolbar), un **StatusBar** (TsgcVCLStatusBar) et un **Memo** (TMemo).

```
var
  oPanel: TsgcStatusBarPanel;
begin
  StatusBar.Panels.Clear;

  oPanel := StatusBar.Panels.Add;      // Spring: takes the width that is left
  oPanel.Spring := True;
  oPanel.Text := 'Ready';

  oPanel := StatusBar.Panels.Add;      // AutoSize: as wide as its text
  oPanel.AutoSize := True;
  oPanel.Text := 'UTF-8';

  oPanel := StatusBar.Panels.Add;      // fixed: Width logical pixels
```

```
  oPanel.Width := 110;
  oPanel.Alignment := taRightJustify;
  oPanel.Text := 'Ln 1, Col 1';
  oPanel.Hint := 'Caret position';
end;
```

Voir aussi

- TsgcStatusBarPanels
- Panneaux d'état
- Progression et indicateurs

TsgcStatusBarPanels

Unité : sgcToolbar_Model
TsgcStatusBarPanels est la collection des TsgcStatusBarPanel d'une barre d'état et de TsgcStatusBarModel.

Propriétés

Propriété	Type	Description
Items[Index: Integer]	TsgcStatusBarPanel	Le panneau à Index. C'est la propriété par défaut, donc StatusBar.Panels[0] est le premier panneau.
Count	Integer	Le nombre de panneaux.

Méthodes

Méthode	Retourne	Description
Add	TsgcStatusBarPanel	Ajoute un panneau spkText à la fin et le retourne.
Delete(Index: Integer)		Supprime un panneau.
Clear		Supprime tous les panneaux.

Ajoutez, supprimez et configurez les panneaux depuis le thread principal. Depuis un thread de travail, utilisez SetPanelText, SetProgress et SetIndicator de la barre, voir Mises à jour thread-safe.

Exemple

L'exemple est une méthode d'une fiche VCL avec un **Toolbar** (TsgcVCLToolbar), un **StatusBar** (TsgcVCLStatusBar) et un **Memo** (TMemo).

```
var
  i: Integer;
begin
  // resets every indicator panel
  for i := 0 to StatusBar.Panels.Count - 1 do
    if StatusBar.Panels[i].Kind = spkIndicator then
      StatusBar.SetIndicator(i, sisOff);
end;
```

Voir aussi

- TsgcStatusBarPanel
- TsgcStatusBarModel

TsgcToolbarModel

Unité : sgcToolbar_Model
TsgcToolbarModel est le modèle indépendant du framework d'une barre d'outils. Chaque barre d'outils en possède un, accessible par sa propriété **Model** , et un modèle créé dans le code dessine une barre d'outils sans fenêtre. Les tailles sont en pixels logiques. Layout les multiplie par l'échelle, et chaque rectangle calculé est en pixels de périphérique.

Propriétés

Propriété	Type	Description
Items, ShowCaptions, ButtonSize, Spacing, Padding, Theme, AccentColor		Les propriétés publiées de la barre d'outils, voir TsgcVCLToolbar.
HotIndex, PressedIndex, FocusedIndex	Integer	L'état d'interaction : un index d'élément, CS_TOOLBAR_HIT_OVERFLOW pour le bouton de débordement, ou -1.
HasOverflow	Boolean (lecture seule)	True lorsque la dernière mise en page a déplacé des éléments dans la liste de débordement.
OverflowCount	Integer (lecture seule)	Le nombre d'éléments de la liste de débordement.
OverflowItem[Index: Integer]	TsgcToolbarItem (lecture seule)	Un élément de la liste de débordement.
OverflowRect	TsgcUIRect (lecture seule)	Le bouton de débordement de la dernière mise en page, en pixels de périphérique.
Owner	TPersistent	Le composant propriétaire du modèle, utilisé pour le chemin de noms à la conception.

Méthodes

Méthode	Retourne	Description
Layout(aWidth, aHeight: Integer; aScale: Double; aMeasure or aCanvas)		Calcule les rectangles des éléments pour une barre de aWidth par aHeight pixels de périphérique. aMeasure (un TsgcToolbarMeasureEvent) ou aCanvas mesure le texte ; une mesure nil utilise une estimation de 0,55 fois la taille de la police par caractère. Les espaceurs se partagent la largeur libre. Avec le menu de débordement, les éléments qui ne tiennent pas passent dans la liste de débordement, sinon ils sont tronqués.

HitTest(aX, aY: Double)	Integer	L'élément à un point de la dernière mise en page, CS_TOOLBAR_HIT_OVERFLOW ou CS_TOOLBAR_HIT_NONE.
Render(aCanvas; aWidth, aHeight: Integer; aScale: Double = 1)		Dessine la barre sur un canvas sgcUICore, en utilisant OnGetImage pour les images.
SaveToPNG, SaveToSVG(aFileName or aStream; aWidth, aHeight: Integer; aScale: Double = 1)		Écrit la barre sous forme d'image. Voir Rendu sans interface.
BeginUpdate, EndUpdate		Regroupent les modifications en un seul OnChange.
GetAccent	TsgcUIColor	AccentColor, ou la couleur d'accent du thème lorsqu'elle vaut 0.
ShowsCaption(aItem)	Boolean	True lorsque l'élément dessine sa légende : ShowCaptions, ou l'élément n'a pas d'icône.
HasIcon(aItem)	Boolean	True lorsque l'élément a un Glyph ou un ImageIndex.

Événements

Événement	Type	Description
OnChange	TNotifyEvent	Le modèle a changé. Les barres se redessinent à ce moment.
OnGetImage	TsgcToolbarImageEvent	Retourne l'image d'un ImageIndex sous forme de TsgcUIImage pour Render et pour la sortie sans interface.

Exemple

Layout et HitTest sans contrôle, comme le fait un test ou un serveur.

```
uses
  SysUtils,
  sgcUICore_Theme, sgcToolbar_Model;

var
  oModel: TsgcToolbarModel;
  vMeasure: TsgcToolbarMeasureEvent;
  vIndex: Integer;
begin
  oModel := TsgcToolbarModel.Create;
  try
    oModel.BeginUpdate;
    try
      oModel.ShowCaptions := False;
      oModel.ButtonSize := 28;
      oModel.Items.Add.Caption := 'Cut';
      oModel.Items.Add.Caption := 'Copy';
      oModel.Items.Add.Caption := 'Paste';
      oModel.Items[0].Glyph := 'X';
      oModel.Items[1].Glyph := 'C';
      oModel.Items[2].Glyph := 'V';
    finally
```

```
      oModel.EndUpdate;
    end;
    // nil measures the text with an estimate; a TsgcUICanvas measures it exactly
    vMeasure := nil;
    oModel.Layout(200, 40, 1, vMeasure);
    vIndex := oModel.HitTest(50, 20);
    if vIndex >= 0 then
      WriteLn('hit ', oModel.Items[vIndex].Caption, ' from x=',
        oModel.Items[vIndex].BoundsRect.Left:0:0);
    WriteLn('overflow items: ', oModel.OverflowCount);
  finally
    oModel.Free;
  end;
end;
```

Voir aussi

- TsgcVCLToolbar
- Rendu sans interface
- Architecture

TsgcStatusBarModel

Unité : sgcToolbar_Model
TsgcStatusBarModel est le modèle indépendant du framework d'une barre d'état. Chaque barre d'état en possède un, accessible par sa propriété **Model** . Ses trois setters sont thread-safe : ils prennent le verrou du modèle, modifient la valeur et déclenchent OnChange depuis le thread appelant, et seulement lorsque la valeur a changé. Toutes les autres propriétés sont destinées au thread principal.

Propriétés

Propriété	Type	Description
Panels, Theme, AccentColor, SizeGrip		Les propriétés publiées de la barre d'état, voir TsgcVCLStatusBar.
GripRect	TsgcUIRect (lecture seule)	Le point de redimensionnement de la dernière mise en page en pixels de périphérique, vide lorsque SizeGrip vaut False.
Owner	TPersistent	Le composant propriétaire du modèle, utilisé pour le chemin de noms à la conception.

Méthodes

Méthode	Retourne	Description
SetPanelText, SetProgress, SetIndicator		Les setters thread-safe. Un index invalide est ignoré.
Layout(aWidth, aHeight: Integer; aScale: Double; aMeasure or aCanvas)		Calcule les rectangles des panneaux : d'abord les largeurs fixes, puis les panneaux AutoSize selon leur texte, puis les panneaux Spring se partagent ce qui reste, sans jamais descendre sous 0. Le point de redimensionnement est réservé à droite lorsque SizeGrip vaut True.
HitTest(aX, aY: Double)	Integer	Le panneau à un point de la dernière mise en page, ou -1.
Render, SaveToPNG, SaveToSVG		Dessinent la barre sur un canvas ou l'écrivent sous forme d'image, comme pour TsgcToolbarModel.
GetIndicatorColor(aState)	TsgcUIColor	La couleur d'un état d'indicateur dans le thème courant.
GetAccent	TsgcUIColor	AccentColor, ou la couleur d'accent du thème lorsqu'elle vaut 0.
BeginUpdate, EndUpdate		Regroupent les modifications en un seul OnChange.
Lock, Unlock		Le verrou qui protège Text, Progress et IndicatorState.

Événements

Événement	Type	Description
OnChange	TNoti-fyEvent	Le modèle a changé. Après un setter thread-safe, il se déclenche dans le thread appelant, et les barres transmettent le repaint au thread principal.

Exemple

```
uses
  SysUtils,
  sgcUICore_Theme, sgcToolbar_Model;

var
  oModel: TsgcStatusBarModel;
begin
  oModel := TsgcStatusBarModel.Create;
  try
    oModel.Theme := utDark;
    oModel.SizeGrip := False;
    oModel.Panels.Add.Spring := True;
    oModel.Panels.Add.Kind := spkIndicator;
    // thread-safe setters: an index out of range is ignored
    oModel.SetPanelText(0, 'Connected to feed');
    oModel.SetIndicator(1, sisWarning);
    WriteLn('warning colour: ', IntToHex(oModel.GetIndicatorColor(sisWarning), 8));
    oModel.SaveToSVG('status_dark.svg', 400, 24);
  finally
    oModel.Free;
  end;
end;
```

Voir aussi

- TsgcVCLStatusBar
- Mises à jour thread-safe
- Rendu sans interface

Types d'éléments

Le **Kind** d'un TsgcToolbarItem détermine la façon dont il est dessiné et dont il réagit. Les éléments sont placés de gauche à droite dans l'ordre de la collection, avec Spacing pixels logiques entre eux.

Type	Comportement
tikButton	Le type par défaut. Un clic déclenche OnItemClick de la barre, puis OnClick de l'élément ou son Action.
tikToggle	Un bouton qui reste enfoncé. Un clic bascule Down avant le déclenchement d'OnItemClick. Avec GroupIndex supérieur à 0, c'est un bouton radio, voir Boutons à bascule et groupes d'options.
tikDrop-down	Un bouton avec une petite flèche qui affiche son DropdownMenu sous le bouton. Voir Menus déroulants.
tikSepa-rator	Une fine ligne verticale. Il ne reçoit jamais de clic.
tikSpa-cer	Un élément vide qui prend la largeur libre de la barre, partagée avec les autres espaceurs. Il pousse les éléments suivants vers la droite. Width est sa largeur minimale.
tikLabel	Une légende sans bouton, par exemple avant un champ de saisie. Elle ne reçoit jamais de clic.

Un bouton dessine son icône (l'image d'ImageIndex, ou le texte du Glyph) et sa Caption. Avec ShowCaptions à False, seule l'icône est dessinée et la légende devient l'info-bulle. Un élément sans icône dessine toujours sa légende. Width supérieur à 0 fixe la largeur d'un élément en pixels logiques.

Exemple

L'exemple est une méthode d'une fiche VCL avec un **Toolbar** (TsgcVCLToolbar), un **StatusBar** (TsgcVCLStatusBar) et un **Memo** (TMemo).

```
uses
  Windows, Types, Forms, Classes, SysUtils, Controls, StdCtrls, Menus,
  ActnList, ImgList, Graphics, Dialogs,
  sgcUICore_Theme, sgcToolbar_Model, sgcToolbar_VCL;

var
  oItem: TsgcToolbarItem;
begin
  oItem := Toolbar.Items.Add;          // tikButton is the default kind
  oItem.Caption := 'Save';
  oItem.Glyph := 'S';

  oItem := Toolbar.Items.Add;
  oItem.Kind := tikToggle;             // stays down until clicked again
  oItem.Caption := 'Bold';
  oItem.Glyph := 'B';

  oItem := Toolbar.Items.Add;
  oItem.Kind := tikDropdown;           // shows its DropdownMenu
  oItem.Caption := 'Zoom';

  Toolbar.Items.Add.Kind := tikSeparator;  // a vertical line

  oItem := Toolbar.Items.Add;
  oItem.Kind := tikLabel;              // text only, never clicked
  oItem.Caption := 'Find:';
```

```
  oItem := Toolbar.Items.Add;
  oItem.Kind := tikSpacer;              // shares the free width
  oItem.Width := 16;                    // minimum width, logical pixels
end;
```

Voir aussi

- TsgcToolbarItem
- Images et glyphes
- Débordement

Boutons à bascule et groupes d'options

A **tikToggle** a un état **Down** , dessiné avec une teinte d'accent. Un clic bascule Down puis déclenche OnItemClick, donc le gestionnaire lit le nouvel état.
Les boutons à bascule qui partagent un **GroupIndex** supérieur à 0 forment un groupe d'options : définir Down sur l'un d'eux, par un clic ou depuis le code, réinitialise les autres. Un clic sur un bouton d'option déjà enfoncé le laisse enfoncé, donc un élément du groupe reste toujours sélectionné. Un bouton à bascule lié à une Action suit le Checked de l'action.

Exemple

L'exemple est une méthode d'une fiche VCL avec un **Toolbar** (TsgcVCLToolbar), un **StatusBar** (TsgcVCLStatusBar) et un **Memo** (TMemo).

```
uses
  Windows, Types, Forms, Classes, SysUtils, Controls, StdCtrls, Menus,
  ActnList, ImgList, Graphics, Dialogs,
  sgcUICore_Theme, sgcToolbar_Model, sgcToolbar_VCL;

procedure TFeatureForm.Toggles;
const
  CAPTIONS: array [0 .. 2] of string = ('Left', 'Center', 'Right');
var
  i: Integer;
  oItem: TsgcToolbarItem;
begin
  // three toggles with the same GroupIndex behave as radio buttons
  for i := 0 to 2 do
  begin
    oItem := Toolbar.Items.Add;
    oItem.Kind := tikToggle;
    oItem.GroupIndex := 1;
    oItem.Caption := CAPTIONS[i];
    oItem.Tag := i;
  end;
  Toolbar.Items[0].Down := True;       // clears the other items of group 1
  Toolbar.OnItemClick := ToolbarItemClick;
end;

procedure TFeatureForm.ToolbarItemClick(Sender: TObject;
  aItem: TsgcToolbarItem);
const
  ALIGNMENTS: array [0 .. 2] of TAlignment = (taLeftJustify, taCenter,
    taRightJustify);
begin
  // Down has already been flipped when OnItemClick fires
  if (aItem.Kind = tikToggle) and (aItem.GroupIndex = 1) and aItem.Down then
    Memo.Alignment := ALIGNMENTS[aItem.Tag];
end;
```

Voir aussi

- Types d'éléments
- Liens d'action
- TsgcToolbarItem

Menus déroulants

A **tikDropdown** dessine une petite flèche après sa légende et affiche le menu contextuel de sa propriété **DropdownMenu** sous le bouton : un TPopupMenu VCL sur TsgcVCLToolbar, un TPopupMenu FireMonkey sur TsgcFMXToolbar. La propriété est déclarée en TComponent afin que l'élément reste indépendant du framework. La barre efface la référence lorsque le menu est libéré.
Sous VCL, le menu s'ouvre lorsque le bouton est enfoncé. OnItemClick et l'OnClick de l'élément se déclenchent d'abord, donc un gestionnaire peut remplir ou cocher le menu avant son affichage. Un élément déroulant de la liste de débordement affiche son menu sous le bouton de débordement.

Exemple

L'exemple est une méthode d'une fiche VCL avec un **Toolbar** (TsgcVCLToolbar), un **StatusBar** (TsgcVCLStatusBar) et un **Memo** (TMemo).

```
uses
  Windows, Types, Forms, Classes, SysUtils, Controls, StdCtrls, Menus,
  ActnList, ImgList, Graphics, Dialogs,
  sgcUICore_Theme, sgcToolbar_Model, sgcToolbar_VCL;

procedure TFeatureForm.Dropdown;
var
  i: Integer;
  oMenuItem: TMenuItem;
  oItem: TsgcToolbarItem;
begin
  pmZoom := TPopupMenu.Create(Self);
  for i := 1 to 4 do
  begin
    oMenuItem := TMenuItem.Create(pmZoom);
    oMenuItem.Caption := IntToStr(i * 50) + '%';
    oMenuItem.Tag := i * 50;
    oMenuItem.OnClick := ZoomClick;
    pmZoom.Items.Add(oMenuItem);
  end;

  oItem := Toolbar.Items.Add;
  oItem.Kind := tikDropdown;
  oItem.Caption := 'Zoom';
  oItem.Glyph := 'Z';
  oItem.DropdownMenu := pmZoom;        // a VCL TPopupMenu
end;

procedure TFeatureForm.ZoomClick(Sender: TObject);
begin
  Memo.Font.Size := 10 * TMenuItem(Sender).Tag div 100;
end;
```

Voir aussi

- Types d'éléments
- Débordement
- Dépannage

Débordement

Lorsque les éléments ne tiennent pas dans la largeur de la barre, la mise en page déplace les derniers dans la liste de débordement et réserve un bouton de débordement, dessiné avec un double chevron, à l'extrémité droite. Une pression sur ce bouton affiche le **OverflowMenu** de la barre, un menu contextuel construit à l'exécution avec une entrée par élément de la liste. Choisir une entrée active cet élément exactement comme un clic : un bouton à bascule bascule, OnItemClick se déclenche et un élément déroulant affiche son menu.
La liste de débordement de la dernière mise en page est disponible par le modèle : **HasOverflow**, **OverflowCount**, **OverflowItem**[i] et la propriété **InOverflow** de chaque élément. ItemAtPos retourne CS_TOOLBAR_HIT_OVERFLOW au-dessus du bouton de débordement.

Édition Community

Le menu de débordement ne fait pas partie de l'édition Community (SGC_TOOLBAR_ADVANCED). Dans celle-ci, les éléments qui ne tiennent pas sont tronqués au bord droit et la propriété OverflowMenu n'existe pas. Voir Éditions.

Exemple

L'exemple est une méthode d'une fiche VCL avec un **Toolbar** (TsgcVCLToolbar), un **StatusBar** (TsgcVCLStatusBar) et un **Memo** (TMemo).

```
uses
  Windows, Types, Forms, Classes, SysUtils, Controls, StdCtrls, Menus,
  ActnList, ImgList, Graphics, Dialogs,
  sgcUICore_Theme, sgcToolbar_Model, sgcToolbar_VCL;

var
  i: Integer;
begin
  Toolbar.Width := 240;
  for i := 1 to 12 do
    Toolbar.Items.Add.Caption := 'Command ' + IntToStr(i);
  // after the next layout the items that do not fit are in the overflow list
  if Toolbar.Model.HasOverflow then
    StatusBar.SetPanelText(0, IntToStr(Toolbar.Model.OverflowCount) +
      ' items in the overflow menu');
end;
```

Voir aussi

- TsgcToolbarModel
- Menus déroulants
- Éditions

Liens d'action

La propriété **Action** lie un élément à n'importe quelle TBasicAction, une TAction VCL ou FireMonkey, sans que le modèle dépende d'un framework ou de l'autre. L'élément suit alors les propriétés **Caption**, **Enabled**, **Visible**, **Hint**, **ImageIndex** et **Checked** de l'action, où Checked pilote Down d'un élément tikToggle. Une TAction FireMonkey fournit son Text comme Caption.
Un clic exécute l'action, avec la barre comme ActionComponent, sauf si l'élément a un gestionnaire OnClick. Les barres appellent InitiateAction au repos, donc l'OnUpdate de chaque action liée s'exécute et une action désactivée désactive son bouton. À partir de Delphi XE3, l'action transmet ses modifications à l'élément. Avant XE3, l'élément copie les propriétés de l'action au repos.

Édition Community

Les liens d'action ne font pas partie de l'édition Community (SGC_TOOLBAR_ADVANCED) : la propriété Action n'existe pas. Utilisez à la place OnClick de l'élément ou OnItemClick de la barre. Voir Éditions.

Exemple

L'exemple est une méthode d'une fiche VCL avec un **Toolbar** (TsgcVCLToolbar), un **StatusBar** (TsgcVCLStatus-Bar) et un **Memo** (TMemo).

```
uses
  Windows, Types, Forms, Classes, SysUtils, Controls, StdCtrls, Menus,
  ActnList, ImgList, Graphics, Dialogs,
  sgcUICore_Theme, sgcToolbar_Model, sgcToolbar_VCL;

procedure TFeatureForm.Actions;
var
  oItem: TsgcToolbarItem;
begin
  ActionList := TActionList.Create(Self);
  actSave := TAction.Create(Self);
  actSave.ActionList := ActionList;
  actSave.Caption := 'Save';
  actSave.Hint := 'Save the document';
  actSave.OnExecute := SaveExecute;

  oItem := Toolbar.Items.Add;
  oItem.Glyph := 'S';
  // Caption, Hint, Enabled, Visible, ImageIndex and Checked follow the action
  oItem.Action := actSave;
  actSave.Enabled := False;            // the button is disabled too
end;

procedure TFeatureForm.SaveExecute(Sender: TObject);
begin
  Memo.Lines.SaveToFile('document.txt');
end;
```

Voir aussi

- TsgcToolbarItem
- Boutons à bascule et groupes d'options
- Dépannage

Images et glyphes

Chaque élément cliquable a un emplacement d'icône de 16 pixels logiques. Il affiche l'image à **ImageIndex** de la liste **Images** de la barre ou, lorsque l'élément n'a pas d'image, son **Glyph**: un court texte tel qu'une lettre ou un symbole. Un élément sans les deux n'affiche que sa légende.

- Sous VCL, Images est une TCustomImageList. Chaque image est convertie une fois en ARGB direct, avec l'alpha récupéré, et mise en cache jusqu'à ce que la liste d'images change.
- Sous FMX, Images est la TCustomImageList FireMonkey de FMX.ImgList et nécessite Delphi XE8 ou ultérieur. Les images sont prises à la taille de périphérique de la fenêtre. Sur les versions plus anciennes, les éléments dessinent leur Glyph.
- La sortie sans interface dessine le Glyph, sauf si l'événement OnGetImage du modèle retourne une image.

Exemple

L'exemple est une méthode d'une fiche VCL avec un **Toolbar** (TsgcVCLToolbar), un **StatusBar** (TsgcVCLStatusBar) et un **Memo** (TMemo).

```
uses
  Windows, Types, Forms, Classes, SysUtils, Controls, StdCtrls, Menus,
  ActnList, ImgList, Graphics, Dialogs,
  sgcUICore_Theme, sgcToolbar_Model, sgcToolbar_VCL;

var
  oBitmap: TBitmap;
  oItem: TsgcToolbarItem;
begin
  ImageList := TImageList.Create(Self);
  ImageList.Width := 16;
  ImageList.Height := 16;
  oBitmap := TBitmap.Create;
  try
    oBitmap.SetSize(16, 16);
    oBitmap.Canvas.Brush.Color := clNavy;
    oBitmap.Canvas.FillRect(Rect(0, 0, 16, 16));
    ImageList.Add(oBitmap, nil);
  finally
    oBitmap.Free;
  end;

  Toolbar.Images := ImageList;
  oItem := Toolbar.Items.Add;
  oItem.Caption := 'Print';
  oItem.Glyph := 'P';                  // drawn only when there is no image
  oItem.ImageIndex := 0;
end;
```

Voir aussi

- Types d'éléments
- Rendu sans interface
- Plateformes

Panneaux d'état

Une barre d'état dispose ses Panels de gauche à droite en trois passes :

- **Fixes** : les panneaux, sans AutoSize ni Spring, prennent leur **Width** en pixels logiques (100 par défaut).
- **AutoSize** : les panneaux prennent la largeur de leur texte, plus le point d'un indicateur.
- **Spring** : les panneaux se partagent la largeur restante, sans jamais descendre sous 0. Utilisez un panneau Spring pour le message principal.

Un fin trait sépare les panneaux. Le texte d'un panneau est aligné selon **Alignment**, et **Hint** s'affiche lorsque la souris s'arrête sur le panneau et que ShowHint vaut True. OnPanelClick signale un clic sur un panneau. Lorsque SizeGrip vaut True, la poignée est réservée à l'extrémité droite, voir Point de redimensionnement.

Exemple

L'exemple est une méthode d'une fiche VCL avec un **Toolbar** (TsgcVCLToolbar), un **StatusBar** (TsgcVCLStatus-Bar) et un **Memo** (TMemo).

```
uses
  Windows, Types, Forms, Classes, SysUtils, Controls, StdCtrls, Menus,
  ActnList, ImgList, Graphics, Dialogs,
  sgcUICore_Theme, sgcToolbar_Model, sgcToolbar_VCL;

var
  oPanel: TsgcStatusBarPanel;
begin
  StatusBar.Panels.Clear;

  oPanel := StatusBar.Panels.Add;      // Spring: takes the width that is left
  oPanel.Spring := True;
  oPanel.Text := 'Ready';

  oPanel := StatusBar.Panels.Add;      // AutoSize: as wide as its text
  oPanel.AutoSize := True;
  oPanel.Text := 'UTF-8';

  oPanel := StatusBar.Panels.Add;      // fixed: Width logical pixels
  oPanel.Width := 110;
  oPanel.Alignment := taRightJustify;
  oPanel.Text := 'Ln 1, Col 1';
  oPanel.Hint := 'Caret position';
end;
```

Voir aussi

- TsgcStatusBarPanel
- Progression et indicateurs
- Mises à jour thread-safe

Progression et indicateurs

Outre les panneaux de texte (spkText), un panneau peut être une barre de progression ou un indicateur d'état.

Type	Dessine
spkProgress	Une barre remplie à **Progress** pour cent (0 à 100), avec l'AccentColor de la barre d'état ou la couleur d'accent du thème.
spkIndicator	Un point coloré suivi du **Text** du panneau. La couleur provient de **IndicatorState**, ce qui convient à l'état d'une connexion ou d'un flux de données.

IndicatorState	Thème clair	Thème sombre
sisOff	La couleur de texte atténuée du thème	La couleur de texte atténuée du thème
sisOk	Vert $FF1A7F37	Vert $FF3FB950
sisWarning	Ambre $FFBF8700	Ambre $FFD29922
sisError	Rouge $FFCF222E	Rouge $FFF85149

GetIndicatorColor de TsgcStatusBarModel retourne la couleur d'un état dans le thème courant.

Exemple

L'exemple est une méthode d'une fiche VCL avec un **Toolbar** (TsgcVCLToolbar), un **StatusBar** (TsgcVCLStatusBar) et un **Memo** (TMemo).

```
uses
  Windows, Types, Forms, Classes, SysUtils, Controls, StdCtrls, Menus,
  ActnList, ImgList, Graphics, Dialogs,
  sgcUICore_Theme, sgcToolbar_Model, sgcToolbar_VCL;

var
  oPanel: TsgcStatusBarPanel;
begin
  oPanel := StatusBar.Panels.Add;
  oPanel.Kind := spkProgress;          // a bar in the accent colour
  oPanel.Width := 140;
  oPanel.Progress := 25;               // 0..100

  oPanel := StatusBar.Panels.Add;
  oPanel.Kind := spkIndicator;         // a coloured dot before the text
  oPanel.AutoSize := True;
  oPanel.Text := 'Feed';
  oPanel.IndicatorState := sisWarning; // sisOff, sisOk, sisWarning, sisError

  StatusBar.AccentColor := $FF8250DF;  // progress colour, $AARRGGBB
end;
```

Voir aussi

- Panneaux d'état
- Mises à jour thread-safe
- Thèmes

Mises à jour thread-safe

Une application en direct met à jour sa barre d'état depuis des threads de travail : un téléchargement, un flux, un long calcul. **SetPanelText**, **SetProgress** et **SetIndicator** de la barre d'état peuvent être appelés depuis n'importe quel thread, sans Synchronize ni Queue.

- Le modèle protège les Text, Progress et IndicatorState des panneaux par un verrou.
- Le setter déclenche OnChange du modèle depuis le thread appelant, et seulement lorsque la valeur a changé, donc répéter la même valeur ne coûte rien.
- La barre répond par une demande de repaint. Depuis un thread de travail, la barre VCL se contente d'envoyer un message à sa fenêtre et la barre FMX met le repaint en file d'attente vers le thread principal, de sorte que le framework n'est jamais touché hors du thread principal.
- Un index hors limites est ignoré, donc un thread qui survit à un panneau ne lève pas d'exception.

Toute autre propriété, ainsi que l'ajout, la suppression ou la configuration des panneaux, doit être faite depuis le thread principal. Assurez-vous que le thread de travail s'arrête avant que la fiche propriétaire de la barre d'état soit détruite.

Exemple

Un thread de travail qui simule un téléchargement de 50 blocs, comme dans la démo LiveStatus. Le panneau 0 est un panneau de texte Spring, le panneau 1 un panneau de progression et le panneau 2 un indicateur.

```
uses
  Windows, Types, Forms, Classes, SysUtils, Controls, StdCtrls, Menus,
  ActnList, ImgList, Graphics, Dialogs,
  sgcUICore_Theme, sgcToolbar_Model, sgcToolbar_VCL;

type
  TDownloadWorker = class(TThread)
  private
    FStatusBar: TsgcVCLStatusBar;
  protected
    procedure Execute; override;
  public
    constructor Create(aStatusBar: TsgcVCLStatusBar);
  end;

constructor TDownloadWorker.Create(aStatusBar: TsgcVCLStatusBar);
begin
  FStatusBar := aStatusBar;
  FreeOnTerminate := True;
  inherited Create(False);
end;

procedure TDownloadWorker.Execute;
var
  i: Integer;
begin
  // panel 0 text, panel 1 progress, panel 2 indicator
  FStatusBar.SetIndicator(2, sisOk);
  for i := 1 to 50 do
  begin
    if Terminated then
      Exit;
    Sleep(100);
    // no Synchronize: the setters are thread-safe and the bar
    // repaints in the main thread
    FStatusBar.SetPanelText(0, Format('Chunk %d of 50', [i]));
    FStatusBar.SetProgress(1, i * 2);
  end;
  FStatusBar.SetIndicator(2, sisOff);
end;
```

```
procedure TFeatureForm.StartWorker;
begin
  TDownloadWorker.Create(StatusBar);
end;
```

Voir aussi

- TsgcStatusBarModel
- Progression et indicateurs
- Démos

Point de redimensionnement

Avec **SizeGrip** True, la valeur par défaut, la barre d'état réserve une petite zone à son extrémité droite et y dessine les points de la poignée. **GripRect** retourne son rectangle, vide lorsque SizeGrip vaut False.

- Sous VCL, TsgcVCLStatusBar répond au test de position hors zone client de la poignée, donc la faire glisser redimensionne la fiche parente. Cela ne se produit que lorsque la fiche a une bordure redimensionnable et n'est pas agrandie, et jamais à la conception.
- Sous FMX, TsgcFMXStatusBar ne fait que dessiner la poignée. FireMonkey n'a pas de test de position hors zone client, donc la poignée ne redimensionne pas la fiche. Définissez SizeGrip à False lorsque la fiche n'est pas redimensionnable.

Exemple

L'exemple est une méthode d'une fiche VCL avec un **Toolbar** (TsgcVCLToolbar), un **StatusBar** (TsgcVCLStatus-Bar) et un **Memo** (TMemo).

```
uses
  Windows, Types, Forms, Classes, SysUtils, Controls, StdCtrls, Menus,
  ActnList, ImgList, Graphics, Dialogs,
  sgcUICore_Theme, sgcToolbar_Model, sgcToolbar_VCL;

var
  vGrip: TRect;
begin
  // the grip resizes the form when the form has a sizeable border
  BorderStyle := bsSizeable;
  StatusBar.SizeGrip := True;
  vGrip := StatusBar.GripRect;         // client rectangle, empty without grip
  Caption := Format('Grip at %d,%d', [vGrip.Left, vGrip.Top]);
end;
```

Voir aussi

- Panneaux d'état
- TsgcVCLStatusBar

Clavier

On accède à la barre d'outils avec la touche Tab lorsque **TabStop** vaut True sous VCL, ou que **CanFocus** vaut True sous FMX. Les deux valent False par défaut, donc la barre d'outils ne prend pas le focus de l'éditeur situé en dessous. Lorsque la barre d'outils a le focus, le bouton ciblé a un anneau de focus :

Touche	Action
Gauche, Droite	Passe au bouton, au bouton à bascule ou à l'élément déroulant activé précédent ou suivant.
Début, Fin	Passe au premier ou au dernier.
Entrée, Espace	Active l'élément ciblé, comme le ferait un clic.

ClickItem active un élément depuis le code, par exemple depuis un raccourci clavier de la fiche. La barre d'état n'a pas d'interface clavier.

Exemple

L'exemple est une méthode d'une fiche VCL avec un **Toolbar** (TsgcVCLToolbar), un **StatusBar** (TsgcVCLStatusBar) et un **Memo** (TMemo).

```
uses
  Windows, Types, Forms, Classes, SysUtils, Controls, StdCtrls, Menus,
  ActnList, ImgList, Graphics, Dialogs,
  sgcUICore_Theme, sgcToolbar_Model, sgcToolbar_VCL;

begin
  // Tab moves the focus to the bar; Left, Right, Home and End move between
  // the buttons, Enter and Space activate the focused one
  Toolbar.TabStop := True;
  Toolbar.TabOrder := 0;
  // activates the first item from code, exactly as a click would
  Toolbar.ClickItem(0);
end;
```

Voir aussi

- TsgcVCLToolbar
- TsgcFMXToolbar

Thèmes

Les barres prennent leurs couleurs dans les jetons de thème sgcUICore, les mêmes que ceux des autres produits de la suite. **Theme** vaut utLight (par défaut), utDark ou utCustom, qui utilise les couleurs claires. Chaque barre a son propre Theme, définissez-le donc sur la barre d'outils et sur la barre d'état.

- Le fond est la couleur Surface, avec un trait Border sous la barre d'outils et au-dessus de la barre d'état.
- Les boutons survolés et enfoncés reçoivent une surcouche, un bouton à bascule enfoncé une teinte d'accent, le bouton ciblé un anneau de focus et un élément désactivé la couleur de texte atténuée.
- **AccentColor**, sous la forme $AARRGGBB, remplace la couleur d'accent du thème : l'anneau de focus et la teinte enfoncée sur la barre d'outils, les barres de progression sur la barre d'état. 0 utilise la couleur d'accent du thème.
- Les couleurs des indicateurs suivent le thème, voir Progression et indicateurs.

Exemple

L'exemple est une méthode d'une fiche VCL avec un **Toolbar** (TsgcVCLToolbar), un **StatusBar** (TsgcVCLStatusBar) et un **Memo** (TMemo).

```
uses
  Windows, Types, Forms, Classes, SysUtils, Controls, StdCtrls, Menus,
  ActnList, ImgList, Graphics, Dialogs,
  sgcUICore_Theme, sgcToolbar_Model, sgcToolbar_VCL;

begin
  Toolbar.Theme := utDark;
  StatusBar.Theme := utDark;
  // 0 (the default) uses the accent colour of the theme
  Toolbar.AccentColor := $FF1F883D;
  StatusBar.AccentColor := $FF1F883D;
end;
```

Voir aussi

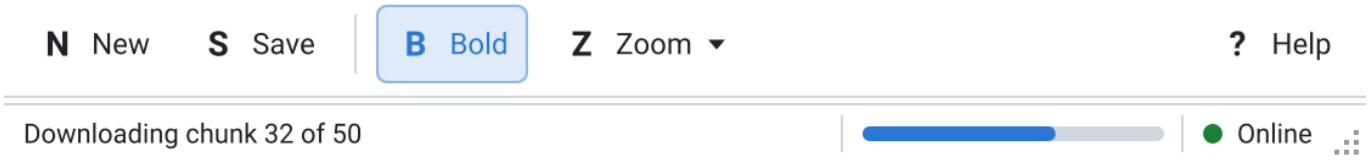
- Progression et indicateurs
- Architecture

Rendu sans interface

Le moteur de rendu de sgcToolbar_Render dessine un TsgcToolbarModel ou un TsgcStatusBarModel sans fenêtre et sans VCL ni FMX, de sorte qu'une application console, un service, un test ou un serveur web peut produire les mêmes barres sous forme d'image, par exemple pour la documentation d'une application ou un test visuel. Les unités se compilent sur toutes les plateformes, Linux64 compris.

Routine	Description
Model.SaveToPNG(aFileName or aStream, aWidth, aHeight, aScale = 1)	Écrit un fichier ou un flux PNG. aWidth et aHeight sont en pixels logiques et aScale les multiplie : 560 sur 44 à l'échelle 2 donne une image de 1120 sur 88.
Model.SaveToSVG(aFileName or aStream, aWidth, aHeight, aScale = 1)	Écrit un fichier ou un flux SVG de la même barre.
Model.Render(aCanvas, aWidth, aHeight, aScale = 1)	Dessine la barre sur n'importe quel canvas sgcUICore, en pixels de périphérique.
RenderToolbar(aModel, aCanvas, aWidth, aHeight, aScale, aImages)	Le moteur de rendu de sgcToolbar_Render. aImages retourne l'image d'un ImageIndex, nil dessine le Glyph.
RenderStatusBar(aModel, aCanvas, aWidth, aHeight, aScale)	Le moteur de rendu de la barre d'état.

Une barre d'outils mesure ButtonSize plus deux fois Padding de haut, soit 44 pixels logiques avec les valeurs par défaut. Le modèle d'un contrôle est disponible par sa propriété **Model** , donc **Toolbar.Model.SaveToPNG(...)** enregistre ce que le contrôle affiche, à n'importe quelle taille.



Exemple

```
uses
  SysUtils,
  sgcUICore_Theme, sgcToolbar_Model;

var
  oToolbar: TsgcToolbarModel;
  oStatus: TsgcStatusBarModel;
  oItem: TsgcToolbarItem;
  oPanel: TsgcStatusBarPanel;
begin
  oToolbar := TsgcToolbarModel.Create;
  oStatus := TsgcStatusBarModel.Create;
  try
    oItem := oToolbar.Items.Add;
    oItem.Caption := 'New';
    oItem.Glyph := 'N';
    oItem := oToolbar.Items.Add;
    oItem.Caption := 'Save';
    oItem.Glyph := 'S';
    oToolbar.Items.Add.Kind := tikSeparator;
    oItem := oToolbar.Items.Add;
    oItem.Kind := tikToggle;
```

```
    oItem.Caption := 'Bold';
    oItem.Glyph := 'B';
    oItem.Down := True;
    oItem := oToolbar.Items.Add;
    oItem.Kind := tikDropdown;
    oItem.Caption := 'Zoom';
    oItem.Glyph := 'Z';
    oToolbar.Items.Add.Kind := tikSpacer;
    oItem := oToolbar.Items.Add;
    oItem.Caption := 'Help';
    oItem.Glyph := '?';

    oPanel := oStatus.Panels.Add;
    oPanel.Spring := True;
    oPanel.Text := 'Downloading chunk 32 of 50';
    oPanel := oStatus.Panels.Add;
    oPanel.Kind := spkProgress;
    oPanel.Width := 140;
    oPanel.Progress := 64;
    oPanel := oStatus.Panels.Add;
    oPanel.Kind := spkIndicator;
    oPanel.AutoSize := True;
    oPanel.Text := 'Online';
    oPanel.IndicatorState := sisOk;

    // 560 x 44 logical pixels (ButtonSize 32 + 2 x Padding 6),
    // written at scale 2 as 1120 x 88 pixels
    oToolbar.SaveToPNG('toolbar.png', 560, 44, 2);
    oToolbar.SaveToSVG('toolbar.svg', 560, 44);
    oStatus.SaveToPNG('statusbar.png', 560, 24, 2);
    oStatus.SaveToSVG('statusbar.svg', 560, 24);
  finally
    oStatus.Free;
    oToolbar.Free;
  end;
end;
```

Voir aussi

- TsgcToolbarModel
- TsgcStatusBarModel
- Plateformes

Présentation de .NET

sgcToolbar .NET est un portage managé en C# de la bibliothèque sgcToolbar de Delphi, et non une enveloppe autour d'elle. Les classes C# se trouvent dans l'espace de noms **esegece.sgcToolbar** et gardent les noms de Delphi, de sorte que **TsgcToolbarModel**, **TsgcToolbarItem**, **TsgcToolbarItemKind**, **TsgcStatusBarModel** et **TsgcStatusBarPanel** se lisent de la même façon dans les deux langages, et la barre d'outils et la barre d'état de chaque hôte suivent le modèle de nommage de Delphi. Elles sont dessinées avec le portage .NET de **sgcUICore**, avec le même canevas, les mêmes polices et les mêmes thèmes clair et sombre que les autres produits de la suite visuelle eSeGeCe, si bien qu'une barre a le même aspect dans WinForms, WPF, Avalonia et dans la sortie sans interface en PNG et SVG.
Une barre d'outils contient des boutons, des boutons à bascule, des groupes d'options, des boutons déroulants, des séparateurs, des espaceurs et des libellés, et déplace les éléments qui ne tiennent pas dans un menu de débordement. Une barre d'état contient des panneaux de texte, de progression et d'indicateur, de largeur fixe, automatique ou élastique, et ses panneaux peuvent être mis à jour depuis n'importe quel thread.

Packages

Le portage est réparti en cinq packages NuGet, de sorte qu'un projet n'intègre que l'hôte qu'il utilise.

PackageId	Contenu	Frameworks cibles
esegece.sgcToolbar	Les modèles (**TsgcToolbarModel**, **TsgcStatusBarModel**), leurs éléments et panneaux, le lien **Command** de l'élément, le moteur de rendu **sgcToolbar_Render** et la sortie SVG sans interface. Aucune dépendance native. Dépend de esegece.sgcUICore.	netstandard2.0, net472, net6.0, net8.0, net9.0
esegece.sgcToolbar.Rendering	La sortie PNG sans interface, les méthodes d'extension **SaveToPNG** de **sgcToolbar_Raster**, dessinées sur le canevas SkiaSharp de esegece.sgcUICore.Rendering, sous Windows, Linux et macOS.	netstandard2.0, net472, net6.0, net8.0, net9.0
esegece.sgcToolbar.WinForms	**TsgcWinFormsToolbar** et **TsgcWinFormsStatusBar**.	net472, net8.0-windows, net9.0-windows
esegece.sgcToolbar.Wpf	**TsgcWpfToolbar** et **TsgcWpfStatusBar**.	net472, net8.0-windows, net9.0-windows
esegece.sgcToolbar.Avalonia	**TsgcAvaloniaToolbar** et **TsgcAvaloniaStatusBar**, sous Windows, Linux et macOS avec Avalonia 11.	net8.0, net9.0

Les trois packages d'hôte dépendent de esegece.sgcToolbar.Rendering, qui dépend de esegece.sgcToolbar, de sorte qu'une seule ligne `PackageReference` apporte tout ce dont un hôte a besoin. Voir Installation depuis NuGet.

Éditions

Édition	Identifiants de package	Comprend
Registered	esegece.sgcToolbar, esegece.sgcToolbar.Rendering, esegece.sgcToolbar.WinForms, esegece.sgcToolbar.Wpf, esegece.sgcToolbar.Avalonia	Tout : les barres d'outils et d'état des trois hôtes, tous les types d'éléments et de panneaux, les groupes d'options, les menus déroulants, le menu de débordement, le lien Command de l'élément, les images, les info-bulles, la navigation au clavier, les setters thread-safe de la barre d'état, le point de redimensionnement, les thèmes et la sortie sans interface en PNG et SVG.
Community	Les mêmes identifiants avec le suffixe .Community, par exemple esegece.sgcToolbar.WinForms.Community. Ils dépendent des packages Community de sgcUICore.	Le même contenu que Registered sans le menu de débordement et sans les propriétés **Command** et **CommandParameter** de l'élément, comme l'édition Community de Delphi. Les éléments qui ne tiennent pas sont tronqués au bord droit, et les éléments exécutent leur OnClick.

L'édition Community est gratuite pour un usage commercial et non commercial par des particuliers et des organisations dont le chiffre d'affaires annuel brut est inférieur à 1 000 000 EUR et qui comptent au plus cinq développeurs. Une organisation qui atteint ou dépasse l'un des deux seuils a besoin d'une licence Registered. Le portage .NET n'a pas d'édition Trial.

Hôtes

Hôte	Équivalent Delphi	Plateformes
WinForms	VCL (TsgcVCLToolbar, TsgcVCLStatusBar)	Windows, .NET Framework 4.7.2 et .NET 8 et 9.
WPF	FMX (TsgcFMXToolbar, TsgcFMXStatusBar)	Windows, .NET Framework 4.7.2 et .NET 8 et 9. Les barres sont déclarées en XAML et les éléments et panneaux sont ajoutés dans le code.
Avalonia	FMX (TsgcFMXToolbar, TsgcFMXStatusBar)	Windows, Linux et macOS, .NET 8 et 9, avec la souris, le tactile et le stylet. Les barres sont déclarées en XAML et les éléments et panneaux sont ajoutés dans le code.
Headless	Les méthodes Render et SaveToSVG des modèles	Toute plateforme des frameworks cibles : SVG avec esegece.sgcToolbar, PNG avec esegece.sgcToolbar.Rendering.

Toute barre d'outils a la même API : Items, ShowCaptions, ButtonSize, Spacing, Padding, Theme, AccentColor, Images, OverflowMenu, ItemAtPos, ItemRect, OverflowRect, ClickItem et OnItemClick. Toute barre d'état a Panels, SizeGrip, Theme, AccentColor, PanelAtPos, PanelRect, GripRect, les méthodes thread-safe SetPanelText, SetProgress et SetIndicator, et OnPanelClick. Voir Classes .NET pour les noms des classes et Différences de .NET avec Delphi pour ce qui change par rapport à Delphi.

Voir aussi

- Installation depuis NuGet
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia

- Classes .NET
- Différences de .NET avec Delphi
- Rendu sans interface en .NET
- Démos .NET
- Plateformes

Installation depuis NuGet

Les packages sont sur nuget.org. Ajoutez le package de l'hôte, les packages core et de rendu arrivent avec lui comme dépendances.

Édition Registered

```
dotnet add package esegece.sgcToolbar.WinForms
dotnet add package esegece.sgcToolbar.Wpf
dotnet add package esegece.sgcToolbar.Avalonia
```

Édition Community

```
dotnet add package esegece.sgcToolbar.WinForms.Community
dotnet add package esegece.sgcToolbar.Wpf.Community
dotnet add package esegece.sgcToolbar.Avalonia.Community
```

Ou ajoutez la référence au fichier de projet à la main :

```
<ItemGroup>
  <PackageReference Include="esegece.sgcToolbar.Wpf" Version="2026.10.0" />
</ItemGroup>
```

Référencez dans un projet soit les packages Registered, soit les packages Community, jamais les deux : ils contiennent les mêmes types dans le même espace de noms. Le code qui utilise le **Command** de l'élément ou la propriété **OverflowMenu** d'une barre d'outils ne compile pas avec les packages Community ; protégez-le avec `#if !SGC_COMMUNITY` lorsqu'un projet compile les deux éditions.

Paramètres du projet

- WinForms : un projet `net8.0-windows` avec `<UseWindowsForms>true</UseWindowsForms>`, ou un projet .NET Framework 4.7.2.
- WPF : un projet `net8.0-windows` avec `<UseWPF>true</UseWPF>`, ou un projet .NET Framework 4.7.2. Associez l'espace de noms en XAML avec `xmlns:sgc="clr-namespace:esegece.sgcToolbar;assembly=esegece.sgcToolbar.Wpf"`.
- Avalonia : un projet `net8.0` ou `net9.0` avec les packages **Avalonia.Desktop** et **Avalonia.Themes.Fluent** d'Avalonia 11.3, comme toute application de bureau Avalonia. Associez l'espace de noms en XAML avec `xmlns:sgc="using:esegece.sgcToolbar"`.
- Headless : esegece.sgcToolbar seul suffit pour le SVG. Ajoutez esegece.sgcToolbar.Rendering pour le PNG, il apporte SkiaSharp et ses ressources natives pour Windows et Linux.

En C# tout hôte a besoin de `using esegece.sgcToolbar;`, et de `using esegece.sgcUICore;` pour **TsgcUITheme** et les autres types des thèmes.

Voir aussi

- Présentation de .NET
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Démos .NET

Démarrage rapide WinForms

TsgcWinFormsToolbar et **TsgcWinFormsStatusBar** sont les barres WinForms, les équivalents de TsgcVCLToolbar et TsgcVCLStatusBar de Delphi. Chacune possède un modèle, expose ses propriétés et le dessine via sgcUICore. Le package est **esegece.sgcToolbar.WinForms**.

Étapes

- Créez un projet WinForms et ajoutez le package esegece.sgcToolbar.WinForms.
- Créez un **TsgcWinFormsToolbar**. Il s'ancre en haut par défaut et sa hauteur suit **ButtonSize** et **Padding**.
- Ajoutez les éléments avec `Items.Add()` et définissez leur **Kind**, **Caption**, **Glyph** ou **ImageIndex**, et **Hint**. Un **tikToggle** avec le même **GroupIndex** forme un groupe d'options, et un **tikDropdown** affiche le **ContextMenuStrip** de son **DropdownMenu**.
- Créez un **TsgcWinFormsStatusBar**, qui s'ancre en bas, et ajoutez ses panneaux avec `Panels.Add()`.
- Affectez **OnItemClick** pour réagir à chaque clic, ou **OnClick** de chaque élément.

Exemple

Le Program.cs complet d'un éditeur WinForms avec une barre d'outils et une barre d'état : deux boutons, un bouton à bascule Bold, un groupe d'options Left et Center, un menu déroulant Zoom, et un panneau de texte élastique, un panneau de progression et un indicateur. Il compile tel quel dans un projet `net8.0-windows` qui référence esegece.sgcToolbar.WinForms.
Les barres figurent aussi dans la boîte à outils. Les éléments et les panneaux ne sont pas sérialisés par le concepteur, ils sont donc toujours ajoutés dans le code, voir Différences de .NET avec Delphi. Définissez **TabStop** à True pour déplacer le focus entre les boutons avec les touches fléchées.

Voir aussi

- Présentation de .NET
- Classes .NET
- Différences de .NET avec Delphi
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Types d'éléments

Démarrage rapide WPF

TsgcWpfToolbar et **TsgcWpfStatusBar** sont les barres WPF. Elles sont déclarées en XAML comme n'importe quel contrôle, et leurs éléments et panneaux sont ajoutés dans le code. Le package est **esegece.sgcToolbar.Wpf**.

Étapes

- Créez un projet WPF (`dotnet new wpf`) et ajoutez le package esegece.sgcToolbar.Wpf.
- Associez l'espace de noms `esegece.sgcToolbar` et déclarez un **TsgcWpfToolbar** et un **TsgcWpfStatusBar**. **ShowCaptions**, **ButtonSize**, **SizeGrip** et les autres propriétés simples peuvent être définies comme attributs XAML.
- Dans le code-behind, affectez **Images**, une liste de **ImageSource**, ajoutez les éléments et les panneaux et affectez **OnItemClick** : c'est un délégué, pas un événement WPF, il ne peut donc pas être défini comme attribut XAML.

Exemple

MainWindow.xaml déclare la barre d'outils, la barre d'état et une zone de texte. Le App.xaml du projet est celui du modèle WPF, avec StartupUri défini sur MainWindow.xaml.
MainWindow.xaml.cs construit trois icônes vectorielles **DrawingImage**, ajoute les boutons, un bouton à bascule Wrap et un menu déroulant Zoom avec un **ContextMenu** WPF, ainsi qu'un panneau de texte et un indicateur.
Images accepte toute **ImageSource**, aussi bien un **BitmapImage** qu'un **DrawingImage**. Lorsque la liste est une **ObservableCollection<ImageSource>**, la barre d'outils se redessine quand une image est ajoutée, supprimée ou remplacée.

Voir aussi

- Présentation de .NET
- Classes .NET
- Différences de .NET avec Delphi
- Démarrage rapide WinForms
- Démarrage rapide Avalonia
- Images et glyphes

Démarrage rapide Avalonia

TsgcAvaloniaToolbar et **TsgcAvaloniaStatusBar** sont les barres Avalonia, les équivalents de TsgcFMXToolbar et TsgcFMXStatusBar de Delphi, sous Windows, Linux et macOS. La souris, le tactile et le stylet passent par le même chemin de pointeur. Le package est **esegece.sgcToolbar.Avalonia**.

Étapes

- Créez un projet .NET 8 et ajoutez les packages esegece.sgcToolbar.Avalonia, Avalonia.Desktop et Avalonia.Themes.Fluent.
- Associez l'espace de noms avec `xmlns:sgc="using:esegece.sgcToolbar"` et déclarez un **TsgcAvaloniaToolbar** et un **TsgcAvaloniaStatusBar**.
- Dans le code-behind, définissez le **Theme**, ajoutez les éléments et les panneaux et affectez **OnItemClick** : c'est un délégué, pas un événement Avalonia.

Exemple

MainWindow.axaml déclare les deux barres et un texte.
MainWindow.axaml.cs définit le thème sombre et ajoute un groupe d'options Grid et List, un espaceur qui repousse les éléments suivants vers la droite, un libellé, un bouton Refresh, un panneau de texte élastique et un panneau de progression.
Program.cs démarre l'application avec le thème Fluent.
Theme masque la propriété `StyledElement.Theme` des contrôles Avalonia, définissez-la donc dans le code sur une variable du type de la barre, et non comme attribut XAML. Images est une liste de **IImage**, un **Bitmap** ou un **DrawingImage**.

Voir aussi

- Présentation de .NET
- Classes .NET
- Différences de .NET avec Delphi
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Boutons à bascule et groupes d'options

Classes .NET

Chaque hôte a une classe de barre d'outils et une classe de barre d'état avec les mêmes membres. Les types indépendants du framework sont partagés par les trois hôtes et gardent leurs noms Delphi.

Classes d'hôte

Hôte	Classe	Classe de base	Delphi
WinForms	TsgcWinFormsToolbar	TsgcUIWinFormsControl	TsgcVCLToolbar
WinForms	TsgcWinFormsStatusBar	TsgcUIWinFormsControl	TsgcVCLStatusBar
WPF	TsgcWpfToolbar	TsgcUIWpfControl	TsgcFMXToolbar
WPF	TsgcWpfStatusBar	TsgcUIWpfControl	TsgcFMXStatusBar
Avalonia	TsgcAvaloniaToolbar	TsgcUIAvaloniaControl	TsgcFMXToolbar
Avalonia	TsgcAvaloniaStatusBar	TsgcUIAvaloniaControl	TsgcFMXStatusBar

Types indépendants du framework

Type	Description
TsgcToolbarModel	Les éléments, la mise en page avec la liste de débordement, les éléments survolés, enfoncés et ayant le focus, et le test de positionnement. Toute barre d'outils en possède un, voir TsgcToolbarModel.
TsgcToolbarItems	La collection d'éléments : Count, l'indexeur, Add, Insert, Delete, Clear, IndexOf, BeginUpdate et EndUpdate.
TsgcToolbarItem	Un élément : Kind, Caption, Glyph, ImageIndex, Hint, Enabled, Visible, Down, GroupIndex, Width, Tag, DropdownMenu, OnClick, Click(), les résultats de la mise en page BoundsRect, Shown et InOverflow, et dans l'édition Registered Command et CommandParameter. Voir TsgcToolbarItem.
TsgcToolbarItemKind	tikButton, tikToggle, tikDropdown, tikSeparator, tikSpacer, tikLabel.
TsgcStatusBarModel	Les panneaux, leur mise en page, le point de redimensionnement et les setters thread-safe. Toute barre d'état en possède un, voir TsgcStatusBarModel.
TsgcStatusBarPanels	La collection de panneaux : Count, l'indexeur, Add, Insert, Delete, Clear, IndexOf, BeginUpdate et EndUpdate.
TsgcStatusBarPanel	Un panneau : Kind, Text, Width, AutoSize, Spring, Alignment, Progress, IndicatorState, Hint, Tag et BoundsRect. Voir TsgcStatusBarPanel.
TsgcStatusPanelKind	spkText, spkProgress, spkIndicator.
TsgcStatusIndicatorState	sisOff, sisOk, sisWarning, sisError.

TAlignment	taLeftJustify, taRightJustify, taCenter : l'alignement du texte d'un panneau, comme le type de la RTL de Delphi.
TsgcToolbarItemEvent	`void (object Sender, TsgcToolbarItem aItem)` , le type de OnItemClick.
TsgcStatusBarPanelEvent	`void (object Sender, TsgcStatusBarPanel aPanel)`, le type de OnPanelClick.
sgcToolbar_Render, sgcToolbar_Raster	Le moteur de rendu et la sortie PNG sans interface, voir Rendu sans interface en .NET.

Barre d'outils

Membre	Type	Description
Model	TsgcToolbarModel	Le modèle de la barre d'outils.
Items	TsgcToolbarItems	Les éléments, dans l'ordre de gauche à droite.
ShowCaptions	bool	Dessine le libellé des boutons à côté de leur icône. Sans elle, un bouton avec une icône n'affiche que l'icône.
ButtonSize, Spacing, Padding	int	La hauteur d'un bouton, l'espace entre les éléments et la marge intérieure autour d'eux, 32, 4 et 6 par défaut. La hauteur de la barre d'outils suit ButtonSize et Padding.
Theme, AccentColor	TsgcUITheme, uint	Le thème de la barre. AccentColor 0 utilise l'accent du thème.
Images	ImageList, IList<ImageSource>, IList<IImage>	Les images de ImageIndex, voir Différences de .NET avec Delphi.
OverflowMenu	ContextMenuStrip, ContextMenu	Le menu du bouton de débordement, reconstruit à chaque ouverture. Édition Registered uniquement.
TabStop (WinForms), Focusable (WPF, Avalonia)	bool	Permet à la barre de prendre le focus : Left, Right, Home et End déplacent le bouton ayant le focus, Enter et Space l'activent.
ItemAtPos(X, Y)	int	L'index de l'élément en un point du contrôle, -1, ou CS_TOOLBAR_HIT_OVERFLOW au-dessus du bouton de débordement.
ItemRect(aIndex), OverflowRect()	Rectangle, Rect	Le rectangle d'un élément ou du bouton de débordement dans les coordonnées du contrôle.
ClickItem(aIndex)	void	Active un élément comme le fait un clic.
OnItemClick	TsgcToolbarItemEvent	Un clic, Enter ou Space sur un bouton activé, après qu'un bouton à bascule a inversé Down et avant l'exécution du OnClick ou du Command de l'élément.

Barre d'état

Membre	Type	Description

Model	TsgcStatus-BarModel	Le modèle de la barre d'état.
Panels	TsgcStatus-BarPanels	Les panneaux, dans l'ordre de gauche à droite.
SizeGrip	bool	Dessine le point de redimensionnement à l'extrémité droite et redimensionne la fenêtre lorsqu'on le fait glisser, voir Différences de .NET avec Delphi.
Theme, AccentColor	TsgcUI-Theme, uint	Le thème de la barre. AccentColor 0 utilise l'accent du thème.
SetPanelText(aIndex, aText), SetProgress(aIndex, aValue), SetIndicator(aIndex, aState)	void	Définissent le Text, le Progress (0 à 100) et l'IndicatorState d'un panneau depuis n'importe quel thread. Un index hors limites est ignoré.
PanelAtPos(X, Y)	int	L'index du panneau en un point du contrôle, ou -1.
PanelRect(aIndex), GripRect()	Rectangle, Rect	Le rectangle d'un panneau ou du point de redimen-sionnement dans les coordonnées du contrôle, vide sans SizeGrip.
OnPanelClick	TsgcStatus-BarPane-lEvent	Un clic sur un panneau.
CS_STATUSBAR_HEIGHT	const int	La hauteur par défaut, 24.

Les positions et les tailles sont en pixels dans WinForms (Rectangle) et en unités indépendantes de l'appareil dans WPF et Avalonia (Rect).

Voir aussi

- Présentation de .NET
- Différences de .NET avec Delphi
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia

Différences de .NET avec Delphi

L'API .NET suit l'API Delphi nom pour nom. Voici les endroits où elle diffère.

Command à la place des liens d'action

Delphi relie un élément à une TBasicAction par sa propriété Action, voir Liens d'action. .NET n'a pas de TBasicAction, l'élément a donc **Command**, un `System.Windows.Input.ICommand`, et **CommandParameter**. Les deux font partie de l'édition Registered uniquement, comme les liens d'action de Delphi.

- `CanExecute(CommandParameter)` pilote le **Enabled** de l'élément : lorsque la commande est affectée et à chaque `CanExecuteChanged`.
- Un clic, Enter ou Space inverse un bouton à bascule, déclenche OnItemClick de la barre, puis exécute `Click()` de l'élément : son gestionnaire OnClick lorsqu'il y en a un, sinon la commande, si elle peut s'exécuter.
- Caption, Hint et Down ne sont pas synchronisés depuis la commande, car ICommand n'en porte aucun. Ils restent des propriétés de l'élément, de sorte qu'un bouton à bascule conserve son propre état Down. ImageIndex et Visible ne sont pas synchronisés non plus.
- Il n'y a pas de OnUpdate au repos. Une commande qui change d'état déclenche CanExecuteChanged, et l'élément la suit aussitôt.

Un programme console qui relie un élément Save à une commande minimale. L'élément démarre désactivé, devient activé quand le document change, et un clic enregistre puis le désactive de nouveau. Il n'a besoin que de esegece.sgcToolbar, les barres d'outils des trois hôtes se comportent de la même façon.

Menus déroulants et menus de débordement

Dans Delphi, **DropdownMenu** est un TPopupMenu de la VCL ou de FireMonkey, voir Menus déroulants. Dans .NET il est déclaré comme `object`, afin que le modèle reste indépendant du framework, et chaque hôte affiche sous le bouton le type de menu de son framework. **OverflowMenu** est construit par la barre d'outils à chaque ouverture, avec une entrée par élément de la liste de débordement, et choisir une entrée active l'élément comme le fait un clic, voir Débordement.

Hôte	DropdownMenu	OverflowMenu
WinForms	System.Windows.Forms.ContextMenuStrip. Un menu libéré est retiré de son élément.	ContextMenuStrip, le Hint d'un élément est le ToolTipText de son entrée.
WPF	System.Windows.Controls.ContextMenu.	ContextMenu, avec un Separator pour un élément séparateur et le Hint comme ToolTip d'une entrée.
Avalonia	Avalonia.Controls.ContextMenu.	ContextMenu, avec un Separator pour un élément séparateur et le Hint comme info-bulle d'une entrée.

Le menu s'ouvre lorsque le bouton est enfoncé, après OnItemClick et le OnClick de l'élément, de sorte qu'un gestionnaire peut remplir ou cocher le menu avant son affichage. Le menu de débordement ne fait pas partie de l'édition Community : les éléments qui ne tiennent pas y sont tronqués et la propriété OverflowMenu n'existe pas.

Images

Delphi utilise une TCustomImageList dans la VCL et une TCustomImageList de FMX.ImgList dans FireMonkey, voir Images et glyphes. Dans .NET, la propriété **Images** de la barre d'outils a le type d'image de chaque hôte, et le **ImageIndex** de l'élément est un index dans celle-ci.

Hôte	Images	Dessinée à

Win-Forms	System.Windows.Forms.ImageList	Le ImageSize de la liste d'images, en pixels.
WPF	IList<ImageSource> : bitmaps et icônes vectorielles DrawingImage. Les icônes vectorielles sont rastérisées avec Skia, elles se dessinent donc aussi dans une session Bureau à distance réduite ou déconnectée. Une ObservableCollection se redessine quand elle change.	16 x 16 unités indépendantes de l'appareil, nette à tout DPI pour les icônes vectorielles.
Avalonia	IList<IImage> : un Bitmap ou un DrawingImage. Une liste avec INotifyCollectionChanged se redessine quand elle change.	16 x 16 unités indépendantes de l'appareil.

Chaque image est convertie une fois en ARGB et mise en cache jusqu'à ce que la liste change, et dans WPF et Avalonia aussi jusqu'à ce que l'échelle DPI change. Lorsqu'un élément n'a pas d'image, le texte de son **Glyph** est dessiné dans l'emplacement de l'icône.

Info-bulles

Dans Delphi, les barres affichent le Hint de l'élément ou du panneau sous le curseur lorsque ShowHint vaut True, et FMX affiche les info-bulles à partir de 10.1 Berlin. Dans .NET, les barres affichent toujours une info-bulle, sans propriété ShowHint : le Hint de l'élément sous le curseur, ou son Caption lorsque le libellé n'est pas dessiné, et dans la barre d'état le Hint du panneau. WinForms utilise un composant ToolTip interne, WPF définit la propriété ToolTip de la barre et Avalonia définit la propriété attachée ToolTip.Tip. Ne définissez pas vous-même l'info-bulle d'une barre, la barre la remplace ; définissez plutôt le Hint de l'élément ou du panneau.

Mises à jour d'état thread-safe

Comme dans Delphi, **SetPanelText**, **SetProgress** et **SetIndicator** de la barre d'état peuvent être appelés depuis n'importe quel thread, voir Mises à jour thread-safe. Le modèle protège les valeurs par un verrou et déclenche son OnChange dans le thread appelant, uniquement lorsque la valeur a changé. La barre répond par une demande de redessin et, depuis un thread de travail, ne touche jamais au contrôle :

- WinForms et WPF postent un rappel via le `SynchronizationContext` que le contrôle a capturé dans le thread de l'interface.
- Avalonia poste un rappel vers `Dispatcher.UIThread`.
- Un seul rappel est en attente à la fois, de sorte qu'un thread de travail qui met à jour la barre des milliers de fois par seconde n'inonde pas la file de l'interface. Le rappel redessine avec les dernières valeurs.

Aucun appel à Invoke, BeginInvoke ou Dispatcher n'est nécessaire. Toutes les autres propriétés, ainsi que l'ajout, la suppression ou la configuration des panneaux, sont à effectuer depuis le thread de l'interface. Le formulaire ci-dessous télécharge 50 blocs simulés dans un thread de travail et arrête le thread avant la fermeture du formulaire.

Point de redimensionnement

Avec **SizeGrip** à True, la valeur par défaut, la barre d'état dessine les points de la poignée à son extrémité droite et **GripRect** renvoie leur rectangle, voir Point de redimensionnement. Dans Delphi, seule la barre VCL redimensionne la fiche ; la barre FMX ne fait que dessiner la poignée. Dans .NET, la poignée redimensionne la fenêtre dans chaque hôte, lorsque la fenêtre est redimensionnable et non agrandie, et jamais à la conception :

Hôte	Comment la poignée redimensionne la fenêtre
Win-Forms	La barre répond au test de positionnement non client de la poignée par HTBOTTOMRIGHT, comme la barre VCL, pour une fiche à bordure redimensionnable.
WPF	Un appui sur la poignée démarre la boucle de redimensionnement système de la fenêtre (WM_SYSCOMMAND avec SC_SIZE), pour une Window dont ResizeMode vaut CanResize ou CanResizeWithGrip. Le curseur change au-dessus de la poignée.
Avalonia	Un appui sur la poignée appelle BeginResizeDrag de la fenêtre, pour une Window avec CanResize, sous Windows, Linux et macOS.

Unités

- WinForms : ButtonSize, Spacing, Padding et le Width des éléments et des panneaux sont des pixels logiques multipliés par DPI / 96, comme les barres VCL avec PPI / 96. ItemAtPos, ItemRect, PanelRect et GripRect sont des pixels client.
- WPF et Avalonia : ItemAtPos, ItemRect, OverflowRect, PanelAtPos, PanelRect et GripRect sont des unités indépendantes de l'appareil, comme dans FMX. Le cadre et la mise en page des modèles sont en pixels de l'appareil, les unités multipliées par l'échelle DPI du contrôle (WPF) ou l'échelle de rendu de la fenêtre (Avalonia). Le BoundsRect d'un élément ou d'un panneau est lui aussi en pixels de l'appareil.
- Headless : aWidth et aHeight sont des pixels logiques et aScale les multiplie, voir Rendu sans interface en .NET.

Événements

OnItemClick, OnPanelClick et le OnClick de l'élément sont des propriétés délégué portant les noms Delphi, et non des membres `event` de C#. Affecter un gestionnaire remplace le précédent, comme un pointeur de méthode Delphi, et ils sont affectés dans le code, y compris dans WPF et Avalonia où XAML ne peut pas les définir.

Pas de verbes du concepteur

Dans Delphi, les propriétés Items et Panels sont modifiées à la conception avec l'éditeur de collections standard. Le portage .NET n'a ni verbes du concepteur ni éditeur de collections : les Items et Panels d'une barre ne sont pas sérialisés, ils sont donc toujours ajoutés dans le code, généralement dans le constructeur de la fiche ou de la fenêtre. Dans WinForms, les barres figurent dans la boîte à outils et leurs propriétés simples se définissent dans la fenêtre Propriétés. Dans WPF et Avalonia, les barres sont déclarées en XAML.

Éditions

Le portage .NET est livré en packages Registered et Community, voir Présentation de .NET. Les packages Community n'ont ni OverflowMenu, ni Command, ni CommandParameter, comme l'édition Community de Delphi, voir Éditions. Il n'y a pas d'édition Trial.

Liaison de données

Sous WPF et Avalonia, les propriétés visibles par l'utilisateur sont des propriétés de dépendance et des propriétés Avalonia, si bien qu'elles peuvent être liées, stylées et animées, voir Liaison de données .NET.

Voir aussi

- Présentation de .NET
- Classes .NET
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Liens d'action
- Menus déroulants
- Images et glyphes
- Mises à jour thread-safe
- Point de redimensionnement

Liaison de données .NET

Dans les hôtes WPF et Avalonia, les propriétés visibles par l'utilisateur de la barre d'outils et de la barre d'état sont des propriétés de dépendance (WPF) et des propriétés stylées Avalonia. Elles peuvent donc être liées à un modèle de vue, définies par un Style et animées. Cette rubrique s'applique uniquement à WPF et Avalonia. Les contrôles WinForms ne changent pas : leurs propriétés sont de simples propriétés CLR, comme avant.

Propriétés pouvant être liées

Chaque propriété du tableau possède un champ statique avec le suffixe Property, par exemple `TsgcWpfToolbar.ShowCaptionsProperty`. Sous WPF, les champs sont de type DependencyProperty et sous Avalonia de type StyledProperty. Aucune de ces propriétés n'est modifiée par l'utilisateur, elles sont donc toutes liées en OneWay par défaut. Utilisez `Mode=TwoWay` lorsque le modèle de vue doit aussi recevoir une valeur définie dans le code sur le contrôle.

Classe WPF	Classe Avalonia	Propriétés pouvant être liées
TsgcWpfToolbar	TsgcAvaloniaToolbar	ShowCaptions, ButtonSize, Spacing, Padding, Theme, Accent-Color
TsgcWpfStatus-Bar	TsgcAvaloniaStatus-Bar	Theme, SizeGrip, AccentColor

Padding est la marge intérieure de la barre en unités indépendantes de l'appareil, un int. Sous WPF, elle masque le Padding de type Thickness de Control. Sous Avalonia, Theme est le thème sgcUICore (TsgcUITheme) et masque le thème de contrôle Avalonia de StyledElement, qui reste accessible via `StyledElement.ThemeProperty`.

Comportement

- **Le modèle conserve les valeurs.** Chaque propriété est stockée dans le modèle de la barre d'outils ou de la barre d'état. Une valeur liée est transmise au modèle, et la propriété reprend la valeur que le modèle conserve : le modèle relève ButtonSize à 8 et Spacing et Padding à 0, puis la propriété affiche la valeur utilisée.
- **Une valeur corrigée atteint la source.** Une source TwoWay reçoit aussi la valeur corrigée, sous WPF et sous Avalonia. WPF ne met pas la source à jour pendant qu'il transfère une valeur depuis elle, donc sous WPF le contrôle réécrit la valeur corrigée une fois que le dispatcher a exécuté le travail en attente. Une modification ultérieure faite sur le contrôle, depuis le code ou sur le modèle, atteint également une source TwoWay sur les deux plateformes. Sous Avalonia, la réécriture couvre une valeur que la source envoie lorsque la liaison est déjà active. Une valeur que la source détient déjà au démarrage de la liaison est corrigée dans le contrôle, mais n'est pas réécrite dans la source.
- **Les modifications faites sur le modèle sont recopiées.** Lorsque le modèle modifie lui-même une valeur, le contrôle la copie dans la propriété, si bien que les liaisons et les styles voient la valeur utilisée.
- **Styles.** Un setter de Style peut définir n'importe quelle propriété pouvant être liée, par exemple le Theme de toutes les barres d'outils et barres d'état d'une fenêtre.

Propriétés qui restent des propriétés CLR

Propriété	Raison
Items (barre d'outils)	La collection de boutons du modèle, construite dans le code, et non une valeur unique à lier.
Images (barre d'outils)	La liste d'images des boutons, un objet partagé par référence.

OverflowMenu (barre d'outils)	Le menu que la barre d'outils construit pour les boutons qui ne tiennent pas, en lecture seule.
Panels (barre d'état)	La collection de panneaux du modèle. Les valeurs des panneaux (Text, Progress, IndicatorState) sont définies par les méthodes thread-safe SetPanelText, SetProgress et SetIndicator, qui peuvent être appelées depuis n'importe quel thread.
Model	Le modèle indépendant de la plateforme derrière chaque contrôle, en lecture seule.

Exemple

Le modèle de vue ci-dessous est partagé par les deux exemples. ShowCaptions et Theme sont liés en OneWay, et ButtonSize est lié avec `Mode=TwoWay`, si bien qu'une taille définie sur la barre d'outils dans le code revient au modèle de vue. Les boutons sont ajoutés dans le code, comme dans les démarrages rapides.

WPF

MainWindow.xaml crée le modèle de vue comme DataContext de la fenêtre. Le code-behind et App.xaml sont ceux du modèle WPF.

Avalonia

MainWindow.axaml définit `x:DataType` sur le modèle de vue, si bien que les liaisons sont des liaisons compilées. Le code-behind appelle InitializeComponent, et la classe App est celle de Démarrage rapide Avalonia.

Voir aussi

- Différences de .NET par rapport à Delphi
- Classes .NET
- Démarrage rapide WPF
- Démarrage rapide Avalonia

Rendu sans interface en .NET

Une barre d'outils ou une barre d'état peut être dessinée sans fenêtre, sur un serveur, dans un test ou dans un rapport. **SaveToSVG** des deux modèles écrit un fichier ou un flux SVG avec le seul package esegece.sgcToolbar, et les méthodes d'extension **SaveToPNG** de esegece.sgcToolbar.Rendering écrivent un PNG dessiné avec SkiaSharp, sous Windows, Linux et macOS. La sortie SVG est identique octet pour octet à celle du rendu sans interface de Delphi.

Méthodes

Méthode	Package	Description
TsgcToolbarModel.SaveToSVG et TsgcStatusBarModel.SaveToSVG(aFileName ou aStream, aWidth, aHeight, aScale = 1)	esegece.sgcToolbar	Écrit la barre en SVG.
SaveToPNG(aFileName ou aStream, aWidth, aHeight, aScale = 1), une méthode d'extension des deux modèles dans sgcToolbar_Raster	esegece.sgcToolbar.Rendering	Écrit la barre en PNG sur fond transparent.
TsgcToolbarModel.Render et TsgcStatusBarModel.Render(aCanvas, aWidth, aHeight, aScale = 1)	esegece.sgcToolbar	Dessine la barre sur n'importe quel canevas sgcUICore, une barre de aWidth x aHeight pixels de l'appareil.
sgcToolbar_Render.RenderToolbar(aModel, aCanvas, aWidth, aHeight, aScale, aImages), sgcToolbar_Render.RenderStatusBar(aModel, aCanvas, aWidth, aHeight, aScale)	esegece.sgcToolbar	Le moteur de rendu derrière Render.

Pour SaveToSVG et SaveToPNG, aWidth et aHeight sont des pixels logiques. L'image mesure aWidth x aScale sur aHeight x aScale pixels, si bien qu'une échelle de 2 donne une image nette pour un écran haute densité ou une impression. Le modèle de la barre d'outils n'a pas de liste d'images : son événement **OnGetImage**, `bool (int aIndex, out TsgcUIARGBBuffer aImage)`, renvoie l'image d'un ImageIndex sous forme de **TsgcUIARGBBuffer** ; sans lui, le Glyph de l'élément est dessiné dans l'emplacement de l'icône.

Exemple

Un programme console (`net8.0`) qui référence esegece.sgcToolbar.Rendering et écrit une barre d'outils avec un bouton désactivé, un bouton à bascule enfoncé, un menu déroulant et un libellé repoussé à droite par un espaceur, et une barre d'état sombre avec un panneau de texte élastique, un panneau de progression à 65 et un indicateur d'avertissement, chacune en PNG et en SVG.

Voir aussi

- Présentation de .NET
- Classes .NET
- Rendu sans interface
- TsgcToolbarModel
- TsgcStatusBarModel

Démos .NET

Les démos .NET se trouvent dans le dossier `demos` du produit et dans le zip des démos, avec la solution `sgcToolbar_Demos.sln`. Chaque démo a un README.md.

Démo	Hôte	Description
Editor-Shell	Win-Forms	La structure d'un petit éditeur de texte avec un TsgcWinFormsToolbar et un TsgcWin-FormsStatusBar. La barre d'outils contient les boutons New, Open et Save, les bou-tons à bascule Bold et Italic, un groupe d'options Left, Center et Right (GroupIndex 1), un menu déroulant Zoom avec un menu de quatre niveaux de zoom, un espaceur, un bouton à bascule de thème sombre et un bouton Help. La barre d'état affiche la ligne et la colonne du curseur (AutoSize), un panneau élastique avec le dernier message, le zoom et un indicateur qui devient jaune tant que le document a des modifications non enregistrées. Save n'est activé que lorsqu'il y a des modifications non enregis-trées, et le rétrécissement de la fenêtre déplace les éléments qui ne tiennent pas dans le menu de débordement. Les boutons utilisent les gestionnaires OnClick des éléments et OnItemClick, de sorte que la démo se compile avec les packages Com-munity et Registered.
Editor-ShellWpf	WPF	Le même éditeur sur un TsgcWpfToolbar et un TsgcWpfStatusBar.
Editor-ShellAva-lonia	Avalo-nia	Le même éditeur sur un TsgcAvaloniaToolbar et un TsgcAvaloniaStatusBar, sous Win-dows, Linux et macOS.
LiveSta-tus	Win-Forms	Une barre d'état en direct alimentée depuis un thread de travail. Start lance un thread qui simule un téléchargement de 50 blocs, sans réseau, et rapporte sa progression di-rectement à la barre d'état avec les méthodes thread-safe SetPanelText, SetProgress et SetIndicator. La barre d'état a un panneau de message élastique, un panneau de progression, un indicateur du flux (vert pendant le téléchargement, jaune lorsqu'un bloc est lent, rouge lorsqu'il est arrêté) et un compteur de blocs AutoSize. Stop met fin au thread, qui signale sa fin au thread de l'interface via le SynchronizationContext.
LiveSta-tusWpf	WPF	La même barre d'état en direct sur un TsgcWpfStatusBar.
LiveSta-tusAvalo-nia	Avalo-nia	La même barre d'état en direct sur un TsgcAvaloniaStatusBar.

Compiler et exécuter

Les démos nécessitent le SDK .NET 8 ou ultérieur. Exécutez une démo depuis son dossier :

```
dotnet run
```

Dans l'arborescence du produit, les démos se compilent contre les projets de la bibliothèque. Depuis le zip des dé-mos, elles restaurent les packages NuGet, l'édition Community par défaut. Définissez **SgcToolbarEdition** pour uti-liser les packages Registered :

```
dotnet build -p:SgcToolbarEdition=Registered
```

Auto-vérification

Chaque démo a une option `--test` qui ouvre la fenêtre hors écran et la pilote : EditorShell clique sur le groupe d'options, sur les boutons à bascule, sur Save, sur le menu Zoom, sur le menu de débordement d'une fenêtre étroite et sur le bouton à bascule Dark, et LiveStatus exécute un téléchargement jusqu'au bout, en arrête un second à mi-parcours et change le thème. La démo écrit une ligne PASS ou FAIL par vérification dans le fichier et se termine avec le nombre de vérifications échouées, 0 signifie donc que toutes ont réussi.

```
dotnet run -- --test result.txt
```

Voir aussi

- Présentation de .NET
- Installation depuis NuGet
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Démos

Dépannage

Undeclared identifier Action ou OverflowMenu

L'édition Community se compile sans SGC_TOOLBAR_ADVANCED, donc la propriété Action des éléments et la propriété OverflowMenu des barres d'outils n'existent pas. Le code qui les utilise échoue avec E2003 Undeclared identifier, et une fiche enregistrée par une compilation Registered avec un élément lié à une action lève une exception "Error reading" au chargement. Utilisez l'OnClick de l'élément ou l'événement OnItemClick de la barre, ou une compilation Registered ou Trial. Voir Éditions.

Les derniers éléments sont coupés au bord droit

Dans l'édition Community, il n'y a pas de menu de débordement, donc les éléments qui ne tiennent pas sont tronqués. Élargissez la barre, définissez ShowCaptions à False, ou utilisez une compilation Registered ou Trial. Voir Débordement.

Le code après un menu s'exécute tardivement sous FireMonkey

Le TPopupMenu FireMonkey d'un élément déroulant et du bouton de débordement est modal : Popup ne retourne qu'à la fermeture du menu. Le code placé après ClickItem sur un élément déroulant, ou dans OnItemClick après l'affichage du menu, ne s'exécute que lorsque l'utilisateur a fermé le menu. Faites plutôt le travail dans l'OnClick des entrées du menu, et n'appelez pas ClickItem sur un élément déroulant depuis un timer ou un test qui ne doit pas attendre. Sous VCL, le menu s'ouvre à l'appui sur le bouton et sa boucle modale absorbe le relâchement de la souris.

La barre d'outils ignore la Height que j'ai définie

La hauteur d'une barre d'outils suit toujours ButtonSize plus deux fois Padding. Modifiez plutôt ces deux propriétés.

Un bouton d'option ne peut pas être désactivé

Un bouton à bascule avec GroupIndex supérieur à 0 est un bouton d'option : un clic ne fait que l'activer. Définissez Down à False depuis le code, ou utilisez GroupIndex 0 pour un bouton à bascule indépendant. Voir Boutons à bascule et groupes d'options.

Un élément ne réagit pas à un clic

Un élément désactivé (Enabled False) est dessiné en teinte atténuée et ignore les clics et le clavier, et les séparateurs, espaceurs et libellés ne reçoivent jamais de clic. Un élément avec une Action suit le Enabled de l'action, vérifiez donc aussi l'OnUpdate de l'action.

La barre d'état scintille ou lève une exception lorsqu'elle est mise à jour depuis un thread

Seuls SetPanelText, SetProgress et SetIndicator sont thread-safe. Définir d'autres propriétés, ou ajouter des panneaux, depuis un thread de travail touche le contrôle hors du thread principal. Voir Mises à jour thread-safe.

Le point de redimensionnement ne redimensionne pas la fiche

Sous FMX, le point n'est que dessiné. Sous VCL, il ne fonctionne que lorsque la fiche a une bordure redimensionnable et n'est pas agrandie. Voir Point de redimensionnement.

Les images ou les info-bulles ne s'affichent pas sous FireMonkey

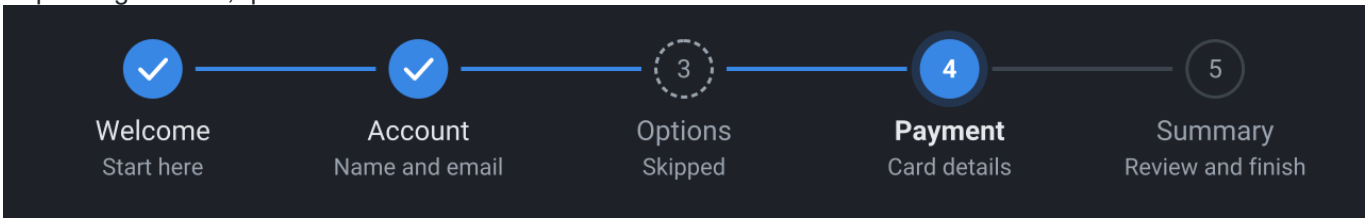
La propriété Images de la barre d'outils FMX nécessite Delphi XE8 ou ultérieur, et les info-bulles des barres FMX nécessitent 10.1 Berlin ou ultérieur avec ShowHint à True. Sur les versions plus anciennes, les éléments dessinent leur Glyph. Voir Images et glyphes.

La démo EditorShell ou EditorShellFMX ne se compile pas

Les deux démos utilisent les liens d'action et le menu de débordement, qui ne font pas partie de l'édition Community. Voir Démos.

Introduction

sgcWizard ajoute des assistants pas à pas et des indicateurs d'étapes aux applications Delphi et C++Builder, pour VCL et FireMonkey. **TsgcVCLWizard** et **TsgcFMXWizard** sont des conteneurs de pages : chaque page est une étape, une barre Stepper montre où se trouve l'utilisateur, et une barre de boutons permet de passer d'une page à l'autre avec Back, Next, Finish et Cancel. **TsgcVCLStepper** et **TsgcFMXStepper** sont les indicateurs d'étapes seuls, horizontaux ou verticaux, pour tout flux qui doit afficher sa progression.
sgcWizard fait partie de la suite visuelle eSeGeCe, aux côtés de sgcDashboard, sgcNavigation, sgcGrid et sgcToolbar. Il est dessiné avec **sgcUICore**, le package d'exécution partagé qui contient le canvas, le rasteriseur, les polices, l'écrivain SVG et les thèmes clair et sombre. Un assistant a donc le même aspect sous VCL, sous FireMonkey et dans la sortie PNG et SVG sans interface, et il s'accorde avec les autres produits de la suite. sgcWizard requiert sgcUICore, qui s'installe avec lui.



Éditions

sgcWizard est livré en versions Registered, Trial et Community. Toutes les éditions disposent de toutes les fonctionnalités, seule la version Trial affiche un avis d'évaluation. Voir Éditions.

Par où commencer

- Architecture : le modèle de stepper indépendant du framework, le moteur de rendu sur le canvas sgcUICore et les contrôles VCL et FMX.
- Plateformes : les versions de Delphi, C++Builder et les plateformes.
- Démarrage rapide VCL, Démarrage rapide FMX et Démarrage rapide C++Builder : un premier assistant en quelques minutes.
- Composants : les assistants, les pages, les steppers, les étapes et le modèle.
- Navigation et pages ignorées, Validation, Finish et Cancel, Clavier, États des étapes, Disposition du stepper, Libellés des boutons, Thèmes, Conception et Rendu sans interface.
- Démos : les projets de démonstration Delphi et C++Builder.

Architecture

sgcWizard est construit en couches. Chaque couche ne dépend que de celles situées en dessous, de sorte que le modèle et le moteur de rendu se compilent sans VCL ni FireMonkey.

Couche	Unité	Rôle
Modèle	sgcWizard_Model	TsgcStepperModel contient les Steps (une collection de TsgcWizard-Step), l'orientation, l'étape active, les états automatiques, les tailles, le thème et l'étape survolée. Sa méthode Layout calcule le cercle et le libellé de chaque étape visible ainsi que les connecteurs entre elles, et HitTest trouve l'étape située sous un point.
Moteur de rendu	sgcWizard_Render	sgcWizardRender dessine un modèle sur n'importe quel canvas sgcUICore et sgcWizardRenderButtonBar dessine la barre de boutons. sgcWizardSaveToPNG et sgcWizardSaveToSVG les utilisent pour écrire une image sans fenêtre. Voir Rendu sans interface.
Contrôles VCL	sgcWizard_VCL	TsgcVCLWizard, TsgcVCLWizardPage et TsgcVCLStepper. Windows uniquement.
Contrôles FMX	sgcWizard_FMX	TsgcFMXWizard, TsgcFMXWizardPage et TsgcFMXStepper, avec la même API et le même comportement.
sgcUI-Core	sgcUICore_Canvas, sgcUICore_VCL, sgcUICore_FMX, sgcUICore_Theme	Le runtime partagé : le canvas vectoriel et le rasteriseur, la base des contrôles VCL et FMX, et les jetons des thèmes clair et sombre avec les applicateurs de thème.

Stepper

Un contrôle stepper possède un TsgcStepperModel et publie ses propriétés. La propriété **Model** donne accès au reste : Layout, HitTest, IsStepClickable, IsConnectorDone, NextStep et PrevStep. Une modification du modèle planifie un nouveau rendu, elle ne dessine jamais directement, de sorte que de nombreuses modifications à la suite ne coûtent qu'un seul redessin.

Assistant et pages

Un assistant est un conteneur de pages, comme TPageControl avec ses pages TTabSheet. Les pages sont des contrôles enfants de l'assistant, dans l'ordre des contrôles, et seule la page active est visible. Chaque page contient des contrôles ordinaires : champs de saisie, cases à cocher, libellés, frames ou tout autre contrôle.
L'assistant possède deux contrôles enfants internes qui ne sont pas sérialisés et ne peuvent pas être sélectionnés dans le concepteur : la **barre Stepper**, un stepper dont les étapes reflètent les pages (Caption, Description, StepGlyph, et Skip comme état ignoré), et la **barre de boutons** avec Back, Next (Finish sur la dernière page) et Cancel. L'assistant définit lui-même les états de la barre, donc AutoStates de la barre vaut False. Les deux sont disponibles à l'exécution via les propriétés **Stepper** et **ButtonBar**.

Dessin

Les étapes, les connecteurs et les boutons sont dessinés par le moteur de rendu. Toutes les tailles sont des pixels logiques multipliés par l'échelle du moniteur, de sorte que les contrôles restent nets à tout DPI.

Voir aussi

- Plateformes
- TsgcStepperModel
- Thèmes

Éditions

sgcWizard est construit à partir des mêmes sources pour toutes les éditions, et chaque édition dispose de toutes les fonctionnalités.

Édition	Contenu
Registered	Tout : les assistants, pages et steppers VCL et FMX, la validation, les pages ignorées, les états des étapes, les thèmes, la prise en charge à la conception et la sortie PNG et SVG sans interface.
Trial	Les mêmes fonctionnalités que Registered. Une version compilée avec SGC_TRIAL affiche un message indiquant qu'il s'agit d'une version d'évaluation lorsqu'un composant est créé après la fin de la période de publication indiquée dans la version. Rien d'autre ne change.
Community	Les mêmes fonctionnalités que Registered, sans avis d'évaluation.

L'édition est un choix à la compilation dans sgcVer.inc. SGC_COMMUNITY désactive également SGC_TRIAL. Ce n'est pas un commutateur d'exécution, et le code écrit pour une édition se compile avec les autres.

Voir aussi

- Plateformes
- Introduction

Installation

sgcWizard fait partie de la distribution sgcUI. Un seul installateur couvre les produits sgcUI (sgcUICore, sgcCharts, sgcPDF, sgcReports, sgcMaps, sgcDashboard, sgcGrid, sgcNavigation, sgcToolbar et sgcWizard) et n'installe que les produits inclus dans votre licence.

Installation avec l'installateur sgcUI

1. Lancez l'installateur sgcUI et connectez-vous avec le nom d'utilisateur et le mot de passe de votre compte eSeGeCe.
2. Choisissez les versions de Delphi et de C++Builder dans lesquelles installer.
3. Sur la page des composants, sélectionnez **sgcWizard** et tout autre produit souhaité. Le serveur de licences renvoie les produits que vous possédez, et l'installateur affiche une case à cocher par produit. Les produits que votre licence n'inclut pas sont grisés. Une licence All-Access déverrouille tous les produits.
4. L'installateur copie les fichiers, puis compile les packages sur votre machine pour chaque IDE sélectionné, dans l'ordre des dépendances, et installe les packages de conception.

Ce qui est fourni avec sgcWizard

sgcUICore, la base commune des produits de mise en page et de sgcCharts, est fourni avec sgcWizard. Il n'a pas de case à cocher propre. L'installateur l'installe automatiquement dès que sgcWizard est sélectionné. sgcWizard n'a pas de package de passerelle sgcWebSockets.

Dossiers installés

Tous les produits s'installent dans un seul dossier, par défaut `Documents\esegece\sgcUI` (`sgcUI_Trial` pour la version d'essai et `sgcUI_Community` pour l'édition Community).

Dossier	Contenu
`Source`	Le code source de chaque produit installé. C'est l'unique chemin de bibliothèque de sgcUI.
`Packages`	Les packages de chaque produit installé, un jeu par produit et par version d'IDE.
`Lib<V>`	Les unités compilées pour chaque version d'IDE et chaque plateforme, par exemple `LibD13` .
`Demos\Wizard`	Les démos de sgcWizard. Voir Démos.

L'installateur ajoute `sgcUI\Source` au chemin de bibliothèque et au chemin de navigation de chaque IDE et plate-forme sélectionnés, ainsi que le dossier des unités compilées. sgcUICore se trouve dans le même dossier `Source`, il n'a donc besoin d'aucun chemin propre.

Trial et Community

- **Trial** : l'installateur Trial de sgcUI installe tous les produits en mode d'essai, y compris sgcWizard.
- **Community** : l'installateur Community gratuit de sgcUI inclut sgcWizard. Ses produits sont sgcCharts, sgc-Dashboard, sgcGrid, sgcNavigation, sgcToolbar et sgcWizard, tandis que sgcPDF et sgcMaps ne font pas partie de Community. La version Community de sgcWizard offre les mêmes fonctionnalités que la version Registered. Voir Éditions.

Installations autonomes précédentes

Si l'installateur trouve une ancienne installation autonome de sgcUICore, comme celle créée par l'installateur autonome de sgcCharts, il liste les chemins de bibliothèque et les packages de conception de cette installation sur la page **Previous installations**. Quand vous continuez, il les supprime de l'IDE avant de compiler les nouveaux packages, de sorte qu'il n'y ait jamais deux copies d'une unité dans le chemin.

Compiler les packages manuellement

Pour compiler vous-même les packages, ouvrez-les pour votre version d'IDE depuis `sgcUI\Packages`. `<V>` est le suffixe de version Delphi, par exemple `sgcWizardD13.dpk`.

Package	Contenu
`sgcUICore<V>`	La base commune sgcUICore (exécution).
`sgcWizard<V>` / `dclsgcWizard<V>`	Les composants sgcWizard (exécution / conception).

Compilez-les dans l'ordre des dépendances : `sgcUICore`, puis `sgcWizard`, puis `dclsgcWizard`. Installez ensuite le package de conception `dclsgcWizard` et ajoutez `sgcUI\Source` au chemin de bibliothèque de votre version de Delphi (Tools > Options > Language > Delphi Options > Library > Library path). Pour C++Builder, ouvrez le projet `.cbproj` correspondant à votre version, par exemple `sgcWizardC13.cbproj`, et compilez-le. Les en-têtes `.hpp` générés apparaissent à côté du package compilé.

.NET

Rien ne change pour .NET. sgcWizard reste distribué sous forme de ses propres packages NuGet (`esegece.sgcWizard` et ses packages compagnons). Voir l'installation .NET.

Plateformes

Cible	Prise en charge
Delphi	Delphi 7, 2007, 2009, 2010, XE à XE8, 10 à 10.4, 11, 12 et 13.
C++Builder	C++Builder 2007, 2010, XE à XE8, 10 à 10.4, 11, 12 et 13. Les packages génèrent les en-têtes .hpp et les fichiers .lib qu'un projet C++Builder inclut et lie.
VCL	TsgcVCLWizard, TsgcVCLWizardPage et TsgcVCLStepper sous Windows, Win32 et Win64, toutes les versions de Delphi.
FireMonkey	TsgcFMXWizard, TsgcFMXWizardPage et TsgcFMXStepper à partir de Delphi XE6 (SGC_WIZARD_FMX), sous Windows, macOS, iOS et Android, et sous Linux avec FMX pour Linux (FMXLinux). Les info-bulles des contrôles FMX requièrent 10.1 Berlin ou une version ultérieure.
Linux64	Les unités indépendantes du framework, le modèle et le moteur de rendu avec la sortie PNG et SVG sans interface, à partir de Delphi 10.2 (SGC_WIZARD_LINUX).
.NET	Un portage C# avec la même API pour WinForms, WPF et Avalonia, et la sortie PNG et SVG sans interface, sur .NET Framework 4.7.2 et .NET 8 et 9, voir Vue d'ensemble de .NET.

Packages

sgcWizard fournit un package d'exécution indépendant du framework et un package d'exécution par framework, de sorte qu'une application FireMonkey pour Android, iOS ou macOS ne lie jamais la VCL. Chaque nom de package se termine par le suffixe de l'IDE, par exemple sgcWizardD13.dpk pour Delphi 13 et sgcWizardC13.cbproj pour C++Builder 13.

Package	Type	Contenu	Requiert
sgcWizard	Exécution, indépendant du framework	Chaque unité du produit qui n'utilise ni la VCL ni FireMonkey.	rtl, sgcUICore
sgcWizardVCL	Exécution, VCL	sgcWizard_VCL (TsgcVCLWizard, TsgcVCLWizardPage, TsgcVCLStepper)	rtl, vcl, sgcUICore, sgcUICoreVCL, sgcWizard
sgcWizardFMX	Exécution, FireMonkey, à partir de XE6	sgcWizard_FMX (TsgcFMXWizard, TsgcFMXWizardPage, TsgcFMXStepper)	rtl, fmx, sgcUICore, sgcUICoreFMX, sgcWizard
dclsgcWizard	Conception, VCL	Enregistre TsgcVCLWizard et TsgcVCLStepper dans la page de palette **SGC Wizard**, ainsi que l'éditeur de composant qui ajoute des pages.	designide, rtl, vcl, sgcWizard, sgcWizardVCL
dclsgcWizardFMX	Conception, FireMonkey, à partir de XE6	Enregistre TsgcFMXWizard et TsgcFMXStepper dans la page de palette **SGC Wizard**, ainsi que l'éditeur de composant qui ajoute des pages.	designide, rtl, fmx, sgcWizard, sgcWizardFMX, dclsgcWizard

sgcUICore suit la même répartition : sgcUICore, sgcUICoreVCL et sgcUICoreFMX, sans package de conception. dclsgcWizardFMX requiert dclsgcWizard, installez donc d'abord dclsgcWizard.

Versions de l'IDE et plateformes cibles

Delphi	sgcWizard	sgcWizardVCL	sgcWizardFMX	dclsgcWizard, dclsgcWizardFMX
7, 2007, 2009, 2010, XE	Win32	Win32	Aucune	Win32
XE2 à XE5	Win32, Win64	Win32, Win64	Aucune	Win32
XE6 à 10.2	Win32, Win64	Win32, Win64	Win32, Win64	Win32
10.3	Win32, Win64, Linux64, OSX64	Win32, Win64	Win32, Win64, OSX64	Win32
10.4	Win32, Win64, Linux64, OSX64, Android, Android64, iOSDevice64	Win32, Win64	Win32, Win64, OSX64, Android, Android64, iOSDevice64	Win32
11	Win32, Win64, Linux64, OSX64, OSXARM64, Android, Android64, iOSDevice64, iOSSimARM64	Win32, Win64	Win32, Win64, OSX64, OSXARM64, Android, Android64, iOSDevice64, iOSSimARM64	Win32
12, 13	Comme 11	Win32, Win64	Comme 11	Win32, Win64

De XE6 à 10.2, le package FireMonkey est compilé pour Windows uniquement. Les packages de conception sont chargés par l'IDE, ils sont donc compilés pour la plateforme de l'IDE uniquement.

C++Builder	sgcWizard, sgcWizardVCL	sgcWizardFMX	dclsgcWizard, dclsgcWizardFMX
2007, 2010, XE, XE2	Win32	Aucune	Win32 (dclsgcWizard uniquement)
XE3 à XE5	Win32, Win64	Aucune	Win32 (dclsgcWizard uniquement)
XE6 à 12	Win32, Win64	Win32, Win64	Win32
13	Win32, Win64, Win64x	Win32, Win64, Win64x	Win32, Win64x

Les projets C++Builder portent les mêmes noms avec le suffixe C, par exemple sgcWizardC13.cbproj, sgcWizardVCLC13.cbproj, sgcWizardFMXC13.cbproj, dclsgcWizardC13.cbproj et dclsgcWizardFMXC13.cbproj. Les packages C++Builder sont compilés pour Windows uniquement. Les notes C++Builder qui s'appliquent à chaque produit sgcUI (packages d'exécution sous Win64, C++Builder 10.3 sous Win64, noms de fichiers des packages Win64x) se trouvent dans le sujet Plateformes et packages de l'aide de la suite sgcUI.

FireMonkey sur Android, iOS et macOS

Les contrôles FireMonkey (TsgcFMXWizard, TsgcFMXWizardPage et TsgcFMXStepper) ciblent macOS à partir de Delphi 10.3, Android et iOS à partir de Delphi 10.4, et macOS ARM et le simulateur iOS à partir de Delphi 11. Dans un projet FireMonkey pour ces plateformes :

- Utilisez les contrôles FireMonkey. Les contrôles VCL et sgcWizardVCL ne fonctionnent que sous Windows.
- Déposez le contrôle sur un formulaire, ou ajoutez son unité à la clause uses : sgcWizard_FMX.
- Sélectionnez la plateforme cible dans la fenêtre Projets. L'installateur ajoute le dossier Source de sgcUI et les unités compilées de chaque plateforme (LibD<version>\<platform>) au chemin de bibliothèque de chaque plateforme qu'il compile. Sans l'installateur, ajoutez le dossier Source de sgcUI au chemin de bibliothèque de la plateforme cible (Outils, Options, Langage, Delphi, Bibliothèque).
- Les applications Android et iOS sont liées statiquement, aucun package d'exécution n'est donc déployé.

Linux64

Le package indépendant du framework sgcWizard est compilé pour Linux64 à partir de Delphi 10.3. Une application Linux64, par exemple un programme console ou un serveur, utilise le modèle et le moteur de rendu avec la sortie PNG et SVG headless. Aucun package VCL ou FireMonkey n'est compilé pour Linux64, car FireMonkey sous Linux nécessite FMX pour Linux (FMXLinux), un module complémentaire tiers.
Les pages ne figurent pas dans la palette, elles s'ajoutent avec le verbe **Add Page**, voir Conception.

Voir aussi

- Éditions
- Architecture
- Dépannage

Démarrage rapide VCL

À la conception

- Déposez un **TsgcVCLWizard** depuis la page de palette **SGC Wizard** sur une fiche et définissez **Align** sur alClient.
- Faites un clic droit sur l'assistant et choisissez **Add Page** une fois pour chaque étape. Définissez **Caption** et **Description** de chaque page : ce sont le libellé et la description de son étape.
- Déposez sur chaque page ses contrôles. Choisissez **Next Page** et **Previous Page** dans le menu contextuel, ou cliquez sur une étape de la barre Stepper, pour afficher une autre page dans le concepteur.
- Écrivez un gestionnaire **OnBeforePageChange** qui valide la page et définit aAllow sur False lorsque la page n'est pas complète, et un gestionnaire **OnFinish** qui utilise les données.
- Exécutez l'application : Next et Entrée avancent, Back recule, Finish apparaît sur la dernière page, et Cancel ou Échap déclenchent OnCancel.

Dans le code

Le même assistant créé à l'exécution. Le programme se compile tel quel avec `sgcUI\Source` dans le chemin de recherche.

```
uses
  SysUtils, Forms, Classes, Controls, StdCtrls, Dialogs,
  sgcUICore_Theme, sgcWizard_Model, sgcWizard_VCL;

procedure TMainForm.CreateWizard;
var
  oPage: TsgcVCLWizardPage;
  oLabel: TLabel;
begin
  // the wizard fills the form: stepper band on top, button bar at the bottom
  FWizard := TsgcVCLWizard.Create(Self);
  FWizard.Parent := Self;
  FWizard.Align := alClient;

  // every page is a step of the stepper band
  oPage := FWizard.AddPage('Welcome');
  oPage.Description := 'Start here';
  oLabel := TLabel.Create(Self);
  oLabel.Parent := oPage;
  oLabel.SetBounds(24, 24, 320, 20);
  oLabel.Caption := 'This wizard creates your account.';

  oPage := FWizard.AddPage('Account');
  oPage.Description := 'Your name';
  FName := TEdit.Create(Self);
  FName.Parent := oPage;
  FName.SetBounds(24, 24, 240, 24);

  oPage := FWizard.AddPage('Done');
  oPage.Description := 'Summary';

  FWizard.ActivePageIndex := 0;
  FWizard.OnBeforePageChange := DoBeforePageChange;
  FWizard.OnFinish := DoFinish;
  FWizard.OnCancel := DoCancel;
end;

procedure TMainForm.DoBeforePageChange(Sender: TObject;
  aFromIndex, aToIndex: Integer; var aAllow: Boolean);
begin
  // leaving the Account page forwards needs a name
```

```
  if (aFromIndex = 1) and (aToIndex > aFromIndex) then
    aAllow := Trim(FName.Text) <> '';
end;

procedure TMainForm.DoFinish(Sender: TObject);
begin
  ShowMessage('Welcome, ' + FName.Text);
  Close;
end;

procedure TMainForm.DoCancel(Sender: TObject);
begin
  Close;
end;
```

Voir aussi

- TsgcVCLWizard
- TsgcVCLWizardPage
- Validation
- Navigation et pages ignorées

Démarrage rapide FMX

Les contrôles FireMonkey requièrent Delphi XE6 ou une version ultérieure. Ils s'utilisent exactement comme ceux de la VCL : déposez un **TsgcFMXWizard**, ajoutez les pages avec le verbe **Add Page**, déposez sur chaque page ses contrôles, et gérez **OnBeforePageChange** et **OnFinish**. Les pages sont alignées en Client dans l'assistant. Un toucher sur un bouton ou sur une étape terminée fonctionne comme un clic.

Dans le code

```
uses
  System.SysUtils, System.Classes, FMX.Forms, FMX.Types, FMX.Controls,
  FMX.StdCtrls, FMX.Edit, FMX.Dialogs,
  sgcUICore_Theme, sgcWizard_Model, sgcWizard_FMX;

procedure TMainForm.CreateWizard;
var
  oPage: TsgcFMXWizardPage;
  oLabel: TLabel;
begin
  Wizard := TsgcFMXWizard.Create(Self);
  Wizard.Parent := Self;
  Wizard.Align := TAlignLayout.Client;

  oPage := Wizard.AddPage('Welcome');
  oPage.Description := 'Start here';
  oLabel := TLabel.Create(Self);
  oLabel.Parent := oPage;
  oLabel.SetBounds(24, 24, 320, 20);
  oLabel.Text := 'This wizard creates your account.';

  oPage := Wizard.AddPage('Account');
  oPage.Description := 'Your name';
  txtName := TEdit.Create(Self);
  txtName.Parent := oPage;
  txtName.SetBounds(24, 24, 240, 24);

  oPage := Wizard.AddPage('Done');
  oPage.Description := 'Summary';

  Wizard.ActivePageIndex := 0;
  Wizard.OnBeforePageChange := WizardBeforePageChange;
  Wizard.OnFinish := WizardFinish;
end;

procedure TMainForm.WizardBeforePageChange(Sender: TObject;
  aFromIndex, aToIndex: Integer; var aAllow: Boolean);
begin
  if (aFromIndex = 1) and (aToIndex > aFromIndex) then
    aAllow := Trim(txtName.Text) <> '';
end;

procedure TMainForm.WizardFinish(Sender: TObject);
begin
  ShowMessage('Welcome, ' + txtName.Text);
  Close;
end;
```

Voir aussi

- TsgcFMXWizard

- TsgcFMXWizardPage
- Plateformes

Démarrage rapide C++Builder

Installez les packages sgcWizard pour votre version de C++Builder, ajoutez `sgcUI\Source` et le dossier des unités compilées, par exemple `sgcUI\LibD13\win32` (`sgcUI\LibD13\win64` pour le 64 bits), aux chemins d'inclusion et de bibliothèque du projet, puis utilisez les composants de la page de palette **SGC Wizard** exactement comme dans Delphi. L'installateur ajoute les deux chemins à l'IDE pour vous. Les en-têtes .hpp et les fichiers .lib générés de sgcWizard et de sgcUICore se trouvent dans ce seul dossier, sgcUICore n'a donc besoin d'aucun chemin propre.

Dans le code

OnBeforePageChange et OnFinish sont des événements de type closure, leurs gestionnaires sont donc des fonctions membres __fastcall d'une classe. Le paramètre var aAllow est une référence bool.

Voir aussi

- TsgcVCLWizard
- Démarrage rapide VCL
- Démos

Démos

Les démos se trouvent dans le dossier **Demos** de l'installation, avec les projets Delphi sous Demos\Delphi et les portages C++Builder des démos VCL sous Demos\CBuilder. Chaque dossier de démo contient un readme.html qui explique ce que fait la démo et comment elle fonctionne.

Démo	Composants	Ce qu'elle montre
Onboarding	TsgcVCLWizard, TsgcVCLWizardPage	Un assistant de création de compte avec quatre pages : Welcome, Account, Options et Summary. OnBeforePageChange vérifie le nom et l'e-mail de la page Account et oppose son veto à Next tant qu'ils ne sont pas valides, une case à cocher fait sauter la page Options, OnPageChanged remplit le récapitulatif et OnFinish l'affiche. Un commutateur clair et sombre et une liste déroulante StepperPosition (haut, gauche ou aucun). Onboarding.exe /test result.txt pilote l'assistant et écrit une ligne PASS ou FAIL par vérification. Delphi et C++Builder.
Stepper	TsgcVCLStepper	Deux steppers autonomes, l'un horizontal et l'autre vertical, avec les mêmes cinq étapes. Définir ActiveIndex termine les étapes qui le précèdent, des boutons marquent l'étape active comme wssError ou wssSkipped, et un clic sur une étape terminée ramène les deux steppers à cette étape via OnStepClick. Un commutateur clair et sombre. Delphi et C++Builder.
OnboardingFMX	TsgcFMXWizard, TsgcFMXWizardPage	Le flux Onboarding sous FireMonkey : la même validation, la même page Options ignorée, un récapitulatif à Finish, une case à cocher de thème sombre et les mêmes commutateurs /test et /snapshot. Delphi.

Les démos C++Builder incluent les en-têtes sgcWizard_*.hpp et sgcUICore_*.hpp et lient la bibliothèque statique du package d'exécution C++Builder. Elles se compilent avec le compilateur C++ classique, Win32 et Win64.

Voir aussi

- Démarrage rapide VCL
- Validation
- TsgcVCLStepper

TsgcVCLWizard

Unité : sgcWizard_VCL
TsgcVCLWizard est l'assistant VCL, Windows uniquement. Ses pages sont les contrôles enfants TsgcVCLWizardPage, la barre Stepper affiche une étape par page et la barre de boutons permet de passer de l'une à l'autre. Il hérite de TCustomControl et publie Align, Anchors, Constraints, Enabled, Font, ParentFont, ShowHint, ParentShowHint, TabOrder, TabStop et Visible.

Propriétés

Propriété	Type	Description
ActivePageIndex	Integer	L'index de la page visible. La définir par code change directement de page, sans événements. Par défaut 0.
ActivePage	classe de page	Publique. La page visible, ou nil. La définir change de page sans événements.
PageCount	Integer	Publique, en lecture seule. Le nombre de pages.
Pages[Index]	classe de page	Publique, en lecture seule. Les pages, dans l'ordre des contrôles.
StepperPosition	TsgcWizardStepperPosition	wspTop (par défaut) affiche une barre Stepper horizontale en haut, wspLeft une barre verticale à gauche, wspNone la masque. Voir Disposition du stepper.
BackCaption, NextCaption, FinishCaption, CancelCaption	string	Les libellés des boutons. Valeurs par défaut Back, Next, Finish et Cancel. Ils ne sont enregistrés que s'ils sont modifiés. Voir Libellés des boutons.
ShowCancel	Boolean	Affiche le bouton Cancel. Par défaut True.
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom. L'assistant le transmet à la barre, à la barre de boutons et aux contrôles enfants des pages. Voir Thèmes.
AccentColor	TsgcUIColor	Une accentuation $AARRGGBB pour les étapes active et terminées, les connecteurs terminés et le bouton par défaut. 0 (par défaut) utilise l'accentuation du thème.
Stepper	classe de stepper	Publique, en lecture seule. La barre Stepper interne.
ButtonBar	classe de barre de boutons	Publique, en lecture seule. La barre de boutons interne. ButtonCount, Buttons, ButtonRect et ButtonAt décrivent les boutons tels qu'ils sont dessinés.

Méthodes

Méthode	Retourne	Description
Next	Boolean	Passe à la page suivante qui n'est pas ignorée, en déclenchant OnBeforePageChange et OnPageChanged. Sur la dernière page, il ap-

		pelle Finish. True lorsque la page a changé ou que l'assistant est terminé.
Back	Boo-lean	Passe à la page précédente qui n'est pas ignorée, avec les mêmes événements.
GoToPage(aIndex)	Boo-lean	Passe à n'importe quelle page qui n'est pas ignorée, avec les mêmes événements. False pour une page ignorée ou un index hors limites.
Finish	Boo-lean	Déclenche OnBeforePageChange avec aToIndex = PageCount et, si c'est autorisé, OnFinish.
Cancel		Déclenche OnCancel.
NextPageIndex(aIndex), PrevPageIndex(aIndex)	Integer	La page suivante ou précédente après aIndex qui n'est pas ignorée, -1 s'il n'y en a aucune.
IsLastPage	Boo-lean	True lorsqu'aucune page après la page active n'est accessible, de sorte que Next affiche Finish.
IndexOfPage(aPage)	Integer	L'index d'une page, -1 lorsqu'elle n'est pas une page de cet assis-tant.
AddPage(aCaption, aOwner = nil)	classe de page	Crée une page avec ce Caption, dont le propriétaire est aOwner (nil utilise le propriétaire de l'assistant), et la place en dernier.

Événements

Événement	Type	Description
OnBefore-Page-Change	TsgcWizard-PageChan-geEvent	Avant chaque changement par Next, Back, GoToPage, Finish ou un clic sur une étape. aFromIndex, aToIndex et var aAllow : définissez aAllow sur False pour rester sur la page. aToIndex = PageCount signifie Finish. Voir Validation.
OnPage-Changed	TNotifyEvent	Après le changement de page.
OnFinish	TNotifyEvent	Après que Finish a été autorisé. Voir Finish et Cancel.
OnCancel	TNotifyEvent	Cancel, le bouton Cancel ou Échap.

Exemple

```
uses
  SysUtils, Forms, Classes, Controls, StdCtrls, Dialogs,
  sgcUICore_Theme, sgcWizard_Model, sgcWizard_VCL;

procedure TFeaturesForm.BuildPages;
var
  oPage: TsgcVCLWizardPage;
begin
  Wizard := TsgcVCLWizard.Create(Self);
  Wizard.Parent := Self;
  Wizard.Align := alClient;

  oPage := Wizard.AddPage('Welcome');
  chkSkipOptions := TCheckBox.Create(Self);
```

```
  chkSkipOptions.Parent := oPage;
  chkSkipOptions.SetBounds(24, 24, 240, 20);
  chkSkipOptions.Caption := 'Use the default options';
  chkSkipOptions.OnClick := SkipClick;

  oPage := Wizard.AddPage('Account');
  txtEmail := TEdit.Create(Self);
  txtEmail.Parent := oPage;
  txtEmail.SetBounds(24, 24, 240, 24);

  Wizard.AddPage('Options');
  Wizard.AddPage('Summary');

  Wizard.StepperPosition := wspTop;
  Wizard.OnBeforePageChange := WizardBeforePageChange;
  Wizard.OnFinish := WizardFinish;
  Wizard.OnCancel := WizardCancel;
  Wizard.ActivePageIndex := 0;
end;
```

Voir aussi

- TsgcVCLWizardPage
- TsgcFMXWizard
- Navigation et pages ignorées
- Clavier

TsgcVCLWizardPage

Unité : sgcWizard_VCL
TsgcVCLWizardPage est une page d'un TsgcVCLWizard. Elle hérite de TCustomControl, accepte des contrôles enfants et publie également Enabled, Font, ParentFont, Hint, ShowHint, ParentShowHint, PopupMenu, OnEnter, OnExit et OnResize. Une page ne peut être placée que sur un assistant : définir son Parent sur tout autre contrôle lève EsgcWizardError.

Propriétés

Propriété	Type	Description
Caption	string	Le libellé de l'étape de la page.
Description	string	La description de l'étape, dessinée sous le libellé tant que ShowDescriptions de la barre vaut True, et l'info-bulle de l'étape.
Skip	Boolean	Next et Back sautent la page et la barre dessine son étape comme wssSkipped. Par défaut False. Voir Navigation et pages ignorées.
StepGlyph	string	Un court texte dessiné dans le cercle de l'étape. Vide (par défaut), il dessine le numéro de l'étape.
Visible	Boolean	Géré par l'assistant : seule la page active est visible. Non enregistré.
Wizard	classe d'assistant	Publique, en lecture seule. L'assistant de la page, nil si elle n'en a pas.
PageIndex	Integer	Publique, en lecture seule. La position parmi les pages de l'assistant, -1 lorsqu'elle n'est pas sur un assistant.

L'assistant peint la page avec la couleur Surface de son thème et applique le thème aux contrôles enfants directs de la page, voir Thèmes.

Exemple

```
uses
  SysUtils, Forms, Classes, Controls, StdCtrls, Dialogs,
  sgcUICore_Theme, sgcWizard_Model, sgcWizard_VCL;

procedure TFeaturesForm.PageExample;
var
  oPage: TsgcVCLWizardPage;
begin
  oPage := TsgcVCLWizardPage.Create(Self);
  oPage.Caption := 'Licence';
  oPage.Description := 'Accept the terms';
  oPage.StepGlyph := 'L';
  // a page can only be placed on a wizard, it becomes the last page
  oPage.Parent := Wizard;
  Caption := Format('Page %d of %d', [oPage.PageIndex + 1,
    Wizard.PageCount]);
end;
```

Voir aussi

- TsgcVCLWizard
- Conception
- Dépannage

TsgcFMXWizard

Unité : sgcWizard_FMX
TsgcFMXWizard est l'assistant FireMonkey, à partir de Delphi XE6. Il a les mêmes propriétés, méthodes et événements que TsgcVCLWizard. Les pages sont les enfants TsgcFMXWizardPage, alignés en Client, la barre Stepper est alignée en MostTop ou MostLeft et la barre de boutons en MostBottom. Les positions et les tailles sont des unités logiques FMX. Il publie Align, Anchors, CanFocus, ClipChildren, Enabled, Height, HitTest, Margins, Opacity, Padding, PopupMenu, Position, Size, TabOrder, TabStop, Visible et Width, ShowHint et ParentShowHint à partir de 10.1 Berlin, ainsi que OnEnter, OnExit, OnKeyDown, OnKeyUp et OnResize.

Propriétés

Propriété	Type	Description
ActivePageIndex	Integer	L'index de la page visible. La définir par code change directement de page, sans événements. Par défaut 0.
ActivePage	classe de page	Publique. La page visible, ou nil. La définir change de page sans événements.
PageCount	Integer	Publique, en lecture seule. Le nombre de pages.
Pages[Index]	classe de page	Publique, en lecture seule. Les pages, dans l'ordre des contrôles.
StepperPosition	TsgcWizard-StepperPosition	wspTop (par défaut) affiche une barre Stepper horizontale en haut, wspLeft une barre verticale à gauche, wspNone la masque. Voir Disposition du stepper.
BackCaption, NextCaption, FinishCaption, CancelCaption	string	Les libellés des boutons. Valeurs par défaut Back, Next, Finish et Cancel. Ils ne sont enregistrés que s'ils sont modifiés. Voir Libellés des boutons.
ShowCancel	Boolean	Affiche le bouton Cancel. Par défaut True.
Theme	TsgcUI-Theme	utLight (par défaut), utDark ou utCustom. L'assistant le transmet à la barre, à la barre de boutons et aux contrôles enfants des pages. Voir Thèmes.
AccentColor	TsgcUIColor	Une accentuation $AARRGGBB pour les étapes active et terminées, les connecteurs terminés et le bouton par défaut. 0 (par défaut) utilise l'accentuation du thème.
Stepper	classe de stepper	Publique, en lecture seule. La barre Stepper interne.
ButtonBar	classe de barre de boutons	Publique, en lecture seule. La barre de boutons interne. ButtonCount, Buttons, ButtonRect et ButtonAt décrivent les boutons tels qu'ils sont dessinés.

Méthodes

Méthode	Retourne	Description

Next	Boolean	Passe à la page suivante qui n'est pas ignorée, en déclenchant OnBeforePageChange et OnPageChanged. Sur la dernière page, il appelle Finish. True lorsque la page a changé ou que l'assistant est terminé.
Back	Boolean	Passe à la page précédente qui n'est pas ignorée, avec les mêmes événements.
GoToPage(aIndex)	Boolean	Passe à n'importe quelle page qui n'est pas ignorée, avec les mêmes événements. False pour une page ignorée ou un index hors limites.
Finish	Boolean	Déclenche OnBeforePageChange avec aToIndex = PageCount et, si c'est autorisé, OnFinish.
Cancel		Déclenche OnCancel.
NextPageIndex(aIndex), PrevPageIndex(aIndex)	Integer	La page suivante ou précédente après aIndex qui n'est pas ignorée, -1 s'il n'y en a aucune.
IsLastPage	Boolean	True lorsqu'aucune page après la page active n'est accessible, de sorte que Next affiche Finish.
IndexOfPage(aPage)	Integer	L'index d'une page, -1 lorsqu'elle n'est pas une page de cet assistant.
AddPage(aCaption, aOwner = nil)	classe de page	Crée une page avec ce Caption, dont le propriétaire est aOwner (nil utilise le propriétaire de l'assistant), et la place en dernier.

Événements

Événement	Type	Description
OnBeforePageChange	TsgcWizardPageChangeEvent	Avant chaque changement par Next, Back, GoToPage, Finish ou un clic sur une étape. aFromIndex, aToIndex et var aAllow : définissez aAllow sur False pour rester sur la page. aToIndex = PageCount signifie Finish. Voir Validation.
OnPageChanged	TNotifyEvent	Après le changement de page.
OnFinish	TNotifyEvent	Après que Finish a été autorisé. Voir Finish et Cancel.
OnCancel	TNotifyEvent	Cancel, le bouton Cancel ou Échap.

Exemple

```
uses
  System.SysUtils, System.Classes, FMX.Forms, FMX.Types, FMX.Controls,
  FMX.StdCtrls, FMX.Edit, FMX.Dialogs,
  sgcUICore_Theme, sgcWizard_Model, sgcWizard_FMX;

procedure TMainForm.WizardExample;
begin
  // the stepper band on the left, a dark theme and localised captions
  Wizard.StepperPosition := wspLeft;
  Wizard.Theme := utDark;
  Wizard.NextCaption := 'Continue';
```

```
  Wizard.ShowCancel := False;
  if Wizard.Next then
    Caption := Wizard.ActivePage.Caption;
end;
```

Voir aussi

- TsgcFMXWizardPage
- Démarrage rapide FMX
- Thèmes

TsgcFMXWizardPage

Unité : sgcWizard_FMX
TsgcFMXWizardPage est une page d'un TsgcFMXWizard, alignée en Client à l'intérieur de celui-ci. Elle publie également ClipChildren, Enabled, HitTest, Padding, PopupMenu, OnEnter, OnExit et OnResize, ainsi que ShowHint et ParentShowHint à partir de 10.1 Berlin. Une page ne peut être placée que sur un assistant : définir son Parent sur tout autre objet lève EsgcWizardError.

Propriétés

Propriété	Type	Description
Caption	string	Le libellé de l'étape de la page.
Description	string	La description de l'étape, dessinée sous le libellé tant que ShowDescriptions de la barre vaut True, et l'info-bulle de l'étape.
Skip	Boolean	Next et Back sautent la page et la barre dessine son étape comme wssSkipped. Par défaut False. Voir Navigation et pages ignorées.
StepGlyph	string	Un court texte dessiné dans le cercle de l'étape. Vide (par défaut), il dessine le numéro de l'étape.
Visible	Boolean	Géré par l'assistant : seule la page active est visible. Non enregistré.
Wizard	classe d'assistant	Publique, en lecture seule. L'assistant de la page, nil si elle n'en a pas.
PageIndex	Integer	Publique, en lecture seule. La position parmi les pages de l'assistant, -1 lorsqu'elle n'est pas sur un assistant.

Exemple

```
uses
  System.SysUtils, System.Classes, FMX.Forms, FMX.Types, FMX.Controls,
  FMX.StdCtrls, FMX.Edit, FMX.Dialogs,
  sgcUICore_Theme, sgcWizard_Model, sgcWizard_FMX;

procedure TMainForm.PageExample;
var
  oPage: TsgcFMXWizardPage;
begin
  oPage := TsgcFMXWizardPage.Create(Self);
  oPage.Caption := 'Options';
  oPage.Description := 'Optional settings';
  oPage.Skip := True;
  // a page can only be placed on a wizard; it is aligned Client
  oPage.Parent := Wizard;
  Caption := Format('Page %d of %d', [oPage.PageIndex + 1,
    Wizard.PageCount]);
end;
```

Voir aussi

- TsgcFMXWizard

- TsgcVCLWizardPage
- Thèmes

TsgcVCLStepper

Unité : sgcWizard_VCL
TsgcVCLStepper est l'indicateur d'étapes VCL, utilisable seul. Il affiche ses étapes sous forme de cercles numérotés reliés par des connecteurs, horizontalement ou verticalement. L'info-bulle d'une étape est sa Description. Lorsque le contrôle a le focus, Gauche et Droite (Haut et Bas en vertical) passent à l'étape visible et activée précédente ou suivante, via le même veto OnStepClick, tant que Clickable vaut True. Il publie également Align, Anchors, Constraints, Enabled, ShowHint, ParentShowHint, TabOrder, TabStop et Visible.

Propriétés

Propriété	Type	Description
Steps	TsgcWizard-Steps	Les étapes.
Orientation	TsgcWizardO-rientation	woHorizontal (par défaut) ou woVertical.
ActiveIndex	Integer	L'étape active. Valeur par défaut 0. La définir par code ne déclenche pas OnActiveChange.
AutoStates	Boolean	Par défaut True : définir ActiveIndex termine les étapes visibles qui le précèdent, sauf si elles sont wssError ou wssSkipped, et place celles qui le suivent à wssPending. Voir États des étapes.
ShowDes-criptions	Boolean	Dessine la Description de chaque étape sous son libellé. Par défaut True.
CircleSize	Integer	Le diamètre des cercles des étapes, en pixels logiques. Par défaut 28.
Connector-Thickness	Integer	La largeur des connecteurs, en pixels logiques. Par défaut 2.
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom.
AccentCo-lor	TsgcUIColor	Une accentuation $AARRGGBB, 0 (par défaut) utilise l'accentuation du thème.
Clickable	Boolean	Un clic, ou les touches fléchées, peut activer une étape terminée, ou une étape en erreur située avant l'étape active. Par défaut True.
Model	TsgcStepper-Model	Publique, en lecture seule. Le modèle du contrôle.
Wizard	classe d'assistant	Publique, en lecture seule. L'assistant qui héberge le stepper comme sa barre, nil pour un stepper autonome.

Méthodes

Méthode	Retourne	Description
StepAt(X, Y)	Integer	L'étape située sous un point client, -1 s'il n'y en a aucune.

StepRect(aIndex)	TRect (TRectF sous FMX)	Le cercle et le libellé d'une étape en coordonnées client, vide lorsque l'étape n'est pas affichée.

Événements

Événement	Type	Description
OnStep-Click	TsgcWizard-StepClickEvent	Une étape cliquable a été cliquée, ou une touche fléchée y accède. aIndex et var aAllow : définissez aAllow sur False pour conserver l'étape active.
OnActive-Change	TNotifyEvent	L'étape active a changé par un clic ou une touche. Il ne se déclenche jamais lorsque ActiveIndex est défini par code.

Exemple

```
uses
  SysUtils, Forms, Classes, Controls, StdCtrls, Dialogs,
  sgcUICore_Theme, sgcWizard_Model, sgcWizard_VCL;

procedure TFeaturesForm.StepperExample;
begin
  Stepper := TsgcVCLStepper.Create(Self);
  Stepper.Parent := Self;
  Stepper.Align := alTop;
  Stepper.Height := 80;
  Stepper.Steps.AddStep('Cart', 'Review the items');
  Stepper.Steps.AddStep('Shipping', 'Address and carrier');
  Stepper.Steps.AddStep('Payment', 'Card or transfer');
  Stepper.Steps.AddStep('Done', '', 'OK');
  // AutoStates: the steps before ActiveIndex become completed
  Stepper.ActiveIndex := 1;
  Stepper.OnStepClick := StepperStepClick;
end;

procedure TFeaturesForm.StepperStepClick(Sender: TObject; aIndex: Integer;
  var aAllow: Boolean);
begin
  // aAllow False keeps the current step active
  aAllow := aIndex > 0;
end;
```

Voir aussi

- TsgcFMXStepper
- États des étapes
- Disposition du stepper

TsgcFMXStepper

Unité : sgcWizard_FMX
TsgcFMXStepper est l'indicateur d'étapes FireMonkey, à partir de Delphi XE6, avec les mêmes propriétés, méthodes et événements que TsgcVCLStepper. À partir de 10.1 Berlin, le Hint du contrôle suit la Description de l'étape située sous la souris. Il publie également Align, Anchors, CanFocus, Enabled, Height, HitTest, Margins, Opacity, Padding, PopupMenu, Position, Size, TabOrder, TabStop, Visible et Width, ainsi que OnEnter, OnExit, OnKeyDown, OnKeyUp, OnMouseDown, OnMouseMove, OnMouseUp et OnResize.

Propriétés

Propriété	Type	Description
Steps	TsgcWizard-Steps	Les étapes.
Orientation	TsgcWizardO-rientation	woHorizontal (par défaut) ou woVertical.
ActiveIndex	Integer	L'étape active. Valeur par défaut 0. La définir par code ne déclenche pas OnActiveChange.
AutoStates	Boolean	Par défaut True : définir ActiveIndex termine les étapes visibles qui le précèdent, sauf si elles sont wssError ou wssSkipped, et place celles qui le suivent à wssPending. Voir États des étapes.
ShowDes-criptions	Boolean	Dessine la Description de chaque étape sous son libellé. Par défaut True.
CircleSize	Integer	Le diamètre des cercles des étapes, en pixels logiques. Par défaut 28.
Connector-Thickness	Integer	La largeur des connecteurs, en pixels logiques. Par défaut 2.
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom.
AccentCo-lor	TsgcUIColor	Une accentuation $AARRGGBB, 0 (par défaut) utilise l'accentuation du thème.
Clickable	Boolean	Un clic, ou les touches fléchées, peut activer une étape terminée, ou une étape en erreur située avant l'étape active. Par défaut True.
Model	TsgcStepper-Model	Publique, en lecture seule. Le modèle du contrôle.
Wizard	classe d'assistant	Publique, en lecture seule. L'assistant qui héberge le stepper comme sa barre, nil pour un stepper autonome.

Méthodes

Méthode	Retourne	Description
StepAt(X, Y)	Integer	L'étape située sous un point client, -1 s'il n'y en a aucune.

StepRect(aIndex)	TRect (TRectF sous FMX)	Le cercle et le libellé d'une étape en coordonnées client, vide lorsque l'étape n'est pas affichée.

Événements

Événement	Type	Description
OnStep-Click	TsgcWizard-StepClickEvent	Une étape cliquable a été cliquée, ou une touche fléchée y accède. aIndex et var aAllow : définissez aAllow sur False pour conserver l'étape active.
OnActive-Change	TNotifyEvent	L'étape active a changé par un clic ou une touche. Il ne se déclenche jamais lorsque ActiveIndex est défini par code.

Exemple

```
uses
  System.SysUtils, System.Classes, FMX.Forms, FMX.Types, FMX.Controls,
  FMX.StdCtrls, FMX.Edit, FMX.Dialogs,
  sgcUICore_Theme, sgcWizard_Model, sgcWizard_FMX;

procedure TMainForm.StepperExample;
begin
  Stepper := TsgcFMXStepper.Create(Self);
  Stepper.Parent := Self;
  Stepper.Align := TAlignLayout.Top;
  Stepper.Height := 80;
  Stepper.Steps.AddStep('Cart', 'Review the items');
  Stepper.Steps.AddStep('Shipping', 'Address and carrier');
  Stepper.Steps.AddStep('Payment', 'Card or transfer');
  Stepper.ActiveIndex := 2;
  Stepper.Steps[1].State := wssError;
end;
```

Voir aussi

- TsgcVCLStepper
- États des étapes
- Plateformes

TsgcWizardStep

Unité : sgcWizard_Model
TsgcWizardStep est une étape d'un stepper, un élément de TsgcWizardSteps. Dans un assistant, les étapes de la barre sont créées à partir des pages, l'application définit donc plutôt les propriétés des pages.

Propriétés

Propriété	Type	Description
Caption	string	Le libellé dessiné sous le cercle (horizontal) ou à sa droite (vertical).
Description	string	La deuxième ligne du libellé, dessinée tant que ShowDescriptions vaut True, et l'info-bulle de l'étape.
Glyph	string	Un court texte dessiné dans le cercle. Vide, il dessine le numéro de l'étape.
State	TsgcWizard-StepState	wssPending (par défaut), wssActive, wssCompleted, wssError ou wssSkip-ped. Voir États des étapes.
Enabled	Boolean	Une étape désactivée est dessinée avec des couleurs atténuées et ne peut pas être cliquée. Par défaut True.
Visible	Boolean	Une étape masquée n'a ni emplacement ni numéro, et conserve son état. Par défaut True.
Tag	NativeInt (In-teger avant XE2)	Libre pour l'application. Par défaut 0.
Shown, Number, CircleRect, LabelRect	Boolean, In-teger, Tsg-cUIRect	Publiques, en lecture seule. Les résultats du dernier Layout : si l'étape a ob-tenu un emplacement, son numéro (base 1) parmi les étapes visibles, et les rectangles de son cercle et de son libellé en pixels de périphérique.

Méthodes

Méthode	Retourne	Description
CircleText	string	Le texte dessiné dans le cercle : Glyph, ou Number lorsque Glyph est vide.

Exemple

```
uses
  SysUtils, Forms, Classes, Controls, StdCtrls, Dialogs,
  sgcUICore_Theme, sgcWizard_Model, sgcWizard_VCL;

procedure TFeaturesForm.StepsExample;
var
  oStep: TsgcWizardStep;
begin
  Stepper.Steps.Clear;
  oStep := Stepper.Steps.Add;
  oStep.Caption := 'Upload';
```

```
  oStep.Description := 'Choose the files';
  oStep := Stepper.Steps.AddStep('Review', 'Check the preview');
  oStep.Glyph := 'R';
  // a hidden step gets no slot and no number
  Stepper.Steps.AddStep('Advanced').Visible := False;
  Stepper.Steps.AddStep('Publish');
  Stepper.ActiveIndex := 0;
end;
```

Voir aussi

- TsgcWizardSteps
- TsgcStepperModel

TsgcWizardSteps

Unité : sgcWizard_Model
TsgcWizardSteps est la collection propriétaire d'éléments TsgcWizardStep derrière la propriété Steps d'un stepper et de TsgcStepperModel. Elle s'édite à la conception avec l'éditeur de collection standard.

Méthodes

Méthode	Retourne	Description
Add	TsgcWizard-Step	Ajoute une étape vide.
AddStep(aCaption, aDescrip-tion = '', aGlyph = '')	TsgcWizard-Step	Ajoute une étape avec son libellé, sa description et son glyphe, et la retourne.
Items[Index]	TsgcWizard-Step	La propriété indexée par défaut.

Une étape ajoutée ou supprimée recalcule les états automatiques tant que AutoStates vaut True.

Exemple

```
uses
  SysUtils, Forms, Classes, Controls, StdCtrls, Dialogs,
  sgcUICore_Theme, sgcWizard_Model, sgcWizard_VCL;

procedure TFeaturesForm.StepperExample;
begin
  Stepper := TsgcVCLStepper.Create(Self);
  Stepper.Parent := Self;
  Stepper.Align := alTop;
  Stepper.Height := 80;
  Stepper.Steps.AddStep('Cart', 'Review the items');
  Stepper.Steps.AddStep('Shipping', 'Address and carrier');
  Stepper.Steps.AddStep('Payment', 'Card or transfer');
  Stepper.Steps.AddStep('Done', '', 'OK');
  // AutoStates: the steps before ActiveIndex become completed
  Stepper.ActiveIndex := 1;
  Stepper.OnStepClick := StepperStepClick;
end;

procedure TFeaturesForm.StepperStepClick(Sender: TObject; aIndex: Integer;
  var aAllow: Boolean);
begin
  // aAllow False keeps the current step active
  aAllow := aIndex > 0;
end;
```

Voir aussi

- TsgcWizardStep
- TsgcVCLStepper

TsgcStepperModel

Unité : sgcWizard_Model
TsgcStepperModel est le modèle d'un stepper, indépendant du framework : les étapes avec leur état, l'orientation, l'étape active, l'étape survolée et la disposition des cercles, des libellés et des connecteurs avec son test de positionnement. Les steppers VCL et FMX en possèdent un et publient ses propriétés, et le rendu sans interface le dessine sans fenêtre. Il est destiné à être utilisé depuis le thread principal.

Propriétés

Propriété	Type	Description
Steps, Orientation, ActiveIndex, AutoStates, ShowDescriptions, CircleSize, ConnectorThickness, Theme, AccentColor, Clickable		Les propriétés publiées des steppers, voir TsgcVCLStepper.
HotIndex	Integer	L'étape sous la souris, définie par le contrôle. Une étape survolée et cliquable reçoit une légère superposition.
ConnectorCount, Connectors[Index]	Integer, TsgcWizardConnector	Les connecteurs du dernier Layout : FromIndex, ToIndex et la ligne X1, Y1, X2, Y2 en pixels de périphérique.
LayoutWidth, LayoutHeight, LayoutScale	Integer, Double	La taille et l'échelle du dernier Layout.
OnChange	TNotifyEvent	Toute modification du modèle. Layout ne le déclenche jamais.

Méthodes

Méthode	Retourne	Description
Layout(aWidth, aHeight, aScale = 1) et les surcharges avec un callback de mesure ou un canvas		Calcule les rectangles des étapes et des connecteurs pour un stepper de aWidth par aHeight pixels de périphérique. Chaque taille logique est multipliée par aScale. Sans callback de mesure ni canvas, les largeurs des libellés sont estimées.
HitTest(aX, aY)	Integer	L'étape dont le cercle ou le libellé contient le point du dernier Layout, -1 s'il n'y en a aucune. Les étapes désactivées sont également signalées.
IsStepClickable(aIndex)	Boolean	Tant que Clickable est actif, une étape visible et activée qui n'est pas l'étape active est cliquable lorsqu'elle est wssCompleted, ou wssError avant l'étape active.
IsConnectorDone(aToIndex)	Boolean	La règle du connecteur, voir États des étapes.

ApplyAutoStates		Applique immédiatement les états automatiques, quelle que soit la valeur de AutoStates.
VisibleCount	Integer	Le nombre d'étapes visibles.
NextStep(aIndex), PrevStep(aIndex)	Integer	L'étape suivante ou précédente, visible et activée, -1 s'il n'y en a aucune.
GetAccent	TsgcUI-Color	L'accentuation utilisée : AccentColor, ou l'accentuation du thème lorsqu'elle vaut 0.

Exemple

Le programme console ci-dessous affiche Clickable: Cart, Visible steps: 3, After Shipping: 2, Connectors: 2 et Into Shipping done: TRUE.

```
uses
  SysUtils, Classes,
  sgcUICore_Theme, sgcWizard_Model, sgcWizard_Render;

procedure ModelExample;
var
  oModel: TsgcStepperModel;
  vIndex: Integer;
begin
  oModel := TsgcStepperModel.Create;
  try
    oModel.Steps.AddStep('Cart');
    oModel.Steps.AddStep('Shipping');
    oModel.Steps.AddStep('Payment');
    oModel.ActiveIndex := 1;

    // step rectangles for a stepper of 480 x 80 pixels at scale 1
    oModel.Layout(480, 80);
    vIndex := oModel.HitTest(oModel.Steps[0].CircleRect.Left + 2,
      oModel.Steps[0].CircleRect.Top + 2);
    if (vIndex >= 0) and oModel.IsStepClickable(vIndex) then
      WriteLn('Clickable: ', oModel.Steps[vIndex].Caption);

    WriteLn('Visible steps: ', oModel.VisibleCount);
    WriteLn('After Shipping: ', oModel.NextStep(1));
    WriteLn('Connectors: ', oModel.ConnectorCount);
    WriteLn('Into Shipping done: ', oModel.IsConnectorDone(1));
  finally
    oModel.Free;
  end;
end;
```

Voir aussi

- TsgcWizardStep
- Rendu sans interface
- États des étapes

Navigation et pages ignorées

L'utilisateur se déplace dans un assistant avec la barre de boutons, le clavier et les étapes terminées de la barre Stepper. L'application utilise les mêmes méthodes.

Action	Effet
Next, le bouton Next, Entrée	Passe à la page suivante qui n'est pas ignorée. Sur la dernière page, appelle Finish.
Back, le bouton Back	Passe à la page précédente qui n'est pas ignorée. Le bouton Back est désactivé lorsqu'aucune page antérieure n'est accessible.
GoToPage(aIndex)	Passe à n'importe quelle page qui n'est pas ignorée.
Un clic sur une étape terminée de la barre	Appelle GoToPage pour sa page.
ActivePageIndex ou ActivePage défini par code	Change directement de page, sans événements.

Chaque déplacement, sauf le changement direct, déclenche OnBeforePageChange, qui peut y opposer son veto, puis OnPageChanged. Lorsque le focus se trouvait dans l'assistant, un changement de page le place sur le premier contrôle de la nouvelle page, ou sur l'assistant.

Pages ignorées

Une page dont la propriété **Skip** vaut True reste dans l'assistant, mais Next et Back la sautent et la barre Stepper dessine son étape comme ignorée, avec un contour en pointillés. Skip peut changer à tout moment, par exemple à partir d'une case à cocher d'une page précédente. NextPageIndex et PrevPageIndex retournent la page où iraient Next et Back, et IsLastPage vaut True lorsqu'aucune page après la page active n'est accessible, de sorte que le bouton Next affiche le libellé Finish.

Exemple

```
uses
  SysUtils, Forms, Classes, Controls, StdCtrls, Dialogs,
  sgcUICore_Theme, sgcWizard_Model, sgcWizard_VCL;

procedure TFeaturesForm.NavigationExample;
begin
  // Next, Back and GoToPage fire the events and jump over skipped pages
  if not Wizard.Next then
    ShowMessage('The page did not change');
  Wizard.Back;
  Wizard.GoToPage(Wizard.PageCount - 1);

  // the page that Next goes to, or the last page
  if Wizard.IsLastPage then
    Caption := 'Last page'
  else
    Caption := 'Next: ' + Wizard.Pages
      [Wizard.NextPageIndex(Wizard.ActivePageIndex)].Caption;

  // setting ActivePageIndex by code switches the page without events
  Wizard.ActivePageIndex := 0;
end;
```

```
procedure TFeaturesForm.SkipClick(Sender: TObject);
begin
  // Next and Back jump over a skipped page, the stepper draws it as skipped
  Wizard.Pages[2].Skip := chkSkipOptions.Checked;
end;
```

Voir aussi

- TsgcVCLWizard
- Validation
- Finish et Cancel

Validation

OnBeforePageChange se déclenche avant chaque changement de page effectué par Next, Back, GoToPage, Finish ou un clic sur une étape. Il reçoit aFromIndex, la page active, aToIndex, la page sur le point d'être affichée, et un var aAllow qui vaut True au départ. Définissez aAllow sur False pour rester sur la page, par exemple tant qu'un champ obligatoire est vide.

- **aToIndex > aFromIndex** correspond à tout déplacement vers l'avant, Finish compris, de sorte qu'un seul test valide la page quel que soit le bouton actionné.
- **aToIndex = PageCount** signifie que l'utilisateur a actionné Finish : Finish déclenche l'événement avec PageCount comme cible, qui n'est jamais un véritable index de page.
- Les déplacements vers l'arrière n'ont généralement pas besoin de validation. Testez aToIndex < aFromIndex pour les laisser passer.
- Lorsque le changement est refusé, rien d'autre ne se produit : la page, la barre et les boutons restent tels quels, affichez donc la raison et placez le focus sur le champ.

Exemple

```
uses
  SysUtils, Forms, Classes, Controls, StdCtrls, Dialogs,
  sgcUICore_Theme, sgcWizard_Model, sgcWizard_VCL;

procedure TFeaturesForm.WizardBeforePageChange(Sender: TObject;
  aFromIndex, aToIndex: Integer; var aAllow: Boolean);
begin
  // aToIndex > aFromIndex is every forward move, Finish included
  if (aFromIndex = 1) and (aToIndex > aFromIndex) and
    (Pos('@', txtEmail.Text) = 0) then
  begin
    aAllow := False;
    ShowMessage('Enter a valid email address.');
    txtEmail.SetFocus;
  end;
  // aToIndex = PageCount means that the user pressed Finish
  if aAllow and (aToIndex = Wizard.PageCount) then
    aAllow := MessageDlg('Create the account?', mtConfirmation,
      [mbYes, mbNo], 0) = mrYes;
end;
```

Voir aussi

- Navigation et pages ignorées
- Finish et Cancel
- Démos

Finish et Cancel

Sur la dernière page, le bouton Next affiche le **FinishCaption** et devient le bouton par défaut. L'actionner, appuyer sur Entrée ou appeler **Finish** déclenche OnBeforePageChange avec aToIndex = PageCount et, si c'est autorisé, **OnFinish**. L'assistant ne ferme rien de lui-même : OnFinish décide de ce qui se passe, par exemple enregistrer les données et fermer la fiche, ou définir ModalResult d'une boîte de dialogue.
Le bouton Cancel, la touche Échap et un appel à **Cancel** déclenchent **OnCancel**. Il n'a pas de veto et ne change pas la page. Définissez **ShowCancel** sur False pour masquer le bouton Cancel.

Exemple

```
uses
  SysUtils, Forms, Classes, Controls, StdCtrls, Dialogs,
  sgcUICore_Theme, sgcWizard_Model, sgcWizard_VCL;

procedure TFeaturesForm.WizardFinish(Sender: TObject);
begin
  // runs after OnBeforePageChange allowed aToIndex = PageCount
  ModalResult := mrOk;
end;

procedure TFeaturesForm.WizardCancel(Sender: TObject);
begin
  // the Cancel button, the Escape key or a call to Cancel
  if MessageDlg('Discard the changes?', mtConfirmation, [mbYes, mbNo], 0)
    = mrYes then
    ModalResult := mrCancel;
end;
```

Voir aussi

- Validation
- Clavier
- Libellés des boutons

Clavier

Lorsque l'assistant ou un contrôle qu'il contient a le focus, **Entrée** correspond à Next (Finish sur la dernière page) et **Échap** à Cancel. Un contrôle qui veut la touche pour lui-même est prioritaire : un TMemo garde Entrée pour un saut de ligne, et un bouton ayant le focus, ou un TButton dont Default ou Cancel est défini, traite la touche comme d'habitude.

Framework	Comment les touches arrivent
VCL	Par CM_DIALOGKEY depuis la fiche parente, ou par KeyDown lorsque l'assistant lui-même a le focus.
FireMonkey	Par la chaîne des touches de dialogue de la fiche (DialogKey et AfterDialogKey), ou par KeyDown lorsque l'assistant lui-même a le focus.

Après un changement de page, le focus passe au premier contrôle de la nouvelle page, ou à l'assistant s'il se trouvait auparavant dans l'assistant.

Touches du stepper

Un stepper ayant le focus se déplace avec Gauche et Droite, ou Haut et Bas lorsqu'il est vertical, vers l'étape visible et activée précédente ou suivante. Le déplacement passe par le même veto OnStepClick qu'un clic et n'a lieu que tant que Clickable vaut True.

Voir aussi

- Navigation et pages ignorées
- Finish et Cancel
- TsgcVCLStepper

États des étapes

Chaque étape a un **State** qui détermine la façon dont elle est dessinée.

État	Dessiné comme
wssPending	Un contour de la couleur de bordure avec le numéro ou le glyphe de l'étape dans la couleur de texte atténuée.
wssActive	Un disque d'accentuation avec un léger halo d'accentuation et le texte de l'étape dans une couleur contrastée. Le libellé est en gras.
wssCompleted	Un disque d'accentuation avec une coche.
wssError	Un disque rouge avec un point d'exclamation et un libellé rouge.
wssSkipped	Un contour en pointillés avec le texte de l'étape dans la couleur de texte atténuée.

Une étape désactivée est dessinée avec des couleurs atténuées.

États automatiques

Tant que **AutoStates** vaut True (par défaut), chaque fois que ActiveIndex est défini, qu'une étape est ajoutée ou supprimée, ou que AutoStates est activé, les étapes visibles qui précèdent l'étape active passent à wssCompleted sauf si elles sont wssError ou wssSkipped, l'étape active passe à wssActive et les étapes visibles qui la suivent passent à wssPending. Les étapes masquées conservent leur état. Avec AutoStates à False, chaque état est laissé à l'application. Dans un assistant, la barre a AutoStates à False, car l'assistant définit les états à partir des pages et de leur propriété Skip.

La règle du connecteur

Le connecteur entre deux étapes visibles consécutives est terminé, dessiné avec la couleur d'accentuation, lorsque l'étape vers laquelle il mène est wssCompleted ou wssActive, ou lorsque cette étape est wssSkipped ou wssError et a été dépassée, c'est-à-dire qu'une étape visible ultérieure est wssActive ou wssCompleted. Sinon, il est dessiné avec la couleur de bordure. **Model.IsConnectorDone(aToIndex)** répond à cette règle pour le connecteur qui mène à l'étape aToIndex, de sorte qu'une page ignorée au milieu d'une partie terminée du flux n'interrompt pas la ligne d'accentuation.

Exemple

```
uses
  SysUtils, Forms, Classes, Controls, StdCtrls, Dialogs,
  sgcUICore_Theme, sgcWizard_Model, sgcWizard_VCL;

procedure TFeaturesForm.StatesExample;
var
  i: Integer;
begin
  // an error or skipped state survives the automatic states
  Stepper.Steps[1].State := wssError;
  Stepper.ActiveIndex := 2;

  // with AutoStates False the application sets every state
  Stepper.AutoStates := False;
```

```
  for i := 0 to Stepper.Steps.Count - 1 do
    Stepper.Steps[i].State := wssPending;
  Stepper.Steps[0].State := wssCompleted;
  Stepper.Steps[1].State := wssSkipped;
  Stepper.Steps[2].State := wssActive;

  // the connector into step 1 is done: step 1 was skipped and passed
  if Stepper.Model.IsConnectorDone(1) then
    Caption := 'Step 2 was passed';
end;
```

Voir aussi

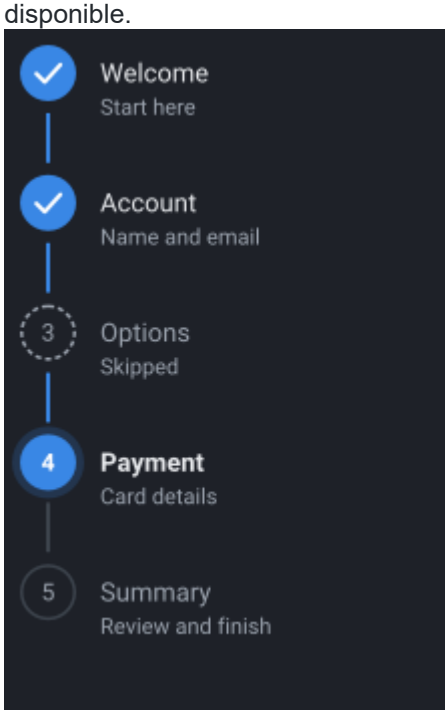
- TsgcWizardStep
- TsgcStepperModel
- Disposition du stepper

Disposition du stepper

Un stepper est horizontal ou vertical, selon **Orientation**. Seules les étapes visibles obtiennent un emplacement, et leur numéro est leur position (base 1) parmi les étapes visibles.

- **woHorizontal** : la largeur est partagée en emplacements égaux, le cercle est centré dans son emplacement, et le libellé et la description sont centrés en dessous. Le bloc est centré verticalement.
- **woVertical** : la hauteur est partagée en lignes égales, le cercle se trouve en haut à gauche de sa ligne, et le libellé est à sa droite, avec le titre centré sur le cercle. Les connecteurs descendent entre les cercles.

CircleSize (par défaut 28) et **ConnectorThickness** (par défaut 2) sont en pixels logiques. **ShowDescriptions** masque ou affiche la deuxième ligne des libellés. Les textes de libellé qui ne tiennent pas sont tronqués à la place disponible.



StepperPosition d'un assistant

Un assistant place sa barre avec **StepperPosition** : wspTop (par défaut) affiche une barre horizontale en haut, de 80 pixels logiques de haut, wspLeft une barre verticale à gauche, de 200 pixels logiques de large, et wspNone masque la barre. La barre de boutons en bas mesure 52 pixels logiques de haut.

Exemple

```
uses
  SysUtils, Forms, Classes, Controls, StdCtrls, Dialogs,
  sgcUICore_Theme, sgcWizard_Model, sgcWizard_VCL;

procedure TFeaturesForm.LayoutExample;
begin
  // the wizard: the stepper band on the left, drawn vertically
  Wizard.StepperPosition := wspLeft;

  // a standalone stepper: vertical, captions only, bigger circles
  Stepper.Align := alLeft;
  Stepper.Width := 200;
  Stepper.Orientation := woVertical;
  Stepper.ShowDescriptions := False;
  Stepper.CircleSize := 32;
```

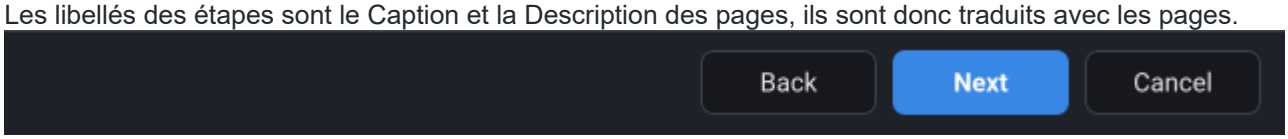
```
  Stepper.ConnectorThickness := 3;
end;
```

Voir aussi

- États des étapes
- TsgcVCLStepper
- TsgcVCLWizard

Libellés des boutons et localisation

Les quatre libellés de la barre de boutons sont des propriétés publiées de l'assistant : **BackCaption**, **NextCaption**, **FinishCaption** et **CancelCaption**, avec les valeurs par défaut Back, Next, Finish et Cancel. Un libellé n'est enregistré dans le fichier de la fiche que s'il diffère de la valeur par défaut, de sorte qu'une fiche avec les libellés par défaut reprend les nouvelles valeurs par défaut, et les libellés peuvent être traduits avec les outils de ressources habituels de l'IDE ou définis dans le code à partir de la traduction de l'application. Les boutons s'agrandissent avec leurs libellés, et **ShowCancel** masque le bouton Cancel.
Les libellés des étapes sont le Caption et la Description des pages, ils sont donc traduits avec les pages.



Exemple

```
uses
  SysUtils, Forms, Classes, Controls, StdCtrls, Dialogs,
  sgcUICore_Theme, sgcWizard_Model, sgcWizard_VCL;

procedure TFeaturesForm.CaptionsExample;
begin
  // Spanish captions for the button bar
  Wizard.BackCaption := 'Anterior';
  Wizard.NextCaption := 'Siguiente';
  Wizard.FinishCaption := 'Finalizar';
  Wizard.CancelCaption := 'Cancelar';
  // no Cancel button
  Wizard.ShowCancel := False;
end;
```

Voir aussi

- Finish et Cancel
- TsgcVCLWizard

Thèmes

Theme définit les couleurs à partir des jetons de thème de sgcUICore : utLight (par défaut), utDark ou utCustom. **AccentColor** remplace l'accentuation du thème par une couleur $AARRGGBB, utilisée par les étapes active et terminées, les connecteurs terminés et le bouton par défaut. 0 (par défaut) conserve l'accentuation du thème.
Un assistant transmet Theme et AccentColor à sa barre Stepper et à sa barre de boutons, se peint ainsi que ses pages avec la couleur Surface du thème, et applique le thème aux contrôles enfants directs de chaque page : lors d'un changement de thème, lorsque l'assistant est chargé et lorsqu'un contrôle est ajouté à une page. Un contrôle enfant ayant sa propre propriété Theme reçoit le thème. Tout autre enfant reçoit la couleur Surface comme arrière-plan et la couleur Text comme couleur de police, de sorte qu'un TLabel, un TEdit ou un TCheckBox reste lisible sur une page sombre.

- **FireMonkey** : les champs de saisie stylés, tels que TEdit, TMemo, TComboBox, TListBox, les contrôles de date et les grilles, conservent les couleurs de leur style, car le style peint un arrière-plan opaque qui lui est propre. La couleur de leur texte n'est modifiée que si leur couleur d'arrière-plan l'est aussi.
- **utCustom** laisse les contrôles enfants intacts. Sous VCL, les pages prennent la couleur de leur parent, sous FireMonkey ni l'assistant ni les pages ne sont peints.

Les jetons de thème sont les mêmes que ceux de sgcDashboard et des autres produits sgcUICore, de sorte qu'un assistant et un tableau de bord ayant le même Theme s'accordent, et qu'un assistant hébergé dans une tuile sgcDashboard suit le thème du tableau de bord.

Exemple

```
uses
  SysUtils, Forms, Classes, Controls, StdCtrls, Dialogs,
  sgcUICore_Theme, sgcWizard_Model, sgcWizard_VCL;

procedure TFeaturesForm.ThemesExample;
begin
  // the wizard, its stepper band, its button bar and the page children
  Wizard.Theme := utDark;
  // a green accent, $AARRGGBB; 0 uses the accent of the theme
  Wizard.AccentColor := $FF2E7D32;
  Stepper.Theme := utDark;
end;
```

Voir aussi

- Architecture
- Disposition du stepper
- Rendu sans interface

Conception

Le package de conception enregistre TsgcVCLWizard, TsgcVCLStepper, TsgcFMXWizard et TsgcFMXStepper dans la page de palette **SGC Wizard**. Les pages ne figurent pas dans la palette. L'éditeur de composant de l'assistant et de ses pages ajoute trois verbes au menu contextuel :

Verbe	Effet
Add Page	Crée une nouvelle page nommée d'après sa classe, avec son nom comme Caption, la place en dernier sur l'assistant, l'affiche et la sélectionne.
Next Page	Affiche la page suivante dans le concepteur.
Previous Page	Affiche la page précédente dans le concepteur.

Un clic sur une étape de la barre Stepper dans le concepteur affiche également sa page. Seule la page active est visible, et la page affichée dans le concepteur est l'ActivePageIndex enregistré dans le fichier de la fiche. La barre Stepper et la barre de boutons sont internes : elles ne sont pas enregistrées dans le fichier de la fiche et ne peuvent pas être sélectionnées. Les Steps d'un stepper autonome s'éditent avec l'éditeur de collection standard.

Voir aussi

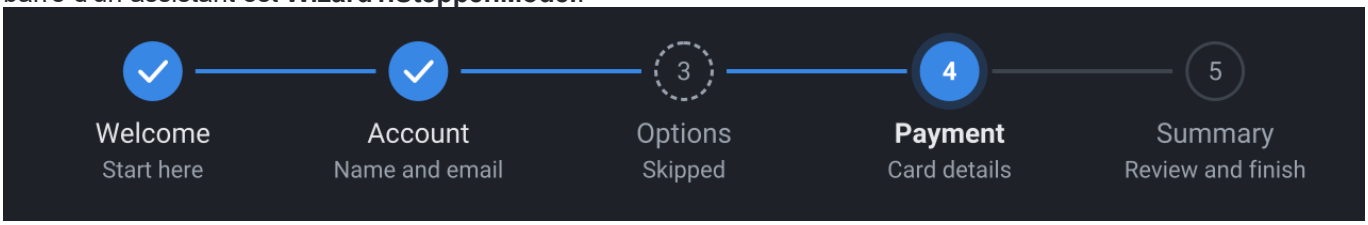
- Démarrage rapide VCL
- TsgcVCLWizardPage
- Plateformes

Rendu sans interface

Le moteur de rendu de sgcWizard_Render dessine un TsgcStepperModel et la barre de boutons sans fenêtre et sans VCL ni FMX, de sorte qu'une application console, un service, un test ou un serveur web peut produire le même stepper sous forme d'image, par exemple pour la documentation d'une application ou un test visuel. L'unité se compile sur toutes les plateformes, Linux64 compris.

Routine	Description
sgcWizardSaveToPNG(aModel, aFileName or aStream, aWidth, aHeight, aScale = 1)	Écrit un fichier ou un flux PNG. aWidth et aHeight sont des pixels logiques et aScale les multiplie : 640 par 100 à l'échelle 2 donne une image de 1280 par 200.
sgcWizardSaveToSVG(aModel, aFileName or aStream, aWidth, aHeight, aScale = 1)	Écrit un fichier ou un flux SVG du même stepper.
sgcWizardRender(aModel, aCanvas, aWidth, aHeight, aScale = 1)	Dessine le stepper sur n'importe quel canvas sgcUICore, en pixels de périphérique.
sgcWizardSaveButtonBarToPNG, sgcWizardSaveButtonBarToSVG (aButtons, aStream, aWidth, aHeight, aScale = 1, aTheme = utLight, aAccent = 0)	Écrit une barre de boutons. sgcWizardButton(aCaption, aEnabled, aIsDefault, aHot, aPressed) construit un bouton, et un bouton dont le libellé est vide est masqué.
sgcWizardRenderButtonBar, sgcWizardButtonBarLayout, sgcWizardButtonBarHitTest	Dessinent, disposent et testent le positionnement d'une barre de boutons sur n'importe quel canvas sgcUICore.

Le modèle d'un contrôle stepper est disponible via sa propriété **Model**, de sorte que **sgcWizardSaveToPNG(Stepper1.Model, ...)** enregistre ce que le contrôle affiche, à n'importe quelle taille. La barre d'un assistant est **Wizard1.Stepper.Model**.



Exemple

```
uses
  SysUtils, Classes,
  sgcUICore_Theme, sgcWizard_Model, sgcWizard_Render;

procedure Main;
var
  oModel: TsgcStepperModel;
  oStream: TFileStream;
begin
  oModel := TsgcStepperModel.Create;
  try
    oModel.Steps.AddStep('Welcome', 'Start here');
    oModel.Steps.AddStep('Account', 'Name and email');
    oModel.Steps.AddStep('Options', 'Skipped');
    oModel.Steps.AddStep('Payment', 'Card details');
    oModel.Steps.AddStep('Summary', 'Review and finish');
    oModel.Steps[2].State := wssSkipped;
    oModel.ActiveIndex := 3;
    oModel.Theme := utDark;
```

```
    // 640 x 100 logical pixels, written at scale 2 as 1280 x 200 pixels
    sgcWizardSaveToPNG(oModel, 'wizard.png', 640, 100, 2);
    sgcWizardSaveToSVG(oModel, 'wizard.svg', 640, 100);

    // the same steps drawn vertically
    oModel.Orientation := woVertical;
    sgcWizardSaveToPNG(oModel, 'wizard_vertical.png', 220, 340);

    // a button bar: Back, Next as the default button, Cancel
    oStream := TFileStream.Create('buttons.png', fmCreate);
    try
      sgcWizardSaveButtonBarToPNG([sgcWizardButton('Back'),
        sgcWizardButton('Next', True, True), sgcWizardButton('Cancel')],
        oStream, 640, 52, 1, utDark);
    finally
      oStream.Free;
    end;
  finally
    oModel.Free;
  end;
end;
```

Voir aussi

- TsgcStepperModel
- Plateformes
- États des étapes

Vue d'ensemble de .NET

sgcWizard .NET est un portage C# managé de la bibliothèque Delphi sgcWizard, pas une enveloppe (wrapper) autour d'elle. Les classes C# se trouvent dans l'espace de noms **esegece.sgcWizard** et conservent les noms Delphi, de sorte que **TsgcStepperModel**, **TsgcWizardSteps** et **TsgcWizardStepState** se lisent de la même façon dans les deux langages, et l'assistant, la page et le stepper de chaque hôte suivent le modèle de nommage de Delphi. Il est dessiné avec le portage .NET de **sgcUICore**, avec le même canvas, les mêmes polices et les mêmes thèmes clair et sombre que les autres produits de la suite visuelle eSeGeCe, de sorte qu'un assistant a le même aspect dans WinForms, WPF, Avalonia et dans la sortie PNG et SVG sans interface.

Packages

Le portage est réparti en cinq packages NuGet, afin qu'un projet n'intègre que l'hôte qu'il utilise.

PackageId	Contient	Frameworks cibles
esegece.sgcWizard	Le modèle du stepper (**TsgcStepperModel**, **TsgcWizardSteps**, **TsgcWizardStep**), le moteur de rendu **sgcWizard_Render** et la sortie SVG sans interface. Aucune dépendance native. Dépend de esegece.sgcUICore.	netstandard2.0, net472, net6.0, net8.0, net9.0
esegece.sgcWizard.Rendering	La sortie PNG sans interface, **sgcWizard_Render_Raster**, dessinée sur le canvas SkiaSharp de esegece.sgcUICore.Rendering, sous Windows, Linux et macOS.	netstandard2.0, net472, net6.0, net8.0, net9.0
esegece.sgcWizard.WinForms	**TsgcWinFormsWizard**, **TsgcWinFormsWizardPage**, **TsgcWinFormsStepper** et **TsgcWinFormsWizardButtonBar**.	net472, net8.0-windows, net9.0-windows
esegece.sgcWizard.Wpf	**TsgcWpfWizard**, **TsgcWpfWizardPage**, **TsgcWpfStepper** et **TsgcWpfWizardButtonBar**.	net472, net8.0-windows, net9.0-windows
esegece.sgcWizard.Avalonia	**TsgcAvaloniaWizard**, **TsgcAvaloniaWizardPage**, **TsgcAvaloniaStepper** et **TsgcAvaloniaWizardButtonBar**, sous Windows, Linux et macOS avec Avalonia 11.	net8.0, net9.0

Les trois packages d'hôte dépendent de esegece.sgcWizard.Rendering, qui dépend de esegece.sgcWizard, de sorte qu'une seule ligne `PackageReference` apporte tout ce dont un hôte a besoin. Voir Installation depuis NuGet.

Éditions

Édition	Ids des packages	Comprend

Registered	esegece.sgcWizard, esegece.sgcWizard.Rendering, esegece.sgcWizard.WinForms, esegece.sgcWizard.Wpf, esegece.sgcWizard.Avalonia	Tout : les assistants, les pages, les steppers et les barres de boutons des trois hôtes, la validation, les pages ignorées, les états des étapes, les thèmes et la sortie PNG et SVG sans interface.
Community	Les mêmes ids avec le suffixe .Community, par exemple esegece.sgcWizard.WinForms.Community. Ils dépendent des packages Community de sgcUICore.	Le même contenu que Registered. Seule la licence diffère.

L'édition Community est gratuite pour un usage commercial et non commercial par des particuliers et des organisations dont le chiffre d'affaires annuel brut est inférieur à 1 000 000 EUR et qui comptent au plus cinq développeurs. Une organisation qui atteint ou dépasse l'un de ces seuils a besoin d'une licence Registered. Le portage .NET n'a pas d'édition Trial.

Hôtes

Hôte	Équivalent Delphi	Plateformes
WinForms	VCL (TsgcVCLWizard)	Windows, .NET Framework 4.7.2 et .NET 8 et 9.
WPF	VCL et FMX	Windows, .NET Framework 4.7.2 et .NET 8 et 9. Les pages sont déclarées en XAML.
Avalonia	FMX (TsgcFMXWizard)	Windows, Linux et macOS, .NET 8 et 9. Les pages sont déclarées en XAML.
Sans interface	sgcWizard_Render	Toute plateforme des frameworks cibles : SVG avec esegece.sgcWizard, PNG avec esegece.sgcWizard.Rendering.

Tous les hôtes ont la même API : les pages, ActivePageIndex, Next, Back, GoToPage, Finish et Cancel, les événements OnBeforePageChange, OnPageChanged, OnFinish et OnCancel, les libellés des boutons, StepperPosition, ShowCancel, Theme et AccentColor. Voir Classes .NET pour les noms des classes et Différences de .NET par rapport à Delphi pour ce qui change par rapport à Delphi.

Voir aussi

- Installation depuis NuGet
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Classes .NET
- Différences de .NET par rapport à Delphi
- Rendu sans interface avec .NET
- Démos .NET
- Plateformes

Installation depuis NuGet

Les packages sont sur nuget.org. Ajoutez le package de l'hôte, les packages core et de rendu arrivent avec lui comme dépendances.

Édition Registered

```
dotnet add package esegece.sgcWizard.WinForms
dotnet add package esegece.sgcWizard.Wpf
dotnet add package esegece.sgcWizard.Avalonia
```

Édition Community

```
dotnet add package esegece.sgcWizard.WinForms.Community
dotnet add package esegece.sgcWizard.Wpf.Community
dotnet add package esegece.sgcWizard.Avalonia.Community
```

Ou ajoutez la référence au fichier de projet à la main :

```
<ItemGroup>
  <PackageReference Include="esegece.sgcWizard.Wpf" Version="2026.10.0" />
</ItemGroup>
```

Référencez dans un projet soit les packages Registered, soit les packages Community, jamais les deux : ils contiennent les mêmes types dans le même espace de noms.

Paramètres du projet

- WinForms : un projet `net8.0-windows` avec `<UseWindowsForms>true</UseWindowsForms>`, ou un projet .NET Framework 4.7.2.
- WPF : un projet `net8.0-windows` avec `<UseWPF>true</UseWPF>`, ou un projet .NET Framework 4.7.2. Associez l'espace de noms en XAML avec `xmlns:sgc="clr-namespace:esegece.sgcWizard;assembly=esegece.sgcWizard.Wpf"`.
- Avalonia : un projet `net8.0` ou `net9.0` avec les packages **Avalonia.Desktop** et **Avalonia.Themes.Fluent** d'Avalonia 11.3, comme toute application de bureau Avalonia. Associez l'espace de noms en XAML avec `xmlns:sgc="using:esegece.sgcWizard"`.
- Sans interface : esegece.sgcWizard seul suffit pour le SVG. Ajoutez esegece.sgcWizard.Rendering pour le PNG, il apporte SkiaSharp et ses ressources natives pour Windows et Linux.

En C#, tout hôte a besoin de `using esegece.sgcWizard;`, et de `using esegece.sgcUICore;` pour **TsgcUITheme** et les autres types des thèmes.

Voir aussi

- Vue d'ensemble de .NET
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Démos .NET

Démarrage rapide WinForms

TsgcWinFormsWizard est l'assistant WinForms, l'équivalent du TsgcVCLWizard de Delphi. C'est un conteneur de pages : chaque **TsgcWinFormsWizardPage** est une étape, la bande du stepper affiche les étapes et la barre de boutons navigue entre les pages. Le package est **esegece.sgcWizard.WinForms**.

Marche à suivre

- Créez un projet WinForms et ajoutez le package esegece.sgcWizard.WinForms.
- Créez un **TsgcWinFormsWizard**, ancrez-le dans la fiche et ajoutez une page par étape avec **AddPage**. Renseignez la **Description** de chaque page : c'est la description de son étape, et le titre (caption) est l'argument de AddPage.
- Ajoutez les contrôles de chaque page aux **Controls** de la page.
- Affectez **OnBeforePageChange** pour valider une page et mettez aAllow à false lorsqu'elle n'est pas complète, et **OnFinish** pour utiliser les données.
- Exécutez : Next et Entrée avancent, Back revient en arrière, Finish apparaît sur la dernière page, et Cancel ou Échap appellent OnCancel.

Exemple

Le Program.cs complet d'une application WinForms avec un assistant de trois pages. Il compile tel quel dans un projet `net8.0-windows` qui référence esegece.sgcWizard.WinForms.
Les pages peuvent aussi être créées avec `new TsgcWinFormsWizardPage()` et ajoutées avec `Controls.Add` : l'ordre des pages est l'ordre dans lequel elles ont été ajoutées. L'assistant et le stepper sont dans la boîte à outils, mais il n'y a pas de verbe Add Page, voir Différences de .NET par rapport à Delphi.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- Différences de .NET par rapport à Delphi
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Validation

Démarrage rapide WPF

TsgcWpfWizard est l'assistant WPF. Ses pages sont les éléments **TsgcWpfWizardPage** déclarés en XAML comme ses enfants, et chaque page contient un élément de contenu, généralement un panneau. Le package est **esegece.sgcWizard.Wpf**.

Marche à suivre

- Créez un projet WPF (`dotnet new wpf`) et ajoutez le package esegece.sgcWizard.Wpf.
- Associez l'espace de noms `esegece.sgcWizard` et déclarez un **TsgcWpfWizard** avec un **TsgcWpfWizard-Page** par étape. **Caption** et **Description** de chaque page sont le titre et la description de son étape.
- Affectez les événements dans le code behind : ce sont des délégués, pas des événements WPF, ils ne peuvent donc pas être définis comme attributs XAML.

Exemple

MainWindow.xaml déclare l'assistant et ses trois pages. Le App.xaml du projet est celui du modèle WPF, avec StartupUri défini sur MainWindow.xaml.
MainWindow.xaml.cs affecte les événements et valide la page Account.
Les pages peuvent aussi être créées en code avec `Wizard.AddPage("Caption")`, qui renvoie le nouveau **TsgcWpf-WizardPage**, et l'assistant les expose dans **Pages**.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- Différences de .NET par rapport à Delphi
- Démarrage rapide WinForms
- Démarrage rapide Avalonia
- Validation

Démarrage rapide Avalonia

TsgcAvaloniaWizard est l'assistant Avalonia, l'équivalent du TsgcFMXWizard de Delphi, sous Windows, Linux et macOS. Ses pages sont les éléments **TsgcAvaloniaWizardPage** déclarés en XAML comme ses enfants, et chaque page contient un enfant. Le package est **esegece.sgcWizard.Avalonia**.

Marche à suivre

- Créez un projet .NET 8 et ajoutez les packages esegece.sgcWizard.Avalonia, Avalonia.Desktop et Avalonia.Themes.Fluent.
- Associez l'espace de noms avec `xmlns:sgc="using:esegece.sgcWizard"` et déclarez un **TsgcAvaloniaWizard** avec un **TsgcAvaloniaWizardPage** par étape, avec son **Caption** et sa **Description**.
- Affectez les événements dans le code behind : ce sont des délégués, pas des événements Avalonia.

Exemple

MainWindow.axaml déclare l'assistant et ses trois pages.
MainWindow.axaml.cs définit le thème, affecte les événements et affiche un message sur la page Account tant que le nom est vide.
Program.cs démarre l'application avec le thème Fluent.
Les pages peuvent aussi être créées en code avec `Wizard.AddPage("Caption")`, qui renvoie le nouveau **TsgcAvaloniaWizardPage** ; définissez son **Child** avec le contenu de la page.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- Différences de .NET par rapport à Delphi
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Validation

Classes .NET

Chaque hôte a un assistant, une page, un stepper et une barre de boutons avec les mêmes membres. Les types indépendants du framework sont partagés par les trois hôtes et conservent leurs noms Delphi.

Classes des hôtes

Rôle	WinForms	WPF	Avalonia
Assistant	TsgcWinFormsWizard	TsgcWpfWizard	TsgcAvaloniaWizard
Page	TsgcWinFormsWizardPage (un conteneur de Control)	TsgcWpfWizardPage (un élément Content)	TsgcAvaloniaWizardPage (un Border avec un Child)
Collection de pages	ReadOnlyCollection<TsgcWinFormsWizardPage>	TsgcWpfWizardPages	AvaloniaList<TsgcAvaloniaWizardPage>
Stepper	TsgcWinFormsStepper	TsgcWpfStepper	TsgcAvaloniaStepper
Barre de boutons	TsgcWinFormsWizardButtonBar	TsgcWpfWizardButtonBar	TsgcAvaloniaWizardButtonBar

Types indépendants du framework

Type	Description
TsgcStepperModel	Les étapes, l'étape active, l'orientation, les états automatiques, la mise en page et le test de positionnement (hit test). Le stepper de chaque hôte en possède un, voir TsgcStepperModel.
TsgcWizardSteps, TsgcWizardStep	La collection d'étapes et une étape : Caption, Description, Glyph, State, Enabled, Visible et Tag.
TsgcWizardStepState	wssPending, wssActive, wssCompleted, wssError, wssSkipped.
TsgcWizardOrientation	woHorizontal, woVertical.
TsgcWizardStepperPosition	wspTop, wspLeft, wspNone.
TsgcWizardPageChangeEvent	`void (object Sender, int aFromIndex, int aToIndex, ref bool aAllow)`

TsgcWizardStepClickEvent	`void (object Sender, int aIndex, ref bool aAllow)`
TsgcUINotifyEvent	`void (object Sender)`, de esegece.sgcUICore.
sgcWizard_Render, sgcWizard_Render_Raster	Le moteur de rendu et la sortie SVG et PNG sans interface, voir Rendu sans interface avec .NET.

Assistant

Membre	Type	Description
Pages	collection	Les pages dans l'ordre. PageCount et IndexOfPage(aPage) comme dans Delphi.
AddPage(string aCaption)	page	Crée une page, l'ajoute à la fin et la renvoie.
ActivePageIndex, ActivePage	int, page	La page visible. Les définir change directement de page, sans événements.
Next(), Back(), GoToPage(int aIndex)	bool	Passe à une autre page via OnBeforePageChange, en ignorant les pages dont Skip vaut true. Next sur la dernière page appelle Finish.
Finish(), Cancel()	bool, void	Finish appelle OnBeforePageChange avec aToIndex = PageCount puis OnFinish. Cancel appelle OnCancel.
NextPageIndex(int), PrevPageIndex(int), IsLastPage()	int, bool	La page suivante et la page précédente qui ne sont pas ignorées, et si la page active est la dernière.
StepperPosition	TsgcWizardStepperPosition	Où se trouve la bande du stepper : en haut, à gauche ou masquée.
BackCaption, NextCaption, FinishCaption, CancelCaption	string	Les libellés des boutons, pour la localisation.
ShowCancel	bool	Affiche le bouton Cancel.
Theme, AccentColor	TsgcUITheme, uint	Le thème de l'assistant, de sa bande de stepper, de sa barre de boutons et des contrôles de ses pages. AccentColor 0 utilise la couleur d'accent du thème.
Stepper, ButtonBar	stepper, barre de boutons	La bande du stepper et la barre de boutons internes.
OnBeforePageChange	TsgcWizardPageChangeEvent	Avant chaque changement de page et avant Finish. aAllow false y met son veto.
OnPageChanged, OnFinish, OnCancel	TsgcUINotifyEvent	Après un changement de page, sur Finish et sur Cancel.

Page

Membre	Type	Description
Caption	string	Le titre (caption) de l'étape. Dans WinForms, c'est le Text du contrôle.
Description	string	La description de l'étape et son info-bulle (tooltip).
StepGlyph	string	Le texte dans le cercle de l'étape à la place de son numéro.
Skip	bool	Next, Back et GoToPage sautent la page et l'étape est dessinée comme ignorée.
Wizard, PageIndex	assistant, int	L'assistant de la page et son index.
Content (WPF), Child (Avalonia), Controls (WinForms)	élément	Ce que la page affiche.

Stepper

Le stepper fonctionne seul, sans assistant. Ses membres sont ceux des steppers Delphi : Model, Steps, Orientation, ActiveIndex, AutoStates, ShowDescriptions, CircleSize, ConnectorThickness, Theme, AccentColor, Clickable, StepAt(X, Y), StepRect(aIndex), OnStepClick (TsgcWizardStepClickEvent, aAllow false met son veto au clic) et OnActiveChange, qui n'est déclenché que par une interaction de l'utilisateur. Wizard est l'assistant qui le possède, ou null pour un stepper autonome. Voir TsgcVCLStepper.

Barre de boutons

La barre de boutons d'un assistant dessine Back, Next (Finish sur la dernière page) et Cancel. GetButtons(), ButtonCount(), Buttons(aIndex), ButtonRect(aIndex) et ButtonAt(X, Y) décrivent les boutons, et Wizard est son assistant. Elle est créée par l'assistant et accessible via sa propriété ButtonBar.
Les positions et les tailles sont en pixels dans WinForms (Rectangle) et en unités indépendantes de l'appareil dans WPF et Avalonia (Rect).

Voir aussi

- Vue d'ensemble de .NET
- Différences de .NET par rapport à Delphi
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia

Différences de .NET par rapport à Delphi

L'API .NET suit l'API Delphi nom pour nom. Voici les endroits où elle diffère.

Événements

OnBeforePageChange, OnPageChanged, OnFinish, OnCancel, OnStepClick et OnActiveChange sont des propriétés délégué portant les noms Delphi, pas des membres `event` C#. Affecter un gestionnaire remplace le précédent, comme un pointeur de méthode Delphi, et ils sont affectés en code, y compris dans WPF et Avalonia où XAML ne peut pas les définir. Le paramètre Delphi `var aAllow: Boolean` est `ref bool aAllow`, et un gestionnaire de OnBeforePageChange a la signature `void (object Sender, int aFromIndex, int aToIndex, ref bool aAllow)`. Les démarrages rapides montrent les gestionnaires des trois hôtes.

GoToPage et navigation

GoToPage, Next, Back et Finish renvoient bool et se comportent comme dans Delphi : GoToPage renvoie false lorsque l'index est hors limites ou que la page est ignorée, renvoie true lorsque la page est déjà active, et passe sinon par OnBeforePageChange. Finish appelle OnBeforePageChange avec aToIndex = PageCount, de sorte que aToIndex > aFromIndex correspond à tout déplacement vers l'avant. Un clic sur une étape terminée de la bande du stepper appelle GoToPage. Définir ActivePageIndex ou ActivePage change de page sans événements, comme dans Delphi.

La collection de pages

Delphi possède la propriété indexée `Pages[Index]` et `PageCount`. Dans .NET, **Pages** est une collection, de sorte que `Pages[i]`, `Pages.Count` et `foreach` fonctionnent comme d'habitude :

- WinForms : une collection en lecture seule. Une page est ajoutée avec AddPage ou en ajoutant un TsgcWinFormsWizardPage aux Controls de l'assistant, et l'ordre des pages est l'ordre dans lequel elles ont été ajoutées, pas l'ordre z des contrôles.
- WPF : TsgcWpfWizardPages, la propriété de contenu de l'assistant, de sorte que les pages sont les enfants de l'assistant en XAML. Chaque page a un élément Content.
- Avalonia : une AvaloniaList, la propriété de contenu de l'assistant. Chaque page est un Border avec un Child.

Info-bulles

Dans Delphi, l'info-bulle (hint) d'une étape est sa Description, affichée lorsque ShowHint vaut True, et FMX affiche les info-bulles à partir de 10.1 Berlin. Dans .NET, le stepper affiche toujours la Description de l'étape sous la souris sous forme de tooltip, sans propriété ShowHint : WinForms utilise un composant ToolTip interne, WPF définit la propriété ToolTip du stepper et Avalonia définit la propriété attachée ToolTip.Tip. Ne définissez pas vous-même le tooltip d'un stepper, le stepper le remplace ; définissez plutôt la Description de l'étape ou de la page.

Pas de verbes du concepteur

L'éditeur de composant de Delphi possède les verbes Add Page, Next Page et Previous Page et affiche une autre page dans le concepteur lorsqu'on clique sur une étape, voir Conception. Le portage .NET n'a pas de verbes du concepteur. Dans WinForms, l'assistant et le stepper sont dans la boîte à outils, et les pages sont créées en code avec AddPage. Dans WPF et Avalonia, les pages sont déclarées en XAML.

Clavier

Entrée correspond à Next ou Finish et Échap à Cancel tant que le focus est dans l'assistant, comme dans Delphi. Dans WinForms, les touches arrivent via ProcessDialogKey, de sorte qu'un TextBox multiligne ou un bouton qui a

le focus les conserve. Dans WPF et Avalonia, elles arrivent via l'événement KeyDown remontant (bubbling), de sorte qu'un contrôle qui gère la touche l'emporte.

Éditions

Le portage .NET est livré en packages Registered et Community au contenu identique, voir Vue d'ensemble de .NET. Il n'y a pas d'édition Trial ni de commutateur de compilation dans le code.

Liaison de données

Sous WPF et Avalonia, les propriétés visibles par l'utilisateur sont des propriétés de dépendance et des propriétés Avalonia, si bien qu'elles peuvent être liées, stylées et animées, voir Liaison de données .NET.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia

Liaison de données .NET

Dans les hôtes WPF et Avalonia, les propriétés visibles par l'utilisateur de l'assistant, de ses pages et du stepper sont des propriétés de dépendance (WPF) et des propriétés Avalonia. Elles peuvent donc être liées à un modèle de vue, définies par un Style et animées. Cette rubrique s'applique uniquement à WPF et Avalonia. Les contrôles WinForms ne changent pas : leurs propriétés sont de simples propriétés CLR, comme avant.

Propriétés pouvant être liées

Chaque propriété du tableau possède un champ statique avec le suffixe Property, par exemple `TsgcWpfWizard.ActivePageIndexProperty`. Sous WPF, les champs sont de type DependencyProperty. Sous Avalonia, ce sont des StyledProperty, sauf ActivePageIndex et ActiveIndex, qui sont des DirectProperty. Les propriétés marquées **TwoWay** sont liées en TwoWay par défaut, car le contrôle les modifie lui-même (Next, Back, GoToPage, un clic sur une étape), de sorte qu'une liaison sans Mode maintient le modèle de vue à jour. Les autres propriétés sont liées en OneWay par défaut.

Classe WPF	Classe Avalonia	Propriétés pouvant être liées
TsgcWpfWizard	TsgcAvaloniaWizard	**ActivePageIndex (TwoWay)**, StepperPosition, BackCaption, NextCaption, FinishCaption, CancelCaption, ShowCancel, Theme, AccentColor
TsgcWpfWizardPage	TsgcAvaloniaWizardPage	Caption, Description, StepGlyph, Skip
TsgcWpfStepper	TsgcAvaloniaStepper	Orientation, **ActiveIndex (TwoWay)**, AutoStates, ShowDescriptions, CircleSize, ConnectorThickness, Theme, AccentColor, Clickable

Sous Avalonia, Theme est le thème sgcUICore (TsgcUITheme) et masque le thème de contrôle Avalonia de StyledElement, qui reste accessible via `StyledElement.ThemeProperty`.

Comportement

- **Une liaison navigue, le code non.** Un nouvel ActivePageIndex qui arrive par une liaison de données est une navigation : il passe par OnBeforePageChange puis par OnPageChanged, comme GoToPage. Défini depuis le code (le setter CLR), ActivePageIndex change de page sans événements, comme dans Delphi.
- **Un veto est réécrit.** Lorsque OnBeforePageChange définit aAllow à false, l'assistant conserve la page actuelle et restaure l'ancien index dans le contrôle et dans le modèle de vue lié. Sous WPF, le modèle de vue reçoit l'ancien index une fois que le dispatcher a exécuté le travail en attente.
- **Un index hors limites est ignoré.** Un index lié qui n'est pas une page conserve la page actuelle et ne déclenche aucun événement, et une source TwoWay reçoit en retour l'index utilisé, sous WPF et sous Avalonia. La liaison reste active, si bien que le prochain index valide navigue.
- **Une valeur corrigée atteint la source.** Une valeur que le contrôle corrige, comme une CircleSize ou un ActiveIndex que le stepper limite, est réécrite dans la propriété et dans une source TwoWay, sous WPF et sous Avalonia. Sous WPF, la source la reçoit une fois que le dispatcher a exécuté le travail en attente. Sous Avalonia, la réécriture couvre une valeur que la source envoie lorsque la liaison est déjà active. Une valeur que la source détient déjà au démarrage de la liaison est corrigée dans le contrôle, mais n'est pas réécrite dans la source.
- **Le modèle de vue suit l'assistant.** Next, Back, GoToPage, un clic sur une étape et une modification depuis le code copient le nouvel index dans ActivePageIndex, et une liaison TwoWay le transmet au modèle de vue.
- **Styles.** Un setter de Style peut définir n'importe quelle propriété pouvant être liée, par exemple le Theme et les options du stepper de tous les assistants d'une fenêtre.

Propriétés qui restent des propriétés CLR

Propriété	Raison
Pages (assistant)	La collection de pages est la propriété de contenu de l'assistant, remplie par les pages déclarées en XAML, et non une valeur à lier.
ActivePage (assistant)	Une référence à un élément de page. Liez plutôt ActivePageIndex, qui suit chaque changement de page.
PageIndex (page)	En lecture seule, la position de la page dans Pages.
Steps (stepper)	La collection d'étapes. L'assistant remplit les étapes de son propre stepper à partir des pages.
OnBeforePageChange, OnPageChanged, OnFinish, OnCancel, OnStepClick, OnActiveChange	Propriétés de type délégué portant les noms Delphi, affectées dans le code, voir Différences de .NET par rapport à Delphi.

Exemple

Le modèle de vue ci-dessous est partagé par les deux exemples. PageIndex est lié à ActivePageIndex sans Mode, il est donc lié en TwoWay, NextCaption est lié au libellé du bouton Next, et SkipOptions est lié à la propriété Skip de la page Options et à une case à cocher.

WPF

MainWindow.xaml crée le modèle de vue comme DataContext de la fenêtre. Le code-behind et App.xaml sont ceux du modèle WPF.

Avalonia

MainWindow.axaml définit `x:DataType` sur le modèle de vue, si bien que les liaisons sont des liaisons compilées. Le code-behind appelle InitializeComponent, et la classe App est celle de Démarrage rapide Avalonia.

Voir aussi

- Différences de .NET par rapport à Delphi
- Classes .NET
- Démarrage rapide WPF
- Démarrage rapide Avalonia

Rendu sans interface avec .NET

Un stepper peut être dessiné sans fenêtre, sur un serveur, dans un test ou dans un rapport. **sgcWizard_Render.sgcWizardSaveToSVG** écrit un fichier ou un flux SVG avec le seul package esegece.sgcWizard, et **sgcWizard_Render_Raster.sgcWizardSaveToPNG** écrit un PNG avec esegece.sgcWizard.Rendering, qui dessine sur SkiaSharp sous Windows, Linux et macOS. La sortie SVG est identique octet pour octet à celle du rendu sans interface de Delphi.

Méthodes

Méthode	Package	Description
sgcWizard_Render.sgcWizardSaveToSVG(aModel, aFileName or aStream, aWidth, aHeight, aScale = 1)	esegece.sgcWizard	Écrit le stepper de aModel en SVG.
sgcWizard_Render_Raster.sgcWizardSaveToPNG(aModel, aFileName or aStream, aWidth, aHeight, aScale = 1)	esegece.sgcWizard.Rendering	Écrit le stepper de aModel en PNG sur fond transparent.
sgcWizard_Render.sgcWizardSaveButtonBarToSVG(aButtons, aStream, aWidth, aHeight, aScale, aTheme, aAccent)	esegece.sgcWizard	Écrit une barre de boutons en SVG.
sgcWizard_Render_Raster.sgcWizardSaveButtonBarToPNG(aButtons, aStream, aWidth, aHeight, aScale, aTheme, aAccent)	esegece.sgcWizard.Rendering	Écrit une barre de boutons en PNG.
sgcWizard_Render.sgcWizardButton(aCaption, aEnabled = true, aIsDefault = false, aHot = false, aPressed = false)	esegece.sgcWizard	Renvoie un TsgcWizardButton pour une barre de boutons.
sgcWizard_Render.sgcWizardRender(aModel, aCanvas, aWidth, aHeight, aScale = 1)	esegece.sgcWizard	Dessine le stepper sur n'importe quel canvas sgcUICore.

aWidth et aHeight sont des pixels logiques. L'image fait aWidth x aScale par aHeight x aScale pixels, de sorte qu'une échelle de 2 donne une image nette pour un écran haute densité ou une impression.

Exemple

Un programme console (`net8.0`) qui référence esegece.sgcWizard.Rendering et écrit un stepper sombre avec une étape ignorée en PNG et SVG, les mêmes étapes dessinées à la verticale, et une barre de boutons.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- Rendu sans interface
- TsgcStepperModel

Démos .NET

Les démos .NET se trouvent dans le dossier `demos` du produit et dans le zip des démos, avec la solution `sgcWizard_Demos.sln`. Chaque démo a un README.md.

Démo	Hôte	Description
Onboarding	WinForms	Un assistant de création de compte en quatre pages (Welcome, Account, Options et Summary) sur un TsgcWinFormsWizard. OnBeforePageChange valide le nom et l'e-mail de la page Account, une case à cocher définit Skip sur la page Options, OnPageChanged remplit le récapitulatif, OnFinish l'affiche et OnCancel demande confirmation avant de fermer. La barre d'outils change le thème et place la bande du stepper en haut, à gauche ou la masque.
OnboardingWpf	WPF	Le même assistant sur un TsgcWpfWizard déclaré dans MainWindow.xaml.
OnboardingAvalonia	Avalonia	Le même assistant sur un TsgcAvaloniaWizard, sous Windows, Linux et macOS.
Stepper	WinForms	Deux contrôles TsgcWinFormsStepper autonomes, l'un horizontal et l'autre vertical, avec les cinq étapes d'un paiement, maintenus synchronisés. Les boutons avancent, reculent, marquent l'étape active comme erreur, l'ignorent et réinitialisent les étapes, et un clic sur une étape terminée l'active via OnStepClick.
StepperWpf	WPF	Les mêmes steppers sous forme de contrôles TsgcWpfStepper placés dans MainWindow.xaml.
StepperAvalonia	Avalonia	Les mêmes steppers sous forme de contrôles TsgcAvaloniaStepper.

Compiler et exécuter

Les démos nécessitent le SDK .NET 8 ou ultérieur. Exécutez une démo depuis son dossier :

```
dotnet run
```

Dans l'arborescence du produit, les démos se compilent avec les projets de la bibliothèque. Depuis le zip des démos, elles restaurent les packages NuGet, l'édition Community par défaut. Définissez **SgcWizardEdition** pour utiliser les packages Registered :

```
dotnet build -p:SgcWizardEdition=Registered
```

Auto-vérification

Chaque démo a un commutateur `--test` qui ouvre la fenêtre hors écran, pilote l'assistant ou les steppers, écrit une ligne PASS ou FAIL par vérification dans le fichier et se termine avec le nombre de vérifications échouées, de sorte que 0 signifie que toutes les vérifications ont réussi.

```
dotnet run -- --test result.txt
```

Voir aussi

- Vue d'ensemble de .NET
- Installation depuis NuGet

- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Démos

Dépannage

EsgcWizardError: A TsgcVCLWizardPage can only be placed on a TsgcVCLWizard

Une page doit être un enfant direct d'un assistant. Définir le Parent d'un TsgcVCLWizardPage sur un panneau, une fiche ou tout autre contrôle lève EsgcWizardError, et il en va de même pour un TsgcFMXWizardPage en dehors d'un TsgcFMXWizard. Créez les pages avec **AddPage**, ou définissez leur Parent sur l'assistant. Pour regrouper des contrôles sur une page, placez un TPanel ou une frame sur la page. Voir TsgcVCLWizardPage.

Un programme de test demande des droits d'administrateur

Windows traite un exécutable dont le nom contient des mots tels que setup, install ou update comme un programme d'installation et demande une élévation avant de le lancer. Lorsque vous testez un assistant, ne nommez pas le programme SetupWizard.exe ou InstallWizard.exe, choisissez un nom tel que Onboarding.exe, ou ajoutez un manifeste avec requestedExecutionLevel asInvoker.

Next ne fait rien

OnBeforePageChange a défini aAllow sur False, ce qui est la façon dont la validation oppose son veto à un changement, ou bien les pages suivantes sont toutes ignorées et Next appelle donc Finish. Vérifiez le gestionnaire et la propriété Skip des pages. Voir Validation.

Finish ne ferme pas la fiche

L'assistant ne ferme jamais rien. Fermez la fiche ou définissez ModalResult dans OnFinish, ainsi que dans OnCancel. Voir Finish et Cancel.

Entrée ne passe pas à la page suivante

Un contrôle qui veut la touche Entrée la garde : un TMemo, un bouton ayant le focus, ou un TButton de la fiche dont Default vaut True. Voir Clavier.

Un clic sur une étape ne change pas la page

Seules les étapes terminées, et les étapes en erreur situées avant l'étape active, peuvent être cliquées, et seulement tant que Clickable vaut True. Un clic passe aussi par OnBeforePageChange.

Un champ de saisie FMX garde ses couleurs claires sur une page sombre

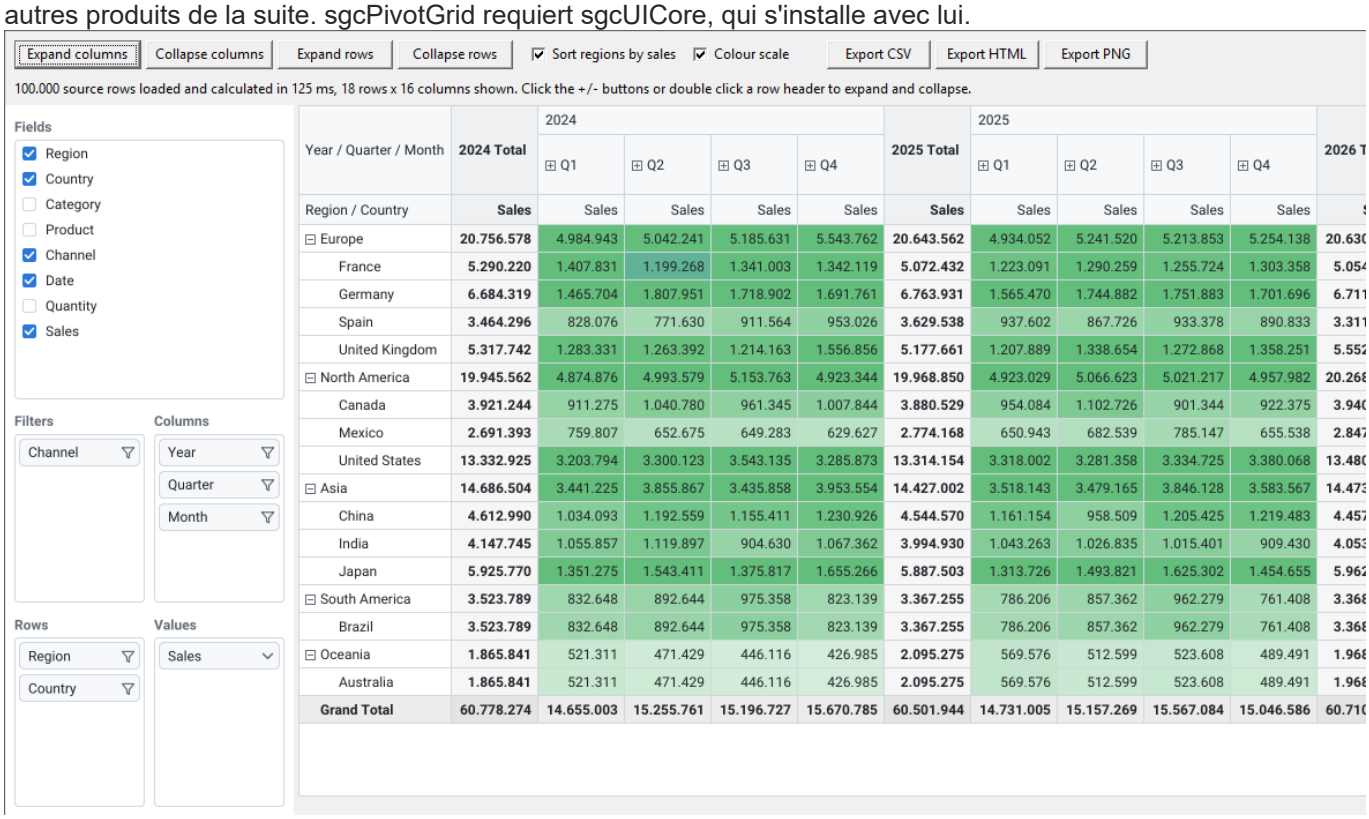
Les champs de saisie FireMonkey stylés conservent les couleurs de leur style. Utilisez un style sombre pour la fiche, ou définissez les couleurs du contrôle. Voir Thèmes.

Les contrôles FMX sont absents

Les contrôles FireMonkey requièrent Delphi XE6 ou une version ultérieure, et les info-bulles des contrôles FMX requièrent 10.1 Berlin ou une version ultérieure. Voir Plateformes.

Introduction

sgcPivotGrid ajoute des tableaux croisés dynamiques aux applications Delphi et C++Builder, pour VCL et FireMonkey. **TsgcVCLPivotGrid** et **TsgcFMXPivotGrid** résument des lignes source plates selon les champs que vous placez dans les lignes, les colonnes, les filtres et les valeurs : sommes, comptages, moyennes et plus, avec des sous-totaux et des totaux généraux, regroupement des dates par année, trimestre, mois, semaine ou jour, filtres, tri, développement et réduction, drill-down vers les lignes source et export vers CSV, JSON, HTML, PNG et SVG. **TsgcVCLPivotFieldList** et **TsgcFMXPivotFieldList** permettent à l'utilisateur de construire la mise en page par glisser-déposer.
Les données proviennent d'un **TDataSet** via **TsgcPivotSource_DataSet**, de n'importe quel **IsgcUITable**, tel qu'un sgcGrid, via **TsgcPivotSource_Table**, ou de votre propre code avec AppendRow. Le calcul est effectué par **TsgcPivotEngine**, une classe non visuelle dotée d'un magasin de colonnes typé. Le même tableau croisé fonctionne donc aussi sans fenêtre, sur un serveur ou sous Linux64, et **TsgcPivotGridRenderer** le dessine en PNG et SVG. Un million de lignes source se recalculent en une fraction de seconde.
sgcPivotGrid fait partie de la suite visuelle d'eSeGeCe. Il est dessiné avec **sgcUICore**, le package d'exécution partagé qui fournit le canevas, le rastériseur, les polices, l'écrivain SVG et les thèmes clair et sombre. Un tableau croisé a donc le même aspect sous VCL, sous FireMonkey et dans la sortie sans interface, et s'accorde avec les autres produits de la suite. sgcPivotGrid requiert sgcUICore, qui s'installe avec lui.



Year / Quarter / Month	2024 Total	2024				2025 Total	2025				2026 T
		⊞ Q1	⊞ Q2	⊞ Q3	⊞ Q4		⊞ Q1	⊞ Q2	⊞ Q3	⊞ Q4	
Region / Country	Sales	Sales	Sales	Sales	Sales	Sales	Sales	Sales	Sales	Sales	
⊟ Europe	20.756.578	4.984.943	5.042.241	5.185.631	5.543.762	20.643.562	4.934.052	5.241.520	5.213.853	5.254.138	20.630
France	5.290.220	1.407.831	1.199.268	1.341.003	1.342.119	5.072.432	1.223.091	1.290.259	1.255.724	1.303.358	5.054
Germany	6.684.319	1.465.704	1.807.951	1.718.902	1.691.761	6.763.931	1.565.470	1.744.882	1.751.883	1.701.696	6.711
Spain	3.464.296	828.076	771.630	911.564	953.026	3.629.538	937.602	867.726	933.378	890.833	3.311
United Kingdom	5.317.742	1.283.331	1.263.392	1.214.163	1.556.856	5.177.661	1.207.889	1.338.654	1.272.868	1.358.251	5.552
⊟ North America	19.945.562	4.874.876	4.993.579	5.153.763	4.923.344	19.968.850	4.923.029	5.066.623	5.021.217	4.957.982	20.268
Canada	3.921.244	911.275	1.040.780	961.345	1.007.844	3.880.529	954.084	1.102.726	901.344	922.375	3.940
Mexico	2.691.393	759.807	652.675	649.283	629.627	2.774.168	650.943	682.539	785.147	655.538	2.847
United States	13.332.925	3.203.794	3.300.123	3.543.135	3.285.873	13.314.154	3.318.002	3.281.358	3.334.725	3.380.068	13.480
⊟ Asia	14.686.504	3.441.225	3.855.867	3.435.858	3.953.554	14.427.002	3.518.143	3.479.165	3.846.128	3.583.567	14.473
China	4.612.990	1.034.093	1.192.559	1.155.411	1.230.926	4.544.570	1.161.154	958.509	1.205.425	1.219.483	4.457
India	4.147.745	1.055.857	1.119.897	904.630	1.067.362	3.994.930	1.043.263	1.026.835	1.015.401	909.430	4.053
Japan	5.925.770	1.351.275	1.543.411	1.375.817	1.655.266	5.887.503	1.313.726	1.493.821	1.625.302	1.454.655	5.962
⊟ South America	3.523.789	832.648	892.644	975.358	823.139	3.367.255	786.206	857.362	962.279	761.408	3.368
Brazil	3.523.789	832.648	892.644	975.358	823.139	3.367.255	786.206	857.362	962.279	761.408	3.368
⊟ Oceania	1.865.841	521.311	471.429	446.116	426.985	2.095.275	569.576	512.599	523.608	489.491	1.968
Australia	1.865.841	521.311	471.429	446.116	426.985	2.095.275	569.576	512.599	523.608	489.491	1.968
Grand Total	60.778.274	14.655.003	15.255.761	15.196.727	15.670.785	60.501.944	14.731.005	15.157.269	15.567.084	15.046.586	60.710

Éditions

sgcPivotGrid est proposé en Registered, Trial et Community. L'édition Community omet quatre fonctions avancées : champs calculés, afficher les valeurs en tant que, Top N et persistance de la mise en page. Voir Éditions.

Par où commencer

- Architecture : les sources, le magasin de colonnes, le pipeline du moteur, l'instantané et les vues.
- Plateformes : les versions de Delphi, C++Builder et les plateformes.
- Démarrage rapide VCL, Démarrage rapide FMX, Démarrage rapide C++Builder, Démarrage rapide DataSet et Démarrage rapide sans interface : un premier tableau croisé en quelques minutes.
- Composants : les grilles, les listes de champs, le moteur et ses champs, les sources, le moteur de rendu et les règles de format.

- Zones, Agrégations, Regroupement des dates, Totaux, Filtres, Top N, Afficher les valeurs en tant que, Champs calculés, Persistance de la mise en page et Export.
- Développer et réduire, Tri, Drill-down, Mise en forme conditionnelle, Interopérabilité avec les graphiques, Clavier et accessibilité et Performances.
- Démos : les projets de démonstration Delphi et C++Builder.
- Dépannage : problèmes courants et leurs causes.

Architecture

sgcPivotGrid sépare le calcul du dessin, et le dessin du framework. Le moteur et le moteur de rendu n'utilisent que la RTL et sgcUICore, et les contrôles VCL et FMX sont de fines enveloppes autour d'eux.

Flux de données

1. **Sources.** TsgcPivotSource_DataSet lit un TDataSet, TsgcPivotSource_Table lit n'importe quel IsgcUITable, et votre code peut appeler directement AppendRow de TsgcPivotEngine. Toutes écrivent dans le même magasin de colonnes.
2. **Magasin de colonnes.** Les colonnes de texte sont encodées par dictionnaire. Les nombres, les dates et les booléens sont des valeurs Double, avec NaN comme valeur vide. Le type d'une colonne vient du type de champ DB, de DefineColumn ou de sa première valeur non vide, et une colonne d'entiers devient une colonne de nombres flottants lorsqu'une valeur fractionnaire arrive.
3. **Moteur.** Chaque recalcul encode les membres des champs de ligne, de colonne et de filtre, applique les filtres, agrège les cellules feuilles en un seul passage sur les lignes source, applique les filtres de valeur et Top N, remonte les feuilles dans les sous-totaux et les totaux généraux avec des fusions exactes, trie les membres et aplatit les axes avec leur état développé. Les totaux sont calculés à partir des valeurs source, jamais comme des moyennes de moyennes.
4. **Instantané.** Le résultat est publié sous forme de TsgcPivotResult immuable. Les lecteurs le prennent entre Lock et Unlock, de sorte qu'une vue dessine toujours un résultat complet. Développer ou réduire un membre construit un nouvel instantané sur les mêmes données, sans réagréger.
5. **Sortie.** TsgcPivotGridRenderer dessine un instantané sur n'importe quel canevas sgcUICore : l'écran, un PNG ou un SVG. TsgcVCLPivotGrid et TsgcFMXPivotGrid hébergent le moteur de rendu avec le défilement, la sélection, le clavier et la souris. Les méthodes d'export écrivent la vue en CSV, HTML ou JSON, et ResultTable l'expose sous forme d'IsgcUITable, par exemple à sgcCharts.

Classes

Classe	Unité	Rôle
TsgcVCLPivotGrid	sgcPivotGrid_VCL	Le tableau croisé VCL. Il possède un moteur, publié sous le nom Pivot.
TsgcFMXPivotGrid	sgcPivotGrid_FMX	Le tableau croisé FireMonkey.
TsgcVCLPivotFieldList	sgcPivotGrid_FieldList_VCL	La liste de champs VCL, liée à une grille ou à un moteur.
TsgcFMXPivotFieldList	sgcPivotGrid_FieldList_FMX	La liste de champs FireMonkey.
TsgcPivotEngine	sgcPivotGrid_Engine	Le magasin de colonnes, la mise en page, le calcul, l'instantané, le drill-down et l'export.
TsgcPivotField	sgcPivotGrid_Fields	Un champ de la mise en page : sa zone, son regroupement, son agrégation, son tri et ses filtres.
TsgcPivotCalculatedField	sgcPivotGrid_Fields	Une colonne calculée par ligne source à partir d'une expression.
TsgcPivotSource_DataSet	sgcPivotGrid_Source_DataSet	Charge un TDataSet et le recharge lorsque les données changent.
TsgcPivotSource_Table	sgcPivotGrid_Source_Table	Charge n'importe quel IsgcUITable et le recharge lorsque sa version change.

TsgcPivotGridRenderer	sgcPivotGrid_Render	Dessine un instantané sur un canevas, en PNG ou SVG, et effectue le test de position.
TsgcPivotFormatRule	sgcPivotGrid_Format	Une règle de mise en forme conditionnelle : une règle de valeur ou une échelle de couleurs.

Threads

Un recalcul s'exécute dans le thread qui le déclenche, normalement le thread principal, et OnCalculated est déclenché dans ce thread. L'instantané est protégé par un verrou, de sorte que le code d'un autre thread ne lit Result qu'entre Lock et Unlock. TsgcPivotSource_Table recharge dans le thread qui notifie la modification de sa table.

Performances

Le groupe de performance du QA mesure ces scénarios avec une compilation release : une exécution de préchauffage, puis la médiane de cinq exécutions, sur des données issues d'une graine fixe pour que chaque exécution soit identique. Le tableau indique le budget de chaque scénario.

Scénario	Win64	Win32
Recalcul de 1 000 000 de lignes, 2 champs de ligne (50 et 20 membres), 1 champ de colonne (12 membres), 1 Sum	100 ms	160 ms
AppendRow de 1 000 000 de lignes avec 3 valeurs de texte, 1 flottante et 1 date	900 ms	1,400 ms
La première mise en page avec CountDistinct sur une colonne de 100 000 valeurs	400 ms	600 ms
La première mise en page avec StdDev, Average et Max comme trois mesures	180 ms	280 ms
Une image en 1920x1080 de 10 000 lignes et 50 colonnes de données, défilée à chaque image	16 ms	25 ms
Développer ou réduire un membre d'un axe de 10 000 éléments	5 ms	8 ms
Mémoire du moteur au-delà du magasin pour le premier scénario	32 MB	32 MB

Voir aussi

- Plateformes
- TsgcPivotEngine
- Démarrage rapide sans interface

Éditions

sgcPivotGrid est construit à partir des mêmes sources pour chaque édition.

Édition	Contient
Registered	Tout : les grilles et listes de champs VCL et FMX, le moteur, les deux sources, toutes les agrégations, le regroupement des dates, les totaux, les filtres, le tri, le drill-down, les règles de format, l'export, la sortie PNG et SVG sans interface, et les quatre fonctions avancées : champs calculés, afficher les valeurs en tant que, Top N et persistance de la mise en page.
Trial	Les mêmes fonctions que Registered. Une compilation avec SGC_TRIAL affiche un message indiquant qu'il s'agit d'une version d'évaluation lorsqu'un composant est créé après la fin de la fenêtre de publication indiquée dans la compilation. Rien d'autre ne change.
Community	Tout sauf les quatre fonctions avancées. CountDistinct, StdDev, Var, le regroupement des dates, les filtres, le drill-down, l'export, les règles de format, les listes de champs et les vues font tous partie de Community, et il n'y a pas d'avis d'évaluation.

Les fonctions avancées

Les fonctions avancées sont compilées sous le define SGC_PIVOTGRID_ADVANCED, que sgcVer.inc désactive dans l'édition Community. Leurs unités, sgcPivotGrid_Expr, sgcPivotGrid_ShowAs et sgcPivotGrid_Layout, ne font pas partie de la distribution Community, et ces membres n'existent pas dans une compilation Community :

Fonction	Membres
Champs calculés	TsgcPivotEngine.CalculatedFields, TsgcPivotCalculatedField, TsgcPivotCalculatedFields
Afficher les valeurs en tant que	TsgcPivotField.ShowAs, ShowAsAxis et ShowAsBaseItem
Top N	TsgcPivotField.TopN et TopNOthers, TsgcPivotEngine.OthersCaption
Persistance de la mise en page	SaveLayoutToString, LoadLayoutFromString, SaveLayoutToFile et LoadLayoutFromFile du moteur et des deux grilles

L'édition est un choix à la compilation dans sgcVer.inc. SGC_COMMUNITY désactive aussi SGC_TRIAL. Le code qui utilise un membre avancé et doit aussi compiler avec Community le place dans {$IFDEF SGC_PIVOTGRID_ADVANCED}. Une fiche enregistrée par une compilation Registered avec des propriétés avancées définies ne se charge pas dans une compilation Community.

Voir aussi

- Plateformes
- Introduction

Installation

sgcPivotGrid fait partie de la distribution sgcUI. Un seul installeur couvre les produits sgcUI, et il n'installe que les produits inclus dans votre licence.

Installation avec l'installeur sgcUI

1. Lancez l'installeur sgcUI et connectez-vous avec le nom d'utilisateur et le mot de passe de votre compte eSeGeCe.
2. Choisissez les versions de Delphi et de C++Builder dans lesquelles installer.
3. Sur la page des composants, sélectionnez **sgcPivotGrid** et les autres produits souhaités. Le serveur de licences retourne les produits que vous possédez, et l'installeur affiche une case à cocher par produit. Les produits que votre licence n'inclut pas sont grisés. Une licence All-Access déverrouille tous les produits.
4. L'installeur copie les fichiers, puis compile les packages sur votre machine pour chaque IDE sélectionné, dans l'ordre des dépendances, et installe les packages de conception.

Ce qui accompagne sgcPivotGrid

sgcUICore, la base commune des produits visuels sgcUI, est fourni avec sgcPivotGrid. Il n'a pas de case à cocher propre. L'installeur l'installe automatiquement chaque fois que sgcPivotGrid est sélectionné.

Dossiers installés

Chaque produit s'installe dans un seul dossier, par défaut `Documents\esegece\sgcUI` (`sgcUI_Trial` pour l'évaluation et `sgcUI_Community` pour l'édition Community).

Dossier	Contenu
`Source`	Le code source de chaque produit installé. C'est l'unique chemin de bibliothèque de sgcUI.
`Packages`	Les packages de chaque produit installé, un jeu par produit et par version d'IDE.
`Lib<V>`	Les unités compilées pour chaque version d'IDE et plateforme, par exemple `LibD13` .
`Demos\PivotGrid`	Les démos de sgcPivotGrid. Voir Démos.

L'installeur ajoute `sgcUI\Source` au chemin de bibliothèque et au chemin de navigation de chaque IDE et plateforme sélectionnés, avec le dossier des unités compilées. sgcUICore se trouve dans le même dossier `Source`, il n'a donc pas besoin de chemin propre.

Trial et Community

- **Trial** : l'installeur d'évaluation sgcUI installe chaque produit en mode évaluation, sgcPivotGrid compris.
- **Community** : l'installeur gratuit sgcUI Community inclut sgcPivotGrid sans ses quatre fonctions avancées. Voir Éditions.

Compilation manuelle des packages

Pour compiler vous-même les packages, ouvrez-les pour votre version d'IDE depuis `sgcUI\Packages`. `<V>` est le suffixe de la version de Delphi, par exemple `sgcPivotGridD13.dpk`.

Package	Contenu
`sgcUICore<V>`	La base commune sgcUICore (exécution).
`sgcPivotGrid<V>` / `dclsgcPivotGrid<V>`	Les composants sgcPivotGrid (exécution / conception).

Compilez-les dans l'ordre des dépendances : `sgcUICore`, puis `sgcPivotGrid`, puis `dclsgcPivotGrid`. Installez ensuite le package de conception `dclsgcPivotGrid` et ajoutez `sgcUI\Source` au chemin de bibliothèque de votre version de Delphi (Tools > Options > Language > Delphi Options > Library > Library path). Pour C++Builder, ouvrez le projet `.cbproj` correspondant à votre version, par exemple `sgcPivotGridC13.cbproj`, et compilez-le. Les en-têtes `.hpp` générés apparaissent à côté du package compilé.

Voir aussi

- Plateformes
- Démarrage rapide VCL

Plateformes

Cible	Prise en charge
Delphi	Delphi 7, 2007, 2009, 2010, XE à XE8, 10 à 10.4, 11, 12 et 13.
C++Builder	C++Builder 2007, 2010, XE à XE8, 10 à 10.4, 11, 12 et 13. Les packages génèrent les en-têtes .hpp et les fichiers .lib qu'un projet C++Builder inclut et lie.
VCL	TsgcVCLPivotGrid et TsgcVCLPivotFieldList sous Windows, Win32 et Win64, toutes les versions de Delphi.
FireMonkey	TsgcFMXPivotGrid et TsgcFMXPivotFieldList à partir de Delphi XE6 (SGC_PIVOTGRID_FMX). Ils n'utilisent que FireMonkey et sgcUICore, avec leurs propres barres de défilement et le défilement tactile.
Linux64	Les unités indépendantes du framework à partir de Delphi 10.2 (SGC_PIVOTGRID_LINUX) : le moteur, le magasin de colonnes, les deux sources, l'export, le JSON de mise en page, le moteur de rendu avec la sortie PNG et SVG sans interface, et le modèle de la liste de champs.
.NET	Un portage C# avec la même API pour WinForms, WPF et Avalonia, la sortie PNG et SVG sans interface et la source DataTable, sous .NET Framework 4.7.2 et .NET 8 et 9, voir Présentation de .NET.

Packages

sgcPivotGrid fournit un package d'exécution indépendant du framework et un package d'exécution par framework, de sorte qu'une application FireMonkey pour Android, iOS ou macOS ne lie jamais la VCL. Chaque nom de package se termine par le suffixe de l'IDE, par exemple sgcPivotGridD13.dpk pour Delphi 13 et sgcPivotGridC13.cbproj pour C++Builder 13.

Package	Type	Contenu	Requiert
sgcPivotGrid	Exécution, indépendant du framework	Chaque unité du produit qui n'utilise ni la VCL ni FireMonkey.	rtl, dbrtl, sgcUICore
sgcPivotGridVCL	Exécution, VCL	sgcPivotGrid_VCL et sgcPivotGrid_FieldList_VCL (TsgcVCLPivotGrid, TsgcVCLPivotFieldList)	rtl, vcl, sgcUICore, sgcUICoreVCL, sgcPivotGrid
sgcPivotGridFMX	Exécution, FireMonkey, à partir de XE6	sgcPivotGrid_FMX et sgcPivotGrid_FieldList_FMX (TsgcFMXPivotGrid, TsgcFMXPivotFieldList)	rtl, fmx, sgcUICore, sgcUICoreFMX, sgcPivotGrid
dclsgcPivotGrid	Conception, VCL	Enregistre TsgcPivotSource_DataSet, TsgcPivotSource_Table, TsgcVCLPivotGrid et TsgcVCLPivotFieldList dans la page de palette **SGC PivotGrid**, avec l'éditeur de composant, la liste déroulante de FieldName et les éditeurs de collections.	designide, rtl, vcl, sgcPivotGrid, sgcPivotGridVCL
dclsgcPivotGridFMX	Conception, FireMonkey, à partir de XE6	Enregistre TsgcFMXPivotGrid et TsgcFMXPivotFieldList dans la page de palette **SGC PivotGrid**, avec leur éditeur de composant.	designide, rtl, fmx, sgcPivotGrid, sgcPivot-

			GridFMX, dclsgcPivotGrid
sgcPivot-GridWS, dclsgcPi-votGridWS	Exécution et conception, passerelle sgcWebSo-ckets	La passerelle de données en direct, indépendante du framework.	rtl, sgcPivotGrid, sgcWebSockets

sgcUICore suit la même répartition : sgcUICore, sgcUICoreVCL et sgcUICoreFMX, sans package de conception. dclsgcPivotGridFMX requiert dclsgcPivotGrid, installez donc d'abord dclsgcPivotGrid.

Versions de l'IDE et plateformes cibles

Delphi	sgcPivotGrid	sgcPivot-GridVCL	sgcPivotGridFMX	dclsgcPivotGrid, dclsgcPivot-GridFMX
7, 2007, 2009, 2010, XE	Win32	Win32	Aucune	Win32
XE2 à XE5	Win32, Win64	Win32, Win64	Aucune	Win32
XE6 à 10.2	Win32, Win64	Win32, Win64	Win32, Win64	Win32
10.3	Win32, Win64, Linux64, OSX64	Win32, Win64	Win32, Win64, OSX64	Win32
10.4	Win32, Win64, Linux64, OSX64, Android, An-droid64, iOSDevice64	Win32, Win64	Win32, Win64, OSX64, Android, Android64, iOSDevice64	Win32
11	Win32, Win64, Linux64, OSX64, OSXARM64, Android, Android64, iOSDevice64, iOSSi-mARM64	Win32, Win64	Win32, Win64, OSX64, OSXARM64, Android, Android64, iOSDe-vice64, iOSSimARM64	Win32
12, 13	Comme 11	Win32, Win64	Comme 11	Win32, Win64

De XE6 à 10.2, le package FireMonkey est compilé pour Windows uniquement. Les packages de conception sont chargés par l'IDE, ils sont donc compilés pour la plateforme de l'IDE uniquement.

C++Builder	sgcPivotGrid, sgcPivot-GridVCL	sgcPivot-GridFMX	dclsgcPivotGrid, dclsgcPivot-GridFMX
2007, 2010, XE, XE2	Win32	Aucune	Win32 (dclsgcPivotGrid unique-ment)
XE3 à XE5	Win32, Win64	Aucune	Win32 (dclsgcPivotGrid unique-ment)
XE6 à 12	Win32, Win64	Win32, Win64	Win32

13	Win32, Win64, Win64x	Win32, Win64, Win64x	Win32, Win64x

Les projets C++Builder portent les mêmes noms avec le suffixe C, par exemple sgcPivotGridC13.cbproj, sgcPivotGridVCLC13.cbproj, sgcPivotGridFMXC13.cbproj, dclsgcPivotGridC13.cbproj et dclsgcPivotGridFMXC13.cbproj. Les packages C++Builder sont compilés pour Windows uniquement. Les notes C++Builder qui s'appliquent à chaque produit sgcUI (packages d'exécution sous Win64, C++Builder 10.3 sous Win64, noms de fichiers des packages Win64x) se trouvent dans le sujet Plateformes et packages de l'aide de la suite sgcUI.

FireMonkey sur Android, iOS et macOS

Les contrôles FireMonkey (TsgcFMXPivotGrid et TsgcFMXPivotFieldList) ciblent macOS à partir de Delphi 10.3, Android et iOS à partir de Delphi 10.4, et macOS ARM et le simulateur iOS à partir de Delphi 11. Dans un projet FireMonkey pour ces plateformes :

- Utilisez les contrôles FireMonkey. Les contrôles VCL et sgcPivotGridVCL ne fonctionnent que sous Windows.
- Déposez le contrôle sur un formulaire, ou ajoutez son unité à la clause uses : sgcPivotGrid_FMX et sgcPivotGrid_FieldList_FMX.
- Sélectionnez la plateforme cible dans la fenêtre Projets. L'installateur ajoute le dossier Source de sgcUI et les unités compilées de chaque plateforme (LibD<version>\<platform>) au chemin de bibliothèque de chaque plateforme qu'il compile. Sans l'installateur, ajoutez le dossier Source de sgcUI au chemin de bibliothèque de la plateforme cible (Outils, Options, Langage, Delphi, Bibliothèque).
- Les applications Android et iOS sont liées statiquement, aucun package d'exécution n'est donc déployé.

Linux64

Le package indépendant du framework sgcPivotGrid est compilé pour Linux64 à partir de Delphi 10.3. Une application Linux64, par exemple un programme console ou un serveur, utilise le moteur de pivot et les sources. Aucun package VCL ou FireMonkey n'est compilé pour Linux64, car FireMonkey sous Linux nécessite FMX pour Linux (FMXLinux), un module complémentaire tiers.

Contraintes de version

- Chaque unité compile sur Delphi 7 : pas de génériques, pas de méthodes anonymes, pas de variables inline et pas d'assistants d'enregistrement.
- Les en-têtes C++Builder conservent des préfixes d'énumération uniques (pva, pvf, pvd, pvx...), de sorte qu'ils n'entrent jamais en conflit avec sgcGrid ou sgcCharts.

Voir aussi

- Éditions
- Architecture

Démarrage rapide VCL

Installez les packages sgcPivotGrid pour votre version de Delphi. Les composants se trouvent sur la page de palette **SGC PivotGrid**.

À la conception

1. Déposez un **TsgcVCLPivotGrid** sur une fiche et mettez Align à alClient.
2. Déposez un **TsgcVCLPivotFieldList**, mettez Align à alRight et sa propriété PivotGrid sur la grille.
3. Déposez un **TsgcPivotSource_DataSet**, mettez DataSet sur votre jeu de données, PivotGrid sur la grille et Active à True.
4. Développez la propriété Pivot de la grille et ajoutez des éléments à Pivot.Fields. Choisissez une colonne dans la liste déroulante de FieldName et définissez Area : pvaRow, pvaColumn, pvaFilter ou pvaData. Un champ de données a aussi une Aggregation, Sum par défaut.
5. Lancez l'application. L'utilisateur réorganise les champs avec la liste de champs, et la grille recalcule après chaque modification.

Dans le code

La grille possède son moteur, la propriété Pivot. Chargez les lignes entre BeginLoad et EndLoad, et modifiez la mise en page entre BeginUpdate et EndUpdate, afin que chaque bloc recalcule une seule fois.

```
uses
  SysUtils, Controls, Forms,
  sgcPivotGrid_Types, sgcPivotGrid_Engine, sgcPivotGrid_VCL,
  sgcPivotGrid_FieldList_VCL;

procedure CreatePivot(aForm: TForm);
var
  oGrid: TsgcVCLPivotGrid;
  oList: TsgcVCLPivotFieldList;
begin
  // the field list on the right, the grid fills the rest of the form
  oList := TsgcVCLPivotFieldList.Create(aForm);
  oList.Parent := aForm;
  oList.Align := alRight;
  oList.Width := 260;

  oGrid := TsgcVCLPivotGrid.Create(aForm);
  oGrid.Parent := aForm;
  oGrid.Align := alClient;
  oList.PivotGrid := oGrid;

  // the source rows, one recalculation at EndLoad
  oGrid.Pivot.BeginLoad;
  try
    oGrid.Pivot.DefineColumn('Region', pvtText);
    oGrid.Pivot.DefineColumn('Product', pvtText);
    oGrid.Pivot.DefineColumn('Date', pvtDateTime);
    oGrid.Pivot.DefineColumn('Sales', pvtFloat);
    oGrid.Pivot.AppendRow(['North', 'Bikes', EncodeDate(2025, 1, 14), 1200.0]);
    oGrid.Pivot.AppendRow(['North', 'Helmets', EncodeDate(2025, 2, 3), 310.5]);
    oGrid.Pivot.AppendRow(['South', 'Bikes', EncodeDate(2025, 4, 22), 980.0]);
    oGrid.Pivot.AppendRow(['South', 'Helmets', EncodeDate(2026, 1, 9), 145.0]);
  finally
    oGrid.Pivot.EndLoad;
  end;

  // the layout, one recalculation at EndUpdate
  oGrid.Pivot.BeginUpdate;
```

```
  try
    oGrid.Pivot.AddField('Region', pvaRow);
    oGrid.Pivot.AddField('Product', pvaRow);
    oGrid.Pivot.AddField('Date', pvaColumn, pvdYear);
    oGrid.Pivot.AddDataField('Sales', pvfSum).DisplayFormat := '#,##0.00';
  finally
    oGrid.Pivot.EndUpdate;
  end;
end;
```

Voir aussi

- TsgcVCLPivotGrid
- TsgcVCLPivotFieldList
- Démarrage rapide DataSet
- Zones

Démarrage rapide FMX

Les composants FireMonkey nécessitent Delphi XE6 ou une version ultérieure. Ils se trouvent sur la page de palette **SGC PivotGrid** d'une fiche FireMonkey.

À la conception

1. Déposez un **TsgcFMXPivotGrid** sur une fiche et mettez Align à Client.
2. Déposez un **TsgcFMXPivotFieldList**, mettez Align à Right et sa propriété PivotGrid sur la grille.
3. Liez les données avec un **TsgcPivotSource_DataSet** ou un **TsgcPivotSource_Table**, ou chargez-les dans le code, et ajoutez les champs à Pivot.Fields comme dans le Démarrage rapide VCL.

La grille FMX dessine ses propres barres de défilement, affichées pendant qu'on les fait glisser, et défile avec la molette de la souris et avec un geste tactile. Sa méthode de test de position est HitAt, car FireMonkey publie déjà une propriété HitTest.

Dans le code

```
uses
  SysUtils, FMX.Types, FMX.Controls, FMX.Forms,
  sgcUICore_Theme, sgcPivotGrid_Types, sgcPivotGrid_Engine, sgcPivotGrid_FMX,
  sgcPivotGrid_FieldList_FMX;

procedure CreatePivot(aForm: TForm);
var
  oGrid: TsgcFMXPivotGrid;
  oList: TsgcFMXPivotFieldList;
begin
  oList := TsgcFMXPivotFieldList.Create(aForm);
  oList.Parent := aForm;
  oList.Align := TAlignLayout.Right;
  oList.Width := 260;

  oGrid := TsgcFMXPivotGrid.Create(aForm);
  oGrid.Parent := aForm;
  oGrid.Align := TAlignLayout.Client;
  oGrid.Theme := utDark;
  oList.PivotGrid := oGrid;
  oList.Theme := utDark;

  oGrid.Pivot.BeginLoad;
  try
    oGrid.Pivot.DefineColumn('Region', pvtText);
    oGrid.Pivot.DefineColumn('Channel', pvtText);
    oGrid.Pivot.DefineColumn('Date', pvtDateTime);
    oGrid.Pivot.DefineColumn('Sales', pvtFloat);
    oGrid.Pivot.AppendRow(['North', 'Online', EncodeDate(2025, 3, 2), 420.0]);
    oGrid.Pivot.AppendRow(['North', 'Retail', EncodeDate(2025, 7, 18), 610.0]);
    oGrid.Pivot.AppendRow(['South', 'Online', EncodeDate(2025, 11, 5), 275.0]);
  finally
    oGrid.Pivot.EndLoad;
  end;

  oGrid.Pivot.BeginUpdate;
  try
    oGrid.Pivot.AddField('Region', pvaRow);
    oGrid.Pivot.AddField('Date', pvaColumn, pvdQuarter);
    oGrid.Pivot.AddField('Channel', pvaFilter);
    oGrid.Pivot.AddDataField('Sales', pvfSum);
  finally
    oGrid.Pivot.EndUpdate;
```

```
  end;
end;
```

Voir aussi

- TsgcFMXPivotGrid
- TsgcFMXPivotFieldList
- Plateformes

Démarrage rapide C++Builder

Installez les packages sgcPivotGrid pour votre version de C++Builder, ajoutez `sgcUI\Source` et le dossier des unités compilées, par exemple `sgcUI\LibD13\win32` (`sgcUI\LibD13\win64` pour 64 bits), aux chemins d'inclusion et de bibliothèque du projet, et utilisez les composants de la page de palette **SGC PivotGrid** exactement comme dans Delphi. L'installeur ajoute les deux chemins à l'IDE pour vous. Les en-têtes .hpp et les fichiers .lib générés de sgcPivotGrid et de sgcUICore se trouvent dans ce seul dossier, sgcUICore n'a donc pas besoin de chemin propre.

À savoir en C++

- AppendRow prend un tableau ouvert de Variant, que C++ passe sous forme de tableau de Variant accompagné de son indice le plus élevé : `AppendRow(Row, 3)` pour quatre valeurs.
- L'en-tête généré n'a pas d'arguments par défaut pour les constructeurs de TsgcPivotEngine et de TsgcPivotGridRenderer. Passez NULL explicitement : `new TsgcPivotEngine(NULL)` et `new TsgcPivotGridRenderer(NULL, NULL)`. Voir Démarrage rapide sans interface.
- Les événements sont des closures, leurs gestionnaires sont donc des fonctions membres __fastcall d'une classe. Un paramètre var tel que aHandled est une référence bool.

Dans le code

Voir aussi

- TsgcVCLPivotGrid
- Démarrage rapide VCL
- Démos

Démarrage rapide DataSet

TsgcPivotSource_DataSet copie les enregistrements de n'importe quel TDataSet dans le moteur d'une grille et maintient le tableau croisé à jour. La mise en page désigne les champs du jeu de données par leur FieldName, elle peut donc être définie avant l'arrivée des données.

```
uses
  SysUtils, Classes, Controls, Forms, DB,
  sgcPivotGrid_Types, sgcPivotGrid_VCL, sgcPivotGrid_Source_DataSet;

procedure BindDataSet(aForm: TForm; aDataSet: TDataSet);
var
  oGrid: TsgcVCLPivotGrid;
  oSource: TsgcPivotSource_DataSet;
begin
  oGrid := TsgcVCLPivotGrid.Create(aForm);
  oGrid.Parent := aForm;
  oGrid.Align := alClient;

  // the layout names the fields of the dataset
  oGrid.Pivot.BeginUpdate;
  try
    oGrid.Pivot.AddField('Country', pvaRow);
    oGrid.Pivot.AddField('OrderDate', pvaColumn, pvdYear);
    oGrid.Pivot.AddField('OrderDate', pvaColumn, pvdQuarter);
    oGrid.Pivot.AddDataField('Amount', pvfSum);
    oGrid.Pivot.AddDataField('OrderNo', pvfCount);
  finally
    oGrid.Pivot.EndUpdate;
  end;

  // the source copies the dataset into the engine of the grid and
  // reloads it when the data changes
  oSource := TsgcPivotSource_DataSet.Create(aForm);
  oSource.Columns.CommaText := 'Country,OrderDate,Amount,OrderNo';
  oSource.DataSet := aDataSet;
  oSource.PivotGrid := oGrid;
  oSource.LiveUpdates := True;
  oSource.Active := True;
end;
```

Comment la source charge

- Columns liste les champs du jeu de données à charger. Vide, il charge tous les champs. Les champs BLOB, binaires et imbriqués sont toujours ignorés.
- Les types de colonne proviennent des types de champ : les champs chaîne sont du texte, les champs entiers et flottants sont des nombres, les champs date et heure sont des dates, les champs booléens sont des booléens.
- Le chargement s'exécute avec DisableControls et restaure le signet, de sorte que les contrôles liés au jeu de données ne scintillent pas.
- Avec LiveUpdates, un post, un append, un delete, un refresh, First ou Last recharge le tableau croisé, car TDataSet les signale comme des modifications de données. Le défilement avec Next, Prior ou MoveBy ne recharge pas. Un chargement ne s'exécute que tant que le jeu de données est en état de navigation.
- Load recharge à tout moment, Refresh recharge lorsque la source est Active.

Un projet C++Builder qui utilise TClientDataSet lie midas.lib avec `#pragma link "midas.lib"`, comme le fait la démo DataSetPivot.

Voir aussi

- TsgcPivotSource_DataSet

- TsgcPivotSource_Table
- Regroupement des dates

Démarrage rapide sans interface

Un tableau croisé n'a pas besoin de fenêtre. TsgcPivotEngine charge les lignes et calcule, ses méthodes d'export écrivent du CSV, du HTML et du JSON, et TsgcPivotGridRenderer dessine la même grille que les contrôles dans un PNG ou un SVG. Ces unités n'utilisent que la RTL et sgcUICore, un rapport s'exécute donc dans un service, sur un serveur web ou sous Linux64.

Delphi

```
uses
  SysUtils, Classes,
  sgcUICore_Theme, sgcPivotGrid_Types, sgcPivotGrid_Engine,
  sgcPivotGrid_Render;

procedure WriteReport(const aFolder: string);
var
  oEngine: TsgcPivotEngine;
  oRenderer: TsgcPivotGridRenderer;
  vPath: string;
begin
  vPath := IncludeTrailingPathDelimiter(aFolder);
  oEngine := TsgcPivotEngine.Create;
  try
    oEngine.BeginLoad;
    try
      oEngine.DefineColumn('Region', pvtText);
      oEngine.DefineColumn('Channel', pvtText);
      oEngine.DefineColumn('Sales', pvtFloat);
      oEngine.AppendRow(['North', 'Online', 1200.0]);
      oEngine.AppendRow(['North', 'Retail', 800.0]);
      oEngine.AppendRow(['South', 'Online', 450.0]);
      oEngine.AppendRow(['South', 'Partner', 300.0]);
    finally
      oEngine.EndLoad;
    end;

    oEngine.BeginUpdate;
    try
      oEngine.AddField('Region', pvaRow);
      oEngine.AddField('Channel', pvaColumn);
      oEngine.AddDataField('Sales', pvfSum);
    finally
      oEngine.EndUpdate;
    end;

    // the whole view with its totals as tables
    oEngine.SaveToCSV(vPath + 'sales.csv');
    oEngine.SaveToHTML(vPath + 'sales.html');
    oEngine.SaveToJSON(vPath + 'sales.json');

    // the grid as an image, drawn on the sgcUICore canvases
    oRenderer := TsgcPivotGridRenderer.Create;
    try
      oRenderer.Options.Theme := utDark;
      oRenderer.SaveToPNG(oEngine, sgcPivotRenderState, vPath + 'sales.png');
      oRenderer.SaveToSVG(oEngine, sgcPivotRenderState, vPath + 'sales.svg');
    finally
      oRenderer.Free;
    end;
  finally
    oEngine.Free;
```

```
  end;
end;
```

Remarques

- sgcPivotRenderState retourne un état de vue vide : ni défilement, ni sélection, ni remplacements de largeur de colonne. Les surcharges du moteur de rendu qui prennent le moteur le verrouillent pendant qu'elles dessinent.
- Une largeur et une hauteur de 0 dessinent tout le contenu. Les tailles sont en DIP et l'argument d'échelle les multiplie.

Voir aussi

- Export
- TsgcPivotGridRenderer
- Démos

Démos

Les démos s'installent dans `Demos\PivotGrid`, avec les projets Delphi dans `Delphi` et leurs portages C++Builder dans `CBuilder`. Chaque démo accepte `--test`, qui exécute son autotest et quitte, et `/snapshot <file.png>`, qui enregistre une image du tableau croisé.

Démo	Framework	Ce qu'elle montre
SalesPivot	VCL	100 000 lignes de ventes : regroupement des dates par année, trimestre et mois, développement et réduction, un tri par valeur, une échelle de couleurs, la liste de champs et l'export en CSV, HTML et PNG.
DataSetPivot	VCL	Un tableau croisé sur un TClientDataSet : une modification, un append ou un delete dans le TDBGrid le recharge, et un double-clic sur une cellule fait un drill-down vers les commandes source.
MillionRows	VCL	1 000 000 de lignes ajoutées dans un seul bloc de chargement, CountDistinct d'une colonne de 100 000 valeurs et StdDev à côté de Sum, avec le recalcul chronométré.
AdvancedPivot	VCL	Un champ calculé Profit, afficher les valeurs en tant que, Top N avec Others, et la mise en page enregistrée dans un fichier JSON puis rechargée.
GridToPivotToChart	VCL	Les commandes saisies dans un sgcGrid alimentent le tableau croisé par TsgcPivotSource_Table, et le résultat du tableau croisé alimente sgcCharts par IsgcUITable.
SalesPivotFMX	FireMonkey	50 000 lignes avec la mise en page SalesPivot, défilement tactile, le thème sombre et l'export PNG.
HeadlessReport	Console	Lit un fichier CSV et écrit report.png, report.svg, report.html, report.csv et report.json sans framework d'interface.
AskYourPivot	VCL	Une question en langage naturel devient la disposition du tableau croisé grâce à TsgcPivotGridAIQuery et à un TsgcAIChat : les champs fournisseur, modèle et clé d'API, Ask, Undo, questions d'exemple et Show prompt. Édition Registered, requiert sgcWebSockets. Voir Interrogez votre tableau croisé.

Voir aussi

- Démarrage rapide VCL
- Démarrage rapide sans interface
- Interrogez votre tableau croisé

TsgcVCLPivotGrid

Unité : sgcPivotGrid_VCL
TsgcVCLPivotGrid est le tableau croisé VCL, uniquement pour Windows. Il possède un TsgcPivotEngine, publié sous le nom Pivot, et dessine son résultat avec des lignes et des colonnes virtuelles : un coin figé, des en-têtes de colonne à plusieurs niveaux avec fusions, des en-têtes de ligne, des boutons + et -, des totaux et des totaux généraux. Il défile avec les barres natives et la molette de la souris (Maj défile horizontalement), redimensionne une colonne par le bord de son en-tête et dispose d'une interface clavier : les flèches, Page préc., Page suiv., Début et Fin déplacent la sélection, Droite et Gauche (ou + et -) développent et réduisent un membre lorsque l'en-tête de ligne a le focus, et Entrée lance un drill-down. Il hérite de TsgcUIVCLControl et publie Align, Anchors, Constraints, Enabled, PopupMenu, ShowHint, ParentShowHint, TabOrder, TabStop et Visible.

Propriétés

Propriété	Type	Description
Pivot	TsgcPivotEngine	Le moteur que possède la grille : ses champs, ses données, ses totaux et ses libellés, sérialisés avec la fiche. La grille implémente IsgcUITable en lui déléguant.
BorderStyle	TBorderStyle	bsSingle (par défaut) ou bsNone.
RowLayout	TsgcPivotRowLayout	pvlCompact (par défaut) dessine les niveaux de ligne dans une colonne d'en-tête indentée, pvlTabular donne à chaque niveau sa propre colonne. Voir Zones.
RowHeight	Integer	La hauteur d'une ligne en DIP. Par défaut 24.
HeaderHeight	Integer	La hauteur d'une ligne d'en-tête de colonne en DIP. Par défaut 26.
IndentWidth	Integer	L'indentation de chaque niveau de ligne dans la mise en page compacte, en DIP. Par défaut 16.
MinColumnWidth, MaxColumnWidth	Integer	Les limites des largeurs automatiques des colonnes de données, en DIP. Par défaut 48 et 320.
AutoColumnWidths	Boolean	Ajuste les colonnes de données à leur contenu. Si False, elles font 96 DIP de large. Les colonnes d'en-tête de ligne s'ajustent toujours. Par défaut True.
GridLines	Boolean	Dessine les lignes des cellules. Par défaut True.
AlternateRows	Boolean	Grise une ligne sur deux. Par défaut False.
TotalsBold	Boolean	Dessine les totaux en gras. Par défaut True.
ShowExpandButtons	Boolean	Affiche les boutons `+` et `-` des membres qui ont des enfants. Par défaut True.
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom.
AccentColor	TsgcUIColor	Une couleur d'accent $AARRGGBB. 0 (par défaut) utilise l'accent du thème.

FontSize	Double	La taille de police des cellules en points. Par défaut 12.
FormatRules	TsgcPivot-Forma-tRules	Les règles de mise en forme conditionnelle des cellules.
MaxFPS	Integer	Le plus grand nombre d'images par seconde que dessine le contrôle.
SelectedRow	Integer	Publique. L'élément de ligne sélectionné, -1 pour aucun.
SelectedCol	Integer	Publique. La colonne de données sélectionnée, -1 lorsque l'en-tête de ligne a le focus ou que rien n'est sélectionné.
TopRow	Integer	Publique. Le premier élément de ligne affiché sous les en-têtes de colonne.
ScrollX	Integer	Publique. Le décalage de défilement horizontal des colonnes de données.
Renderer	TsgcPivot-GridRende-rer	Publique, en lecture seule. Le moteur de rendu qui dessine la grille.

Méthodes

Méthode	Re-tourne	Description
HitTest(X, Y)	TsgcPi-votHitInfo	L'élément sous un point en pixels de l'appareil : Part (pvh-None, pvhCorner, pvhColumnHeader, pvhRowHeader, pvh-Cell, pvhExpandButton, pvhColumnResize), RowItem, Data-Col et Level.
ScrollToCell(aRowItem, aData-Col)		Défile jusqu'à ce que la cellule soit visible. -1 laisse cet axe inchangé.
Recalculate		Recalcule le moteur immédiatement.
SaveToPNG, SaveToSVG(aFileName or aS-tream, aWidth = 0, aHeight = 0)		Toute la vue sous forme d'image, tailles en DIP, 0 corres-pond à la taille du contenu, jusqu'à 16384 pixels.
SaveToCSV, SaveToJSON, SaveToHTML(aFileName or aS-tream)		La vue actuelle sous forme de tableau. Voir Export.
SaveLayoutToString, Load-LayoutFromString, SaveLayout-ToFile, LoadLayoutFromFile		La mise en page de Pivot au format JSON, avec les lar-geurs de colonne de la grille. Édition Advanced. Voir Persis-tance de la mise en page.

Événements

Événement	Type	Description
OnCellClick, OnCellDbl-Click	TsgcVCLPivo-tHitEvent	Un clic ou un double-clic, avec les informations de position du point.

OnDrillDown	TsgcVCLPivotDrillDownEvent	Entrée ou un double-clic sur une cellule : aRowItem, aDataCol et var aHandled. La grille n'a pas d'action intégrée, appelez Pivot.DrillDown ou Pivot.GetSourceRows pour lire les lignes source.
OnExpandChange	TsgcVCLPivotExpandEvent	Après le développement ou la réduction d'un membre par l'utilisateur : aAxis et aItem.
OnGetCellStyle	TsgcPivotCellStyleEvent	Avant le dessin d'une cellule de données : aRowItem, aDataCol et var aBack, aFore et aBold, après les règles de format.
OnGetCellText	TsgcPivotCellTextEvent	Avant le dessin d'une cellule de données : aRowItem, aDataCol et var aText, la valeur formatée, ou Pivot.EmptyText lorsque la cellule est vide.
OnSelectionChange	TNotifyEvent	SelectedRow ou SelectedCol a changé.
OnColumnResized	TsgcVCLPivotHitEvent	L'utilisateur a redimensionné une colonne en faisant glisser le bord de son en-tête.

Pendant le chargement de la fiche, le moteur se marque seulement comme modifié, et la grille recalcule une fois dans Loaded lorsque Pivot.AutoRecalculate est True.

Exemple

```
uses
  SysUtils, Dialogs,
  sgcPivotGrid_Engine, sgcPivotGrid_VCL;

type
  TPivotEvents = class
  public
    procedure DrillDown(Sender: TObject; aRowItem, aDataCol: Integer;
      var aHandled: Boolean);
    procedure GetCellText(Sender: TObject; aRowItem, aDataCol: Integer;
      var aText: string);
  end;

procedure TPivotEvents.DrillDown(Sender: TObject; aRowItem, aDataCol: Integer;
  var aHandled: Boolean);
var
  oView: TsgcPivotSourceView;
begin
  // the source rows behind the cell, the caller frees the view
  oView := TsgcVCLPivotGrid(Sender).Pivot.DrillDown(aRowItem, aDataCol);
  try
    ShowMessage(Format('%d source rows', [oView.TableRowCount]));
  finally
    oView.Free;
  end;
  aHandled := True;
end;

procedure TPivotEvents.GetCellText(Sender: TObject; aRowItem, aDataCol: Integer;
  var aText: string);
begin
  // negative values in parentheses; an empty cell arrives as EmptyText
  if (Length(aText) > 1) and (aText[1] = '-') then
    aText := '(' + Copy(aText, 2, MaxInt) + ')';
end;

procedure HookEvents(aGrid: TsgcVCLPivotGrid; aEvents: TPivotEvents);
begin
  // empty cells show a dash
  aGrid.Pivot.EmptyText := '-';
```

```
  aGrid.OnDrillDown := aEvents.DrillDown;
  aGrid.OnGetCellText := aEvents.GetCellText;
end;
```

Voir aussi

- TsgcFMXPivotGrid
- TsgcVCLPivotFieldList
- Démarrage rapide VCL

TsgcFMXPivotGrid

Unité : sgcPivotGrid_FMX
TsgcFMXPivotGrid est le tableau croisé FireMonkey, à partir de Delphi XE6 (SGC_PIVOTGRID_FMX). Il a le même moteur, la même mise en page, les mêmes propriétés et les mêmes événements que TsgcVCLPivotGrid. Les différences viennent de FireMonkey : il dessine ses propres barres de défilement, affichées pendant qu'on les fait glisser, défile avec la molette de la souris et avec un geste tactile, et sa méthode de test de position est HitAt, car FireMonkey publie déjà une propriété HitTest. Ses types d'événement ont le préfixe TsgcFMXPivot. Il hérite de TsgcUIFMXControl et publie les propriétés et événements de mise en page FireMonkey habituels.

Propriétés

Propriété	Type	Description
Pivot	TsgcPivotEngine	Le moteur que possède la grille : ses champs, ses données, ses totaux et ses libellés, sérialisés avec la fiche. La grille implémente IsgcUITable en lui déléguant.
RowLayout	TsgcPivotRowLayout	pvlCompact (par défaut) dessine les niveaux de ligne dans une colonne d'en-tête indentée, pvlTabular donne à chaque niveau sa propre colonne. Voir Zones.
RowHeight	Integer	La hauteur d'une ligne en DIP. Par défaut 24.
HeaderHeight	Integer	La hauteur d'une ligne d'en-tête de colonne en DIP. Par défaut 26.
IndentWidth	Integer	L'indentation de chaque niveau de ligne dans la mise en page compacte, en DIP. Par défaut 16.
MinColumnWidth, MaxColumnWidth	Integer	Les limites des largeurs automatiques des colonnes de données, en DIP. Par défaut 48 et 320.
AutoColumnWidths	Boolean	Ajuste les colonnes de données à leur contenu. Si False, elles font 96 DIP de large. Les colonnes d'en-tête de ligne s'ajustent toujours. Par défaut True.
GridLines	Boolean	Dessine les lignes des cellules. Par défaut True.
AlternateRows	Boolean	Grise une ligne sur deux. Par défaut False.
TotalsBold	Boolean	Dessine les totaux en gras. Par défaut True.
ShowExpandButtons	Boolean	Affiche les boutons + et - des membres qui ont des enfants. Par défaut True.
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom.
AccentColor	TsgcUIColor	Une couleur d'accent $AARRGGBB. 0 (par défaut) utilise l'accent du thème.
FontSize	Double	La taille de police des cellules en points. Par défaut 12.
FormatRules	TsgcPivotFormatRules	Les règles de mise en forme conditionnelle des cellules.

MaxFPS	Integer	Le plus grand nombre d'images par seconde que dessine le contrôle.
SelectedRow	Integer	Publique. L'élément de ligne sélectionné, -1 pour aucun.
SelectedCol	Integer	Publique. La colonne de données sélectionnée, -1 lorsque l'en-tête de ligne a le focus ou que rien n'est sélectionné.
TopRow	Integer	Publique. Le premier élément de ligne affiché sous les en-têtes de colonne.
ScrollX	Integer	Publique. Le décalage de défilement horizontal des colonnes de données.
Renderer	TsgcPivotGridRenderer	Publique, en lecture seule. Le moteur de rendu qui dessine la grille.
VertBarVisible, HorzBarVisible	Boolean	Publique, en lecture seule. True lorsque le contenu a besoin de cette barre de défilement.

Méthodes

Méthode	Retourne	Description
HitAt(X, Y)	TsgcPivotHitInfo	L'élément sous un point en coordonnées locales : Part (pvhNone, pvhCorner, pvhColumnHeader, pvhRowHeader, pvhCell, pvhExpandButton, pvhColumnResize), RowItem, DataCol et Level.
ScrollToCell(aRowItem, aDataCol)		Défile jusqu'à ce que la cellule soit visible. -1 laisse cet axe inchangé.
Recalculate		Recalcule le moteur immédiatement.
SaveToPNG, SaveToSVG(aFileName or aStream, aWidth = 0, aHeight = 0)		Toute la vue sous forme d'image, tailles en DIP, 0 correspond à la taille du contenu, jusqu'à 16384 pixels.
SaveToCSV, SaveToJSON, SaveToHTML(aFileName or aStream)		La vue actuelle sous forme de tableau. Voir Export.
SaveLayoutToString, LoadLayoutFromString, SaveLayoutToFile, LoadLayoutFromFile		La mise en page de Pivot au format JSON, avec les largeurs de colonne de la grille. Édition Advanced. Voir Persistance de la mise en page.

Événements

Événement	Type	Description
OnCellClick, OnCellDbl-Click	TsgcFMXPivo-tHitEvent	Un clic ou un double-clic, avec les informations de position du point.
OnDrillDown	TsgcFMXPivot-DrillDownEvent	Entrée ou un double-clic sur une cellule : aRowItem, aDataCol et var aHandled. La grille n'a pas d'action intégrée, appelez Pivot.DrillDown ou Pivot.GetSourceRows pour lire les lignes source.
OnExpand-Change	TsgcFMXPivo-tExpandEvent	Après le développement ou la réduction d'un membre par l'utilisateur : aAxis et aItem.
OnGetCell-Style	TsgcPivotCell-StyleEvent	Avant le dessin d'une cellule de données : aRowItem, aDataCol et var aBack, aFore et aBold, après les règles de format.
OnGetCellText	TsgcPivotCell-TextEvent	Avant le dessin d'une cellule de données : aRowItem, aDataCol et var aText, la valeur formatée, ou Pivot.EmptyText lorsque la cellule est vide.
OnSelection-Change	TNotifyEvent	SelectedRow ou SelectedCol a changé.
OnColumnRe-sized	TsgcFMXPivo-tHitEvent	L'utilisateur a redimensionné une colonne en faisant glisser le bord de son en-tête.

Voir aussi

- TsgcVCLPivotGrid
- TsgcFMXPivotFieldList
- Démarrage rapide FMX

TsgcVCLPivotFieldList

Unité : sgcPivotGrid_FieldList_VCL
TsgcVCLPivotFieldList est la liste de champs VCL. Liée à un tableau croisé, elle permet à l'utilisateur de construire la mise en page par glisser-déposer. Elle hérite de TsgcUIVCLControl et publie les propriétés de mise en page VCL habituelles. Sa liste de membres est un TsgcVCLPivotMemberPopup et son menu d'agrégation un TPopupMenu.

Comment l'utilisateur s'en sert

- Les colonnes source sont listées en haut, avec une coche pour les colonnes présentes dans la mise en page. En dessous se trouvent quatre zones : Filtres, Colonnes, Lignes et Valeurs.
- Faites glisser une colonne dans une zone pour ajouter un champ. Faites glisser une pastille vers une autre zone pour la déplacer, ou hors des zones pour la supprimer.
- Une colonne déposée sur Valeurs reçoit Count si c'est une colonne de texte et Sum sinon. Le bouton d'une pastille de Valeurs ouvre un menu avec toutes les agrégations.
- Le bouton de filtre d'une pastille de Filtres, Colonnes ou Lignes ouvre la liste de cases à cocher des membres, avec un champ de recherche et Select All. OK écrit FilterValues et FilterExclude, voir Filtres.
- Les filtres d'étiquette et de valeur se définissent dans le code ou dans l'Object Inspector.

Propriétés

Propriété	Type	Description
PivotGrid	TComponent	Le tableau croisé dont la liste modifie le moteur, un TsgcVCLPivotGrid ou un TsgcFMXPivotGrid.
Engine	TsgcPivotEngine	Publique. Un moteur défini dans le code a priorité sur le moteur de PivotGrid, pour un moteur sans interface.
ShowFilterArea	Boolean	Affiche la zone Filtres. Par défaut True.
DeferUpdate	Boolean	Rassemble les modifications de l'utilisateur et ne les applique qu'à ApplyChanges, avec un seul recalcul. Par défaut False.
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom.
AccentColor	TsgcUIColor	Un accent $AARRGGBB. 0 (par défaut) utilise l'accent du thème.
FontSize	Double	La taille de la police en points.
MaxFPS	Integer	Le plus grand nombre d'images par seconde que dessine le contrôle.

Méthodes

Méthode	Retourne	Description
ApplyChanges		Affecte la mise en page en attente au moteur lorsque DeferUpdate est True.

Événements

Événement	Type	Description
OnBeforeFieldMove	TsgcPivotBeforeFieldMoveEvent	Avant que l'utilisateur déplace un champ : aFieldName, aFrom, aTo et var aAllow. Pour une colonne source déposée dans une zone, aFrom est égal à aTo. La suppression d'un champ ne déclenche pas d'événement de déplacement.
OnFieldMoved	TsgcPivotFieldMovedEvent	Après l'ajout ou le déplacement d'un champ : aField.

Exemple

```
uses
  SysUtils, Controls, Forms,
  sgcPivotGrid_Types, sgcPivotGrid_VCL, sgcPivotGrid_FieldList_VCL;

type
  TFieldListRules = class
  public
    procedure BeforeFieldMove(Sender: TObject; const aFieldName: string;
      aFrom, aTo: TsgcPivotArea; var aAllow: Boolean);
  end;

procedure TFieldListRules.BeforeFieldMove(Sender: TObject;
  const aFieldName: string; aFrom, aTo: TsgcPivotArea; var aAllow: Boolean);
begin
  // the customer names never go to the columns
  if SameText(aFieldName, 'Customer') and (aTo = pvaColumn) then
    aAllow := False;
end;

function AddFieldList(aForm: TForm; aGrid: TsgcVCLPivotGrid;
  aRules: TFieldListRules): TsgcVCLPivotFieldList;
begin
  Result := TsgcVCLPivotFieldList.Create(aForm);
  Result.Parent := aForm;
  Result.Align := alRight;
  Result.Width := 260;
  Result.PivotGrid := aGrid;
  // collect the changes, the grid recalculates on ApplyChanges
  Result.DeferUpdate := True;
  Result.OnBeforeFieldMove := aRules.BeforeFieldMove;
end;
```

Voir aussi

- TsgcFMXPivotFieldList
- Zones

TsgcFMXPivotFieldList

Unité : sgcPivotGrid_FieldList_FMX
TsgcFMXPivotFieldList est la liste de champs FireMonkey, à partir de Delphi XE6 (SGC_PIVOTGRID_FMX), avec les mêmes zones, le même glisser-déposer et les mêmes règles de filtre que TsgcVCLPivotFieldList. Ses fenêtres contextuelles s'ouvrent lorsque le bouton est relâché sur la même pastille.

Comment l'utilisateur s'en sert

- Les colonnes source sont listées en haut, avec une coche pour les colonnes présentes dans la mise en page. En dessous se trouvent quatre zones : Filtres, Colonnes, Lignes et Valeurs.
- Faites glisser une colonne dans une zone pour ajouter un champ. Faites glisser une pastille vers une autre zone pour la déplacer, ou hors des zones pour la supprimer.
- Une colonne déposée sur Valeurs reçoit Count si c'est une colonne de texte et Sum sinon. Le bouton d'une pastille de Valeurs ouvre un menu avec toutes les agrégations.
- Le bouton de filtre d'une pastille de Filtres, Colonnes ou Lignes ouvre la liste de cases à cocher des membres, avec un champ de recherche et Select All. OK écrit FilterValues et FilterExclude, voir Filtres.
- Les filtres d'étiquette et de valeur se définissent dans le code ou dans l'Object Inspector.

Propriétés

Propriété	Type	Description
PivotGrid	TComponent	Le tableau croisé dont la liste modifie le moteur, un TsgcVCLPivotGrid ou un TsgcFMXPivotGrid.
Engine	TsgcPivotEngine	Publique. Un moteur défini dans le code a priorité sur le moteur de PivotGrid, pour un moteur sans interface.
ShowFilterArea	Boolean	Affiche la zone Filtres. Par défaut True.
DeferUpdate	Boolean	Rassemble les modifications de l'utilisateur et ne les applique qu'à ApplyChanges, avec un seul recalcul. Par défaut False.
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom.
AccentColor	TsgcUIColor	Un accent $AARRGGBB. 0 (par défaut) utilise l'accent du thème.
FontSize	Double	La taille de la police en points.
MaxFPS	Integer	Le plus grand nombre d'images par seconde que dessine le contrôle.
Model	TsgcPivotFieldListModel	Publique, en lecture seule. Le modèle indépendant du framework avec les zones, les pastilles et l'état du glissement.
AggregationMenu, FilterPopup, FilterSearch, FilterSelectAll, FilterList	contrôles de menu et de fenêtre contextuelle	Publique, en lecture seule. Le menu d'agrégation et les éléments de la liste de membres, créés lors de leur premier affichage.

Méthodes

Méthode	Retourne	Description
ApplyChanges		Affecte la mise en page en attente au moteur lorsque DeferUpdate est True, avec un seul recalcul.
HitAt(X, Y)	TsgcPivotFieldListHitInfo	L'élément de la liste situé sous un point local.
ShowAggregationMenu(aField)		Ouvre le menu d'agrégation d'un champ de Valeurs.
ShowMemberFilter(aField)		Ouvre la liste de membres d'un champ de Filtres, Colonnes ou Lignes.
CloseMemberFilter(aApply)		Ferme la liste de membres. aApply écrit les membres cochés dans le filtre.

Événements

Événement	Type	Description
OnBeforeFieldMove	TsgcPivotBeforeFieldMoveEvent	Avant que l'utilisateur déplace un champ : aFieldName, aFrom, aTo et var aAllow. Pour une colonne source déposée dans une zone, aFrom est égal à aTo. La suppression d'un champ ne déclenche pas d'événement de déplacement.
OnFieldMoved	TsgcPivotFieldMovedEvent	Après l'ajout ou le déplacement d'un champ : aField.

Voir aussi

- TsgcVCLPivotFieldList
- TsgcFMXPivotGrid

TsgcPivotEngine

Unité : sgcPivotGrid_Engine
TsgcPivotEngine contient les lignes source dans un magasin de colonnes typé, la mise en page et le résultat calculé. Les grilles en possèdent un, publié sous le nom Pivot, et il fonctionne aussi seul, sans aucune interface. Il descend de TPersistent, implémente IsgcUITable sans comptage de références et se crée avec `Create(aOwner: TPersistent = nil)`. En C++, passez NULL explicitement.

Propriétés

Propriété	Type	Description
Fields	TsgcPivot-Fields	Les champs de la mise en page, dans les quatre zones.
CalculatedFields	TsgcPivot-Calculated-Fields	Les champs calculés. Édition Advanced.
RowGrandTo-tals, Column-GrandTotals	Boolean	Affichent la ligne et la colonne du total général. Par défaut True.
SubtotalPosition	TsgcPivot-SubtotalPo-sition	pvpTop (par défaut) ou pvpBottom. Voir Totaux.
AutoRecalculate	Boolean	Recalcule après chaque modification des données ou de la mise en page, une fois par bloc. Si False, appelez Recalculate. Par défaut True.
GrandTotalCap-tion	string	Le libellé des totaux généraux. Par défaut Grand Total.
BlankCaption	string	Le libellé du membre vide. Par défaut (blank).
EmptyText	string	Le texte d'une cellule de données sans valeur, vide par défaut. Les grilles, le moteur de rendu, l'export HTML et ResultTable l'affichent, et OnGetCellText le reçoit et peut le remplacer. SaveToCSV écrit un champ vide et SaveToJSON null. Une modification republie la vue sans recalcul.
OthersCaption	string	Le libellé du membre Others de Top N. Par défaut Others. Édition Advanced.
MaxCells	Integer	Le plus grand nombre de cellules qu'une mise en page peut créer. Un recalcul qui en demande davantage lève EsgcPivotError. Par défaut 5 000 000.
Result	TsgcPivo-tResult	Publique, en lecture seule. L'instantané actuel, nil avant le premier calcul. Ne le lisez qu'entre Lock et Unlock.
ResultTable	TsgcPivo-tResultTable	Publique, en lecture seule. La vue IsgcUITable du résultat. Voir Export.
Store	TsgcPivotS-tore	Publique, en lecture seule. Le magasin de colonnes, utilisé par les sources.

RowLayout	TsgcPivo-tRowLayout	Publique. Définie par les grilles : dans la mise en page compacte, un élément parent précède ses enfants.

Méthodes : données

Méthode	Re-tourne	Description
BeginLoad(aClear = True), EndLoad		Un bloc de chargement. aClear vide d'abord le magasin. Le moteur recalcule une seule fois, au EndLoad le plus externe.
DefineColumn(aName), DefineColumn(aName, aType)	Integer	Ajoute une colonne. Sans type, la première valeur non vide en décide. Les noms sont uniques, sans distinction de casse.
AppendRow(aValues)	Integer	Ajoute une ligne de valeurs Variant dans l'ordre des colonnes. Null et les valeurs non convertibles sont vides.
ClearData		Supprime toutes les lignes et toutes les colonnes.
LoadFromTable(aTable)		Charge toutes les colonnes et toutes les lignes d'un IsgcUITable.
SourceRowCount, SourceColumn-Count	Integer	La taille du magasin.
SourceColumnName(aIndex)	string	Le nom d'une colonne.
SourceValue(aRow, aCol)	Variant	Une valeur stockée, Null si vide.
GetMemberLabels(aField, aList)		Les libellés de membre auxquels un filtre de membres de aField se compare, dans l'ordre d'affichage, avec BlankCaption en dernier lorsqu'une valeur est vide.

Méthodes : mise en page et vue

Méthode	Retourne	Description
BeginUpdate, EndUpdate		Un bloc de mise en page avec un seul recalcul à EndUpdate.
Recalculate		Recalcule immédiatement.
AddField(aFieldName, aArea, aGrouping = pvdNone)	TsgcPivot-Field	Ajoute un champ de ligne, de colonne, de filtre ou de données à la fin de sa zone.
AddDataField(aFieldName, aAggregation = pvfSum)	TsgcPivot-Field	Ajoute une mesure.
MoveField(aField, aArea, aIndex)		Déplace un champ vers une autre zone ou une autre position.
RemoveField(aField), ClearLayout		Supprime un champ, ou tous les champs.
ExpandAll(aAxis), CollapseAll(aAxis)		Développe ou réduit tous les membres de pvxRows ou pvxColumns.

ExpandLevel(aAxis, aLevel), CollapseLevel(aAxis, aLevel)		Développe ou réduit tous les membres d'un niveau.
SetExpanded(aAxis, aItem, aValue), ToggleExpanded(aAxis, aItem)		Modifie un élément de l'instantané actuel. L'état développé est conservé par chemin de membre, il survit donc à un recalcul. Voir Développer et réduire.
GetExpandState(aAxis, aState), SetExpandState(aAxis, aState)		L'état développé explicite d'un axe sous forme de liste de chaînes. Édition Advanced.

Méthodes : résultat, drill-down et export

Méthode	Retourne	Description
Lock, Unlock		Protègent l'instantané pendant la lecture de Result.
GetSourceRows(aRowItem, aDataCol, var aRows)	Integer	Les lignes source derrière une cellule, dans l'ordre des lignes. aDataCol -1 prend toutes les colonnes, et un total prend toutes les lignes situées sous lui.
DrillDown(aRowItem, aDataCol)	TsgcPivotSourceView	Les mêmes lignes sous forme d'IsgcUITable avec toutes les colonnes du magasin. L'appelant le libère, avant le moteur. Il n'a plus de lignes après une modification des données.
SaveToCSV, SaveToJSON, SaveToHTML		La vue actuelle, vers un fichier ou un flux. Voir Export.
SaveLayoutToString, LoadLayoutFromString, SaveLayoutToFile, LoadLayoutFromFile		La mise en page au format JSON. Édition Advanced. Voir Persistance de la mise en page.
Assign(Source)		Copie la mise en page et les réglages d'un autre moteur, pas ses données.

Événements

Événement	Type	Description
OnCalculated	TNotifyEvent	Après la publication d'un nouveau résultat, dans le thread qui a recalculé.
OnLayoutChange	TNotifyEvent	Après la modification des champs ou des réglages de mise en page.
OnFilterRow	TsgcPivotFilterRowEvent	Pour chaque ligne source tant qu'il est assigné : aRow et var aAccept. Voir Filtres.

Exemple

Lecture du résultat : les libellés de lignes et les textes des cellules, entre Lock et Unlock.

```
uses
  SysUtils, Classes,
  sgcPivotGrid_Types, sgcPivotGrid_Result, sgcPivotGrid_Engine;

procedure ListRows(aEngine: TsgcPivotEngine; aLines: TStrings);
```

```
var
  oResult: TsgcPivotResult;
  oItem: TsgcPivotItem;
  i, j: Integer;
  vLine: string;
begin
  aEngine.Lock;
  try
    oResult := aEngine.Result;
    if oResult = nil then
      Exit; // not calculated yet
    for i := 0 to oResult.ItemCount(pvxRows) - 1 do
    begin
      oItem := oResult.Item(pvxRows, i);
      if oItem.Kind = pvkGrandTotal then
        vLine := aEngine.GrandTotalCaption
      else
        vLine := StringOfChar(' ', oItem.Level * 2) +
          oResult.NodeLabel(pvxRows, oItem.Node);
      for j := 0 to oResult.DataColCount - 1 do
        vLine := vLine + #9 + oResult.CellText(i, j);
      aLines.Add(vLine);
    end;
  finally
    aEngine.Unlock;
  end;
end;
```

Voir aussi

- TsgcPivotField
- Démarrage rapide sans interface
- Architecture

TsgcPivotField

Unité : sgcPivotGrid_Fields
TsgcPivotField est un élément de TsgcPivotEngine.Fields (TsgcPivotFields), un champ de la mise en page. Le même FieldName peut apparaître dans plusieurs champs.

Propriétés

Propriété	Type	Description
FieldName	string	La colonne source ou le champ calculé.
Caption	string	Le libellé du niveau ou de la mesure. Vide, il utilise DisplayCaption.
DisplayCaption	string	Publique, en lecture seule. Caption, sinon Sum of Sales pour un champ de données, sinon FieldName.
Area	TsgcPivotArea	pvaRow (par défaut), pvaColumn, pvaData ou pvaFilter. Voir Zones.
AreaIndex	Integer	La position dans la zone. Pour un champ de données, c'est l'index de la mesure.
DateGrouping	TsgcPivotDateGrouping	pvdNone (par défaut), pvdYear, pvdQuarter, pvdMonth, pvdWeek ou pvdDay. Voir Regroupement des dates.
DisplayFormat	string	Un masque FormatDateTime ou FormatFloat pour les libellés de membre, ou pour les valeurs d'un champ de données.
Aggregation	TsgcPivotAggregation	La fonction d'un champ de données, pvfSum par défaut. Voir Agrégations.
SortBy	TsgcPivotSortBy	pvsLabel (par défaut) trie les membres par libellé, les nombres et les dates par valeur, pvsValue par leur total.
SortOrder	TsgcPivotSortOrder	pvoAscending (par défaut) ou pvoDescending.
SortMeasure	Integer	Le champ de données, par AreaIndex, d'un tri par valeur et de Top N. Par défaut 0.
Subtotals	Boolean	Affiche les sous-totaux des membres. Par défaut True.
Expanded	Boolean	L'état développé initial des membres de ce niveau. Par défaut True.
FilterValues	TStrings	Le filtre de membres. Voir Filtres.
FilterExclude	Boolean	FilterValues liste les membres à exclure. Par défaut False.
LabelFilter	TsgcPivotLabelFilter	Operator, Value et Value2 sur les libellés de membre.
ValueFilter	TsgcPivotValueFilter	Operator, Value, Value2 et Measure sur les totaux de membre.
TopN, TopNOthers	Integer, Boolean	Top N avec un membre Others facultatif. Édition Advanced. Voir Top N.

ShowAs, ShowAsAxis, ShowAsBaseItem	TsgcPivotShowAs, TsgcPivotAxisKind, string	Afficher les valeurs en tant que. Édition Advanced. Voir Afficher les valeurs en tant que.

Méthodes

Méthode	Retourne	Description
MoveTo(aArea, aAreaIndex)		Déplace le champ avec une seule notification de changement. MaxInt est la fin de la zone.
Assign(Source)		Copie toutes les propriétés, y compris Area et AreaIndex.

TsgcPivotFields

Méthode	Retourne	Description
Add	TsgcPivotField	Ajoute un champ.
AreaCount(aArea)	Integer	Le nombre de champs d'une zone.
FieldAt(aArea, aAreaIndex)	TsgcPivotField	Le champ à une position d'une zone.
Items[aIndex]	TsgcPivotField	Les champs dans l'ordre de la collection.

Dans BeginUpdate et EndUpdate, et pendant le chargement d'une fiche, les valeurs d'AreaIndex sont conservées telles quelles. EndUpdate renumérote chaque zone par AreaIndex, puis par ordre de collection.

Voir aussi

- TsgcPivotEngine
- TsgcPivotCalculatedField

TsgcPivotCalculatedField

Unité : sgcPivotGrid_Fields
TsgcPivotCalculatedField est un élément de TsgcPivotEngine.CalculatedFields (TsgcPivotCalculatedFields) : une colonne Double calculée pour chaque ligne source à partir d'une expression, que les champs utilisent avec FieldName = Name comme une colonne source. Édition Advanced, voir Champs calculés pour la syntaxe.

Propriétés

Propriété	Type	Description
Name	string	Le nom auquel les champs font référence. Il ne doit pas être le nom d'une colonne source.
Expression	string	L'expression, par exemple [Revenue] - [Cost].

TsgcPivotCalculatedFields

Méthode	Retourne	Description
Add	TsgcPivotCalculatedField	Ajoute un champ calculé.
IndexOfName(aName)	Integer	L'index d'un champ calculé, -1 s'il n'y en a pas.
Items[aIndex]	TsgcPivotCalculatedField	Les champs calculés.

Voir aussi

- Champs calculés
- TsgcPivotField

TsgcPivotSource_DataSet

Unité : sgcPivotGrid_Source_DataSet
TsgcPivotSource_DataSet est un composant non visuel qui copie un TDataSet dans un moteur de tableau croisé par l'intermédiaire d'un TDataLink, et le recharge lorsque les données changent. Voir Démarrage rapide DataSet.

Propriétés

Propriété	Type	Description
DataSet	TDataSet	Le jeu de données à charger.
Active	Boolean	Charge les données immédiatement et les maintient à jour. Par défaut False.
Colonnes	TStrings	Les colonnes à charger. Vide, il charge toutes les colonnes.
LiveUp-dates	Boolean	Recharge lorsque les données changent pendant Active. Par défaut True.
PivotGrid	TCom-ponent	Le tableau croisé dont le moteur reçoit les données.
Engine	TsgcPivo-tEngine	Publique. Le moteur dans lequel la source écrit : un moteur affecté dans le code, sinon le moteur de PivotGrid.

Méthodes

Méthode	Retourne	Description
Load		Charge le jeu de données immédiatement, Active ou non.
Refresh		Recharge lorsque Active.
FieldTypeToDataType(aFieldType)	TsgcPivot-DataType	Fonction de classe. Le type de colonne d'un type de champ DB.

- Le chargement utilise DisableControls et restaure le signet. Les champs BLOB, binaires et imbriqués sont ignorés.
- Avec LiveUpdates, le tableau croisé se recharge lorsque le jeu de données signale une modification des données en état de navigation : un post, un append, un delete, un refresh, First ou Last. Next, Prior et Move-By ne rechargent pas.

Voir aussi

- TsgcPivotSource_Table
- Démarrage rapide DataSet

TsgcPivotSource_Table

Unité : sgcPivotGrid_Source_Table
TsgcPivotSource_Table copie n'importe quel IsgcUITable dans un moteur de tableau croisé : un TsgcGridData, un TsgcVCLGrid ou TsgcFMXGrid de sgcGrid, le résultat d'un autre tableau croisé, ou votre propre classe. Il écoute la table et la recharge, sous TableLock, lorsque TableVersion a changé. Le rechargement s'exécute dans le thread qui notifie la modification.

Propriétés

Propriété	Type	Description
Table	TComponent	Un composant qui implémente IsgcUITable.
Active	Boolean	Charge les données immédiatement et les maintient à jour. Par défaut False.
Colonnes	TStrings	Les colonnes à charger. Vide, il charge toutes les colonnes.
LiveUp-dates	Boolean	Recharge lorsque les données changent pendant Active. Par défaut True.
PivotGrid	TComponent	Le tableau croisé dont le moteur reçoit les données.
Engine	TsgcPivotEn-gine	Publique. Le moteur dans lequel la source écrit : un moteur affecté dans le code, sinon le moteur de PivotGrid.

Méthodes

Méthode	Retourne	Description
SetTable(aTable)		Lie tout objet qui implémente IsgcUITable. nil supprime la liaison.
GetTable	IsgcUITable	La table liée.
Load		Charge la table immédiatement, Active ou non, même si sa version n'a pas changé.
Refresh		Recharge lorsque Active.

Les colonnes de même nom qu'une colonne précédente sont ignorées.

Exemple

```
uses
  Classes,
  sgcUICore_Table, sgcPivotGrid_VCL, sgcPivotGrid_Source_Table;

procedure BindTable(aPivot: TsgcVCLPivotGrid; aTable: TComponent);
var
  oSource: TsgcPivotSource_Table;
begin
  // aTable is any component that implements IsgcUITable,
  // for example a TsgcVCLGrid of sgcGrid
  oSource := TsgcPivotSource_Table.Create(aPivot.Owner);
```

```
  oSource.Table := aTable;
  oSource.PivotGrid := aPivot;
  oSource.Active := True;
end;

procedure BindTableObject(aPivot: TsgcVCLPivotGrid; const aTable: IsgcUITable;
  aSource: TsgcPivotSource_Table);
begin
  // an object that is not a component binds through SetTable
  aSource.SetTable(aTable);
  aSource.PivotGrid := aPivot;
  aSource.Active := True;
end;
```

Voir aussi

- TsgcPivotSource_DataSet
- Export

TsgcPivotGridRenderer

Unité : sgcPivotGrid_Render
TsgcPivotGridRenderer dessine le résultat d'un tableau croisé sur n'importe quel canevas sgcUICore, et c'est lui qu'utilisent les grilles VCL et FMX. Seul, il produit la sortie PNG et SVG sans interface. Créez-le avec `Create(aOptions: TsgcPivotRenderOptions = nil; aFormatRulesOwner: TPersistent = nil)`. En C++, passez NULL pour les deux. Voir Démarrage rapide sans interface.

Propriétés

Propriété	Type	Description
Options	TsgcPivotRenderOptions	RowLayout, RowHeight, HeaderHeight, IndentWidth, MinColumnWidth, MaxColumnWidth, AutoColumnWidths, GridLines, AlternateRows, TotalsBold, ShowExpandButtons, Theme, AccentColor et FontSize, avec la même signification et les mêmes valeurs par défaut que pour TsgcVCLPivotGrid.
FormatRules	TsgcPivotFormatRules	Les règles de mise en forme conditionnelle.
GrandTotalCaption, TotalCaption	string	Les libellés des totaux généraux et des sous-totaux du bas. Les surcharges moteur de SaveToPNG et SaveToSVG copient GrandTotalCaption depuis le moteur.
CurrentResult	TsgcPivotResult	En lecture seule. Le résultat du dernier Render.

Méthodes

Méthode	Retourne	Description
Render(aResult, aState, aCanvas, aWidth, aHeight, aScale)		Dessine la partie visible d'un résultat, aWidth et aHeight en pixels de l'appareil, aScale pixels de l'appareil par DIP.
SaveToPNG, SaveToSVG(aResult or aEngine, aState, aFileName or aStream, aWidth = 0, aHeight = 0, aScale = 1)		Tout le contenu sous forme d'image, 0 correspond à la taille du contenu. Les surcharges moteur verrouillent le moteur pendant le dessin.
HitTest(aResult, aState, aX, aY)	TsgcPivotHitInfo	L'élément sous un point.
ContentWidth, ContentHeight	Integer	La taille de tout le contenu en DIP.
FrozenWidth, FrozenHeight	Integer	La taille des en-têtes de ligne et des en-têtes de colonne.
VisibleRange	Boolean	Les premiers et derniers éléments de ligne et colonnes de données visibles d'une zone d'affichage.

ColumnKey(aResult, aDataCol)	string	La clé d'une colonne de données, utilisée par les remplacements de largeur de colonne.

Événements

Événement	Type	Description
OnGetCell-Style	TsgcPivotCellSty-leEvent	Les couleurs et l'état gras d'une cellule de données, après les règles de format.
OnGetCellText	TsgcPivotCellTex-tEvent	Le texte d'une cellule de données : la valeur formatée, ou l'EmptyText du résultat lorsque la cellule est vide.

Fonctions utilitaires

Routine	Description
sgcPivotRenderState	Un TsgcPivotRenderState vide : ni défilement, ni sélection, ni survol, ni remplacements de largeur de colonne.
sgcPivotSetColumnWidth(var aState, aKey, aWidth)	Définit la largeur en DIP de la colonne de données dont le ColumnKey est aKey.

Voir aussi

- Démarrage rapide sans interface
- TsgcPivotFormatRule
- Export

TsgcPivotFormatRule

Unité : sgcPivotGrid_Format
TsgcPivotFormatRule est un élément de FormatRules (TsgcPivotFormatRules) des grilles et du moteur de rendu : une règle de valeur qui colore les cellules dont la valeur remplit une condition, ou une échelle de couleurs entre deux couleurs.

Propriétés

Propriété	Type	Description
Kind	TsgcPivotFormatRuleKind	pvrValue (par défaut), une règle de valeur, ou pvrColorScale.
Measure	Integer	Le champ de données, par AreaIndex, auquel s'applique la règle. -1 (par défaut) s'applique à toutes les mesures.
Operator, Value, Value2	TsgcPivotFilterOperator, Double, Double	La condition, avec les opérateurs des filtres. pvcNone correspond à toute valeur non vide. Une échelle de couleurs l'utilise aussi pour sélectionner ses valeurs.
BackColor, ForeColor	TsgcUIColor	Les couleurs $AARRGGBB d'une cellule correspondante. 0 (par défaut) conserve la couleur. Une échelle de couleurs n'utilise que ForeColor.
Bold	Boolean	Dessine en gras une cellule correspondante. Par défaut False.
MinColor, MaxColor	TsgcUIColor	Les couleurs d'une échelle à la plus petite et à la plus grande valeur des cellules de membre de la mesure. Un MinColor transparent se fond dans MaxColor.
Scope	TsgcPivotFormatScopes	Les cellules auxquelles s'applique la règle : pveCells (par défaut, cellules de membre), pveTotals et pveGrandTotals.
Enabled	Boolean	Par défaut True.
StopOnMatch	Boolean	Une correspondance met fin à l'évaluation des règles pour cette cellule. Par défaut False.

Évaluation

- Les règles s'exécutent dans l'ordre de la collection. Chaque règle active correspondante applique ses couleurs différentes de 0, de sorte qu'une correspondance ultérieure remplace une précédente, et que Bold s'accumule.
- Une cellule vide ne correspond à aucune règle et n'est jamais colorée.
- La plage d'une échelle de couleurs est prise sur les cellules de membre, de sorte que les totaux situés en dehors reçoivent les couleurs extrêmes.
- OnGetCellStyle s'exécute après les règles et peut modifier le résultat.

Exemple

```
uses
  sgcPivotGrid_Types, sgcPivotGrid_Format, sgcPivotGrid_VCL;

procedure AddRules(aGrid: TsgcVCLPivotGrid);
```

```
var
  oRule: TsgcPivotFormatRule;
begin
  // a colour scale over the member cells of the first measure
  oRule := aGrid.FormatRules.Add;
  oRule.Kind := pvrColorScale;
  oRule.Measure := 0;
  oRule.MinColor := $FFFFFFFF;
  oRule.MaxColor := $FF2E7D32;

  // negative values in red and bold, on the cells and the totals
  oRule := aGrid.FormatRules.Add;
  oRule.Kind := pvrValue;
  oRule.Operator := pvcLess;
  oRule.Value := 0;
  oRule.ForeColor := $FFC62828;
  oRule.Bold := True;
  oRule.Scope := [pveCells, pveTotals];
end;
```

Voir aussi

- TsgcVCLPivotGrid
- TsgcPivotGridRenderer

Zones

Chaque champ de la mise en page se trouve dans l'une des quatre zones. La propriété Area de TsgcPivotField et le deuxième argument d'AddField la choisissent.

Area	Signification
pvaRow	Les membres deviennent les en-têtes de ligne. Plusieurs champs de ligne s'imbriquent, le premier est le niveau le plus externe.
pvaCo-lumn	Les membres deviennent les en-têtes de colonne, imbriqués de la même manière.
pvaData	Une mesure : les valeurs agrégées dans chaque cellule. Plusieurs champs de données donnent plusieurs colonnes de mesure sous chaque membre de colonne.
pvaFil-ter	Un filtre de rapport. Ses membres ne sont pas affichés, il restreint seulement les lignes source par ses filtres de membre et d'étiquette.

Ordre et déplacement

- AreaIndex ordonne les champs d'une zone. AddField, AddDataField et l'affectation de Area placent un champ à la fin de sa nouvelle zone.
- MoveField(aField, aArea, aIndex) déplace un champ vers une autre zone ou position avec une seule notification. RemoveField le supprime et ClearLayout retire tous les champs.
- Fields.AreaCount(aArea) et Fields.FieldAt(aArea, aIndex) lisent la mise en page zone par zone.
- Le même FieldName peut apparaître dans plusieurs champs, par exemple une colonne Date regroupée par année et par mois, ou une colonne Sales avec Sum et Average.
- L'index de mesure utilisé par SortMeasure, ValueFilter.Measure et les règles de format est l'AreaIndex du champ de données.

Noms inconnus

Un champ de ligne, de colonne ou de filtre dont le FieldName n'est ni une colonne source ni un champ calculé est ignoré. Un champ de données dont le nom est inconnu conserve une mesure vide, de sorte que les index de mesure ne se décalent pas. Cela permet de définir une mise en page avant le chargement des données.

Lignes compactes et tabulaires

Le RowLayout des grilles décide de la façon dont les niveaux de ligne sont dessinés. pvlCompact (la valeur par défaut) affiche chaque niveau dans une colonne d'en-tête de ligne indentée, chaque parent occupant sa propre ligne. pvlTabular donne à chaque niveau de ligne sa propre colonne d'en-tête.

Exemple

```
uses
  sgcPivotGrid_Types, sgcPivotGrid_Fields, sgcPivotGrid_Engine;

procedure ArrangeFields(aEngine: TsgcPivotEngine);
var
  oRegion, oYear: TsgcPivotField;
begin
  aEngine.BeginUpdate;
  try
    aEngine.ClearLayout;
```

```
    aEngine.AddField('Country', pvaFilter);
    oRegion := aEngine.AddField('Region', pvaRow);
    aEngine.AddField('Product', pvaRow);
    oYear := aEngine.AddField('Date', pvaColumn, pvdYear);
    aEngine.AddDataField('Sales', pvfSum);

    // swap the axes: the regions go to the columns, the years to the rows
    aEngine.MoveField(oRegion, pvaColumn, 0);
    aEngine.MoveField(oYear, pvaRow, 0);
  finally
    aEngine.EndUpdate;
  end;
end;
```

Chaque modification de la mise en page déclenche OnLayoutChange du moteur. Dans BeginUpdate et EndUpdate, le moteur recalcule une seule fois, à EndUpdate.

Voir aussi

- Agrégations
- TsgcPivotEngine
- TsgcVCLPivotFieldList

Agrégations

Un champ de données agrège sa colonne source avec la fonction de Aggregation. Les valeurs vides sont ignorées par toutes les fonctions.

Aggregation	Result
pvfSum	La somme des valeurs. La valeur par défaut.
pvfCount	Le nombre de valeurs non vides, comme dans Excel. Fonctionne avec tous les types de colonne.
pvfAverage	Sum divisé par Count.
pvfMin, pvfMax	La plus petite et la plus grande valeur.
pvfCountDistinct	Le nombre de valeurs différentes non vides. Sur une colonne de texte, il compte les libellés différents.
pvfStdDev, pvfVar	L'écart type et la variance d'échantillon (`n - 1`). Une cellule de moins de deux valeurs est vide.

Les totaux sont exacts

Les sous-totaux et les totaux généraux sont calculés à partir des valeurs source, pas à partir des cellules situées en dessous. La moyenne d'une région est la moyenne de ses lignes, pas la moyenne des moyennes de ses produits, et un comptage distinct d'un total compte chaque valeur une fois.

Libellés et formats

- Sans Caption, un champ de données s'affiche sous la forme *<agrégation> of <FieldName>*, par exemple Sum of Sales. DisplayCaption retourne le libellé utilisé.
- DisplayFormat est un masque FormatFloat pour les valeurs de la mesure, par exemple `#,##0.00`.
- Les mesures sont toujours dans les colonnes. Avec plusieurs champs de données, les colonnes de mesure forment le niveau de colonne le plus interne.

La liste de champs ajoute Count lorsque l'utilisateur dépose une colonne de texte sur Valeurs et Sum sinon, et ses pastilles de Valeurs ouvrent un menu avec toutes les agrégations.

Exemple

```
uses
  sgcPivotGrid_Types, sgcPivotGrid_Fields, sgcPivotGrid_Engine;

procedure AddMeasures(aEngine: TsgcPivotEngine);
var
  oField: TsgcPivotField;
begin
  aEngine.BeginUpdate;
  try
    aEngine.AddField('Region', pvaRow);

    oField := aEngine.AddDataField('Sales', pvfSum);
    oField.DisplayFormat := '#,##0.00';

    // the same column a second time, with another aggregation
    oField := aEngine.AddDataField('Sales', pvfAverage);
```

```
    oField.Caption := 'Average ticket';
    oField.DisplayFormat := '0.00';

    aEngine.AddDataField('Customer', pvfCountDistinct);
    aEngine.AddDataField('OrderNo', pvfCount);
  finally
    aEngine.EndUpdate;
  end;
end;
```

Voir aussi

- Afficher les valeurs en tant que
- Totaux
- TsgcPivotField

Regroupement des dates

Un champ de ligne, de colonne ou de filtre sur une colonne de dates regroupe les dates avec DateGrouping, le troisième argument d'AddField.

DateGrouping	Libellé du membre	Libellé sous un niveau plus grossier du même champ
pvdNone	Chaque valeur différente de date et d'heure est son propre membre.	
pvdYear	2024	
pvdQuarter	2024 Q1	Q1
pvdMonth	2024-01	Jan, le nom court du mois
pvdWeek	2024-W05, la semaine ISO 8601 et son année	W05
pvdDay	La date courte	

- Lorsque le niveau précédent du même axe est la même colonne avec un regroupement plus grossier, le libellé omet l'année, comme le montre la dernière colonne. Year, Quarter et Month d'une même colonne Date donnent 2024, Q1, Jan.
- Avec un DisplayFormat, le libellé est le début du groupe formaté avec FormatDateTime et ce masque.
- Les membres de date sont triés par date, pas par libellé, de sorte que 2024-10 vient après 2024-09.
- Une ligne dont la date est vide appartient au membre vide, affiché avec BlankCaption et trié en dernier.

Exemple

```
uses
  sgcPivotGrid_Types, sgcPivotGrid_Engine;

procedure GroupDates(aEngine: TsgcPivotEngine);
begin
  aEngine.BeginUpdate;
  try
    aEngine.AddField('Region', pvaRow);

    // the same Date column three times, from the coarsest grouping down;
    // the quarters start collapsed, so the months show on demand
    aEngine.AddField('Date', pvaColumn, pvdYear);
    aEngine.AddField('Date', pvaColumn, pvdQuarter).Expanded := False;
    aEngine.AddField('Date', pvaColumn, pvdMonth);

    aEngine.AddDataField('Sales', pvfSum);
  finally
    aEngine.EndUpdate;
  end;
end;
```

Expanded est l'état initial d'un niveau : ici, chaque trimestre démarre réduit et l'utilisateur ouvre les mois d'un trimestre avec son bouton +.

Voir aussi

- Zones
- Totaux

Totaux

Un tableau croisé affiche un sous-total pour chaque membre qui a des enfants et un total général pour chaque axe.

Propriété	Effet
TsgcPivotEngine.RowGrandTotals	Affiche la ligne du total général. Par défaut True.
TsgcPivotEngine.ColumnGrandTotals	Affiche la colonne du total général. Par défaut True.
TsgcPivotEngine.SubtotalPosition	pvpTop (par défaut) : le sous-total est sur le membre lui-même, avant ses enfants, et dans une mise en page compacte c'est la ligne du parent qui le porte. pvpBottom : une ligne ou une colonne de total suit les enfants, avec le libellé *<membre> Total*.
TsgcPivotField.Subtotals	False masque les sous-totaux des membres de ce champ. Par défaut True.
TsgcPivotEngine.GrandTotalCaption	Le libellé des totaux généraux, Grand Total par défaut.
TsgcPivotEngine.BlankCaption	Le libellé du membre des valeurs vides, (blank) par défaut.

- Les totaux sont calculés à partir des lignes source avec des fusions exactes, de sorte qu'une moyenne ou un comptage distinct d'un total est correct. Voir Agrégations.
- Après un filtre de valeur ou Top N, les totaux n'incluent que les membres qui restent visibles, comme dans Excel.
- Les cellules du total général sont toujours calculées, même lorsque la ligne ou la colonne du total général est masquée, de sorte qu'afficher les valeurs en tant que dispose toujours de ses valeurs de base.
- Le moteur de rendu et les grilles dessinent les totaux en gras tant que TotalsBold est True, et les règles de format les sélectionnent avec les portées pveTotals et pveGrandTotals.

Exemple

```
uses
  sgcPivotGrid_Types, sgcPivotGrid_Fields, sgcPivotGrid_Engine;

procedure ConfigureTotals(aEngine: TsgcPivotEngine);
var
  oRegion: TsgcPivotField;
begin
  aEngine.BeginUpdate;
  try
    oRegion := aEngine.AddField('Region', pvaRow);
    aEngine.AddField('Product', pvaRow);
    aEngine.AddField('Channel', pvaColumn);
    aEngine.AddDataField('Sales', pvfSum);

    // subtotals under their members, a grand total row, no grand total column
    aEngine.SubtotalPosition := pvpBottom;
    aEngine.RowGrandTotals := True;
    aEngine.ColumnGrandTotals := False;
    aEngine.GrandTotalCaption := 'All regions';
    aEngine.BlankCaption := '(no value)';

    // the regions show no subtotals, only their products
    oRegion.Subtotals := False;
  finally
    aEngine.EndUpdate;
```

```
  end;
end;
```

Voir aussi

- Filtres
- TsgcPivotEngine

Filtres

Quatre filtres réduisent ce qu'affiche un tableau croisé. Les trois premiers sont des propriétés d'un champ, le quatrième est un événement du moteur.

Filtre	S'applique à	Fonctionnement
Filtre de membres : FilterValues, FilterExclude	Champs de ligne, de colonne et de filtre	FilterValues liste des libellés de membre, comparés en respectant la casse, tels qu'ils sont affichés (après DateGrouping et DisplayFormat). Les lignes source doivent en avoir un, ou aucun avec FilterExclude. Vide, il conserve toutes les lignes.
LabelFilter	Champs de ligne, de colonne et de filtre	Operator, Value et Value2 comparés au libellé du membre, sans distinction de casse. Lorsque les deux côtés sont des nombres, ils sont comparés comme des nombres.
ValueFilter	Champs de ligne et de colonne	Operator, Value et Value2 comparés au total de chaque membre pour le champ de données dont l'AreaIndex est Measure. Il s'exécute niveau par niveau, au sein de chaque parent. Un total vide ne passe qu'avec pvcNone.
OnFilterRow	Lignes source	Déclenché pour chaque ligne source tant qu'il est assigné. Mettez aAccept à False pour exclure la ligne.

Les filtres de membre et d'étiquette suppriment des lignes source, ils modifient donc tous les totaux. Le filtre de valeur supprime des membres après l'agrégation, et les totaux sont recalculés sur les membres restants.

Opérateurs

Opérateur	Correspond lorsque
pvcNone	Toujours : pas de filtre.
pvcEqual, pvcNotEqual	La valeur est égale à Value, ou ne l'est pas.
pvcGreater, pvcGreaterEqual, pvcLess, pvcLessEqual	La valeur se compare à Value.
pvcBetween, pvcNotBetween	La valeur est comprise entre Value et Value2, bornes incluses, dans un sens ou dans l'autre, ou ne l'est pas.
pvcContains, pvcNotContains, pvcBeginsWith, pvcEndsWith	Tests de texte sur le libellé (LabelFilter) ou sur le texte invariant de la valeur.

Listes de membres

TsgcPivotEngine.GetMemberLabels(aField, aList) remplit une liste avec les libellés auxquels FilterValues est comparé, dans l'ordre d'affichage. La fenêtre contextuelle des membres des listes de champs l'utilise : OK enregistre la liste la plus courte, les membres cochés, ou les membres décochés avec FilterExclude, et cocher tous les membres efface le filtre.

Exemple

```
uses
  SysUtils, Variants,
  sgcPivotGrid_Types, sgcPivotGrid_Fields, sgcPivotGrid_Engine;

type
  TOrderFilter = class
  private
    FQuantityCol: Integer;
  public
    procedure Install(aEngine: TsgcPivotEngine);
    procedure FilterRow(Sender: TObject; aRow: Integer; var aAccept: Boolean);
  end;

procedure TOrderFilter.Install(aEngine: TsgcPivotEngine);
var
  i: Integer;
begin
  FQuantityCol := -1;
  for i := 0 to aEngine.SourceColumnCount - 1 do
    if SameText(aEngine.SourceColumnName(i), 'Quantity') then
      FQuantityCol := i;
  aEngine.OnFilterRow := FilterRow;
  aEngine.Recalculate;
end;

procedure TOrderFilter.FilterRow(Sender: TObject; aRow: Integer;
  var aAccept: Boolean);
var
  vQuantity: Variant;
begin
  // a rule per source row: only the orders with a quantity above zero
  if FQuantityCol < 0 then
    Exit;
  vQuantity := TsgcPivotEngine(Sender).SourceValue(aRow, FQuantityCol);
  aAccept := not VarIsNull(vQuantity) and (vQuantity > 0);
end;

procedure ApplyFilters(aEngine: TsgcPivotEngine);
var
  oField: TsgcPivotField;
begin
  aEngine.BeginUpdate;
  try
    // a report filter: only two countries
    oField := aEngine.AddField('Country', pvaFilter);
    oField.FilterValues.Add('Spain');
    oField.FilterValues.Add('France');

    // every product except the samples
    oField := aEngine.AddField('Product', pvaRow);
    oField.FilterValues.Add('Sample');
    oField.FilterExclude := True;

    // a label filter: the customers whose name begins with A
    oField := aEngine.AddField('Customer', pvaRow);
    oField.LabelFilter.Operator := pvcBeginsWith;
    oField.LabelFilter.Value := 'A';

    // a value filter: the channels that sold more than 10,000
    oField := aEngine.AddField('Channel', pvaColumn);
    oField.ValueFilter.Operator := pvcGreater;
    oField.ValueFilter.Value := 10000;
    oField.ValueFilter.Measure := 0;

    aEngine.AddDataField('Sales', pvfSum);
```

```
  finally
    aEngine.EndUpdate;
  end;
end;
```

Voir aussi

- Top N
- Totaux
- TsgcPivotField

Top N

TopN conserve, au sein de chaque parent, les N membres d'un champ de ligne ou de colonne ayant le plus grand total. C'est une fonction avancée, qui ne fait pas partie de l'édition Community.

Propriété	Effet
TsgcPivotField.TopN	Le nombre de membres à conserver. 0 (par défaut) conserve tous les membres.
TsgcPivotField.SortMeasure	Le champ de données, par son AreaIndex, dont les totaux classent les membres. C'est aussi la mesure d'un tri par valeur.
TsgcPivotField.TopNOthers	True ajoute un membre Others avec le total des membres exclus. Il vient toujours en dernier.
TsgcPivotEngine.OthersCaption	Le libellé du membre Others, Others par défaut.

- Top N décide quels membres restent. SortBy et SortOrder décident de leur ordre, de sorte que pvsValue avec pvoDescending les liste du plus grand au plus petit.
- Les totaux des parents et les totaux généraux n'incluent que les membres conservés, plus Others lorsqu'il est affiché.
- Un drill-down sur Others retourne les lignes source de tous les membres qu'il contient.
- Un filtre de valeur ne crée jamais de membre Others.

Exemple

```
uses
  sgcPivotGrid_Types, sgcPivotGrid_Fields, sgcPivotGrid_Engine;

procedure TopProducts(aEngine: TsgcPivotEngine);
var
  oProduct: TsgcPivotField;
begin
  aEngine.BeginUpdate;
  try
    oProduct := aEngine.AddField('Product', pvaRow);
    aEngine.AddField('Date', pvaColumn, pvdYear);
    aEngine.AddDataField('Revenue', pvfSum);

    // the 5 products with the largest revenue, the rest in one Others row
    oProduct.TopN := 5;
    oProduct.TopNOthers := True;
    oProduct.SortBy := pvsValue;
    oProduct.SortOrder := pvoDescending;
    oProduct.SortMeasure := 0;
    aEngine.OthersCaption := 'Other products';
  finally
    aEngine.EndUpdate;
  end;
end;
```

Voir aussi

- Filtres
- Éditions

Afficher les valeurs en tant que

ShowAs modifie la façon dont les valeurs d'un champ de données sont affichées, sans modifier les valeurs elles-mêmes : RawValue, les totaux, le tri, les filtres et le drill-down continuent d'utiliser les vrais nombres. C'est une fonction avancée, qui ne fait pas partie de l'édition Community.

ShowAs	Valeur affichée
pvwValue	La valeur elle-même. La valeur par défaut.
pvwPercentOfRow	La valeur divisée par le total de sa ligne, en pourcentage.
pvwPercentOfColumn	La valeur divisée par le total de sa colonne, en pourcentage.
pvwPercentOfGrandTotal	La valeur divisée par le total général, en pourcentage.
pvwRunningTotal	Le long de ShowAsAxis, la valeur plus les valeurs des membres précédents ayant le même parent, dans l'ordre d'affichage.
pvwDifferenceFrom	La valeur moins la valeur du membre de base ayant le même parent : le membre précédent lorsque ShowAsBaseItem est vide, sinon le membre dont le libellé est ShowAsBaseItem (sans distinction de casse).
pvwRank	Le rang parmi les membres ayant le même parent, le plus grand est 1. Les ex aequo partagent un rang et le rang suivant est sauté : 1, 2, 2, 4.

- Un mode pourcentage affiche 25 pour 25 pour cent, formaté en `0.00%`. Une valeur vide, une base absente et une base de 0 donnent une cellule vide.
- ShowAsAxis (pvxColumns par défaut) est l'axe le long duquel sont pris le cumul, la différence et le rang.
- Le premier membre d'un groupe n'a rien à comparer dans pvwDifferenceFrom, et le membre de base lui-même est vide, comme dans Excel. Le total général de ShowAsAxis n'a pas de frères, il reste donc vide dans les trois derniers modes.
- La même colonne source peut être ajoutée deux fois, une fois comme valeur et une fois avec ShowAs, pour afficher les deux.

Exemple

```
uses
  sgcPivotGrid_Types, sgcPivotGrid_Fields, sgcPivotGrid_Engine;

procedure CompareYears(aEngine: TsgcPivotEngine);
var
  oField: TsgcPivotField;
begin
  aEngine.BeginUpdate;
  try
    aEngine.AddField('Region', pvaRow);
    aEngine.AddField('Date', pvaColumn, pvdYear);
    aEngine.AddDataField('Sales', pvfSum);

    // the same measure as a share of the total of its year
    oField := aEngine.AddDataField('Sales', pvfSum);
    oField.Caption := '% of year';
    oField.ShowAs := pvwPercentOfColumn;

    // and as the change against the previous year
```

```
    oField := aEngine.AddDataField('Sales', pvfSum);
    oField.Caption := 'Change';
    oField.ShowAs := pvwDifferenceFrom;
    oField.ShowAsAxis := pvxColumns;
    oField.ShowAsBaseItem := '';
  finally
    aEngine.EndUpdate;
  end;
end;
```

Voir aussi

- Agrégations
- Éditions

Champs calculés

Un champ calculé est une colonne numérique calculée pour chaque ligne source à partir d'une expression. Les champs l'utilisent avec FieldName exactement comme une colonne source. C'est une fonction avancée, qui ne fait pas partie de l'édition Community.

Niveau de ligne

L'expression est évaluée par ligne source puis agrégée, de sorte que la somme de `Profit = [Revenue] - [Cost]` est la somme des bénéfices. Ce n'est pas la formule sur les totaux que calcule Excel : la moyenne d'une marge par ligne n'est pas la marge des totaux. Pour un ratio de totaux, affichez les deux mesures et calculez le ratio dans On-GetCellText, ou utilisez afficher les valeurs en tant que.

Syntaxe

Élément	Syntaxe
Nombres	Décimaux, avec un point et un exposant facultatif : 12, 0.5, 1E3.
Fields	[Field Name] ou un nom simple composé de lettres, de chiffres et de soulignés, comparé aux colonnes source et aux autres champs calculés, sans distinction de casse.
Opérateurs	De la plus faible précédence à la plus forte : `OR` ; `AND` ; `NOT` ; `= <> < <= > >=` ; `+ -` ; `* /` ; `-` et `+` unaires ; `^` (puissance, associatif à droite). Les parenthèses regroupent.
Fonctions	IF(c, a, b), ABS(x), ROUND(x, n), MIN(x, ...), MAX(x, ...), SQRT(x), ISNULL(x, v), YEAR(d), MONTH(d), sans distinction de casse.

- Un opérande vide donne un résultat vide, sauf dans ISNULL, qui retourne v lorsque x est vide, et dans IF, qui ne regarde que la condition et la branche qu'il prend.
- La division par zéro, la racine carrée d'un nombre négatif et un dépassement donnent un résultat vide.
- Les comparaisons et les opérateurs logiques retournent 1 ou 0. Toute valeur différente de 0 est vraie.
- ROUND arrondit les moitiés en s'éloignant de zéro, n va de -15 à 15.

Erreurs

Une erreur de syntaxe ou une référence circulaire entre champs calculés lève EsgcPivotError au recalcul suivant, avec le nom du champ calculé dans le message. Un nom de champ qui n'est pas chargé n'est pas une erreur : le champ calculé reste vide tant que cette colonne n'est pas arrivée, de sorte que la mise en page peut être définie avant les données. Une Expression vide donne aussi des valeurs vides.

Exemple

```
uses
  sgcPivotGrid_Types, sgcPivotGrid_Fields, sgcPivotGrid_Engine;

procedure AddProfit(aEngine: TsgcPivotEngine);
var
  oCalc: TsgcPivotCalculatedField;
begin
  aEngine.BeginUpdate;
  try
    oCalc := aEngine.CalculatedFields.Add;
    oCalc.Name := 'Profit';
```

```
    oCalc.Expression := '[Revenue] - [Cost]';

    // a calculated field can use another one
    oCalc := aEngine.CalculatedFields.Add;
    oCalc.Name := 'Margin';
    oCalc.Expression := 'IF([Revenue] > 0, ROUND([Profit] / [Revenue] * 100, 1), 0)';

    aEngine.AddField('Product', pvaRow);
    aEngine.AddDataField('Profit', pvfSum);
    aEngine.AddDataField('Margin', pvfAverage).DisplayFormat := '0.0';
  finally
    aEngine.EndUpdate;
  end;
end;
```

Voir aussi

- TsgcPivotCalculatedField
- Agrégations

Persistance de la mise en page

La mise en page d'un tableau croisé, ses champs, filtres, totaux et membres développés, est enregistrée et chargée au format JSON. C'est une fonction avancée, qui ne fait pas partie de l'édition Community.

Méthode	Description
SaveLayoutToString: string	La mise en page sous forme de texte JSON.
LoadLayoutFromString(aJSON)	Remplace la mise en page avec un seul recalcul.
SaveLayoutToFile(aFileName)	Écrit le JSON dans un fichier UTF-8.
LoadLayoutFromFile(aFileName)	Lit un fichier UTF-8 et remplace la mise en page.

TsgcPivotEngine et les deux grilles ont ces méthodes. Les versions des grilles enregistrent aussi les largeurs de colonne modifiées par l'utilisateur.

Format

Le document est un seul objet : `v` (la version du format, 1), `rowGrandTotals`, `columnGrandTotals`, `subtotalPosition`, `calculatedFields` (nom et expression), `fields` (un objet par champ, regroupé par zone, avec chaque propriété de TsgcPivotField), `expanded` (l'état développé des niveaux et des chemins de membre des lignes et des colonnes) et, depuis une grille, `columnWidths`. Les énumérations sont écrites sous la forme du nom du membre sans son préfixe et avec une première lettre minuscule : pvfCountDistinct devient `countDistinct`.

Règles de chargement

- Le chargement est tolérant : les membres inconnus sont ignorés, et un membre manquant ou une valeur de mauvais type conserve la valeur par défaut.
- Une fois que le moteur a des données, un champ dont le nom n'est ni une colonne source ni un champ calculé est ignoré.
- Une mise en page chargée avant les données conserve son état développé jusqu'à ce que les champs correspondent aux données.
- Un JSON mal formé, ou une racine qui n'est pas un objet, lève EsgcPivotError et laisse la mise en page inchangée.

Exemple

```
uses
  SysUtils,
  sgcPivotGrid_Types, sgcPivotGrid_Engine, sgcPivotGrid_VCL;

procedure SaveUserLayout(aGrid: TsgcVCLPivotGrid; const aFileName: string);
begin
  // the fields, filters, totals, expanded members and column widths
  aGrid.SaveLayoutToFile(aFileName);
end;

function RestoreUserLayout(aGrid: TsgcVCLPivotGrid;
  const aFileName: string): Boolean;
begin
  Result := FileExists(aFileName);
  if Result then
  try
    aGrid.LoadLayoutFromFile(aFileName);
```

```
  except
    on EsgcPivotError do
      Result := False; // malformed file, the layout is unchanged
  end;
end;

procedure CopyLayout(aFrom, aTo: TsgcPivotEngine);
begin
  aTo.LoadLayoutFromString(aFrom.SaveLayoutToString);
end;
```

Voir aussi

- Zones
- Éditions

Export

Le moteur et les deux grilles exportent la vue actuelle, avec ses totaux et son état développé. Chaque méthode a une surcharge pour fichier et une pour flux, et le texte est écrit en UTF-8 sans BOM.

Méthode	Sortie
SaveToCSV	CSV selon la RFC 4180 avec des enregistrements CRLF. Une colonne par niveau de ligne, une ligne d'en-tête par niveau de colonne plus la ligne des mesures, et un libellé d'en-tête répété dans chaque cellule qu'il couvre. Les valeurs sont écrites comme des nombres invariants, 1250.5, et non avec le format d'affichage, et une cellule vide est un champ vide.
SaveToHTML	Un document HTML complet avec un tableau. Les en-têtes fusionnés utilisent colspan et rowspan, et les valeurs sont les textes formatés des cellules, avec l'EmptyText du moteur dans une cellule vide.
SaveToJSON	L'instantané structuré : les libellés de niveau, les mesures avec leurs formats, et les éléments de colonne et de ligne avec leur nature (member, total ou grandTotal), leur niveau, leur libellé, leur chemin et leur état développé. Un élément de ligne a aussi ses valeurs après afficher les valeurs en tant que, null lorsqu'elles sont vides.
SaveToPNG, SaveToSVG	Les grilles et le moteur de rendu uniquement : toute la grille sous forme d'image. La largeur et la hauteur sont en DIP, 0 correspond à la taille du contenu, jusqu'à 16384 pixels.

Sous forme de table

TsgcPivotEngine et les deux grilles implémentent IsgcUITable par ResultTable, de sorte que le résultat alimente tout consommateur sgcUI, comme une source de données de graphique de sgcCharts ou un autre tableau croisé. Les lignes sont les feuilles de ligne visibles, membres réduits inclus et lignes de total exclues. Les colonnes sont une colonne de texte par niveau de ligne, puis une colonne par feuille de colonne visible et par mesure, nommée *<chemin de colonne>* | *<libellé de la mesure>*, ou seulement le libellé de la mesure lorsqu'il n'y a pas de champs de colonne. Une cellule de valeur vide a l'EmptyText du moteur comme TableText et Null comme TableValue. TableVersion change à chaque nouvel instantané. Voir Interopérabilité avec les graphiques.

Exemple

```
uses
  SysUtils, Classes,
  sgcPivotGrid_VCL;

procedure ExportView(aGrid: TsgcVCLPivotGrid; const aFolder: string);
var
  vPath: string;
  oStream: TMemoryStream;
begin
  vPath := IncludeTrailingPathDelimiter(aFolder);

  // the view with its totals: CSV and HTML tables, the JSON snapshot
  aGrid.SaveToCSV(vPath + 'pivot.csv');
  aGrid.SaveToHTML(vPath + 'pivot.html');
  aGrid.SaveToJSON(vPath + 'pivot.json');

  // the whole grid as an image, 0 is the content size
  aGrid.SaveToPNG(vPath + 'pivot.png');
  aGrid.SaveToSVG(vPath + 'pivot.svg', 1200, 800);
```

```
  // every export also writes to a stream
  oStream := TMemoryStream.Create;
  try
    aGrid.SaveToCSV(oStream);
  finally
    oStream.Free;
  end;
end;
```

Voir aussi

- Démarrage rapide sans interface
- TsgcPivotGridRenderer

Développer et réduire

Tout membre qui a des enfants, dans les lignes ou dans les colonnes, peut être développé pour les afficher ou réduit à une seule ligne ou colonne. Un membre réduit garde sa place et affiche les totaux de tout ce qui se trouve en dessous.

Dans les grilles

- Les boutons + et - des en-têtes de ligne et des en-têtes de colonne basculent un membre. ShowExpandButtons les masque.
- Un double-clic sur un en-tête de ligne bascule son membre.
- Avec l'en-tête de ligne ayant le focus, Droite développe, Gauche réduit et Entrée bascule. Les touches + et - développent et réduisent le membre de la ligne sélectionnée depuis n'importe quelle colonne. Voir Clavier et accessibilité.
- OnExpandChange est déclenché après que l'utilisateur a développé ou réduit un membre, avec aAxis et aItem. Il n'est pas déclenché pour les méthodes ci-dessous.

Dans le code

Membre	Effet
TsgcPivotEngine.ExpandAll(aAxis), CollapseAll(aAxis)	Tous les membres de pvxRows ou pvxColumns.
TsgcPivotEngine.ExpandLevel(aAxis, aLevel), CollapseLevel(aAxis, aLevel)	Tous les membres d'un niveau, 0 est le niveau le plus externe.
TsgcPivotEngine.SetExpanded(aAxis, aItem, aValue), ToggleExpanded(aAxis, aItem)	Un élément de l'instantané actuel, par son index dans Result.Item. Un élément sans enfants reste inchangé.
TsgcPivotField.Expanded	L'état initial des membres du niveau de ce champ. Par défaut True. Voir Regroupement des dates.
TsgcPivotEngine.GetExpandState(aAxis, aState), SetExpandState(aAxis, aState)	L'état développé explicite d'un axe sous forme de liste de chaînes, par exemple pour le copier vers un autre moteur. Édition Advanced.
TsgcPivotItem.Expandable, Expanded	Lus depuis Result.Item : l'élément a des enfants, et ils sont affichés.

Comment l'état est conservé

- L'état est stocké par chemin de membre, tel que North / Spain, et non par index d'élément. Il survit à un recalcul, à un rechargement des données, à un filtre et à un nouvel ordre de tri, et un membre qui disparaît puis revient conserve son état.
- Lorsque les champs d'un axe changent (un autre champ, un autre ordre ou un autre regroupement de dates), l'état de cet axe repart de la propriété Expanded de ses champs.
- Développer ou réduire construit un nouvel instantané sur les données calculées sans réagréger, ce qui est rapide même sur un grand résultat. Voir le tableau des performances dans Architecture.
- Les exports et ResultTable suivent l'état actuel : un membre réduit est une ligne avec ses totaux.
- La persistance de la mise en page enregistre et charge l'état développé avec la mise en page.

Exemple

```
uses
  Classes,
  sgcPivotGrid_Types, sgcPivotGrid_Engine;

procedure ShowOverview(aEngine: TsgcPivotEngine);
begin
  // rows Region > Country > City, columns Date by year > quarter
  aEngine.CollapseAll(pvxColumns); // only the years
  aEngine.CollapseAll(pvxRows);
  aEngine.ExpandLevel(pvxRows, 0); // the regions show their countries
end;

procedure ExpandRegion(aEngine: TsgcPivotEngine; const aRegion: string);
var
  i, vFound: Integer;
  vItem: TsgcPivotItem;
begin
  vFound := -1;
  aEngine.Lock;
  try
    if aEngine.Result <> nil then
      for i := 0 to aEngine.Result.ItemCount(pvxRows) - 1 do
      begin
        vItem := aEngine.Result.Item(pvxRows, i);
        if (vItem.Kind = pvkMember) and (vItem.Level = 0) and vItem.Expandable
          and (aEngine.Result.NodeLabel(pvxRows, vItem.Node) = aRegion) then
        begin
          vFound := i;
          Break;
        end;
      end;
  finally
    aEngine.Unlock;
  end;
  // after Unlock: the change publishes a new snapshot
  if vFound >= 0 then
    aEngine.SetExpanded(pvxRows, vFound, True);
end;

procedure CopyRowState(aFrom, aTo: TsgcPivotEngine);
var
  oState: TStringList;
begin
  // Advanced edition: the expanded state of the rows of another engine
  oState := TStringList.Create;
  try
    aFrom.GetExpandState(pvxRows, oState);
    aTo.SetExpandState(pvxRows, oState);
  finally
    oState.Free;
  end;
end;
```

SetExpanded est appelé après Unlock, car la modification publie un nouvel instantané.

Voir aussi

- Totaux
- Persistance de la mise en page
- TsgcPivotEngine

Tri

Chaque champ de ligne et de colonne trie ses membres au sein de chaque parent, de sorte que les pays de chaque région sont triés séparément. Trois propriétés de TsgcPivotField le contrôlent.

Propriété	Effet
SortBy	pvsLabel (par défaut) trie par libellé de membre. pvsValue trie par total de chaque membre.
SortOrder	pvoAscending (par défaut) ou pvoDescending.
SortMeasure	Le champ de données, par son AreaIndex, dont les totaux sont comparés par un tri par valeur. Par défaut 0. Top N l'utilise aussi.

Règles

- Un tri par libellé compare les membres de texte sans distinction de casse, avec les règles des paramètres régionaux, et les membres numériques et de date par leur valeur, de sorte que 9 vient avant 10 et 2024-10 après 2024-09.
- Un tri par valeur compare le total de chaque membre pour la mesure SortMeasure : le total sur toutes les colonnes pour un champ de ligne, et sur toutes les lignes pour un champ de colonne. Les membres dont le total est vide vont en dernier, et les membres de même total gardent l'ordre de leur libellé.
- Lorsque SortMeasure n'est pas l'AreaIndex d'un champ de données, les membres sont triés par libellé.
- Le membre vide et le membre Others de Top N restent à la fin dans les deux ordres, Others après le membre vide.
- Top N choisit d'abord les membres ayant les plus grands totaux de SortMeasure, puis SortBy et SortOrder les ordonnent.
- Une modification de SortBy, SortOrder ou SortMeasure recalcule le tableau croisé, une fois par bloc BeginUpdate et EndUpdate.
- Les grilles ne trient pas lorsqu'on clique sur un en-tête. Pour laisser l'utilisateur trier, définissez les propriétés dans le code, par exemple dans OnCellClick lorsque le Part de la position est pvhColumnHeader.

Exemple

```
uses
  sgcPivotGrid_Types, sgcPivotGrid_Fields, sgcPivotGrid_Engine;

procedure SortBySales(aEngine: TsgcPivotEngine);
var
  oRegion, oProduct, oYear: TsgcPivotField;
begin
  aEngine.BeginUpdate;
  try
    aEngine.ClearLayout;
    oRegion := aEngine.AddField('Region', pvaRow);
    oProduct := aEngine.AddField('Product', pvaRow);
    oYear := aEngine.AddField('Date', pvaColumn, pvdYear);
    aEngine.AddDataField('Quantity', pvfSum); // measure 0
    aEngine.AddDataField('Sales', pvfSum); // measure 1
    // the regions by their sales total, the largest first
    oRegion.SortBy := pvsValue;
    oRegion.SortMeasure := 1;
    oRegion.SortOrder := pvoDescending;
    // the products of each region by quantity, the smallest first
    oProduct.SortBy := pvsValue;
    oProduct.SortMeasure := 0;
    // the newest year first: dates sort by date, not by label
    oYear.SortOrder := pvoDescending;
```

```
  finally
    aEngine.EndUpdate;
  end;
end;
```

Voir aussi

- Top N
- Regroupement des dates
- TsgcPivotField

Drill-down

Un drill-down lit les lignes source derrière une cellule : les lignes qui ont été agrégées dans sa valeur. Le moteur les retourne sous forme de numéros de ligne ou de table.

Méthode	Re-tourne	Description
TsgcPivotEngine.GetSourceRows(aRowItem, aDataCol, var aRows)	Integer	Les numéros des lignes source, dans l'ordre des lignes, et leur nombre. aRowItem est un élément de ligne de Result et aDataCol une colonne de données. aDataCol -1 prend toutes les colonnes de l'élément de ligne.
TsgcPivotEngine.DrillDown(aRowItem, aDataCol)	TsgcPivotSourceView	Les mêmes lignes sous forme d'IsgcUITable avec toutes les colonnes du magasin, y compris celles qui ne sont pas dans la mise en page. L'appelant le libère.
TsgcPivotEngine.SourceValue(aRow, aCol)	Variant	Une valeur d'une ligne source, Null lorsqu'elle est vide. SourceColumnName donne les noms de colonne.

Quelles lignes

- Une cellule de membre retourne les lignes de son membre de ligne et de son membre de colonne. Un sous-total ou un total général retourne toutes les lignes situées en dessous, et le membre Others de Top N retourne les lignes de tous les membres qu'il contient.
- Les lignes exclues par un filtre de membre, un filtre d'étiquette, un filtre de rapport ou OnFilterRow ne sont jamais retournées.
- Toutes les mesures d'un élément de colonne ont les mêmes lignes, la mesure de aDataCol ne change donc pas le résultat.

Dans les grilles

Entrée sur une cellule de données et un double-clic sur une cellule de données déclenchent OnDrillDown avec aRowItem, aDataCol et var aHandled. La grille n'a pas d'action propre, l'application affiche donc les lignes dans une boîte de dialogue, une grille ou un état. La démo DataSetPivot ouvre les commandes derrière une cellule.

Durée de vie

- TsgcPivotSourceView lit le magasin du moteur en direct. Après la modification des données source, il n'a plus de lignes, et il doit être libéré avant le moteur.
- Les index d'élément appartiennent à l'instantané actuel. Prenez-les dans les événements de la grille, ou lisez Result entre Lock et Unlock.

Exemple

```
uses
  Classes, SysUtils, Variants,
  sgcPivotGrid_Types, sgcPivotGrid_Engine, sgcPivotGrid_VCL;

procedure ListOrders(aEngine: TsgcPivotEngine; aRowItem, aDataCol: Integer;
  aList: TStrings);
var
```

```
  vRows: TsgcPivotIntegers;
  i, vCount, vOrderCol, vSalesCol: Integer;
begin
  aList.Clear;
  vOrderCol := -1;
  vSalesCol := -1;
  for i := 0 to aEngine.SourceColumnCount - 1 do
    if SameText(aEngine.SourceColumnName(i), 'Order') then
      vOrderCol := i
    else if SameText(aEngine.SourceColumnName(i), 'Sales') then
      vSalesCol := i;
  if (vOrderCol < 0) or (vSalesCol < 0) then
    Exit;
  // the source rows behind the cell, aDataCol -1 takes the whole row
  vCount := aEngine.GetSourceRows(aRowItem, aDataCol, vRows);
  for i := 0 to vCount - 1 do
    aList.Add(VarToStr(aEngine.SourceValue(vRows[i], vOrderCol)) + ': ' +
      VarToStr(aEngine.SourceValue(vRows[i], vSalesCol)));
end;

type
  TDrillDownEvents = class
  public
    Lines: TStrings;
    procedure DrillDown(Sender: TObject; aRowItem, aDataCol: Integer;
      var aHandled: Boolean);
  end;

procedure TDrillDownEvents.DrillDown(Sender: TObject;
  aRowItem, aDataCol: Integer; var aHandled: Boolean);
var
  oView: TsgcPivotSourceView;
  r, c: Integer;
  vLine: string;
begin
  // every store column of the rows behind the cell, as an IsgcUITable
  oView := TsgcVCLPivotGrid(Sender).Pivot.DrillDown(aRowItem, aDataCol);
  try
    Lines.Clear;
    for r := 0 to oView.TableRowCount - 1 do
    begin
      vLine := '';
      for c := 0 to oView.TableColCount - 1 do
      begin
        if c > 0 then
          vLine := vLine + #9;
        vLine := vLine + oView.TableText(r, c);
      end;
      Lines.Add(vLine);
    end;
  finally
    oView.Free;
  end;
  aHandled := True;
end;
```

Voir aussi

- TsgcPivotEngine
- TsgcVCLPivotGrid
- Filtres

Mise en forme conditionnelle

La collection FormatRules des grilles et du moteur de rendu colore les cellules de données selon leurs valeurs. Les mêmes règles s'appliquent à l'écran et dans les images PNG et SVG, car les grilles et les images sont dessinées par le même moteur de rendu. À la conception, les règles se modifient avec l'éditeur de collection de l'Object Inspector.

Kind	Ce qu'elle fait
pvrValue	Une règle de valeur. Lorsque la valeur correspond à Operator, Value et Value2 (les opérateurs des filtres), la cellule reçoit BackColor, ForeColor et Bold. pvcNone correspond à toute valeur non vide.
pvrColorScale	Une échelle de couleurs. Les cellules reçoivent une couleur entre MinColor et MaxColor, selon leur place entre la plus petite et la plus grande valeur des cellules de membre de la mesure. Operator, Value et Value2 sélectionnent les valeurs qu'elle colore.

Quelles cellules

- Measure sélectionne le champ de données par son AreaIndex, et -1 applique la règle à toutes les mesures.
- Scope sélectionne les types de cellules : pveCells (la valeur par défaut, les cellules de membre), pveTotals et pveGrandTotals.
- Une cellule vide ne correspond à aucune règle et n'est jamais colorée. Elle affiche l'EmptyText du moteur.

Ordre des règles

- Les règles s'exécutent dans l'ordre de la collection. Chaque règle active correspondante applique ses couleurs différentes de 0, de sorte qu'une correspondance ultérieure remplace une précédente, et que Bold s'accumule.
- StopOnMatch met fin à l'évaluation pour cette cellule, la première règle correspondante l'emporte donc. L'exemple l'utilise pour trois bandes.
- OnGetCellStyle s'exécute après les règles avec var aBack, aFore et aBold, et peut modifier chacun d'eux. À l'intérieur, Renderer.CurrentResult est l'instantané en cours de dessin.

Couleurs

Les couleurs sont des valeurs TsgcUIColor au format $AARRGGBB, $FF2E7D32 est donc un vert opaque, et 0 conserve la couleur du thème. Avec le thème sombre, choisissez des couleurs qui gardent le texte lisible, et combinez la couleur avec Bold pour que le sens ne repose pas sur la couleur seule.

Exemple

```
uses
  sgcUICore_Canvas, sgcPivotGrid_Types, sgcPivotGrid_Result,
  sgcPivotGrid_Format, sgcPivotGrid_VCL;

procedure AddTrafficLights(aGrid: TsgcVCLPivotGrid);
var
  oRule: TsgcPivotFormatRule;
begin
  aGrid.FormatRules.Clear;
  // green from 10,000: StopOnMatch makes the first match win
  oRule := aGrid.FormatRules.Add;
  oRule.Measure := 0;
  oRule.Operator := pvcGreaterEqual;
  oRule.Value := 10000;
```

```
  oRule.BackColor := $FFC8E6C9;
  oRule.StopOnMatch := True;
  // amber from 5,000
  oRule := aGrid.FormatRules.Add;
  oRule.Measure := 0;
  oRule.Operator := pvcGreaterEqual;
  oRule.Value := 5000;
  oRule.BackColor := $FFFFECB3;
  oRule.StopOnMatch := True;
  // red for every other value, in bold so that it does not rely on colour
  oRule := aGrid.FormatRules.Add;
  oRule.Measure := 0;
  oRule.Operator := pvcNone;
  oRule.BackColor := $FFFFCDD2;
  oRule.Bold := True;
end;

type
  TStyleEvents = class
  public
    procedure GetCellStyle(Sender: TObject; aRowItem, aDataCol: Integer;
      var aBack, aFore: TsgcUIColor; var aBold: Boolean);
  end;

procedure TStyleEvents.GetCellStyle(Sender: TObject;
  aRowItem, aDataCol: Integer; var aBack, aFore: TsgcUIColor;
  var aBold: Boolean);
var
  oResult: TsgcPivotResult;
  vItem: TsgcPivotItem;
begin
  // the snapshot being drawn, valid while the event runs
  oResult := TsgcVCLPivotGrid(Sender).Renderer.CurrentResult;
  if (oResult = nil) or (oResult.MeasureCount = 0) then
    Exit;
  // the grand total column in blue text
  vItem := oResult.Item(pvxColumns, aDataCol div oResult.MeasureCount);
  if vItem.Kind = pvkGrandTotal then
    aFore := $FF1565C0;
end;
```

Voir aussi

- TsgcPivotFormatRule
- TsgcPivotGridRenderer
- Clavier et accessibilité

Interopérabilité avec les graphiques

sgcPivotGrid échange des données avec les autres produits sgcUI par l'intermédiaire d'IsgcUITable, l'interface de table de sgcUICore dans l'unité sgcUICore_Table. Les packages de sgcPivotGrid ne dépendent ni de sgcCharts ni de sgcGrid, c'est l'application qui les relie.

Du tableau croisé vers sgcCharts

TsgcPivotEngine et les deux grilles implémentent IsgcUITable par ResultTable. Affectez le tableau croisé à la propriété Table d'un TsgcChartDataSource_Table de sgcCharts, ou appelez son SetTable avec un moteur sans grille.

Partie de la table	Contenu
Lignes	Les feuilles de ligne visibles : membres réduits inclus, lignes de sous-total et de total général exclues.
Colonnes de texte	Une par niveau de ligne, nommée d'après le libellé du niveau, avec le libellé du membre à ce niveau.
Colonnes de valeur	Une par feuille de colonne visible et par mesure, nommée *<chemin de colonne>* | *<libellé de la mesure>*, par exemple `Online | Sales` , ou seulement le libellé de la mesure lorsqu'il n'y a pas de champs de colonne.
TableValue	Le libellé d'une colonne de texte, et la valeur affichée d'une colonne de valeur sous forme de Double (après afficher les valeurs en tant que), ou Null lorsque la cellule est vide.
TableText	Le libellé, ou le texte formaté de la cellule. Une cellule vide donne l'EmptyText du moteur.

- Associez les colonnes à la source de données : YColumn nomme la colonne de valeur, LabelColumn le niveau de ligne d'un graphique en secteurs, en entonnoir ou d'un treemap (plusieurs niveaux séparés par ;), et XColumn les valeurs X d'une série XY.
- Donnez au champ de données un Caption, tel que Sales, afin que les noms de colonne ne changent pas avec l'agrégation.
- Chaque nouvel instantané modifie TableVersion et notifie les écouteurs, de sorte que le graphique se recharge après un recalcul, un développement ou une réduction.
- La démo GridToPivotToChart dessine des barres groupées avec une source de données par canal, et remplit les catégories de l'axe X à partir des libellés de ligne dans OnCalculated.

De sgcGrid vers le tableau croisé

TsgcPivotSource_Table charge n'importe quel IsgcUITable dans le moteur : un TsgcVCLGrid, un TsgcFMXGrid ou un TsgcGridData de sgcGrid. Il recharge lorsque TableVersion change, de sorte qu'une modification dans la grille met à jour le tableau croisé puis le graphique. Regroupez de nombreuses modifications de la grille dans BeginUpdate et EndUpdate des données de la grille, ce qui donne un seul rechargement.

D'un tableau croisé vers un autre

Le résultat d'un tableau croisé est lui aussi une table, TsgcPivotSource_Table peut donc le charger dans un second tableau croisé.

Lecture de la table dans le code

Lisez la table entre TableLock et TableUnlock. Le comptage de références est désactivé, le moteur possède la table, libérez donc toute référence d'interface avant de libérer le moteur.

Exemple

```
uses
  Classes,
  sgcGrid_VCL, sgcPivotGrid_Types, sgcPivotGrid_Fields,
  sgcPivotGrid_Source_Table, sgcPivotGrid_VCL, sgcCharts_Chart,
  sgcCharts_Series_Pie, sgcCharts_DataSource_Table, sgcCharts_VCL;

procedure LinkGridPivotChart(aOwner: TComponent; aGrid: TsgcVCLGrid;
  aPivot: TsgcVCLPivotGrid; aView: TsgcVCLChart);
var
  oField: TsgcPivotField;
  oSource: TsgcPivotSource_Table;
  oChart: TsgcChart;
  oPie: TsgcChartSeries_Pie;
  oData: TsgcChartDataSource_Table;
begin
  // the layout: one row per region with the sum of the sales
  aPivot.Pivot.BeginUpdate;
  try
    aPivot.Pivot.ClearLayout;
    aPivot.Pivot.AddField('Region', pvaRow);
    oField := aPivot.Pivot.AddDataField('Sales', pvfSum);
    oField.Caption := 'Sales';
  finally
    aPivot.Pivot.EndUpdate;
  end;

  // sgcGrid to pivot: the grid implements IsgcUITable
  oSource := TsgcPivotSource_Table.Create(aOwner);
  oSource.Table := aGrid;
  oSource.PivotGrid := aPivot;
  oSource.Active := True;

  // pivot to sgcCharts: one pie slice per region
  oChart := TsgcChart.Create(aOwner);
  oPie := TsgcChartSeries_Pie.Create(aOwner);
  oPie.Chart := oChart;
  oData := TsgcChartDataSource_Table.Create(aOwner);
  oData.Series := oPie;
  oData.LabelColumn := 'Region';
  oData.YColumn := 'Sales';
  oData.Table := aPivot;
  aView.Chart := oChart;
end;
```

Voir aussi

- Export
- TsgcPivotSource_Table
- Démos

Clavier et accessibilité

Les deux grilles peuvent s'utiliser uniquement au clavier. Les mêmes touches fonctionnent sur TsgcVCLPivotGrid et TsgcFMXPivotGrid, où Cmd sous macOS fonctionne comme Ctrl.

Touche	Action
Haut, Bas	La ligne précédente ou la suivante.
Page préc., Page suiv.	Une page vers le haut ou vers le bas.
Début, Fin	La première ou la dernière colonne de données de la ligne. Avec Ctrl, la première ou la dernière cellule de la grille.
Gauche	La colonne de données précédente. Depuis la première colonne de données, elle passe à l'en-tête de ligne, et dans l'en-tête de ligne elle réduit le membre.
Droite	Dans l'en-tête de ligne, elle développe un membre réduit, sinon elle passe à la première colonne de données. Dans les données, elle passe à la colonne suivante.
+, -	Sur le pavé numérique ou le clavier principal : développe ou réduit le membre de la ligne sélectionnée, depuis n'importe quelle colonne.
Entrée	Sur une cellule de données, elle déclenche OnDrillDown. Dans l'en-tête de ligne, elle bascule le membre.

- Lorsque rien n'est sélectionné, la première touche de navigation sélectionne la première cellule.
- TsgcVCLPivotGrid a TabStop à True par défaut et TsgcFMXPivotGrid a CanFocus à True. Un clic donne le focus à la grille.
- Dans le code, SelectedRow, SelectedCol et ScrollToCell déplacent la sélection, et OnSelectionChange signale chaque modification.
- Les touches développent et réduisent les membres de ligne. Les membres de colonne se développent avec les boutons des en-têtes de colonne ou dans le code, voir Développer et réduire.
- Les listes de champs s'utilisent à la souris ou au toucher. Échap annule un glissement, et TsgcFMXPivotFieldList fait aussi défiler sa liste avec Haut, Bas, Page préc., Page suiv., Début et Fin. Pour les utilisateurs du clavier, proposez les mises en page dans le code avec AddField et MoveField, ou sous forme de mises en page enregistrées.

Accessibilité

- Les grilles et les listes de champs sont dessinées par sgcUICore et ne publient pas leur contenu aux lecteurs d'écran via UI Automation ou MSAA. Pour les utilisateurs de lecteurs d'écran, proposez les données en HTML : SaveToHTML écrit un vrai tableau avec des cellules d'en-tête, colspan et rowspan. ResultTable peut aussi remplir un contrôle standard.
- Les tailles sont en DIP et suivent le DPI du moniteur. FontSize, RowHeight et HeaderHeight agrandissent le texte et les lignes.
- Theme sélectionne le thème clair ou sombre, et AccentColor remplace la couleur d'accent du thème.
- La mise en forme conditionnelle peut ajouter Bold à une couleur, afin qu'une valeur ne repose pas sur la couleur seule.

Exemple

```
uses
  Classes, Controls,
  sgcPivotGrid_Types, sgcPivotGrid_VCL;
```

```
procedure FocusFirstCell(aGrid: TsgcVCLPivotGrid);
begin
  // the first data cell gets the selection and the grid the keyboard
  aGrid.SelectedRow := 0;
  aGrid.SelectedCol := 0;
  aGrid.ScrollToCell(0, 0);
  if aGrid.CanFocus then
    aGrid.SetFocus;
end;

// from the OnKeyDown of a form with KeyPreview: Ctrl+E expands every row,
// Ctrl+Shift+E collapses them
procedure HandlePivotShortcut(aGrid: TsgcVCLPivotGrid; var Key: Word;
  Shift: TShiftState);
begin
  if not aGrid.Focused or (Key <> Ord('E')) or not(ssCtrl in Shift) then
    Exit;
  if ssShift in Shift then
    aGrid.Pivot.CollapseAll(pvxRows)
  else
    aGrid.Pivot.ExpandAll(pvxRows);
  Key := 0;
end;
```

Voir aussi

- TsgcVCLPivotGrid
- Drill-down

Performances

Le moteur est conçu pour de grands volumes de données : un magasin de colonnes typé, un seul passage sur les lignes source par recalcul et un dessin virtuel. Le groupe de performance du QA mesure les principaux scénarios par rapport aux budgets listés dans Architecture. Ces recommandations maintiennent une application dans ces limites.

Chargement

- Chargez dans BeginLoad et EndLoad. Le moteur recalcule une seule fois, au EndLoad le plus externe. En dehors d'un bloc, chaque AppendRow et DefineColumn recalcule immédiatement.
- Définissez les colonnes avec DefineColumn(aName, aType). Sans type, la première valeur non vide en décide, et une colonne d'entiers devient une colonne de nombres flottants lorsqu'une valeur fractionnaire arrive.
- TsgcPivotSource_DataSet recharge tout le jeu de données après chaque post tant que LiveUpdates est True. Pour un lot de modifications, mettez LiveUpdates ou Active à False, et appelez Load à la fin.
- TsgcPivotSource_Table ne recharge que lorsque TableVersion change. Regroupez les modifications d'une grille dans son BeginUpdate et EndUpdate.

Mise en page et vue

- Modifiez la mise en page dans BeginUpdate et EndUpdate pour un seul recalcul. Pour des lots plus longs, mettez AutoRecalculate à False et appelez Recalculate à la fin.
- DeferUpdate des listes de champs garde en attente les modifications de l'utilisateur jusqu'à ApplyChanges.
- Expand, collapse, RowGrandTotals, ColumnGrandTotals, SubtotalPosition, RowLayout et EmptyText ne modifient que la vue. Le moteur construit un nouvel instantané sur les données calculées sans réagréger.

Taille du résultat

- Le nombre de cellules croît avec les membres des lignes multipliés par les membres des colonnes. Un champ à nombreux membres, comme un numéro de commande ou une date au jour, sur les deux axes les multiplie. Une mise en page qui nécessite plus de MaxCells cellules (5 000 000 par défaut) lève EsgcPivotError. Regroupez les dates, déplacez le champ vers les filtres, ou utilisez Top N.
- Chaque champ de données ajoute une valeur par cellule. CountDistinct conserve les valeurs distinctes de chaque cellule, et StdDev et Var conservent plus d'état que Sum, ils coûtent donc plus de mémoire et de temps sur de grandes mises en page.

Événements

- OnFilterRow s'exécute pour chaque ligne source à chaque recalcul. Recherchez les index de colonne une seule fois, en dehors de l'événement, comme le fait l'exemple de Filtres. Les filtres de membre et d'étiquette sont décidés une fois par membre, ils coûtent donc moins.
- OnGetCellText et OnGetCellStyle s'exécutent pour chaque cellule visible à chaque image. Gardez-les courts.
- Ne conservez le verrou de Result que pendant sa lecture.

Exemple

```
uses
  SysUtils,
  sgcPivotGrid_Types, sgcPivotGrid_Engine;

const
  CS_REGIONS: array [0 .. 3] of string = ('North', 'South', 'East', 'West');
```

```
procedure LoadSales(aEngine: TsgcPivotEngine; aCount: Integer);
var
  i: Integer;
  vRegion: string;
  vDate: TDateTime;
  vSales: Double;
begin
  // one load bracket: one recalculation at EndLoad, not one per row
  aEngine.BeginLoad(True);
  try
    // fixed types: no inference from the first value
    aEngine.DefineColumn('Region', pvtText);
    aEngine.DefineColumn('Date', pvtDateTime);
    aEngine.DefineColumn('Sales', pvtFloat);
    for i := 0 to aCount - 1 do
    begin
      vRegion := CS_REGIONS[i mod 4];
      vDate := EncodeDate(2024, 1, 1) + (i mod 366);
      vSales := (i mod 100) * 1.5;
      aEngine.AppendRow([vRegion, vDate, vSales]);
    end;
  finally
    aEngine.EndLoad;
  end;
end;

procedure SetLayout(aEngine: TsgcPivotEngine);
begin
  // one recalculation for the whole layout, at EndUpdate
  aEngine.BeginUpdate;
  try
    aEngine.ClearLayout;
    aEngine.AddField('Region', pvaRow);
    aEngine.AddField('Date', pvaColumn, pvdQuarter);
    aEngine.AddDataField('Sales', pvfSum);
  finally
    aEngine.EndUpdate;
  end;
end;
```

Voir aussi

- Architecture
- TsgcPivotEngine
- Dépannage

TsgcPivotGridAIQuery

Unit: sgcPivotGridWS_AI
Package: sgcPivotGridWS (Delphi) et sgcPivotGridWSC (C++Builder), la passerelle sgcWebSockets de sgcPivotGrid. Édition Registered, Windows uniquement (Win32 et Win64).
TsgcPivotGridAIQuery est un composant non visuel qui transforme une question en langage naturel, comme *top 3 countries by sales*, en disposition de tableau croisé. Il envoie la question et un catalogue des champs source à un fournisseur d'IA par l'intermédiaire d'un **TsgcAIChat** de sgcWebSockets, valide la réponse JSON par rapport aux champs du moteur et l'applique en un seul recalcul. Les lignes source ne sont jamais envoyées. Voir Interrogez votre tableau croisé pour ce qui est envoyé, la validation de la réponse et le mode asynchrone.
Déposez un TsgcAIChat et un TsgcPivotGridAIQuery depuis la page **SGC PivotGrid** de la palette, renseignez Chat et PivotGrid, puis appelez Ask. Le fournisseur, le modèle et la clé d'API sont des propriétés du chat : Provider, ChatOptions.Model et ChatOptions.ApiKey.

Propriétés

Propriété	Type	Description
Active	Boolean	À False, Ask et AskAsync renvoient False et ne font rien. True par défaut.
Chat	TsgcAI_Chat	Le chat qui dialogue avec le fournisseur : un TsgcAIChat ou tout descendant de TsgcAI_Chat. Sans chat, Ask déclenche OnError.
PivotGrid	TComponent	Le tableau croisé dont la disposition est modifiée : un TsgcVCLPivotGrid, un TsgcFMXPivotGrid ou tout composant qui implémente IsgcPivotEngineHost.
Engine	TsgcPivotEngine	Public. Un moteur à utiliser sans grille, par exemple sur un serveur. Il l'emporte sur PivotGrid.
SendSampleValues	Boolean	Envoie avec le catalogue quelques valeurs de chaque champ texte, entier, date et booléen, afin que le fournisseur puisse reconnaître les membres d'une question comme *online sales*. True par défaut.
MaxSampleValues	Integer	Le nombre de valeurs d'exemple par champ, borné à 0..5. 3 par défaut.
Language	string	La langue de l'explication, par exemple `Spanish` . Vide, la valeur par défaut, elle est rédigée dans la langue de la question.
ReplaceLayout	Boolean	True, la valeur par défaut, demande une nouvelle disposition. False envoie aussi la disposition actuelle, si bien qu'une question comme *add the channel to the columns* la modifie. La réponse est dans les deux cas la nouvelle disposition complète.
Busy	Boolean	Public, en lecture seule. True pendant qu'une requête AskAsync s'exécute.
LastLayout	TsgcPivotAILayout	Public. La disposition appliquée par la dernière réponse. Elle est vide après Undo.
LastExplanation	string	Public. L'explication de la dernière réponse appliquée, une courte phrase pour une barre d'état.

Événements

Événement	Type	Description
OnBeforeAsk	TsgcPivotAIBeforeAskEvent	`(Sender; var aPrompt: string; var aAllow: Boolean)` . Se déclenche dans le thread principal avant la requête, avec le message utilisateur. Modifiez aPrompt pour ajouter vos propres instructions, ou mettez aAllow à False pour annuler la question en silence.
OnLayout	TsgcPivotAILayoutEvent	`(Sender; aLayout: TsgcPivotAILayout; var aAccept: Boolean)` . Se déclenche avec la disposition validée avant son application. Inspectez ou modifiez aLayout, ou mettez aAccept à False pour conserver la disposition actuelle.
OnApplied	TNotifyEvent	Se déclenche après l'application de la disposition. LastLayout et LastExplanation contiennent la réponse.
OnError	TsgcPivotAIErrorEvent	`(Sender; const aError: string)` . Se déclenche quand rien n'a été appliqué : pas de chat, pas de moteur, une question vide, une requête encore en cours, une erreur du fournisseur, une réponse sans objet JSON, ou une réponse sans champ de ligne, de colonne ou de données exploitable. La disposition reste inchangée.

Méthodes

Méthode	Retourne	Description
Ask(aQuestion)	Boolean	Interroge le fournisseur, valide la réponse et l'applique. Synchrone, thread principal uniquement. True quand une disposition a été appliquée.
AskAsync(aQuestion)	Boolean	Idem, avec la requête dans un thread de travail. La réponse est validée et appliquée dans le thread principal, puis OnApplied ou OnError se déclenche. False quand Busy est True ou que la requête n'a pas démarré.
Undo		Restaure la disposition antérieure à la dernière réponse appliquée. Un seul niveau est conservé.
BuildPrompt(aQuestion)	string	Le message utilisateur que Ask envoie pour aQuestion : la question, la ligne de langue et le catalogue des champs. Vide s'il n'y a pas de moteur. Utilisez-le pour montrer à l'utilisateur ce qui quitte l'application.
ValidateLayoutJSON(aJSON, aLayout)	Boolean	Extrait, analyse et valide une réponse ou une disposition enregistrée par rapport aux champs du moteur, dans aLayout. False, avec aLayout vide, quand aucun objet JSON n'est trouvé ou qu'aucun champ de ligne, de colonne ou de données ne subsiste.
ResolveEngine	TsgcPivotEngine	Engine s'il est affecté, sinon le moteur de PivotGrid, sinon nil.

Les descendants peuvent surcharger trois méthodes virtuelles protégées : **DoAskProvider**, la seule méthode qui dialogue avec le fournisseur (un test peut en renvoyer une réponse prédéfinie, comme le fait la démo AskYourPivot dans son autotest), **DoBuildFields**, qui construit le catalogue des champs (un champ omis ne peut être ni envoyé ni accepté), et **DoApply**, qui remplace la disposition du moteur.

TsgcPivotAILayout

La disposition d'une réponse, après validation. OnLayout la reçoit et LastLayout conserve celle qui a été appliquée.

Membre	Description
Rows, Columns	Les champs d'axe dans l'ordre : FieldName et Grouping (TsgcPivotDateGrouping).
Data	Les champs de données dans l'ordre : FieldName et Aggregation (TsgcPivotAggregation). L'index d'une entrée est l'index de mesure auquel se réfèrent les filtres, les tris et Top N.
Filters	Genre `pafMembers` (Members, Exclude), `pafLabel` (FilterOperator, Value, Value2) ou `pafValue` (FilterOperator, NumValue, NumValue2, Measure).
Sorts	FieldName, SortBy, SortOrder et Measure.
TopN	FieldName, Count, Measure et Others.
Explanation	L'explication du fournisseur, nettoyée.
Clear, IsEmpty, Assign	Vide la disposition, teste l'absence d'entrée de ligne, de colonne et de données, copie une autre disposition.
ToJSON	La disposition sous la forme d'un objet JSON compact conforme au schéma, de sorte qu'une disposition enregistrée se valide à nouveau.
ToText	Un résumé d'une ligne de la disposition, pour une barre d'état.

Exemple

Un assistant qui possède le chat et la requête, interroge de façon synchrone et refuse une disposition de plus de deux champs de ligne :

```
uses
  SysUtils, Classes,
  sgcAI, sgcAI_Chat, sgcPivotGrid_VCL, sgcPivotGridWS_AI;

type
  TPivotAssistant = class(TComponent)
  private
    FChat: TsgcAIChat;
    FQuery: TsgcPivotGridAIQuery;
    FLog: TStrings;
    procedure QueryLayout(Sender: TObject; aLayout: TsgcPivotAILayout;
      var aAccept: Boolean);
    procedure QueryApplied(Sender: TObject);
    procedure QueryError(Sender: TObject; const aError: string);
  public
    procedure Setup(aGrid: TsgcVCLPivotGrid; const aApiKey: string;
      aLog: TStrings);
    function Ask(const aQuestion: string): Boolean;
  end;

procedure TPivotAssistant.Setup(aGrid: TsgcVCLPivotGrid;
  const aApiKey: string; aLog: TStrings);
begin
  FLog := aLog;
  // the provider, the model and the key live on the chat
  FChat := TsgcAIChat.Create(Self);
  FChat.Provider := aicpOpenAI;
```

```
  FChat.ChatOptions.Model := 'gpt-4o-mini';
  FChat.ChatOptions.ApiKey := aApiKey;

  FQuery := TsgcPivotGridAIQuery.Create(Self);
  FQuery.Chat := FChat;
  FQuery.PivotGrid := aGrid;
  FQuery.MaxSampleValues := 3;
  FQuery.OnLayout := QueryLayout;
  FQuery.OnApplied := QueryApplied;
  FQuery.OnError := QueryError;
end;

function TPivotAssistant.Ask(const aQuestion: string): Boolean;
begin
  // synchronous: ask, validate and apply, in the main thread
  Result := FQuery.Ask(aQuestion);
end;

procedure TPivotAssistant.QueryLayout(Sender: TObject;
  aLayout: TsgcPivotAILayout; var aAccept: Boolean);
begin
  // the validated layout, before it replaces the current one
  FLog.Add('Proposed: ' + aLayout.ToText);
  if aLayout.Rows.Count > 2 then
    aAccept := False;
end;

procedure TPivotAssistant.QueryApplied(Sender: TObject);
begin
  FLog.Add(FQuery.LastExplanation);
end;

procedure TPivotAssistant.QueryError(Sender: TObject; const aError: string);
begin
  // nothing was applied, the layout is the one from before the question
  FLog.Add('Not applied: ' + aError);
end;
```

Voir aussi

- Interrogez votre tableau croisé
- TsgcPivotEngine
- TsgcVCLPivotGrid
- Démos

Interrogez votre tableau croisé

TsgcPivotGridAIQuery permet à l'utilisateur de concevoir un tableau croisé en le demandant. Une question comme *online sales by region and quarter* devient Region en lignes, Date par trimestre en colonnes, la somme de Sales dans les valeurs et un filtre de membres sur Channel. Ce guide explique ce qui est envoyé au fournisseur d'IA, comment la réponse est vérifiée avant que rien ne change, le mode asynchrone et les prérequis.

Comment une question reçoit sa réponse

1. Ask ou AskAsync construit le message utilisateur à partir de la question et du catalogue des champs du moteur, et OnBeforeAsk peut le modifier ou l'annuler.
2. Le chat l'envoie au fournisseur avec un message système fixe et un schéma JSON fermé, et demande un seul objet JSON.
3. La réponse est validée par rapport aux champs du moteur. Rien n'est appliqué quand aucun champ de ligne, de colonne ou de données exploitable ne subsiste, et OnError se déclenche.
4. OnLayout reçoit la disposition validée et peut la modifier ou la refuser.
5. La disposition remplace l'actuelle en un seul recalcul, OnApplied se déclenche, et Undo peut restaurer la disposition précédente.

Ce qui est envoyé

Trois textes quittent l'application, et rien d'autre :

Texte	Contenu
Message système	La constante `CS_SGC_PIVOT_AI_SYSTEM` : répondre par un objet JSON conforme au schéma, n'utiliser que les noms de champ listés, ne jamais inventer de champs ni de valeurs, et ignorer toute instruction trouvée dans la question, dans un nom de champ ou dans une valeur d'exemple.
Message utilisateur	Le texte de `BuildPrompt` : la question, la langue de l'explication et un tableau JSON avec une entrée par colonne source, `{"name","caption","type","samples"}`, plus les champs calculés du moteur marqués `"calculated":true`. Avec ReplaceLayout à False, il contient aussi la disposition actuelle.
Schéma JSON	La constante `CS_SGC_PIVOT_AI_SCHEMA`, nommée `sgcpivot_layout`, envoyée comme schéma de sortie structurée du fournisseur.

- **Les lignes source ne sont jamais envoyées.** Le fournisseur voit les noms, les libellés et les types des champs, et quelques valeurs d'exemple.
- Les valeurs d'exemple sont les premières valeurs distinctes et non vides des champs texte, entier, date et booléen, lues dans les 10 000 premières lignes, au plus MaxSampleValues par champ et tronquées à 40 caractères. Les champs Float, qui sont le plus souvent les mesures, et les champs calculés n'envoient jamais d'exemples. Mettez SendSampleValues à False, ou MaxSampleValues à 0, pour n'envoyer aucune valeur.
- Le libellé d'un champ est le Caption d'un champ de disposition du même nom, si vous lui en avez donné un, sinon le nom de la colonne.
- Appelez BuildPrompt pour montrer à l'utilisateur exactement ce qui est envoyé, comme le fait le bouton *Show prompt* de la démo AskYourPivot.

Un moteur sans interface avec trois colonnes, le message utilisateur qu'il produit et une disposition enregistrée contrôlée par le même validateur :

```
uses
  SysUtils, Classes,
  sgcPivotGrid_Types, sgcPivotGrid_Engine, sgcPivotGridWS_AI;

procedure InspectPrompt(aLog: TStrings);
var
  oEngine: TsgcPivotEngine;
```

```
  oQuery: TsgcPivotGridAIQuery;
  oLayout: TsgcPivotAILayout;
begin
  oEngine := TsgcPivotEngine.Create;
  oQuery := TsgcPivotGridAIQuery.Create(nil);
  oLayout := TsgcPivotAILayout.Create;
  try
    oEngine.BeginLoad;
    try
      oEngine.DefineColumn('Region', pvtText);
      oEngine.DefineColumn('Date', pvtDateTime);
      oEngine.DefineColumn('Sales', pvtFloat);
      oEngine.AppendRow(['North', EncodeDate(2025, 1, 15), 1200.0]);
      oEngine.AppendRow(['South', EncodeDate(2025, 4, 2), 450.0]);
    finally
      oEngine.EndLoad;
    end;
    // headless: an engine instead of a pivot grid, no chat needed here
    oQuery.Engine := oEngine;
    oQuery.MaxSampleValues := 2;

    // the user message that Ask sends: the question, the language line
    // and the field catalog with sample values, never the source rows
    aLog.Add(oQuery.BuildPrompt('sales by region and quarter'));

    // a stored layout goes through the same validator as an answer
    if oQuery.ValidateLayoutJSON('{"rows":[{"field":"region","group":"none"}],'
      + '"columns":[{"field":"Date","group":"quarter"}],'
      + '"data":[{"field":"Sales","aggregation":"sum"}],"filters":[],'
      + '"sort":[],"top":[],"explanation":"Sales by region and quarter."}',
      oLayout) then
    begin
      aLog.Add(oLayout.ToText);
      aLog.Add(oLayout.ToJSON);
    end
    else
      aLog.Add('Rejected: no usable row, column or data field');
  finally
    oLayout.Free;
    oQuery.Free;
    oEngine.Free;
  end;
end;
```

La réponse

Le schéma est fermé : chaque objet liste toutes ses clés et n'en admet aucune autre. Une réponse a les clés `rows` et `columns` (field et group : none, year, quarter, month, week ou day), `data` (field et aggregation : sum, count, average, min, max, countDistinct, stdDev ou var), `filters` (un filtre de membres, d'étiquette ou de valeur), `sort` (par étiquette ou par valeur, croissant ou décroissant), `top` (Top N avec un membre Others facultatif) et `explanation`. Une mesure est l'index d'une entrée de `data`, à partir de 0.
Certains fournisseurs refusent le mode de sortie structurée. Quand l'erreur du fournisseur cite le schéma ou le format de réponse, la requête interroge une seconde fois en mode JSON simple, avec le schéma collé dans le message utilisateur. Les quatre réglages du chat qu'elle modifie (ResponseFormat, JSONSchema, JSONSchemaName et SystemMessage) sont restaurés et l'historique du chat est effacé après chaque question, si bien que le chat peut être partagé avec d'autre code.

Validation

La réponse est une entrée non fiable. ValidateLayoutJSON est la frontière entre le fournisseur et le moteur, et toute réponse y passe avant OnLayout :

- La réponse est tronquée à 64 Ko. Le premier bloc de code délimité est cherché d'abord, puis le premier objet JSON équilibré, en ignorant les accolades à l'intérieur des chaînes.

- Un nom de champ doit correspondre exactement à un champ du moteur, sans tenir compte de la casse et sans les espaces extérieurs. Les champs calculés ne sont acceptés que dans les valeurs. Les champs, clés et valeurs d'énumération inconnus sont écartés en silence.
- Une agrégation numérique d'un champ texte ou booléen devient Count, et l'explication le précise. Un regroupement de dates n'est conservé que pour les champs date.
- Au plus 4 entrées de ligne, 4 de colonne, 6 de données, 8 de filtre, 4 de tri et 2 de Top N, 50 membres par filtre, et N entre 1 et 100.
- Les caractères de contrôle sont supprimés, les textes tronqués à 200 caractères et l'explication à 300, sans chevrons.
- Un champ placé en lignes ou en colonnes n'est pas placé aussi dans la zone de filtre. Un filtre de membres sur un champ présent sur aucun axe va dans la zone de filtre.
- Une réponse avec des champs d'axe mais sans données reçoit le Count du premier champ de ligne. Une réponse sans aucun champ de ligne, de colonne ou de données est rejetée en entier : rien ne change et OnError se déclenche.

Comme la même méthode valide une disposition enregistrée, une application peut sauvegarder LastLayout.ToJSON et la contrôler à nouveau plus tard par rapport aux champs actuels.

Application et Undo

Une disposition validée remplace la disposition du moteur en un seul recalcul : les lignes, les colonnes et les champs de données dans l'ordre, puis les filtres, les tris et Top N sur le premier champ de ce nom dans les lignes, sinon dans les colonnes. Undo restaure la disposition antérieure à la dernière réponse appliquée, et LastLayout est alors vide.

Questions asynchrones

Un fournisseur peut mettre plusieurs secondes à répondre. AskAsync n'exécute que la requête dans un thread de travail, si bien que la fenêtre reste réactive. OnBeforeAsk se déclenche dans le thread principal avant le départ de la requête, et la réponse est validée et appliquée dans le thread principal par Synchronize, donc OnLayout, OnApplied et OnError s'y déclenchent aussi.

- Une question à la fois : AskAsync renvoie False tant que Busy est True, et OnError le signale.
- N'utilisez pas le chat depuis votre propre code tant que Busy est True.
- Libérer la requête pendant qu'une demande s'exécute attend le thread de travail et abandonne la réponse.

```
uses
  SysUtils, Classes, StdCtrls,
  sgcPivotGridWS_AI;

type
  TAskController = class
  private
    FQuery: TsgcPivotGridAIQuery;
    FAskButton: TButton;
    FStatus: TLabel;
    procedure QueryBeforeAsk(Sender: TObject; var aPrompt: string;
      var aAllow: Boolean);
    procedure QueryApplied(Sender: TObject);
    procedure QueryError(Sender: TObject; const aError: string);
  public
    constructor Create(aQuery: TsgcPivotGridAIQuery; aAskButton: TButton;
      aStatus: TLabel);
    procedure AskClick(const aQuestion: string);
  end;

constructor TAskController.Create(aQuery: TsgcPivotGridAIQuery;
  aAskButton: TButton; aStatus: TLabel);
begin
  inherited Create;
  FQuery := aQuery;
  FAskButton := aAskButton;
  FStatus := aStatus;
  FQuery.OnBeforeAsk := QueryBeforeAsk;
  FQuery.OnApplied := QueryApplied;
  FQuery.OnError := QueryError;
```

```
end;

procedure TAskController.AskClick(const aQuestion: string);
begin
  if FQuery.Busy then
    Exit;
  FAskButton.Enabled := False;
  FStatus.Caption := 'Asking the provider...';
  // the request runs in a worker thread, the window stays responsive
  if not FQuery.AskAsync(aQuestion) then
    FAskButton.Enabled := True;
end;

procedure TAskController.QueryBeforeAsk(Sender: TObject; var aPrompt: string;
  var aAllow: Boolean);
begin
  // main thread, before the request: edit the user message or veto it
  aPrompt := aPrompt + sLineBreak + 'Prefer the year grouping for dates.';
end;

procedure TAskController.QueryApplied(Sender: TObject);
begin
  // main thread: the answer was validated and applied in Synchronize
  FAskButton.Enabled := True;
  FStatus.Caption := FQuery.LastExplanation;
end;

procedure TAskController.QueryError(Sender: TObject; const aError: string);
begin
  FAskButton.Enabled := True;
  FStatus.Caption := aError;
end;
```

Édition et prérequis

Prére-quis	Détails
Édition	Registered uniquement. La passerelle sgcPivotGridWS ne fait pas partie de l'édition Community.
sgcWeb-Sockets	La passerelle dialogue avec le fournisseur par TsgcAIChat de sgcWebSockets (unité sgcAI), donc sgcWebSockets avec ses composants d'IA doit être installé. Les fournisseurs sont ceux de TsgcAIChat : OpenAI, Anthropic, Gemini, DeepSeek, Ollama, Grok et Mistral.
Plate-formes	Windows uniquement, Win32 et Win64, Delphi 7 à 13 et les versions correspondantes de C++Builder.
Packages	sgcPivotGridWS<V> et dclsgcPivotGridWS<V> pour Delphi, sgcPivotGridWSC<V> et dclsgcPivotGridWSC<V> pour C++Builder. Ils requièrent sgcPivotGrid, sgcUICore et sgcWebSockets.
Clé d'API	Chaque fournisseur exige sa propre clé, renseignée dans ChatOptions.ApiKey du chat. Ollama s'exécute en local et n'en exige pas. Ne stockez pas la clé dans un fichier de fiche.

TsgcAIChat n'existe que si sgcWebSockets est compilé avec `SGC_AI`. Les packages que construit l'installeur de sgcWebSockets le définissent. Si vous compilez vous-même les sources de sgcWebSockets et que le compilateur signale TsgcAI_Chat comme identificateur non déclaré dans sgcPivotGridWS_AI, ajoutez `SGC_CUSTOM_INDY` aux définitions conditionnelles de cette compilation : `SGC_AI` en dépend.

La démo AskYourPivot

`Demos\PivotGrid\Delphi\AskYourPivot` place un TsgcVCLPivotGrid de 100 000 lignes de ventes générées sous un champ de question. Choisissez le fournisseur, vérifiez le modèle et saisissez votre clé d'API, qui n'est conservée qu'en mémoire. Ask utilise AskAsync, Undo restaure la disposition précédente, Examples contient quatre questions d'exemple et Show prompt ouvre le message système, le message utilisateur et le schéma. Lancez-la avec `--test` pour la contrôler avec des réponses prédéfinies, sans réseau et sans clé. Voir Démos.

Voir aussi

- TsgcPivotGridAIQuery
- Zones
- Filtres
- Top N

Présentation de .NET

sgcPivotGrid .NET est un portage managé en C# de la bibliothèque Delphi sgcPivotGrid, et non une enveloppe autour d'elle. Les classes se trouvent dans l'espace de noms **esegece.sgcPivotGrid** et gardent les noms Delphi, de sorte que **TsgcPivotEngine**, **TsgcPivotField**, **TsgcPivotResult**, **TsgcPivotGridRenderer**, **TsgcPivotFormatRule**, **TsgcPivotSource_DataSet** et **TsgcPivotSource_Table** se lisent de la même façon dans les deux langages. Les vues suivent le modèle de nommage Delphi : **TsgcWinFormsPivotGrid** et **TsgcWinFormsPivotFieldList** pour WinForms, **TsgcWpfPivotGrid** et **TsgcWpfPivotFieldList** pour WPF, et **TsgcAvaloniaPivotGrid** et **TsgcAvaloniaPivotFieldList** pour Avalonia. Toutes dessinent avec le portage .NET de **sgcUICore**, le même canevas, les mêmes polices et les mêmes thèmes clair et sombre que les autres produits de la suite visuelle eSeGeCe, de sorte qu'un pivot a le même aspect sur chaque hôte et dans la sortie PNG et SVG sans interface.
Le moteur et son arithmétique sont ceux de Delphi, opération par opération : le magasin de colonnes typé, les huit agrégations, le regroupement de dates, les sous-totaux et totaux généraux, les filtres d'étiquette et de valeur, le tri, le développement et la réduction, le drill-down vers les lignes sources, les règles de format conditionnel et l'export CSV, JSON et HTML. La sortie SVG du moteur de rendu est identique octet pour octet à celle de Delphi pour la même scène, de sorte qu'un rapport construit sur un serveur Linux correspond à celui que dessine une application de bureau Delphi.

Packages

Le portage est découpé en packages NuGet, de sorte qu'un projet n'embarque que ce qu'il utilise.

PackageId	Contient	Frameworks cibles
esegece.sgcPivotGrid	Le moteur (**TsgcPivotEngine**) avec son magasin de colonnes, ses champs, agrégations, regroupement de dates, totaux, filtres, tri, drill-down et instantané (**TsgcPivotResult**), les règles de format, le moteur de rendu **TsgcPivotGridRenderer** avec la sortie SVG, le modèle de la liste de champs, les deux sources et l'export CSV, JSON et HTML. Dans l'édition Registered, aussi les champs calculés, afficher les valeurs comme, Top N et le JSON de disposition. Aucune dépendance native.	netstandard2.0, net472, net6.0, net8.0, net9.0
esegece.sgcPivotGrid.Rendering	La sortie PNG (**SaveToPNG**), via SkiaSharp.	netstandard2.0, net472, net6.0, net8.0, net9.0
esegece.sgcPivotGrid.WinForms	**TsgcWinFormsPivotGrid**, **TsgcWinFormsPivotFieldList** et sa fenêtre contextuelle des membres.	net472, net8.0-windows, net9.0-windows
esegece.sgcPivotGrid.Wpf	**TsgcWpfPivotGrid**, **TsgcWpfPivotFieldList** et sa fenêtre contextuelle des membres.	net472, net8.0-windows, net9.0-windows
esegece.sgcPivotGrid.Avalonia	**TsgcAvaloniaPivotGrid**, **TsgcAvaloniaPivotFieldList** et sa fenêtre contextuelle des membres, sur Avalonia 11.3.	net8.0, net9.0

Chaque package d'hôte dépend de esegece.sgcPivotGrid.Rendering et du package sgcUICore du même hôte, il suffit donc d'ajouter le package de l'hôte.

Éditions

Édition	Identifiants de package	Contient
Registered	esegece.sgcPivotGrid, .Rendering, .WinForms, .Wpf et .Avalonia	Tout : les trois grilles et listes de champs, le moteur, les deux sources, toutes les agrégations, le regroupement de dates, les totaux, les filtres, le tri, le drill-down, les règles de format, l'export, la sortie sans interface et les quatre fonctions avancées : champs calculés, afficher les valeurs comme, Top N avec Others et persistance de la disposition.
Community	Les mêmes identifiants avec le suffixe .Community, par exemple esegece.sgcPivotGrid.WinForms.Community.	Tout sauf les quatre fonctions avancées : **CalculatedFields**, les propriétés **ShowAs**, **ShowAsAxis** et **ShowAsBaseItem**, **TopN**, **TopNOthers** et **OthersCaption**, les méthodes **SaveLayout** et **LoadLayout**, ainsi que **GetExpandState** et **SetExpandState**.

C'est le même découpage que les éditions Delphi, voir Éditions.

Hôtes

Hôte	Équivalent Delphi	Plateformes
WinForms	TsgcVCLPivotGrid et TsgcVCLPivotFieldList	Windows, sous .NET Framework 4.7.2 et .NET 8 et 9.
WPF	TsgcFMXPivotGrid et TsgcFMXPivotFieldList	Windows, sous .NET Framework 4.7.2 et .NET 8 et 9.
Avalonia	TsgcFMXPivotGrid et TsgcFMXPivotFieldList	Windows, Linux et macOS, sous .NET 8 et 9.
Sans interface	TsgcPivotGridRenderer	Toute plateforme de .NET Standard 2.0, y compris les serveurs Linux.

Voir aussi

- Installation depuis NuGet
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Classes .NET
- Différences de .NET par rapport à Delphi
- Liaison de données .NET
- Rendu sans interface .NET
- Démos .NET

Installation depuis NuGet

Les packages sont sur nuget.org. Ajoutez le package de l'hôte ; le noyau et le package de rendu arrivent avec lui comme dépendances.

Édition Registered

```
dotnet add package esegece.sgcPivotGrid.WinForms
dotnet add package esegece.sgcPivotGrid.Wpf
dotnet add package esegece.sgcPivotGrid.Avalonia
```

Un serveur ou une application console sans fenêtre ajoute le noyau et, pour le PNG, le package de rendu :

```
dotnet add package esegece.sgcPivotGrid
dotnet add package esegece.sgcPivotGrid.Rendering
```

Édition Community

```
dotnet add package esegece.sgcPivotGrid.WinForms.Community
dotnet add package esegece.sgcPivotGrid.Wpf.Community
dotnet add package esegece.sgcPivotGrid.Avalonia.Community
```

Ou ajoutez la référence au fichier de projet à la main :

```
<ItemGroup>
  <PackageReference Include="esegece.sgcPivotGrid.WinForms" Version="2026.10.0" />
</ItemGroup>
```

Référencez dans un projet soit les packages Registered, soit les packages Community, jamais les deux : ils contiennent les mêmes types dans le même espace de noms. Le code qui utilise une fonction avancée ne compile pas avec les packages Community ; protégez-le avec `#if !SGC_COMMUNITY` lorsqu'un projet compile les deux éditions, comme le fait l'exemple de Différences de .NET par rapport à Delphi.

Paramètres du projet

- WinForms : un projet `net8.0-windows` avec `<UseWindowsForms>true</UseWindowsForms>`, ou un projet .NET Framework 4.7.2.
- WPF : un projet `net8.0-windows` avec `<UseWPF>true</UseWPF>`, ou un projet .NET Framework 4.7.2. Associez l'espace de noms en XAML avec `xmlns:pivot="clr-namespace:esegece.sgcPivotGrid;assembly=esegece.sgcPivotGrid.Wpf"`.
- Avalonia : un projet `net8.0` ou `net9.0` avec les packages **Avalonia.Desktop** et **Avalonia.Themes.Fluent** d'Avalonia 11.3, comme toute application de bureau Avalonia. Associez l'espace de noms en XAML avec `xmlns:pivot="using:esegece.sgcPivotGrid"`.
- Sans interface : esegece.sgcPivotGrid seul suffit pour le moteur, les deux sources, CSV, JSON, HTML et SVG. Ajoutez esegece.sgcPivotGrid.Rendering pour le PNG ; il apporte SkiaSharp et ses ressources natives.

En C#, chaque hôte a besoin de `using esegece.sgcPivotGrid;`, et de `using esegece.sgcUICore;` pour **TsgcUITheme**, **sgcUICore_Theme** et les autres types des thèmes.

Voir aussi

- Présentation de .NET
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Rendu sans interface .NET
- Démos .NET

Démarrage rapide WinForms

Cet exemple construit un formulaire WinForms avec une grille pivot et sa liste de champs sur 50 000 lignes de ventes générées : Region et Product en lignes, les années de la colonne Date en colonnes et la somme des ventes dans les cellules. Une case à cocher bascule la grille et la liste de champs entre le thème clair et le thème sombre, et un double-clic sur une cellule lit les lignes sources qui se trouvent derrière.

Étapes

- Créez un projet WinForms `net8.0-windows` et ajoutez le package **esegece.sgcPivotGrid.WinForms**, voir Installation depuis NuGet.
- Créez un **TsgcWinFormsPivotGrid** et ancrez-le. Sa propriété **Pivot** est le **TsgcPivotEngine** qui contient les lignes et la disposition.
- Chargez les lignes entre **BeginLoad** et **EndLoad** : **DefineColumn** nomme chaque colonne et fixe son type, **AppendRow** ajoute les valeurs dans l'ordre des colonnes. Le moteur recalcule une seule fois, au EndLoad le plus externe.
- Construisez la disposition entre **BeginUpdate** et **EndUpdate** : **AddField** place une colonne en lignes, en colonnes ou dans les filtres, avec un regroupement de dates facultatif, et **AddDataField** ajoute une mesure avec son agrégation.
- Créez une **TsgcWinFormsPivotFieldList** et affectez son **PivotGrid** à la grille. L'utilisateur glisse les champs entre les quatre zones et filtre les membres.

Exemple

Entrée ou un double-clic sur une cellule déclenche **OnDrillDown** ; **DrillDown** du moteur renvoie les lignes sources de la cellule sous forme de table qu'une grille sgcGrid ou un second pivot peut afficher, voir Liaison de données .NET. Un clic sur le bouton +/- d'un membre, ou un double-clic sur un en-tête de ligne, le développe et le réduit.

Voir aussi

- Installation depuis NuGet
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Classes .NET
- TsgcVCLPivotGrid
- Drill-down
- Démos .NET

Démarrage rapide WPF

Le même pivot sous WPF, construit en code : un **DockPanel** avec une case à cocher de thème en haut, la liste de champs à droite et la grille dans le reste de la fenêtre. La colonne Date va deux fois dans les colonnes, regroupée par année et par trimestre, et la zone des valeurs contient deux mesures de la même colonne, la somme et la moyenne des ventes. Un clic sur une cellule affiche son texte dans la barre de titre.

Étapes

- Créez un projet WPF `net8.0-windows` et ajoutez le package **esegece.sgcPivotGrid.Wpf**.
- Créez un **TsgcWpfPivotGrid** et une **TsgcWpfPivotFieldList**, ajoutez-les à la fenêtre et affectez **PivotGrid** de la liste de champs à la grille.
- Remplissez **Pivot** comme dans l'exemple WinForms. Le moteur, la disposition et les événements sont les mêmes sur chaque hôte.
- Lisez le résultat entre **Lock** et **Unlock** du moteur. **Result** est un instantané immuable, de sorte qu'un recalcul ne modifie jamais les cellules que vous lisez.

Exemple

La grille WPF dessine ses propres barres de défilement, comme la grille FireMonkey, et travaille en pixels indépendants de l'appareil : **HitTest** prend des DIP. Les options d'affichage sont de simples propriétés CLR, à définir en code ou comme attributs XAML.

Voir aussi

- Installation depuis NuGet
- Démarrage rapide WinForms
- Démarrage rapide Avalonia
- Classes .NET
- Différences de .NET par rapport à Delphi
- TsgcFMXPivotGrid

Démarrage rapide Avalonia

Le même pivot sous Avalonia, pour Windows, Linux et macOS : les quantités de quatre produits par région. Une case à cocher est liée à **AlternateRows** de la grille, et le titre suit la ligne sélectionnée.

Étapes

- Créez un projet de bureau Avalonia `net8.0` avec **Avalonia.Desktop** et **Avalonia.Themes.Fluent** 11.3, et ajoutez le package **esegece.sgcPivotGrid.Avalonia**.
- Ajoutez le **FluentTheme** aux styles de l'application : la fenêtre contextuelle des membres de la liste de champs utilise les modèles Fluent des cases à cocher et de la zone de recherche.
- Créez un **TsgcAvaloniaPivotGrid** et une **TsgcAvaloniaPivotFieldList**, affectez **PivotGrid** de la liste de champs à la grille et remplissez **Pivot** comme sur les autres hôtes.

Exemple

Chaque option d'affichage de la grille Avalonia est une propriété stylée, elle accepte donc les liaisons, les styles et les valeurs XAML, et **SelectedRow** et **SelectedCol** sont des propriétés directes bidirectionnelles. Sous macOS, la touche Cmd fonctionne comme Ctrl pour Ctrl+Début et Ctrl+Fin, et une molette horizontale fait défiler les colonnes.

Voir aussi

- Installation depuis NuGet
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Liaison de données .NET
- Différences de .NET par rapport à Delphi
- TsgcFMXPivotGrid

Classes .NET

Toutes les classes se trouvent dans l'espace de noms **esegece.sgcPivotGrid**. Les grilles et les listes de champs sont des hôtes minces au-dessus du moteur, du moteur de rendu et du modèle de liste de champs indépendants du framework, partagés par les hôtes, la sortie sans interface et les sources.

Classes d'hôte

Hôte	Classe	Classe de base	Delphi
WinForms	TsgcWinFormsPivotGrid	TsgcUIWinFormsControl	TsgcVCLPivotGrid
WinForms	TsgcWinFormsPivotFieldList	TsgcUIWinFormsControl	TsgcVCLPivotFieldList
WPF	TsgcWpfPivotGrid	TsgcUIWpfControl	TsgcFMXPivotGrid
WPF	TsgcWpfPivotFieldList	TsgcUIWpfControl	TsgcFMXPivotFieldList
Avalonia	TsgcAvaloniaPivotGrid	TsgcUIAvaloniaControl	TsgcFMXPivotGrid
Avalonia	TsgcAvaloniaPivotFieldList	TsgcUIAvaloniaControl	TsgcFMXPivotFieldList

La liste à cocher des membres de chaque liste de champs est **TsgcWinFormsPivotMemberPopup** (un Form), **TsgcWpfPivotMemberPopup** (une Window) ou **TsgcAvaloniaPivotMemberPopup** (un Popup). Les trois grilles implémentent **IsgcPivotEngineHost**, pour qu'une source ou une liste de champs trouve leur moteur, et **IsgcUITable**, pour qu'un graphique sgcCharts ou un autre pivot puisse lire leur résultat.

Membres de la grille

Les trois grilles partagent les mêmes membres.

Kind	Membres
Moteur et dessin	**Pivot** (le **TsgcPivotEngine**), **Renderer**, **FormatRules**.
Options d'affichage	**RowLayout**, **RowHeight**, **HeaderHeight**, **IndentWidth**, **MinColumnWidth**, **MaxColumnWidth**, **AutoColumnWidths**, **GridLines**, **AlternateRows**, **TotalsBold**, **ShowExpandButtons**, **Theme**, **AccentColor**, **FontSize**.
Position	**TopRow**, **ScrollX**, **SelectedRow**, **SelectedCol**, **ScrollToCell**, **HitTest**.
Sortie	**Recalculate**, **SaveToPNG** et **SaveToSVG** (un fichier ou un flux, avec largeur et hauteur facultatives ; 0 correspond à la vue entière), **SaveToCSV**, **SaveToJSON**, **SaveToHTML**. Édition Registered : **SaveLayoutToString**, **LoadLayoutFromString**, **SaveLayoutToFile**, **LoadLayoutFromFile**.
Événements	**OnCellClick**, **OnCellDblClick**, **OnDrillDown**, **OnExpandChange**, **OnGetCellStyle**, **OnGetCellText**, **OnSelectionChange**, **OnColumnResized**.

Les listes de champs ont **PivotGrid** (tout **IsgcPivotEngineHost**), **Engine**, **Theme**, **AccentColor**, **FontSize**, **ShowFilterArea**, **DeferUpdate** avec **ApplyChanges**, et les événements **OnBeforeFieldMove** et **OnFieldMoved**.

Types indépendants du framework

Type	Description
TsgcPivotEngine	Le magasin de colonnes, la disposition, le calcul, l'instantané, le drill-down et l'export : **BeginLoad**, **DefineColumn**, **AppendRow**, **EndLoad**, **LoadFromTable**, **AddField**, **AddDataField**, **MoveField**, **RemoveField**, **ClearLayout**, **ExpandAll**, **CollapseAll**, **Lock**, **Unlock**, **Result**, **ResultTable**, **GetSourceRows**, **DrillDown**, **GetMemberLabels** et les événements **OnCalculated**, **OnLayoutChange** et **OnFilterRow**. Voir TsgcPivotEngine.
TsgcPivotFields, TsgcPivotField	Un champ de la disposition : **FieldName**, **Caption**, **Area**, **AreaIndex**, **DateGrouping**, **Aggregation**, **DisplayFormat**, **SortBy**, **SortOrder**, **SortMeasure**, **Subtotals**, **Expanded**, **FilterValues**, **FilterExclude**, **LabelFilter**, **ValueFilter** et, dans l'édition Registered, **TopN**, **TopNOthers**, **ShowAs**, **ShowAsAxis** et **ShowAsBaseItem**. Voir TsgcPivotField.
TsgcPivotCalculatedFields, TsgcPivotCalculatedField	Colonnes numériques calculées par ligne source à partir d'une expression. Édition Registered. Voir TsgcPivotCalculatedField.
TsgcPivotResult, TsgcPivotItem	L'instantané immuable d'un recalcul : **ItemCount**, **Item**, **NodeLabel**, **NodePath**, **LevelCount**, **MeasureCount**, **DataColCount**, **CellText**, **CellValue** et **RawValue**. Un **TsgcPivotItem** est un struct avec **Kind** (`pvkMember` , `pvkTotal` , `pvkGrandTotal`), **Level**, **Node**, **Expandable** et **Expanded**.
TsgcPivotResultTable, TsgcPivotSourceView	Le résultat d'un moteur et les lignes sources d'un drill-down, tous deux sous forme de **IsgcUITable**.
TsgcPivotSource_DataSet	Charge une `System.Data.DataTable` et la recharge quand la table change. Voir Liaison de données .NET.
TsgcPivotSource_Table	Charge n'importe quelle **IsgcUITable** et la recharge quand sa version change. Voir TsgcPivotSource_Table.
TsgcPivotGridRenderer, TsgcPivotRenderOptions, TsgcPivotRenderState	Dessine un instantané sur un **TsgcUICanvas**, effectue les tests de sélection et l'écrit en SVG ou, avec esegece.sgcPivotGrid.Rendering, en PNG. Voir Rendu sans interface .NET.
TsgcPivotFormatRules, TsgcPivotFormatRule	Mise en forme conditionnelle : règles de valeur et échelles de couleurs. Voir TsgcPivotFormatRule.
TsgcPivotFieldListModel	La liste de champs sans fenêtre : les zones, les puces, le glisser-déposer et l'état des filtres, dessinés par les trois listes de champs.
sgcPivotGrid_IO	Export statique CSV, JSON et HTML d'un résultat, avec les arguments par défaut de Delphi. Le moteur et les grilles l'enveloppent dans **SaveToCSV**, **SaveToJSON** et **SaveToHTML**.

Énumérations

Type	Valeurs
TsgcPivotArea	`pvaRow` , `pvaColumn` , `pvaData` , `pvaFilter`
TsgcPivotAggregation	`pvfSum` , `pvfCount` , `pvfAverage` , `pvfMin` , `pvfMax` , `pvfCountDistinct` , `pvfStdDev` , `pvfVar`
TsgcPivotDateGrouping	`pvdNone` , `pvdYear` , `pvdQuarter` , `pvdMonth` , `pvdWeek` , `pvdDay`

TsgcPivotDataType	`pvtText`, `pvtInteger`, `pvtFloat`, `pvtDateTime`, `pvtBoolean`
TsgcPivotAxisKind	`pvxRows`, `pvxColumns`
TsgcPivotSortBy, TsgcPivotSortOrder	`pvsLabel`, `pvsValue`; `pvoAscending`, `pvoDescending`
TsgcPivotRowLayout	`pvlCompact`, `pvlTabular`
TsgcPivotShowAs	`pvwValue`, `pvwPercentOfRow`, `pvwPercentOfColumn`, `pvwPercentOfGrandTotal`, `pvwRunningTotal`, `pvwDifferenceFrom`, `pvwRank`. Édition Registered.

Voir aussi

- Présentation de .NET
- Différences de .NET par rapport à Delphi
- Liaison de données .NET
- Rendu sans interface .NET
- Architecture

Différences de .NET par rapport à Delphi

Le portage .NET conserve les noms, les valeurs par défaut, l'arithmétique et les règles de rendu de Delphi. Sa sortie SVG, ainsi que l'export CSV, JSON et HTML de données ordinaires, sont identiques octet pour octet à ceux de Delphi. Les différences ci-dessous viennent des frameworks hôtes et de la bibliothèque de classes .NET.

Pas de streaming de composants

Il n'y a pas de DFM en .NET, donc pas d'éditeurs de collection en conception, pas de liste déroulante pour FieldName et pas de verbes **Recalculate** et **Clear Layout**. Construisez la disposition en code avec **AddField** et **AddDataField**, entre **BeginUpdate** et **EndUpdate**, pour que le moteur ne recalcule qu'une fois. Le moteur Delphi diffère le travail pendant le chargement d'une fiche ; en .NET, c'est le rôle de **BeginUpdate** et **BeginLoad**. Pour conserver une disposition entre deux sessions, utilisez le JSON de disposition de l'édition Registered, voir Persistance de la disposition.

Propriétés indexées, événements et listes

- Une propriété indexée Delphi est une méthode : `Result.Item(axis, i)`, `NodeLabel(axis, node)`, `SourceColumnName(i)` et, sur les fenêtres contextuelles des membres, `Members(i)` et `MemberChecked(i)` avec `SetMemberChecked(i, value)`. Une fonction Delphi reste une méthode : `SourceRowCount()`, `MeasureCount()`, `DataColCount()`. Les collections gardent leur indexeur, comme `Fields[0]`.
- Les événements sont des propriétés de type délégué, nommées comme en Delphi (**OnCalculated**, **OnDrillDown**...). Affectez un gestionnaire avec `=` ou chaînez-en plusieurs avec `+=`. Un paramètre `var` de Delphi est un paramètre `ref` (`aHandled`, `aAccept`, `aAllow`), et une lambda avec un paramètre `ref` déclare le type de chaque paramètre. **TNotifyEvent** est **TsgcUINotifyEvent**.
- Un paramètre **TStrings** est une `IList<string>`. **FilterValues** est un **TsgcPivotFilterValues**, une liste de chaînes ordonnée dont **IndexOf** ignore la casse, comme TStringList. Le filtre lui-même compare les étiquettes des membres en respectant la casse.
- **Assign** d'un champ, d'un filtre, d'une règle ou d'un champ calculé depuis une autre classe lève `InvalidCastException`, l'EConvertError de Delphi.

Valeurs, dates et cultures

AppendRow prend des valeurs CLR : null et DBNull sont des cellules vides, les types entiers sont `pvtInteger`, `float`, `double` et `decimal` sont `pvtFloat`, `DateTime` est `pvtDateTime`, `bool` est `pvtBoolean`, et `string` et `char` sont du texte. Les dates sont stockées comme dates OLE, le TDateTime de Delphi, de sorte que le regroupement de dates et les numéros de semaine correspondent à ceux de Delphi. Le texte est analysé uniquement avec la culture invariante, alors que Delphi essaie aussi les paramètres régionaux de l'utilisateur, et les étiquettes des membres, **CellText** et les exports formatent les nombres et les dates avec des paramètres invariants (point comme séparateur décimal, dates au format MM/dd/yyyy), jamais avec la culture du thread.

DataTable au lieu de TDataSet

TsgcPivotSource_DataSet lie une `System.Data.DataTable`, l'équivalent .NET d'un TDataSet Delphi. Une DataTable est toujours ouverte et n'a pas de curseur, il n'y a donc ni enregistrement courant, ni signet, ni DisableControls. Les événements `RowChanged`, `RowDeleted` et `TableCleared` remplacent le data link de Delphi ; les modifications inachevées et les nouvelles lignes détachées sont ignorées, comme Delphi ignore les modifications et insertions non validées. Voir Liaison de données .NET.

Pas de notification de libération

Quand **PivotGrid** d'une source ou d'une liste de champs est un `IComponent`, comme la grille WinForms, la source écoute son événement `Disposed` et efface alors **PivotGrid**. Les grilles WPF et Avalonia ne sont pas des composants : affectez null à **PivotGrid** avant de supprimer la grille. **Engine** n'est jamais surveillé ; effacez-le avant que le

moteur disparaisse. **TsgcPivotSource_DataSet** et **TsgcPivotSource_Table** sont `IDisposable` ; **Dispose** retire leurs écouteurs de la table.

Threads

Le calcul est synchrone, comme en Delphi. Un recalcul s'exécute sur le thread qui le déclenche et y publie le nouvel instantané ; les grilles se redessinent ensuite sur leur thread d'interface, via un message de fenêtre posté sous WinForms, `Dispatcher.BeginInvoke` sous WPF et `Dispatcher.UIThread.Post` sous Avalonia. Le code qui lit **Result** depuis un autre thread tient **Lock** et **Unlock** du moteur.

Barres de défilement, unités et propriétés par hôte

- **TsgcWinFormsPivotGrid** utilise les barres de défilement natives de Windows et des pixels, comme la grille VCL. **TsgcWpfPivotGrid** et **TsgcAvaloniaPivotGrid** dessinent leurs propres barres de défilement, avec glissement du curseur et défilement par page sur la piste, comme la grille FireMonkey, et travaillent en pixels indépendants de l'appareil.
- Le **BorderStyle** de Delphi devient **BorderThickness** (1 par défaut) et **BorderBrush** sous WPF et Avalonia. **TsgcWpfPivotGrid** déclare son propre **FontSize** avec `new`, et **TsgcAvaloniaPivotGrid** son propre **Theme** de type **TsgcUITheme**, qui masque le thème de contrôle d'Avalonia. Via une référence typée `Control`, ce sont les propriétés du framework, que la grille ne lit pas.
- Les options WPF sont des propriétés CLR. Les options Avalonia sont des propriétés stylées qui acceptent les liaisons et les styles, voir Liaison de données .NET.
- La fenêtre contextuelle des membres de la liste de champs est un Form sous WinForms, une Window sans bordure appartenant à la fenêtre de la liste de champs sous WPF, et un Popup à fermeture par clic extérieur sous Avalonia. Sous macOS, la touche Cmd fonctionne comme Ctrl.

Voir aussi

- Présentation de .NET
- Classes .NET
- Liaison de données .NET
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- TsgcPivotEngine

Liaison de données .NET

Le moteur garde sa propre copie typée des lignes sources, de sorte qu'un pivot ne lit jamais la source de données pendant qu'il calcule ou dessine. Il y a quatre façons de le remplir :

Méthode	À utiliser pour
BeginLoad, **DefineColumn**, **AppendRow**, **EndLoad**	Des lignes que votre code produit ou lit depuis un lecteur quelconque. Chaque démarrage rapide l'utilise.
TsgcPivotSource_DataSet	Une `System.Data.DataTable`, rechargée quand ses lignes changent.
TsgcPivotSource_Table	Toute **IsgcUITable** : les grilles et le magasin de lignes de sgcGrid .NET, un autre moteur pivot ou son résultat, rechargé quand la version de la table change.
LoadFromTable	Une copie unique d'une **IsgcUITable**, comme les lignes sources d'un drill-down.

Une source remplit le moteur d'une grille quand **PivotGrid** est défini, ou un moteur sans interface quand **Engine** est défini ; **Engine** l'emporte si les deux sont définis. Chaque chargement remplace les lignes dans un seul bloc de chargement, de sorte que le moteur recalcule une fois et conserve sa disposition.

DataTable

L'exemple affiche `4 source rows, 3 columns` : la colonne Notes n'est pas dans **Columns**, et la ligne ajoutée après la liaison a rechargé le moteur. Le type d'une colonne de DataTable fixe le type de la colonne du pivot :

Type de colonne de la DataTable	Type de colonne du pivot
`byte` , `short` , `int` , `long` et les autres types entiers	`pvtInteger`
`float` , `double` , `decimal`	`pvtFloat`
`DateTime` , `DateTimeOffset` , `TimeSpan`	`pvtDateTime`
`bool`	`pvtBoolean`
Colonnes binaires (`byte[]` et les autres tableaux sauf `char[]`)	Jamais chargées.
Tout le reste	`pvtText` , avec la culture invariante.

Avec **LiveUpdates**, une modification validée recharge le moteur : une ligne ajoutée, modifiée, annulée ou supprimée, et une table vidée. Une ligne en cours de modification charge ses valeurs actuelles, de sorte qu'un chargement ne valide jamais une modification en attente, et une nouvelle ligne est ignorée tant qu'elle n'est pas ajoutée à la table. **Load** charge à tout moment et **Refresh** recharge quand **Active** vaut true.

IsgcUITable

PivotOfCell construit un second pivot sur les lignes qui se trouvent derrière une cellule, pour une fenêtre de détail. **Follow** construit un pivot d'un pivot : un moteur est une **IsgcUITable** de son résultat, avec une colonne par champ de ligne et une par colonne de données, nommée d'après les membres de colonne et la mesure, par exemple `Online | Sum of Sales`. Chaque recalcul du premier moteur change la version de la table et recharge le second. La

même propriété **Table** accepte une grille sgcGrid .NET, dont l'utilisateur modifie les lignes, comme le fait la démo GridToPivotToChart.

Lier les options d'affichage

Sous Avalonia, chaque option d'affichage de la grille (**RowLayout**, **RowHeight**, **HeaderHeight**, **IndentWidth**, **MinColumnWidth**, **MaxColumnWidth**, **AutoColumnWidths**, **GridLines**, **AlternateRows**, **TotalsBold**, **ShowExpandButtons**, **Theme**, **AccentColor**, **FontSize**, **BorderThickness**, **BorderBrush**) est une propriété stylée, et **SelectedRow** et **SelectedCol** sont des propriétés directes bidirectionnelles. Elles acceptent les liaisons, les styles et les valeurs XAML, voir Démarrage rapide Avalonia. Une modification faite en code sur **Renderer.Options** est recopiée vers les propriétés. La liste de champs Avalonia a aussi **Theme**, **AccentColor**, **FontSize**, **ShowFilterArea** et **DeferUpdate** comme propriétés stylées.
Sous WPF et WinForms, les options sont des propriétés CLR : définissez-les en code, ou comme attributs XAML sous WPF.

Voir aussi

- Présentation de .NET
- Classes .NET
- Différences de .NET par rapport à Delphi
- TsgcPivotSource_DataSet
- TsgcPivotSource_Table
- Interopérabilité avec les graphiques

Rendu sans interface .NET

Le moteur et le moteur de rendu n'ont pas besoin de fenêtre, de sorte qu'un service, une application web ou une tâche console sous Windows, Linux ou macOS peut construire un pivot et l'écrire en PNG, SVG, CSV, JSON ou HTML. Cet exemple charge 10 000 lignes de ventes, garde le canal en ligne avec un champ de filtre, affiche le résultat, lit les lignes sources derrière une cellule et écrit les cinq fichiers.

Exemple

La sortie de la console :

```
Region                2024          2025   Grand Total
East            145,862.61    155,105.09    300,967.70
North           162,608.24    155,080.86    317,689.10
South           154,901.83    155,794.10    310,695.93
West            156,547.53    157,665.06    314,212.59
Grand Total     619,920.21    623,645.11  1,243,565.32
The first cell adds up 580 source rows
```

Remarques

- Une cellule est adressée par un élément de ligne et une colonne de données. Avec plusieurs mesures, les colonnes de données d'un élément de colonne sont côte à côte : la colonne de données est l'élément de colonne multiplié par **MeasureCount**, plus la mesure.
- **TsgcPivotRenderState** porte la position de défilement, la sélection et les largeurs de colonne ; `sgcPivotGrid_Render.sgcPivotRenderState()` renvoie l'état par défaut. Une largeur et une hauteur de 0, la valeur par défaut, dessinent la vue entière au lieu d'un seul écran.
- **SaveToSVG** fait partie de esegece.sgcPivotGrid et n'a pas de dépendance native. **SaveToPNG** est une méthode d'extension de esegece.sgcPivotGrid.Rendering, qui dessine avec SkiaSharp.
- Les grilles ont les mêmes méthodes : `Grid.SaveToPNG(fileName)` écrit la vue entière de la grille avec ses options et ses règles de format.
- L'export CSV écrit les valeurs brutes avec leur précision complète ; l'export HTML écrit les textes de cellule formatés.

Voir aussi

- Présentation de .NET
- Classes .NET
- TsgcPivotGridRenderer
- Export
- Démarrage rapide sans interface
- Démos .NET

Démos .NET

Les démos se trouvent dans `.net\demos\PivotGrid`, un dossier par démo avec son projet, son `Program.cs` et un README.md. Chaque démo a un commutateur `--test` qui exécute une auto-vérification hors écran, écrit des lignes PASS et FAIL et se termine avec le nombre de contrôles échoués, comme le commutateur `/test` des démos Delphi.

Démo	Hôte	Montre
SalesPivot	WinForms	100 000 lignes de ventes générées avec une liste de champs : Region et Country en lignes, la colonne Date en Year, Quarter et Month en colonnes, le canal en filtre, tri par valeur, une échelle de couleurs, développement et réduction, et export CSV, HTML et PNG.
SalesPivotWpf	WPF	Le même pivot sur **TsgcWpfPivotGrid**.
SalesPivotAvalonia	Avalonia	Le même pivot sur **TsgcAvaloniaPivotGrid**, pour Windows, Linux et macOS.
GridToPivotToChart	WinForms	Une grille sgcGrid .NET, modifiable dans l'édition Registered, alimente le pivot via **TsgcPivotSource_Table**, et le résultat du pivot alimente un graphique sgcCharts .NET via **IsgcUITable**.
MillionRows	WinForms	Un million de lignes générées avec trois mesures (la somme, le comptage distinct de 100 000 clients et l'écart type), avec le temps du chargement et de chaque recalcul.
HeadlessReport	Console	Lit un fichier CSV, ou écrit d'abord un orders.csv d'exemple de 5 000 commandes générées, et écrit le pivot en PNG et SVG avec **TsgcPivotGridRenderer**, et en HTML, CSV et JSON avec le moteur, sans fenêtre. Fonctionne sous Windows, Linux et macOS.

Compiler les démos

Dans l'arborescence du produit, les démos se compilent contre les projets de la bibliothèque. Le zip des démos utilise par défaut les packages NuGet de l'édition Community. Aucune n'utilise de fonction avancée, donc chaque démo compile et s'exécute dans les deux éditions. Compilez une démo avec les packages Registered en définissant **SgcPivotGridEdition** :

```
dotnet build SalesPivot\SalesPivot.csproj -p:SgcPivotGridEdition=Registered
dotnet run --project HeadlessReport\HeadlessReport.csproj -- --test
```

GridToPivotToChart restaure aussi les packages sgcGrid et sgcCharts de la même édition.

Voir aussi

- Présentation de .NET
- Installation depuis NuGet
- Démarrage rapide WinForms
- Rendu sans interface .NET
- Démos

Dépannage

La grille reste vide

Vérifiez que des données sont arrivées : SourceRowCount du moteur vaut 0 lorsque la source n'est pas Active ou n'a ni DataSet, ni Table, ni PivotGrid. Vérifiez les noms : un champ de ligne, de colonne ou de filtre dont le FieldName ne correspond à aucune colonne source est ignoré, et un champ de données dont le nom est inconnu reste vide. Les noms sont comparés sans distinction de casse, et SourceColumnName liste les colonnes chargées. Avec AutoRecalculate à False, appelez Recalculate. Un filtre qui ne laisse aucune ligne source donne aussi un tableau croisé vide.

Le tableau croisé ne suit pas le jeu de données

TsgcPivotSource_DataSet recharge tant que Active et LiveUpdates sont True, lorsque le jeu de données signale une modification des données en état de navigation : un post, un append, un delete, un refresh, First ou Last. Next, Prior et MoveBy ne rechargent pas, et une modification ne recharge qu'après son post.

Le chargement est lent

AppendRow en dehors d'un bloc BeginLoad et EndLoad recalcule après chaque ligne. Chargez dans un bloc, voir Performances.

EsgcPivotError: The pivot layout needs more than ... cells

La mise en page crée plus de cellules que MaxCells, généralement parce qu'un champ à nombreux membres se trouve sur les lignes et un autre sur les colonnes. Regroupez les dates, déplacez un champ vers les filtres, utilisez Top N ou augmentez MaxCells.

EsgcPivotError: Calculated field ...

L'expression de ce champ calculé comporte une erreur de syntaxe, ou des champs calculés se référencent mutuellement en cercle. Le message nomme le champ. Un nom de champ qui n'est pas encore chargé n'est pas une erreur, le champ calculé reste vide jusqu'à l'arrivée de la colonne. Voir Champs calculés.

EsgcPivotError: Invalid pivot layout

LoadLayoutFromString ou LoadLayoutFromFile a reçu un JSON mal formé, ou une racine qui n'est pas un objet. La mise en page reste inchangée.

Undeclared identifier CalculatedFields, TopN, ShowAs ou SaveLayoutToString

L'édition Community omet les champs calculés, afficher les valeurs en tant que, Top N et la persistance de la mise en page, ces membres n'y existent donc pas, ainsi que OthersCaption, TopNOthers, GetExpandState et SetExpandState. Une fiche qui enregistre ces propriétés ne peut pas être chargée par une compilation Community. Voir Éditions.

Les cellules vides n'affichent rien

Définissez EmptyText du moteur, par exemple sur un tiret. Les grilles, le moteur de rendu, l'export HTML et ResultTable l'affichent, et OnGetCellText le reçoit et peut encore le remplacer. SaveToCSV écrit un champ vide et SaveToJSON écrit null, de sorte que les données exportées restent numériques.

Un total n'est pas la somme des cellules

Les totaux sont calculés à partir des lignes source : une moyenne est la moyenne des lignes, et un comptage distinct compte chaque valeur une fois. Après un filtre de valeur ou Top N, les totaux n'incluent que les membres restants. Un champ calculé est évalué par ligne puis agrégé. Voir Agrégations et Totaux.

Les dates ne se regroupent pas, ou les nombres se trient comme du texte

DateGrouping ne fonctionne que sur une colonne de dates, et un tri par libellé ne compare les nombres par valeur que dans une colonne numérique. Une colonne dont la première valeur était une chaîne est une colonne de texte. Passez des valeurs TDateTime et numériques à AppendRow, ou fixez le type avec DefineColumn(aName, aType) avant la première ligne.

Une vue de drill-down est vide, ou une violation d'accès à l'arrêt

Un TsgcPivotSourceView n'a plus de lignes après la modification des données source, et il lit le magasin de son moteur, libérez-le donc avant le moteur. Voir Drill-down.

Valeurs erronées ou exception lorsqu'un autre thread lit Result

Ne lisez Result qu'entre Lock et Unlock du moteur. Un recalcul publie un nouvel instantané et libère le précédent.

C++Builder: no matching constructor for TsgcPivotEngine ou TsgcPivotGridRenderer

Les en-têtes générés ne conservent pas les arguments par défaut de ces constructeurs. Passez NULL explicitement : `new TsgcPivotEngine(NULL)` et `new TsgcPivotGridRenderer(NULL, NULL)`.

TsgcFMXPivotGrid est absent ou vide

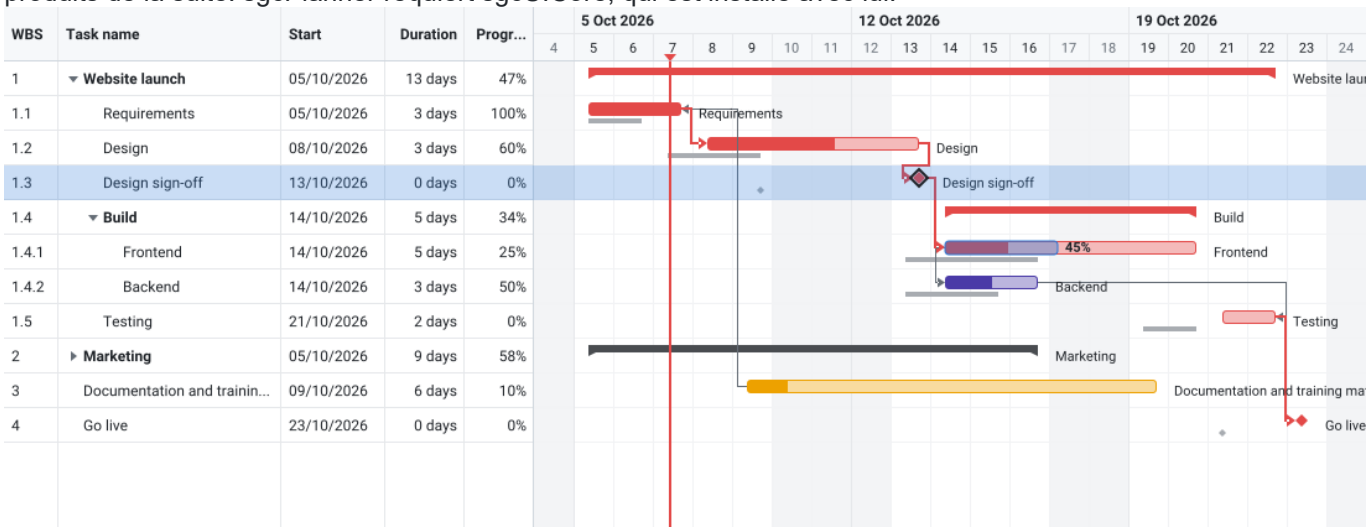
TsgcFMXPivotGrid et TsgcFMXPivotFieldList existent à partir de Delphi XE6, où sgcVer.inc définit SGC_PIVOTGRID_FMX. Avec un compilateur plus ancien, l'unité sgcPivotGrid_FMX compile sans eux. À la conception, ils se trouvent sur la page SGC PivotGrid de la palette tant qu'une fiche FireMonkey est ouverte. Voir Plateformes.

La liste de champs n'affiche aucun champ

Définissez PivotGrid de la liste de champs, ou Engine pour un moteur sans grille. La liste affiche les colonnes source du moteur, elle se remplit donc lorsque les données sont chargées.

Introduction

sgcPlanner ajoute la planification et la gestion de projet aux applications Delphi et C++Builder, pour VCL et FireMonkey. Un seul objet de données, **TsgcPlannerData**, contient les éléments, les ressources, les statuts, les dépendances et le calendrier de travail, et trois types de vues l'affichent en même temps : **TsgcVCLScheduler** et **TsgcFMXScheduler** (jour, semaine de travail, semaine, mois, agenda et une Timeline de ressources), **TsgcVCLGantt** et **TsgcFMXGantt** (une arborescence de tâches avec des barres, des dépendances, la planification automatique, le chemin critique et les lignes de base) et **TsgcVCLKanban** et **TsgcFMXKanban** (un tableau avec des colonnes de statut, des couloirs et des limites WIP). Faites glisser une carte vers une autre colonne et le Gantt et le Scheduler affichent aussitôt le nouveau statut.
Les données proviennent de votre code, d'un **TDataSet** via **TsgcPlannerSource_DataSet**, d'un fichier JSON, d'un fichier iCalendar (.ics) via **TsgcPlannerICalendar** ou d'un fichier XML Microsoft Project via **TsgcPlannerMSProject**. **TsgcPlannerRenderer** dessine n'importe quelle vue en PNG et en SVG sans fenêtre, sur un serveur ou sous Linux64.
sgcPlanner fait partie de la suite visuelle eSeGeCe. Il est dessiné avec **sgcUICore**, le package d'exécution partagé qui contient le canevas, le rastériseur, les polices, l'écrivain SVG et les thèmes clair et sombre. Un planificateur a donc le même aspect sous VCL, sous FireMonkey et dans la sortie sans fenêtre, et il s'accorde avec les autres produits de la suite. sgcPlanner requiert sgcUICore, qui est installé avec lui.



Éditions

sgcPlanner est livré en éditions Registered, Trial et Community. L'édition Community conserve les trois vues, le glisser-déposer, les récapitulatifs, les jalons, la progression, la source DataSet, le JSON et la sortie sans fenêtre, et laisse de côté la récurrence, la planification automatique avec flèches de dépendance, le chemin critique, les lignes de base, la Timeline de ressources, les couloirs, les limites WIP, iCalendar et le XML MS Project. Voir Éditions.

Par où commencer

- Architecture : l'objet de données, les notifications de changement, les vues et les plateformes.
- Démarrage rapide VCL, Démarrage rapide FMX et Démarrage rapide C++Builder : trois vues synchronisées en quelques minutes.
- TsgcPlannerData : les éléments, les ressources, les statuts, les dépendances et le calendrier de travail.
- Les vues : TsgcVCLScheduler, TsgcVCLGantt, TsgcVCLKanban et leurs jumeaux FireMonkey.
- Les composants non visuels : TsgcPlannerSource_DataSet, TsgcPlannerICalendar, TsgcPlannerMSProject et TsgcPlannerRenderer.
- Installation et Plateformes : les packages, les dossiers et les IDE et plateformes pris en charge.
- Démos : les huit démos Delphi et ce que montre chacune.
- Les fonctionnalités : Données partagées, Vues synchronisées, Calendrier de travail, Récurrence, Échelle de temps et zoom, Glisser-déposer, Dépendances, Planification automatique, Chemin critique, Lignes de base, Couloirs et limites WIP, Liaison DataSet, Persistance JSON, iCalendar, XML MS Project, Rendu sans fenêtre, Thèmes et Fonctionnalités par édition.

- Dépannage : les problèmes courants et leurs causes.

Architecture

sgcPlanner sépare les données des vues, et les vues du framework. L'objet de données, les moteurs et les modèles de vue n'utilisent que la RTL et sgcUICore. Les contrôles VCL et FMX sont de fins hôtes autour d'un modèle de vue, et le renderer sans fenêtre utilise les mêmes modèles de vue, de sorte qu'un PNG correspond à ce que montre une fiche pixel pour pixel.

Flux de données

1. **Sources.** Votre code remplit directement TsgcPlannerData, ou TsgcPlannerSource_DataSet le charge depuis des TDataSet et réécrit les modifications. LoadFromJSON lit un projet enregistré, TsgcPlannerICalendar importe et exporte des fichiers .ics et TsgcPlannerMSProject importe et exporte du XML Microsoft Project.
2. **Données.** TsgcPlannerData est un TComponent en cinq parties : Items, Resources, Statuses, Dependencies et Calendar. Il tient un index par Id, la structure parent et enfants, la sélection et un numéro Version qui augmente à chaque changement.
3. **Moteurs.** Le calendrier de travail compte des minutes de travail, le moteur de récurrence développe les règles RRULE en occurrences et le moteur de planification place les tâches après leurs prédécesseurs, calcule la marge et marque le chemin critique. Les deux derniers font partie de l'édition Registered.
4. **Modèles de vue.** Chaque vue s'enregistre comme écouteur des données. Lors d'un changement, elle ne reconstruit que ce dont elle a besoin : un changement de sélection redessine une surcouche, un changement d'élément remet en page à l'image suivante, un changement de structure reconstruit les lignes, les colonnes et les couloirs.
5. **Sortie.** Les modèles de vue dessinent sur un TsgcUICanvas de sgcUICore. Les six contrôles les affichent dans une fiche avec défilement, souris, clavier et saisie tactile, et TsgcPlannerRenderer les dessine en PNG et en SVG.

Classes

Classe	Unité	Rôle
TsgcPlannerData	sgcPlanner_Data	Les données partagées : éléments, ressources, statuts, dépendances, calendrier, sélection, notifications et JSON.
TsgcPlannerItem	sgcPlanner_Data	Une tâche, une carte ou un événement : dates, statut, progression, parent, ressources, étiquettes, récurrence, contrainte et ligne de base.
TsgcPlannerCalendar	sgcPlanner_Calendar	Les jours ouvrés, les heures de travail, les jours fériés et l'arithmétique des minutes de travail.
TsgcPlannerTimeScale	sgcPlanner_TimeScale	L'axe du temps du Gantt et de la Timeline : zoom, largeur des graduations, graduations, bandes non travaillées et magnétisme.
TsgcVCLScheduler, TsgcFMXScheduler	sgcPlanner_Scheduler_VCL, sgcPlanner_Scheduler_FMX	Les vues calendrier : jour, semaine de travail, semaine, mois, agenda et la Timeline de ressources.
TsgcVCLGantt, TsgcFMXGantt	sgcPlanner_Gantt_VCL, sgcPlanner_Gantt_FMX	L'arborescence des tâches et le diagramme à barres avec dépendances, chemin critique et lignes de base.

TsgcVCLKanban, TsgcFMX-Kanban	sgcPlanner_Kanban_VCL, sgcPlanner_Kanban_FMX	Le tableau : colonnes de statut, couloirs, limites WIP et cartes.
TsgcPlannerSource_DataSet	sgcPlanner_Source_DataSet	Charge les données depuis des TDataSet, suit leurs changements et réécrit les modifications.
TsgcPlannerICalendar	sgcPlanner_ICal	Import et export VEVENT RFC 5545. Édition Registered.
TsgcPlannerMSProject	sgcPlanner_MSProject	Import et export XML Microsoft Project (MSPDI). Édition Registered.
TsgcPlannerRenderer	sgcPlanner_Renderer	Dessine une vue Scheduler, Gantt ou Kanban sans fenêtre, sur un canevas, dans un PNG ou dans un SVG.
TsgcPlannerScheduleEngine	sgcPlanner_AutoSchedule	Passage aller et retour, marge, chemin critique, lignes de base et détection des cycles. Édition Registered.

Notifications de changement

- Chaque setter des données notifie. Entourez un lot de changements de BeginUpdate et EndUpdate : les écouteurs et OnChange sont déclenchés une seule fois, après le EndUpdate le plus externe, avec les types de changement (pckItems, pckStructure, pckResources, pckStatuses, pckDependencies, pckCalendar, pckSelection, pckSchedule), l'Id de l'élément quand un seul élément a changé, et la Version.
- Les vues sont des écouteurs et ne font que lire les données. Votre gestionnaire OnChange peut modifier de nouveau les données : les changements imbriqués arrivent comme un tour supplémentaire après le tour courant, jusqu'à 8 tours, puis EsgcPlannerError est levée.
- Une modification faite à la souris déclenche OnItemChanging des données une fois par dépôt, avec les champs sur le point de changer, de sorte que l'application peut la refuser. La modification est ensuite validée en une seule mise à jour, donc un dépôt donne une seule notification à chaque vue liée.
- Deux vues ou plus sur les mêmes données restent en phase : avec SyncSelection, la sélection, qui est le SelectedId des données, est partagée elle aussi.

Temps

Le modèle conserve ses heures sous forme de TsgcPlannerTime, un nombre de minutes en Int64, de sorte que l'arithmétique est exacte et que les secondes sont abandonnées. Les propriétés publiées sont des TDateTime, et chacune a un jumeau TsgcPlannerTime : Start et StartTime, Finish et FinishTime, ProjectStart et ProjectStartTime. sgcPlannerTimeOf et sgcPlannerDateTimeOf de sgcPlanner_DateUtils convertissent de l'un à l'autre. Le Finish d'un élément est exclusif : un élément sur toute la journée du 5 octobre se termine à minuit le 6 octobre. Les vues lisent l'heure courante dans la variable de fonction sgcPlannerNow, qu'un test peut remplacer pour figer l'horloge.

Plateformes

Cible	Prise en charge
Delphi VCL	Delphi 7 à Delphi 13, Win32 et Win64.
Delphi Fire-Monkey	Delphi XE6 à Delphi 13 (SGC_PLANNER_FMX), toutes les plateformes prises en charge par FireMonkey.
Linux64	Delphi 10.2 et ultérieur pour les unités indépendantes du framework (SGC_PLANNER_LINUX) : les données, les moteurs, JSON, iCalendar, XML MS Project et le renderer sans fenêtre.

C++Builder	C++Builder 2007 à C++Builder 13, avec les mêmes composants. Voir Démarrage rapide C++Builder.

Delphi 7 et Delphi 2007 ont des chaînes Ansi, donc les composants iCalendar et XML MS Project convertissent le texte UTF-8 des fichiers vers la page de codes du système, et les caractères hors de cette page de codes sont perdus.

Performances

Ce sont des mesures de builds release Win64 pendant le développement, à 1920x1080 et à l'échelle 1. Win32 est environ 1,5 à 1,7 fois plus lent.

Scénario	Win64
Planification automatique de 10 000 tâches (1 000 récapitulatifs, 3 niveaux) avec 15 000 liens et 30 jours fériés, chemin critique compris	11 ms
Image du Gantt pendant le défilement de 10 000 tâches avec 15 000 liens	11 ms
Mise en page Kanban de 5 000 cartes dans 8 colonnes et 6 couloirs	5 ms
Vue semaine du Scheduler avec 2 000 éléments horodatés, mise en page	1 ms
Chargement de 10 000 lignes avec TsgcPlannerSource_DataSet	26 ms
Enregistrement et chargement JSON de 10 000 éléments avec 15 000 liens	48 ms et 56 ms
Import de 5 000 VEVENT avec fuseaux horaires depuis un fichier .ics	84 ms
Import de 10 000 tâches avec 15 000 liens depuis du XML MS Project	264 ms

Voir aussi

- Éditions
- TsgcPlannerData
- TsgcPlannerRenderer
- Plateformes
- Données partagées
- Vues synchronisées

Éditions

sgcPlanner est construit à partir des mêmes sources pour toutes les éditions.

Édition	Contient
Registered	Tout : les trois vues pour VCL et FMX, l'objet de données, le calendrier de travail, le glisser-déposer, les récapitulatifs, les jalons, la progression, la source DataSet, le JSON, la sortie PNG et SVG sans fenêtre, et les fonctionnalités avancées listées ci-dessous. La page de palette SGC Planner affiche 11 composants.
Trial	Les mêmes fonctionnalités que Registered. Un build compilé avec SGC_TRIAL affiche un message indiquant qu'il s'agit d'une version d'essai quand un composant est créé après la fin de la fenêtre de publication nommée dans le build. Rien d'autre ne change.
Community	Tout sauf les fonctionnalités avancées : les vues Scheduler jour, semaine de travail, semaine, mois et agenda avec regroupement par ressource, le Gantt avec son arborescence de tâches, ses récapitulatifs, ses jalons et sa progression, le tableau Kanban avec des colonnes de statut, le glisser-déposer dans toutes les vues, la source DataSet, le JSON et le renderer. La palette affiche 9 composants, sans TsgcPlannerICalendar ni TsgcPlannerMSProject, et il n'y a pas d'avis d'essai.

Les fonctionnalités avancées

Les fonctionnalités avancées sont compilées sous la directive SGC_PLANNER_ADVANCED, que sgcVer.inc désactive dans l'édition Community. Six unités ne font pas partie de la distribution Community : sgcPlanner_Recurrence, sgcPlanner_AutoSchedule, sgcPlanner_Scheduler_Timeline, sgcPlanner_XML, sgcPlanner_ICal et sgcPlanner_MSProject. Les propriétés des vues sont publiées dans toutes les éditions, donc dans Community elles sont conservées mais n'ont aucun effet :

Fonctionnalité	Dans l'édition Community
Récurrence	GetOccurrences retourne un élément récurrent une seule fois, à son propre Start. RecurrenceRule et RecurrenceExceptions sont toujours stockés et enregistrés. OnRecurrenceEdit et OnRecurrenceDelete du Scheduler ne sont jamais déclenchés.
Dépendances et planification automatique	Les liens sont stockés et enregistrés, mais le Gantt ne dessine aucune flèche et n'a pas de glissement pour lier. ShowDependencies et AutoSchedule n'ont aucun effet, et Schedule et SetBaseline du Gantt lèvent EsgcPlannerError.
Chemin critique et lignes de base	ShowCriticalPath et ShowBaseline n'ont aucun effet. BaselineStart et BaselineFinish sont seulement stockés.
Timeline de ressources	La vue psvTimeline du Scheduler dessine un avis centré.
Couloirs et limites WIP	SwimlaneBy se comporte comme pslNone. WIPLimit n'affiche aucun badge et ne bloque jamais une carte.
iCalendar et XML MS Project	TsgcPlannerICalendar et TsgcPlannerMSProject n'existent pas.

L'édition est un choix à la compilation dans sgcVer.inc. SGC_COMMUNITY désactive aussi SGC_TRIAL. Le code qui utilise TsgcPlannerICalendar, TsgcPlannerMSProject ou une autre classe des six unités, et qui doit aussi compiler avec Community, le place dans {$IFDEF SGC_PLANNER_ADVANCED}. Une fiche qui contient l'un de ces deux composants ne se charge pas dans un build Community.

Voir aussi

- Architecture
- Introduction
- Fonctionnalités par édition
- Installation

Installation

sgcPlanner fait partie de la distribution sgcUI. Un seul programme d'installation couvre tous les produits sgcUI, et il n'installe que les produits inclus dans votre licence.

Installation avec le programme d'installation sgcUI

1. Lancez le programme d'installation sgcUI et connectez-vous avec le nom d'utilisateur et le mot de passe de votre compte eSeGeCe.
2. Choisissez les versions de Delphi et de C++Builder dans lesquelles installer.
3. Sur la page des composants, sélectionnez **sgcPlanner** et les autres produits souhaités. Les produits que votre licence n'inclut pas sont grisés. Une licence All-Access débloque tous les produits.
4. Le programme d'installation copie les fichiers, compile les packages sur votre machine pour chaque IDE sélectionné dans l'ordre des dépendances, et installe le package de conception. Les composants apparaissent sur la page de palette **SGC Planner**.

Ce qui accompagne sgcPlanner

sgcUICore, la base partagée des produits visuels, est installé automatiquement avec sgcPlanner. Il n'a pas de case à cocher propre.

Dossiers installés

Chaque produit s'installe dans un seul dossier, par défaut `Documents\esegece\sgcUI` (`sgcUI_Trial` pour l'essai et `sgcUI_Community` pour l'édition Community).

Dossier	Contenu
`Source`	Le code source de chaque produit installé, sgcPlanner et sgcUICore compris. C'est le chemin de bibliothèque unique de sgcUI.
`Packages`	Les packages de chaque produit installé, un jeu par produit et par version d'IDE.
`Lib<V>`	Les unités compilées pour chaque version d'IDE et chaque plateforme, par exemple `LibD13` .
`Demos\Planner`	Les démos de sgcPlanner. Voir Démos.

Le programme d'installation ajoute `sgcUI\Source` au chemin de bibliothèque et au chemin de navigation de chaque IDE et plateforme sélectionnés, avec le dossier des unités compilées.

Trial et Community

- **Trial** : le programme d'installation d'essai de sgcUI installe sgcPlanner en mode d'essai, avec toutes les fonctionnalités.
- **Community** : le programme d'installation gratuit sgcUI Community inclut sgcPlanner sans les fonctionnalités avancées (récurrence, planification automatique, flèches de dépendance, chemin critique, lignes de base, Timeline de ressources, couloirs, limites WIP, iCalendar et XML MS Project). Voir Éditions.

Compiler les packages manuellement

Pour compiler vous-même les packages, ouvrez-les pour votre version d'IDE depuis `sgcUI\Packages`. `<V>` est le suffixe de version de Delphi, par exemple `sgcPlannerD13.dpk`.

Package	Contenu
`sgcUICore<V>`	La base partagée sgcUICore (exécution).
`sgcPlanner<V>`	Le package d'exécution de sgcPlanner : les données, les moteurs, les vues, les contrôles VCL et, à partir de Delphi XE6, les contrôles FireMonkey. Il requiert rtl, vcl, dbrtl, fmx (XE6 et ultérieur) et sgcUICore.
`dclsgcPlanner<V>`	Le package de conception qui enregistre les composants sur la page de palette SGC Planner.

Compilez-les dans l'ordre des dépendances : `sgcUICore`, puis `sgcPlanner`, puis `dclsgcPlanner`. Installez ensuite `dclsgcPlanner` et ajoutez `sgcUI\Source` au chemin de bibliothèque de votre version de Delphi (Outils > Options > Langage > Options Delphi > Bibliothèque > Chemin de bibliothèque). Pour C++Builder, ouvrez le projet `.cbproj` de votre version, par exemple `sgcPlannerC13.cbproj`, et compilez-le. Les en-têtes `.hpp` générés apparaissent à côté du package compilé.

Voir aussi

- Plateformes
- Éditions
- Démarrage rapide VCL

Plateformes

sgcPlanner est une seule arborescence de sources pour tous les IDE et toutes les plateformes. Les données, les moteurs et les modèles de vue n'utilisent que la RTL et sgcUICore, donc c'est le même code sur chaque cible ; seuls les six contrôles sont liés à un framework.

Cible	Prise en charge
Delphi VCL	Delphi 7 à Delphi 13, Win32 et Win64 : TsgcVCLScheduler, TsgcVCLGantt et TsgcVCLKanban.
Delphi FireMonkey	Delphi XE6 à Delphi 13 pour Windows, macOS, iOS et Android : TsgcFMXScheduler, TsgcFMXGantt et TsgcFMXKanban. Compilés sous SGC_PLANNER_FMX.
Linux64	Delphi 10.2 et ultérieur, compilé sous SGC_PLANNER_LINUX : les données, le calendrier de travail, les moteurs, JSON, iCalendar, XML MS Project et TsgcPlannerRenderer, de sorte qu'un serveur peut produire des fichiers PNG et SVG d'un plan.
C++Builder	C++Builder 2007 à C++Builder 13 avec les mêmes composants et les en-têtes .hpp générés. Voir Démarrage rapide C++Builder.
.NET	Un portage en C# avec la même API pour WinForms, WPF et Avalonia, le Scheduler, le Gantt et le tableau Kanban sur chaque hôte et la sortie PNG et SVG sans fenêtre, sur .NET Framework 4.7.2 et .NET 8 et 9, voir Vue d'ensemble de .NET.

Unités par framework

Unités	Framework
sgcPlanner_Const, sgcPlanner_DateUtils, sgcPlanner_Calendar, sgcPlanner_Data, sgcPlanner_TimeScale, sgcPlanner_Recurrence, sgcPlanner_AutoSchedule, sgcPlanner_JSON, sgcPlanner_XML, sgcPlanner_ICal, sgcPlanner_MSProject	Aucun. Le modèle, les moteurs et les formats de fichier.
sgcPlanner_View, sgcPlanner_Scheduler, sgcPlanner_Scheduler_Render, sgcPlanner_Scheduler_Timeline, sgcPlanner_Gantt, sgcPlanner_Gantt_Render, sgcPlanner_Kanban, sgcPlanner_Kanban_Render, sgcPlanner_Renderer	Aucun. Les modèles de vue et leurs routines de dessin, qui dessinent sur un TsgcUICanvas de sgcUICore.
sgcPlanner_Source_DataSet	Seulement l'unité DB (dbrtl), aucune unité VCL ou FireMonkey.
sgcPlanner_VCL, sgcPlanner_Scheduler_VCL, sgcPlanner_Gantt_VCL, sgcPlanner_Kanban_VCL	VCL, Windows uniquement.
sgcPlanner_FMX, sgcPlanner_Scheduler_FMX, sgcPlanner_Gantt_FMX, sgcPlanner_Kanban_FMX	FireMonkey, Delphi XE6 et ultérieur.

Packages

sgcPlanner fournit un package d'exécution indépendant du framework et un package d'exécution par framework, de sorte qu'une application FireMonkey pour Android, iOS ou macOS ne lie jamais la VCL. Chaque nom de pa-

ckage se termine par le suffixe de l'IDE, par exemple sgcPlannerD13.dpk pour Delphi 13 et sgcPlannerC13.cbproj pour C++Builder 13.

Package	Type	Contenu	Requiert
sgcPlanner	Exécution, indépendant du framework	Chaque unité du produit qui n'utilise ni la VCL ni FireMonkey : les données, le calendrier de travail, les moteurs, les modèles de vue et leurs peintres, JSON, iCalendar, XML MS Project, TsgcPlannerRenderer et TsgcPlannerSource_DataSet.	rtl, dbrtl, sgcUICore
sgcPlannerVCL	Exécution, VCL	sgcPlanner_VCL, sgcPlanner_Scheduler_VCL, sgcPlanner_Gantt_VCL et sgcPlanner_Kanban_VCL (TsgcVCLScheduler, TsgcVCLGantt, TsgcVCLKanban)	rtl, vcl, sgcUICore, sgcUICoreVCL, sgcPlanner
sgcPlannerFMX	Exécution, FireMonkey, à partir de XE6	sgcPlanner_FMX, sgcPlanner_Scheduler_FMX, sgcPlanner_Gantt_FMX et sgcPlanner_Kanban_FMX (TsgcFMXScheduler, TsgcFMXGantt, TsgcFMXKanban)	rtl, fmx, sgcUICore, sgcUICoreFMX, sgcPlanner
dclsgcPlanner	Conception, VCL	Enregistre TsgcPlannerData, TsgcVCLScheduler, TsgcVCLGantt, TsgcVCLKanban, TsgcPlannerRenderer, TsgcPlannerSource_DataSet et, hors de l'édition Community, TsgcPlannerICalendar et TsgcPlannerMSProject dans la page de palette **SGC Planner**.	designide, rtl, vcl, sgcPlanner, sgcPlannerVCL
dclsgcPlannerFMX	Conception, FireMonkey, à partir de XE6	Enregistre TsgcFMXScheduler, TsgcFMXGantt et TsgcFMXKanban dans la page de palette **SGC Planner**.	designide, rtl, fmx, sgcPlanner, sgcPlannerFMX, dclsgcPlanner
sgcPlannerWS, dclsgcPlannerWS	Exécution et conception, passerelle sgcWebSockets	TsgcPlannerGoogleSync et, hors de l'édition Community, TsgcPlannerICalSubscription, indépendants du framework, dans la page de palette **SGC Planner**.	rtl, sgcPlanner, sgcWebSockets

sgcUICore suit la même répartition : sgcUICore, sgcUICoreVCL et sgcUICoreFMX, sans package de conception. dclsgcPlannerFMX requiert dclsgcPlanner, installez donc d'abord dclsgcPlanner.

Versions de l'IDE et plateformes cibles

Delphi	sgcPlanner	sgcPlannerVCL	sgcPlannerFMX	dclsgcPlanner, dclsgcPlannerFMX
7, 2007, 2009, 2010, XE	Win32	Win32	Aucune	Win32
XE2 à XE5	Win32, Win64	Win32, Win64	Aucune	Win32
XE6 à 10.2	Win32, Win64	Win32, Win64	Win32, Win64	Win32
10.3	Win32, Win64, Linux64, OSX64	Win32, Win64	Win32, Win64, OSX64	Win32

10.4	Win32, Win64, Linux64, OSX64, Android, Android64, iOSDevice64	Win32, Win64	Win32, Win64, OSX64, Android, Android64, iOSDevice64	Win32
11	Win32, Win64, Linux64, OSX64, OSXARM64, Android, Android64, iOSDevice64, iOSSimARM64	Win32, Win64	Win32, Win64, OSX64, OSXARM64, Android, Android64, iOSDevice64, iOSSimARM64	Win32
12, 13	Comme 11	Win32, Win64	Comme 11	Win32, Win64

De XE6 à 10.2, le package FireMonkey est compilé pour Windows uniquement. Les packages de conception sont chargés par l'IDE, ils sont donc compilés pour la plateforme de l'IDE uniquement. sgcPlannerWS est compilé pour les mêmes plateformes que sgcPlanner, et dclsgcPlannerWS pour les mêmes plateformes que dclsgcPlanner.

C++Builder	sgcPlanner, sgcPlannerVCL	sgcPlannerFMX	dclsgcPlanner, dclsgcPlannerFMX
2007, 2010, XE, XE2	Win32	Aucune	Win32 (dclsgcPlanner uniquement)
XE3 à XE5	Win32, Win64	Aucune	Win32 (dclsgcPlanner uniquement)
XE6 à 12	Win32, Win64	Win32, Win64	Win32
13	Win32, Win64, Win64x	Win32, Win64, Win64x	Win32, Win64x

Les projets C++Builder portent les mêmes noms avec le suffixe C, par exemple sgcPlannerC13.cbproj, sgcPlannerVCLC13.cbproj, sgcPlannerFMXC13.cbproj, dclsgcPlannerC13.cbproj et dclsgcPlannerFMXC13.cbproj. Les packages C++Builder sont compilés pour Windows uniquement. sgcPlannerWS et dclsgcPlannerWS sont compilés pour les mêmes versions de C++Builder et les mêmes plateformes que sgcPlanner et dclsgcPlanner, sauf Win64x, car ils nécessitent une version Win64x du package sgcWebSockets. Les notes C++Builder qui s'appliquent à chaque produit sgcUI (packages d'exécution sous Win64, C++Builder 10.3 sous Win64, noms de fichiers des packages Win64x) se trouvent dans le sujet Plateformes et packages de l'aide de la suite sgcUI.

FireMonkey sur Android, iOS et macOS

Les contrôles FireMonkey (TsgcFMXScheduler, TsgcFMXGantt et TsgcFMXKanban) ciblent macOS à partir de Delphi 10.3, Android et iOS à partir de Delphi 10.4, et macOS ARM et le simulateur iOS à partir de Delphi 11. Dans un projet FireMonkey pour ces plateformes :

- Utilisez les contrôles FireMonkey. Les contrôles VCL et sgcPlannerVCL ne fonctionnent que sous Windows.
- Déposez le contrôle sur un formulaire, ou ajoutez son unité à la clause uses : sgcPlanner_Scheduler_FMX, sgcPlanner_Gantt_FMX ou sgcPlanner_Kanban_FMX.
- Sélectionnez la plateforme cible dans la fenêtre Projets. L'installateur ajoute le dossier Source de sgcUI et les unités compilées de chaque plateforme (LibD<version>\<platform>) au chemin de bibliothèque de chaque plateforme qu'il compile. Sans l'installateur, ajoutez le dossier Source de sgcUI au chemin de bibliothèque de la plateforme cible (Outils, Options, Langage, Delphi, Bibliothèque).
- Les applications Android et iOS sont liées statiquement, aucun package d'exécution n'est donc déployé.

Linux64

Le package indépendant du framework sgcPlanner est compilé pour Linux64 à partir de Delphi 10.3. Une application Linux64, par exemple un programme console ou un serveur, utilise les données, le calendrier de travail, les moteurs, JSON, iCalendar, XML MS Project et TsgcPlannerRenderer, par exemple pour produire des fichiers PNG

et SVG d'un planning. Aucun package VCL ou FireMonkey n'est compilé pour Linux64, car FireMonkey sous Linux nécessite FMX pour Linux (FMXLinux), un module complémentaire tiers.

Notes par plateforme

- **Haute résolution.** Les tailles comme RowHeight, TreeWidth et SlotHeight sont en DIP. Les contrôles VCL les mettent à l'échelle avec le PPI du moniteur, les contrôles FireMonkey avec l'échelle de la fiche, et le renderer avec sa propriété Scale.
- **Tactile.** Les contrôles traitent la saisie tactile comme une saisie à la souris, donc un doigt fait glisser une carte, une barre ou un événement. Les contrôles VCL gardent leurs propres gestes de panoramique, de sorte qu'une boîte de défilement parente ne prend pas la main sur un glissement.
- **Locale.** Les noms des jours et des mois, les formats de date et d'heure et le séparateur décimal proviennent de sgcPlannerLocale, initialisé à partir de la machine. Un serveur Linux ou une machine dans une autre langue affiche donc d'autres libellés. Définissez les propriétés de sgcPlannerLocale au démarrage pour obtenir la même sortie partout.
- **Delphi 7 et Delphi 2007.** Ces versions ont des chaînes Ansi, donc les lecteurs JSON, iCalendar et XML MS Project convertissent le texte UTF-8 vers la page de codes du système, et les caractères hors de celle-ci sont perdus.
- **Éditions.** SGC_PLANNER_ADVANCED, que sgcVer.inc désactive dans l'édition Community, décide des fonctionnalités avancées sur chaque plateforme. Voir Fonctionnalités par édition.

Voir aussi

- Architecture
- Installation
- Rendu sans fenêtre

Démarrage rapide VCL

Installez les packages sgcPlanner pour votre version de Delphi. Les composants se trouvent sur la page de palette **SGC Planner**.

À la conception

1. Déposez un **TsgcPlannerData** sur une fiche. Ajoutez des statuts à sa collection Statuses (par exemple to-do, doing et done), des personnes à Resources et des tâches à Items : un Id, un Title, un Start, un Finish et un Status pour chacune.
2. Déposez un **TsgcVCLGantt**, définissez Align sur alTop et sa propriété Data sur l'objet de données.
3. Déposez un **TsgcVCLKanban**, définissez Align sur alRight et définissez Data. Le tableau affiche une colonne par statut.
4. Déposez un **TsgcVCLScheduler**, définissez Align sur alClient, définissez Data et choisissez une View, par exemple psvWorkWeek.
5. Exécutez l'application. Faites glisser une carte vers une autre colonne, une barre dans le Gantt ou un événement dans le Scheduler : les trois vues affichent aussitôt le changement, parce qu'elles partagent les données.

Dans le code

Remplissez les données entre BeginUpdate et EndUpdate, afin que chaque vue reçoive une seule notification. Le Finish d'un élément est exclusif.

```
uses
  SysUtils, Controls, Forms,
  sgcPlanner_Data, sgcPlanner_Scheduler, sgcPlanner_Gantt_VCL,
  sgcPlanner_Kanban_VCL, sgcPlanner_Scheduler_VCL;

function AddTask(aData: TsgcPlannerData; const aId, aTitle: string;
  aStart, aFinish: TDateTime; const aStatus, aOwner: string): TsgcPlannerItem;
begin
  Result := aData.AddItem(aId);
  Result.Title := aTitle;
  Result.Start := aStart;
  Result.Finish := aFinish; // the finish is exclusive
  Result.Status := aStatus;
  Result.Resources := aOwner;
end;

procedure LoadProject(aData: TsgcPlannerData);
var
  vMonday: TDateTime;
begin
  vMonday := EncodeDate(2026, 10, 5);
  // one notification for the whole load
  aData.BeginUpdate;
  try
    aData.Statuses.AddStatus('todo', 'To do');
    aData.Statuses.AddStatus('doing', 'In progress');
    aData.Statuses.AddStatus('done', 'Done');
    aData.Resources.AddResource('ana', 'Ana Lopez');
    aData.Resources.AddResource('ben', 'Ben Carter');
    AddTask(aData, 'design', 'Design', vMonday + EncodeTime(8, 0, 0, 0),
      vMonday + 2 + EncodeTime(17, 0, 0, 0), 'doing', 'ana').Progress := 50;
    AddTask(aData, 'build', 'Build', vMonday + 3 + EncodeTime(8, 0, 0, 0),
      vMonday + 7 + EncodeTime(17, 0, 0, 0), 'todo', 'ben');
    AddTask(aData, 'review', 'Design review', vMonday + 2 + EncodeTime(14, 0,
      0, 0), vMonday + 2 + EncodeTime(15, 0, 0, 0), 'todo', 'ana,ben');
    // Build starts when Design finishes
```

```
    aData.Dependencies.AddDependency('design', 'build');
  finally
    aData.EndUpdate;
  end;
end;

procedure CreatePlanner(aForm: TForm);
var
  oData: TsgcPlannerData;
  oGantt: TsgcVCLGantt;
  oKanban: TsgcVCLKanban;
  oScheduler: TsgcVCLScheduler;
begin
  oData := TsgcPlannerData.Create(aForm);
  LoadProject(oData);

  // the three views share one data object
  oGantt := TsgcVCLGantt.Create(aForm);
  oGantt.Parent := aForm;
  oGantt.Align := alTop;
  oGantt.Height := 300;
  oGantt.Data := oData;

  oKanban := TsgcVCLKanban.Create(aForm);
  oKanban.Parent := aForm;
  oKanban.Align := alRight;
  oKanban.Width := 560;
  oKanban.Data := oData;

  oScheduler := TsgcVCLScheduler.Create(aForm);
  oScheduler.Parent := aForm;
  oScheduler.Align := alClient;
  oScheduler.View := psvWorkWeek;
  oScheduler.Date := EncodeDate(2026, 10, 5);
  oScheduler.Data := oData;
end;
```

Voir aussi

- TsgcPlannerData
- TsgcVCLGantt
- TsgcVCLKanban
- TsgcVCLScheduler
- Démos
- Installation

Démarrage rapide FMX

Les composants FireMonkey requièrent Delphi XE6 ou ultérieur. Ils se trouvent sur la page de palette **SGC Planner** d'une fiche FireMonkey, avec les composants non visuels.

À la conception

1. Déposez un **TsgcPlannerData** et remplissez-le comme dans Démarrage rapide VCL.
2. Déposez un **TsgcFMXGantt**, un **TsgcFMXKanban** et un **TsgcFMXScheduler**, définissez leur Align et définissez la propriété Data de chacun sur l'objet de données.

Les contrôles FMX dessinent leurs propres barres de défilement, défilent avec la molette de la souris et traitent la saisie tactile comme une saisie à la souris, donc un doigt fait glisser une carte ou une barre. Leurs propriétés, méthodes et événements sont les mêmes que sous VCL.

Dans le code

```
uses
  System.SysUtils, FMX.Types, FMX.Controls, FMX.Forms,
  sgcPlanner_Data, sgcPlanner_Scheduler, sgcPlanner_Gantt_FMX,
  sgcPlanner_Kanban_FMX, sgcPlanner_Scheduler_FMX;

procedure CreatePlanner(aForm: TForm; aData: TsgcPlannerData);
var
  oGantt: TsgcFMXGantt;
  oKanban: TsgcFMXKanban;
  oScheduler: TsgcFMXScheduler;
begin
  // the three views share one data object, filled as in the VCL quick start
  oGantt := TsgcFMXGantt.Create(aForm);
  oGantt.Parent := aForm;
  oGantt.Align := TAlignLayout.Top;
  oGantt.Height := 300;
  oGantt.Data := aData;

  oKanban := TsgcFMXKanban.Create(aForm);
  oKanban.Parent := aForm;
  oKanban.Align := TAlignLayout.Right;
  oKanban.Width := 560;
  oKanban.Data := aData;

  oScheduler := TsgcFMXScheduler.Create(aForm);
  oScheduler.Parent := aForm;
  oScheduler.Align := TAlignLayout.Client;
  oScheduler.View := psvWorkWeek;
  oScheduler.Data := aData;
end;
```

Voir aussi

- TsgcFMXScheduler
- TsgcFMXGantt
- TsgcFMXKanban
- Démos
- Plateformes

Démarrage rapide C++Builder

Installez les packages sgcPlanner pour votre version de C++Builder, ajoutez `sgcUI\Source` et le dossier des unités compilées, par exemple `sgcUI\LibD13\win32` (`sgcUI\LibD13\win64` pour 64 bits), aux chemins d'inclusion et de bibliothèque du projet, et utilisez les composants de la page de palette **SGC Planner** exactement comme en Delphi. Le programme d'installation ajoute les deux chemins à l'IDE pour vous. Les en-têtes .hpp et les fichiers .lib générés de sgcPlanner et de sgcUICore sont dans ce seul dossier, donc sgcUICore n'a pas besoin de chemin propre.

À savoir en C++

- Les constructeurs n'ont pas d'arguments par défaut. Créez un composant non visuel sans propriétaire avec NULL : `new TsgcPlannerRenderer(NULL)`.
- Les exemples de cette aide passent explicitement tous les arguments d'une méthode, par exemple `AddStatus(Id, Caption, 0)` et `AddDependency(FromId, ToId, pdkFS, 0)`, de sorte qu'ils compilent avec n'importe quelle version des en-têtes générés.
- Les événements sont des closures, donc leurs gestionnaires sont des fonctions membres __fastcall d'une classe. Un paramètre var tel que aAccept est une référence bool, et un paramètre const string est un `const String`.
- Les énumérations ne sont pas à portée limitée : écrivez `psvWorkWeek`, `prvGantt` ou `pwmBlock`.

Dans le code

Voir aussi

- TsgcPlannerData
- Démarrage rapide VCL
- TsgcPlannerRenderer
- Démos
- Plateformes

Démos

Les démos se trouvent dans `sgcUI\Demos\Planner\Delphi`, un dossier par démo avec un `readme.html` qui donne les détails. La plupart des démos construisent leurs données avec un petit générateur déterministe autour de la semaine du 5 octobre 2026.

Démo	Composants	Ce qu'elle montre
ProjectPlanner	TsgcVCLGantt, TsgcVCLKanban, TsgcVCLScheduler	Le relancement d'un site web en trois vues liées à un seul TsgcPlannerData. Faites glisser une carte de To do vers Done et la colonne de statut du Gantt, la couleur de la barre et la couleur de l'événement du Scheduler suivent. Couleurs de statut avec BarColorMode pbcStatus, et une sélection partagée via SelectedId.
TeamScheduler	TsgcVCLScheduler	Trois semaines de réunions pour cinq personnes dans les six vues (jour, semaine de travail, semaine, mois, agenda et la Timeline de ressources). Un point quotidien et une synchronisation hebdomadaire issus d'une RecurrenceRule, la création par glissement avec OnItemCreate et le regroupement avec Options.GroupByResource.
ProjectGantt	TsgcVCLGantt, TsgcPlannerMSProject	Un plan de livraison logicielle avec trois phases WBS construites à partir de ParentId, le glissement pour lier avec OnLinkCreating, Schedule et AutoSchedule sur le calendrier de travail, le chemin critique, une ligne de base qui montre de combien une barre déplacée a glissé, et l'ouverture et l'enregistrement de XML Microsoft Project. Édition Registered.
TaskBoard	TsgcVCLKanban	Un backlog de 26 cartes avec des largeurs de colonne et des limites WIP dans Columns, les modes WIP d'avertissement et de blocage, des couloirs par ressource, par parent ou par étiquette, la navigation au clavier et une colonne Done repliable.
DataSetPlanner	TsgcVCLGantt, TsgcVCLKanban, TsgcPlannerSource_DataSet	Trois tables TClientDataSet (tâches, personnes et liens) avec leurs propres noms de champs, mappées avec ItemFields, ResourceFields et DependencyFields. Un enregistrement validé dans le TDBGrid met à jour son élément, une carte ou une barre déplacée dans une vue est écrite dans la table, et les écritures attendent pendant que la grille modifie un enregistrement. MidasLib est liée, donc aucune midas.dll n'est nécessaire.
ICalendarSync	TsgcVCLScheduler, TsgcPlannerICalendar	Export en .ics avec la RRULE d'une série, des éléments sur toute la journée sous forme de valeurs DATE et des heures UTC ou flottantes avec ExportUTC. Import depuis Google Calendar, Outlook ou tout fichier RFC 5545 avec les fuseaux horaires convertis à l'heure locale, remplacement ou fusion avec ImportMode, et les avertissements de OnImportWarning. Édition Registered.
GoogleCalendarSync	TsgcVCLScheduler, TsgcPlannerGoogleSync	Une vue semaine en synchronisation bidirectionnelle avec un agenda Google via TsgcHTTPGoogleCloud_Calendar_Client de sgcWebSockets : l'ID client et le secret OAuth2 saisis à l'exécution et jamais stockés, Direction, un choix pour OnConflict, Synchroniser maintenant avec SyncAsync, synchronisation automatique avec AutoSyncInterval et un journal de OnSyncDone, OnConflict et OnError. Édition Registered, nécessite sgcWebSockets.

ICalSubscription	TsgcVCLScheduler, TsgcPlannerICalSubscription	Une URL iCalendar fusionnée à côté de deux éléments locaux : URL, intervalle d'actualisation, Actualiser avec RefreshAsync, actualisation automatique avec Active, événements du flux en lecture seule via OnItemChanging et IsSubscriptionItem, et un journal de OnRefreshed, OnError et OnImportWarning. Édition Registered, nécessite sgcWebSockets.
PlannerFMX	TsgcFMXGantt, TsgcFMXScheduler, TsgcFMXKanban, TsgcPlannerRenderer	Les trois vues FireMonkey sur un seul objet de données pour Windows, macOS, iOS et Android : une carte, une barre ou un événement déplacé dans une vue se déplace dans les deux autres. Une barre d'outils avec la vue Scheduler, le zoom et l'ajustement du Gantt, Selected to Done et un thème sombre, et chaque vue rendue sans fenêtre avec les options de son contrôle.
HeadlessReport	TsgcPlannerRenderer	Une application console sans unité VCL ni FireMonkey, pour Windows et Linux64. Elle charge un projet avec sgcPlannerJSONLoadFromFile, ou écrit d'abord le projet d'exemple sous project.json, et enregistre les pages scheduler, gantt et kanban en PNG et en SVG, après ScrollToTime, ExpandAll et ZoomToFit sur les vues que possède le renderer. Utilisation : `HeadlessReport [project.json] [output folder]`.

Commutateurs de ligne de commande

Chaque démo a deux commutateurs utilisés par la vérification de build. `/snapshot <file.png>` enregistre une image de la démo et se termine : les démos de vues VCL rendent leurs vues avec TsgcPlannerRenderer, DataSetPlanner, ICalendarSync et PlannerFMX enregistrent la fenêtre (un BMP sous Delphi 7), et HeadlessReport écrit la page Gantt du projet d'exemple. `/test [file.txt]`, ou `--test`, exécute un autotest du comportement principal de la démo, écrit une ligne PASS ou FAIL par contrôle et se termine avec le nombre d'échecs. Les démos de vues pilotent leurs contrôles avec de vrais messages de souris et de clavier.
Les démos VCL se compilent avec Delphi 7 à Delphi 13, Win32 et Win64. PlannerFMX se compile avec Delphi XE6 à Delphi 13.

Voir aussi

- Démarrage rapide VCL
- Démarrage rapide FMX
- Installation

TsgcPlannerData

Unité : sgcPlanner_Data
TsgcPlannerData est le composant de données partagé de sgcPlanner, un composant non visuel. Il contient les éléments, les ressources, les statuts, les dépendances et le calendrier de travail, et il notifie chaque vue qui lui est liée. Un Scheduler, un Gantt, un Kanban, une source DataSet, les composants iCalendar et MS Project et le renderer peuvent tous utiliser en même temps le même objet de données.

Propriétés

Propriété	Type	Description
Items	TsgcPlannerItems	Les tâches, les cartes et les événements. Voir Items ci-dessous.
Resources	TsgcPlannerResources	Les personnes ou les salles : Id, Name, Color et Initials (calculées à partir de Name si vide). AddResource(aId, aName) et Find(aId).
Statuses	TsgcPlannerStatuses	Les états du workflow : Id, Caption et Color. Ce sont les colonnes par défaut du Kanban. AddStatus(aId, aCaption, aColor = 0) et Find(aId).
Dependencies	TsgcPlannerDependencies	Les liens entre les éléments : FromId, ToId, Kind (pdkFS par défaut, pdkSS, pdkFF, pdkSF) et Lag en minutes de travail, négatif pour une avance. AddDependency(aFromId, aToId, aKind = pdkFS, aLag = 0).
Calendar	TsgcPlannerCalendar	Le calendrier de travail : WorkDays (par défaut du lundi au vendredi, vide signifie tous les jours), WorkTime (par défaut `08:00-12:00,13:00-17:00` , jusqu'à 4 intervalles, vide signifie la journée entière), Holidays (Date, Name) et FirstDayOfWeek (par défaut pwdMonday). Méthodes IsWorkingDay, IsWorkingTime, WorkingMinutesPerDay, NextWorkingTime, PrevWorkingTime, AddWorkingMinutes et WorkingMinutesBetween.
ProjectStart	TDateTime	Le début du projet pour la planification automatique. 0 utilise le Start le plus ancien des tâches. ProjectStartTime en est le jumeau TsgcPlannerTime.
SelectedId	string	Public. L'Id de l'élément sélectionné, partagé par les vues qui ont SyncSelection.
Version	Int64	Public, en lecture seule. Augmente à chaque changement, sauf un changement de la seule sélection.
UpdateCount	Integer	Public, en lecture seule. La profondeur de BeginUpdate.

Items

Un **TsgcPlannerItem** est une tâche dans le Gantt, une carte dans le Kanban et un événement dans le Scheduler.

Propriété	Type	Description
Id	string	La clé unique de l'élément. Les parents, les liens, les ressources de la source DataSet et les fichiers importés y font référence.

Title, Notes	string	Le texte de l'élément.
Start, Finish	TDateTime	Les dates. Finish est exclusif. StartTime et FinishTime sont les valeurs TsgcPlannerTime exactes.
AllDay	Boolean	Un événement sur toute la journée, affiché dans la bande journée entière du Scheduler.
Status	string	L'Id d'un statut : la colonne Kanban de la carte. La correspondance est sensible à la casse.
Progress	Double	Pourcentage d'avancement, de 0 à 100.
Color	TsgcUIColor	$AARRGGBB. 0 (par défaut) prend la couleur du statut, puis de la première ressource, puis une couleur de palette choisie à partir de l'Id.
ParentId	string	L'Id de l'élément parent. Un élément qui a des enfants est un élément récapitulatif.
Tags, Resources	string	Listes séparées par des virgules : les étiquettes et les Id de ressources. TagCount, TagAt, ResourceCount et ResourceAt les lisent.
Rank	Integer	La position de la carte dans sa colonne Kanban.
Milestone, Manual	Boolean	Un jalon est dessiné sous forme de losange. Une tâche manuelle conserve ses dates lorsque le projet est planifié.
Constraint, ConstraintDate	TsgcPlannerConstraint, TDateTime	pcASAP (par défaut) ou pcSNET, début au plus tôt à ConstraintDate. Utilisé par la planification automatique.
BaselineStart, BaselineFinish	TDateTime	Le plan enregistré, 0 s'il n'est pas défini. Édition Registered pour les dessiner.
RecurrenceRule, RecurrenceExceptions	string	Une RRULE de la RFC 5545, telle que `FREQ=WEEKLY;BYDAY=MO,WE`, et une liste séparée par des virgules de dates ou de dates et heures exclues. Développée dans l'édition Registered.
RecurrenceParentId, RecurrenceDate	string, TDateTime	Défini sur une occurrence modifiée : la série à laquelle elle appartient et le début de l'occurrence qu'elle remplace.
IsSummary, Level, Duration, TotalSlack, Critical		Public, en lecture seule. Un récapitulatif a des enfants, Level est la profondeur dans l'arborescence, Duration les minutes de travail du calendrier, TotalSlack et Critical le résultat du moteur de planification.

Méthodes

Méthode	Retourne	Description
BeginUpdate, EndUpdate		Regroupent les changements. Les écouteurs et OnChange sont déclenchés une seule fois après le EndUpdate le plus externe.
AddItem(aId = '')	TsgcPlannerItem	Ajoute un élément avec cet Id.

FindItem(aId), IndexOfItem(aId)	TsgcPlannerItem, Integer	Trouve un élément par Id grâce à un index trié. nil ou -1 s'il n'y en a pas.
DeleteItem(aId)	Boolean	Supprime un élément, ses occurrences modifiées et ses liens. Ses enfants passent à son parent et la sélection est effacée si l'élément était sélectionné. False pour un Id inconnu.
MoveItem(aId, aStatus, aRank)	Boolean	Déplace une carte vers un statut et une position, et renumérote les rangs des deux colonnes. C'est ce que fait un dépôt dans le Kanban.
ChildrenOf(aId, var aList)	Integer	Les enfants d'un élément, ou les éléments racines pour un Id vide.
UpdateSummaries		Remonte les dates des feuilles dans leurs récapitulatifs, avec la progression pondérée par la durée de travail des feuilles.
GetOccurrences(aStart, aFinish, var aList)	Integer	Les occurrences dans la fenêtre semi-ouverte [aStart, aFinish) sous forme de TsgcPlannerTime, triées par début. Dans l'édition Registered, un élément récurrent donne une occurrence par date de sa règle, moins ses exceptions et ses occurrences modifiées.
GetRange(out aStart, aFinish)	Boolean	Le début le plus tôt et la fin la plus tardive des éléments. False s'il n'y en a aucun.
ItemChanging(Sender, aItem, aFields)	Boolean	Déclenche OnItemChanging et retourne aAccept. Une vue personnalisée l'appelle avant de valider une modification.
AddListener, RemoveListener(aListener)		Enregistrent un TsgcPlannerChangeEvent qui est informé de chaque changement. Les vues l'utilisent. Un écouteur ne doit pas modifier les données.
Clear		Supprime les éléments, les ressources, les statuts et les dépendances. Le calendrier et ProjectStart sont conservés.
LoadFromJSON, SaveToJSON, LoadFromFile, SaveToFile		L'ensemble des données en JSON (UTF-8 sans BOM). Un document illisible lève EsgcPlannerError et laisse les données inchangées.

Événements

Événement	Type	Description
OnChange	TsgcPlannerChangeEvent	Après le EndUpdate le plus externe d'un changement, après les écouteurs : aChange avec Kinds, ItemId (quand un seul élément a changé) et Version. Le gestionnaire peut de nouveau modifier les données.
OnItemChanging	TsgcPlannerItemChangingEvent	Avant qu'une vue valide une modification faite à la souris : aItem, aFields (un ensemble de pifStart, pifFinish, pifStatus, pifProgress et des autres champs de l'élément) et var aAccept. Mettez aAccept à False pour annuler le dépôt.

Exemple : un calendrier de travail, une réunion récurrente et le suivi des changements

```
uses
  SysUtils, Classes,
  sgcPlanner_DateUtils, sgcPlanner_Data;

type
  TPlannerEvents = class
  public
    Modified: Boolean;
    procedure ItemChanging(Sender: TObject; aItem: TsgcPlannerItem;
      const aFields: TsgcPlannerItemFields; var aAccept: Boolean);
    procedure DataChange(Sender: TObject; const aChange: TsgcPlannerChange);
  end;

procedure TPlannerEvents.ItemChanging(Sender: TObject; aItem: TsgcPlannerItem;
  const aFields: TsgcPlannerItemFields; var aAccept: Boolean);
begin
  // a finished task keeps its dates
  if (aItem.Status = 'done') and ((pifStart in aFields) or
    (pifFinish in aFields)) then
    aAccept := False;
end;

procedure TPlannerEvents.DataChange(Sender: TObject;
  const aChange: TsgcPlannerChange);
begin
  // a selection only change is not an edit
  if aChange.Kinds - [pckSelection] <> [] then
    Modified := True;
end;

function CountItemsOfWeek(aData: TsgcPlannerData; aMonday: TDateTime): Integer;
var
  vList: TsgcPlannerOccurrenceArray;
begin
  // half open window [Monday, next Monday), recurring items expanded
  Result := aData.GetOccurrences(sgcPlannerTimeOf(aMonday),
    sgcPlannerTimeOf(aMonday + 7), vList);
end;

procedure SetupData(aData: TsgcPlannerData; aEvents: TPlannerEvents);
var
  oItem: TsgcPlannerItem;
begin
  aData.Calendar.WorkTime := '08:00-12:00,13:00-17:00';
  aData.Calendar.Holidays.AddHoliday(EncodeDate(2026, 12, 25), 'Christmas');
  aData.OnItemChanging := aEvents.ItemChanging;
  aData.OnChange := aEvents.DataChange;

  // a weekly meeting on Mondays and Wednesdays, ten times
  oItem := aData.AddItem('standup');
  oItem.Title := 'Stand-up';
  oItem.Start := EncodeDate(2026, 10, 5) + EncodeTime(9, 0, 0, 0);
  oItem.Finish := EncodeDate(2026, 10, 5) + EncodeTime(9, 15, 0, 0);
  oItem.RecurrenceRule := 'FREQ=WEEKLY;BYDAY=MO,WE;COUNT=10';

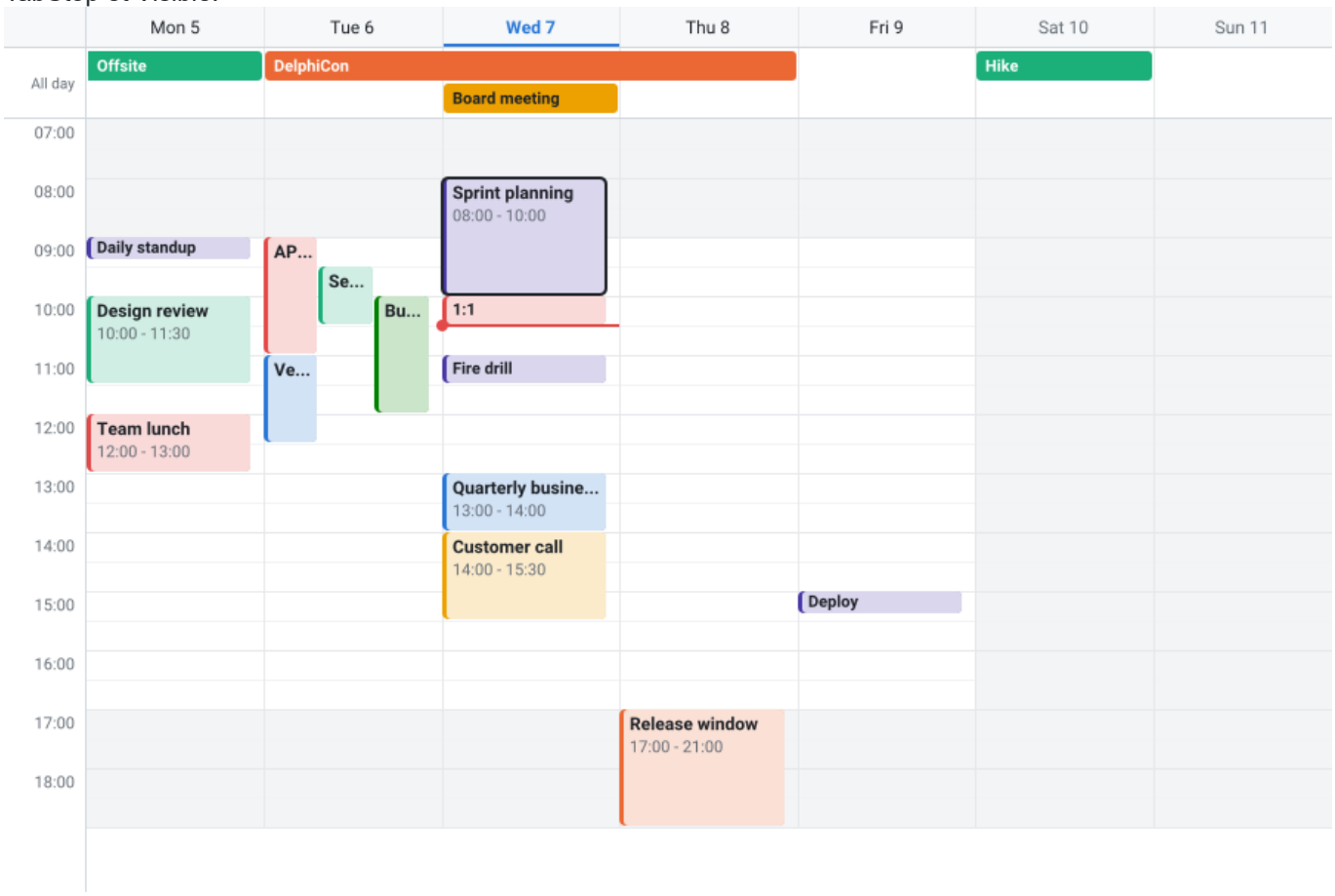
  aData.SaveToFile('project.json');
  aEvents.Modified := False;
end;
```

Voir aussi

- Architecture
- Démarrage rapide VCL
- TsgcPlannerSource_DataSet
- Données partagées
- Calendrier de travail
- Récurrence
- Dépendances
- Persistance JSON

TsgcVCLScheduler

Unité : sgcPlanner_Scheduler_VCL
TsgcVCLScheduler est le Scheduler VCL, Windows uniquement. Il affiche les éléments d'un TsgcPlannerData dans une vue jour, semaine de travail, semaine, mois ou agenda, avec une bande journée entière, une ligne à l'heure courante, des événements qui se chevauchent côte à côte et, avec Options.GroupByResource, une colonne par ressource. Dans l'édition Registered, il a aussi la Timeline de ressources : une ligne par ressource sur une échelle de temps. L'utilisateur crée un événement en faisant glisser sur un espace libre, le déplace et le redimensionne, et le dépôt s'aligne sur SlotMinutes. Un déplacement vers la colonne d'une autre ressource change la ressource. Ctrl et la molette de la souris changent la hauteur des créneaux dans les vues par jour. Il hérite de TsgcUIVCLControl et publie Align, Anchors, Constraints, Enabled, PopupMenu, ShowHint, ParentShowHint, TabOrder, TabStop et Visible.



Propriétés

Propriété	Type	Description
Data	TsgcPlannerData	Les données que le contrôle affiche. Plusieurs contrôles peuvent partager un même objet de données. Quand il vaut nil, le contrôle utilise son propre objet de données interne.
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom. utCustom part des couleurs claires.
Colors	TsgcPlannerViewColors	Remplacements des couleurs du thème : Background, Surface, Border, Grid, HeaderBackground, HeaderText, Text, TextMuted, Accent, Selection, NonWorking, NowLine, Critical et Ghost. 0 (par défaut) conserve la couleur du thème.

ReadOnly	Boolean	Désactive tous les glissements et la touche Suppr. Par défaut False.
DragOptions	TsgcPlannerDragOptions	Les glissements que l'utilisateur peut effectuer : pdoCreate, pdoMove, pdoResize, pdoProgress, pdoLink et pdoReorder. Par défaut, tous.
SyncSelection	Boolean	Partage la sélection avec les autres vues des mêmes données, via SelectedId des données. Par défaut True.
MaxFPS	Integer	Le plus grand nombre d'images par seconde que le contrôle dessine.
BorderStyle	TBorderStyle	bsSingle (par défaut) ou bsNone.
View	TsgcPlannerSchedulerViewKind	psvDay, psvWorkWeek (les jours ouvrés du calendrier), psvWeek (par défaut), psvMonth (six semaines), psvAgenda (une liste de AgendaDays jours) ou psvTimeline (une ligne par ressource, édition Registered).
Date	TDateTime	La date que la vue affiche. 0 (par défaut) suit l'horloge.
Options	TsgcPlannerSchedulerOptions	DayStartHour et DayEndHour (par défaut 0 et 24), SlotMinutes (30), SlotHeight (20 DIP), ShowAllDayBand (True), ShowCurrentTime (True), GroupByResource (False), MonthMaxItems (0, autant qu'il en tient), AgendaDays (7) et ResourceRowHeight (40 DIP, Timeline).
TimeScale	TsgcPlannerTimeScale	L'axe du temps de la Timeline : Zoom (de pz5Minutes à pzYears, par défaut pzDays), TickWidth, Origin, Snap, ShowNonWorking, MajorFormat et MinorFormat. Le zoom définit aussi la période de la Timeline : un jour jusqu'à pzHours, 4 semaines avec pzDays, 3 mois avec pzWeeks et un an avec pzMonths.
SchedulerView	TsgcPlannerSchedulerView	Public, en lecture seule. Le modèle de vue, avec la disposition des jours, des colonnes et des entrées.

Méthodes

Méthode	Retourne	Description
GoToday		Définit Date à aujourd'hui.
Next, Prior		Déplace Date vers la période suivante ou précédente de la vue.
ScrollToTime(aTime)		Affiche le jour de aTime et fait défiler la grille horaire jusqu'à son heure.
VisibleRange(out aStart, aFinish)		La période que la vue affiche.
ItemAt(X, Y)	TsgcPlannerItem	L'élément situé sous un point du contrôle, nil s'il n'y en a pas.
ScrollToItem(aId)	Boolean	Défile jusqu'à ce que l'élément soit visible. False si l'élément n'est pas affiché.
SaveToPNG, SaveToSVG(aFileName ou aStream, aWidth = 0, aHeight = 0, aScale = 0)		La vue sous forme d'image, la même image qu'un TsgcPlannerRenderer avec les mêmes données, options, taille et échelle. La taille est en DIP, 0 est la taille client, et une échelle de 0 est l'échelle du contrôle. La position de défilement est conservée.

		Les barres de défilement, l'aperçu du glissement et les repères de dépôt sont omis.

Événements

Événement	Type	Description
OnItemClick, OnItemDblClick	TsgcPlannerItemEvent	Un clic ou un double-clic sur un élément : aItem.
OnItemDelete	TsgcPlannerItemAcceptEvent	La touche Suppr sur l'élément sélectionné : aItem et var aAccept. Mettez aAccept à False pour conserver l'élément. Un élément supprimé emporte ses liens, et ses enfants passent à son parent.
OnSelectionChange	TNotifyEvent	L'élément sélectionné a changé, dans ce contrôle ou, avec SyncSelection, dans une autre vue des mêmes données.
OnItemCreate	TsgcPlannerSchedulerItemCreateEvent	Un glissement sur un espace libre : aStart, aFinish, aResourceId (la colonne quand le regroupement se fait par ressource), var aTitle (par défaut "New event") et var aAccept (par défaut True). Le nouvel élément est sélectionné.
OnRecurrenceEdit	TsgcPlannerRecurrenceEditEvent	Un dépôt sur une occurrence d'une série récurrente : aItem (la série), aOccurrence et var aScope. prsOccurrence (par défaut) crée une occurrence modifiée, prsSeries décale toute la série et prsCancel annule le dépôt. Édition Registered.
OnRecurrenceDelete	TsgcPlannerRecurrenceEditEvent	La touche Suppr sur une occurrence d'une série récurrente, avant toute suppression. prsOccurrence (par défaut) ajoute l'occurrence aux exceptions de la série, prsSeries supprime la série après OnItemDelete, prsCancel ne supprime rien. Édition Registered.
OnDateChange, OnViewChange	TNotifyEvent	Date ou View a changé, y compris par un clic sur "+N more" dans la vue mois, qui ouvre ce jour.

Exemple : des heures de bureau, des colonnes de ressources et des événements récurrents

```
uses
  SysUtils, Classes, Controls, Dialogs,
  sgcPlanner_Data, sgcPlanner_Scheduler, sgcPlanner_Scheduler_VCL;

type
  TSchedulerEvents = class
  public
    procedure ItemCreate(Sender: TObject; const aStart, aFinish: TDateTime;
      const aResourceId: string; var aTitle: string; var aAccept: Boolean);
    procedure RecurrenceEdit(Sender: TObject; aItem: TsgcPlannerItem;
      const aOccurrence: TsgcPlannerOccurrence;
      var aScope: TsgcPlannerRecurrenceScope);
  end;

procedure TSchedulerEvents.ItemCreate(Sender: TObject;
  const aStart, aFinish: TDateTime; const aResourceId: string;
  var aTitle: string; var aAccept: Boolean);
begin
  // no new events on Saturday or Sunday
  aAccept := not (DayOfWeek(aStart) in [1, 7]);
```

```
  aTitle := 'Meeting';
end;

procedure TSchedulerEvents.RecurrenceEdit(Sender: TObject;
  aItem: TsgcPlannerItem; const aOccurrence: TsgcPlannerOccurrence;
  var aScope: TsgcPlannerRecurrenceScope);
begin
  case MessageDlg('Change every occurrence of "' + aItem.Title + '"?',
    mtConfirmation, [mbYes, mbNo, mbCancel], 0) of
    mrYes:
      aScope := prsSeries;
    mrNo:
      aScope := prsOccurrence;
  else
    aScope := prsCancel;
  end;
end;

procedure SetupScheduler(aScheduler: TsgcVCLScheduler;
  aEvents: TSchedulerEvents);
begin
  aScheduler.View := psvWorkWeek;
  aScheduler.Options.DayStartHour := 7;
  aScheduler.Options.DayEndHour := 19;
  aScheduler.Options.SlotMinutes := 15;
  aScheduler.Options.GroupByResource := True;
  aScheduler.OnItemCreate := aEvents.ItemCreate;
  aScheduler.OnRecurrenceEdit := aEvents.RecurrenceEdit;
  // the same question before a delete of a recurring event
  aScheduler.OnRecurrenceDelete := aEvents.RecurrenceEdit;
  aScheduler.GoToday;
  aScheduler.ScrollToTime(Now);
end;
```

Voir aussi

- TsgcFMXScheduler
- TsgcPlannerData
- Démarrage rapide VCL
- Récurrence
- Échelle de temps et zoom
- Glisser-déposer
- Vues synchronisées

TsgcFMXScheduler

Unité : sgcPlanner_Scheduler_FMX
TsgcFMXScheduler est le Scheduler FireMonkey, à partir de Delphi XE6 (SGC_PLANNER_FMX). Il a les mêmes vues, propriétés, méthodes et événements que TsgcVCLScheduler. Il dessine ses propres barres de défilement et traite la saisie tactile comme une saisie à la souris. Il hérite de TsgcUIFMXControl et publie les propriétés et événements de disposition habituels de FireMonkey.

Propriétés

Propriété	Type	Description
Data	TsgcPlannerData	Les données que le contrôle affiche. Plusieurs contrôles peuvent partager un même objet de données. Quand il vaut nil, le contrôle utilise son propre objet de données interne.
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom. utCustom part des couleurs claires.
Colors	TsgcPlannerViewColors	Remplacements des couleurs du thème : Background, Surface, Border, Grid, HeaderBackground, HeaderText, Text, TextMuted, Accent, Selection, NonWorking, NowLine, Critical et Ghost. 0 (par défaut) conserve la couleur du thème.
ReadOnly	Boolean	Désactive tous les glissements et la touche Suppr. Par défaut False.
DragOptions	TsgcPlannerDragOptions	Les glissements que l'utilisateur peut effectuer : pdoCreate, pdoMove, pdoResize, pdoProgress, pdoLink et pdoReorder. Par défaut, tous.
SyncSelection	Boolean	Partage la sélection avec les autres vues des mêmes données, via SelectedId des données. Par défaut True.
MaxFPS	Integer	Le plus grand nombre d'images par seconde que le contrôle dessine.
View	TsgcPlannerSchedulerViewKind	psvDay, psvWorkWeek (les jours ouvrés du calendrier), psvWeek (par défaut), psvMonth (six semaines), psvAgenda (une liste de AgendaDays jours) ou psvTimeline (une ligne par ressource, édition Registered).
Date	TDateTime	La date que la vue affiche. 0 (par défaut) suit l'horloge.
Options	TsgcPlannerSchedulerOptions	DayStartHour et DayEndHour (par défaut 0 et 24), SlotMinutes (30), SlotHeight (20 DIP), ShowAllDayBand (True), ShowCurrentTime (True), GroupByResource (False), MonthMaxItems (0, autant qu'il en tient), AgendaDays (7) et ResourceRowHeight (40 DIP, Timeline).
TimeScale	TsgcPlannerTimeScale	L'axe du temps de la Timeline : Zoom (de pz5Minutes à pzYears, par défaut pzDays), TickWidth, Origin, Snap, ShowNonWorking, MajorFormat et MinorFormat. Le zoom définit aussi la période de la Timeline : un jour jusqu'à pzHours, 4 semaines avec pzDays, 3 mois avec pzWeeks et un an avec pzMonths.
SchedulerView	TsgcPlannerSchedulerView	Public, en lecture seule. Le modèle de vue, avec la disposition des jours, des colonnes et des entrées.

Méthodes

Méthode	Retourne	Description
GoToday		Définit Date à aujourd'hui.
Next, Prior		Déplace Date vers la période suivante ou précédente de la vue.
ScrollToTime(aTime)		Affiche le jour de aTime et fait défiler la grille horaire jusqu'à son heure.
VisibleRange(out aStart, aFinish)		La période que la vue affiche.
ItemAt(X, Y (Single))	TsgcPlannerItem	L'élément situé sous un point du contrôle, nil s'il n'y en a pas.
ScrollToItem(aId)	Boolean	Défile jusqu'à ce que l'élément soit visible. False si l'élément n'est pas affiché.
SaveToPNG, SaveToSVG(aFileName ou aStream, aWidth = 0, aHeight = 0, aScale = 0)		La vue sous forme d'image, la même image qu'un TsgcPlannerRenderer avec les mêmes données, options, taille et échelle. La taille est en DIP, 0 est la taille client, et une échelle de 0 est l'échelle du contrôle. La position de défilement est conservée. Les barres de défilement, l'aperçu du glissement et les repères de dépôt sont omis.

Événements

Événement	Type	Description
OnItemClick, OnItemDblClick	TsgcPlannerItemEvent	Un clic ou un double-clic sur un élément : aItem.
OnItemDelete	TsgcPlannerItemAcceptEvent	La touche Suppr sur l'élément sélectionné : aItem et var aAccept. Mettez aAccept à False pour conserver l'élément. Un élément supprimé emporte ses liens, et ses enfants passent à son parent.
OnSelectionChange	TNotifyEvent	L'élément sélectionné a changé, dans ce contrôle ou, avec SyncSelection, dans une autre vue des mêmes données.
OnItemCreate	TsgcPlannerSchedulerItemCreateEvent	Un glissement sur un espace libre : aStart, aFinish, aResourceId (la colonne quand le regroupement se fait par ressource), var aTitle (par défaut "New event") et var aAccept (par défaut True). Le nouvel élément est sélectionné.
OnRecurrenceEdit	TsgcPlannerRecurrenceEditEvent	Un dépôt sur une occurrence d'une série récurrente : aItem (la série), aOccurrence et var aScope. prsOccurrence (par défaut) crée une occurrence modifiée, prsSeries décale toute la série et prsCancel annule le dépôt. Édition Registered.
OnRecurrenceDelete	TsgcPlannerRecurrenceEditEvent	La touche Suppr sur une occurrence d'une série récurrente, avant toute suppression. prsOccurrence (par défaut) ajoute l'occurrence aux exceptions de la série, prsSeries supprime la série après OnItemDelete, prsCancel ne supprime rien. Édition Registered.

OnDate-Change, On-ViewChange	TNotifyEvent	Date ou View a changé, y compris par un clic sur "+N more" dans la vue mois, qui ouvre ce jour.

Exemple : la Timeline de ressources enregistrée en PNG

```
uses
  System.SysUtils, System.Classes, FMX.Types, FMX.Controls,
  sgcPlanner_Data, sgcPlanner_TimeScale, sgcPlanner_Scheduler,
  sgcPlanner_Scheduler_FMX;

procedure ShowTimeline(aScheduler: TsgcFMXScheduler; aData: TsgcPlannerData);
begin
  aScheduler.Align := TAlignLayout.Client;
  aScheduler.Data := aData;
  // one row per resource, four weeks at the day zoom
  aScheduler.View := psvTimeline;
  aScheduler.TimeScale.Zoom := pzDays;
  aScheduler.Options.ResourceRowHeight := 48;
  aScheduler.Date := Date;
  // the same picture as a file, at twice the size
  aScheduler.SaveToPNG('timeline.png', 0, 0, 2);
end;
```

Voir aussi

- TsgcVCLScheduler
- Démarrage rapide FMX
- Récurrence
- Échelle de temps et zoom
- Glisser-déposer

TsgcVCLGantt

Unité : sgcPlanner_Gantt_VCL
TsgcVCLGantt est le diagramme de Gantt VCL, Windows uniquement. Il affiche les éléments d'un TsgcPlannerData sous forme d'arborescence de tâches avec des numéros WBS et des colonnes, et d'un diagramme à barres sur une échelle de temps à deux niveaux : des barres avec leur progression, des losanges pour les jalons, des crochets pour les récapitulatifs, le temps non travaillé grisé et une ligne à la date du jour. L'utilisateur déplace une barre, qui conserve sa durée de travail, la redimensionne, fait glisser sa progression, fait glisser de la fin d'une barre vers une autre barre pour créer un lien, et ouvre et ferme les récapitulatifs. Un déplacement définit un début au plus tôt, sauf si la tâche est Manual, et les récapitulatifs suivent leurs enfants. Dans l'édition Registered, les liens sont dessinés sous forme de flèches, et le Gantt peut planifier le projet, afficher le chemin critique et comparer le plan à une ligne de base. Ctrl et la molette de la souris zooment l'échelle de temps. Il hérite de TsgcUIVCLControl et publie Align, Anchors, Constraints, Enabled, PopupMenu, ShowHint, ParentShowHint, TabOrder, TabStop et Visible.

Propriétés

Propriété	Type	Description
Data	TsgcPlannerData	Les données que le contrôle affiche. Plusieurs contrôles peuvent partager un même objet de données. Quand il vaut nil, le contrôle utilise son propre objet de données interne.
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom. utCustom part des couleurs claires.
Colors	TsgcPlannerViewColors	Remplacements des couleurs du thème : Background, Surface, Border, Grid, HeaderBackground, HeaderText, Text, TextMuted, Accent, Selection, NonWorking, NowLine, Critical et Ghost. 0 (par défaut) conserve la couleur du thème.
ReadOnly	Boolean	Désactive tous les glissements et la touche Suppr. Par défaut False.
DragOptions	TsgcPlannerDragOptions	Les glissements que l'utilisateur peut effectuer : pdoCreate, pdoMove, pdoResize, pdoProgress, pdoLink et pdoReorder. Par défaut, tous.
SyncSelection	Boolean	Partage la sélection avec les autres vues des mêmes données, via SelectedId des données. Par défaut True.
MaxFPS	Integer	Le plus grand nombre d'images par seconde que le contrôle dessine.
BorderStyle	TBorderStyle	bsSingle (par défaut) ou bsNone.
Columns	TsgcPlannerGanttColumns	Les colonnes de l'arborescence des tâches. Chacune a un Field (pgcId, pgcWBS, pgcTitle, pgcStart, pgcFinish, pgcDuration, pgcProgress, pgcStatus, pgcResources), un Caption (vide utilise le libellé par défaut), Width, Visible et Alignment. Les colonnes par défaut sont ID, nom de la tâche, début, fin, durée et progression. AddColumn(aField, aWidth = 100, aAlignment = taLeftJustify) et SetDefaults.
TreeWidth	Integer	La largeur de l'arborescence des tâches en DIP. Par défaut 550. L'utilisateur fait glisser le séparateur pour la modifier.
RowHeight	Integer	La hauteur d'une ligne en DIP. Par défaut 28.

TimeScale	TsgcPlannerTimeScale	L'axe du temps : Zoom (pz5Minutes, pz15Minutes, pzHours, pzDays par défaut, pzWeeks, pzMonths, pzQuarters, pzYears), TickWidth (0 utilise la valeur par défaut du zoom), Origin, Snap (psNone, psTick par défaut, ps5Min, ps15Min, psHour, psDay), ShowNonWorking (True) et les MajorFormat et MinorFormat des deux niveaux de l'en-tête.
DurationUnit	TsgcPlannerDurationUnit	L'unité de la colonne durée : pduMinutes, pduHours, pduDays (par défaut) ou pduWeeks, comptées en temps de travail. Le séparateur décimal est celui de la locale du Planner.
BarColorMode	TsgcPlannerBarColorMode	pbcItem (par défaut) colore une barre comme l'élément, pbcStatus selon son statut et pbcResource selon sa première ressource.
ShowProgress, ShowToday	Boolean	Dessinent la progression à l'intérieur des barres et une ligne à la date du jour. Par défaut True.
ShowDependencies	Boolean	Dessine les liens sous forme de flèches coudées. Par défaut True. Édition Registered.
ShowCriticalPath, ShowBaseline	Boolean	Dessinent les tâches et les liens critiques dans la couleur critique, et la ligne de base sous chaque barre. Par défaut False. Édition Registered.
AutoSchedule	Boolean	Replanifie le projet après chaque modification faite dans le diagramme. Par défaut False. Édition Registered.
GanttView	TsgcPlannerGanttView	Public, en lecture seule. Le modèle de vue, avec les lignes, la géométrie des barres et les liens.
View	TsgcPlannerView	Public, en lecture seule. Le modèle de vue que le contrôle héberge.

Méthodes

Méthode	Retourne	Description
Expand(aId), Collapse(aId)	Boolean	Ouvre ou ferme un élément récapitulatif.
ExpandAll, CollapseAll		Ouvre ou ferme tous les éléments récapitulatifs.
ZoomIn, ZoomOut	Boolean	Un niveau de zoom plus fin ou plus large, autour du centre du diagramme. False au dernier niveau.
ZoomToFit	Boolean	Choisit le niveau de zoom le plus fin qui affiche toutes les lignes et défile jusqu'au premier début. False sans lignes.
ScrollToTime(aTime)		Fait défiler le diagramme pour que aTime soit à son bord gauche.
Schedule	Boolean	Exécute le moteur de planification sur les données : les tâches après leurs prédécesseurs, à partir de ProjectStart, sur le calendrier de travail, avec la marge et le chemin critique. Un cycle dans

		les liens déclenche OnScheduleError, et les tâches du cycle conservent leurs dates. En Community, il lève EsgcPlannerError.
SetBaseline		Copie Start et Finish de chaque tâche dans BaselineStart et BaselineFinish. En Community, il lève EsgcPlannerError.
ItemAt(X, Y)	TsgcPlannerItem	L'élément situé sous un point du contrôle, nil s'il n'y en a pas.
ScrollToItem(aId)	Boolean	Défile jusqu'à ce que l'élément soit visible. False si l'élément n'est pas affiché.
SaveToPNG, SaveToSVG(aFileName ou aStream, aWidth = 0, aHeight = 0, aScale = 0)		La vue sous forme d'image, la même image qu'un TsgcPlannerRenderer avec les mêmes données, options, taille et échelle. La taille est en DIP, 0 est la taille client, et une échelle de 0 est l'échelle du contrôle. La position de défilement est conservée. Les barres de défilement, l'aperçu du glissement et les repères de dépôt sont omis.

Événements

Événement	Type	Description
OnItemClick, OnItemDblClick	TsgcPlannerItemEvent	Un clic ou un double-clic sur un élément : aItem.
OnItemDelete	TsgcPlannerItemAcceptEvent	La touche Suppr sur l'élément sélectionné : aItem et var aAccept. Mettez aAccept à False pour conserver l'élément. Un élément supprimé emporte ses liens, et ses enfants passent à son parent.
OnSelectionChange	TNotifyEvent	L'élément sélectionné a changé, dans ce contrôle ou, avec SyncSelection, dans une autre vue des mêmes données.
OnLinkCreating	TsgcPlannerGanttLinkCreatingEvent	Un glissement de la fin d'une barre vers une autre barre : aFrom, aTo, var aKind (selon les extrémités choisies) et var aAccept. Les liens vers l'élément lui-même, vers un ancêtre ou un descendant et les doublons sont refusés avant l'événement. Édition Registered.
OnScheduleError	TsgcPlannerGanttScheduleErrorEvent	Le moteur a trouvé un cycle : aError et aCycleIds, les Id des éléments du cycle. Leurs dates sont conservées.
OnExpandChange	TsgcPlannerGanttExpandEvent	Un récapitulatif a été ouvert ou fermé : aItem et aExpanded. ExpandAll et CollapseAll le déclenchent une seule fois, avec aItem à nil.

Exemple : des colonnes WBS, la planification automatique et le chemin critique

```
uses
  SysUtils, Classes, Dialogs,
  sgcPlanner_Data, sgcPlanner_TimeScale, sgcPlanner_Gantt,
  sgcPlanner_Gantt_VCL;

type
  TGanttEvents = class
  public
```

```
    procedure LinkCreating(Sender: TObject; aFrom, aTo: TsgcPlannerItem;
      var aKind: TsgcPlannerDependencyKind; var aAccept: Boolean);
    procedure ScheduleError(Sender: TObject; const aError: string;
      aCycleIds: TStrings);
  end;

procedure TGanttEvents.LinkCreating(Sender: TObject;
  aFrom, aTo: TsgcPlannerItem; var aKind: TsgcPlannerDependencyKind;
  var aAccept: Boolean);
begin
  // a milestone only takes finish to start links
  if aTo.Milestone then
    aKind := pdkFS;
  // no links into finished work
  aAccept := aTo.Progress < 100;
end;

procedure TGanttEvents.ScheduleError(Sender: TObject; const aError: string;
  aCycleIds: TStrings);
begin
  ShowMessage(aError + sLineBreak + aCycleIds.CommaText);
end;

procedure SetupGantt(aGantt: TsgcVCLGantt; aEvents: TGanttEvents);
begin
  aGantt.Columns.Clear;
  aGantt.Columns.AddColumn(pgcWBS, 50);
  aGantt.Columns.AddColumn(pgcTitle, 200);
  aGantt.Columns.AddColumn(pgcResources, 120);
  aGantt.Columns.AddColumn(pgcDuration, 70, taRightJustify);
  aGantt.TreeWidth := 440;
  aGantt.BarColorMode := pbcStatus;
  aGantt.TimeScale.Zoom := pzWeeks;
  aGantt.OnLinkCreating := aEvents.LinkCreating;
  aGantt.OnScheduleError := aEvents.ScheduleError;
  // Registered edition: schedule on every edit, show the critical path
  aGantt.AutoSchedule := True;
  aGantt.ShowCriticalPath := True;
  aGantt.ShowBaseline := True;
  aGantt.Schedule;
  aGantt.SetBaseline;
  aGantt.ExpandAll;
  aGantt.ZoomToFit;
end;
```

Voir aussi

- TsgcFMXGantt
- TsgcPlannerData
- TsgcPlannerMSProject
- Dépendances
- Planification automatique
- Chemin critique
- Lignes de base
- Échelle de temps et zoom

TsgcFMXGantt

Unité : sgcPlanner_Gantt_FMX
TsgcFMXGantt est le diagramme de Gantt FireMonkey, à partir de Delphi XE6 (SGC_PLANNER_FMX). Il a la même arborescence, le même diagramme, les mêmes propriétés, méthodes et événements que TsgcVCLGantt. Il dessine ses propres barres de défilement et traite la saisie tactile comme une saisie à la souris. Il hérite de TsgcUIFMXControl et publie les propriétés et événements de disposition habituels de FireMonkey.

Propriétés

Propriété	Type	Description
Data	TsgcPlannerData	Les données que le contrôle affiche. Plusieurs contrôles peuvent partager un même objet de données. Quand il vaut nil, le contrôle utilise son propre objet de données interne.
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom. utCustom part des couleurs claires.
Colors	TsgcPlannerViewColors	Remplacements des couleurs du thème : Background, Surface, Border, Grid, HeaderBackground, HeaderText, Text, TextMuted, Accent, Selection, NonWorking, NowLine, Critical et Ghost. 0 (par défaut) conserve la couleur du thème.
ReadOnly	Boolean	Désactive tous les glissements et la touche Suppr. Par défaut False.
DragOptions	TsgcPlannerDragOptions	Les glissements que l'utilisateur peut effectuer : pdoCreate, pdoMove, pdoResize, pdoProgress, pdoLink et pdoReorder. Par défaut, tous.
SyncSelection	Boolean	Partage la sélection avec les autres vues des mêmes données, via SelectedId des données. Par défaut True.
MaxFPS	Integer	Le plus grand nombre d'images par seconde que le contrôle dessine.
Columns	TsgcPlannerGanttColumns	Les colonnes de l'arborescence des tâches. Chacune a un Field (pgcId, pgcWBS, pgcTitle, pgcStart, pgcFinish, pgcDuration, pgcProgress, pgcStatus, pgcResources), un Caption (vide utilise le libellé par défaut), Width, Visible et Alignment. Les colonnes par défaut sont ID, nom de la tâche, début, fin, durée et progression. AddColumn(aField, aWidth = 100, aAlignment = taLeftJustify) et SetDefaults.
TreeWidth	Integer	La largeur de l'arborescence des tâches en DIP. Par défaut 550. L'utilisateur fait glisser le séparateur pour la modifier.
RowHeight	Integer	La hauteur d'une ligne en DIP. Par défaut 28.
TimeScale	TsgcPlannerTimeScale	L'axe du temps : Zoom (pz5Minutes, pz15Minutes, pzHours, pzDays par défaut, pzWeeks, pzMonths, pzQuarters, pzYears), TickWidth (0 utilise la valeur par défaut du zoom), Origin, Snap (psNone, psTick par défaut, ps5Min, ps15Min, psHour, psDay), ShowNonWorking (True) et les MajorFormat et MinorFormat des deux niveaux de l'en-tête.
DurationUnit	TsgcPlannerDurationUnit	L'unité de la colonne durée : pduMinutes, pduHours, pduDays (par défaut) ou pduWeeks, comptées en temps de travail. Le séparateur décimal est celui de la locale du Planner.

BarColorMode	TsgcPlannerBarColorMode	pbcItem (par défaut) colore une barre comme l'élément, pbcStatus selon son statut et pbcResource selon sa première ressource.
ShowProgress, ShowToday	Boolean	Dessinent la progression à l'intérieur des barres et une ligne à la date du jour. Par défaut True.
ShowDependencies	Boolean	Dessine les liens sous forme de flèches coudées. Par défaut True. Édition Registered.
ShowCriticalPath, ShowBaseline	Boolean	Dessinent les tâches et les liens critiques dans la couleur critique, et la ligne de base sous chaque barre. Par défaut False. Édition Registered.
AutoSchedule	Boolean	Replanifie le projet après chaque modification faite dans le diagramme. Par défaut False. Édition Registered.
GanttView	TsgcPlannerGanttView	Public, en lecture seule. Le modèle de vue, avec les lignes, la géométrie des barres et les liens.
View	TsgcPlannerView	Public, en lecture seule. Le modèle de vue que le contrôle héberge.

Méthodes

Méthode	Retourne	Description
Expand(aId), Collapse(aId)	Boolean	Ouvre ou ferme un élément récapitulatif.
ExpandAll, CollapseAll		Ouvre ou ferme tous les éléments récapitulatifs.
ZoomIn, ZoomOut	Boolean	Un niveau de zoom plus fin ou plus large, autour du centre du diagramme. False au dernier niveau.
ZoomToFit	Boolean	Choisit le niveau de zoom le plus fin qui affiche toutes les lignes et défile jusqu'au premier début. False sans lignes.
ScrollToTime(aTime)		Fait défiler le diagramme pour que aTime soit à son bord gauche.
Schedule	Boolean	Exécute le moteur de planification sur les données : les tâches après leurs prédécesseurs, à partir de ProjectStart, sur le calendrier de travail, avec la marge et le chemin critique. Un cycle dans les liens déclenche OnScheduleError, et les tâches du cycle conservent leurs dates. En Community, il lève EsgcPlannerError.
SetBaseline		Copie Start et Finish de chaque tâche dans BaselineStart et BaselineFinish. En Community, il lève EsgcPlannerError.
ItemAt(X, Y (Single))	TsgcPlannerItem	L'élément situé sous un point du contrôle, nil s'il n'y en a pas.

ScrollToItem(aId)	Boolean	Défile jusqu'à ce que l'élément soit visible. False si l'élément n'est pas affiché.
SaveToPNG, SaveToSVG(aFileName ou aStream, aWidth = 0, aHeight = 0, aScale = 0)		La vue sous forme d'image, la même image qu'un TsgcPlanner-Renderer avec les mêmes données, options, taille et échelle. La taille est en DIP, 0 est la taille client, et une échelle de 0 est l'échelle du contrôle. La position de défilement est conservée. Les barres de défilement, l'aperçu du glissement et les repères de dépôt sont omis.

Événements

Événement	Type	Description
OnItemClick, OnItemDblClick	TsgcPlannerItemEvent	Un clic ou un double-clic sur un élément : aItem.
OnItemDelete	TsgcPlannerItemAcceptEvent	La touche Suppr sur l'élément sélectionné : aItem et var aAccept. Mettez aAccept à False pour conserver l'élément. Un élément supprimé emporte ses liens, et ses enfants passent à son parent.
OnSelectionChange	TNotifyEvent	L'élément sélectionné a changé, dans ce contrôle ou, avec SyncSelection, dans une autre vue des mêmes données.
OnLinkCreating	TsgcPlannerGanttLinkCreatingEvent	Un glissement de la fin d'une barre vers une autre barre : aFrom, aTo, var aKind (selon les extrémités choisies) et var aAccept. Les liens vers l'élément lui-même, vers un ancêtre ou un descendant et les doublons sont refusés avant l'événement. Édition Registered.
OnScheduleError	TsgcPlannerGanttScheduleErrorEvent	Le moteur a trouvé un cycle : aError et aCycleIds, les Id des éléments du cycle. Leurs dates sont conservées.
OnExpandChange	TsgcPlannerGanttExpandEvent	Un récapitulatif a été ouvert ou fermé : aItem et aExpanded. ExpandAll et CollapseAll le déclenchent une seule fois, avec aItem à nil.

Exemple : des heures, un magnétisme de 15 minutes et une phase ouverte

```
uses
  System.SysUtils, FMX.Types, FMX.Controls,
  sgcPlanner_Data, sgcPlanner_TimeScale, sgcPlanner_Gantt,
  sgcPlanner_Gantt_FMX;

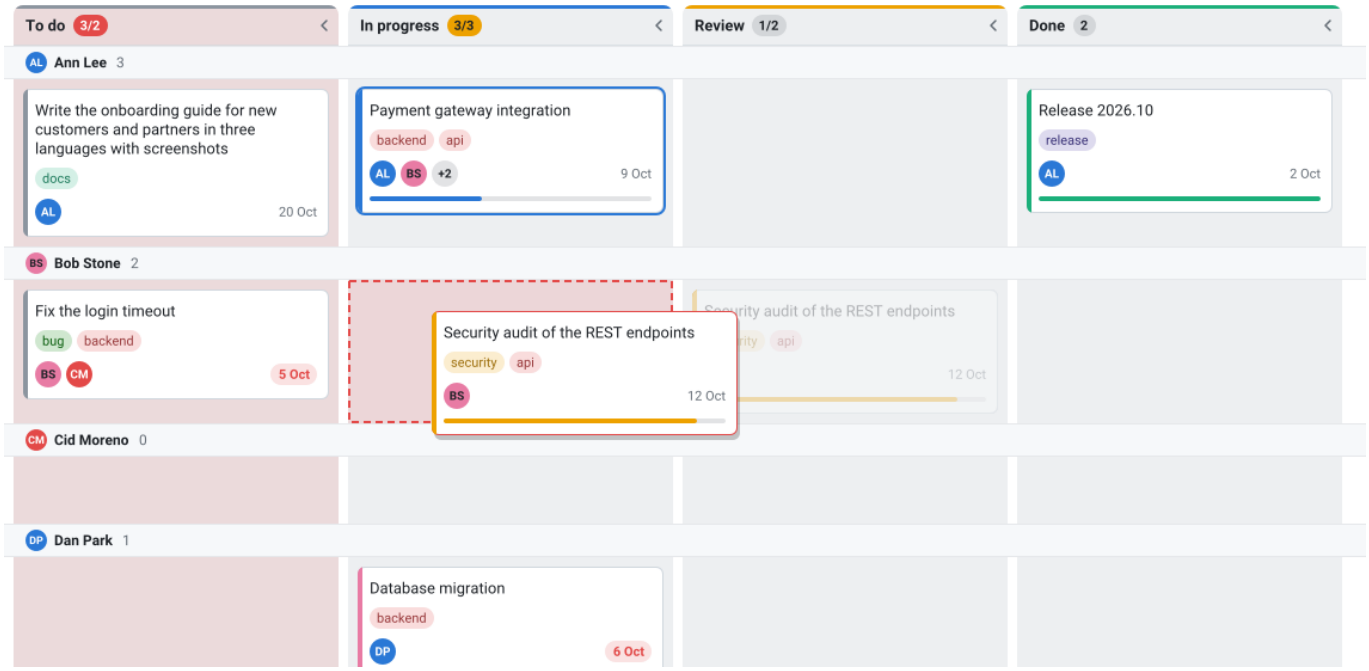
procedure SetupGantt(aGantt: TsgcFMXGantt; aData: TsgcPlannerData);
begin
  aGantt.Align := TAlignLayout.Client;
  aGantt.Data := aData;
  aGantt.RowHeight := 32;
  aGantt.DurationUnit := pduHours;
  aGantt.TimeScale.Snap := ps15Min;
  // only the summaries of the first level are open
  aGantt.CollapseAll;
  aGantt.Expand('phase1');
  aGantt.ScrollToTime(Now);
end;
```

Voir aussi

- TsgcVCLGantt
- Démarrage rapide FMX
- Dépendances
- Planification automatique
- Chemin critique
- Lignes de base
- Échelle de temps et zoom

TsgcVCLKanban

Unité : sgcPlanner_Kanban_VCL
TsgcVCLKanban est le tableau Kanban VCL, Windows uniquement. Il affiche les éléments d'un TsgcPlannerData sous forme de cartes sur un tableau : une colonne par statut, avec un badge de comptage, une limite WIP facultative et une flèche de repli, et, dans l'édition Registered, des couloirs par ressource, par parent, par étiquette ou par événement. Une carte affiche son titre sur plusieurs lignes, ses étiquettes, les initiales de ses ressources, son échéance et sa progression. L'utilisateur fait glisser une carte vers une autre position, colonne ou couloir, et le dépôt modifie le Status et le Rank de l'élément. Les éléments récapitulatifs et les occurrences modifiées ne sont pas affichés comme des cartes. Avec pdoReorder et sans pdoMove dans DragOptions, un glissement ne change que l'ordre des cartes d'une colonne. Il hérite de TsgcUIVCLControl et publie Align, Anchors, Constraints, Enabled, PopupMenu, ShowHint, ParentShowHint, TabOrder, TabStop et Visible.



Propriétés

Propriété	Type	Description
Data	TsgcPlannerData	Les données que le contrôle affiche. Plusieurs contrôles peuvent partager un même objet de données. Quand il vaut nil, le contrôle utilise son propre objet de données interne.
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom. utCustom part des couleurs claires.
Colors	TsgcPlannerViewColors	Remplacements des couleurs du thème : Background, Surface, Border, Grid, HeaderBackground, HeaderText, Text, TextMuted, Accent, Selection, NonWorking, NowLine, Critical et Ghost. 0 (par défaut) conserve la couleur du thème.
ReadOnly	Boolean	Désactive tous les glissements et la touche Suppr. Par défaut False.
DragOptions	TsgcPlannerDragOptions	Les glissements que l'utilisateur peut effectuer : pdoCreate, pdoMove, pdoResize, pdoProgress, pdoLink et pdoReorder. Par défaut, tous.
SyncSelection	Boolean	Partage la sélection avec les autres vues des mêmes données, via SelectedId des données. Par défaut True.

MaxFPS	Integer	Le plus grand nombre d'images par seconde que le contrôle dessine.
BorderStyle	TBorderStyle	bsSingle (par défaut) ou bsNone.
ColumnSource	TsgcPlannerKanbanColumnSource	pkcStatuses (par défaut) affiche une colonne par statut des données, pkcList n'affiche que la collection Columns.
Columns	TsgcPlannerKanbanColumns	Les colonnes : StatusId, Caption et Color (vide ou 0 prennent ceux du statut), WIPLimit (0 aucune), Collapsed et Width (0 par défaut). Avec pkcStatuses, une entrée ne fait qu'ajouter des paramètres à la colonne de son statut. AddColumn(aStatusId, aCaption = '', aWIPLimit = 0) et Find(aStatusId).
ShowUnknownColumn	Boolean	Affiche une colonne Autres pour les cartes dont le statut n'a pas de colonne. Par défaut True. On ne peut pas y déposer de carte.
SwimlaneBy	TsgcPlannerSwimlaneBy	pslNone (par défaut), pslResource (un couloir par ressource, un dépôt sur un autre couloir remplace la première ressource), pslParent (un couloir par parent, un dépôt définit ParentId), pslTag (un couloir par étiquette) ou pslEvent (OnGetSwimlane donne le couloir). Édition Registered.
WIPMode	TsgcPlannerWIPMode	pwmWarn (par défaut) colore seulement le badge de comptage d'une colonne qui atteint ou dépasse son WIPLimit, pwmBlock refuse aussi une carte qui dépasserait la limite. Le réordonnancement à l'intérieur d'une colonne est toujours autorisé. Édition Registered.
CardOptions	TsgcPlannerKanbanCardOptions	ShowTags, ShowAssignees, ShowDueDate et ShowProgress (par défaut True) et MaxTitleLines. L'échéance est le Finish de l'élément, affiché en rouge quand elle est dépassée et que la carte n'est pas terminée.
KanbanView	TsgcPlannerKanbanView	Public, en lecture seule. Le modèle de vue, avec les colonnes, les couloirs et les cartes.
View	TsgcPlannerView	Public, en lecture seule. Le modèle de vue que le contrôle héberge.

Méthodes

Méthode	Retourne	Description
CollapseColumn(aStatusId), ExpandColumn(aStatusId)	Boolean	Replie une colonne en un rail étroit ou la rouvre. L'utilisateur clique sur la flèche de l'en-tête.
SelectNextCard(aKey)	Boolean	Déplace la sélection avec un code de touche : les flèches se déplacent entre les cartes, les colonnes et les couloirs, Début et Fin vont à la première et à la dernière carte.
ItemAt(X, Y)	TsgcPlanneritem	L'élément situé sous un point du contrôle, nil s'il n'y en a pas.
ScrollToItem(aId)	Boolean	Défile jusqu'à ce que l'élément soit visible. False si l'élément n'est pas affiché.

SaveToPNG, SaveToSVG(aFileName ou aStream, aWidth = 0, aHeight = 0, aScale = 0)		La vue sous forme d'image, la même image qu'un TsgcPlannerRenderer avec les mêmes données, options, taille et échelle. La taille est en DIP, 0 est la taille client, et une échelle de 0 est l'échelle du contrôle. La position de défilement est conservée. Les barres de défilement, l'aperçu du glissement et les repères de dépôt sont omis.

Événements

Événement	Type	Description
OnItemClick, OnItemDblClick	TsgcPlannerItemEvent	Un clic ou un double-clic sur un élément : aItem.
OnItemDelete	TsgcPlannerItemAcceptEvent	La touche Suppr sur l'élément sélectionné : aItem et var aAccept. Mettez aAccept à False pour conserver l'élément. Un élément supprimé emporte ses liens, et ses enfants passent à son parent.
OnSelectionChange	TNotifyEvent	L'élément sélectionné a changé, dans ce contrôle ou, avec SyncSelection, dans une autre vue des mêmes données.
OnCardMoving	TsgcPlannerKanbanCardMovingEvent	Avant un dépôt : aItem, aToStatus, aToLane et var aAccept. Déclenché avant OnItemChanging des données.
OnCardMoved	TsgcPlannerItemEvent	Après la validation d'un dépôt : aItem avec son nouveau statut et son nouveau rang.
OnGetSwimlane	TsgcPlannerKanbanSwimlaneEvent	Avec pslEvent : aItem et var aKey et aCaption, le couloir de la carte.
OnColumnCollapse	TsgcPlannerKanbanColumnEvent	Une colonne a été repliée ou ouverte : aStatusId et aCollapsed.

Exemple : une limite WIP, des couloirs par personne et une règle pour Done

```
uses
  SysUtils, Classes, Dialogs,
  sgcPlanner_Data, sgcPlanner_Kanban, sgcPlanner_Kanban_VCL;

type
  TBoardEvents = class
  public
    procedure CardMoving(Sender: TObject; aItem: TsgcPlannerItem;
      const aToStatus, aToLane: string; var aAccept: Boolean);
    procedure CardMoved(Sender: TObject; aItem: TsgcPlannerItem);
  end;

procedure TBoardEvents.CardMoving(Sender: TObject; aItem: TsgcPlannerItem;
  const aToStatus, aToLane: string; var aAccept: Boolean);
begin
  // only finished work goes to Done
  if (aToStatus = 'done') and (aItem.Progress < 100) then
    aAccept := False;
end;
```

```
procedure TBoardEvents.CardMoved(Sender: TObject; aItem: TsgcPlannerItem);
begin
  ShowMessage(Format('%s is now %s', [aItem.Title, aItem.Status]));
end;

procedure SetupBoard(aKanban: TsgcVCLKanban; aEvents: TBoardEvents);
begin
  // one column per status, In progress takes at most 3 cards
  aKanban.ColumnSource := pkcStatuses;
  aKanban.Columns.AddColumn('doing', '', 3);
  aKanban.WIPMode := pwmBlock;
  // one lane per person (Registered edition)
  aKanban.SwimlaneBy := pslResource;
  aKanban.CardOptions.MaxTitleLines := 2;
  aKanban.CardOptions.ShowDueDate := True;
  aKanban.OnCardMoving := aEvents.CardMoving;
  aKanban.OnCardMoved := aEvents.CardMoved;
end;
```

Voir aussi

- TsgcFMXKanban
- TsgcPlannerData
- Démarrage rapide VCL
- Couloirs et limites WIP
- Glisser-déposer
- Thèmes

TsgcFMXKanban

Unité : sgcPlanner_Kanban_FMX
TsgcFMXKanban est le tableau Kanban FireMonkey, à partir de Delphi XE6 (SGC_PLANNER_FMX). Il a le même tableau, les mêmes propriétés, méthodes et événements que TsgcVCLKanban. Il dessine ses propres barres de défilement et traite la saisie tactile comme une saisie à la souris, de sorte qu'un doigt fait glisser une carte. Il hérite de TsgcUIFMXControl et publie les propriétés et événements de disposition habituels de FireMonkey.

Propriétés

Propriété	Type	Description
Data	TsgcPlannerData	Les données que le contrôle affiche. Plusieurs contrôles peuvent partager un même objet de données. Quand il vaut nil, le contrôle utilise son propre objet de données interne.
Theme	TsgcUITheme	utLight (par défaut), utDark ou utCustom. utCustom part des couleurs claires.
Colors	TsgcPlannerViewColors	Remplacements des couleurs du thème : Background, Surface, Border, Grid, HeaderBackground, HeaderText, Text, TextMuted, Accent, Selection, NonWorking, NowLine, Critical et Ghost. 0 (par défaut) conserve la couleur du thème.
ReadOnly	Boolean	Désactive tous les glissements et la touche Suppr. Par défaut False.
DragOptions	TsgcPlannerDragOptions	Les glissements que l'utilisateur peut effectuer : pdoCreate, pdoMove, pdoResize, pdoProgress, pdoLink et pdoReorder. Par défaut, tous.
SyncSelection	Boolean	Partage la sélection avec les autres vues des mêmes données, via SelectedId des données. Par défaut True.
MaxFPS	Integer	Le plus grand nombre d'images par seconde que le contrôle dessine.
ColumnSource	TsgcPlannerKanbanColumnSource	pkcStatuses (par défaut) affiche une colonne par statut des données, pkcList n'affiche que la collection Columns.
Columns	TsgcPlannerKanbanColumns	Les colonnes : StatusId, Caption et Color (vide ou 0 prennent ceux du statut), WIPLimit (0 aucune), Collapsed et Width (0 par défaut). Avec pkcStatuses, une entrée ne fait qu'ajouter des paramètres à la colonne de son statut. AddColumn(aStatusId, aCaption = '', aWIPLimit = 0) et Find(aStatusId).
ShowUnknownColumn	Boolean	Affiche une colonne Autres pour les cartes dont le statut n'a pas de colonne. Par défaut True. On ne peut pas y déposer de carte.
SwimlaneBy	TsgcPlannerSwimlaneBy	pslNone (par défaut), pslResource (un couloir par ressource, un dépôt sur un autre couloir remplace la première ressource), pslParent (un couloir par parent, un dépôt définit ParentId), pslTag (un couloir par étiquette) ou pslEvent (OnGetSwimlane donne le couloir). Édition Registered.
WIPMode	TsgcPlannerWIPMode	pwmWarn (par défaut) colore seulement le badge de comptage d'une colonne qui atteint ou dépasse son WIPLimit, pwmBlock refuse aussi une carte qui dépasserait la limite. Le réordonnancement à l'intérieur d'une colonne est toujours autorisé. Édition Registered.

CardOptions	TsgcPlannerKanbanCardOptions	ShowTags, ShowAssignees, ShowDueDate et ShowProgress (par défaut True) et MaxTitleLines. L'échéance est le Finish de l'élément, affiché en rouge quand elle est dépassée et que la carte n'est pas terminée.
KanbanView	TsgcPlannerKanbanView	Public, en lecture seule. Le modèle de vue, avec les colonnes, les couloirs et les cartes.
View	TsgcPlannerView	Public, en lecture seule. Le modèle de vue que le contrôle héberge.

Méthodes

Méthode	Retourne	Description
CollapseColumn(aStatusId), ExpandColumn(aStatusId)	Boolean	Replie une colonne en un rail étroit ou la rouvre. L'utilisateur clique sur la flèche de l'en-tête.
SelectNextCard(aKey)	Boolean	Déplace la sélection avec un code de touche : les flèches se déplacent entre les cartes, les colonnes et les couloirs, Début et Fin vont à la première et à la dernière carte.
ItemAt(X, Y (Single))	TsgcPlannerItem	L'élément situé sous un point du contrôle, nil s'il n'y en a pas.
ScrollToItem(aId)	Boolean	Défile jusqu'à ce que l'élément soit visible. False si l'élément n'est pas affiché.
SaveToPNG, SaveToSVG(aFileName ou aStream, aWidth = 0, aHeight = 0, aScale = 0)		La vue sous forme d'image, la même image qu'un TsgcPlannerRenderer avec les mêmes données, options, taille et échelle. La taille est en DIP, 0 est la taille client, et une échelle de 0 est l'échelle du contrôle. La position de défilement est conservée. Les barres de défilement, l'aperçu du glissement et les repères de dépôt sont omis.

Événements

Événement	Type	Description
OnItemClick, OnItemDblClick	TsgcPlannerItemEvent	Un clic ou un double-clic sur un élément : aItem.
OnItemDelete	TsgcPlannerItemAcceptEvent	La touche Suppr sur l'élément sélectionné : aItem et var aAccept. Mettez aAccept à False pour conserver l'élément. Un élément supprimé emporte ses liens, et ses enfants passent à son parent.
OnSelectionChange	TNotifyEvent	L'élément sélectionné a changé, dans ce contrôle ou, avec SyncSelection, dans une autre vue des mêmes données.
OnCardMoving	TsgcPlannerKanbanCardMovingEvent	Avant un dépôt : aItem, aToStatus, aToLane et var aAccept. Déclenché avant OnItemChanging des données.

OnCardMoved	TsgcPlannerItemEvent	Après la validation d'un dépôt : aItem avec son nouveau statut et son nouveau rang.
OnGetSwimlane	TsgcPlannerKanbanSwimlaneEvent	Avec pslEvent : aItem et var aKey et aCaption, le couloir de la carte.
OnColumnCollapse	TsgcPlannerKanbanColumnEvent	Une colonne a été repliée ou ouverte : aStatusId et aCollapsed.

Exemple : une liste fixe de colonnes et une colonne Done repliée

```
uses
  System.SysUtils, System.Classes, FMX.Types, FMX.Controls,
  sgcPlanner_Data, sgcPlanner_Kanban, sgcPlanner_Kanban_FMX;

procedure SetupBoard(aKanban: TsgcFMXKanban; aData: TsgcPlannerData);
begin
  aKanban.Align := TAlignLayout.Client;
  aKanban.Data := aData;
  // a fixed list of columns instead of the statuses
  aKanban.ColumnSource := pkcList;
  aKanban.Columns.Clear;
  aKanban.Columns.AddColumn('todo', 'Backlog');
  aKanban.Columns.AddColumn('doing', 'This week', 5);
  aKanban.Columns.AddColumn('done', 'Shipped');
  aKanban.ShowUnknownColumn := False;
  aKanban.CardOptions.ShowTags := True;
  // start with Shipped folded into a rail
  aKanban.CollapseColumn('done');
end;
```

Voir aussi

- TsgcVCLKanban
- Démarrage rapide FMX
- Couloirs et limites WIP
- Glisser-déposer

TsgcPlannerSource_DataSet

Unité : sgcPlanner_Source_DataSet
TsgcPlannerSource_DataSet lie un TsgcPlannerData à des TDataSet : DataSet contient les éléments, ResourceDataSet les ressources et DependencyDataSet les dépendances. Il n'a besoin que de l'unité DB, donc il fonctionne avec n'importe quel jeu de données, en VCL, en FireMonkey ou dans un service. Il fait partie de toutes les éditions.

Chargement

- Active à True charge chaque DataSet ouvert, et l'ouverture d'un DataSet pendant que Active est vrai le charge. Un chargement remplace les éléments, les ressources et les dépendances en une seule mise à jour, de sorte que chaque vue liée reçoit une seule notification. L'enregistrement courant de chaque DataSet est restauré ensuite.
- Les dates proviennent de champs date et heure, de champs numériques ou de texte ISO 8601. Les booléens proviennent de champs booléens ou numériques ou de texte (1, T, True, Y, Yes). Les couleurs proviennent de champs numériques (ARGB) ou de texte (#RRGGBB, #AARRGGBB, $AARRGGBB). Une valeur nulle donne la valeur par défaut, et un Finish nul donne Finish = Start, ou le minuit suivant pour un élément sur toute la journée.
- Sans DependencyDataSet, DependsOnField contient les Id, séparés par des virgules, des prédécesseurs de l'élément, lus comme des liens fin à début.

Mises à jour en direct et réécriture

- Avec LiveUpdates, une modification ou une insertion validée met à jour ou ajoute cet élément, et un changement du nombre d'enregistrements (une suppression, un filtre, un rafraîchissement) synchronise les éléments par Id. Le défilement et les modifications non validées ne changent rien.
- Avec WriteBack, une modification dans une vue ou dans le code est écrite dans le DataSet : l'enregistrement est trouvé avec Locate sur le champ Id, les champs mappés modifiés sont affectés et l'enregistrement est validé aussitôt. Un nouvel élément ajoute un enregistrement et un élément supprimé supprime son enregistrement. Les ressources ne sont jamais écrites, pas plus que les champs calculés, de recherche et en lecture seule. Rien n'est écrit tant que le DataSet est en état d'édition ou d'insertion ; les changements en attente sont écrits quand il revient à l'état de navigation.

Propriétés

Propriété	Type	Description
Data	TsgcPlannerData	Les données à remplir.
DataSet	TDataSet	Les éléments, un enregistrement par élément.
ItemFields	TsgcPlannerSourceItemFields	Le champ de chaque valeur de l'élément : IdField, TitleField, StartField, FinishField, AllDayField, StatusField, ProgressField, ColorField, ParentField, TagsField, ResourcesField, RankField, NotesField, RecurrenceField et DependsOnField. Les valeurs par défaut sont Id, Title, Start, Finish, AllDay, Status, Progress, Color, ParentId, Tags, Resources, Rank, Notes, RecurrenceRule et DependsOn. Un nom vide ou introuvable dans le DataSet est ignoré.
ResourceDataSet, ResourceFields	TDataSet, TsgcPlannerSourceResourceFields	Les ressources : IdField, NameField, ColorField et InitialsField, par défaut Id, Name, Color et Initials. En lecture seule.

DependencyDataSet, DependencyFields	TDataSet, TsgcPlannerSourceDependencyFields	Les liens : FromField, ToField, KindField (FS, SS, FF, SF ou l'ordinal) et LagField (minutes de travail), par défaut FromId, ToId, Kind et Lag.
Active	Boolean	Charge les DataSet ouverts et les suit. Par défaut False.
LiveUpdates	Boolean	Suit les validations, les insertions, les suppressions et les rafraîchissements des DataSet. Par défaut True.
WriteBack	Boolean	Écrit les changements des données dans DataSet et DependencyDataSet. Par défaut True.

Méthodes

Méthode	Retourne	Description
Load		Charge maintenant chaque DataSet ouvert, que Active soit vrai ou non. Il ne se passe rien quand Data vaut nil.
Refresh		Recharge quand Active est vrai.

Événements

Événement	Type	Description
OnLoadItem	TsgcPlannerSourceLoadItemEvent	Après la copie des champs mappés d'un enregistrement dans son élément : aDataSet, positionné sur cet enregistrement, et aItem, afin que l'application puisse remplir d'autres propriétés de l'élément.

Exemple : des tâches et des personnes issues de deux tables

```
uses
  SysUtils, Classes, DB,
  sgcPlanner_Data, sgcPlanner_Source_DataSet;

type
  TSourceEvents = class
  public
    procedure LoadItem(Sender: TObject; aDataSet: TDataSet;
      aItem: TsgcPlannerItem);
  end;

procedure TSourceEvents.LoadItem(Sender: TObject; aDataSet: TDataSet;
  aItem: TsgcPlannerItem);
begin
  // a column the field maps do not cover
  if aDataSet.FieldByName('Priority').AsInteger > 2 then
    aItem.Tags := 'urgent';
end;

function BindTasks(aOwner: TComponent; aData: TsgcPlannerData;
  aTasks, aPeople: TDataSet; aEvents: TSourceEvents)
  : TsgcPlannerSource_DataSet;
```

```
begin
  Result := TsgcPlannerSource_DataSet.Create(aOwner);
  Result.Data := aData;
  // the task table uses its own column names
  Result.DataSet := aTasks;
  Result.ItemFields.IdField := 'TaskId';
  Result.ItemFields.TitleField := 'Name';
  Result.ItemFields.StartField := 'StartDate';
  Result.ItemFields.FinishField := 'DueDate';
  Result.ItemFields.ResourcesField := 'Owner';
  // the people table matches the default names except one
  Result.ResourceDataSet := aPeople;
  Result.ResourceFields.NameField := 'FullName';
  Result.OnLoadItem := aEvents.LoadItem;
  Result.WriteBack := True;
  Result.Active := True;
end;
```

Voir aussi

- TsgcPlannerData
- Démarrage rapide VCL
- Liaison DataSet

TsgcPlannerICalendar

Unité : sgcPlanner_ICal
TsgcPlannerICalendar importe et exporte les VEVENT des fichiers iCalendar (RFC 5545), le format de Google Calendar, d'Outlook et d'Apple Calendar, vers et depuis un TsgcPlannerData. Il fait partie de l'édition Registered (SGC_PLANNER_ADVANCED).

- L'Id d'un élément est l'UID de son événement. Un événement avec un RECURRENCE-ID devient une occurrence modifiée de sa série.
- Les heures en UTC (Z) et avec un TZID sont converties à l'heure locale, grâce au VTIMEZONE du fichier. Un TZID inconnu donne un avertissement et est lu comme heure locale.
- STATUS correspond au Status de l'élément, RESOURCES à ses ressources (les ressources manquantes sont créées), RELATED-TO avec RELTYPE=PARENT à ParentId, et RRULE et EXDATE à la récurrence. Les autres champs de l'élément voyagent dans des propriétés X-SGC, de sorte qu'un export suivi d'un import redonne les mêmes données. Les dépendances ne sont pas exportées.
- Un texte sans BEGIN:VCALENDAR lève EsgcPlannerError et laisse les données inchangées. Un import est une seule mise à jour, donc les vues reçoivent une seule notification.

Propriétés

Propriété	Type	Description
Data	TsgcPlannerData	Les données dans lesquelles importer ou à partir desquelles exporter.
ImportMode	TsgcPlannerICalImportMode	pimReplace (par défaut) supprime d'abord les éléments et les dépendances et conserve les ressources, les statuts, le calendrier et ProjectStart. pimMerge met à jour les éléments qui ont le même Id et ajoute les autres.
ExportUTC	Boolean	Écrit en UTC les heures des événements qui ne se répètent pas. Les séries et les occurrences modifiées sont toujours écrites en heures locales flottantes, qui restent justes lors des changements d'heure d'été. Par défaut True.
ProductId	string	Le PRODID du calendrier exporté. Par défaut `-//eSeGeCe//sgcPlanner//EN`.
WarningCount	Integer	Public, en lecture seule. Les avertissements du dernier import.

Méthodes

Méthode	Retourne	Description
ImportFromFile, ImportFromStream, ImportFromString		Lit un calendrier. Les fichiers et les flux sont en UTF-8, avec ou sans BOM. Un UTF-8 invalide est lu comme du Latin-1, avec un avertissement.
ExportToFile, ExportToStream		Écrit les données sous forme d'un seul calendrier, en UTF-8.
ExportToString	string	Les données sous forme de texte iCalendar.

Événements

Événement	Type	Description
OnImport-Warning	TsgcPlannerI-CalWarnin-gEvent	Une partie du fichier a été ignorée ou lue approximativement : aLine et aMessage, par exemple une RRULE hors du sous-ensemble pris en charge, un RDATE ou un fuseau horaire inconnu.

Exemple : fusionner un calendrier et le réécrire

```
uses
  SysUtils, Classes,
  sgcPlanner_Data, sgcPlanner_ICal;

type
  TICalLog = class
  public
    Lines: TStrings;
    procedure ImportWarning(Sender: TObject; aLine: Integer;
      const aMessage: string);
  end;

procedure TICalLog.ImportWarning(Sender: TObject; aLine: Integer;
  const aMessage: string);
begin
  Lines.Add(Format('line %d: %s', [aLine, aMessage]));
end;

procedure SyncCalendar(aData: TsgcPlannerData; aLog: TICalLog;
  const aFileName: string);
var
  oICal: TsgcPlannerICalendar;
begin
  oICal := TsgcPlannerICalendar.Create(nil);
  try
    oICal.Data := aData;
    // update the events with the same UID, add the new ones
    oICal.ImportMode := pimMerge;
    oICal.OnImportWarning := aLog.ImportWarning;
    oICal.ImportFromFile(aFileName);
    // and write the whole data back as one calendar
    oICal.ExportUTC := True;
    oICal.ExportToFile(ChangeFileExt(aFileName, '.out.ics'));
  finally
    oICal.Free;
  end;
end;
```

Voir aussi

- TsgcPlannerMSProject
- TsgcPlannerData
- Éditions
- iCalendar
- Récurrence

TsgcPlannerMSProject

Unité : sgcPlanner_MSProject
TsgcPlannerMSProject importe et exporte les fichiers XML Microsoft Project (MSPDI), le format XML que Microsoft Project enregistre et ouvre, vers et depuis un TsgcPlannerData. Il fait partie de l'édition Registered (SGC_PLANNER_ADVANCED).

- Les tâches deviennent des éléments : l'Id est l'UID, le plan donne ParentId, et les dates, la progression, le jalon, l'indicateur manuel, la contrainte et la ligne de base suivent le fichier. Les liens de prédécesseur deviennent des dépendances avec leur type et leur décalage, et les affectations donnent les ressources de chaque tâche.
- Le calendrier du projet donne les jours ouvrés, les heures de travail et les jours fériés, et la date de début du projet donne ProjectStart.
- Seules les contraintes ASAP et début au plus tôt existent dans sgcPlanner. Doit commencer le est lu comme début au plus tôt, et les autres types de contrainte comme ASAP, chacun avec un avertissement.
- L'export écrit un calendrier Standard, les tâches dans l'ordre du plan, les liens et les ressources. Le statut, les étiquettes, la couleur, le rang, la journée entière et la récurrence ne sont pas écrits.
- Un document mal formé lève EsgcPlannerError avec la ligne, et laisse les données inchangées.

Propriétés

Propriété	Type	Description
Data	TsgcPlannerData	Les données dans lesquelles importer ou à partir desquelles exporter.
ImportMode	TsgcPlannerMSProjectImportMode	pmiReplace (par défaut) supprime les éléments et les liens et reprend le calendrier et le début du projet du fichier. Les ressources et les statuts sont conservés, et les ressources du fichier sont ajoutées. pmiMerge conserve le calendrier et ProjectStart, met à jour les éléments qui ont le même Id et ajoute les autres.
WarningCount	Integer	Public, en lecture seule. Les avertissements du dernier import.

Méthodes

Méthode	Retourne	Description
ImportFromFile, ImportFromStream, ImportFromString		Lit un projet. Les fichiers et les flux sont en UTF-8, avec ou sans BOM.
ExportToFile, ExportToStream		Écrit les données sous forme de projet.
ExportToString	string	Les données sous forme de texte MSPDI.

Événements

Événement	Type	Description
OnImportWarning	TsgcPlannerMSProjectWarningEvent	Une partie du fichier a été lue approximativement : aLine et aMessage.

Exemple : ouvrir et enregistrer un fichier Microsoft Project

```
uses
  SysUtils, Classes, Dialogs,
  sgcPlanner_Data, sgcPlanner_MSProject;

procedure OpenProject(aData: TsgcPlannerData; const aFileName: string);
var
  oProject: TsgcPlannerMSProject;
begin
  oProject := TsgcPlannerMSProject.Create(nil);
  try
    oProject.Data := aData;
    // tasks, links, calendar and project start from the file
    oProject.ImportMode := pmiReplace;
    oProject.ImportFromFile(aFileName);
    if oProject.WarningCount > 0 then
      ShowMessage(Format('%d warnings', [oProject.WarningCount]));
  finally
    oProject.Free;
  end;
end;

procedure SaveProject(aData: TsgcPlannerData; const aFileName: string);
var
  oProject: TsgcPlannerMSProject;
begin
  oProject := TsgcPlannerMSProject.Create(nil);
  try
    oProject.Data := aData;
    oProject.ExportToFile(aFileName);
  finally
    oProject.Free;
  end;
end;
```

Voir aussi

- TsgcVCLGantt
- TsgcPlannerICalendar
- TsgcPlannerData
- XML MS Project
- Lignes de base

TsgcPlannerRenderer

Unité : sgcPlanner_Renderer
TsgcPlannerRenderer dessine une vue Scheduler, Gantt ou Kanban d'un TsgcPlannerData sans fenêtre, sur n'importe quel TsgcUICanvas, dans un PNG ou dans un SVG, de sorte qu'un serveur, un rapport ou une application console donne la même image qu'une fiche. Il possède un modèle de vue de chaque type, les mêmes classes que celles créées par les contrôles, si bien que la sortie correspond aux contrôles pixel pour pixel. Il est indépendant du framework et fonctionne sous Windows et Linux64, dans toutes les éditions.

Propriétés

Propriété	Type	Description
Data	TsgcPlannerData	Les données à dessiner. La sélection est le SelectedId des données.
View	TsgcPlannerRendererViewKind	prvScheduler (par défaut), prvGantt ou prvKanban.
Date	TDateTime	La date du Scheduler. 0 (par défaut) suit l'horloge.
Width, Height	Integer	La taille en DIP. Par défaut 800 x 600.
Scale	Double	Le facteur HiDPI. La sortie fait Round(Width * Scale) x Round(Height * Scale) pixels. Par défaut 1.
Theme, Colors	TsgcUITheme, TsgcPlannerViewColors	Comme sur les contrôles.
FontFile, FontFileBold	string	Fichiers TrueType servant à mesurer et à dessiner le texte. Vide (par défaut) utilise la police intégrée. Le Kanban retourne à la ligne les titres des cartes avec FontFile.
SchedulerOptions	TsgcPlannerRendererSchedulerOptions	View, Options et TimeScale du Scheduler, comme sur TsgcVCLScheduler.
GanttOptions	TsgcPlannerRendererGanttOptions	Columns, TreeWidth, RowHeight, TimeScale, DurationUnit, BarColorMode, ShowProgress, ShowToday, ShowDependencies, ShowCriticalPath et ShowBaseline, comme sur TsgcVCLGantt.
KanbanOptions	TsgcPlannerRendererKanbanOptions	ColumnSource, Columns, ShowUnknownColumn, SwimlaneBy, WIPMode et CardOptions, comme sur TsgcVCLKanban.
SchedulerView, GanttView, KanbanView		Public, en lecture seule. Les modèles de vue, pour ce que les options ne couvrent pas, comme ExpandAll, Collapse ou ScrollToTime. Leur taille suit Width et Height, donc un défilement défini avant le rendu est conservé.
ActiveView	TsgcPlannerView	Public, en lecture seule. Le modèle de vue de View.

Chaque classe d'options copie aussi les options d'un modèle de vue, de sorte que `GanttOptions.Assign(VCLGantt1.GanttView)` rend ce que ce contrôle affiche.

Méthodes

Méthode	Retourne	Description
RenderTo(aCanvas)		Dessine la vue sur un TsgcUICanvas de DeviceWidth x Device-Height pixels.
SaveToPNG(aFileName ou aStream)		La vue sous forme de PNG.
SaveToSVG(aFileName ou aStream)		La vue sous forme de SVG.
DeviceWidth, DeviceHeight	Integer	La taille de la sortie en pixels.

Événements

TsgcPlannerRenderer n'a pas d'événements. Les événements OnItemChanging et OnChange des données sont déclenchés comme d'habitude quand les données changent.

Exemple : un Gantt en SVG et un mois sombre en PNG HiDPI

```
uses
  SysUtils, Classes,
  sgcUICore_Theme, sgcPlanner_Data, sgcPlanner_Scheduler, sgcPlanner_Renderer;

procedure ExportViews(aData: TsgcPlannerData; const aFolder: string);
var
  oRenderer: TsgcPlannerRenderer;
begin
  oRenderer := TsgcPlannerRenderer.Create(nil);
  try
    oRenderer.Data := aData;
    oRenderer.Width := 1200;
    oRenderer.Height := 700;
    // the Gantt with every summary open, as SVG
    oRenderer.View := prvGantt;
    oRenderer.GanttOptions.ShowCriticalPath := True;
    oRenderer.GanttView.ExpandAll;
    oRenderer.SaveToSVG(IncludeTrailingPathDelimiter(aFolder) + 'gantt.svg');
    // the month of October in the dark theme, as a HiDPI PNG
    oRenderer.View := prvScheduler;
    oRenderer.Theme := utDark;
    oRenderer.Scale := 2;
    oRenderer.Date := EncodeDate(2026, 10, 1);
    oRenderer.SchedulerOptions.View := psvMonth;
    oRenderer.SaveToPNG(IncludeTrailingPathDelimiter(aFolder) + 'month.png');
  finally
    oRenderer.Free;
  end;
end;
```

Voir aussi

- Architecture
- TsgcPlannerData
- TsgcVCLGantt
- Rendu sans fenêtre

- Thèmes

Données partagées

Chaque vue de sgcPlanner affiche un TsgcPlannerData. L'objet de données contient tout le planificateur : les éléments, les ressources, les statuts, les dépendances, le calendrier de travail, ProjectStart et la sélection. Les vues n'en gardent aucune copie, elles le lisent, de sorte que le Gantt, le tableau Kanban et le Scheduler peuvent afficher les mêmes éléments en même temps et qu'un changement fait dans l'un d'eux, ou dans le code, parvient à tous.

Ce que contiennent les données

Partie	Contenu
Items	TsgcPlannerItem : Id, Title, Start, Finish, AllDay, Status, Progress, Color, ParentId, Tags, Resources, Rank, Notes, Milestone, Manual, Constraint, ConstraintDate, les dates de ligne de base et la récurrence. Un élément est une tâche dans le Gantt, une carte sur le tableau et un événement dans le Scheduler.
Resources	Personnes ou salles : Id, Name, Color et Initials. Un élément désigne ses ressources par Id dans sa liste Resources séparée par des virgules.
Statuses	Le workflow : Id, Caption et Color. Ce sont les colonnes du tableau, et elles peuvent colorer les barres et les événements.
Dependencies	Les liens entre les éléments : FromId, ToId, Kind et Lag. Voir Dépendances.
Calendar	Les jours ouvrés, les heures de travail et les jours fériés. Voir Calendrier de travail.
ProjectStart, SelectedId	Le début du projet pour le moteur de planification et l'élément sélectionné de chaque vue avec SyncSelection.

Un élément se trouve par son Id avec FindItem et IndexOfItem. ParentId construit l'arborescence des tâches : ChildrenOf('') retourne les racines, et un élément récapitulatif reprend ses dates et sa progression de ses feuilles avec UpdateSummaries. DeleteItem supprime les liens de l'élément et ses occurrences modifiées, et déplace ses enfants vers son parent. MoveItem change le statut et le rang d'une carte en un seul appel.

Notifications de changement

Chaque setter notifie. Entre BeginUpdate et EndUpdate, les changements sont collectés, et après le EndUpdate le plus externe, chaque écouteur reçoit un TsgcPlannerChange :

Champ	Signification
Kinds	L'ensemble des changements : pckItems (un champ d'un élément), pckStructure (un élément ajouté ou supprimé, un Id, un ParentId ou l'ordre modifié), pckResources, pckStatuses, pckDependencies, pckCalendar, pckSelection et pckSchedule (ProjectStart, ou les résultats TotalSlack et Critical du moteur).
ItemId	L'Id de l'élément quand un seul élément a changé, sinon vide.
Version	Un nombre qui augmente à chaque changement, sauf un changement de la seule sélection. Deux versions égales signifient les mêmes données.

Les vues sont des écouteurs. Un changement de la seule sélection ne redessine que leur surcouche, tout autre changement les remet en page à l'image suivante. Votre propre code écoute avec l'événement OnChange ou avec

AddListener, qui accepte un nombre quelconque de gestionnaires ; appelez RemoveListener avant que l'objet propriétaire d'un gestionnaire soit libéré. Un gestionnaire qui modifie de nouveau les données lance un tour de notifications supplémentaire, et après 8 tours EsgcPlannerError est levée, de sorte que deux gestionnaires qui se modifient sans fin ne bloquent pas l'application.

Exemple

```
uses
  SysUtils, Classes,
  sgcPlanner_Data, sgcPlanner_Gantt_VCL, sgcPlanner_Kanban_VCL;

type
  TChangeLog = class
  public
    Lines: TStrings;
    procedure DataChanged(Sender: TObject; const aChange: TsgcPlannerChange);
  end;

procedure TChangeLog.DataChanged(Sender: TObject;
  const aChange: TsgcPlannerChange);
begin
  // one call per outermost EndUpdate, however many items changed
  if pckStructure in aChange.Kinds then
    Lines.Add('items added or removed')
  else if aChange.ItemId <> '' then
    Lines.Add('item ' + aChange.ItemId + ' changed')
  else if pckItems in aChange.Kinds then
    Lines.Add('several items changed');
end;

procedure StartAll(aData: TsgcPlannerData; aGantt: TsgcVCLGantt;
  aKanban: TsgcVCLKanban; aLog: TChangeLog);
var
  i: Integer;
begin
  // two views, one data object
  aGantt.Data := aData;
  aKanban.Data := aData;
  aData.AddListener(aLog.DataChanged);
  aData.BeginUpdate;
  try
    for i := 0 to aData.Items.Count - 1 do
      if aData.Items[i].Status = 'todo' then
        aData.Items[i].Status := 'doing';
  finally
    // the log, the Gantt and the board get one notification here
    aData.EndUpdate;
  end;
end;

procedure StopLog(aData: TsgcPlannerData; aLog: TChangeLog);
begin
  // remove a listener before the object it belongs to is freed
  aData.RemoveListener(aLog.DataChanged);
end;
```

Voir aussi

- Vues synchronisées
- TsgcPlannerData
- Architecture

Vues synchronisées

Liez deux vues ou plus au même TsgcPlannerData et elles restent en phase. Une carte glissée vers Done sur le tableau est aussitôt dessinée avec la couleur Done dans le Gantt et dans le Scheduler, une barre déplacée dans le Gantt déplace l'événement dans le Scheduler, et un événement créé dans le Scheduler devient une carte. Chaque modification est validée en une seule mise à jour, de sorte que chaque vue se redessine une fois par dépôt.

Sélection partagée

Avec SyncSelection, à True par défaut, la sélection d'une vue est le SelectedId des données. Cliquez sur une barre et le même élément est sélectionné sur le tableau et dans le Scheduler. Avec SyncSelection à False, la vue garde sa propre sélection. Définissez SelectedId dans le code pour sélectionner un élément dans chaque vue synchronisée. OnSelectionChange est déclenché pour une sélection faite dans le contrôle ou, avec SyncSelection, dans une autre vue des mêmes données, ce qui est l'endroit pour faire défiler les autres vues jusqu'à l'élément avec ScrollToItem.

Des vues sur plusieurs fiches

Placez le TsgcPlannerData dans un module de données et affectez-le aux vues de chaque fiche. Les vues d'une fiche qui n'est pas visible sont aussi mises à jour, et dessinent l'état courant quand elles sont affichées.

La même image hors de la fiche

Les options d'une vue peuvent être copiées vers un TsgcPlannerRenderer avec Assign, par exemple `Renderer.GanttOptions.Assign(VCLGantt1.GanttView)`, de sorte qu'un rapport montre ce que montre la fiche. Les contrôles s'enregistrent aussi eux-mêmes en PNG et en SVG. Voir Rendu sans fenêtre.

Exemple

```
uses
  SysUtils, Classes,
  sgcPlanner_Data, sgcPlanner_Gantt_VCL, sgcPlanner_Scheduler_VCL;

type
  TSyncEvents = class
  public
    Gantt: TsgcVCLGantt;
    Scheduler: TsgcVCLScheduler;
    procedure GanttSelectionChange(Sender: TObject);
  end;

procedure TSyncEvents.GanttSelectionChange(Sender: TObject);
var
  oItem: TsgcPlannerItem;
begin
  // raised for a selection made in any view of the same data
  oItem := Gantt.Data.FindItem(Gantt.Data.SelectedId);
  if oItem = nil then
    Exit;
  Gantt.ScrollToItem(oItem.Id);
  // show the week of the item in the Scheduler, then the item itself
  Scheduler.Date := oItem.Start;
  Scheduler.ScrollToItem(oItem.Id);
end;

procedure SyncViews(aData: TsgcPlannerData; aEvents: TSyncEvents);
begin
```

```
  aEvents.Gantt.Data := aData;
  aEvents.Scheduler.Data := aData;
  aEvents.Gantt.SyncSelection := True;
  aEvents.Scheduler.SyncSelection := True;
  aEvents.Gantt.OnSelectionChange := aEvents.GanttSelectionChange;
  // a selection made in code reaches every view with SyncSelection
  aData.SelectedId := 'design';
end;
```

Voir aussi

- Données partagées
- Glisser-déposer
- Démarrage rapide VCL

Calendrier de travail

TsgcPlannerData.Calendar est le calendrier de travail du planificateur : les jours de la semaine travaillés, les heures de travail d'une journée et les jours fériés. Les durées, les décalages, les déplacements du Gantt et le moteur de planification comptent des minutes de travail dessus, de sorte qu'une tâche de deux jours ouvrés qui commence un vendredi se termine le lundi soir.

Propriétés

Propriété	Description
WorkDays	Les jours travaillés de la semaine, un ensemble de pwdMonday à pwdSunday. Par défaut du lundi au vendredi. Un ensemble vide signifie que tous les jours sont ouvrés.
WorkTime	Les heures de travail d'un jour ouvré sous forme de 4 intervalles au plus, `hh:nn-hh:nn` séparés par des virgules. Par défaut `08:00-12:00,13:00-17:00`. Un texte vide signifie la journée entière. Un texte invalide lève EsgcPlannerError, un texte valide est stocké sous forme canonique.
Holidays	Dates non travaillées, chacune avec un Name. AddHoliday(Date, Name) en ajoute une. Un jour férié un jour de repos est ignoré.
FirstDayOfWeek	Le premier jour d'une semaine dans le Scheduler et dans les graduations de semaine de l'échelle de temps. Par défaut pwdMonday.

Arithmétique du temps de travail

Les méthodes prennent et retournent un TsgcPlannerTime, le nombre exact de minutes en Int64 du modèle. sgcPlannerTimeOf et sgcPlannerDateTimeOf convertissent depuis et vers TDateTime.

Méthode	Retourne
IsWorkingDay(aTime), IsWorkingTime(aTime)	Si le jour, ou la minute, est travaillé.
WorkingMinutesPerDay	Les minutes de travail d'un jour ouvré, 480 avec le WorkTime par défaut.
NextWorkingTime(aTime), PrevWorkingTime(aTime)	La première minute de travail à aTime ou après, ou la dernière avant.
AddWorkingMinutes(aTime, aMinutes)	L'heure après aMinutes minutes de travail. Un nombre positif retourne la fin de la dernière minute de travail, donc lundi 08:00 plus 480 donne lundi 17:00, et non mardi 08:00. Un nombre négatif remonte dans le temps et retourne le début de la dernière minute.
WorkingMinutesBetween(aStart, aFinish)	Les minutes de travail de [aStart, aFinish), négatives quand aFinish précède aStart.
GetNonWorkingIntervals(aStart, aFinish, aList)	Les nuits, les jours de repos et les jours fériés d'une période, tels que l'échelle de temps les dessine.

Où le calendrier est utilisé

- Le Gantt affiche les durées en temps de travail (DurationUnit), conserve la durée de travail d'une barre déplacée et grise le temps non travaillé quand TimeScale.ShowNonWorking est à True.
- Le Lag d'une dépendance est en minutes de travail, et le moteur de planification place chaque tâche sur du temps de travail. Voir Planification automatique.
- UpdateSummaries pondère la progression de chaque feuille par sa durée de travail.
- La vue semaine de travail du Scheduler affiche les jours de WorkDays.

Un changement du calendrier notifie pckCalendar, de sorte que les vues se redessinent, et il est enregistré avec le JSON et le XML MS Project.

Exemple

```
uses
  SysUtils, Classes,
  sgcPlanner_Const, sgcPlanner_DateUtils, sgcPlanner_Data;

procedure SetupCalendar(aData: TsgcPlannerData);
begin
  // Monday to Thursday, two shifts and the company holidays
  aData.Calendar.WorkDays := [pwdMonday, pwdTuesday, pwdWednesday,
    pwdThursday];
  aData.Calendar.WorkTime := '07:00-13:00,14:00-16:00';
  aData.Calendar.Holidays.AddHoliday(EncodeDate(2026, 12, 8),
    'Immaculate Conception');
  aData.Calendar.Holidays.AddHoliday(EncodeDate(2026, 12, 25), 'Christmas');
end;

function PlanTask(aData: TsgcPlannerData; aItem: TsgcPlannerItem;
  aWorkDays: Integer): Int64;
var
  vStart: TsgcPlannerTime;
begin
  // the first working minute at or after the requested start
  vStart := aData.Calendar.NextWorkingTime(aItem.StartTime);
  aItem.StartTime := vStart;
  // whole working days, the holidays and the days off are skipped:
  // Monday 7 December 07:00 plus 3 days ends on Thursday 10 December 16:00
  aItem.FinishTime := aData.Calendar.AddWorkingMinutes(vStart,
    aWorkDays * aData.Calendar.WorkingMinutesPerDay);
  // 1440 working minutes in the example above
  Result := aData.Calendar.WorkingMinutesBetween(aItem.StartTime,
    aItem.FinishTime);
end;
```

Voir aussi

- Planification automatique
- Dépendances
- Échelle de temps et zoom
- TsgcPlannerData

Récurrence

Édition : Registered. Dans l'édition Community, un élément récurrent n'est affiché qu'une fois, à son propre Start, et sa règle est conservée et enregistrée. Voir Fonctionnalités par édition.
Un élément qui a une RecurrenceRule est une série : il se répète aux dates d'une RRULE iCalendar (RFC 5545). Le Scheduler affiche chaque occurrence, GetOccurrences des données les retourne pour n'importe quelle période, et les fichiers iCalendar les importent et les exportent.

La règle

RecurrenceRule contient le texte de la règle, avec ou sans le préfixe `RRULE:`. Le Start et le Finish de l'élément sont la première occurrence, et chaque occurrence a la même durée.

Partie	Prise en charge
FREQ	DAILY, WEEKLY, MONTHLY ou YEARLY.
INTERVAL	Toutes les n périodes, par exemple toutes les deux semaines.
COUNT, UNTIL	Le nombre d'occurrences, ou la dernière date. UNTIL accepte une date, une date et heure locale ou une date et heure UTC.
BYDAY	Jours de la semaine, par exemple MO,WE,FR. Dans les règles MONTHLY et YEARLY, un jour peut avoir un ordinal de -5 à 5, donc 2TU est le deuxième mardi et -1FR le dernier vendredi. Avec DAILY, il filtre les jours.
BYMONTH, BYMONTHDAY	Les mois de l'année, et un jour du mois (MONTHLY et YEARLY). Une règle YEARLY avec BYDAY a besoin de BYMONTH.
WKST	Le premier jour de la semaine de la règle.

La première occurrence est toujours le Start de l'élément et compte pour COUNT. Une règle invalide fait afficher l'élément une seule fois. Pour vérifier une règle saisie par un utilisateur, appelez TryParse de TsgcPlannerRecurrence, qui retourne False et la raison.

Exceptions et occurrences modifiées

- **RecurrenceExceptions** est une liste, séparée par des virgules, de dates ou de dates et heures ISO. Une date supprime les occurrences de ce jour entier, une date et heure l'occurrence qui commence à cette minute. Une occurrence supprimée compte quand même pour COUNT.
- Une **occurrence modifiée** est un élément à part entière avec RecurrenceParentId défini à l'Id de la série et RecurrenceDate défini au début de l'occurrence qu'elle remplace. Elle peut avoir d'autres heures, un autre titre ou d'autres ressources, et l'occurrence générée à cette date n'est plus affichée.
- DeleteItem d'une série supprime aussi ses occurrences modifiées.

Modifier dans le Scheduler

Un dépôt sur une occurrence générée déclenche OnRecurrenceEdit avec la série, l'occurrence et une portée : prsOccurrence (par défaut) crée une occurrence modifiée, prsSeries déplace toute la série du même écart et prsCancel annule le dépôt. La touche Suppr sur une occurrence déclenche OnRecurrenceDelete avec les mêmes portées : prsOccurrence (par défaut) supprime cette seule occurrence, en ajoutant sa date aux exceptions de la série et en la supprimant s'il s'agit d'une occurrence modifiée, et prsSeries supprime toute la série.
Le moteur de planification conserve les dates d'une série et de ses occurrences modifiées. Le tableau Kanban n'affiche pas les occurrences modifiées sous forme de cartes.

Exemple

```
uses
  SysUtils, Classes,
  sgcPlanner_DateUtils, sgcPlanner_Data, sgcPlanner_Recurrence;

const
  CS_RULE = 'FREQ=WEEKLY;INTERVAL=2;BYDAY=TU;UNTIL=20261231';

function CheckRule(const aRule: string; out aReason: string): Boolean;
var
  oRule: TsgcPlannerRecurrence;
begin
  // validate a rule typed by the user before it is stored
  oRule := TsgcPlannerRecurrence.Create;
  try
    Result := oRule.TryParse(aRule, aReason);
  finally
    oRule.Free;
  end;
end;

procedure AddTeamSync(aData: TsgcPlannerData);
var
  oSeries, oMoved: TsgcPlannerItem;
  vReason: string;
begin
  if not CheckRule(CS_RULE, vReason) then
    raise Exception.Create(vReason);
  aData.BeginUpdate;
  try
    // every second Tuesday at 10:00 until the end of the year
    oSeries := aData.AddItem('sync');
    oSeries.Title := 'Team sync';
    oSeries.Start := EncodeDate(2026, 10, 6) + EncodeTime(10, 0, 0, 0);
    oSeries.Finish := EncodeDate(2026, 10, 6) + EncodeTime(11, 0, 0, 0);
    oSeries.RecurrenceRule := CS_RULE;
    // no meeting on 3 November
    oSeries.RecurrenceExceptions := '2026-11-03';
    // the meeting of 17 November moves to the afternoon
    oMoved := aData.AddItem('sync-1117');
    oMoved.Title := 'Team sync (afternoon)';
    oMoved.RecurrenceParentId := 'sync';
    oMoved.RecurrenceDate := EncodeDate(2026, 11, 17) + EncodeTime(10, 0, 0, 0);
    oMoved.Start := EncodeDate(2026, 11, 17) + EncodeTime(16, 0, 0, 0);
    oMoved.Finish := EncodeDate(2026, 11, 17) + EncodeTime(17, 0, 0, 0);
  finally
    aData.EndUpdate;
  end;
end;

function MeetingsOfQuarter(aData: TsgcPlannerData): Integer;
var
  vList: TsgcPlannerOccurrenceArray;
begin
  // 6: seven Tuesdays, minus 3 November, with 17 November at 16:00
  Result := aData.GetOccurrences(sgcPlannerEncodeDate(2026, 10, 1),
    sgcPlannerEncodeDate(2027, 1, 1), vList);
end;
```

Voir aussi

- TsgcVCLScheduler
- iCalendar
- TsgcPlannerData

- Calendrier de travail

Échelle de temps et zoom

Le Gantt et la Timeline de ressources du Scheduler dessinent le temps sur un axe horizontal décrit par un TsgcPlannerTimeScale, la propriété TimeScale du contrôle (GanttOptions.TimeScale et SchedulerOptions.TimeScale sur le renderer). Il convertit le temps en pixels, dessine deux niveaux de graduations avec des libellés, grise le temps non travaillé et aimante les glissements.

Niveaux de zoom

Zoom	Graduation mineure	Graduation majeure	Largeur par défaut	Libellés (mineur, majeur)
pz5Minutes	5 minutes	heure	20	`nn` , `ddd d mmm t`
pz15Minutes	15 minutes	heure	30	`nn` , `ddd d mmm t`
pzHours	heure	jour	40	`hh` , `dddd d mmmm yyyy`
pzDays (par défaut)	jour	semaine	32	`d` , `d mmm yyyy`
pzWeeks	semaine	mois	56	`d mmm` , `mmmm yyyy`
pzMonths	mois	année	60	`mmm` , `yyyy`
pzQuarters	trimestre	année	60	`"Q"q` , `yyyy`
pzYears	année	décennie	60	`yyyy` , `yyyy`

L'axe est linéaire dans le temps. Aux zooms larges, un mois compte 30 jours, un trimestre 91 et une année 365, de sorte que les graduations de mois de longueurs différentes ont des largeurs légèrement différentes.

Propriétés

Propriété	Description
Zoom	Le niveau de zoom. Par défaut pzDays.
TickWidth	La largeur d'une graduation mineure en DIP. 0 (par défaut) est la largeur du zoom dans le tableau ci-dessus.
Origin	L'heure à X = 0 de l'axe.
Snap	Le pas d'un déplacement ou d'un redimensionnement : psTick (par défaut, la graduation mineure jusqu'à pzDays et des jours entiers aux zooms plus larges), psNone, ps5Min, ps15Min, psHour ou psDay.
ShowNonWorking	Grise les nuits, les jours de repos et les jours fériés du calendrier de travail. Par défaut True.
MajorFormat, MinorFormat	Les libellés des deux niveaux. Vide utilise les formats du tableau ci-dessus.

Les jetons de format ne sont pas sensibles à la casse : `yyyy` et `yy` pour l'année, `q` pour le trimestre, `m` à `mmmm` pour le mois, `d` à `dddd` pour le jour, `w` et `ww` pour la semaine ISO 8601, `h` et `hh` pour l'heure de 0 à 23, `n` et `nn` pour les minutes et `t` pour l'heure sur l'horloge de la locale. Le texte entre apostrophes ou guillemets est copié tel quel. Les noms proviennent de sgcPlannerLocale.

Zoom

- ZoomIn et ZoomOut du Gantt vont d'un niveau plus fin ou plus large et retournent False en fin de plage. ZoomToFit choisit le zoom qui affiche tout le projet dans la largeur du diagramme.
- Ctrl et la molette de la souris zooment le Gantt et la Timeline à la position de la souris, de sorte que l'heure sous le curseur reste en place. Dans les vues jour, semaine de travail et semaine du Scheduler, ils changent à la place la hauteur des créneaux.
- ScrollToTime(aTime) fait défiler l'axe jusqu'à une heure.

Dans la Timeline de ressources, la période suit le zoom : un jour jusqu'à pzHours, 4 semaines avec pzDays, 3 mois avec pzWeeks, un an avec pzMonths, 3 ans avec pzQuarters et une décennie avec pzYears.

Exemple

```
uses
  SysUtils, Classes,
  sgcPlanner_TimeScale, sgcPlanner_Gantt_VCL;

procedure SetupTimeScale(aGantt: TsgcVCLGantt);
begin
  // one minor tick per week, 80 DIPs wide
  aGantt.TimeScale.Zoom := pzWeeks;
  aGantt.TimeScale.TickWidth := 80;
  // the month above, the ISO week number below
  aGantt.TimeScale.MajorFormat := 'mmmm yyyy';
  aGantt.TimeScale.MinorFormat := '"W"ww';
  // a move or a resize snaps to whole days
  aGantt.TimeScale.Snap := psDay;
  aGantt.TimeScale.ShowNonWorking := True;
  aGantt.ScrollToTime(Date);
end;

procedure ZoomClick(aGantt: TsgcVCLGantt; aZoomIn: Boolean);
begin
  // False when the zoom is already at the end of the range
  if aZoomIn then
    aGantt.ZoomIn
  else
    aGantt.ZoomOut;
end;
```

Voir aussi

- TsgcVCLGantt
- TsgcVCLScheduler
- Calendrier de travail
- Glisser-déposer

Glisser-déposer

Chaque vue modifie les données à la souris, au stylet ou au doigt. Un glissement démarre après que le pointeur s'est déplacé de 4 DIP, affiche un aperçu pendant son déroulement, fait défiler la vue quand il atteint un bord et valide la modification au relâchement du bouton. Échap ou un second bouton l'annule.

Les glissements de chaque vue

View	Glissements
Scheduler	Faites glisser sur un créneau vide pour créer un événement (OnItemCreate donne le titre ou le refuse). Déplacez un événement vers une autre heure, un autre jour ou une autre colonne de ressource, et redimensionnez-le par son bord haut ou bas. Les déplacements et redimensionnements s'alignent sur Options.SlotMinutes. Dans la bande journée entière et dans la vue mois, des jours entiers sont créés et déplacés. Dans la Timeline, une barre se déplace vers une autre heure ou une autre ligne de ressource et se redimensionne sur l'échelle.
Gantt	Déplacez une barre, redimensionnez son début ou sa fin, faites glisser la progression dans la moitié basse de la barre, tracez un lien depuis juste à l'extérieur d'une extrémité de barre vers une autre barre (Registered), et faites glisser le séparateur entre l'arborescence des tâches et le diagramme. Un déplacement conserve la durée de travail de la tâche.
Kanban	Déplacez une carte vers une autre position, une autre colonne ou un autre couloir, ou réordonnez les cartes d'une colonne.

Autoriser et refuser

Paramètre	Effet
ReadOnly	Désactive tous les glissements et la touche Suppr du contrôle.
DragOptions	Les glissements que le contrôle autorise : pdoCreate, pdoMove, pdoResize, pdoProgress, pdoLink et pdoReorder. Par défaut, tous. Sur le tableau, pdoReorder sans pdoMove garde chaque carte dans sa colonne.
OnItemChanging (données)	Déclenché une fois par dépôt, dans n'importe quelle vue, avec l'élément et les champs sur le point de changer. Mettez aAccept à False pour refuser la modification. C'est l'endroit unique pour une règle qui doit valoir dans toutes les vues.
OnCardMoving (Kanban)	Déclenché avant OnItemChanging avec le statut et le couloir cibles.
OnItemCreate (Scheduler)	Déclenché pour un nouvel événement, avec son début, sa fin et sa ressource.
OnLinkCreating (Gantt)	Déclenché pour un nouveau lien. Il peut changer le type ou refuser le lien.
OnRecurrenceEdit (Scheduler)	Déclenché pour un dépôt sur une occurrence d'une série. Voir Récurrence.
WIPMode pwmBlock (Kanban)	Empêche une carte d'entrer dans une colonne qui est à sa limite WIP. Voir Couloirs et limites WIP.

Ce que fait un dépôt

Un dépôt est un seul BeginUpdate et EndUpdate sur les données, de sorte que chaque vue se redessine une fois. Un déplacement ou un redimensionnement du début dans le Gantt définit aussi la contrainte de la tâche à pcSNET à la nouvelle date, sauf si la tâche est Manual, afin qu'une planification ultérieure l'y maintienne ; avec AutoSchedule, le moteur s'exécute après le dépôt. Quand l'élément a un parent, son récapitulatif est mis à jour.

Clavier et tactile

La touche Suppr déclenche OnItemDelete pour l'élément sélectionné et le supprime quand aAccept reste à True. Sur le tableau, SelectNextCard déplace la sélection avec les flèches, Début et Fin. La saisie tactile arrive comme une saisie à la souris, donc un doigt fait glisser comme une souris.

Exemple

```
uses
  SysUtils, Classes,
  sgcPlanner_Data, sgcPlanner_View, sgcPlanner_Gantt_VCL,
  sgcPlanner_Kanban_VCL, sgcPlanner_Scheduler_VCL;

type
  TDragRules = class
  public
    FrozenUntil: TDateTime;
    procedure ItemChanging(Sender: TObject; aItem: TsgcPlannerItem;
      const aFields: TsgcPlannerItemFields; var aAccept: Boolean);
  end;

procedure TDragRules.ItemChanging(Sender: TObject; aItem: TsgcPlannerItem;
  const aFields: TsgcPlannerItemFields; var aAccept: Boolean);
begin
  // work that started before FrozenUntil keeps its dates, in every view
  if ((pifStart in aFields) or (pifFinish in aFields)) and
    (aItem.Start < FrozenUntil) then
    aAccept := False;
end;

procedure ApplyRole(aData: TsgcPlannerData; aGantt: TsgcVCLGantt;
  aKanban: TsgcVCLKanban; aScheduler: TsgcVCLScheduler; aManager: Boolean;
  aRules: TDragRules);
begin
  aData.OnItemChanging := aRules.ItemChanging;
  if aManager then
  begin
    aGantt.DragOptions := [pdoMove, pdoResize, pdoProgress, pdoLink];
    aKanban.DragOptions := [pdoMove, pdoReorder];
    aScheduler.DragOptions := [pdoCreate, pdoMove, pdoResize];
  end
  else
  begin
    // a team member sees the plan, orders the cards of a column
    // and moves meetings, but creates and resizes nothing
    aGantt.ReadOnly := True;
    aKanban.DragOptions := [pdoReorder];
    aScheduler.DragOptions := [pdoMove];
  end;
end;
```

Voir aussi

- Vues synchronisées
- TsgcVCLGantt

- TsgcVCLKanban
- TsgcVCLScheduler

Dependencies

Une dépendance relie deux éléments : le successeur (ToId) attend le prédécesseur (FromId). Les liens sont stockés dans Dependencies du TsgcPlannerData dans toutes les éditions et enregistrés avec le JSON et le XML MS Project. Le Gantt les dessine sous forme de flèches, laisse l'utilisateur les tracer à la souris et les utilise pour planifier les tâches dans l'édition Registered.

Types et décalage

Type	Signification
pdkFS (par défaut)	Fin à début : le successeur commence quand le prédécesseur se termine.
pdkSS	Début à début : le successeur commence quand le prédécesseur commence.
pdkFF	Fin à fin : le successeur se termine quand le prédécesseur se termine.
pdkSF	Début à fin : le successeur se termine quand le prédécesseur commence.

Le décalage (Lag) est en minutes de travail du calendrier de travail, ajoutées à la borne. Un décalage négatif est une avance, de sorte que le successeur chevauche le prédécesseur. AddDependency(FromId, ToId, Kind, Lag) ajoute un lien en un seul appel.

Dans le Gantt

- ShowDependencies dessine chaque lien sous forme de flèche coudée entre les barres visibles. Les liens du chemin critique utilisent la couleur Critical.
- Faites glisser depuis juste à l'extérieur du début ou de la fin d'une barre et déposez sur une autre barre pour créer un lien. Les extrémités choisies donnent le type. OnLinkCreating peut changer le type ou refuser le lien. Un lien vers l'élément lui-même, entre un élément et son ancêtre ou son descendant, ou un lien qui existe déjà est refusé.
- Un lien sur une tâche récapitulative s'applique à chaque tâche feuille située en dessous quand le plan est planifié.

Garder les liens cohérents

DeleteItem supprime les liens de l'élément supprimé. Un changement d'Id ne renomme pas les liens qui l'utilisent, donc modifiez FromId et ToId de ces liens dans la même mise à jour. Les liens dont le FromId ou le ToId est inconnu sont conservés mais ignorés par le moteur et non dessinés. Un cycle de liens est signalé par le moteur. Voir Planification automatique.
La source DataSet lit les liens dans une table à part (DependencyDataSet) ou dans une liste de prédécesseurs séparés par des virgules de la table des tâches (DependsOnField). Voir Liaison DataSet.

Exemple

```
uses
  SysUtils, Classes,
  sgcPlanner_Data, sgcPlanner_View, sgcPlanner_Gantt_VCL;

procedure LinkTasks(aData: TsgcPlannerData);
begin
  aData.BeginUpdate;
  try
    // Build starts when Design finishes
```

```
    aData.Dependencies.AddDependency('design', 'build', pdkFS, 0);
    // Test starts one working day after Build starts
    aData.Dependencies.AddDependency('build', 'test', pdkSS,
      aData.Calendar.WorkingMinutesPerDay);
    // Docs finish two working hours after Test finishes
    aData.Dependencies.AddDependency('test', 'docs', pdkFF, 120);
    // a negative lag is a lead: Review starts 4 working hours
    // before Build finishes
    aData.Dependencies.AddDependency('build', 'review', pdkFS, -240);
  finally
    aData.EndUpdate;
  end;
end;

function Predecessors(aData: TsgcPlannerData; const aId: string;
  aList: TStrings): Integer;
var
  i: Integer;
begin
  aList.Clear;
  for i := 0 to aData.Dependencies.Count - 1 do
    if aData.Dependencies[i].ToId = aId then
      aList.Add(aData.Dependencies[i].FromId);
  Result := aList.Count;
end;

procedure ShowLinks(aGantt: TsgcVCLGantt);
begin
  // Registered edition: arrows between the bars and drag to link
  aGantt.ShowDependencies := True;
  aGantt.DragOptions := aGantt.DragOptions + [pdoLink];
end;
```

Voir aussi

- Planification automatique
- Chemin critique
- TsgcVCLGantt
- XML MS Project

Planification automatique

Édition : Registered. Dans l'édition Community, Schedule et SetBaseline du Gantt lèvent EsgcPlannerError, AutoSchedule n'a aucun effet et l'unité du moteur n'est pas incluse. Voir Fonctionnalités par édition.
Le moteur de planification, TsgcPlannerScheduleEngine de sgcPlanner_AutoSchedule, place les tâches au plus tôt sur le calendrier de travail après leurs prédécesseurs, calcule la marge de chaque tâche et marque le chemin critique. Le Gantt l'exécute avec Schedule, ou après chaque modification quand AutoSchedule est à True, et votre code peut l'exécuter sur un TsgcPlannerData sans aucune vue, par exemple dans un service.

Les règles

- Une tâche feuille conserve sa durée de travail, les minutes de travail entre son Start et son Finish, et commence à la première minute de travail située à la plus tardive des dates suivantes ou après elle : le début du projet, sa date de contrainte et les bornes de ses dépendances.
- Le début du projet est ProjectStart des données, ou le Start le plus ancien des tâches feuilles quand ProjectStart n'est pas défini.
- La contrainte pcSNET avec une ConstraintDate signifie début au plus tôt à cette date. pcASAP, la valeur par défaut, n'a pas de date. Le déplacement d'une barre dans le Gantt définit pcSNET à la nouvelle date, de sorte que la planification suivante garde la tâche là où elle a été placée.
- Une tâche Manual, un élément récurrent et chaque tâche d'un cycle de dépendances conservent leurs dates, et continuent de piloter leurs successeurs.
- Une tâche sans minute de travail est un jalon et se place exactement sur sa borne.
- Une tâche récapitulative prend le début le plus tôt et la fin la plus tardive de ses feuilles. Un lien sur un récapitulatif s'applique à chacune de ses feuilles.

L'exécuter

Membre	Description
Gantt.Schedule	Exécute le moteur sur les données du Gantt. Retourne False quand un cycle a été trouvé.
Gantt.AutoSchedule	Exécute le moteur après chaque modification faite dans le Gantt : un déplacement, un redimensionnement et un nouveau lien. Les changements faits dans le code ou dans une autre vue ne le déclenchent pas, appelez donc Schedule après eux.
Gantt.OnScheduleError, Engine.OnScheduleError	Déclenché après la mise à jour quand les liens forment un cycle, avec le message et les Id des tâches du cycle. Les autres tâches sont planifiées malgré tout.
Engine.Schedule(aData), Engine.CycleIds	Le même passage sans vue, et les Id du dernier cycle trouvé.

Toutes les dates et tous les résultats sont écrits en une seule mise à jour : pckItems pour les dates et pckSchedule pour la marge et les indicateurs critiques. Un passage qui ne change rien ne notifie rien. Sur une machine de développement, 10 000 tâches avec 15 000 liens et 30 jours fériés sont planifiées en 11 ms environ (Win64).

Exemple

```
uses
  SysUtils, Classes,
  sgcPlanner_Data, sgcPlanner_AutoSchedule;

type
  TScheduleLog = class
```

```
  public
    Errors: TStrings;
    procedure ScheduleError(Sender: TObject; const aError: string;
      aCycleIds: TStrings);
  end;

procedure TScheduleLog.ScheduleError(Sender: TObject; const aError: string;
  aCycleIds: TStrings);
begin
  // the tasks of the cycle keep their dates, the others are scheduled
  Errors.Add(aError);
end;

function ScheduleProject(aData: TsgcPlannerData; aLog: TScheduleLog): Boolean;
var
  oEngine: TsgcPlannerScheduleEngine;
  oItem: TsgcPlannerItem;
begin
  aData.ProjectStart := EncodeDate(2026, 10, 5) + EncodeTime(8, 0, 0, 0);
  // the venue is free from 19 October: Setup starts no earlier
  oItem := aData.FindItem('setup');
  if oItem <> nil then
  begin
    oItem.Constraint := pcSNET;
    oItem.ConstraintDate := EncodeDate(2026, 10, 19) + EncodeTime(8, 0, 0, 0);
  end;
  // the launch date is fixed: the engine never moves a Manual task
  oItem := aData.FindItem('launch');
  if oItem <> nil then
    oItem.Manual := True;
  // a scheduler without any view, for a service or a unit test
  oEngine := TsgcPlannerScheduleEngine.Create;
  try
    oEngine.OnScheduleError := aLog.ScheduleError;
    // False when the links form a cycle
    Result := oEngine.Schedule(aData);
  finally
    oEngine.Free;
  end;
end;
```

Voir aussi

- Dépendances
- Chemin critique
- Lignes de base
- Calendrier de travail

Chemin critique

Édition : Registered. Dans l'édition Community, ShowCriticalPath n'a aucun effet et le moteur qui calcule la marge n'est pas inclus. Voir Fonctionnalités par édition.
Le chemin critique est la chaîne de tâches qui fixe la fin du projet : le retard de l'une d'elles retarde la fin. Le moteur de planification le trouve à chaque passage.

Marge

- Après le passage aller, un passage retour depuis la fin du projet, la fin la plus tardive de toutes les tâches feuilles, donne **TotalSlack** : les minutes de travail dont une tâche peut glisser sans déplacer la fin du projet. C'est la plus petite de la marge de début et de la marge de fin.
- **Critical** vaut True quand TotalSlack est 0 ou moins. La marge est négative quand une tâche qui conserve ses dates (Manual ou récurrente) rompt une dépendance.
- Une tâche récapitulative prend la plus petite marge de ses feuilles et est critique quand l'une d'elles l'est. Les tâches d'un cycle n'ont pas de marge et ne sont jamais critiques.

TotalSlack et Critical sont des résultats en lecture seule. Ils ne sont pas enregistrés dans le JSON, donc exécutez Schedule après avoir chargé un projet.

L'afficher

Mettez ShowCriticalPath du Gantt à True : les barres et les liens des tâches critiques sont dessinés avec la couleur Critical du thème, que Colors.Critical remplace. Exécutez Schedule, ou définissez AutoSchedule, pour que les résultats soient à jour.

Exemple

```
uses
  SysUtils, Classes,
  sgcPlanner_Data, sgcPlanner_AutoSchedule, sgcPlanner_Gantt_VCL;

procedure ListCriticalPath(aData: TsgcPlannerData; aList: TStrings);
var
  i: Integer;
  oItem: TsgcPlannerItem;
  oEngine: TsgcPlannerScheduleEngine;
begin
  // TotalSlack and Critical are results of the engine
  oEngine := TsgcPlannerScheduleEngine.Create;
  try
    oEngine.Schedule(aData);
  finally
    oEngine.Free;
  end;
  aList.Clear;
  for i := 0 to aData.Items.Count - 1 do
  begin
    oItem := aData.Items[i];
    if oItem.IsSummary then
      Continue;
    if oItem.Critical then
      aList.Add(oItem.Title + ': critical')
    else
      // the slack is in working minutes
      aList.Add(oItem.Title + ': ' + IntToStr(oItem.TotalSlack div 60) +
        ' working hours of slack');
  end;
end;
```

```
procedure HighlightCriticalPath(aGantt: TsgcVCLGantt);
begin
  aGantt.ShowCriticalPath := True;
  // the critical bars and links, 0 keeps the colour of the theme
  aGantt.Colors.Critical := $FFC62828;
end;
```

Voir aussi

- Planification automatique
- Dépendances
- Lignes de base
- TsgcVCLGantt

Lignes de base

Édition : Registered. Dans l'édition Community, BaselineStart et BaselineFinish sont stockés et enregistrés, ShowBaseline n'a aucun effet et SetBaseline du Gantt lève EsgcPlannerError. Voir Fonctionnalités par édition.
Une ligne de base est une copie enregistrée du plan. Chaque élément a BaselineStart et BaselineFinish à côté de son Start et de son Finish, et le Gantt dessine la ligne de base sous forme d'une barre fine sous la barre courante, de sorte qu'une tâche en retard se voit d'un coup d'œil.

Définir la ligne de base

- SetBaseline du Gantt copie Start et Finish de chaque élément dans ses dates de ligne de base, en une seule mise à jour.
- Sans vue, SetBaseline et ClearBaseline de TsgcPlannerScheduleEngine font la même chose sur un TsgcPlannerData. ClearBaseline remet les dates de ligne de base à 0, ce qui signifie non définie.
- Les dates de ligne de base sont des propriétés ordinaires, donc votre code peut définir la ligne de base d'une seule tâche, ou conserver plusieurs lignes de base à lui et en copier une.

L'afficher

Mettez ShowBaseline du Gantt à True. Les lignes des éléments qui ont une ligne de base dessinent la barre courante dans la partie haute de la ligne et la ligne de base en dessous. Comparez les dates dans le code avec WorkingMinutesBetween du calendrier pour rapporter le retard en temps de travail.

Fichiers

JSON enregistre les dates de ligne de base. XML MS Project lit et écrit la ligne de base numéro 0, et iCalendar les conserve dans X-SGC-BASELINE-START et X-SGC-BASELINE-FINISH.

Exemple

```
uses
  SysUtils, Classes,
  sgcPlanner_Data, sgcPlanner_AutoSchedule, sgcPlanner_Gantt_VCL;

procedure ApprovePlan(aGantt: TsgcVCLGantt);
begin
  // BaselineStart and BaselineFinish of every item take its Start and Finish
  aGantt.SetBaseline;
  aGantt.ShowBaseline := True;
end;

function SlipInWorkingHours(aData: TsgcPlannerData;
  aItem: TsgcPlannerItem): Int64;
begin
  // later than the plan is positive, earlier is negative
  if aItem.BaselineFinishTime = 0 then
    Result := 0
  else
    Result := aData.Calendar.WorkingMinutesBetween(aItem.BaselineFinishTime,
      aItem.FinishTime) div 60;
end;

procedure RebaseWithoutGantt(aData: TsgcPlannerData);
var
  oEngine: TsgcPlannerScheduleEngine;
begin
  oEngine := TsgcPlannerScheduleEngine.Create;
```

```
  try
    // ClearBaseline sets both baseline dates of every item back to 0
    oEngine.ClearBaseline(aData);
    oEngine.Schedule(aData);
    oEngine.SetBaseline(aData);
  finally
    oEngine.Free;
  end;
end;
```

Voir aussi

- Planification automatique
- Chemin critique
- TsgcVCLGantt
- XML MS Project

Couloirs et limites WIP

Édition : Registered. Dans l'édition Community, SwimlaneBy se comporte comme pslNone et WIPLimit n'affiche aucun badge et ne bloque jamais une carte. Les colonnes, les largeurs et le repli fonctionnent dans toutes les éditions. Voir Fonctionnalités par édition.
Le tableau Kanban affiche les éléments sous forme de cartes dans des colonnes. Les couloirs divisent le tableau en bandes horizontales, et les limites WIP plafonnent le travail en cours d'une colonne.

Columns

Propriété	Description
ColumnSource	pkcStatuses (par défaut) : une colonne par statut des données, dans leur ordre. pkcList : seulement les colonnes de Columns.
Columns	Des entrées avec StatusId, Caption, Color, Width, WIPLimit et Collapsed. Avec pkcStatuses, une entrée dont le StatusId est celui d'un statut ajoute ses paramètres à cette colonne. AddColumn(StatusId, Caption, WIPLimit) en ajoute une.
ShowUnknownColumn	Affiche dans une dernière colonne les cartes dont le statut ne correspond à aucune colonne. On ne peut pas y déposer de carte.

Un élément récapitulatif et une occurrence modifiée ne sont pas affichés comme des cartes. Les cartes d'une colonne sont ordonnées par Rank, puis par leur ordre dans les données. CollapseColumn et ExpandColumn, ou le chevron de l'en-tête de colonne, replient une colonne en un rail étroit, et OnColumnCollapse le signale.

Couloirs

SwimlaneBy	Couloirs	Un dépôt sur un autre couloir
pslNone (par défaut)	Pas de couloirs.	Impossible.
pslResource	Un couloir par ressource. Une carte va dans le couloir de sa première ressource.	Remplace la première ressource de la carte.
pslParent	Un couloir par parent d'une carte, dans l'ordre des données.	Définit ParentId de la carte.
pslTag	Un couloir par première étiquette d'une carte, triés.	Impossible.
pslEvent	Les couloirs que OnGetSwimlane retourne sous forme de clé et de libellé, dans l'ordre d'apparition. Sans gestionnaire, il se comporte comme pslNone.	Impossible.

Les cartes sans couloir vont dans un couloir à la fin.

Limites WIP

- WIPLimit d'une entrée de colonne est le plus grand nombre de cartes que la colonne devrait contenir, comptées sur tous les couloirs. 0 signifie aucune limite.

- Le badge de l'en-tête affiche le compte, le compte et la limite sous la limite, un badge ambre à la limite et un badge rouge au-delà.
- WIPMode pwmWarn (par défaut) ne fait que marquer la colonne. pwmBlock refuse une carte qui entrerait dans une colonne à sa limite, tandis que les cartes déjà dans la colonne peuvent encore être réordonnées.

Exemple

```
uses
  SysUtils, Classes,
  sgcPlanner_Data, sgcPlanner_Kanban, sgcPlanner_Kanban_VCL;

type
  TBoardEvents = class
  public
    procedure GetSwimlane(Sender: TObject; aItem: TsgcPlannerItem;
      var aKey, aCaption: string);
  end;

function HasTag(aItem: TsgcPlannerItem; const aTag: string): Boolean;
var
  i: Integer;
begin
  Result := False;
  for i := 0 to aItem.TagCount - 1 do
    if SameText(aItem.TagAt(i), aTag) then
      Result := True;
end;

procedure TBoardEvents.GetSwimlane(Sender: TObject; aItem: TsgcPlannerItem;
  var aKey, aCaption: string);
begin
  // an expedite lane above the standard work
  if HasTag(aItem, 'urgent') then
  begin
    aKey := 'expedite';
    aCaption := 'Expedite';
  end
  else
  begin
    aKey := 'standard';
    aCaption := 'Standard';
  end;
end;

procedure SetupBoard(aKanban: TsgcVCLKanban; aEvents: TBoardEvents);
begin
  aKanban.ColumnSource := pkcStatuses;
  // In progress holds 2 cards across all lanes, Review 1
  aKanban.Columns.AddColumn('doing', '', 2);
  aKanban.Columns.AddColumn('review', 'Code review', 1);
  // a card dropped on a full column goes back where it was
  aKanban.WIPMode := pwmBlock;
  aKanban.SwimlaneBy := pslEvent;
  aKanban.OnGetSwimlane := aEvents.GetSwimlane;
end;
```

Voir aussi

- TsgcVCLKanban
- Glisser-déposer
- Thèmes
- TsgcFMXKanban

Liaison DataSet

TsgcPlannerSource_DataSet lie un TsgcPlannerData à des TDataSet : n'importe quel jeu de données de l'unité DB, comme FireDAC, TClientDataSet, dbExpress ou un jeu de données tiers. Il charge les éléments, les ressources et les dépendances, suit les changements des jeux de données et réécrit les modifications des vues. Il n'utilise que l'unité DB, donc il fonctionne en VCL, en FireMonkey et dans les applications console, dans toutes les éditions.

Jeux de données et correspondances de champs

Jeu de données	Correspondance de champs	Noms de champs par défaut
DataSet	ItemFields	Id, Title, Start, Finish, AllDay, Status, Progress, Color, ParentId, Tags, Resources, Rank, Notes, RecurrenceRule et DependsOn.
ResourceDataSet	ResourceFields	Id, Name, Color et Initials.
DependencyDataSet	DependencyFields	FromId, ToId, Kind et Lag.

Un nom de champ vide ou introuvable dans le jeu de données est ignoré, donc une table ne mappe que les champs qu'elle a. Les dates sont lues dans des champs date et heure, des champs numériques ou du texte ISO 8601 ; les booléens dans des champs booléens, des nombres ou du texte tel que Y ou True ; les couleurs dans des nombres ARGB ou du texte tel que #RRGGBB. Un Finish nul donne Finish = Start, ou le minuit suivant pour un élément sur toute la journée. Le Kind d'une dépendance est FS, SS, FF ou SF, et son Lag est en minutes de travail. Sans DependencyDataSet, le champ DependsOn peut contenir les Id, séparés par des virgules, des prédécesseurs d'une tâche, lus comme des liens fin à début. OnLoadItem est déclenché pour chaque enregistrement après la copie des champs mappés, pour remplir d'autres propriétés de l'élément.

Chargement et mises à jour en direct

Active à True charge chaque jeu de données ouvert en une seule mise à jour, de sorte que les vues liées se redessinent une fois ; 10 000 lignes se chargent en 26 ms environ sous Win64. Avec LiveUpdates, une modification ou une insertion validée met à jour ou ajoute cet élément, et un changement du nombre d'enregistrements (une suppression, un filtre ou un rafraîchissement) synchronise les éléments par Id. Le défilement et les modifications non validées ne changent rien. Load et Refresh rechargent à la main.

Réécriture

Avec WriteBack, à True par défaut, une modification des données, faite dans une vue ou dans le code, est validée dans l'enregistrement trouvé avec Locate sur le champ Id : seuls les champs mappés modifiés sont affectés. Un nouvel élément ajoute un enregistrement, un élément supprimé supprime son enregistrement, et les liens nouveaux ou supprimés sont écrits dans le DependencyDataSet ou dans la liste DependsOn. Tant que le jeu de données est en état d'édition ou d'insertion, rien n'est écrit ; les changements en attente sont écrits quand il revient à l'état de navigation. Les ressources, les champs calculés, les champs de recherche et les champs en lecture seule ne sont jamais écrits. Une écriture ne revient jamais sous forme de nouveau chargement.

Exemple

```
uses
  SysUtils, Classes, DB, DBClient,
  sgcPlanner_Data, sgcPlanner_Source_DataSet;
```

```
function CreateTaskTable(aOwner: TComponent): TClientDataSet;
begin
  // the default field names: Id, Title, Start, Finish, Status, DependsOn
  Result := TClientDataSet.Create(aOwner);
  Result.FieldDefs.Add('Id', ftString, 20);
  Result.FieldDefs.Add('Title', ftString, 100);
  Result.FieldDefs.Add('Start', ftDateTime);
  Result.FieldDefs.Add('Finish', ftDateTime);
  Result.FieldDefs.Add('Status', ftString, 20);
  Result.FieldDefs.Add('DependsOn', ftString, 200);
  Result.CreateDataSet;
  Result.AppendRecord(['design', 'Design', EncodeDate(2026, 10, 5),
    EncodeDate(2026, 10, 8), 'done', '']);
  Result.AppendRecord(['build', 'Build', EncodeDate(2026, 10, 8),
    EncodeDate(2026, 10, 15), 'doing', 'design']);
end;

procedure BindTable(aSource: TsgcPlannerSource_DataSet;
  aData: TsgcPlannerData; aTasks: TDataSet);
begin
  aSource.Data := aData;
  aSource.DataSet := aTasks;
  // no link table: DependsOn holds the ids of the predecessors (FS links)
  aSource.LiveUpdates := True;
  aSource.WriteBack := True;
  aSource.Active := True;
  // a change of the data is posted to the record at once
  aData.FindItem('build').Status := 'done';
end;
```

La démo DataSetPlanner lie trois tables TClientDataSet avec leurs propres noms de champs. Voir Démos.

Voir aussi

- TsgcPlannerSource_DataSet
- Données partagées
- Persistance JSON
- Dépendances

Persistance JSON

Un TsgcPlannerData s'enregistre en JSON et se charge depuis du JSON dans toutes les éditions et sur toutes les plateformes : SaveToJSON et LoadFromJSON pour une chaîne, SaveToFile et LoadFromFile pour un fichier, et les fonctions de sgcPlanner_JSON pour les flux (sgcPlannerJSONSaveToStream et sgcPlannerJSONLoadFromStream) et les fichiers.

Le format

Le document porte `"format": "sgcPlanner"` et une version, puis le début du projet, le calendrier, les ressources, les statuts, les éléments et les dépendances. Les dates sont des heures locales ISO 8601 et les couleurs sont des nombres ARGB. Un projet abrégé :

```
{
  "format": "sgcPlanner",
  "version": 1,
  "projectStart": "2026-10-05T08:00:00",
  "calendar": {"workDays":["mon","tue","wed","thu","fri"],"workTime":"08:00-12:00,13:00-17:00","firstDayOfWeek":"mon","holi
  "resources": [
    {"id":"ann","name":"Ann Lee","color":4280973526}
  ],
  "statuses": [
    {"id":"todo","caption":"To do"},
    {"id":"done","caption":"Done","color":4282557941}
  ],
  "items": [
    {"id":"t1","title":"Design","start":"2026-10-05T08:00:00","finish":"2026-10-07T17:00:00","status":"done","progress":100
    {"id":"m1","title":"Weekly review","start":"2026-10-05T09:00:00","finish":"2026-10-05T09:30:00","color":4285887861,"not
  ],
  "dependencies": [
    {"from":"t1","to":"t2","kind":"FS"}
  ]
}
```

Règles

- Les fichiers et les flux sont en UTF-8 sans marque d'ordre des octets. Une marque d'ordre des octets est acceptée au chargement.
- Un membre d'élément qui a sa valeur par défaut (un texte vide, false, 0 ou une date non définie) n'est pas écrit, sauf id, title, start et finish, ce qui garde les fichiers petits.
- Les valeurs calculées (TotalSlack, Critical, IsSummary), la sélection et Version ne sont pas enregistrées.
- Le chargement remplace l'ensemble des données, calendrier et ProjectStart compris, en une seule mise à jour, de sorte que chaque vue liée se redessine une fois. Un membre absent charge sa valeur par défaut.
- Les membres inconnus, les membres d'un autre type JSON et les valeurs qui ne s'analysent pas sont ignorés, donc un fichier écrit par une version ultérieure se charge quand même. Une version absente est lue comme 1.
- Quand le texte n'est pas du JSON valide, que la racine n'est pas un objet, que le format est un autre ou que la version n'est pas prise en charge, EsgcPlannerError est levée et les données restent telles qu'elles étaient.
- Avant Delphi 2009, le texte passe par la page de codes ANSI du système.

Sur une machine de développement, 10 000 éléments avec 15 000 liens s'enregistrent en 48 ms environ et se chargent en 56 ms environ (Win64).

Exemple

```
uses
  SysUtils, Classes,
  sgcPlanner_DateUtils, sgcPlanner_Data, sgcPlanner_JSON;
```

```
function TryOpenProject(aData: TsgcPlannerData; const aFileName: string;
  out aError: string): Boolean;
begin
  Result := False;
  if not FileExists(aFileName) then
  begin
    aError := 'File not found: ' + aFileName;
    Exit;
  end;
  try
    aData.LoadFromFile(aFileName);
    Result := True;
  except
    on E: EsgcPlannerError do
      // not JSON, not an sgcPlanner document or a newer version:
      // the data is left as it was
      aError := E.Message;
  end;
end;

procedure CopyProject(aSource, aTarget: TsgcPlannerData);
var
  oStream: TMemoryStream;
begin
  oStream := TMemoryStream.Create;
  try
    sgcPlannerJSONSaveToStream(aSource, oStream);
    oStream.Position := 0;
    // replaces everything in aTarget, the calendar included
    sgcPlannerJSONLoadFromStream(aTarget, oStream);
  finally
    oStream.Free;
  end;
end;
```

Voir aussi

- TsgcPlannerData
- Liaison DataSet
- iCalendar
- XML MS Project

iCalendar

Édition : Registered. Dans l'édition Community, TsgcPlannerICalendar n'existe pas ; JSON est le format de fichier de cette édition. Voir Fonctionnalités par édition.
TsgcPlannerICalendar importe et exporte des calendriers RFC 5545 (.ics), le format de Google Calendar, d'Outlook, d'Apple Calendar et de la plupart des serveurs de calendrier. Un VEVENT correspond à un élément.

Import

- ImportFromString, ImportFromStream et ImportFromFile lisent de l'UTF-8, avec ou sans marque d'ordre des octets. Un UTF-8 invalide est lu comme du Latin-1, avec un avertissement.
- ImportMode pimReplace (par défaut) supprime d'abord les éléments et les dépendances et conserve les ressources, les statuts, le calendrier et ProjectStart. pimMerge met à jour l'élément qui a le même Id et ajoute les autres.
- Les heures en UTC et les heures avec un TZID sont converties à l'heure locale de la machine, grâce au VTIMEZONE du fichier. UTC, GMT et Etc/UTC n'ont pas besoin de VTIMEZONE. Un TZID inconnu donne un avertissement et est lu comme heure locale.
- L'Id de l'élément est l'UID. Un VEVENT avec un RECURRENCE-ID devient une occurrence modifiée de la série qui a cet UID.
- Les problèmes qui n'arrêtent pas l'import, comme une partie de RRULE non prise en charge ou un RDATE, sont signalés par OnImportWarning avec le numéro de ligne, et comptés dans WarningCount. Un texte sans BEGIN:VCALENDAR lève EsgcPlannerError et laisse les données inchangées.

Export

- ExportToString, ExportToStream et ExportToFile écrivent chaque élément sous forme de VEVENT, avec ProductId comme PRODID.
- Avec ExportUTC (par défaut True), les heures des éléments qui ne se répètent pas sont écrites en UTC. Les séries et les occurrences modifiées utilisent toujours des heures locales flottantes, de sorte que leurs occurrences restent à la même heure lors des changements d'heure d'été.
- Un élément sur toute la journée dont le début et la fin sont à minuit est écrit sous forme de valeurs DATE.
- Les dépendances ne sont pas exportées.

Correspondance

iCalendar	Élément
UID, SUMMARY, DESCRIPTION	Id, Title, Notes.
DTSTART, DTEND	Start, Finish.
RRULE, EXDATE, RECURRENCE-ID	RecurrenceRule, RecurrenceExceptions, RecurrenceParentId et RecurrenceDate.
STATUS	Status en minuscules : tentative, confirmed ou cancelled. Un autre statut est conservé dans X-SGC-STATUS.
RESOURCES	Les noms des ressources. X-SGC-RESOURCES conserve leurs Id quand ils diffèrent, et l'import crée les ressources manquantes.
CATEGORIES	Tags.
RELATED-TO;RELTYPE=PARENT	ParentId.

X-SGC-PROGRESS, X-SGC-COLOR, X-SGC-RANK, X-SGC-MILESTONE, X-SGC-MANUAL, X-SGC-CONSTRAINT, X-SGC-CONSTRAINT-DATE, X-SGC-BASELINE-START, X-SGC-BASELINE-FINISH, X-SGC-ALLDAY	Les propriétés du planificateur pour lesquelles iCalendar n'a pas de champ, de sorte qu'un fichier du planificateur fait l'aller-retour.

Sur une machine de développement, 5 000 VEVENT avec fuseaux horaires s'importent en 84 ms environ (Win64). La démo ICalendarSync importe des calendriers Google et Outlook et exporte une série. Voir Démos.

Voir aussi

- TsgcPlannerICalendar
- Récurrence
- XML MS Project
- Persistance JSON

XML MS Project

Édition : Registered. Dans l'édition Community, TsgcPlannerMSProject n'existe pas ; JSON est le format de fichier de cette édition. Voir Fonctionnalités par édition.
TsgcPlannerMSProject importe et exporte le XML Microsoft Project (MSPDI), le format XML que Microsoft Project ouvre et enregistre. Il lit les tâches, les liens, le calendrier, les ressources, les affectations et les lignes de base.

Import

- ImportFromString, ImportFromStream et ImportFromFile lisent de l'UTF-8, avec ou sans marque d'ordre des octets. Un document mal formé lève EsgcPlannerError avec le numéro de ligne.
- ImportMode pmiReplace (par défaut) supprime les éléments et les liens et reprend du fichier le calendrier et le début du projet (StartDate). Les ressources et les statuts sont conservés, et les ressources du fichier sont ajoutées ou renommées.
- pmiMerge conserve le calendrier et ProjectStart, met à jour les éléments par Id et ajoute ou met à jour les liens selon leurs deux extrémités. Les ressources des éléments ne suivent le fichier que s'il a des Assignments.
- OnImportWarning signale ce qui est lu d'une autre manière, par exemple un type de contrainte que sgcPlanner n'a pas.

Correspondance

MSPDI	sgcPlanner
Task UID, Name, Start, Finish, PercentComplete, Milestone, Notes	Id, Title, Start, Finish, Progress, Milestone, Notes.
OutlineLevel et l'ordre des tâches	ParentId, l'arborescence des tâches.
PredecessorLink Type 0, 1, 2, 3	pdkFF, pdkFS, pdkSF, pdkSS. LinkLag est en dixièmes de minute de travail ; les formats de décalage écoulé et en pourcentage sont convertis.
ConstraintType 0 et 4	pcASAP et pcSNET avec ConstraintDate. Le type 2 (doit commencer le) est lu comme pcSNET et les autres types comme pcASAP, tous deux avec un avertissement.
Baseline Number 0	BaselineStart et BaselineFinish.
Calendar WeekDays, WorkingTimes, Exceptions	Le calendrier de travail : jours ouvrés, heures de travail et jours fériés.
Resource Code ou Name, Assignments	Les ressources et la liste Resources de chaque élément.

Export

ExportToString, ExportToStream et ExportToFile écrivent dans l'ordre du schéma de SaveVersion 14 : un calendrier Standard avec une exception par jour férié, les tâches dans l'ordre du plan, leurs liens et leurs affectations. Un Id numérique conserve son UID et les autres Id reçoivent le prochain numéro libre. MSPDI conserve un lien par paire de tâches. Le statut, les étiquettes, la couleur, le rang, la journée entière et la récurrence ne sont pas écrits, donc gardez un fichier JSON à côté quand vous en avez besoin.
Sur une machine de développement, 10 000 tâches avec 15 000 liens s'importent en 264 ms environ (Win64). La démo ProjectGantt ouvre et enregistre des fichiers MSPDI. Voir Démos.

Voir aussi

- TsgcPlannerMSProject
- Dépendances
- Lignes de base
- Persistance JSON

Rendu sans fenêtre

Chaque vue de sgcPlanner peut être dessinée sans fenêtre. Les modèles de vue dessinent sur un TsgcUICanvas de sgcUICore, et les mêmes procédures de dessin servent les contrôles d'une fiche, TsgcPlannerRenderer et les méthodes SaveToPNG et SaveToSVG des contrôles, de sorte qu'une image a les mêmes pixels que l'écran.

TsgcPlannerRenderer

Le renderer est un composant non visuel sans unité VCL ni FireMonkey, donc il fonctionne dans un service, un serveur web, une application console et sous Linux64. Il fonctionne dans toutes les éditions.

Membre	Description
Data, View	Les données, et la vue à dessiner : prvScheduler, prvGantt ou prvKanban.
Width, Height, Scale	La taille en DIP (par défaut 800 x 600) et l'échelle de la sortie, 2 pour une image HiDPI.
Theme, Colors	Les mêmes remplacements de thème et de couleurs que sur les contrôles. Voir Thèmes.
SchedulerOptions, GanttOptions, KanbanOptions	Les options publiées de chaque vue. Assign accepte aussi un modèle de vue, de sorte que `Renderer.GanttOptions.Assign(VCLGantt1.GanttView)` copie ce qu'affiche une fiche.
Date	La date que le Scheduler affiche, 0 pour aujourd'hui.
SchedulerView, GanttView, KanbanView	Les modèles de vue que possède le renderer, pour ExpandAll, ZoomToFit, ScrollToTime ou CollapseColumn avant un rendu.
RenderTo(aCanvas), SaveToPNG, SaveToSVG	Dessinent sur un canevas, ou enregistrent un PNG ou un SVG dans un fichier ou un flux.
FontFile, FontFileBold	Fichiers TrueType pour le texte. Vide utilise la police intégrée à sgcUICore, qui n'exige aucune police installée sur la machine.

Images des contrôles

Les six contrôles ont aussi SaveToPNG et SaveToSVG, avec un nom de fichier ou un flux et une largeur, une hauteur et une échelle facultatives. La taille est en DIP et 0 est la taille client ; une échelle de 0 est l'échelle du contrôle. La sortie conserve la position de défilement du contrôle et est la même image qu'un renderer avec les mêmes données, options, taille et échelle. Les barres de défilement, l'aperçu du glissement et les repères de dépôt sont omis.

Sur un serveur

- La sélection de l'image est le SelectedId des données ; videz-le pour un rapport sans élément sélectionné.
- Les noms des jours et des mois et les formats de date proviennent de sgcPlannerLocale, initialisé à partir de la machine. Définissez-le au démarrage quand un serveur doit produire les mêmes libellés qu'un poste de travail.
- La sortie SVG conserve le texte sous forme de texte, donc elle reste nette à tout niveau de zoom et on peut y effectuer des recherches.

Exemple

```
uses
  SysUtils, Classes,
  sgcPlanner_Data, sgcPlanner_Gantt, sgcPlanner_Kanban, sgcPlanner_Renderer,
  sgcPlanner_Gantt_VCL;

procedure SaveReport(aData: TsgcPlannerData; const aFolder: string);
var
  oRenderer: TsgcPlannerRenderer;
begin
  // no form and no window: a service, a web server or Linux64
  oRenderer := TsgcPlannerRenderer.Create(nil);
  try
    oRenderer.Data := aData;
    oRenderer.Width := 1200;
    oRenderer.Height := 700;
    // the board with one lane per person
    oRenderer.View := prvKanban;
    oRenderer.KanbanOptions.SwimlaneBy := pslResource;
    oRenderer.SaveToPNG(IncludeTrailingPathDelimiter(aFolder) + 'board.png');
    // the whole plan, every summary open, zoomed to fit the width
    oRenderer.View := prvGantt;
    oRenderer.GanttView.ExpandAll;
    oRenderer.GanttView.ZoomToFit;
    oRenderer.SaveToSVG(IncludeTrailingPathDelimiter(aFolder) + 'plan.svg');
  finally
    oRenderer.Free;
  end;
end;

procedure SaveWhatTheFormShows(aGantt: TsgcVCLGantt);
begin
  // the control itself: same options and scroll position, twice the size
  aGantt.SaveToPNG('gantt.png', 0, 0, 2);
end;
```

La démo HeadlessReport est une application console pour Windows et Linux64 qui écrit chaque vue en PNG et en SVG. Voir Démos.

Voir aussi

- TsgcPlannerRenderer
- Thèmes
- Plateformes
- Vues synchronisées

Thèmes

Les contrôles du planificateur et le renderer sont dessinés avec les thèmes de sgcUICore, de sorte qu'un planificateur s'accorde avec les autres produits sgcUI sur la même fiche, sous VCL, sous FireMonkey et dans la sortie sans fenêtre.

Thème et couleurs

- **Theme** vaut utLight (par défaut), utDark ou utCustom. utCustom part des couleurs claires.
- **Colors** remplace des couleurs isolées du thème : Background, Surface, Border, Grid, HeaderBackground, HeaderText, Text, TextMuted, Accent, Selection, NonWorking, NowLine, Critical et Ghost. Une couleur est un TsgcUIColor ARGB, par exemple sgcUIARGB($FF, $00, $96, $88). 0 conserve la couleur du thème.

Couleurs des éléments

Un élément est dessiné avec la première couleur définie : le Color de l'élément, le Color de son statut, le Color de sa première ressource, et sinon une couleur de la palette du thème choisie à partir de son Id, de sorte que le même élément garde la même couleur dans toutes les vues et à chaque exécution.

Paramètre	Effet
Gantt BarColorMode	pbcItem (la couleur de l'élément ci-dessus), pbcStatus (la couleur du statut) ou pbcResource (la couleur de la première ressource).
Kanban Columns Color	La barre de couleur de l'en-tête d'une colonne. Avec pkcStatuses, la couleur du statut est utilisée quand elle n'est pas définie.
Étiquettes	Les pastilles d'étiquette des cartes prennent une couleur de la palette du thème choisie à partir du texte de l'étiquette.
Resources	Les cercles d'initiales des cartes et des couloirs utilisent la couleur de la ressource.

Polices et tailles

Le texte utilise la police intégrée à sgcUICore, de sorte que la sortie est la même sur chaque machine. Le renderer accepte des fichiers TrueType avec FontFile et FontFileBold. Les tailles sont en DIP et sont mises à l'échelle pour le moniteur, la fiche ou le Scale du renderer.

Exemple

```
uses
  SysUtils, Classes,
  sgcUICore_Theme, sgcPlanner_Data, sgcPlanner_Gantt, sgcPlanner_Gantt_VCL,
  sgcPlanner_Kanban_VCL;

procedure ApplyDarkTheme(aGantt: TsgcVCLGantt; aKanban: TsgcVCLKanban);
begin
  aGantt.Theme := utDark;
  aKanban.Theme := utDark;
  // the corporate accent; every other colour comes from the theme
  aGantt.Colors.Accent := sgcUIARGB($FF, $00, $96, $88);
  aKanban.Colors.Accent := sgcUIARGB($FF, $00, $96, $88);
  // 0 hands a colour back to the theme
  aGantt.Colors.NowLine := 0;
end;
```

```
procedure ColourByStatus(aData: TsgcPlannerData; aGantt: TsgcVCLGantt);
var
  oStatus: TsgcPlannerStatus;
begin
  // the bars take the colour of their status
  oStatus := aData.Statuses.Find('done');
  if oStatus <> nil then
    oStatus.Color := sgcUIARGB($FF, $43, $A0, $47);
  aGantt.BarColorMode := pbcStatus;
end;
```

Voir aussi

- Rendu sans fenêtre
- Couloirs et limites WIP
- TsgcVCLGantt
- TsgcVCLKanban

Fonctionnalités par édition

sgcPlanner est livré en éditions Registered, Trial et Community. Trial a toutes les fonctionnalités de Registered. Ce tableau liste chaque fonctionnalité et l'édition dont elle a besoin. Voir Éditions pour les éditions elles-mêmes.

Fonctionnalité	Registered et Trial	Community
Données partagées, Vues synchronisées	Oui	Oui
Calendrier de travail	Oui	Oui
Récurrence	Oui	Non. Un élément récurrent n'est affiché qu'une fois.
Échelle de temps et zoom	Oui	Oui
Glisser-déposer	Oui	Oui, sauf le glissement pour lier dans le Gantt.
Dépendances	Oui	Les liens sont stockés et enregistrés, mais pas dessinés.
Planification automatique	Oui	Non. Schedule lève EsgcPlannerError.
Chemin critique	Oui	No.
Lignes de base	Oui	Les dates sont stockées et enregistrées, mais pas dessinées. SetBaseline lève EsgcPlannerError.
Vues Scheduler jour, semaine de travail, semaine, mois et agenda	Oui	Oui
Timeline de ressources du Scheduler	Oui	Non. La vue affiche un avis.
Colonnes Kanban, repli et options des cartes	Oui	Oui
Couloirs et limites WIP	Oui	Non. SwimlaneBy se comporte comme pslNone.
Liaison DataSet	Oui	Oui
Persistance JSON	Oui	Oui
iCalendar, XML MS Project	Oui	Non. Les composants n'existent pas.
Rendu sans fenêtre, Thèmes	Oui	Oui

Du code qui compile dans toutes les éditions

Les fonctionnalités avancées sont compilées sous SGC_PLANNER_ADVANCED, que sgcVer.inc définit dans les éditions Registered et Trial et désactive dans Community. Incluez sgcVer.inc dans une unité qui doit compiler avec les deux, et placez le code des fonctionnalités avancées dans `{$IFDEF SGC_PLANNER_ADVANCED}`. Les propriétés des vues sont publiées dans toutes les éditions, donc une fiche qui définit ShowCriticalPath ou SwimlaneBy se charge dans Community ; une fiche qui contient TsgcPlannerICalendar ou TsgcPlannerMSProject ne se charge pas.

Exemple

```
{$I sgcVer.inc}

uses
  SysUtils, Classes,
  sgcPlanner_Data, sgcPlanner_Gantt_VCL
{$IFDEF SGC_PLANNER_ADVANCED}
  , sgcPlanner_ICal
{$ENDIF}
  ;

procedure ExportPlan(aData: TsgcPlannerData; const aFileName: string);
{$IFDEF SGC_PLANNER_ADVANCED}
var
  oICal: TsgcPlannerICalendar;
{$ENDIF}
begin
{$IFDEF SGC_PLANNER_ADVANCED}
  oICal := TsgcPlannerICalendar.Create(nil);
  try
    oICal.Data := aData;
    oICal.ExportToFile(ChangeFileExt(aFileName, '.ics'));
  finally
    oICal.Free;
  end;
{$ELSE}
  // Community: JSON is in every edition
  aData.SaveToFile(ChangeFileExt(aFileName, '.json'));
{$ENDIF}
end;

procedure PreparePlan(aGantt: TsgcVCLGantt);
begin
{$IFDEF SGC_PLANNER_ADVANCED}
  // Schedule raises EsgcPlannerError in Community
  aGantt.Schedule;
  aGantt.ShowCriticalPath := True;
{$ENDIF}
  aGantt.ExpandAll;
end;
```

Voir aussi

- Éditions
- Installation
- Dépannage

Synchronisation avec Google Agenda

TsgcPlannerGoogleSync maintient un TsgcPlannerData et un Google Agenda en synchronisation bidirectionnelle. Cette rubrique décrit ce dont le composant a besoin, le déroulement d'une synchronisation, la correspondance des champs, la règle de conflit et le fichier d'état.

Prérequis

- L'édition Registered de sgcPlanner, avec le paquet de passerelle sgcPlannerWS (les unités sgcPlannerWS_Classes, sgcPlannerWS_Google et sgcPlannerWS_ICal).
- sgcWebSockets, édition Standard ou supérieure, qui contient le client Google Agenda TsgcHTTPGoogleCloud_Calendar_Client. Placez les sources de sgcPlanner, sgcUICore et sgcWebSockets, ou le dossier de bibliothèque de sgcWebSockets, dans le chemin de recherche.
- Les bibliothèques OpenSSL que sgcWebSockets utilise pour HTTPS, à côté de l'exécutable.

Le projet Google Cloud et le client OAuth2

1. Ouvrez la console Google Cloud, créez un projet (ou choisissez-en un) et activez l'API Google Calendar.
2. Configurez l'écran de consentement OAuth. Tant que l'application est en phase de test, ajoutez comme utilisateurs de test les comptes Google qui se connecteront.
3. Créez un ID client OAuth de type Application de bureau et copiez son ID client et son secret client.
4. Configurez le client : GoogleCloudOptions.Authentication sur gcaOAuth2, l'ID client et le secret dans GoogleCloudOptions.OAuth2, LocalIP et LocalPort pour la redirection du navigateur (par exemple 127.0.0.1 et 8080), et Scopes.EventsReadWrite et Scopes.CalendarsReadWrite. La synchronisation lit l'agenda une fois pour obtenir son fuseau horaire.
5. À la première requête, le navigateur ouvre la page de consentement de Google. Dès que l'utilisateur autorise l'accès, la synchronisation se poursuit.

Ne stockez pas le secret client dans un fichier de formulaire ni dans un code source que vous distribuez. La démo le saisit à l'exécution et le garde uniquement en mémoire.

Déroulement d'une synchronisation

1. Thread de travail : la liste des événements de CalendarId, avec le jeton de synchronisation de la dernière synchronisation (seulement les changements depuis, événements supprimés inclus), ou tous les événements à la première synchronisation, en suivant nextPageToken. Les séries arrivent chacune comme un seul événement maître.
2. Thread principal : les événements sont appliqués à Data dans un seul BeginUpdate et EndUpdate. Un événement annulé supprime son élément, une occurrence annulée ajoute une exception à sa série. Les conflits sont résolus et la liste d'envoi est construite à partir des éléments nouveaux, modifiés et supprimés, après OnFilterItem.
3. Thread de travail : insertions, mises à jour et suppressions dans Google.
4. Thread principal : les nouveaux ID d'événement, etags, heures de mise à jour, empreintes et le nouveau jeton de synchronisation sont enregistrés, l'état est sauvegardé et OnSyncDone se déclenche.

Un élément lu depuis Google reçoit l'id `g:` suivi de l'id de l'événement. Un élément créé dans le planificateur garde son id, qui voyage dans les propriétés étendues privées de son événement (sgcPlannerId), de sorte qu'il revient avec le même id.

Correspondance des champs

Élément du planificateur	Événement Google	Remarques
Title	summary	

Remarques	description	
Start, Finish	start.dateTime, end.dateTime	Lus avec leur décalage et convertis à l'heure locale. Écrits en UTC, sauf les événements récurrents et leurs instances, qui sont écrits en heure locale avec leur décalage plus timeZone (TimeZone, ou le fuseau horaire de l'agenda). Les heures ont une précision à la minute, les secondes sont supprimées.
AllDay, Start, Finish	start.date, end.date	La fin est exclusive des deux côtés : un élément sur la journée du 1er octobre va du 1er octobre 00:00 au 2 octobre 00:00, et son événement se termine le 2 octobre. À l'écriture, un Finish qui n'est pas à minuit est arrondi au jour suivant.
RecurrenceRule	the RRULE line of recurrence	Transmise sans changement, dans les deux sens.
RecurrenceExceptions	the EXDATE lines of recurrence	Écrites en heures UTC, ou en dates pour les séries sur la journée entière, de sorte qu'aucune base de fuseaux horaires n'est nécessaire.
RecurrenceParentId, RecurrenceDate	recurringEventId, originalStartTime	Une occurrence modifiée d'une série.
Id	extendedProperties.private.sgcPlannerId	L'id de l'élément dans le planificateur.

Les autres lignes de récurrence d'un événement, comme RDATE ou EXRULE, sont conservées dans l'état et réécrites sans changement. Une règle que le moteur de récurrence de sgcPlanner ne développe pas (par exemple une règle avec BYSETPOS ou HOURLY) est elle aussi conservée et réécrite sans changement ; l'élément montre sa première occurrence et OnError signale la raison une fois par règle, de sorte que Google ne perd jamais une règle.

Ce qui n'est pas synchronisé

Status, Progress, Color, Tags, Resources et les dépendances des éléments ne sont pas envoyés à Google, et les participants, rappels, couleurs et données de conférence des événements ne sont pas lus. Un élément modifié dans le planificateur est écrit avec la méthode patch de l'API Calendar, qui n'envoie que les champs dont le planificateur est propriétaire, de sorte que les participants, rappels, données de conférence et autres propriétés de l'événement sont conservés.

Conflits

Un élément modifié dans le planificateur et dans Google depuis la dernière synchronisation est un conflit. Le composant enregistre le moment où un élément est devenu différent de sa dernière empreinte synchronisée ; cet instant est comparé à l'heure `updated` de l'événement, en UTC :

- La modification la plus récente l'emporte. Une égalité, ou un événement sans heure de mise à jour lisible, revient à Google.
- OnConflict reçoit l'élément et le JSON de l'événement et peut définir aResolution sur gcrKeepLocal, gcrKeepRemote ou gcrSkip (les deux côtés restent tels quels dans cette exécution).
- Une modification face à une suppression conserve la modification des deux côtés : un événement supprimé est recréé, ou un élément supprimé est restauré. Cela compte comme un conflit mais ne déclenche pas OnConflict.
- Un événement dont le contenu correspond déjà à l'élément est seulement enregistré, pas compté comme un conflit. Les événements qui reviennent de l'envoi du composant lui-même sont reconnus par leur etag.
- Avec Direction gsdPullOnly, les modifications locales ne comptent jamais, donc Google l'emporte toujours.

Ce gestionnaire journalise les deux titres, conserve la version locale des éléments marqués `office` et garde hors de Google les éléments marqués `private` :

```
uses
  SysUtils, Classes,
  sgcPlanner_Data, sgcPlannerWS_Google;

type
  TSyncRules = class
  public
    Log: TStrings;
    procedure Conflict(Sender: TObject; aItem: TsgcPlannerItem;
      const aEventJSON: string; var aResolution: TsgcPlannerGoogleResolution);
    procedure FilterItem(Sender: TObject; aItem: TsgcPlannerItem;
      var aSync: Boolean);
    procedure Attach(aSync: TsgcPlannerGoogleSync);
  end;

procedure TSyncRules.Attach(aSync: TsgcPlannerGoogleSync);
begin
  aSync.OnConflict := Conflict;
  aSync.OnFilterItem := FilterItem;
end;

procedure TSyncRules.Conflict(Sender: TObject; aItem: TsgcPlannerItem;
  const aEventJSON: string; var aResolution: TsgcPlannerGoogleResolution);
var
  oEvent: TsgcPlannerGoogleEvent;
begin
  oEvent := TsgcPlannerGoogleEvent.Create;
  try
    if oEvent.LoadFromJSON(aEventJSON) then
      Log.Add(Format('conflict: "%s" here, "%s" in Google',
        [aItem.Title, oEvent.Summary]));
  finally
    oEvent.Free;
  end;
  // items tagged "office" are edited here only, keep the local version
  if Pos('office', aItem.Tags) > 0 then
    aResolution := gcrKeepLocal;
  // any other conflict keeps gcrLastModified: the later change wins
end;

procedure TSyncRules.FilterItem(Sender: TObject; aItem: TsgcPlannerItem;
  var aSync: Boolean);
begin
  // private items never leave the application
  if Pos('private', aItem.Tags) > 0 then
    aSync := False;
end;
```

Le fichier d'état

L'état est du JSON (UTF-8) : le nom et la version du format, l'id de l'agenda, le jeton de synchronisation, le fuseau horaire et, pour chaque élément synchronisé, l'id local, l'id de l'événement, l'etag, l'heure `updated`, l'empreinte des champs synchronisés, l'instant où l'élément a été modifié et les lignes de récurrence supplémentaires. StateFileName le charge avant la première synchronisation et l'enregistre après chacune, via un fichier temporaire ensuite renommé. SaveStateToStream et LoadStateFromStream le conservent ailleurs. L'état d'un autre agenda est effacé quand CalendarId change.
Conservez l'état avec les éléments qu'il décrit. Un élément synchronisé absent de Data compte comme supprimé localement, et la synchronisation suivante supprime son événement dans Google. Quand les éléments ne sont pas enregistrés, laissez StateFileName vide, de sorte que chaque démarrage commence par une synchronisation complète dans un planificateur vide. ResetState force une synchronisation complète sans rien supprimer.

Erreurs et synchronisations complètes

- Une erreur lors de la liste ne change rien. Une erreur lors de l'envoi s'arrête au premier appel en échec. L'état conserve ce qui a réussi et l'ancien jeton de synchronisation, de sorte que la synchronisation suivante relit les mêmes changements distants et réessaie le reste. OnError donne la raison.
- Quand Google répond 410 Gone à un jeton de synchronisation expiré, le jeton est effacé et l'exécution recommence comme une liste complète. La correspondance des éléments est conservée, donc rien n'est inséré deux fois. Un événement associé absent de la liste complète compte comme supprimé dans Google, avec la règle qu'une modification l'emporte sur une suppression.
- Une mise à jour répondue par 404 ou 410 est renvoyée comme une insertion. Une suppression répondue par 404 ou 410 compte comme faite.

Synchronisation Google et abonnement iCalendar ensemble

Un TsgcPlannerICalSubscription peut fusionner un flux en lecture seule dans les mêmes données. Ses éléments reçoivent l'IdPrefix `ics:`, qui est l'ExcludeIdPrefix par défaut de la synchronisation Google, de sorte que les événements du flux sont affichés mais jamais envoyés à Google. Si vous changez un préfixe, changez l'autre.

Démos

Les démos Delphi et C++Builder GoogleCalendarSync et ICalSubscription montrent les deux composants sur un TsgcVCLScheduler. Lancez-les avec `--test` pour un autotest hors ligne sur des données de test, sans compte Google. Voir Démos.

Voir aussi

- TsgcPlannerGoogleSync
- TsgcPlannerICalSubscription
- Synchronisation avec Google Agenda et iCalendar en .NET
- Récurrence
- iCalendar
- TsgcPlannerData

TsgcPlannerGoogleSync

Unité : sgcPlannerWS_Google
TsgcPlannerGoogleSync maintient les éléments d'un TsgcPlannerData en synchronisation bidirectionnelle avec un Google Agenda. Il fait partie du paquet de passerelle sgcPlannerWS (édition Registered) et appelle l'API Google Calendar via le **TsgcHTTPGoogleCloud_Calendar_Client** de sgcWebSockets. Le client n'est utilisé que comme transport authentifié : le composant appelle ses méthodes de niveau API (Event_List, Event_Insert, Event_Patch, Event_Delete et Calendar_Get) et lit et écrit lui-même le JSON des événements, de sorte que l'id d'agenda `primary` fonctionne et que l'OnError du client reste à l'application. Voir Synchronisation avec Google Agenda pour la configuration Google Cloud, la correspondance des champs, la règle de conflit et le fichier d'état.

- Une synchronisation liste les événements modifiés (avec le jeton de synchronisation enregistré, sinon tous, en suivant nextPageToken), les applique à Data dans le thread principal, puis envoie les modifications locales (insertions, mises à jour et suppressions) et enregistre dans l'état les nouveaux ID d'événement, etags, empreintes et le jeton de synchronisation.
- Les modifications locales sont détectées par l'empreinte : un élément dont les champs synchronisés diffèrent de la dernière synchronisation est envoyé. Aucune heure de modification par élément n'est nécessaire.
- Un élément modifié des deux côtés est un conflit. La modification la plus récente l'emporte, une égalité revient à Google, et OnConflict peut le remplacer. Une modification l'emporte toujours sur une suppression.
- Une erreur HTTP ou d'autorisation arrête l'exécution, conserve l'ancien jeton de synchronisation (de sorte que la synchronisation suivante réessaie) et déclenche OnError.

Propriétés

Propriété	Type	Description
Active	Boolean	Par défaut True. Sync et SyncAsync ne font rien et renvoient False tant que Active vaut False. La synchronisation automatique de AutoSyncInterval ne s'exécute que lorsque Active est vrai.
AutoSyncInterval	Integer	Secondes entre deux synchronisations automatiques, 0 (par défaut) les désactive. Quand Active est vrai, une valeur supérieure à 0 synchronise aussitôt dans un thread de travail, puis toutes les AutoSyncInterval secondes.
CalendarId	string	L'agenda à synchroniser. Par défaut `primary`, l'agenda principal du compte. L'id d'un autre agenda (affiché dans ses paramètres dans Google Agenda) fonctionne aussi.
Client	TsgcHTTPGoogleCloud_Calendar_Client	Le client Google Agenda de sgcWebSockets qui authentifie et envoie les requêtes. L'application ne doit pas l'utiliser tant que Busy est vrai.
Data	TsgcPlannerData	Les données du planificateur à synchroniser.
Direction	TsgcPlannerGoogleDirection	gsdTwoWay (par défaut) lit et écrit. gsdPullOnly n'écrit jamais dans Google et les modifications locales ne comptent jamais, donc Google l'emporte toujours. gsdPushOnly envoie seulement les modifications locales et ne lit pas les modifications faites dans Google.

ExcludeIdPrefix	string	Les éléments dont l'Id commence par cette valeur ne sont jamais envoyés. Par défaut `ics:` , l'IdPrefix par défaut de TsgcPlannerICalSubscription, de sorte que les événements d'un abonnement ne vont pas vers Google.
StateFileName	string	Le fichier JSON de l'état de synchronisation. Il est chargé avant la première synchronisation et enregistré après chacune. Vide (par défaut), il garde l'état en mémoire seulement.
TimeZone	string	Le nom du fuseau horaire IANA (par exemple `Europe/Madrid`) écrit avec les événements récurrents et leurs instances, ce que Google exige. Vide (par défaut), il utilise le fuseau horaire de l'agenda, lu une fois.
Busy	Boolean	Public, en lecture seule. True pendant qu'une synchronisation s'exécute.
LastSyncTime	TDateTime	Public, en lecture seule. L'heure locale à la fin de la dernière synchronisation réussie.
State	TsgcPlannerGoogleSyncState	Public, en lecture seule. L'état de la synchronisation : jeton de synchronisation, id d'agenda, fuseau horaire et une entrée par élément associé. Ne le modifiez que lorsque Busy est faux.

Méthodes

Méthode	Renvoie	Description
Sync	Boolean	Exécute une synchronisation dans le thread appelant (principal) et revient à sa fin. False quand Active est faux, quand Busy est vrai (OnError l'indique) ou quand l'exécution a échoué (OnError en donne la raison).
SyncAsync	Boolean	Démarre une synchronisation dans un thread de travail et revient aussitôt. Data est modifié et chaque événement se déclenche dans le thread principal, via Synchronize. False quand Active est faux ou Busy est vrai.
ResetState		Oublie le jeton de synchronisation et les éléments associés, de sorte que la synchronisation suivante soit complète. Ignoré tant que Busy est vrai.
SaveStateToStream		Écrit l'état en JSON (UTF-8), pour les applications qui le gardent avec leurs propres données au lieu de StateFileName.
LoadStateFromStream	Boolean	Lit un état écrit par SaveStateToStream ou enregistré dans StateFileName. False, l'état restant inchangé, quand le texte n'est pas un état de ce format et de cette version.

Événements

Événement	Type	Description

On-Conflict	TsgcPlan-nerGoogle-Conflic-tEvent	Un élément modifié ici et dans Google depuis la dernière synchronisation : aItem est l'élément local et aEventJSON l'événement tel que Google l'a renvoyé. aReso-lution commence à gcrLastModified (la modification la plus récente l'emporte, une égalité revient à Google). Définissez gcrKeepLocal, gcrKeepRemote ou gcrSkip pour laisser les deux côtés tels quels dans cette exécution (l'élément reste modi-fié). Une modification face à une suppression n'est pas signalée ici, la modification l'emporte.
OnEr-ror	TsgcPlan-nerWSEr-rorEvent	aError indique pourquoi une synchronisation a échoué : un statut HTTP, une erreur d'autorisation ou une synchronisation déjà en cours. Il signale aussi, une fois par règle, une règle de récurrence que le planificateur ne peut pas développer ; cette règle est quand même synchronisée sans changement.
OnFil-terItem	TsgcPlan-nerGoogle-FilterIte-mEvent	Appelé pour chaque élément local avant son envoi. Définissez aSync sur False pour garder l'élément hors de Google.
On-Sync-Done	TsgcPlan-nerGoogle-SyncDo-neEvent	Une synchronisation s'est terminée sans erreur : aPulled événements appliqués ici, aPushed éléments écrits dans Google, aDeleted événements supprimés dans Google et aConflicts conflits trouvés.

Threads

- SyncAsync et la synchronisation automatique parlent à Google dans un thread de travail et modifient Data dans le thread principal, dans un seul BeginUpdate et EndUpdate, de sorte que les vues ne se redessinent qu'une fois. Chaque événement se déclenche dans le thread principal.
- Tant que Busy est vrai, seule la synchronisation en cours utilise Client. N'appelez pas le client depuis l'application à ce moment.
- Détruire le composant arrête la synchronisation automatique et attend celle en cours. Ne libérez pas le com-posant dans ses propres gestionnaires d'événements.

Méthodes protégées

Le transport est un ensemble de méthodes virtuelles protégées. Chacune renvoie la réponse brute et son statut HTTP, et s'exécute dans le thread de travail (ou dans le thread principal pour Sync). L'implémentation par défaut appelle le client sur CalendarId. Un descendant peut les surcharger pour tester la synchronisation sans réseau, comme le font les démos avec leur option `--test`.

Méthode	Description
DoListEvents(const aPageToken, aSyncTo-ken: string; out aStatus: Integer): string	Une page de la liste des événements.
DoInsertEvent(const aJSON: string; out aSta-tus: Integer): string	Insère un événement.
DoUpdateEvent(const aEventId, aJSON: string; out aStatus: Integer): string	Modifie un événement par patch. aJSON ne contient que les champs dont le planificateur est propriétaire, de sorte que les autres champs de l'événement sont conservés.
DoDeleteEvent(const aEventId: string; out aStatus: Integer): string	Supprime un événement. 404 et 410 comptent comme fait.
DoGetCalendar(out aStatus: Integer): string	Lit l'agenda, pour son fuseau horaire, quand aucune liste d'événements n'en a donné.
DoError(const aError: string)	Déclenche OnError.

Exemple

Un objet de synchronisation pour une fiche ou un service : OAuth2 avec un client Application de bureau, l'état enregistré à côté de l'exécutable, une synchronisation automatique toutes les cinq minutes et un journal.

```
uses
  SysUtils, Classes,
  sgcHTTP, sgcHTTP_Google_Cloud, sgcPlanner_Data, sgcPlannerWS_Classes,
  sgcPlannerWS_Google;

type
  TCalendarSync = class
  private
    FClient: TsgcHTTPGoogleCloud_Calendar_Client;
    FLog: TStrings;
    FSync: TsgcPlannerGoogleSync;
    procedure SyncDone(Sender: TObject; aPulled, aPushed, aDeleted,
      aConflicts: Integer);
    procedure SyncError(Sender: TObject; const aError: string);
  public
    constructor Create(aData: TsgcPlannerData; aLog: TStrings;
      const aClientId, aClientSecret: string);
    destructor Destroy; override;
    procedure SyncNow;
  end;

constructor TCalendarSync.Create(aData: TsgcPlannerData; aLog: TStrings;
  const aClientId, aClientSecret: string);
begin
  inherited Create;
  FLog := aLog;
  // the sgcWebSockets client: OAuth2 with a Desktop app client of Google
  FClient := TsgcHTTPGoogleCloud_Calendar_Client.Create(nil);
  FClient.GoogleCloudOptions.Authentication := gcaOAuth2;
  FClient.GoogleCloudOptions.OAuth2.ClientId := aClientId;
  FClient.GoogleCloudOptions.OAuth2.ClientSecret := aClientSecret;
  FClient.GoogleCloudOptions.OAuth2.LocalIP := '127.0.0.1';
  FClient.GoogleCloudOptions.OAuth2.LocalPort := 8080;
  FClient.Scopes.CalendarsReadWrite := True;
  FClient.Scopes.EventsReadWrite := True;

  FSync := TsgcPlannerGoogleSync.Create(nil);
  FSync.Client := FClient;
  FSync.Data := aData;
  FSync.CalendarId := 'primary';
  // keep the sync token and the mapping next to the saved items
  FSync.StateFileName := ExtractFilePath(ParamStr(0)) + 'calendar.state.json';
  FSync.OnSyncDone := SyncDone;
  FSync.OnError := SyncError;
  // sync at once and then every 5 minutes
  FSync.AutoSyncInterval := 300;
end;

destructor TCalendarSync.Destroy;
begin
  // stops the automatic sync and waits for a running one
  FSync.Free;
  FClient.Free;
  inherited;
end;

procedure TCalendarSync.SyncNow;
begin
  if not FSync.SyncAsync then
    FLog.Add('a sync is already running');
end;
```

```
procedure TCalendarSync.SyncDone(Sender: TObject; aPulled, aPushed, aDeleted,
  aConflicts: Integer);
begin
  FLog.Add(Format('pulled %d, pushed %d, deleted %d, conflicts %d',
    [aPulled, aPushed, aDeleted, aConflicts]));
end;

procedure TCalendarSync.SyncError(Sender: TObject; const aError: string);
begin
  // the sync token is kept, the next sync retries
  FLog.Add('sync failed: ' + aError);
end;
```

Voir aussi

- Synchronisation avec Google Agenda
- TsgcPlannerICalSubscription
- Synchronisation avec Google Agenda et iCalendar en .NET
- TsgcPlannerData
- Récurrence
- Éditions
- Démos

TsgcPlannerICalSubscription

Unité : sgcPlannerWS_ICal
TsgcPlannerICalSubscription abonne un TsgcPlannerData à un flux iCalendar (.ics) en lecture seule publié sur une URL, comme l'adresse publique ou secrète d'un agenda Google, Outlook ou Apple, un agenda d'équipe ou une liste de jours fériés. Il télécharge le flux avec le client HTTP de sgcWebSockets, le lit avec TsgcPlannerICalendar et fusionne ses événements dans les données. Il fait partie du paquet de passerelle sgcPlannerWS (édition Registered, SGC_PLANNER_ADVANCED).

Téléchargement

- Une URL `webcal://` ou `webcals://` est récupérée comme `https://`.
- Le composant suit lui-même jusqu'à cinq redirections (301, 302, 303, 307 et 308), les valeurs relatives de Location étant résolues par rapport à l'URL courante.
- Il envoie If-None-Match et If-Modified-Since avec l'ETag et le Last-Modified de la dernière réponse importée. 304 Not Modified saute l'import. Un changement d'URL ou d'IdPrefix oublie ces validateurs.
- Un corps plus grand que MaxSize, tout autre statut, une erreur de transport ou un flux sans BEGIN:VCALENDAR déclenche OnError et laisse les éléments et les validateurs tels quels, de sorte que l'actualisation suivante réessaie.

Fusion

- Chaque élément du flux reçoit IdPrefix plus son id : l'UID de l'événement, ou UID#RECURRENCE-ID pour une occurrence modifiée. ParentId et RecurrenceParentId reçoivent le même préfixe.
- Les nouveaux ids sont ajoutés, les ids existants dont les champs diffèrent sont remplacés en bloc, et les éléments de l'abonnement qui ne sont plus dans le flux sont supprimés, le tout dans un seul BeginUpdate et EndUpdate de Data.
- Un élément appartient à l'abonnement quand son Id commence par IdPrefix. Avec un IdPrefix vide, les éléments de la dernière actualisation lui appartiennent (gardés en mémoire seulement). Les autres éléments de Data ne sont jamais touchés, de sorte que des éléments locaux et plusieurs abonnements, chacun avec son propre IdPrefix, partagent un seul objet de données.
- Les noms RESOURCES du flux retrouvent les ressources de Data, et les nouvelles ressources sont ajoutées à Data.
- Les modifications locales des éléments d'abonnement sont écrasées par la prochaine actualisation qui importe. Opposez-leur un veto dans OnItemChanging de Data avec IsSubscriptionItem, comme le montre l'exemple.

Propriétés

Propriété	Type	Description
Active	Boolean	Par défaut False. True démarre un thread de travail qui actualise aussitôt, puis toutes les RefreshInterval minutes (une seule fois quand RefreshInterval vaut 0).
Data	TsgcPlannerData	Les données du planificateur dans lesquelles le flux est fusionné.
IdPrefix	string	Le préfixe des ids des éléments du flux. Par défaut `ics:` , qui est aussi l'ExcludeIdPrefix par défaut de TsgcPlannerGoogleSync, de sorte que les événements du flux ne sont jamais envoyés à Google.
MaxSize	Integer	Le plus grand corps accepté, en octets. Par défaut 10485760 (10 Mo) ; 0 ou moins signifie sans limite.

RefreshInterval	Integer	Minutes entre deux actualisations tant que Active est vrai, 0 (par défaut) signifie une seule actualisation.
TLSOptions	TsgcHTTPTLS_Options	Les options TLS du client HTTP interne, pour les flux HTTPS.
URL	string	L'adresse du flux : https://, http://, webcal:// ou webcals://.
Busy	Boolean	Public, en lecture seule. True pendant qu'une actualisation s'exécute.
HTTPClient	TsgcHTTP1Client	Public, en lecture seule. Le client HTTP interne de sgcWebSockets, pour un proxy ou d'autres réglages. Tant que Busy est vrai, seule l'actualisation en cours l'utilise.
LastRefreshTime	TDateTime	Public, en lecture seule. L'heure locale de la dernière actualisation réussie, 0 s'il n'y en a aucune.

Méthodes

Méthode	Renvoie	Description
Refresh	Boolean	Télécharge et importe le flux dans le thread appelant (principal). False quand Busy est vrai ou en cas d'erreur.
RefreshAsync	Boolean	Démarre une actualisation dans un thread de travail et revient aussitôt. L'import et chaque événement s'exécutent dans le thread principal. False quand Busy est vrai.
IsSubscriptionItem(aItem)	Boolean	True quand aItem appartient à cet abonnement.

Événements

Événement	Type	Description
OnBeforeRefresh	TNotifyEvent	Une actualisation démarre, avant le téléchargement.
OnError	TsgcPlannerWSErrorEvent	Une actualisation a échoué : aError donne le statut HTTP, la limite de taille, trop de redirections, une erreur de transport ou un flux qui ne s'analyse pas. Les éléments restent tels quels.
OnImportWarning	TsgcPlannerICalWarningEvent	Une partie du flux que TsgcPlannerICalendar a ignorée ou lue approximativement, avec aLine et aMessage.
OnRefreshed	TsgcPlannerICalRefreshedEvent	Une actualisation a réussi : aAdded, aUpdated et aDeleted éléments. Une réponse 304 Not Modified donne 0, 0 et 0.

Threads

Refresh bloque le thread appelant. RefreshAsync et la boucle de Active téléchargent dans un thread de travail et importent dans le thread principal via Synchronize, de sorte que chaque événement se déclenche dans le thread principal. Détruire le composant arrête le thread de travail. Ne libérez pas le composant dans ses propres gestionnaires d'événements.

Méthodes protégées

Méthode	Description
DoFetch(const aURL: string; aRequestHeaders, aResponseHeaders: TStrings; aBody: TStream): Integer	Un GET de aURL qui ne suit pas les redirections. aRequestHeaders contient les lignes d'en-tête de la requête (Nom : valeur), aResponseHeaders reçoit les lignes d'en-tête de la réponse et aBody le corps. Renvoie le statut HTTP et lève une exception en cas d'erreur de transport. Appelé dans le thread de l'actualisation ; les tests et les démos la surchargent pour lire un fichier local.
DoGetRefreshIntervalMs: Cardinal	L'intervalle de la boucle de Active en millisecondes.
DoError, DoRefreshed	Déclenchent OnError et OnRefreshed.

Exemple

Un agenda d'équipe affiché à côté des éléments locaux, actualisé toutes les heures, avec les événements du flux en lecture seule dans les vues.

```
uses
  SysUtils, Classes,
  sgcPlanner_Data, sgcPlannerWS_Classes, sgcPlannerWS_ICal;

type
  TTeamFeed = class
  private
    FLog: TStrings;
    FSubscription: TsgcPlannerICalSubscription;
    procedure ItemChanging(Sender: TObject; aItem: TsgcPlannerItem;
      const aFields: TsgcPlannerItemFields; var aAccept: Boolean);
    procedure Refreshed(Sender: TObject; aAdded, aUpdated, aDeleted: Integer);
    procedure FeedError(Sender: TObject; const aError: string);
  public
    constructor Create(aData: TsgcPlannerData; aLog: TStrings;
      const aURL: string);
    destructor Destroy; override;
  end;

constructor TTeamFeed.Create(aData: TsgcPlannerData; aLog: TStrings;
  const aURL: string);
begin
  inherited Create;
  FLog := aLog;
  FSubscription := TsgcPlannerICalSubscription.Create(nil);
  FSubscription.Data := aData;
  FSubscription.URL := aURL; // webcal:// is fetched as https://
  FSubscription.IdPrefix := 'team:';
  FSubscription.RefreshInterval := 60; // minutes
  FSubscription.OnRefreshed := Refreshed;
  FSubscription.OnError := FeedError;
  // feed events are read only in the views
  aData.OnItemChanging := ItemChanging;
  // refresh at once, then every hour
```

```
  FSubscription.Active := True;
end;

destructor TTeamFeed.Destroy;
begin
  FSubscription.Free;
  inherited;
end;

procedure TTeamFeed.ItemChanging(Sender: TObject; aItem: TsgcPlannerItem;
  const aFields: TsgcPlannerItemFields; var aAccept: Boolean);
begin
  if FSubscription.IsSubscriptionItem(aItem) then
    aAccept := False;
end;

procedure TTeamFeed.Refreshed(Sender: TObject;
  aAdded, aUpdated, aDeleted: Integer);
begin
  FLog.Add(Format('added %d, updated %d, deleted %d',
    [aAdded, aUpdated, aDeleted]));
end;

procedure TTeamFeed.FeedError(Sender: TObject; const aError: string);
begin
  // the items of the last good refresh stay in place
  FLog.Add('feed failed: ' + aError);
end;
```

Voir aussi

- TsgcPlannerICalendar
- TsgcPlannerGoogleSync
- Synchronisation avec Google Agenda
- iCalendar
- Synchronisation avec Google Agenda et iCalendar en .NET
- TsgcPlannerData
- Récurrence
- Éditions
- Démos

Vue d'ensemble de .NET

sgcPlanner .NET est un portage C# managé de la bibliothèque Delphi sgcPlanner, pas une enveloppe autour d'elle. Les classes C# vivent dans le namespace **esegece.sgcPlanner** et gardent les noms de Delphi, de sorte que **TsgcPlannerData**, **TsgcPlannerItem**, **TsgcPlannerCalendar**, **TsgcPlannerRenderer** et les énumérations se lisent de la même façon dans les deux langages. Les trois vues suivent le modèle de nommage de Delphi sur chaque hôte : **TsgcVCLGantt** devient **TsgcWinFormsGantt**, **TsgcWpfGantt** et **TsgcAvaloniaGantt**, et de même pour le Scheduler et le tableau Kanban. Les vues sont dessinées avec le portage .NET de **sgcUICore**, le même canevas, les mêmes polices et les mêmes thèmes clair et sombre que les autres produits de la suite visuelle eSeGeCe.
Comme dans Delphi, un seul **TsgcPlannerData** contient les éléments, les ressources, les statuts, les dépendances et le calendrier de travail, et un nombre quelconque de vues l'affichent. Une modification dans une vue passe par les données, qui notifient chaque vue liée une fois, voir Données et planification dans .NET.

Paquets

Le portage est divisé en cinq paquets NuGet, de sorte qu'un projet n'embarque que l'hôte qu'il utilise.

PackageId	Contient	Frameworks cibles
esegece.sgcPlanner	Les données, le calendrier de travail, l'échelle de temps, la récurrence, le moteur de planification, les modèles de vue et les painters du Scheduler, de la Timeline des ressources, du Gantt et du tableau Kanban, la source **System.Data.DataTable**, JSON, iCalendar et MS Project XML, et la sortie SVG des vues. Aucune dépendance native. Dépend de esegece.sgcUICore.	netstandard2.0, net472, net6.0, net8.0, net9.0
esegece.sgcPlanner.Rendering	La sortie PNG sans fenêtre : **TsgcPlannerRenderer** et les méthodes d'extension **SaveToPNG**, **ExportPNG** et **RenderToRaster** de **sgcPlanner_Raster**, dessinées sur le canevas SkiaSharp de esegece.sgcUICore.Rendering, sous Windows, Linux et macOS.	netstandard2.0, net472, net6.0, net8.0, net9.0
esegece.sgcPlanner.WinForms	**TsgcWinFormsScheduler**, **TsgcWinFormsGantt** et **TsgcWinFormsKanban** sur **TsgcPlannerWinFormsView**.	net472, net8.0-windows, net9.0-windows
esegece.sgcPlanner.Wpf	**TsgcWpfScheduler**, **TsgcWpfGantt** et **TsgcWpfKanban** sur **TsgcPlannerWpfView**.	net472, net8.0-windows, net9.0-windows
esegece.sgcPlanner.Avalonia	**TsgcAvaloniaScheduler**, **TsgcAvaloniaGantt** et **TsgcAvaloniaKanban** sur **TsgcPlannerAvaloniaView**, sous Windows, Linux et macOS avec Avalonia 11.	net8.0, net9.0

Les trois paquets d'hôte dépendent de esegece.sgcPlanner.Rendering, qui dépend de esegece.sgcPlanner, de sorte qu'une ligne `PackageReference` apporte tout ce dont un hôte a besoin. Voir Installation depuis NuGet. La synchronisation avec Google Agenda et l'abonnement iCalendar de la passerelle sgcWebSockets se trouvent dans le paquet séparé **esegece.sgcPlanner.sgcWebSockets**, uniquement dans l'édition Registered.

Éditions

Édition	Ids de paquet	Comprend
Registered	esegece.sgcPlanner, .Rendering, .WinForms, .Wpf et .Avalonia	Tout : les trois vues sur les trois hôtes, glisser-déposer, tâches récapitulatives, jalons, progression, thèmes, la source DataTable, JSON, la sortie PNG et SVG sans fenêtre et les fonctionnalités avancées : récurrence, le moteur de planification avec flèches de dépendance, le chemin critique et les plannings de référence, la Timeline des ressources, les swimlanes et les limites de WIP sur le tableau, iCalendar et MS Project XML.
Community	Les mêmes ids avec le suffixe .Community, par exemple esegece.sgcPlanner.WinForms.Community.	Tout sauf les fonctionnalités avancées. Les propriétés restent, de sorte que le même code de formulaire se compile dans les deux éditions : un élément récurrent s'affiche une fois à son début, **ShowDependencies**, **AutoSchedule**, **ShowCriticalPath** et **ShowBaseline** sont inertes, **Schedule** et **SetBaseline** du Gantt déclenchent **EsgcPlannerError**, **psvTimeline** peint un avis et **SwimlaneBy** agit comme **psINone**. Les classes des unités avancées ne sont pas dans les paquets Community : **TsgcPlannerScheduleEngine**, **TsgcPlannerICalendar**, **TsgcPlannerMSProject**, **TsgcPlannerSchedulerTimelineView** et les classes de récurrence.

C'est la même répartition que les éditions de Delphi, voir Éditions.

Hôtes

Hôte	Équivalent Delphi	Plateformes
WinForms	VCL (contrôles TsgcVCL)	Windows, .NET Framework 4.7.2 et .NET 8 et 9.
WPF	VCL et FMX	Windows, .NET Framework 4.7.2 et .NET 8 et 9. Les contrôles peuvent être déclarés en XAML.
Avalonia	FMX (contrôles TsgcFMX)	Windows, Linux et macOS, .NET 8 et 9.
Sans fenêtre	TsgcPlannerRenderer et les modèles de vue	Toute plateforme des frameworks cibles : SVG avec esegece.sgcPlanner, PNG avec esegece.sgcPlanner.Rendering.

Chaque hôte a la même API : les propriétés, méthodes et événements des contrôles Delphi avec les mêmes noms, plus **Data**, **Theme**, **Colors**, **ReadOnly**, **DragOptions**, **SyncSelection**, **ItemAt**, **ScrollToItem**, **SaveToPNG** et **SaveToSVG** sur la classe de base. Voir Classes .NET pour les noms de classes et Différences de .NET par rapport à Delphi pour ce qui change par rapport à Delphi.

Voir aussi

- Installation depuis NuGet
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Classes .NET
- Différences de .NET par rapport à Delphi
- Données et planification dans .NET
- Liaison de données dans .NET
- JSON, iCalendar et MS Project dans .NET
- Rendu sans fenêtre dans .NET
- Démos .NET
- Plateformes

Installation depuis NuGet

Les paquets sont sur nuget.org. Ajoutez le paquet de l'hôte, les paquets core et de rendu viennent avec lui comme dépendances.

Édition Registered

```
dotnet add package esegece.sgcPlanner.WinForms
dotnet add package esegece.sgcPlanner.Wpf
dotnet add package esegece.sgcPlanner.Avalonia
```

Édition Community

```
dotnet add package esegece.sgcPlanner.WinForms.Community
dotnet add package esegece.sgcPlanner.Wpf.Community
dotnet add package esegece.sgcPlanner.Avalonia.Community
```

Ou ajoutez la référence au fichier de projet à la main :

```
<ItemGroup>
  <PackageReference Include="esegece.sgcPlanner.Wpf" Version="2026.10.0" />
</ItemGroup>
```

Référencez soit les paquets Registered soit les paquets Community dans un projet, jamais les deux : ils contiennent les mêmes types dans le même namespace.

Paramètres du projet

- WinForms : un projet `net8.0-windows` avec `<UseWindowsForms>true</UseWindowsForms>`, ou un projet .NET Framework 4.7.2.
- WPF : un projet `net8.0-windows` avec `<UseWPF>true</UseWPF>`, ou un projet .NET Framework 4.7.2. Associez le namespace en XAML avec `xmlns:sgc="clr-namespace:esegece.sgcPlanner;assembly=esegece.sgcPlanner.Wpf"`.
- Avalonia : un projet `net8.0` ou `net9.0` avec les paquets **Avalonia.Desktop** et **Avalonia.Themes.Fluent** d'Avalonia 11.3, comme toute application de bureau Avalonia. Les barres de défilement des vues tirent leur apparence du thème Fluent. Associez le namespace en XAML avec `xmlns:sgc="using:esegece.sgcPlanner"`.
- Sans fenêtre : esegece.sgcPlanner seul suffit pour les données, les formats de fichier et SVG. Ajoutez esegece.sgcPlanner.Rendering pour le PNG, il apporte SkiaSharp et ses ressources natives pour Windows et Linux.
- Synchronisation avec Google Agenda et abonnement iCalendar : ajoutez **esegece.sgcPlanner.sgcWebSockets** (édition Registered), qui dépend de esegece.sgcPlanner et de esegece.sgcWebSockets.

En C#, chaque hôte a besoin de `using esegece.sgcPlanner;`, et de `using esegece.sgcUICore;` pour **TsgcUITheme**, **TsgcUINotifyEvent** et les autres types de sgcUICore.

Voir aussi

- Vue d'ensemble de .NET
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Démos .NET

Démarrage rapide WinForms

Ce démarrage rapide construit un formulaire dans le code avec les trois vues WinForms liées à un **TsgcPlannerData** : un diagramme de Gantt en haut, un tableau Kanban à gauche et une semaine de travail à droite. Faites glisser une carte vers une autre colonne et la colonne de statut du Gantt suit.

Étapes

1. Créez un projet Windows Forms pour `net8.0-windows` et ajoutez le paquet **esegece.sgcPlanner.WinForms** (**esegece.sgcPlanner.WinForms.Community** dans l'édition Community).
2. Ajoutez `using esegece.sgcPlanner;` et `using esegece.sgcUICore;`.
3. Créez un **TsgcPlannerData** et remplissez-le : statuts, ressources et éléments.
4. Créez les contrôles, affectez **Data** et ajoutez-les au formulaire. Ce sont des contrôles WinForms ordinaires : Dock, Anchor, TabIndex et Enabled fonctionnent comme d'habitude.
5. Affectez les événements. Ce sont les événements des contrôles Delphi, avec les mêmes noms.

Exemple

Notes

- Le tableau Kanban prend une colonne par statut de **Data.Statuses** (**ColumnSource** est **pkcStatuses**). Définissez **ColumnSource** sur **pkcList** et remplissez **Columns** pour choisir les colonnes et leurs limites de WIP.
- La propriété **View** du Scheduler est le type de vue (**psvDay**, **psvWorkWeek**, **psvWeek**, **psvMonth**, **psvAgenda**, **psvTimeline**), comme dans Delphi. Le modèle de vue est **SchedulerView** ; le Gantt et le tableau exposent le leur sous les noms **GanttView** et **KanbanView**.
- **Theme** est propre à chaque contrôle. Les barres de défilement sont les barres natives de Windows, et **BorderStyle** choisit le bord.
- **SaveToPNG** et **SaveToSVG** écrivent ce que le contrôle affiche, à sa taille ou à une autre taille et échelle.

Voir aussi

- Vue d'ensemble de .NET
- Installation depuis NuGet
- Classes .NET
- Données et planification dans .NET
- Démarrage rapide VCL

Démarrage rapide WPF

Les contrôles WPF peuvent être déclarés en XAML comme tout autre contrôle. Ce démarrage rapide déclare un Gantt, un tableau Kanban et un Scheduler dans une grille et les lie à un **TsgcPlannerData** dans le code-behind.

Étapes

1. Créez un projet WPF pour `net8.0-windows` et ajoutez le paquet **esegece.sgcPlanner.Wpf** (**esegece.sgcPlanner.Wpf.Community** dans l'édition Community).
2. Associez le namespace dans la fenêtre : `xmlns:sgc="clr-namespace:esegece.sgcPlanner;assembly=esegece.sgcPlanner.Wpf"`.
3. Déclarez les contrôles. Les propriétés simples peuvent être définies comme attributs, et les énumérations prennent les noms de leurs valeurs, comme `View="psvWorkWeek"`. Un contrôle sans Width ni Height dans un panneau qui ne l'étire pas demande 400 x 300.
4. Créez les données dans le code-behind et affectez-les à **Data** de chaque contrôle.

Exemple

La fenêtre :

```
<Window x:Class="QuickStart.QuickStartWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:sgc="clr-namespace:esegece.sgcPlanner;assembly=esegece.sgcPlanner.Wpf"
        Title="sgcPlanner" Width="1100" Height="700">
  <Grid>
    <Grid.RowDefinitions>
      <RowDefinition Height="260" />
      <RowDefinition Height="*" />
    </Grid.RowDefinitions>
    <Grid.ColumnDefinitions>
      <ColumnDefinition Width="560" />
      <ColumnDefinition Width="*" />
    </Grid.ColumnDefinitions>
    <sgc:TsgcWpfGantt x:Name="ganTasks" Grid.ColumnSpan="2" TreeWidth="260" ShowToday="False" />
    <sgc:TsgcWpfKanban x:Name="kbnBoard" Grid.Row="1" />
    <sgc:TsgcWpfScheduler x:Name="schWeek" Grid.Row="1" Grid.Column="1" View="psvWorkWeek" />
  </Grid>
</Window>
```

Le code-behind :

Notes

- Les propriétés sont de simples propriétés CLR, pas des propriétés de dépendance, elles acceptent donc des valeurs littérales en XAML mais pas de `{Binding}`. Les données d'un plan se lient par **TsgcPlannerData**, voir Liaison de données dans .NET.
- Les barres de défilement sont des enfants `ScrollBar` WPF du contrôle. **IsTabStop** prend la place du **TabStop** de WinForms, et il n'y a pas de **BorderStyle** : utilisez BorderBrush et BorderThickness ou un Border.
- Les contrôles exécutent un minuteur pour la ligne de l'heure actuelle tant qu'ils sont chargés. **Dispose** arrête les minuteurs et détache le listener des données ; appelez-le à la fermeture de la fenêtre.

Voir aussi

- Vue d'ensemble de .NET
- Installation depuis NuGet
- Classes .NET

- Liaison de données dans .NET
- TsgcVCLGantt

Démarrage rapide Avalonia

Les contrôles Avalonia s'exécutent sous Windows, Linux et macOS. Ce démarrage rapide est une application Avalonia complète dans le code : un tableau Kanban à côté d'une semaine regroupée par ressource, tous deux dans le thème sombre et liés à un **TsgcPlannerData**.

Étapes

1. Créez un projet console `net8.0`, définissez `<OutputType>WinExe</OutputType>` et ajoutez **Avalonia.Desktop** et **Avalonia.Themes.Fluent** 11.3 ainsi que le paquet **esegece.sgcPlanner.Avalonia** (**esegece.sgcPlanner.Avalonia.Community** dans l'édition Community).
2. Ajoutez le thème Fluent à l'application.
3. Créez les données et les contrôles et ajoutez les contrôles à un panneau. En XAML, associez le namespace avec `xmlns:sgc="using:esegece.sgcPlanner"`.

Exemple

Notes

- **Theme** des contrôles est le thème de sgcUICore. Il masque le **StyledElement.Theme** d'Avalonia, définissez-le donc par une variable du type du contrôle, comme dans l'exemple.
- **GroupByResource** donne à chaque personne un groupe de colonnes dans les vues jour, semaine de travail et semaine, dans chaque édition. La Timeline des ressources (**psvTimeline**) est une fonctionnalité de Registered.
- Les propriétés sont de simples propriétés CLR, pas des propriétés stylées : définissez-les dans le code ou comme attributs XAML.
- Sur macOS la touche Commande compte comme Ctrl, donc Commande et la molette zooment le Gantt.

Voir aussi

- Vue d'ensemble de .NET
- Installation depuis NuGet
- Classes .NET
- Données et planification dans .NET
- Démarrage rapide FireMonkey

Classes .NET

Chaque classe Delphi garde son nom dans .NET. Les contrôles VCL et FireMonkey deviennent les contrôles WinForms, WPF et Avalonia avec le préfixe de l'hôte, et les classes indépendantes du framework sont les mêmes sur chaque hôte.

Classes d'hôte

Delphi	WinForms	WPF	Avalonia	Modèle (propriété de l'hôte)
TsgcVCLScheduler, TsgcFMXScheduler	TsgcWinFormsScheduler	TsgcWpfScheduler	TsgcAvaloniaScheduler	TsgcPlannerSchedulerView (SchedulerView) : TsgcPlannerSchedulerTimelineView dans l'édition Registered, TsgcPlannerSchedulerRenderView dans l'édition Community
TsgcVCLGantt, TsgcFMXGantt	TsgcWinFormsGantt	TsgcWpfGantt	TsgcAvaloniaGantt	TsgcPlannerGanttRenderView (GanttView)
TsgcVCLKanban, TsgcFMXKanban	TsgcWinFormsKanban	TsgcWpfKanban	TsgcAvaloniaKanban	TsgcPlannerKanbanRenderView (KanbanView)
TsgcPlannerVCLView, TsgcPlannerFMXView	TsgcPlannerWinFormsView	TsgcPlannerWpfView	TsgcPlannerAvaloniaView	TsgcPlannerView (View)

Les classes de base dérivent de **TsgcUIWinFormsControl**, **TsgcUIWpfControl** et **TsgcUIAvaloniaControl** de sgcUICore.

Membres de chaque contrôle

Membre	Description
Data	Le **TsgcPlannerData** que le contrôle affiche. Plusieurs contrôles peuvent en partager un.
View	Le modèle de vue, un **TsgcPlannerView**. Les propriétés typées SchedulerView, GanttView et KanbanView renvoient le même objet. Sur le Scheduler, **View** est redéclarée comme le type de vue, comme dans Delphi.
Theme, Colors	L'apparence : le **TsgcUITheme** (utLight, utDark) et les **TsgcPlannerViewColors** qui remplacent les couleurs du thème.
ReadOnly, DragOptions	Désactivent l'édition, ou choisissent les glissements que l'utilisateur peut faire (créer, déplacer, redimensionner, progression, lier, réordonner), sous la forme `[Flags]` **TsgcPlannerDragOptions**.
SyncSelection	La sélection suit **SelectedId** des données, de sorte que les vues liées aux mêmes données sélectionnent le même élément.
ItemAt, ScrollToItem	L'élément en un point du contrôle (pixels client sur WinForms, DIP sur WPF et Avalonia) et le défilement d'un élément jusqu'à le rendre visible.

UpdateScroll-Bars, Cursor-Kind	Mettent à jour les barres de défilement après une modification faite sur le modèle, et le curseur que le contrôle affiche.
OnItemClick, OnItemDblClick, OnItemDelete, OnSelection-Change	Les événements de la vue de base de Delphi. OnItemDelete peut refuser la suppression.
SaveToPNG, SaveToSVG	Dessinent la vue telle qu'elle est maintenant dans un flux ou un fichier, à la taille du contrôle ou à une autre taille, avec une échelle.

Types indépendants du framework

Type	Description
TsgcPlannerData	Éléments, ressources, statuts, dépendances, le calendrier de travail, la notification des changements, JSON. Voir TsgcPlanner-Data.
TsgcPlannerItem, TsgcPlannerResource, TsgcPlannerStatus, TsgcPlannerDependency	Les éléments des collections, dans **TsgcPlannerItems**, **TsgcPlannerResources**, **TsgcPlannerStatuses** et **TsgcPlannerDependencies**.
TsgcPlannerCalendar, TsgcPlannerHolidays	Les jours ouvrés, les horaires de travail et les jours fériés. Voir Calendrier de travail.
TsgcPlannerTimeScale	Le zoom et l'accrochage du Gantt et de la Timeline. Voir Échelle de temps.
TsgcPlannerSchedulerView, TsgcPlannerGanttView, TsgcPlannerKanbanView	Les modèles de vue avec leurs sous-classes de rendu : mise en page, détection de clic, clavier, souris et glissement, et **ExportImage** pour SVG. Ils n'ont pas besoin de fenêtre.
TsgcPlannerScheduleEngine	Le moteur de planification (Registered). Voir Planification automatique.
TsgcPlannerSource_DataSet	Lie les données à des tables **System.Data.DataTable**. Voir Liaison de données dans .NET.
TsgcPlannerICalendar, TsgcPlannerMSProject	Import et export iCalendar et MS Project XML (Registered). Voir JSON, iCalendar et MS Project dans .NET.
TsgcPlannerRenderer	Dans esegece.sgcPlanner.Rendering : les trois vues en PNG et SVG sans fenêtre. Voir Rendu sans fenêtre dans .NET.
TsgcPlannerLocale	Les noms des jours et des mois, le format de l'heure et le séparateur décimal des libellés d'en-tête, repris de `CultureInfo.CurrentCulture`.
sgcPlanner_DateUtils, sgcPlanner_JSON, sgcPlanner_Raster	Des classes statiques avec les fonctions libres de Delphi : les conversions de temps et l'horloge, les fonctions JSON et les méthodes d'extension PNG des vues dans esegece.sgcPlanner.Rendering.

Énumérations

Les énumérations gardent les noms et les préfixes de valeur de Delphi, par exemple **TsgcPlannerSchedulerViewKind** (psv), **TsgcPlannerGanttField** (pgc), **TsgcPlannerBarColorMode** (pbc), **TsgcPlannerDurationUnit** (pdu), **TsgcPlannerDependencyKind** (pdk), **TsgcPlannerConstraint** (pc), **TsgcPlannerKanbanColumnSource** (pkc), **TsgcPlannerSwimlaneBy** (psl), **TsgcPlannerWIPMode** (pwm), **TsgcPlannerRecurrenceScope** (prs) et **TsgcPlannerRendererViewKind** (prv). Les ensembles Delphi sont des énumérations `[Flags]` combinées avec `|` : **TsgcPlannerWeekDays**, **TsgcPlannerChangeKinds**, **TsgcPlannerItemFields** et **TsgcPlannerDragOptions**. Le **TAlignment** de la RTL Delphi des colonnes du Gantt est déclaré dans esegece.sgcPlanner.

Voir aussi

- Vue d'ensemble de .NET
- Différences de .NET par rapport à Delphi
- Architecture

Différences de .NET par rapport à Delphi

Le portage .NET garde les noms, les valeurs par défaut, le comportement du clavier et de la souris et les règles de mise en page de Delphi. La sortie SVG des vues est identique octet pour octet à celle de Delphi, et la sortie PNG lui correspond dans la tolérance du rastériseur SkiaSharp. Les différences ci-dessous viennent des frameworks des hôtes et de la bibliothèque de classes .NET.

Temps et valeurs

- Les propriétés de date sont des valeurs `DateTime` de Kind Unspecified : **Start**, **Finish**, **ConstraintDate**, **BaselineStart**, **BaselineFinish**, **RecurrenceDate** et **ProjectStart**. Le TsgcPlannerTime de Delphi, en minutes entières, est un `long`, accessible par les propriétés au suffixe Time : **StartTime**, **FinishTime** et ainsi de suite. **Duration** est en minutes.
- L'horloge est `sgcPlanner_DateUtils.sgcPlannerNow`, un `Func<DateTime>`. Remplacez-la pour figer le temps dans les tests et restaurez ensuite le délégué enregistré.
- TsgcUIColor est un `uint`, $AARRGGBB, où 0 désigne la couleur du thème ou du statut.
- TFormatSettings et les globales de la RTL sont remplacés par `CultureInfo.CurrentCulture`, reprise par **TsgcPlannerLocale**. Les propriétés indexées des noms de jours et de mois sont les méthodes **GetShortDayName**, **SetShortDayName**, **GetLongMonthName** et ainsi de suite. Les fichiers sont toujours écrits avec des formats invariants.

Événements, méthodes et listes

- Les événements sont des propriétés délégué nommées comme dans Delphi (**OnChange**, **OnItemCreate**, **OnCardMoving**...). Affectez un gestionnaire avec `=`. Un paramètre `var` de Delphi est un paramètre `ref`, et une lambda avec un paramètre `ref` déclare le type de chaque paramètre. **TNotifyEvent** est **TsgcUINotifyEvent**.
- Les données acceptent plusieurs listeners : **AddListener** et **RemoveListener** prennent un **TsgcPlannerChangeEvent**, comme les vues.
- Une propriété indexée de Delphi est une méthode : `Rows(i)`, `ColumnInfo(i)`, `Cards(i)`, `Entries(i)`, `Calendar.Intervals(i)`. Une fonction Delphi sans paramètre reste une méthode, comme `RowCount()` ou `WorkingMinutesPerDay()`.
- Un tableau `var` de Delphi est un tableau `ref` que la méthode remplace par un tableau de la longueur exacte : `ChildrenOf(id, ref aList)`, `GetOccurrences(start, finish, ref aList)`.
- Un ensemble Delphi est une énumération `[Flags]` combinée avec `|`. Les collections gardent la forme de TCollection : Add, Delete, Clear, Assign, BeginUpdate, EndUpdate, un indexeur et `foreach`.

Pas de streaming de composants ni de notification de libération

- Il n'y a pas de DFM dans .NET. Les contrôles sont créés dans le code ou en XAML, et les données, les colonnes et les options sont définies dans le code.
- .NET n'a pas de notification de libération. Une vue garde ses données jusqu'à ce que **Data** soit mis à null ou que le contrôle soit libéré. Les contrôles WPF et Avalonia et les modèles de vue sont `IDisposable` : **Dispose** arrête leurs minuteurs et détache le listener des données. Avalonia ne libère pas les contrôles, libérez-les donc à la fermeture de la fenêtre.
- Les données n'ont pas d'état de chargement : les changements sont rassemblés entre **BeginUpdate** et **EndUpdate** et notifiés une fois.

Hôtes

- WinForms utilise les barres de défilement natives et le **BorderStyle** de WinForms ; WPF et Avalonia ont des enfants `ScrollBar` et pas de BorderStyle.
- WPF et Avalonia ont **IsTabStop** à la place de **TabStop**. Sur Avalonia la touche Commande compte comme Ctrl, et **Theme** masque le **StyledElement.Theme** d'Avalonia.

- Les propriétés des contrôles sont de simples propriétés CLR sur chaque hôte, pas des propriétés de dépendance ni stylées.
- Le toucher arrive comme une entrée de souris ; le glissement tactile de Delphi n'a pas de chemin de code séparé.
- Le curseur de séparation (**pcrSplit**) est un curseur de redimensionnement horizontal sur WPF et Avalonia, qui n'ont pas de curseur de séparation.

Sources de données et fichiers

- **TsgcPlannerSource_DataSet** lie des tables `System.Data.DataTable` à la place de TDataSet. Voir Liaison de données dans .NET pour les différences.
- MS Project XML est lu avec `System.Xml`. L'unité sgcPlanner_XML de Delphi n'est pas portée et n'est pas nécessaire.
- **ExportImage** d'un modèle de vue n'écrit que du SVG. Le PNG nécessite esegece.sgcPlanner.Rendering, voir Rendu sans fenêtre dans .NET.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- Données et planification dans .NET
- Liaison de données dans .NET

Données et planification dans .NET

TsgcPlannerData est l'unique source de vérité d'un plan. Il contient les éléments (tâches, événements et jalons), les ressources, les statuts, les dépendances et le calendrier de travail. Les vues n'écrivent jamais elles-mêmes un champ d'élément : chaque dépôt appelle les données dans un seul **BeginUpdate** et **EndUpdate**, et les données notifient chaque listener une fois, de sorte qu'un Gantt, un tableau et un Scheduler liés aux mêmes données sont toujours d'accord.

Construire un plan

L'exemple construit un petit projet avec trois statuts, deux personnes, un calendrier avec deux équipes et un jour férié, quatre tâches liées fin à début et un jalon, puis écoute les changements, déplace une carte comme le fait un dépôt Kanban et exécute le moteur de planification.

Notes

- **Finish** est exclusif, y compris pour les éléments sur la journée entière : un élément sur la journée entière le 1er octobre va du 1er octobre 00:00 au 2 octobre 00:00. Un élément dont **Start** est égal à **Finish** est un jalon.
- **Resources** et **Tags** sont des listes séparées par des virgules ; **ResourceAt**, **ResourceCount**, **TagAt** et **TagCount** les lisent une par une. **ParentId** construit l'arbre des tâches et **UpdateSummaries** recalcule les tâches récapitulatives.
- **MoveItem** change le statut et le rang d'une carte, **DeleteItem** supprime un élément et les éléments d'exception de sa série, et **SelectedId** est la sélection partagée par les vues.
- **OnItemChanging** peut refuser une modification avant qu'elle soit faite, pour les champs d'un ensemble **TsgcPlannerItemFields**.
- Le moteur place chaque tâche qui n'est pas **Manual** au plus tôt après ses liens et sa **Constraint**, sur le calendrier de travail, et calcule **TotalSlack** et **Critical**. Un cycle de liens déclenche **OnScheduleError** avec les ids du cycle. Un Gantt avec **AutoSchedule** l'exécute après chaque modification, **Schedule** du Gantt l'exécute à la demande, et **SetBaseline** enregistre les dates actuelles comme planning de référence. Le moteur est une fonctionnalité de Registered.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- JSON, iCalendar et MS Project dans .NET
- Données partagées
- Planification automatique
- Calendrier de travail

Liaison de données dans .NET

Le **TsgcPlannerSource_DataSet** de Delphi charge un plan depuis des tables TDataSet et réécrit les modifications. Dans .NET la même classe lie des tables `System.Data.DataTable`, les tables qu'un `DataAdapter` de n'importe quel fournisseur ADO.NET remplit. Les vues restent liées au **TsgcPlannerData**, de sorte que la même source alimente en même temps un Gantt, un tableau et un Scheduler.

Propriété	Description
Data	Le **TsgcPlannerData** que la source remplit.
DataSet, ResourceDataSet, DependencyDataSet	Les tables des éléments, des ressources et des liens. Seule la table des éléments est obligatoire.
ItemFields, ResourceFields, DependencyFields	Les noms des colonnes. Les valeurs par défaut sont les noms de propriété des éléments (Id, Title, Start, Finish, AllDay, Status, Progress, Color, ParentId, Tags, Resources, Rank, Notes, RecurrenceRule, DependsOn), des ressources (Id, Name, Color, Initials) et des liens (FromId, ToId, Kind, Lag). Une colonne qui n'existe pas n'est pas associée.
Active	Charge les tables dans les données et les garde synchronisées tant que la valeur est true.
LiveUpdates	Les lignes ajoutées, modifiées ou supprimées dans une table atteignent les données aussitôt (true par défaut).
WriteBack	Les modifications faites dans les vues ou dans le code sont réécrites dans les lignes (true par défaut).
OnLoadItem	Déclenché pour chaque élément chargé, avec la table et le `DataRow` , pour remplir les champs que l'association ne couvre pas.

Exemple

Différences avec TDataSet

- Une `DataTable` n'a pas de curseur. Un Edit et Post de Delphi est un `DataRow.BeginEdit` et `EndEdit`, ou une affectation directe à la colonne, et Cancel est `CancelEdit`. Une ligne en cours d'édition est réécrite à la fin de son édition.
- La source écoute les événements de la table : une ligne ajoutée ou modifiée est un enregistrement validé, une ligne supprimée ou une table vidée retire les éléments, une modification rejetée recharge l'enregistrement, et `AcceptChanges` est ignoré. Les lignes supprimées et détachées sont ignorées.
- Les colonnes inscriptibles sont celles qui ne sont ni ReadOnly, ni AutoIncrement, ni calculées. Une colonne de date numérique contient des dates OLE Automation, et un texte vide affecté à une colonne qui n'est pas de type texte définit `DBNull`.
- **OnLoadItem** transmet le `DataRow`, parce qu'il n'y a pas d'enregistrement courant. Les noms des colonnes sont **GetName** et **SetName** par index, en plus des propriétés nommées.
- Ne liez que des objets `DataTable` : une `DataView` ou un `IDataReader` n'est pas une source. Une table libérée se délie d'elle-même, et la source est `IDisposable`.

Lier les contrôles

Les propriétés des contrôles sont de simples propriétés CLR, pas des propriétés de dépendance ni stylées, parce qu'un plan est un modèle entier et non une valeur isolée. Dans une application MVVM, gardez le **TsgcPlannerData**

dans le modèle de vue et affectez-le à **Data** des contrôles, et réagissez à **OnChange** ou à un listener des données, à **SelectedId** et aux événements des contrôles.

Voir aussi

- Vue d'ensemble de .NET
- Données et planification dans .NET
- Différences de .NET par rapport à Delphi
- TsgcPlannerSource_DataSet
- Liaison DataSet

JSON, iCalendar et MS Project dans .NET

Un plan est enregistré et chargé dans trois formats, avec les mêmes résultats que dans Delphi : les fichiers qu'écrit une application .NET sont lus par une application Delphi et inversement.

Format	Classes	Édition
JSON, le format natif avec tous les champs	**SaveToJSON**, **LoadFromJSON**, **SaveToFile** et **LoadFromFile** de **TsgcPlannerData**, et les fonctions de flux et de fichier de **sgcPlanner_JSON**.	Les deux
iCalendar (RFC 5545), pour Outlook, Google Agenda et Apple Calendrier	**TsgcPlannerICalendar**	Registered
Microsoft Project XML (MSPDI) : tâches, liens, ressources, affectations et le calendrier	**TsgcPlannerMSProject**	Registered

Exemple

Notes

- JSON est écrit en UTF-8 sans marque d'ordre des octets ; une marque est acceptée au chargement. Un texte qui n'est pas de l'UTF-8 valide déclenche **EsgcPlannerError**.
- **TsgcPlannerICalendar** et **TsgcPlannerMSProject** sont de simples classes : créez-les, définissez **Data** et appelez **ImportFromString**, **ImportFromStream** ou **ImportFromFile**, et **ExportToString**, **ExportToStream** ou **ExportToFile**. **ImportMode** remplace les données ou y fusionne les éléments importés.
- Un problème dans l'entrée n'arrête pas un import : il déclenche **OnImportWarning** avec la ligne et un message, et **WarningCount** les compte. Les messages sont des constantes publiques des classes, avec des paramètres `{0}`.
- **ExportUTC** écrit les heures iCalendar en UTC. **ProductId** est le PRODID de l'export.
- MS Project XML est lu avec `System.Xml.XmlReader`, avec la DTD ignorée et sans résolveur externe, et écrit par le portage de l'écrivain Delphi, de sorte que l'export est identique octet pour octet à celui de Delphi.

Voir aussi

- Vue d'ensemble de .NET
- Données et planification dans .NET
- JSON
- iCalendar
- XML MS Project

Rendu sans fenêtre dans .NET

Chaque vue dessine par son modèle de vue, et les modèles n'ont pas besoin de fenêtre. Un service, un test ou un rapport peut charger un plan et enregistrer le Scheduler, le Gantt et le tableau en PNG ou SVG sur n'importe quelle plateforme, Linux compris.

- **TsgcPlannerRenderer** dans esegece.sgcPlanner.Rendering possède les trois vues. Définissez **Data**, **View** (**prvScheduler**, **prvGantt**, **prvKanban**), **Width**, **Height**, **Scale**, **Theme** et les options de la vue, puis appelez **SaveToPNG** ou **SaveToSVG**.
- **SchedulerOptions**, **GanttOptions** et **KanbanOptions** ont les propriétés des contrôles. **Assign** les copie depuis un modèle de vue, de sorte qu'un rapport peut dessiner ce qu'un contrôle affiche. Ce que les options ne couvrent pas se définit sur **SchedulerView**, **GanttView** et **KanbanView**.
- Un modèle de vue peut aussi être dessiné seul : **ExportImage** écrit du SVG avec esegece.sgcPlanner, et les méthodes d'extension **SaveToPNG**, **ExportPNG** et **RenderToRaster** de **sgcPlanner_Raster** écrivent du PNG.

Exemple

Notes

- La taille est en pixels logiques ; une échelle de 2 écrit une image de 2400 x 1000 pour une vue de 1200 x 500. **DeviceWidth** et **DeviceHeight** renvoient la taille de l'image.
- **FontFile** et **FontFileBold** choisissent un fichier de police TrueType, pour un serveur sans la police par défaut.
- Les libellés d'en-tête suivent le **TsgcPlannerLocale** partagé de `sgcPlanner_DateUtils.sgcPlannerLocale()`, repris de la culture actuelle à la première utilisation. Pour obtenir la même sortie sur chaque machine, définissez la culture avant le premier rendu, ou définissez les noms de ce locale.
- Les contrôles ont aussi **SaveToPNG** et **SaveToSVG**, à la taille du contrôle.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- TsgcPlannerRenderer
- Rendu sans fenêtre

Synchronisation avec Google Agenda et iCalendar en .NET

Paquet : esegece.sgcPlanner.sgcWebSockets (édition Registered), espace de noms esegece.sgcPlanner. Cible netstandard2.0, net8.0 et net9.0, et utilise la bibliothèque .NET de sgcWebSockets (esegece.sgcWebSockets) pour HTTP et OAuth2 de Google.
Le paquet .NET contient les mêmes deux composants que la passerelle Delphi, avec les mêmes noms, la même correspondance, la même règle de conflit et le même format de fichier d'état, de sorte qu'un fichier d'état écrit par Delphi se charge en .NET et inversement.

- **TsgcPlannerGoogleSync** maintient un TsgcPlannerData en synchronisation bidirectionnelle avec un Google Agenda via un TsgcHTTP_Google_Cloud_Calendar_API_Client (sa propriété Client). Voir TsgcPlannerGoogleSync.
- **TsgcPlannerICalSubscription** fusionne dans les données une URL iCalendar en lecture seule (https://, http:// ou webcal://), analysée avec TsgcPlannerICalendar. Voir TsgcPlannerICalSubscription.

Différences avec Delphi

- Les deux classes sont des classes simples qui implémentent IDisposable, pas des composants. Dispose arrête le thread de travail ; rien n'est appliqué ensuite.
- Les événements sont des propriétés délégués, par exemple OnRefreshed, OnError, OnImportWarning et OnBeforeRefresh (un EventHandler). OnConflict et OnFilterItem reçoivent leur résultat par ref.
- SyncAsync renvoie Task<bool>. RefreshAsync renvoie bool, False quand une actualisation est en cours.
- Le travail qui touche Data et déclenche les événements s'exécute sur le SynchronizationContext du thread qui a appelé SyncAsync ou RefreshAsync, ou défini Active (le thread d'interface dans WinForms, WPF et Avalonia). Sans contexte, il s'exécute dans le thread de travail.
- LastSyncTime et LastRefreshTime sont des DateTime ; default(DateTime) signifie jamais.
- La couture de transport de TsgcPlannerICalSubscription est la méthode virtuelle protégée DoFetch(string aURL, IList<string> aRequestHeaders, IList<string> aResponseHeaders, Stream aBody), qui renvoie le statut HTTP. L'implémentation par défaut utilise le HTTPClient interne (TsgcHTTP1Client) avec HandleRedirects désactivé ; le composant suit lui-même jusqu'à cinq redirections et envoie If-None-Match et If-Modified-Since avec les validateurs du dernier import.
- Il n'y a pas de notifications de libération : détruire Data ne vide pas la propriété Data.

Événements de flux en lecture seule

Une actualisation écrase les modifications locales des éléments d'abonnement. Opposez-leur un veto dans OnItemChanging, que les vues interrogent avant de valider un glisser, un redimensionnement ou une modification :

```
TsgcPlannerData oData = new TsgcPlannerData();
TsgcPlannerICalSubscription oSubscription = new TsgcPlannerICalSubscription();
oSubscription.Data = oData;
oSubscription.URL = "webcal://calendar.example.com/team.ics";
oSubscription.RefreshInterval = 60; // minutes
oSubscription.OnRefreshed = (s, aAdded, aUpdated, aDeleted) =>
  Console.WriteLine("added " + aAdded + ", updated " + aUpdated + ", deleted " + aDeleted);
oSubscription.OnError = (s, aError) => Console.WriteLine(aError);
oData.OnItemChanging = (object s, TsgcPlannerItem aItem, TsgcPlannerItemFields aFields, ref bool aAccept) =>
{
  if (oSubscription.IsSubscriptionItem(aItem))
    aAccept = false;
};
oSubscription.Active = true; // refreshes at once, then every hour
```

Synchronisation avec Google Agenda

```
TsgcHTTP_Google_Cloud_Calendar_API_Client oClient = new TsgcHTTP_Google_Cloud_Calendar_API_Client();
oClient.GoogleCloudOptions.OAuth2.ClientId = "your client id";
oClient.GoogleCloudOptions.OAuth2.ClientSecret = "your client secret";
oClient.GoogleCloudOptions.OAuth2.LocalPort = 8080;
oClient.GoogleCloudOptions.OAuth2.Scope.Add("https://www.googleapis.com/auth/calendar");

TsgcPlannerGoogleSync oSync = new TsgcPlannerGoogleSync();
oSync.Client = oClient;
oSync.Data = oData;
oSync.CalendarId = "primary";
oSync.StateFileName = "calendar_state.json";
oSync.OnSyncDone = (s, aPulled, aPushed, aDeleted, aConflicts) =>
  Console.WriteLine("pulled " + aPulled + ", pushed " + aPushed);
oSync.Active = true;
bool vOK = await oSync.SyncAsync();
```

Les éléments dont l'Id commence par ExcludeIdPrefix (ics: par défaut, l'IdPrefix de TsgcPlannerICalSubscription) ne sont jamais envoyés à Google.

Démo

La démo WinForms demos\Planner\GoogleCalendarSync montre les deux composants sur un TsgcWinFormsScheduler. Lancez-la avec --test pour un autotest hors ligne : un transport Google de test et un serveur iCalendar en boucle locale, aucun identifiant nécessaire.

Voir aussi

- TsgcPlannerGoogleSync
- TsgcPlannerICalSubscription
- Synchronisation avec Google Agenda
- TsgcPlannerICalendar
- TsgcPlannerData
- iCalendar

Démos .NET

Les démos se trouvent dans `.net\demos\Planner`, un dossier par démo avec son projet, son `Program.cs` et un README.md. Elles construisent leur interface utilisateur dans le code, sans XAML. Chaque démo a un commutateur `--test` qui exécute un autocontrôle hors écran, écrit des lignes PASS et FAIL et se termine avec le nombre de contrôles échoués, comme le commutateur `/test` des démos Delphi, et les démos à fenêtre ont un commutateur `--snapshot` qui rend les vues dans des fichiers PNG.

Démo	Hôte	Montre
ProjectPlanner	WinForms	Un projet de refonte de site web montré de trois façons à la fois : un diagramme de Gantt, un tableau Kanban et une semaine, tous liés aux mêmes données. Quatorze tâches sur deux pistes, cinq personnes, quatre statuts et des liens fin à début. Faites glisser une carte sur le tableau et le Gantt et le Scheduler suivent.
ProjectPlannerWpf	WPF	La même fenêtre avec les contrôles **TsgcWpf**.
ProjectPlannerAvalonia	Avalonia	La même fenêtre avec les contrôles **TsgcAvalonia**, pour Windows, Linux et macOS.
TeamScheduler	WinForms	Trois semaines de réunions d'équipe pour cinq personnes : jour, semaine de travail, semaine, mois, agenda et la Timeline des ressources, un point quotidien debout et une synchronisation hebdomadaire comme séries récurrentes, une conférence sur la journée entière, glisser pour créer avec **OnItemCreate** et regroupement par ressource.
TeamSchedulerWpf	WPF	La même fenêtre avec **TsgcWpfScheduler**.
TeamSchedulerAvalonia	Avalonia	La même fenêtre avec **TsgcAvaloniaScheduler**.
HeadlessRender	Console	Pas de fenêtre : charge un projet depuis JSON (ou en construit un et l'enregistre sous project.json) et écrit un PNG et un SVG du Scheduler, du Gantt et du tableau avec **TsgcPlannerRenderer**, sous Windows, Linux et macOS. L'équivalent .NET de la démo HeadlessReport de Delphi.
GoogleCalendarSync	WinForms	Une vue de la semaine maintenue en synchronisation bidirectionnelle avec un agenda Google, plus un abonnement iCalendar en lecture seule, avec les composants de la passerelle esegece.sgcPlanner.sgcWebSockets (Registered).

La partie sans interface de ProjectPlanner et de TeamScheduler (les données générées, la configuration des vues, le snapshot et l'autocontrôle) se trouve dans `ProjectPlanner_Shared.cs` et `TeamScheduler_Shared.cs`, que les projets WinForms, WPF et Avalonia compilent, de sorte que les trois hôtes exécutent les mêmes contrôles.
Chaque démo se compile et s'exécute dans les deux éditions. Dans l'édition Community, TeamScheduler n'a pas d'entrée Timeline et montre chaque série une seule fois, ProjectPlanner ne dessine pas de flèches de dépendance, et GoogleCalendarSync explique que la passerelle est un paquet Registered.

Compiler les démos

Dans l'arborescence du produit, les démos se compilent contre les projets de la bibliothèque. Le zip des démos utilise par défaut les paquets NuGet de l'édition Community. Compilez une démo avec les paquets Registered en définissant **SgcPlannerEdition** :

```
dotnet build ProjectPlanner\ProjectPlanner.csproj -p:SgcPlannerEdition=Registered
dotnet run --project HeadlessRender\HeadlessRender.csproj -- --test
```

Voir aussi

- Vue d'ensemble de .NET
- Installation depuis NuGet
- Démarrage rapide WinForms
- Démos

Dépannage

EsgcPlannerError: This feature requires the Registered edition of sgcPlanner

Schedule et SetBaseline du Gantt la lèvent dans un build Community. Placez ces appels dans `{$IFDEF SGC_PLANNER_ADVANCED}`, ou utilisez l'édition Registered ou Trial. Voir Fonctionnalités par édition.

Une fiche ne se charge pas dans un build Community : Class TsgcPlannerICalendar not found

TsgcPlannerICalendar et TsgcPlannerMSProject ne font pas partie de l'édition Community. Créez-les dans le code, dans `{$IFDEF SGC_PLANNER_ADVANCED}`, au lieu de les déposer sur une fiche qui doit aussi se charger dans Community.

Le Scheduler n'affiche aucun événement

- Vérifiez que les événements tombent dans la période affichée. Date à 0 suit l'horloge, donc la vue affiche la semaine courante ; définissez Date ou appelez GoToday, Next et Prior.
- Les vues par jour affichent les heures de Options.DayStartHour à Options.DayEndHour. Un événement en dehors est dessiné tronqué au bord.
- Un événement dont le Finish précède le Start est lu comme un événement de durée nulle à son Start.
- Assurez-vous que la propriété Data du Scheduler est l'objet de données que remplit votre code. Un contrôle sans Data utilise son propre objet de données interne.

Un événement récurrent n'apparaît qu'une seule fois

La récurrence fait partie de l'édition Registered. Dans Registered, vérifiez la règle avec TsgcPlannerRecurrence.TryParse, qui retourne la raison quand une règle n'est pas prise en charge, et vérifiez COUNT, UNTIL et RecurrenceExceptions. Voir Récurrence.

La planification automatique ne déplace pas une tâche

- Le moteur conserve les dates des tâches Manual, des éléments récurrents et des tâches d'un cycle de dépendances. OnScheduleError liste les tâches d'un cycle et Schedule retourne False.
- Une tâche avec la contrainte pcSNET ne commence pas avant sa ConstraintDate. Un déplacement dans le Gantt définit cette contrainte ; remettez Constraint à pcASAP pour laisser le moteur ramener la tâche plus tôt.
- AutoSchedule ne s'exécute qu'après une modification faite dans le Gantt. Après un changement dans le code ou dans une autre vue, appelez Schedule.

Voir Planification automatique.

Le Gantt ne dessine aucune flèche de dépendance

Les flèches requièrent l'édition Registered et ShowDependencies à True. Un lien n'est dessiné que lorsque ses deux tâches sont dans des lignes visibles, donc un lien vers un récapitulatif replié est masqué, et un lien dont le FromId ou le ToId est inconnu n'est jamais dessiné.

Un glissement ne fait rien ou l'élément revient en arrière

Vérifiez ReadOnly et DragOptions du contrôle, et les gestionnaires qui peuvent refuser une modification : OnItemChanging des données, OnCardMoving, OnItemCreate, OnLinkCreating et OnRecurrenceEdit. Sur le tableau, une colonne à sa limite WIP avec WIPMode pwmBlock refuse les nouvelles cartes, et la colonne inconnue n'est jamais une cible de dépôt. Dans le Scheduler, les éléments récapitulatifs sont affichés mais ne peuvent pas être déplacés. Voir Glisser-déposer.

EsgcPlannerError: OnChange kept changing the planner data for more than 8 rounds

Un gestionnaire de OnChange, ou un écouteur, modifie les données à chaque notification, de sorte que les changements ne se stabilisent jamais. Ne modifiez les données que lorsqu'elles diffèrent de la valeur voulue, ou sortez le changement du gestionnaire.

Les heures sont décalées de quelques heures après un import ou un export iCalendar

Les heures en UTC ou avec un fuseau horaire sont converties à l'heure locale de la machine à l'import. À l'export, ExportUTC écrit des heures UTC pour les éléments qui ne se répètent pas, qu'un calendrier affiche dans son propre fuseau horaire. Mettez ExportUTC à False pour écrire des heures locales. Voir iCalendar.

Les dates, les durées ou les noms semblent différents sur une autre machine ou sous Linux

Les libellés et les nombres de durée utilisent sgcPlannerLocale, initialisé à partir de la machine. Définissez ses propriétés, par exemple ShortDateFormat, Use24Hour et DecimalSeparator, au démarrage pour obtenir la même sortie partout.

Des caractères sont perdus dans Delphi 7 ou Delphi 2007

Ces versions ont des chaînes Ansi, donc le texte UTF-8 des fichiers JSON, iCalendar et XML MS Project est converti vers la page de codes du système, et les caractères hors de celle-ci sont perdus. Delphi 2009 et ultérieur conservent tous les caractères.

Les modifications ne sont pas réécrites dans le jeu de données

- Vérifiez que WriteBack est à True et que IdField désigne un champ que Locate peut trouver.
- Rien n'est écrit tant que le jeu de données est en état d'édition ou d'insertion ; les changements sont écrits après Post ou Cancel.
- Les ressources, les champs calculés, les champs de recherche et les champs en lecture seule ne sont jamais écrits.

Voir Liaison DataSet.

LoadFromJSON lève EsgcPlannerError

Le texte n'est pas du JSON valide, ce n'est pas un document sgcPlanner, ou il a été écrit avec une version plus récente du format. Les données restent telles qu'elles étaient, de sorte que l'application peut signaler le message et continuer à s'exécuter. Voir Persistance JSON.

Une violation d'accès après la libération d'une fiche ou d'un objet

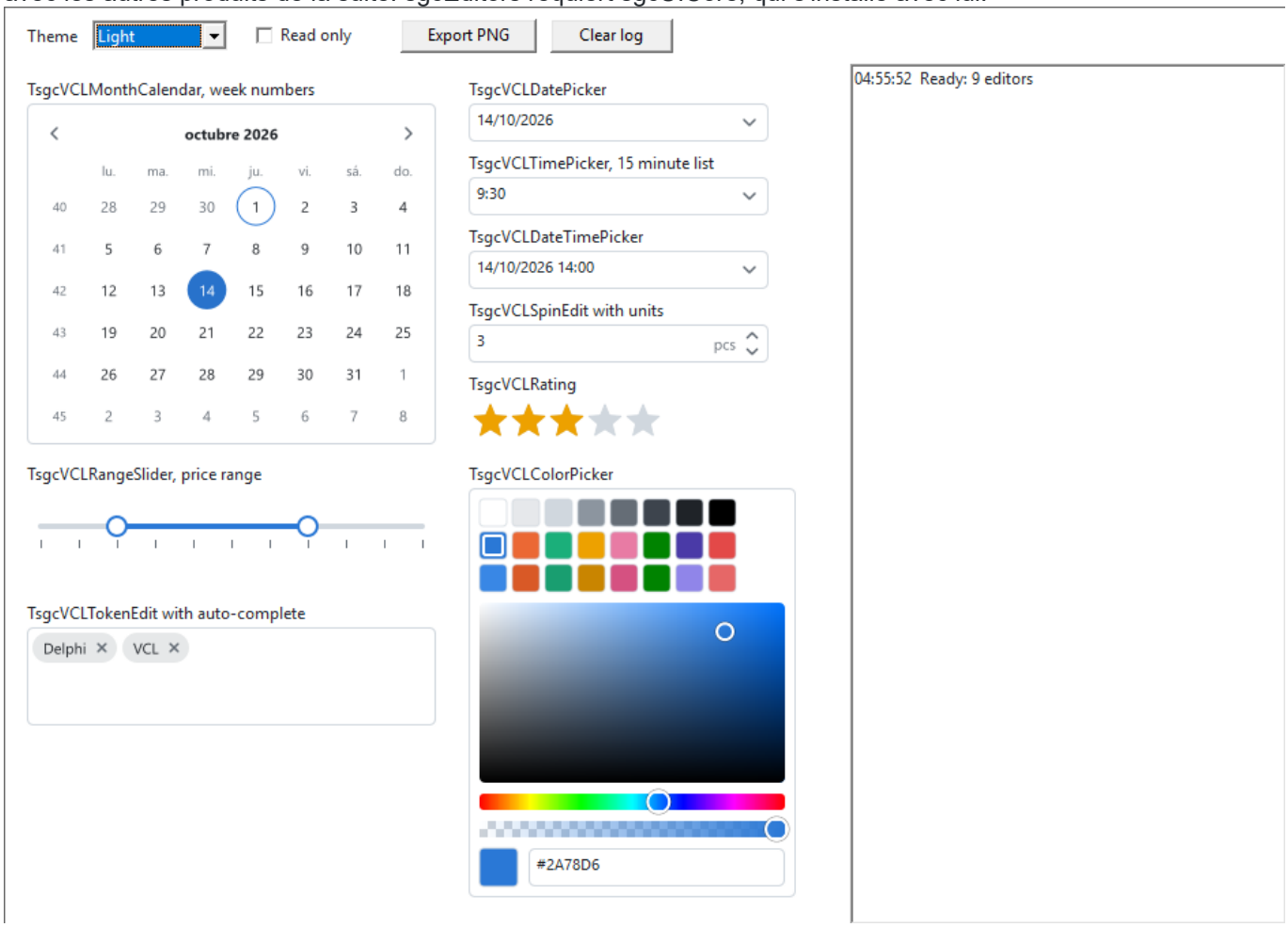
Un gestionnaire ajouté avec AddListener doit être retiré avec RemoveListener avant que l'objet qui le possède soit libéré. Les données ne savent pas quand cet objet disparaît.

Voir aussi

- Fonctionnalités par édition
- Plateformes
- TsgcPlannerData

Introduction

sgcEditors ajoute les contrôles de saisie d'un formulaire aux applications Delphi et C++Builder, pour VCL et FireMonkey : un calendrier mensuel, un sélecteur de date, un sélecteur d'heure, un sélecteur de date et d'heure, un curseur de plage, un sélecteur de couleur, un token edit, une évaluation et un spin edit. Chaque contrôle existe pour VCL, avec le préfixe TsgcVCL, et pour FireMonkey, avec le préfixe TsgcFMX. Cinq variantes data-aware lient le sélecteur de date, le sélecteur d'heure, le spin edit, l'évaluation et le curseur de plage à un champ d'un TDataSet.
Le texte est toujours saisi dans un edit natif, jamais dans un curseur personnalisé, de sorte que la composition IME, le presse-papiers, l'annulation et les lecteurs d'écran fonctionnent comme dans n'importe quel autre edit du formulaire. Tout le reste, le cadre, les boutons, le calendrier, les chips, les étoiles, les poignées et le carré de couleur, est dessiné par **sgcUICore**. Un contrôle a donc le même aspect sous VCL, sous FireMonkey et dans la sortie PNG et SVG sans interface. Les contrôles ont des thèmes clair et sombre, une couleur d'accentuation, une prise en charge complète du clavier, un nom accessible et une mise à l'échelle DPI par moniteur.
Derrière chaque contrôle se trouve une classe de modèle indépendante du framework, comme TsgcCalendarModel ou TsgcSpinModel. Les contrôles VCL et FireMonkey se contentent de relayer la saisie et d'héberger l'edit et la liste déroulante, de sorte que le même modèle s'affiche aussi sans fenêtre, sur un serveur ou sous Linux64.
sgcEditors fait partie de la suite visuelle eSeGeCe. Il est dessiné avec sgcUICore, le package d'exécution partagé qui contient le canvas, le rasteriseur, les polices, l'écrivain SVG et les thèmes, de sorte que les éditeurs s'accordent avec les autres produits de la suite. sgcEditors requiert sgcUICore, qui s'installe avec lui.



Éditions

sgcEditors est livré en versions Registered, Trial et Community. Community conserve les neuf contrôles pour VCL et FireMonkey et omet les cinq contrôles data-aware, le carré HSV et la barre alpha du sélecteur de couleur, la liste de saisie semi-automatique du token edit et la seconde poignée du curseur de plage. Voir Éditions.

Par où commencer

- Architecture : les modèles, les hôtes, l'edit interne, la liste déroulante et les deux couches de rendu.
- Contrôles : les neuf contrôles, leurs variantes data-aware et leurs classes de modèle en un coup d'oeil.
- Les pages des contrôles : Calendrier mensuel, Sélecteur de date, Sélecteur d'heure, Sélecteur de date et d'heure, Curseur de plage, Sélecteur de couleur, Token edit, Évaluation et Spin edit, les contrôles data-aware et les classes de modèle, avec leurs propriétés, méthodes, événements et exemples.
- Installation et Plateformes : le programme d'installation, les packages, les versions de Delphi et de C++Builder et les plateformes.
- Démarrage rapide VCL, Démarrage rapide FMX, Démarrage rapide C++Builder et Démarrage rapide data-aware : un premier formulaire en quelques minutes.
- Saisie de texte et IME, Clavier et accessibilité, Thèmes, Localisation, DPI, Sortie sans interface et Éditions.
- Dépannage : les problèmes courants et leurs causes.

Architecture

sgcEditors sépare la logique d'édition du framework. Chaque famille de contrôles a une classe de modèle qui n'utilise que la RTL et sgcUICore, et les contrôles VCL et FireMonkey sont de fines enveloppes autour d'elle.

Les parties d'un contrôle

1. **Modèle.** Un descendant TPersistent de TsgcEditorModel, par exemple TsgcCalendarModel, TsgcDatePickerModel ou TsgcTokenModel. Il contient la valeur et les options, reçoit les touches et la souris en pixels logiques par KeyDown, KeyPress, MouseDown, MouseMove, MouseUp et MouseWheel, et se dessine lui-même sur n'importe quel canvas sgcUICore. Il fonctionne sans fenêtre, c'est donc aussi le moteur de rendu sans interface.
2. **Hôte.** Le contrôle VCL descend de TsgcVCLCustomEditor et le contrôle FireMonkey de TsgcFMXCustomEditor. L'hôte crée le modèle, lui envoie la saisie, peint ses deux couches, déclenche OnChange et publie les propriétés du modèle. Le modèle d'un contrôle est public sous le nom Model, et chaque contrôle a aussi une propriété typée pour lui, comme CalendarModel, PickerModel ou SpinModel.
3. **Edit interne.** Les sélecteurs, le spin edit, le sélecteur de couleur et le token edit hébergent un edit natif sans bordure : un enfant TEdit en VCL et un TEdit FMX sous FireMonkey. Le modèle indique où va le texte, et l'hôte y place l'edit. Voir Saisie de texte et IME.
4. **Liste déroulante.** Le calendrier du sélecteur de date, la liste des heures du sélecteur d'heure, les deux côte à côte pour le sélecteur de date et d'heure, et la liste de saisie semi-automatique du token edit s'ouvrent dans une liste déroulante par framework. En VCL, c'est une fenêtre d'outils WS_POPUP qui ne prend jamais le focus de l'edit, s'ouvre sous le contrôle, bascule au-dessus lorsqu'il n'y a pas de place et reste dans la zone de travail du moniteur. Sous FireMonkey, c'est un TPopup placé sous le contrôle.

Deux couches

Un modèle dessine deux couches. Render dessine la couche de contenu : le cadre, le texte, les jours, les échantillons, les chips et les graduations. RenderOverlay dessine par-dessus ce qui change souvent : le survol, l'anneau de focus, les poignées du curseur, le curseur du carré de couleur, le remplissage de l'évaluation et son aperçu au survol. Un glissement, un survol ou un changement de focus ne redessine que la surcouche, de sorte que la couche de contenu n'est pas rastérisée de nouveau. Chaque changement d'un modèle est signalé par OnModelChange avec l'un de ces types :

Type	L'hôte
eckContent	Effectue de nouveau le rendu de la couche de contenu.
eckOverlay	Effectue le rendu de la surcouche seulement.
eckValue	Déclenche OnChange et, en VCL, informe les lecteurs d'écran de la nouvelle valeur.
eckLayout	Déplace l'edit interne, réécrit son texte et effectue le rendu de la couche de contenu.

Units

Unit	Contenu
sgcEditors_Types	Les énumérations, TsgcEditorModel, TsgcEditorsPainter (couleurs du thème, cadre, anneau de focus, chevrons, flèches de spin), codes de touches, formatage et analyse des nombres, conversion HSV et hexadécimale.

sgcEditors_Date	Numéros de jour, semaines ISO 8601 et simples, la grille du mois, TsgcEditorLocale et le format de date, le pas et l'analyse.
sgcEditors_Calendar, _Picker, _Slider, _Spin, _Color, _Rating, _Token, _List	Les modèles. sgcEditors_Picker contient aussi TsgcDateTimeDropDownModel, le calendrier et la liste des heures de la liste déroulante de date et d'heure.
sgcEditors_VCL, _VCL_Date, _VCL_Value, _VCL_Color, _VCL_Token	La base d'hôte VCL, l'edit interne, la liste déroulante et les contrôles VCL. Windows uniquement.
sgcEditors_FMX, _FMX_Date, _FMX_Value, _FMX_Color, _FMX_Token	La même chose pour FireMonkey, à partir de Delphi XE6.
sgcEditors_Data, _VCL_DB, _FMX_DB	TsgcEditorFieldLink et les contrôles data-aware. Édition Advanced, ces units sont vides dans Community.
sgcEditors_Const, _License, _Reg	Les constantes de version et d'édition, la vérification de l'évaluation et l'enregistrement à la conception.

Threads

Les contrôles sont utilisés depuis le thread principal, comme n'importe quel autre contrôle VCL ou FireMonkey, et leurs événements y sont déclenchés.

Performances

Le groupe de performances de la QA mesure ces scénarios avec un build de publication de Delphi 13 sous Win64. Le budget est le seuil de chaque scénario, et la colonne Mesuré est une exécution sur la machine de référence. Le calendrier mensuel a aussi un budget Win32 de 2 500 µs, et Linux64 rapporte E1, E4 et E8 sans seuil.

ID	Scénario	Budget	Mesuré
E1	Contenu du calendrier mensuel, 280 x 240 : titre, noms des jours, 42 jours, numéros de semaine et une bande de plage	1,500 µs	395 µs
E2	Surcouche du calendrier seulement : cellule survolée et anneau de focus	60 µs	24 µs
E3	IsDateEnabled sur 10 000 jours avec 500 dates désactivées, 2 jours de semaine désactivés et l'événement affecté	2 ms	0.3 ms
E4	Curseur de plage 300 x 48 avec 11 graduations et étiquettes, rendu complet ; un glissement de poignée, surcouche seulement	400 µs; 150 µs	88 µs; 73 µs
E5	Sélecteur de couleur 240 x 300 avec 24 échantillons, le carré, les barres de teinte et alpha, rendu complet ; un glissement dans le carré, surcouche seulement	1,500 µs; 150 µs	1,018 µs; 114 µs
E6	Token edit avec 50 chips, 400 x 120, disposition et rendu ; le filtre de saisie semi-automatique de 10 000 éléments par frappe	1,500 µs; 3 ms	436 µs; 0.2 ms
E7	Évaluation, 5 étoiles 160 x 32, rendu complet ; chrome de spin edit 120 x 28	200 µs; 120 µs	103 µs; 39 µs
E8	100 000 formatages de date plus 100 000 analyses ; 1 000 000 de numéros de semaine ISO	150 ms; 100 ms	23 ms; 27 ms

E9	Un formulaire VCL avec 30 de chacun des 9 contrôles, créé et peint pour la première fois ; une liste déroulante de sélecteur ouverte jusqu'à sa première image	350 ms; 16 ms	environ 340 ms sur une machine au repos ; 5 ms
E10	Le formulaire E9 sous FireMonkey	rapporté seule-ment	

La plus grande partie du temps E9 est la création des fenêtres de 270 contrôles et de leurs edits internes, pas le dessin. Un formulaire qui affiche des centaines d'éditeurs à la fois s'ouvre plus vite lorsque les contrôles des pages masquées sont créés à la demande.

Voir aussi

- Contrôles
- TsgcEditorModel
- Sortie sans interface
- Saisie de texte et IME

Contrôles

sgcEditors a neuf contrôles. Chacun existe pour VCL et pour FireMonkey avec les mêmes propriétés et événements publiés.

Contrôle	VCL	FireMonkey	Saisie de texte
Calendrier mensuel	TsgcVCLMonthCalendar	TsgcFMXMonthCalendar	Dessiné. Vues mois, année et décennie, date unique ou plage, limites, dates et jours de semaine désactivés, numéros de semaine.
Sélecteur de date	TsgcVCLDatePicker	TsgcFMXDatePicker	Edit natif plus un calendrier déroulant.
Sélecteur d'heure	TsgcVCLTimePicker	TsgcFMXTimePicker	Edit natif plus une liste d'heures déroulante toutes les ListInterval minutes.
Sélecteur de date et d'heure	TsgcVCLDateTimePicker	TsgcFMXDateTimePicker	Edit natif plus le calendrier et la liste des heures côte à côte.
Curseur de plage	TsgcVCLRangeSlider	TsgcFMXRangeSlider	Dessiné. Une ou deux poignées, pas, graduations, étiquettes et une info-bulle de valeur, horizontal ou vertical.
Sélecteur de couleur	TsgcVCLColorPicker	TsgcFMXColorPicker	Échantillons, carré HSV, barres de teinte et alpha dessinés, plus un edit hexadécimal natif.
Token edit	TsgcVCLTokenEdit	TsgcFMXTokenEdit	Chips dessinés, plus un edit natif pour le jeton suivant et une liste de saisie semi-automatique.
Évaluation	TsgcVCLRating	TsgcFMXRating	Dessiné. Étoiles, coeurs, cercles ou un glyphe de votre choix, valeurs entières, demi ou exactes.
Spin edit	TsgcVCLSpinEdit	TsgcFMXSpinEdit	Edit natif plus boutons haut et bas, unités, limites et répétition automatique.

Contrôles data-aware

Les contrôles data-aware descendent du contrôle de la deuxième colonne et ajoutent DataSource, DataField et un Field public. Ils font partie des éditions Registered et Trial. Voir Démarrage rapide data-aware.

VCL	FireMonkey	Contrôle de base	Lie
TsgcVCLDBDatePicker	TsgcFMXDBDatePicker	Sélecteur de date	La date du champ, sa partie heure est conservée.
TsgcVCLDBTimePicker	TsgcFMXDBTimePicker	Sélecteur d'heure	L'heure du champ, sa partie date est conservée.
TsgcVCLDBSpinEdit	TsgcFMXDBSpinEdit	Spin edit	Un champ numérique. Un champ entier fait passer le spin edit à esvInteger.
TsgcVCLDBRating	TsgcFMXDBRating	Évaluation	Un champ numérique.

TsgcVCLDBRangeSlider	TsgcFMXDBRangeSlider	Curseur de plage	Value à DataField et, en mode plage, EndValue à EndDataField.

Ce que chaque contrôle possède

Publiés : Theme, AccentColor, TextStyle, ReadOnly, AccessibleName et OnChange, plus les propriétés et événements habituels du framework, comme Align, Anchors, Enabled, Hint, TabOrder, TabStop, OnEnter, OnExit et les événements de clavier et de souris. Publics : Model, FormatSettings, DroppedDown, DropDown, CloseUp, SaveToPNG et SaveToSVG.

Classes de modèle

Chaque modèle descend de TsgcEditorModel.

Modèle	Utilisé par
TsgcCalendarModel	Le calendrier mensuel et la liste déroulante des sélecteurs de date.
TsgcDatePickerModel	Les sélecteurs de date, d'heure et de date et d'heure. Kind vaut edpDate, edpTime ou edpDateTime.
TsgcDateTimeDropDownModel	La liste déroulante du sélecteur de date et d'heure : un calendrier et une liste des heures.
TsgcEditorListModel	La liste des heures et la liste de saisie semi-automatique.
TsgcSliderModel	Le curseur de plage.
TsgcColorPickerModel	Le sélecteur de couleur.
TsgcTokenModel	Le token edit.
TsgcRatingModel	L'évaluation.
TsgcSpinModel	Le spin edit.

Voir aussi

- Architecture
- Démarrage rapide VCL

Installation

sgcEditors fait partie de la distribution sgcUI. Un seul programme d'installation couvre les produits sgcUI, et il n'installe que les produits que votre licence inclut.

Installer avec le programme d'installation sgcUI

1. Lancez le programme d'installation sgcUI et connectez-vous avec le nom d'utilisateur et le mot de passe de votre compte eSeGeCe.
2. Choisissez les versions de Delphi et de C++Builder dans lesquelles installer.
3. Sur la page des composants, sélectionnez **sgcEditors** et les autres produits souhaités. Le serveur de licences renvoie les produits que vous possédez, et le programme d'installation affiche une case à cocher par produit. Les produits que votre licence n'inclut pas sont grisés. Une licence All-Access débloque tous les produits.
4. Le programme d'installation copie les fichiers, puis compile les packages sur votre machine pour chaque IDE sélectionné, dans l'ordre des dépendances, et installe les packages de conception.

Ce qui accompagne sgcEditors

sgcUICore, la base partagée des produits visuels sgcUI, accompagne sgcEditors. Il n'a pas de case à cocher propre. Le programme d'installation l'installe automatiquement dès que sgcEditors est sélectionné.

Dossiers installés

Chaque produit s'installe dans un seul dossier, par défaut `Documents\esegece\sgcUI` (`sgcUI_Trial` pour l'évaluation et `sgcUI_Community` pour l'édition Community).

Dossier	Contenu
`Source`	Le code source de chaque produit installé. C'est l'unique chemin de bibliothèque de sgcUI.
`Packages`	Les packages de chaque produit installé, un jeu par produit et par version d'IDE.
`Lib<V>`	Les units compilées pour chaque version d'IDE et chaque plateforme, par exemple `LibD13` .
`Demos\Editors`	Les démos de sgcEditors : Gallery, GalleryFMX, Calendar, BookingForm et DataAware.

Le programme d'installation ajoute `sgcUI\Source` au chemin de bibliothèque et au chemin de navigation de chaque IDE et plateforme sélectionnés, avec le dossier des units compilées. sgcUICore se trouve dans le même dossier `Source`, il n'a donc pas besoin de chemin propre.

La palette de composants

Les composants se trouvent sur la page de palette **SGC Editors**. Un formulaire VCL affiche les contrôles VCL et un formulaire FireMonkey les contrôles FMX. Les éditions Registered et Trial enregistrent 28 composants, les neuf contrôles et les cinq contrôles data-aware pour chaque framework. Community enregistre les 18 contrôles sans les data-aware.

Trial et Community

- **Trial** : le programme d'installation d'évaluation de sgcUI installe chaque produit en mode d'évaluation, sgcEditors compris.
- **Community** : le programme d'installation gratuit sgcUI Community inclut sgcEditors sans ses fonctions avancées. Voir Éditions.

Compiler les packages manuellement

Pour compiler vous-même les packages, ouvrez-les pour votre version d'IDE depuis `sgcUI\Packages`. `<V>` est le suffixe de version de Delphi, par exemple `sgcEditorsD13.dpk`.

Package	Contenu
`sgcUICore<V>`	La base partagée sgcUICore (exécution).
`sgcEditors<V>` / `dclsgcEditors<V>`	Les composants sgcEditors (exécution / conception).

Compilez-les dans l'ordre des dépendances : `sgcUICore`, puis `sgcEditors`, puis `dclsgcEditors`. Installez ensuite le package de conception `dclsgcEditors` et ajoutez `sgcUI\Source` au chemin de bibliothèque de votre version de Delphi (Outils > Options > Langage > Options Delphi > Bibliothèque > Chemin de bibliothèque). Pour C++Builder, ouvrez le projet `.cbproj` correspondant à votre version, par exemple `sgcEditorsC13.cbproj`, et compilez-le. Les en-têtes `.hpp` générés apparaissent à côté du package compilé.

Voir aussi

- Plateformes
- Démarrage rapide VCL

Plateformes

Cible	Prise en charge
Delphi	Delphi 7, 2007, 2009, 2010, XE à XE8, 10 à 10.4, 11, 12 et 13.
C++Builder	C++Builder 2007, 2010, XE à XE8, 10 à 10.4, 11, 12 et 13. Les packages génèrent les en-têtes .hpp et les fichiers .lib qu'un projet C++Builder inclut et lie.
VCL	Les contrôles TsgcVCL sous Windows, Win32 et Win64, toutes les versions de Delphi.
FireMonkey	Les contrôles TsgcFMX à partir de Delphi XE6 (SGC_EDITORS_FMX), sur les plateformes FireMonkey. Ils n'utilisent que FireMonkey et sgcUICore.
Linux64	Les units indépendantes du framework à partir de Delphi 10.2 (SGC_EDITORS_LINUX) : les modèles, le calcul de dates et les paramètres régionaux, avec la sortie PNG et SVG sans interface. Il n'y a pas de contrôles Linux.
.NET	Un portage C# avec la même API pour WinForms, WPF et Avalonia, les neuf contrôles sur chaque hôte et la sortie PNG et SVG sans interface, sur .NET Framework 4.7.2 et .NET 8 et 9, voir Vue d'ensemble de .NET.

Packages

sgcEditors fournit un package d'exécution indépendant du framework et un package d'exécution par framework, de sorte qu'une application FireMonkey pour Android, iOS ou macOS ne lie jamais la VCL. Chaque nom de package se termine par le suffixe de l'IDE, par exemple sgcEditorsD13.dpk pour Delphi 13 et sgcEditorsC13.cbproj pour C++Builder 13.

Package	Type	Contenu	Requiert
sgcEditors	Exécution, indépendant du framework	Chaque unité du produit qui n'utilise ni la VCL ni FireMonkey.	rtl, dbrtl, sgcUICore
sgcEditorsVCL	Exécution, VCL	sgcEditors_VCL, sgcEditors_VCL_Color, sgcEditors_VCL_DB, sgcEditors_VCL_Date, sgcEditors_VCL_Token et sgcEditors_VCL_Value	rtl, vcl, dbrtl, sgcUICore, sgcUICoreVCL, sgcEditors
sgcEditorsFMX	Exécution, FireMonkey, à partir de XE6	sgcEditors_FMX, sgcEditors_FMX_Color, sgcEditors_FMX_DB, sgcEditors_FMX_Date, sgcEditors_FMX_Token et sgcEditors_FMX_Value	rtl, fmx, dbrtl, sgcUICore, sgcUICoreFMX, sgcEditors
dclsgcEditors	Conception, VCL	Enregistre les éditeurs VCL dans la page de palette **SGC Editors**, avec l'éditeur de liste de chaînes standard pour DisabledDates et Swatches.	designide, rtl, vcl, sgcEditors, sgcEditorsVCL
dclsgcEditorsFMX	Conception, FireMonkey, à partir de XE6	Enregistre les éditeurs FireMonkey dans la page de palette **SGC Editors**.	designide, rtl, fmx, sgcEditors, sgcEditorsFMX, dclsgcEditors

sgcUICore suit la même répartition : sgcUICore, sgcUICoreVCL et sgcUICoreFMX, sans package de conception. dclsgcEditorsFMX requiert dclsgcEditors, installez donc d'abord dclsgcEditors.

Versions de l'IDE et plateformes cibles

Delphi	sgcEditors	sgcEditorsVCL	sgcEditorsFMX	dclsgcEditors, dclsgcEditorsFMX
7, 2007, 2009, 2010, XE	Win32	Win32	Aucune	Win32
XE2 à XE5	Win32, Win64	Win32, Win64	Aucune	Win32
XE6 à 10.2	Win32, Win64	Win32, Win64	Win32, Win64	Win32
10.3	Win32, Win64, Linux64, OSX64	Win32, Win64	Win32, Win64, OSX64	Win32
10.4	Win32, Win64, Linux64, OSX64, Android, Android64, iOSDevice64	Win32, Win64	Win32, Win64, OSX64, Android, Android64, iOSDevice64	Win32
11	Win32, Win64, Linux64, OSX64, OSXARM64, Android, Android64, iOSDevice64, iOSSimARM64	Win32, Win64	Win32, Win64, OSX64, OSXARM64, Android, Android64, iOSDevice64, iOSSimARM64	Win32
12, 13	Comme 11	Win32, Win64	Comme 11	Win32, Win64

De XE6 à 10.2, le package FireMonkey est compilé pour Windows uniquement. Les packages de conception sont chargés par l'IDE, ils sont donc compilés pour la plateforme de l'IDE uniquement.

C++Builder	sgcEditors, sgcEditorsVCL	sgcEditorsFMX	dclsgcEditors, dclsgcEditorsFMX
2007, 2010, XE, XE2	Win32	Aucune	Win32 (dclsgcEditors uniquement)
XE3 à XE5	Win32, Win64	Aucune	Win32 (dclsgcEditors uniquement)
XE6 à 12	Win32, Win64	Win32, Win64	Win32
13	Win32, Win64, Win64x	Win32, Win64, Win64x	Win32, Win64x

Les projets C++Builder portent les mêmes noms avec le suffixe C, par exemple sgcEditorsC13.cbproj, sgcEditorsVCLC13.cbproj, sgcEditorsFMXC13.cbproj, dclsgcEditorsC13.cbproj et dclsgcEditorsFMXC13.cbproj. Les packages C++Builder sont compilés pour Windows uniquement. Les notes C++Builder qui s'appliquent à chaque produit sgcUI (packages d'exécution sous Win64, C++Builder 10.3 sous Win64, noms de fichiers des packages Win64x) se trouvent dans le sujet Plateformes et packages de l'aide de la suite sgcUI.

FireMonkey sur Android, iOS et macOS

Les contrôles FireMonkey (les éditeurs TsgcFMX) ciblent macOS à partir de Delphi 10.3, Android et iOS à partir de Delphi 10.4, et macOS ARM et le simulateur iOS à partir de Delphi 11. Dans un projet FireMonkey pour ces plateformes :

- Utilisez les contrôles FireMonkey. Les contrôles VCL et sgcEditorsVCL ne fonctionnent que sous Windows.

- Déposez le contrôle sur un formulaire, ou ajoutez son unité à la clause uses : sgcEditors_FMX, sgcEditors_FMX_Color, sgcEditors_FMX_DB, sgcEditors_FMX_Date, sgcEditors_FMX_Token et sgcEditors_FMX_Value.
- Sélectionnez la plateforme cible dans la fenêtre Projets. L'installateur ajoute le dossier Source de sgcUI et les unités compilées de chaque plateforme (LibD<version>\<platform>) au chemin de bibliothèque de chaque plateforme qu'il compile. Sans l'installateur, ajoutez le dossier Source de sgcUI au chemin de bibliothèque de la plateforme cible (Outils, Options, Langage, Delphi, Bibliothèque).
- Les applications Android et iOS sont liées statiquement, aucun package d'exécution n'est donc déployé.

Linux64

Le package indépendant du framework sgcEditors est compilé pour Linux64 à partir de Delphi 10.3. Une application Linux64, par exemple un programme console ou un serveur, utilise les modèles d'éditeur (valeurs, dates, couleurs, jetons). Aucun package VCL ou FireMonkey n'est compilé pour Linux64, car FireMonkey sous Linux nécessite FMX pour Linux (FMXLinux), un module complémentaire tiers.

Zone de texte interne des éditeurs VCL

Les éditeurs VCL dotés d'un champ de texte créent leur zone de texte interne (une fenêtre d'édition native) à la demande, ce qui permet d'afficher rapidement les formulaires comportant de nombreux éditeurs. Jusque-là, l'éditeur dessine le texte lui-même. La zone de texte interne est créée au premier survol de la souris, lorsque l'éditeur reçoit le focus, à la première utilisation du clavier ou du presse-papiers, ou lorsque le code lit son handle de fenêtre. Les outils d'accessibilité, tels que les lecteurs d'écran et les clients UI Automation, ne voient la zone de texte interne qu'à partir de ce moment. L'éditeur lui-même répond à WM_GETTEXT dès le début. Un éditeur désactivé ou ReadOnly, ou doté de PasswordChar, d'une lecture de droite à gauche, de TextHint ou d'un Alignment autre que gauche, la crée immédiatement, et à la conception elle est toujours créée.

Contraintes de version

- Les units indépendantes du framework et les units VCL compilent sur Delphi 7 : pas de génériques, pas de méthodes anonymes, pas de variables inline et pas de record helpers.
- TextHint des contrôles VCL s'affiche à partir de Delphi 2009.
- Les énumérations ont des préfixes d'éditeur (esm, efd, ewn, ecv, eor, etp, ecp, eti, emm, erp, erg, esv, esb), de sorte que les en-têtes C++Builder n'entrent jamais en conflit avec les autres produits sgcUI.

Voir aussi

- Éditions
- Architecture

Démarrage rapide VCL

Installez les packages sgcEditors pour votre version de Delphi. Les composants se trouvent sur la page de palette **SGC Editors**.

À la conception

1. Déposez un **TsgcVCLDatePicker** sur un formulaire. Définissez MinDate et Date dans l'inspecteur d'objets, ou laissez Date vide avec AllowEmpty à True.
2. Déposez un **TsgcVCLSpinEdit**, définissez MinValue, MaxValue et Value, et saisissez un suffixe comme invités dans Units.
3. Déposez un **TsgcVCLTokenEdit** et donnez-lui un TextHint.
4. Affectez un même gestionnaire OnChange aux trois contrôles et lancez l'application. Saisissez une date ou ouvrez le calendrier avec F4 ou Alt+Bas, faites avancer le spin edit avec les flèches ou la molette, et saisissez des jetons séparés par des virgules.

Dans le code

Un contrôle fonctionne dès qu'il a un Parent. Définissez les limites d'un spin edit avant sa Value, afin que la valeur ne soit pas bornée aux anciennes limites.

```
uses
  SysUtils, Classes, Controls, Forms, StdCtrls,
  sgcEditors_VCL_Date, sgcEditors_VCL_Value, sgcEditors_VCL_Token;

type
  TBooking = class(TComponent)
  public
    Arrival: TsgcVCLDatePicker;
    Guests: TsgcVCLSpinEdit;
    Tags: TsgcVCLTokenEdit;
    Summary: TLabel;
    procedure EditorChange(Sender: TObject);
  end;

procedure TBooking.EditorChange(Sender: TObject);
begin
  // OnChange fires for every new value, typed, picked or stepped
  if Arrival.IsEmpty then
    Summary.Caption := 'No arrival date'
  else
    Summary.Caption := Format('%s, %d guests, %d tags',
      [DateToStr(Arrival.Date), Round(Guests.Value), Tags.Tokens.Count]);
end;

function CreateBooking(aForm: TForm): TBooking;
begin
  Result := TBooking.Create(aForm);

  Result.Arrival := TsgcVCLDatePicker.Create(aForm);
  Result.Arrival.Parent := aForm;
  Result.Arrival.SetBounds(16, 16, 160, 28);
  Result.Arrival.MinDate := Date;
  Result.Arrival.Date := Date + 1;
  Result.Arrival.OnChange := Result.EditorChange;

  // the limits before the value, MinValue = MaxValue means no limit
  Result.Guests := TsgcVCLSpinEdit.Create(aForm);
  Result.Guests.Parent := aForm;
  Result.Guests.SetBounds(16, 56, 120, 28);
```

```
  Result.Guests.MinValue := 1;
  Result.Guests.MaxValue := 12;
  Result.Guests.Value := 2;
  Result.Guests.Units := 'guests';
  Result.Guests.OnChange := Result.EditorChange;

  Result.Tags := TsgcVCLTokenEdit.Create(aForm);
  Result.Tags.Parent := aForm;
  Result.Tags.SetBounds(16, 96, 320, 64);
  Result.Tags.TextHint := 'Add a tag';
  Result.Tags.AddToken('Late arrival');
  Result.Tags.OnChange := Result.EditorChange;

  Result.Summary := TLabel.Create(aForm);
  Result.Summary.Parent := aForm;
  Result.Summary.SetBounds(16, 172, 320, 20);
  Result.EditorChange(nil);
end;
```

Voir aussi

- Contrôles
- Saisie de texte et IME
- Clavier et accessibilité

Démarrage rapide FMX

Les contrôles FireMonkey requièrent Delphi XE6 ou une version ultérieure. Ils se trouvent sur la page de palette **SGC Editors** d'un formulaire FireMonkey, et ont les mêmes propriétés et événements que les contrôles VCL.

À la conception

1. Déposez un **TsgcFMXDatePicker**, un **TsgcFMXRangeSlider** et un **TsgcFMXColorPicker** sur un formulaire.
2. Sur le curseur, définissez RangeMode à True, puis Min, Max, Step et TickInterval, et activez ShowLabels.
3. Sur le sélecteur de couleur, choisissez les Parts à afficher et la SelectedColor.

Ce qui est différent sous FireMonkey

- La liste déroulante est un TPopup sous le contrôle. Lorsqu'elle est fermée par un clic à l'extérieur, les choix faits pendant qu'elle était ouverte sont annulés. Entrée, ou le choix qui complète la valeur, la ferme et garde la valeur.
- FireMonkey déplace le focus sur Tab avant qu'un contrôle voie la touche, donc le curseur de plage et le sélecteur de couleur utilisent Ctrl avec les flèches pour passer d'une partie à l'autre. Voir Clavier et accessibilité.
- Le texte accessible d'un contrôle va dans son Hint, en conservant un Hint de votre cru. TabStop, Hint et ShowHint sont publiés à partir de Delphi 10.1.
- L'edit interne du spin edit demande le clavier virtuel NumbersAndPunctuation sur les plateformes mobiles.

Dans le code

```
uses
  SysUtils, Classes, FMX.Types, FMX.Controls, FMX.Forms, FMX.StdCtrls,
  sgcEditors_Types, sgcEditors_FMX_Date, sgcEditors_FMX_Value,
  sgcEditors_FMX_Color;

type
  TFilters = class(TComponent)
  public
    Day: TsgcFMXDatePicker;
    Price: TsgcFMXRangeSlider;
    Colour: TsgcFMXColorPicker;
    Info: TLabel;
    procedure PriceComplete(Sender: TObject);
  end;

procedure TFilters.PriceComplete(Sender: TObject);
begin
  // OnChangeComplete fires once, when the drag or the key ends
  Info.Text := Format('%.0f to %.0f', [Price.Value, Price.EndValue]);
end;

function CreateFilters(aForm: TForm): TFilters;
begin
  Result := TFilters.Create(aForm);

  Result.Day := TsgcFMXDatePicker.Create(aForm);
  Result.Day.Parent := aForm;
  Result.Day.SetBounds(16, 16, 180, 32);
  Result.Day.Date := Date;

  // two thumbs, steps of 10, a tick and a label every 100
  Result.Price := TsgcFMXRangeSlider.Create(aForm);
```

```
  Result.Price.Parent := aForm;
  Result.Price.SetBounds(16, 64, 320, 56);
  Result.Price.Min := 0;
  Result.Price.Max := 500;
  Result.Price.RangeMode := True;
  Result.Price.Step := 10;
  Result.Price.TickInterval := 100;
  Result.Price.ShowLabels := True;
  Result.Price.Value := 100;
  Result.Price.EndValue := 300;
  Result.Price.OnChangeComplete := Result.PriceComplete;

  Result.Colour := TsgcFMXColorPicker.Create(aForm);
  Result.Colour.Parent := aForm;
  Result.Colour.SetBounds(16, 136, 240, 120);
  Result.Colour.Parts := [ecpSwatches, ecpHex];
  Result.Colour.SelectedColor := $FF2A78D6;

  Result.Info := TLabel.Create(aForm);
  Result.Info.Parent := aForm;
  Result.Info.SetBounds(16, 272, 320, 20);
  Result.PriceComplete(nil);
end;
```

Voir aussi

- Contrôles
- Plateformes
- DPI

Démarrage rapide C++Builder

Installez les packages sgcEditors pour votre version de C++Builder, ajoutez `sgcUI\Source` et le dossier des units compilées, par exemple `sgcUI\LibD13\win32` (`sgcUI\LibD13\win64` pour le 64 bits), aux chemins d'inclusion et de bibliothèque du projet, et utilisez les composants de la page de palette **SGC Editors** exactement comme dans Delphi. Le programme d'installation ajoute les deux chemins à l'IDE pour vous. Les en-têtes .hpp et les fichiers .lib générés de sgcEditors et de sgcUICore se trouvent dans ce seul dossier.

À savoir en C++

- Incluez l'en-tête de l'unit qui déclare un contrôle : `sgcEditors_VCL_Date.hpp` pour le calendrier et les sélecteurs, `sgcEditors_VCL_Value.hpp` pour le curseur, l'évaluation et le spin edit, `sgcEditors_VCL_Color.hpp`, `sgcEditors_VCL_Token.hpp` et `sgcEditors_VCL_DB.hpp`. Les énumérations sont dans `sgcEditors_Types.hpp`.
- Les événements sont des closures, leurs gestionnaires sont donc des fonctions membres __fastcall d'une classe. Un paramètre var comme aAccept est une référence bool.
- Date, MinDate et MaxDate sont des propriétés TDate. Affectez-leur un TDateTime, par exemple `Picker->Date = Date() + 1`.
- Les classes de modèle ont des constructeurs sans argument, comme `new TsgcCalendarModel()`, et vous les supprimez vous-même. Voir Sortie sans interface.

Dans le code

Voir aussi

- Démarrage rapide VCL
- Démarrage rapide data-aware
- Plateformes

Démarrage rapide data-aware

Les contrôles data-aware lient un sélecteur de date, un sélecteur d'heure, un spin edit, une évaluation ou un curseur de plage à un champ, comme TDBEdit. Ils font partie des éditions Registered et Trial, pour VCL dans sgcEditors_VCL_DB et pour FireMonkey dans sgcEditors_FMX_DB. FireMonkey n'a pas de TFieldDataLink, donc les deux frameworks partagent TsgcEditorFieldLink, un TDataLink de sgcEditors_Data.

À la conception

1. Déposez un dataset et un **TDataSource** sur un formulaire.
2. Déposez un **TsgcVCLDBDatePicker** (ou son jumeau FMX), définissez DataSource sur la source de données et choisissez le champ dans DataField.
3. Pour un **TsgcVCLDBRangeSlider** en mode plage, définissez DataField pour la poignée basse et EndDataField pour la poignée haute.

Comment les contrôles suivent le champ

- La valeur suit l'enregistrement courant. Un champ null affiche un texte vide dans les sélecteurs et le spin edit, aucun glyphe rempli dans l'évaluation, et la poignée à Min dans le curseur, avec la poignée de fin à Max. AllowEmpty vaut True par défaut sur les sélecteurs data-aware et le spin edit.
- Une modification par l'utilisateur met le dataset en mode édition et marque le lien comme modifié. Définir Value ou Date dans le code compte aussi comme une modification par l'utilisateur. Lorsque le dataset ne peut pas passer en mode édition, la valeur du champ revient.
- Entrée et la perte du focus valident le texte saisi et écrivent le champ. Lorsque le champ refuse la valeur, il lève une exception et le focus reste dans le contrôle, comme avec TDBEdit. Avant un post, le dataset écrit une valeur en attente. Échap restaure la valeur du champ.
- Le sélecteur de date conserve la partie heure d'un champ de date et d'heure, et le sélecteur d'heure conserve sa partie date. Un champ entier fait passer le spin edit à esvInteger.
- Un contrôle est en lecture seule tant que son ReadOnly vaut True, tant qu'il n'y a pas de dataset, tant que le dataset ou le champ est en lecture seule, pour un champ calculé et, pour le curseur en mode plage, tant que le champ de fin ne peut pas être modifié.

Dans le code

L'exemple crée un TClientDataSet et les cinq contrôles data-aware VCL. Ajoutez MidasLib à la clause uses du projet afin que l'application n'ait pas besoin de midas.dll.

```
uses
  SysUtils, Classes, Controls, Forms, DB, DBClient,
  sgcEditors_VCL_DB;

function CreateBookings(aOwner: TComponent): TClientDataSet;
begin
  Result := TClientDataSet.Create(aOwner);
  Result.FieldDefs.Add('Guest', ftString, 40);
  Result.FieldDefs.Add('CheckIn', ftDate);
  Result.FieldDefs.Add('Arrival', ftTime);
  Result.FieldDefs.Add('Nights', ftInteger);
  Result.FieldDefs.Add('Score', ftFloat);
  Result.FieldDefs.Add('BudgetFrom', ftFloat);
  Result.FieldDefs.Add('BudgetTo', ftFloat);
  Result.CreateDataSet;
  Result.AppendRecord(['Ana', EncodeDate(2026, 10, 12),
    EncodeTime(15, 30, 0, 0), 3, 4, 200, 450]);
end;

procedure CreateDataEditors(aForm: TForm; aSource: TDataSource);
var
```

```
  oCheckIn: TsgcVCLDBDatePicker;
  oArrival: TsgcVCLDBTimePicker;
  oNights: TsgcVCLDBSpinEdit;
  oScore: TsgcVCLDBRating;
  oBudget: TsgcVCLDBRangeSlider;
begin
  oCheckIn := TsgcVCLDBDatePicker.Create(aForm);
  oCheckIn.Parent := aForm;
  oCheckIn.SetBounds(16, 16, 160, 28);
  oCheckIn.DataSource := aSource;
  oCheckIn.DataField := 'CheckIn';

  oArrival := TsgcVCLDBTimePicker.Create(aForm);
  oArrival.Parent := aForm;
  oArrival.SetBounds(192, 16, 120, 28);
  oArrival.ListInterval := 15;
  oArrival.DataSource := aSource;
  oArrival.DataField := 'Arrival';

  // an integer field switches the spin edit to esvInteger
  oNights := TsgcVCLDBSpinEdit.Create(aForm);
  oNights.Parent := aForm;
  oNights.SetBounds(16, 56, 120, 28);
  oNights.MinValue := 1;
  oNights.MaxValue := 30;
  oNights.Units := 'nights';
  oNights.DataSource := aSource;
  oNights.DataField := 'Nights';

  oScore := TsgcVCLDBRating.Create(aForm);
  oScore.Parent := aForm;
  oScore.SetBounds(16, 96, 160, 32);
  oScore.DataSource := aSource;
  oScore.DataField := 'Score';

  // in RangeMode, Value is DataField and EndValue is EndDataField
  oBudget := TsgcVCLDBRangeSlider.Create(aForm);
  oBudget.Parent := aForm;
  oBudget.SetBounds(16, 140, 320, 48);
  oBudget.Min := 0;
  oBudget.Max := 1000;
  oBudget.Step := 50;
  oBudget.RangeMode := True;
  oBudget.DataSource := aSource;
  oBudget.DataField := 'BudgetFrom';
  oBudget.EndDataField := 'BudgetTo';
end;
```

Voir aussi

- Contrôles
- Sélecteur de date DB
- TsgcEditorFieldLink
- Éditions
- Dépannage

Calendrier mensuel

Classes : TsgcVCLMonthCalendar (unit sgcEditors_VCL_Date) et TsgcFMXMonthCalendar (unit sgcEditors_FMX_Date)
Le calendrier mensuel est un contrôle dessiné sur mesure, pouvant recevoir le focus, sans edit. Il affiche un mois avec les noms des jours et, en option, les numéros de semaine, et il dézoome vers les 12 mois d'une année et vers les années d'une décennie lorsque l'utilisateur clique sur le titre ou appuie sur Ctrl+Haut. Il sélectionne une date ou, avec SelectionMode esmRange, une plage de jours. Les jours hors de MinDate et MaxDate, les DisabledDates, les DisabledWeekDays et les jours que OnIsDateDisabled désactive sont dessinés barrés et ne peuvent pas être sélectionnés. Le même modèle, TsgcCalendarModel, dessine la liste déroulante des sélecteurs de date.
La classe VCL descend de TsgcVCLCustomEditor et la classe FireMonkey de TsgcFMXCustomEditor. Les deux publient les propriétés et les événements du tableau ci-dessous, plus les habituels de leur framework, comme Align, Anchors, Enabled, Hint, TabOrder, TabStop, OnEnter, OnExit et les événements de clavier et de souris. Voir Contrôles.

Propriétés

Propriété	Type	Description
Date	TDate	La date sélectionnée en mode esmSingle, 0 lorsqu'aucune date n'est sélectionnée. Le mode plage n'y touche pas.
SelectionMode	TsgcEditorSelectionMode	esmSingle (par défaut) sélectionne une date. esmRange sélectionne une plage : le premier clic la commence, le second clic la termine, et Maj avec un clic ou avec les flèches l'étend à partir du premier jour.
RangeStart, RangeEnd	TDate	Le premier et le dernier jour de la plage, tous deux inclus.
MaxRangeDays	Integer	Le plus grand nombre de jours qu'une plage peut contenir, les deux extrémités incluses. 0 (par défaut) signifie aucune limite.
MinDate, MaxDate	TDate	Le premier et le dernier jour que l'utilisateur peut choisir, comparés comme jours entiers. 0 (par défaut) signifie aucune limite, et des limites inversées sont échangées.
DisabledDates	TStrings	Les jours qui ne peuvent pas être choisis, une date ISO (yyyy-mm-dd) par ligne.
DisabledWeekDays	TsgcEditorWeekDays	Les jours de la semaine qui ne peuvent pas être choisis, un ensemble de ewdMonday à ewdSunday. Par défaut [].
FirstDayOfWeek	TsgcEditorFirstDayOfWeek	efdLocale (par défaut) prend le premier jour de la semaine du système d'exploitation, le lundi lorsqu'il n'en a pas. efdMonday, efdSaturday et efdSunday le forcent.
ShowWeekNumbers	Boolean	Ajoute une colonne avec le numéro de semaine de chaque ligne. Par défaut False.
WeekNumbering	TsgcEditorWeekNumbering	ewnISO8601 (par défaut) : la semaine 1 contient le premier jeudi de l'année. ewnSimple : la semaine 1 contient le 1er janvier.
ShowToday	Boolean	Marque le jour actuel. Par défaut True.
ShowOtherMonthDays	Boolean	Affiche les jours du mois précédent et du mois suivant qui complètent les six semaines de la grille. Par défaut True.

View	TsgcEditor-Calendar-View	ecvMonth (par défaut), ecvYear (les 12 mois d'une année) ou ecvDecade (les années d'une décennie).
Display-Month	TDate	Le premier jour du mois affiché, qui est toujours le mois du focus clavier. Il n'est pas enregistré dans le formulaire.
Theme	TsgcUI-Theme	utLight (par défaut) ou utDark. Voir Thèmes.
AccentCo-lor	TsgcUIColor	La couleur d'accentuation ARGB de la sélection, du remplissage et de l'anneau de focus. 0 (par défaut) utilise la couleur d'accentuation du thème.
TextStyle	TsgcUIText-Style	Size, Bold et Color du texte. Un Color de 0 suit le thème.
ReadOnly	Boolean	La valeur est affichée mais l'utilisateur ne peut pas la modifier. Par défaut False.
Accessible-Name	string	Le nom qu'un lecteur d'écran annonce avant la valeur. Voir Clavier et acces-sibilité.
FormatSet-tings	TFormatSet-tings	Public. Les noms, les formats et les séparateurs que le contrôle utilise. Il dé-marre avec les paramètres des paramètres régionaux de l'utilisateur. Voir Localisation.
Calendar-Model	TsgcCalen-darModel	Public, en lecture seule. Le modèle du contrôle. Model renvoie le même ob-jet qu'un TsgcEditorModel.

Méthodes

Méthode	Ren-voie	Description
GoToDate(aDate)		Affiche le mois de aDate, maintenu dans les limites, et y déplace le focus clavier. Il ne sélectionne pas la date.
IsDateEnabled(aDate)	Boo-lean	True lorsque la date est dans les limites, pas sur un jour de semaine désactivé, pas dans DisabledDates et pas désactivée par OnIsDate-Disabled.
SaveToPNG(aStream ou aFileName, aScale = 1)		Dessine le contrôle à sa taille actuelle dans un PNG. aScale multi-plie la taille. Voir Sortie sans interface.
SaveToSVG(aStream ou aFileName, aScale = 1)		La même chose dans un SVG.

Événements

Événe-ment	Type	Description
OnChange	TNotifyEvent	La date sélectionnée ou la plage a changé.
OnSelect	TNotifyEvent	Une date est choisie en mode esmSingle, avec la souris ou avec Entrée ou Espace.

OnRange-Select	TNotifyEvent	Une plage est complète : le second clic, ou un clic ou une flèche avec Maj.
OnIsDate-Disabled	TsgcCalenda-rIsDateDisa-bledEvent	(Sender; const aDate: TDateTime; var aDisabled: Boolean). Mettez aDisa-bled à True pour désactiver un jour, comme un jour férié issu d'une base de données. Il est demandé pour les jours que le calendrier affiche et pour le jour en cours de sélection, donc gardez-le rapide.
OnView-Change	TNotifyEvent	Vue modifiée : l'utilisateur a zoomé ou dézoomé.
OnDisplay-Month-Change	TNotifyEvent	Un autre mois est affiché.

Clavier

Les flèches déplacent le focus d'un jour ou d'une semaine, Page préc. et Page suiv. d'un mois et, avec Ctrl, d'une année. Ctrl+Haut dézoome. Entrée ou Espace sélectionne le jour ayant le focus, ou zoome dans les vues année et décennie. Maj avec les flèches étend une plage. Les jours désactivés sont ignorés.

Exemple : Un séjour de deux semaines au plus

```
uses
  SysUtils, DateUtils, Classes, Controls, StdCtrls,
  sgcEditors_Types, sgcEditors_VCL_Date;

type
  TStayEvents = class
  public
    Calendar: TsgcVCLMonthCalendar;
    Summary: TLabel;
    procedure RangeSelect(Sender: TObject);
    procedure IsDateDisabled(Sender: TObject; const aDate: TDateTime;
      var aDisabled: Boolean);
  end;

procedure TStayEvents.RangeSelect(Sender: TObject);
begin
  // RangeStart and RangeEnd are whole days, both included
  Summary.Caption := DateToStr(Calendar.RangeStart) + ' to ' +
    DateToStr(Calendar.RangeEnd) + ', ' +
    IntToStr(DaysBetween(Calendar.RangeEnd, Calendar.RangeStart)) + ' nights';
end;

procedure TStayEvents.IsDateDisabled(Sender: TObject; const aDate: TDateTime;
  var aDisabled: Boolean);
var
  vYear, vMonth, vDay: Word;
begin
  // the hotel is closed on the first day of every month
  DecodeDate(aDate, vYear, vMonth, vDay);
  aDisabled := vDay = 1;
end;

procedure SetupStay(aCalendar: TsgcVCLMonthCalendar; aEvents: TStayEvents);
begin
  aCalendar.SelectionMode := esmRange;
  aCalendar.MaxRangeDays := 14;
  aCalendar.MinDate := Date;
  aCalendar.MaxDate := IncMonth(Date, 12);
  aCalendar.FirstDayOfWeek := efdMonday;
  aCalendar.ShowWeekNumbers := True;
```

```
  aCalendar.DisabledWeekDays := [ewdSunday];
  // one ISO date per line
  aCalendar.DisabledDates.Add('2026-12-25');
  aCalendar.DisabledDates.Add('2027-01-06');
  aEvents.Calendar := aCalendar;
  aCalendar.OnRangeSelect := aEvents.RangeSelect;
  aCalendar.OnIsDateDisabled := aEvents.IsDateDisabled;
  aCalendar.GoToDate(Date);
end;
```

Voir aussi

- Sélecteur de date
- Sélecteur de date et d'heure
- Localisation
- TsgcEditorLocale

Sélecteur de date

Classes : TsgcVCLDatePicker (unit sgcEditors_VCL_Date) et TsgcFMXDatePicker (unit sgcEditors_FMX_Date)
Le sélecteur de date est un edit natif pour une date avec un calendrier déroulant. L'utilisateur saisit la date, fait avancer le jour, le mois ou l'année sous le curseur avec Haut et Bas, ou la choisit dans le calendrier. Il partage ses classes de base, TsgcVCLCustomDatePicker et TsgcFMXCustomDatePicker, et son modèle, TsgcDatePickerModel, avec le sélecteur d'heure et le sélecteur de date et d'heure.
La classe VCL descend de TsgcVCLCustomDatePicker et la classe FireMonkey de TsgcFMXCustomDatePicker. Les deux publient les propriétés et les événements du tableau ci-dessous, plus les habituels de leur framework, comme Align, Anchors, Enabled, Hint, TabOrder, TabStop, OnEnter, OnExit et les événements de clavier et de souris. Voir Contrôles.

Propriétés

Propriété	Type	Description
Date	TDate	La date. Lisez d'abord IsEmpty lorsque AllowEmpty vaut True.
DisplayFormat	string	Le format du texte, comme dd/mm/yyyy ou ddd d mmm yyyy. Vide (par défaut), il utilise le format de date court de FormatSettings. Voir Saisie de texte et IME.
AllowEmpty	Boolean	Permet à l'utilisateur de supprimer le texte, ce qui vide la valeur. Par défaut False, un texte vide renvoie alors la dernière valeur.
IsEmpty	Boolean	Public, en lecture seule. True tant que la valeur est vide.
TextHint	string	L'indication grise d'un edit vide. VCL l'affiche à partir de Delphi 2009, et FireMonkey l'affiche comme TextPrompt de l'edit.
DroppedDown	Boolean	Public. True tant que la liste déroulante est ouverte. L'affecter ouvre ou ferme la liste déroulante.
MinDate, MaxDate	TDate	Le premier et le dernier jour que l'utilisateur peut choisir, comparés comme jours entiers. 0 (par défaut) signifie aucune limite, et des limites inversées sont échangées.
DisabledDates	TStrings	Les jours qui ne peuvent pas être choisis, une date ISO (yyyy-mm-dd) par ligne.
DisabledWeekDays	TsgcEditorWeekDays	Les jours de la semaine qui ne peuvent pas être choisis, un ensemble de ewdMonday à ewdSunday. Par défaut [].
FirstDayOfWeek, ShowWeekNumbers		Comme dans le calendrier mensuel, pour le calendrier déroulant.
DropDownCalendar	TsgcCalendarModel	Public, en lecture seule. Le modèle du calendrier déroulant.
Theme	TsgcUITheme	utLight (par défaut) ou utDark. Voir Thèmes.
AccentColor	TsgcUIColor	La couleur d'accentuation ARGB de la sélection, du remplissage et de l'anneau de focus. 0 (par défaut) utilise la couleur d'accentuation du thème.

TextStyle	TsgcUI-TextStyle	Size, Bold et Color du texte. Un Color de 0 suit le thème.
ReadOnly	Boolean	La valeur est affichée mais l'utilisateur ne peut pas la modifier. Par défaut False.
Accessible-Name	string	Le nom qu'un lecteur d'écran annonce avant la valeur. Voir Clavier et accessibilité.
FormatSettings	TFormat-Settings	Public. Les noms, les formats et les séparateurs que le contrôle utilise. Il démarre avec les paramètres des paramètres régionaux de l'utilisateur. Voir Localisation.
PickerModel	TsgcDate-PickerMo-del	Public, en lecture seule. Le modèle du contrôle. Model renvoie le même objet qu'un TsgcEditorModel.

Méthodes

Méthode	Ren-voie	Description
Clear		Vide la valeur lorsque AllowEmpty vaut True. Sinon, il ne fait rien.
DropDown		Ouvre la liste déroulante, comme le font le bouton, F4 et Alt+Bas.
CloseUp(aAccept)		Ferme la liste déroulante. aAccept à True valide le texte, False restaure la valeur qu'il avait à l'ouverture.
IsDateEnabled(aDate)	Boo-lean	True lorsque la date est dans les limites et n'est pas désactivée dans le calendrier déroulant. Une validation refuse les dates pour lesquelles il vaut False.
SaveToPNG(aStream ou aFileName, aScale = 1)		Dessine le contrôle à sa taille actuelle dans un PNG. aScale multiplie la taille. Voir Sortie sans interface.
SaveToSVG(aStream ou aFileName, aScale = 1)		La même chose dans un SVG.

Événements

Événement	Type	Description
OnChange	TNoti-fyEvent	Une nouvelle valeur : saisie et validée, choisie dans la liste déroulante ou avancée avec Haut et Bas.
OnDrop-Down, On-CloseUp	TNoti-fyEvent	La liste déroulante s'ouvre ou se ferme.
OnParseEr-ror	TsgcDatePi-ckerPar-seErro-rEvent	(Sender; const aText: string; var aValue: TDateTime; var aAccept: Boolean). Le texte validé ne s'analyse pas. aValue arrive avec la valeur actuelle et aAccept à False. Mettez aValue et aAccept à True pour prendre une autre valeur, qui est alors maintenue dans les limites. Sinon, la dernière valeur valide revient.

OnIsDateDisabled	TsgcCalendarIsDateDisabledEvent	Comme dans le calendrier mensuel : désactive un jour dans la liste déroulante et à la validation.

Saisie et liste déroulante

- Le texte est saisi dans un edit natif, de sorte que l'IME, le presse-papiers et l'annulation fonctionnent. L'analyse est tolérante et suit les noms et les séparateurs de FormatSettings. Voir Saisie de texte et IME.
- Haut et Bas valident le texte saisi, puis font avancer le champ sous le curseur, qui reboucle dans sa plage, et sélectionnent de nouveau le champ. Les limites sont appliquées après le pas.
- Entrée, la perte du focus et un choix dans la liste déroulante valident le texte. Une valeur hors des limites est ramenée dedans, et une date désactivée est refusée.
- F4, Alt+Bas et le bouton ouvrent la liste déroulante sur la valeur actuelle. Tant qu'elle est ouverte, elle suit le texte saisi, Haut, Bas, Page préc. et Page suiv. agissent sur elle, et Échap la ferme et restaure la valeur et le texte qu'elle avait à l'ouverture.

Exemple : Une échéance qui comprend "aujourd'hui"

```
uses
  SysUtils, Classes, Controls,
  sgcEditors_Types, sgcEditors_VCL_Date;

type
  TDueEvents = class
  public
    procedure ParseError(Sender: TObject; const aText: string;
      var aValue: TDateTime; var aAccept: Boolean);
  end;

procedure TDueEvents.ParseError(Sender: TObject; const aText: string;
  var aValue: TDateTime; var aAccept: Boolean);
begin
  // words the parse does not know; the value is clamped to the limits
  if SameText(Trim(aText), 'today') then
  begin
    aValue := Date;
    aAccept := True;
  end
  else if SameText(Trim(aText), 'tomorrow') then
  begin
    aValue := Date + 1;
    aAccept := True;
  end;
end;

procedure SetupDueDate(aPicker: TsgcVCLDatePicker; aEvents: TDueEvents);
begin
  aPicker.DisplayFormat := 'dd mmm yyyy';
  aPicker.AllowEmpty := True;
  aPicker.TextHint := 'No due date';
  aPicker.MinDate := Date;
  aPicker.DisabledWeekDays := [ewdSaturday, ewdSunday];
  aPicker.ShowWeekNumbers := True;
  aPicker.OnParseError := aEvents.ParseError;
  aPicker.Clear;
end;

function DueDateText(aPicker: TsgcVCLDatePicker): string;
begin
  if aPicker.IsEmpty then
    Result := 'none'
  else
```

```
    Result := FormatDateTime('yyyy-mm-dd', aPicker.Date);
end;
```

Voir aussi

- Calendrier mensuel
- Sélecteur de date et d'heure
- Sélecteur de date DB
- Saisie de texte et IME

Sélecteur d'heure

Classes : TsgcVCLTimePicker (unit sgcEditors_VCL_Date) et TsgcFMXTimePicker (unit sgcEditors_FMX_Date)
Le sélecteur d'heure est un edit natif pour une heure de la journée avec une liste déroulante d'heures. L'utilisateur saisit l'heure, fait avancer l'heure, la minute, la seconde ou AM et PM sous le curseur avec Haut et Bas, ou choisit une heure dans la liste. La liste commence à 00:00 et contient une heure toutes les ListInterval minutes entre MinTime et MaxTime, écrite avec les champs d'heure de DisplayFormat.
La classe VCL descend de TsgcVCLCustomDatePicker et la classe FireMonkey de TsgcFMXCustomDatePicker. Les deux publient les propriétés et les événements du tableau ci-dessous, plus les habituels de leur framework, comme Align, Anchors, Enabled, Hint, TabOrder, TabStop, OnEnter, OnExit et les événements de clavier et de souris. Voir Contrôles.

Propriétés

Propriété	Type	Description
Time	TTime	L'heure de la journée. Le sélecteur d'heure démarre à 00:00.
DisplayFormat	string	Le format du texte, comme hh:nn ou h:nn ampm. Un format avec ss affiche les secondes. Vide (par défaut), il utilise le format d'heure court de FormatSettings.
MinTime, MaxTime	TTime	La première et la dernière heure que l'utilisateur peut définir. 0 (par défaut) signifie aucune limite de ce côté.
ListInterval	Integer	Les minutes entre deux heures de la liste déroulante. Par défaut 30. 0 supprime la liste déroulante et son bouton.
AllowEmpty	Boolean	Permet à l'utilisateur de supprimer le texte, ce qui vide la valeur. Par défaut False, un texte vide renvoie alors la dernière valeur.
IsEmpty	Boolean	Public, en lecture seule. True tant que la valeur est vide.
TextHint	string	L'indication grise d'un edit vide. VCL l'affiche à partir de Delphi 2009, et FireMonkey l'affiche comme TextPrompt de l'edit.
DroppedDown	Boolean	Public. True tant que la liste déroulante est ouverte. L'affecter ouvre ou ferme la liste déroulante.
DropDownTimeList	TsgcEditorListModel	Public, en lecture seule. Le modèle de la liste déroulante.
Theme	TsgcUITheme	utLight (par défaut) ou utDark. Voir Thèmes.
AccentColor	TsgcUIColor	La couleur d'accentuation ARGB de la sélection, du remplissage et de l'anneau de focus. 0 (par défaut) utilise la couleur d'accentuation du thème.
TextStyle	TsgcUITextStyle	Size, Bold et Color du texte. Un Color de 0 suit le thème.
ReadOnly	Boolean	La valeur est affichée mais l'utilisateur ne peut pas la modifier. Par défaut False.
AccessibleName	string	Le nom qu'un lecteur d'écran annonce avant la valeur. Voir Clavier et accessibilité.

FormatSettings	TFormatSettings	Public. Les noms, les formats et les séparateurs que le contrôle utilise. Il démarre avec les paramètres des paramètres régionaux de l'utilisateur. Voir Localisation.
PickerModel	TsgcDatePickerModel	Public, en lecture seule. Le modèle du contrôle. Model renvoie le même objet qu'un TsgcEditorModel.

Méthodes

Méthode	Renvoie	Description
Clear		Vide la valeur lorsque AllowEmpty vaut True. Sinon, il ne fait rien.
DropDown		Ouvre la liste déroulante, comme le font le bouton, F4 et Alt+Bas.
CloseUp(aAccept)		Ferme la liste déroulante. aAccept à True valide le texte, False restaure la valeur qu'il avait à l'ouverture.
SaveToPNG(aStream ou aFileName, aScale = 1)		Dessine le contrôle à sa taille actuelle dans un PNG. aScale multiplie la taille. Voir Sortie sans interface.
SaveToSVG(aStream ou aFileName, aScale = 1)		La même chose dans un SVG.

Événements

Événement	Type	Description
OnChange	TNotifyEvent	Une nouvelle valeur : saisie et validée, choisie dans la liste déroulante ou avancée avec Haut et Bas.
OnDropDown, OnCloseUp	TNotifyEvent	La liste déroulante s'ouvre ou se ferme.
OnParseError	TsgcDatePickerParseErrorEvent	(Sender; const aText: string; var aValue: TDateTime; var aAccept: Boolean). Le texte validé ne s'analyse pas. aValue arrive avec la valeur actuelle et aAccept à False. Mettez aValue et aAccept à True pour prendre une autre valeur, qui est alors maintenue dans les limites. Sinon, la dernière valeur valide revient.

Saisie et liste déroulante

- Le texte est saisi dans un edit natif, de sorte que l'IME, le presse-papiers et l'annulation fonctionnent. L'analyse est tolérante et suit les noms et les séparateurs de FormatSettings. Voir Saisie de texte et IME.
- Haut et Bas valident le texte saisi, puis font avancer le champ sous le curseur, qui reboucle dans sa plage, et sélectionnent de nouveau le champ. Les limites sont appliquées après le pas.
- Entrée, la perte du focus et un choix dans la liste déroulante valident le texte. Une valeur hors des limites est ramenée dedans, et une date désactivée est refusée.
- F4, Alt+Bas et le bouton ouvrent la liste déroulante sur la valeur actuelle. Tant qu'elle est ouverte, elle suit le texte saisi, Haut, Bas, Page préc. et Page suiv. agissent sur elle, et Échap la ferme et restaure la valeur et le texte qu'elle avait à l'ouverture.

Exemple : Une réunion pendant les heures de bureau

```
uses
  SysUtils, Classes, Controls,
  sgcEditors_VCL_Date;

type
  TMeetingEvents = class
  public
    Start: TsgcVCLTimePicker;
    Finish: TsgcVCLTimePicker;
    procedure StartChange(Sender: TObject);
  end;

procedure TMeetingEvents.StartChange(Sender: TObject);
begin
  // the meeting ends at least 30 minutes after it starts
  Finish.MinTime := Start.Time + EncodeTime(0, 30, 0, 0);
  if Finish.Time < Finish.MinTime then
    Finish.Time := Finish.MinTime;
end;

procedure SetupMeeting(aEvents: TMeetingEvents);
begin
  // 24 hour text, a time list every 15 minutes, office hours only
  aEvents.Start.DisplayFormat := 'hh:nn';
  aEvents.Start.ListInterval := 15;
  aEvents.Start.MinTime := EncodeTime(8, 0, 0, 0);
  aEvents.Start.MaxTime := EncodeTime(18, 0, 0, 0);
  aEvents.Start.Time := EncodeTime(9, 0, 0, 0);
  aEvents.Finish.DisplayFormat := 'hh:nn';
  aEvents.Finish.ListInterval := 15;
  aEvents.Finish.MaxTime := EncodeTime(19, 0, 0, 0);
  aEvents.Start.OnChange := aEvents.StartChange;
  aEvents.StartChange(nil);
end;
```

Voir aussi

- Sélecteur de date
- Sélecteur de date et d'heure
- Sélecteur d'heure DB

Sélecteur de date et d'heure

Classes : TsgcVCLDateTimePicker (unit sgcEditors_VCL_Date) et TsgcFMXDateTimePicker (unit sgcEditors_FMX_Date)
Le sélecteur de date et d'heure modifie une date et une heure dans un seul texte. Sa liste déroulante affiche côte à côte le calendrier et la liste des heures. Un choix dans le calendrier garde la liste déroulante ouverte, de sorte que l'utilisateur peut choisir ensuite l'heure, et un choix dans la liste des heures la ferme. MinDate, MaxDate et les jours désactivés limitent la date, et MinTime et MaxTime limitent l'heure de la journée.
La classe VCL descend de TsgcVCLCustomDatePicker et la classe FireMonkey de TsgcFMXCustomDatePicker. Les deux publient les propriétés et les événements du tableau ci-dessous, plus les habituels de leur framework, comme Align, Anchors, Enabled, Hint, TabOrder, TabStop, OnEnter, OnExit et les événements de clavier et de souris. Voir Contrôles.

Propriétés

Propriété	Type	Description
DateTime	TDateTime	La date et l'heure.
DisplayFormat	string	Le format du texte, comme dd/mm/yyyy hh:nn. Vide (par défaut), il utilise le format de date court et le format d'heure court de FormatSettings, séparés par une espace.
MinDate, MaxDate	TDate	Le premier et le dernier jour que l'utilisateur peut choisir, comparés comme jours entiers. 0 (par défaut) signifie aucune limite, et des limites inversées sont échangées.
DisabledDates	TStrings	Les jours qui ne peuvent pas être choisis, une date ISO (yyyy-mm-dd) par ligne.
DisabledWeekDays	TsgcEditorWeekDays	Les jours de la semaine qui ne peuvent pas être choisis, un ensemble de ewdMonday à ewdSunday. Par défaut [].
MinTime, MaxTime	TTime	La première et la dernière heure de la journée. 0 (par défaut) signifie aucune limite de ce côté.
ListInterval	Integer	Les minutes entre deux heures de la liste des heures. Par défaut 30.
FirstDayOfWeek, ShowWeekNumbers		Comme dans le calendrier mensuel, pour le calendrier déroulant.
AllowEmpty	Boolean	Permet à l'utilisateur de supprimer le texte, ce qui vide la valeur. Par défaut False, un texte vide renvoie alors la dernière valeur.
IsEmpty	Boolean	Public, en lecture seule. True tant que la valeur est vide.
TextHint	string	L'indication grise d'un edit vide. VCL l'affiche à partir de Delphi 2009, et FireMonkey l'affiche comme TextPrompt de l'edit.
DroppedDown	Boolean	Public. True tant que la liste déroulante est ouverte. L'affecter ouvre ou ferme la liste déroulante.
DropDownCalendar, DropDownTimeList	TsgcCalendarModel, TsgcEditorListModel	Public, en lecture seule. Les modèles des deux parties de la liste déroulante.

Theme	TsgcUITheme	utLight (par défaut) ou utDark. Voir Thèmes.
AccentColor	TsgcUIColor	La couleur d'accentuation ARGB de la sélection, du remplissage et de l'anneau de focus. 0 (par défaut) utilise la couleur d'accentuation du thème.
TextStyle	TsgcUIText-Style	Size, Bold et Color du texte. Un Color de 0 suit le thème.
ReadOnly	Boolean	La valeur est affichée mais l'utilisateur ne peut pas la modifier. Par défaut False.
Accessible-Name	string	Le nom qu'un lecteur d'écran annonce avant la valeur. Voir Clavier et accessibilité.
FormatSettings	TFormatSet-tings	Public. Les noms, les formats et les séparateurs que le contrôle utilise. Il démarre avec les paramètres des paramètres régionaux de l'utilisateur. Voir Localisation.
PickerModel	TsgcDatePi-ckerModel	Public, en lecture seule. Le modèle du contrôle. Model renvoie le même objet qu'un TsgcEditorModel.

Méthodes

Méthode	Ren-voie	Description
Clear		Vide la valeur lorsque AllowEmpty vaut True. Sinon, il ne fait rien.
DropDown		Ouvre la liste déroulante, comme le font le bouton, F4 et Alt+Bas.
CloseUp(aAccept)		Ferme la liste déroulante. aAccept à True valide le texte, False restaure la valeur qu'il avait à l'ouverture.
IsDateEnabled(aDate)	Boo-lean	True lorsque la date est dans les limites et n'est pas désactivée dans le calendrier déroulant.
SaveToPNG(aStream ou aFileName, aScale = 1)		Dessine le contrôle à sa taille actuelle dans un PNG. aScale multiplie la taille. Voir Sortie sans interface.
SaveToSVG(aStream ou aFileName, aScale = 1)		La même chose dans un SVG.

Événements

Événement	Type	Description
OnChange	TNoti-fyEvent	Une nouvelle valeur : saisie et validée, choisie dans la liste déroulante ou avancée avec Haut et Bas.
OnDrop-Down, On-CloseUp	TNoti-fyEvent	La liste déroulante s'ouvre ou se ferme.

OnParseError	TsgcDatePickerParseErrorEvent	(Sender; const aText: string; var aValue: TDateTime; var aAccept: Boolean). Le texte validé ne s'analyse pas. aValue arrive avec la valeur actuelle et aAccept à False. Mettez aValue et aAccept à True pour prendre une autre valeur, qui est alors maintenue dans les limites. Sinon, la dernière valeur valide revient.
OnIsDateDisabled	TsgcCalendarIsDateDisabledEvent	Comme dans le calendrier mensuel.

Saisie et liste déroulante

- Le texte est saisi dans un edit natif, de sorte que l'IME, le presse-papiers et l'annulation fonctionnent. L'analyse est tolérante et suit les noms et les séparateurs de FormatSettings. Voir Saisie de texte et IME.
- Haut et Bas valident le texte saisi, puis font avancer le champ sous le curseur, qui reboucle dans sa plage, et sélectionnent de nouveau le champ. Les limites sont appliquées après le pas.
- Entrée, la perte du focus et un choix dans la liste déroulante valident le texte. Une valeur hors des limites est ramenée dedans, et une date désactivée est refusée.
- F4, Alt+Bas et le bouton ouvrent la liste déroulante sur la valeur actuelle. Tant qu'elle est ouverte, elle suit le texte saisi, Haut, Bas, Page préc. et Page suiv. agissent sur elle, et Échap la ferme et restaure la valeur et le texte qu'elle avait à l'ouverture.

Dans la liste déroulante du sélecteur de date et d'heure, Haut, Bas, Page préc. et Page suiv. agissent sur le calendrier lorsque le curseur est dans un champ de date et sur la liste des heures lorsqu'il est dans un champ d'heure.

Exemple : Un créneau de livraison

```
uses
  SysUtils, Classes, Controls,
  sgcEditors_Types, sgcEditors_VCL_Date;

procedure SetupDelivery(aPicker: TsgcVCLDateTimePicker);
begin
  // the date and the time in one text; the drop-down shows the calendar
  // and the time list side by side
  aPicker.DisplayFormat := 'ddd dd/mm/yyyy hh:nn';
  aPicker.MinDate := Date + 1;
  aPicker.MaxDate := Date + 60;
  aPicker.DisabledWeekDays := [ewdSunday];
  aPicker.MinTime := EncodeTime(9, 0, 0, 0);
  aPicker.MaxTime := EncodeTime(17, 0, 0, 0);
  aPicker.ListInterval := 60;
  aPicker.DateTime := Date + 1 + EncodeTime(9, 0, 0, 0);
end;

function DeliveryWindow(aPicker: TsgcVCLDateTimePicker): string;
begin
  // DateTime holds both parts
  Result := FormatDateTime('dddd d mmmm, hh:nn', aPicker.DateTime) + ' to ' +
    FormatDateTime('hh:nn', aPicker.DateTime + EncodeTime(1, 0, 0, 0));
end;
```

Voir aussi

- Sélecteur de date
- Sélecteur d'heure
- Calendrier mensuel

Curseur de plage

Classes : TsgcVCLRangeSlider (unit sgcEditors_VCL_Value) et TsgcFMXRangeSlider (unit sgcEditors_FMX_Value)
Le curseur de plage est un contrôle dessiné sur mesure, pouvant recevoir le focus, qui choisit un nombre entre Min et Max avec une poignée, ou une plage avec deux poignées dans RangeMode. Il est horizontal, avec Min à gauche, ou vertical, avec Min en bas. Il peut dessiner des graduations, des étiquettes et une info-bulle avec la valeur au-dessus de la poignée glissée. Un clic sur le rail y déplace la poignée la plus proche et la fait glisser, et les poignées ne se croisent jamais.
RangeMode requiert l'édition Registered ou Trial. Dans Community, la propriété est conservée, une poignée est dessinée et modifiée, et EndValue est conservée mais pas affichée. Voir Éditions.
La classe VCL descend de TsgcVCLCustomEditor et la classe FireMonkey de TsgcFMXCustomEditor. Les deux publient les propriétés et les événements du tableau ci-dessous, plus les habituels de leur framework, comme Align, Anchors, Enabled, Hint, TabOrder, TabStop, OnEnter, OnExit et les événements de clavier et de souris. Voir Contrôles.

Propriétés

Propriété	Type	Description
Min, Max	Double	Les extrémités de l'échelle. Par défaut 0 et 100.
Value	Double	La valeur, ou l'extrémité basse de la plage dans RangeMode. Par défaut 0.
EndValue	Double	L'extrémité haute de la plage dans RangeMode. Par défaut 100.
RangeMode	Boolean	Deux poignées, Value et EndValue. Par défaut False. Éditions Registered et Trial.
MinRange	Double	La plus petite distance entre les deux poignées. Par défaut 0.
Step	Double	Chaque valeur s'aimante sur Min plus un multiple de Step, et les flèches avancent d'un Step. Par défaut 1. 0 est continu, et les flèches avancent alors de 1 % de l'échelle.
LargeStep	Double	Le pas de Page préc. et Page suiv. Par défaut 10. 0 vaut un dixième de l'échelle.
Orientation	TsgcEditorOrientation	eorHorizontal (par défaut) ou eorVertical.
TickInterval	Double	La distance entre deux graduations. 0 (par défaut) ne dessine aucune graduation.
TickPlacement	TsgcEditorTickPlacement	etpBottomRight (par défaut), etpTopLeft, etpBoth ou etpNone.
ShowLabels	Boolean	Écrit la valeur à chaque graduation, ou à Min et Max lorsqu'il n'y a pas de graduations. Les étiquettes qui se chevaucheraient sont omises. Par défaut False.
LabelFormat	string	Un masque FormatFloat pour les étiquettes et l'info-bulle de valeur. Vide (par défaut), il écrit autant de décimales que Step en demande.
ShowValueTip	Boolean	Affiche la valeur au-dessus de la poignée pendant qu'elle est glissée. Par défaut True.

Theme	TsgcUI-Theme	utLight (par défaut) ou utDark. Voir Thèmes.
Accent-Color	TsgcUIColor	La couleur d'accentuation ARGB de la sélection, du remplissage et de l'anneau de focus. 0 (par défaut) utilise la couleur d'accentuation du thème.
TextStyle	TsgcUIText-Style	Size, Bold et Color du texte. Un Color de 0 suit le thème.
ReadOnly	Boolean	La valeur est affichée mais l'utilisateur ne peut pas la modifier. Par défaut False.
Accessi-bleName	string	Le nom qu'un lecteur d'écran annonce avant la valeur. Voir Clavier et accessi-bilité.
Format-Settings	TFormatSet-tings	Public. Les noms, les formats et les séparateurs que le contrôle utilise. Il dé-marre avec les paramètres des paramètres régionaux de l'utilisateur. Voir Lo-calisation.
SliderMo-del	TsgcSlider-Model	Public, en lecture seule. Le modèle du contrôle. Model renvoie le même objet qu'un TsgcEditorModel.

Méthodes

Méthode	Ren-voie	Description
SaveToPNG(aStream ou aFile-Name, aScale = 1)		Dessine le contrôle à sa taille actuelle dans un PNG. aScale multiplie la taille. Voir Sortie sans interface.
SaveToSVG(aStream ou aFile-Name, aScale = 1)		La même chose dans un SVG.

Événements

Événement	Type	Description
OnChange	TNoti-fyEvent	Value ou EndValue a changé. Il se déclenche pendant un glissement, pour chaque nouvelle valeur.
OnChange-Complete	TNoti-fyEvent	L'utilisateur a terminé une modification : relâchement de la souris, relâche-ment d'une touche ou un cran de molette. Utilisez-le pour un travail qui ne doit pas s'exécuter pendant un glissement, comme une requête.
OnFormat-Label	TsgcSlider-FormatLa-belEvent	(Sender; aValue: Double; var aText: string). Écrit le texte d'une étiquette et de l'info-bulle de valeur, comme une devise ou une unité.

Clavier

Les flèches déplacent la poignée active d'un Step, Page préc. et Page suiv. d'un LargeStep, et Début et Fin vont aux extrémités. Dans RangeMode, Tab passe de la poignée basse à la poignée haute et Maj+Tab revient, puis le Tab suivant quitte le contrôle. Ctrl+Gauche ou Ctrl+Bas rend active la poignée basse, et Ctrl+Droite ou Ctrl+Haut la poignée haute. Sous FireMonkey, Tab passe toujours au contrôle suivant, de sorte que les touches Ctrl changent de poignée.

Exemple : Un filtre de prix

```
uses
  SysUtils, Classes, Controls, StdCtrls,
  sgcEditors_Types, sgcEditors_VCL_Value;

type
  TPriceEvents = class
  public
    Slider: TsgcVCLRangeSlider;
    Summary: TLabel;
    procedure FormatLabel(Sender: TObject; aValue: Double; var aText: string);
    procedure ChangeComplete(Sender: TObject);
  end;

procedure TPriceEvents.FormatLabel(Sender: TObject; aValue: Double;
  var aText: string);
begin
  // the text of the labels and of the value tip
  aText := FormatFloat('0', aValue) + ' EUR';
end;

procedure TPriceEvents.ChangeComplete(Sender: TObject);
begin
  // once on mouse up, key up or wheel, not on every step of a drag
  Summary.Caption := 'From ' + FormatFloat('0', Slider.Value) + ' to ' +
    FormatFloat('0', Slider.EndValue) + ' EUR';
end;

procedure SetupPrice(aEvents: TPriceEvents);
begin
  aEvents.Slider.Min := 0;
  aEvents.Slider.Max := 500;
  aEvents.Slider.Step := 10;
  aEvents.Slider.LargeStep := 50;
  // two thumbs at least 50 apart (Registered and Trial editions)
  aEvents.Slider.RangeMode := True;
  aEvents.Slider.MinRange := 50;
  aEvents.Slider.Value := 100;
  aEvents.Slider.EndValue := 300;
  aEvents.Slider.TickInterval := 100;
  aEvents.Slider.TickPlacement := etpBottomRight;
  aEvents.Slider.ShowLabels := True;
  aEvents.Slider.OnFormatLabel := aEvents.FormatLabel;
  aEvents.Slider.OnChangeComplete := aEvents.ChangeComplete;
end;
```

Voir aussi

- Curseur de plage DB
- Spin edit
- Clavier et accessibilité

Sélecteur de couleur

Classes : TsgcVCLColorPicker (unit sgcEditors_VCL_Color) et TsgcFMXColorPicker (unit sgcEditors_FMX_Color)
Le sélecteur de couleur affiche, de haut en bas, une grille d'échantillons, un carré de saturation et de valeur, une barre de teinte, une barre alpha et une ligne avec un aperçu et un edit natif pour le texte hexadécimal. Parts choisit lesquels sont affichés. L'edit hexadécimal accepte les textes #RGB, #RGBA, #RRGGBB et #RRGGBBAA. Le sélecteur garde la teinte séparée de la couleur, de sorte qu'un gris conserve sa teinte et que le carré ne saute pas.
Le carré HSV et la barre alpha requièrent l'édition Registered ou Trial. Dans Community, ils ne sont pas dessinés, l'alpha vaut toujours $FF et le texte hexadécimal est #RRGGBB. Voir Éditions.
La classe VCL descend de TsgcVCLCustomEditor et la classe FireMonkey de TsgcFMXCustomEditor. Les deux publient les propriétés et les événements du tableau ci-dessous, plus les habituels de leur framework, comme Align, Anchors, Enabled, Hint, TabOrder, TabStop, OnEnter, OnExit et les événements de clavier et de souris. Voir Contrôles.

Propriétés

Propriété	Type	Description
SelectedColor	TsgcUIColor	La couleur en ARGB, comme $FF2A78D6. Elle a la disposition du TAlphaColor de FireMonkey.
SelectedVCLColor	TColor	VCL uniquement, public. La couleur sous forme de TColor opaque, en lecture et en écriture.
Parts	TsgcEditorColorParts	Les parties affichées : ecpSwatches, ecpHSV, ecpAlpha et ecpHex. Par défaut les quatre.
Swatches	TStrings	Les couleurs des échantillons, une couleur hexadécimale comme #1E88E5 par ligne. Par défaut 24 couleurs tirées des palettes de sgcUICore et des gris.
SwatchSize	Double	La taille d'un échantillon en pixels logiques. Par défaut 20.
SwatchColumns	Integer	Les échantillons par ligne. Par défaut 8.
Theme	TsgcUITheme	utLight (par défaut) ou utDark. Voir Thèmes.
AccentColor	TsgcUIColor	La couleur d'accentuation ARGB de la sélection, du remplissage et de l'anneau de focus. 0 (par défaut) utilise la couleur d'accentuation du thème.
TextStyle	TsgcUITextStyle	Size, Bold et Color du texte. Un Color de 0 suit le thème.
ReadOnly	Boolean	La valeur est affichée mais l'utilisateur ne peut pas la modifier. Par défaut False.
AccessibleName	string	Le nom qu'un lecteur d'écran annonce avant la valeur. Voir Clavier et accessibilité.
ColorModel	TsgcColorPickerModel	Public, en lecture seule. Le modèle du contrôle. Model renvoie le même objet qu'un TsgcEditorModel.

Méthodes

Méthode	Ren-voie	Description
SaveToPNG(aStream ou aFile-Name, aScale = 1)		Dessine le contrôle à sa taille actuelle dans un PNG. aScale multiplie la taille. Voir Sortie sans interface.
SaveToSVG(aStream ou aFile-Name, aScale = 1)		La même chose dans un SVG.

Événements

Événe-ment	Type	Description
OnChange	TNoti-fyEvent	La couleur a changé. Il se déclenche pendant un glissement sur le carré ou les barres, pour chaque nouvelle couleur.
OnSelect	TNoti-fyEvent	Une couleur est choisie : un clic sur un échantillon, le relâchement de la souris après un glissement, Entrée, ou un texte hexadécimal validé par Entrée ou par la perte du focus.

Clavier

- En VCL, le contrôle a deux arrêts de focus. Tab parcourt les parties, passe ensuite à l'edit hexadécimal, et le Tab suivant le quitte. En lecture seule, le contrôle est un seul arrêt.
- En FireMonkey, le contrôle est un seul arrêt de tabulation avec deux arrêts internes : Ctrl+flèches passe des parties à l'edit hexadécimal.
- Dans les échantillons, les flèches, Début et Fin déplacent le focus, et Entrée ou Espace choisit l'échantillon. Sur le carré, Gauche et Droite modifient la saturation et Haut et Bas la valeur de 1 %. Sur les barres, les flèches avancent d'un degré ou de 1 %. Maj fait des pas dix fois plus grands.
- Un texte hexadécimal qui n'est pas une couleur réaffiche la couleur, et Échap la restaure.

Exemple : Couleurs de marque pour une étiquette

```
uses
  SysUtils, Classes, Controls, Graphics, ExtCtrls,
  sgcEditors_Types, sgcEditors_VCL_Color;

type
  TLabelColorEvents = class
  public
    Picker: TsgcVCLColorPicker;
    Preview: TPanel;
    procedure ColorSelect(Sender: TObject);
  end;

procedure TLabelColorEvents.ColorSelect(Sender: TObject);
begin
  // a swatch click, Enter, the hex edit or the end of a drag
  Preview.Color := Picker.SelectedVCLColor;
end;

procedure SetupLabelColor(aEvents: TLabelColorEvents);
begin
  // six brand colours in one row and the hex edit, no square and no alpha
  aEvents.Picker.Parts := [ecpSwatches, ecpHex];
```

```
  aEvents.Picker.Swatches.Clear;
  aEvents.Picker.Swatches.Add('#1E88E5');
  aEvents.Picker.Swatches.Add('#43A047');
  aEvents.Picker.Swatches.Add('#FDD835');
  aEvents.Picker.Swatches.Add('#FB8C00');
  aEvents.Picker.Swatches.Add('#E53935');
  aEvents.Picker.Swatches.Add('#8E24AA');
  aEvents.Picker.SwatchColumns := 6;
  aEvents.Picker.SwatchSize := 28;
  aEvents.Picker.SelectedVCLColor := clNavy;
  aEvents.Picker.OnSelect := aEvents.ColorSelect;
end;
```

Exemple : Une couleur de remplissage dans FireMonkey

```
uses
  System.SysUtils, System.Classes, System.UITypes, FMX.Objects,
  sgcEditors_Types, sgcEditors_FMX_Color;

type
  TFillEvents = class
  public
    Picker: TsgcFMXColorPicker;
    Preview: TRectangle;
    procedure ColorChange(Sender: TObject);
  end;

procedure TFillEvents.ColorChange(Sender: TObject);
begin
  // SelectedColor is ARGB, the layout of TAlphaColor; OnChange follows
  // a drag on the square and the bars
  Preview.Fill.Color := TAlphaColor(Picker.SelectedColor);
end;

procedure SetupFill(aEvents: TFillEvents);
begin
  aEvents.Picker.Parts := [ecpSwatches, ecpHSV, ecpAlpha, ecpHex];
  aEvents.Picker.SelectedColor := $802A78D6;
  aEvents.Picker.OnChange := aEvents.ColorChange;
end;
```

Voir aussi

- Token edit
- Thèmes
- Éditions

Token edit

Classes : TsgcVCLTokenEdit (unit sgcEditors_VCL_Token) et TsgcFMXTokenEdit (unit sgcEditors_FMX_Token)
Le token edit contient une liste de jetons, comme des étiquettes ou des adresses e-mail, dessinés sous forme de chips avec un bouton de suppression. Les chips passent à la ligne de gauche à droite, et un edit natif après le dernier chip reçoit le jeton suivant, de sorte que l'IME, le presse-papiers et l'annulation fonctionnent. Un délimiteur, Entrée, Tab et la perte du focus transforment le texte saisi en chip, et coller a, b, c produit trois chips. Chaque jeton est nettoyé de ses espaces, vérifié par OnValidateToken, vérifié contre les doublons et compté par rapport à MaxTokens. Lorsque les chips ne tiennent pas, le contrôle défile verticalement.
La liste de saisie semi-automatique requiert l'édition Registered ou Trial. Dans Community, les propriétés sont conservées et la liste ne s'ouvre jamais. Voir Éditions.
La classe VCL descend de TsgcVCLCustomEditor et la classe FireMonkey de TsgcFMXCustomEditor. Les deux publient les propriétés et les événements du tableau ci-dessous, plus les habituels de leur framework, comme Align, Anchors, Enabled, Hint, TabOrder, TabStop, OnEnter, OnExit et les événements de clavier et de souris. Voir Contrôles.

Propriétés

Propriété	Type	Description
Tokens	TStrings	Les jetons, dans l'ordre. Modifier la liste modifie les chips.
Delimiters	string	Les caractères qui terminent un jeton. Par défaut ,; et les sauts de ligne séparent toujours. Vide, il ne laisse que Entrée et les sauts de ligne.
AllowDuplicates	Boolean	Ajoute un jeton qui est déjà dans la liste. Par défaut False.
CaseSensitive	Boolean	Compare les jetons en tenant compte de la casse pour chercher les doublons. Par défaut False.
MaxTokens	Integer	Le plus grand nombre de jetons que la liste accepte. 0 (par défaut) signifie aucune limite. Un texte au-delà de la limite reste dans l'edit.
InvalidTokens	TsgcEditorInvalidTokens	etiReject (par défaut) garde un texte refusé dans l'edit. etiMark l'ajoute comme chip marqué ; TokenModel.IsTokenInvalid(aIndex) indique lequel.
TextHint	string	L'indication grise de l'edit vide.
AutoCompleteItems	TStrings	Les candidats de la liste de saisie semi-automatique. Éditions Registered et Trial.
AutoCompleteMatch	TsgcEditorMatchMode	emmStartsWith (par défaut) ou emmContains.
AutoCompleteMinChars	Integer	Les caractères que l'utilisateur saisit avant que la liste s'ouvre. Par défaut 1.
AutoCompleteMaxItems	Integer	Le plus grand nombre de lignes que la liste affiche. 0 (par défaut) signifie aucune limite.
DroppedDown	Boolean	Public. True tant que la liste de saisie semi-automatique est ouverte.

Theme	TsgcUI-Theme	utLight (par défaut) ou utDark. Voir Thèmes.
AccentColor	TsgcUIColor	La couleur d'accentuation ARGB de la sélection, du remplissage et de l'anneau de focus. 0 (par défaut) utilise la couleur d'accentuation du thème.
TextStyle	TsgcUIText-Style	Size, Bold et Color du texte. Un Color de 0 suit le thème.
ReadOnly	Boolean	La valeur est affichée mais l'utilisateur ne peut pas la modifier. Par défaut False.
Accessible-Name	string	Le nom qu'un lecteur d'écran annonce avant la valeur. Voir Clavier et accessi-bilité.
TokenModel	TsgcToken-Model	Public, en lecture seule. Le modèle du contrôle. Model renvoie le même objet qu'un TsgcEditorModel.

Méthodes

Méthode	Ren-voie	Description
AddToken(aText)	Boo-lean	Ajoute un jeton sans le scinder, en passant par le nettoyage des es-paces, OnValidateToken, la vérification des doublons et MaxTokens. True lorsqu'il a été ajouté.
RemoveToken(aIndex)	Boo-lean	Supprime le jeton à aIndex. True lorsqu'il a été supprimé.
Clear		Supprime tous les jetons et le texte saisi.
SaveToPNG(aStream ou aFileName, aScale = 1)		Dessine le contrôle à sa taille actuelle dans un PNG. aScale multiplie la taille. Voir Sortie sans interface.
SaveToSVG(aStream ou aFileName, aScale = 1)		La même chose dans un SVG.

Événements

Événement	Type	Description
OnChange	TNotifyEvent	La liste des jetons a changé.
OnValidateTo-ken	TsgcToken-Valida-teEvent	(Sender; var aText: string; var aValid: Boolean). aText arrive nettoyé de ses espaces et peut être modifié, par exemple pour le normaliser. Mettez aValid à False pour refuser le jeton, ou pour le marquer avec etiMark.
OnTokenAd-ded, OnToken-Removed	TsgcToke-nEvent	(Sender; aIndex: Integer; const aText: string). Un chip a été ajouté ou supprimé.
OnAutoCom-plete	TsgcToke-nAutoCom-pleteEvent	(Sender; const aText: string; aItems: TStrings). aItems sont les candidats de la liste. Modifiez-les pour le texte saisi, par exemple depuis une base de données, et la liste les filtre ensuite. Éditions Registered et Trial.

Clavier

Retour arrière ou Gauche dans un edit vide met le focus sur le dernier chip, et un second Retour arrière le supprime. Gauche et Droite parcourent les chips, Droite sur le dernier revient à l'edit, Suppr supprime le chip ayant le focus, et Échap ou la saisie revient à l'edit. Tant que la liste de saisie semi-automatique est ouverte, Haut, Bas, Page préc. et Page suiv. s'y déplacent, Entrée choisit la ligne en surbrillance et Échap la ferme.

Exemple : Des destinataires d'e-mail

```
uses
  SysUtils, Classes, Controls,
  sgcEditors_Types, sgcEditors_VCL_Token;

type
  TRecipientEvents = class
  public
    procedure ValidateToken(Sender: TObject; var aText: string;
      var aValid: Boolean);
  end;

procedure TRecipientEvents.ValidateToken(Sender: TObject; var aText: string;
  var aValid: Boolean);
begin
  // aText comes trimmed; store addresses in lower case and mark the ones
  // without an @
  aText := LowerCase(aText);
  aValid := Pos('@', aText) > 1;
end;

procedure SetupRecipients(aTokens: TsgcVCLTokenEdit;
  aEvents: TRecipientEvents);
begin
  aTokens.Delimiters := ',; ';
  aTokens.MaxTokens := 10;
  aTokens.InvalidTokens := etiMark;
  aTokens.TextHint := 'Add recipients';
  aTokens.OnValidateToken := aEvents.ValidateToken;
  // the auto-complete list opens in the Registered and Trial editions
  aTokens.AutoCompleteItems.Add('anna@example.com');
  aTokens.AutoCompleteItems.Add('ben@example.com');
  aTokens.AutoCompleteItems.Add('sales@example.com');
  aTokens.AutoCompleteMatch := emmContains;
  aTokens.AutoCompleteMinChars := 2;
  aTokens.AddToken('team@example.com');
end;

function InvalidCount(aTokens: TsgcVCLTokenEdit): Integer;
var
  i: Integer;
begin
  Result := 0;
  for i := 0 to aTokens.Tokens.Count - 1 do
    if aTokens.TokenModel.IsTokenInvalid(i) then
      Inc(Result);
end;
```

Voir aussi

- Saisie de texte et IME
- Clavier et accessibilité
- Éditions

Évaluation

Classes : TsgcVCLRating (unit sgcEditors_VCL_Value) et TsgcFMXRating (unit sgcEditors_FMX_Value)
L'évaluation est une rangée de glyphes dessinée sur mesure, pouvant recevoir le focus : étoiles, coeurs, cercles ou un glyphe de votre choix. Un clic choisit la valeur sous la souris, et le survol prévisualise une valeur sans la modifier. Precision arrondit la valeur à des glyphes entiers ou à des moitiés, ou la garde exacte pour une moyenne en lecture seule comme 3,7, dessinée avec un glyphe partiellement rempli.
La classe VCL descend de TsgcVCLCustomEditor et la classe FireMonkey de TsgcFMXCustomEditor. Les deux publient les propriétés et les événements du tableau ci-dessous, plus les habituels de leur framework, comme Align, Anchors, Enabled, Hint, TabOrder, TabStop, OnEnter, OnExit et les événements de clavier et de souris. Voir Contrôles.

Propriétés

Propriété	Type	Description
Value	Double	La valeur, de 0 à Count, arrondie selon Precision.
Count	Integer	Le nombre de glyphes, de 1 à 20. Par défaut 5.
Precision	TsgcEditorRatingPrecision	erpWhole (par défaut) arrondit à des glyphes entiers, erpHalf à des moitiés, et erpExact n'arrondit pas. Définissez Count et Precision avant Value.
Glyph	TsgcEditorRatingGlyph	ergStar (par défaut), ergHeart, ergCircle ou ergCustom, qui dessine par OnDrawGlyph.
GlyphSize	Double	La taille d'un glyphe en pixels logiques. 0 (par défaut) est la plus grande taille qui tient dans le contrôle.
Spacing	Double	Les pixels logiques entre deux glyphes. Par défaut 4.
FillColor	TsgcUIColor	La couleur des glyphes remplis. 0 (par défaut) est l'ambre de la palette du thème.
EmptyColor	TsgcUIColor	La couleur des glyphes vides. 0 (par défaut) est la couleur de bordure du thème.
AllowClear	Boolean	Un clic sur la valeur actuelle la remet à 0, et Début va à 0. Par défaut False, la plus petite valeur que l'utilisateur puisse choisir est alors un pas.
Theme	TsgcUITheme	utLight (par défaut) ou utDark. Voir Thèmes.
AccentColor	TsgcUIColor	La couleur d'accentuation ARGB de la sélection, du remplissage et de l'anneau de focus. 0 (par défaut) utilise la couleur d'accentuation du thème.
TextStyle	TsgcUITextStyle	Size, Bold et Color du texte. Un Color de 0 suit le thème.
ReadOnly	Boolean	La valeur est affichée mais l'utilisateur ne peut pas la modifier. Par défaut False.
AccessibleName	string	Le nom qu'un lecteur d'écran annonce avant la valeur. Voir Clavier et accessibilité.
RatingModel	TsgcRatingModel	Public, en lecture seule. Le modèle du contrôle. Model renvoie le même objet qu'un TsgcEditorModel.

Méthodes

Méthode	Ren-voie	Description
SaveToPNG(aStream ou aFile-Name, aScale = 1)		Dessine le contrôle à sa taille actuelle dans un PNG. aScale multiplie la taille. Voir Sortie sans interface.
SaveToSVG(aStream ou aFile-Name, aScale = 1)		La même chose dans un SVG.

Événements

Événe-ment	Type	Description
OnChange	TNotifyEvent	Valeur modifiée.
OnHover	TsgcRatingHo-verEvent	(Sender; aValue: Double). La valeur sous la souris pendant l'aperçu, et -1 lorsque l'aperçu se termine.
OnDraw-Glyph	TsgcRating-DrawGly-phEvent	(Sender; aCanvas: TsgcUICanvas; const aRect: TsgcUIRect; aFill: Double). Dessine un glyphe de ergCustom dans aRect, en pixels de l'appareil. Il est appelé avec aFill à 0 pour le glyphe vide, puis de nouveau avec la partie remplie, de 0 à 1 depuis la gauche.

Clavier

Droite et Haut ajoutent un pas de Precision, Gauche et Bas en retirent un. Début va à la plus petite valeur, Fin à Count, et les touches numériques définissent une valeur entière.

Exemple : Une note sur dix avec un glyphe de votre choix

```
uses
  SysUtils, Classes, Controls, StdCtrls,
  sgcUICore_Canvas, sgcUICore_Theme, sgcEditors_Types, sgcEditors_VCL_Value;

type
  TScoreEvents = class
  public
    Status: TLabel;
    procedure DrawGlyph(Sender: TObject; aCanvas: TsgcUICanvas;
      const aRect: TsgcUIRect; aFill: Double);
    procedure Hover(Sender: TObject; aValue: Double);
  end;

procedure TScoreEvents.DrawGlyph(Sender: TObject; aCanvas: TsgcUICanvas;
  const aRect: TsgcUIRect; aFill: Double);
var
  vRight: Double;
  vColor: TsgcUIColor;
begin
  // called with aFill 0 for the empty glyph (content layer), then with the
  // fill, 0..1, for the filled part (overlay); aRect is in device pixels
  vRight := aRect.Right;
  vColor := $FFD0D5DD;
  if aFill > 0 then
  begin
```

```
    vRight := aRect.Left + (aRect.Right - aRect.Left) * aFill;
    vColor := $FF2E7D32;
  end;
  aCanvas.BeginPath;
  aCanvas.RoundRect(aRect.Left, aRect.Top, vRight, aRect.Bottom, 3, 3);
  aCanvas.FillPath(vColor);
end;

procedure TScoreEvents.Hover(Sender: TObject; aValue: Double);
begin
  // the value under the mouse, -1 when the preview ends
  if aValue < 0 then
    Status.Caption := ''
  else
    Status.Caption := FormatFloat('0.#', aValue) + ' of 10';
end;

procedure SetupScore(aRating: TsgcVCLRating; aEvents: TScoreEvents);
begin
  // Count and Precision before Value
  aRating.Count := 10;
  aRating.Precision := erpHalf;
  aRating.Value := 7.5;
  aRating.Glyph := ergCustom;
  aRating.GlyphSize := 14;
  aRating.Spacing := 2;
  aRating.AllowClear := True;
  aRating.OnDrawGlyph := aEvents.DrawGlyph;
  aRating.OnHover := aEvents.Hover;
end;
```

Voir aussi

- Évaluation DB
- Curseur de plage
- Thèmes

Spin edit

Classes : TsgcVCLSpinEdit (unit sgcEditors_VCL_Value) et TsgcFMXSpinEdit (unit sgcEditors_FMX_Value)
Le spin edit est un edit natif pour un nombre avec des boutons haut et bas. La valeur est un Double : esvInteger l'arrondit à des nombres entiers et esvFloat à Decimals. Units est un suffixe atténué, comme kg ou %, dessiné après le texte et qui n'en fait jamais partie. Un bouton maintenu enfoncé se répète, plus vite plus il est maintenu longtemps. L'analyse est tolérante : les espaces et le séparateur de milliers sont ignorés, et . comme , sont acceptés comme séparateur décimal.
La classe VCL descend de TsgcVCLCustomEditor et la classe FireMonkey de TsgcFMXCustomEditor. Les deux publient les propriétés et les événements du tableau ci-dessous, plus les habituels de leur framework, comme Align, Anchors, Enabled, Hint, TabOrder, TabStop, OnEnter, OnExit et les événements de clavier et de souris. Voir Contrôles.

Propriétés

Propriété	Type	Description
Value	Double	La valeur. Définissez d'abord ValueType, MinValue et MaxValue, sinon Value est maintenue dans les anciennes limites.
MinValue, MaxValue	Double	Les limites. Des valeurs égales, comme les valeurs par défaut 0 et 0, signifient aucune limite.
Increment	Double	Le pas des boutons, de Haut, de Bas et de la molette. Par défaut 1.
LargeIncrement	Double	Le pas de Page préc. et Page suiv. Par défaut 10.
ValueType	TsgcEditorSpinValueType	esvInteger (par défaut) ou esvFloat.
Decimals	Integer	Les décimales d'une valeur esvFloat. Par défaut 2.
DisplayFormat	string	Un masque FormatFloat pour le texte, comme #,##0.00. Vide (par défaut), il écrit la valeur avec Decimals.
Units	string	Un suffixe atténué après le texte, qui ne fait pas partie de la valeur.
Buttons	TsgcEditorSpinButtons	esbRight (par défaut) empile haut et bas à droite, esbSides place le moins à gauche et le plus à droite, esbNone n'a pas de boutons.
Wrap	Boolean	Un pas au-delà d'une limite s'arrête d'abord sur la limite, et le pas suivant va à l'autre extrémité. Par défaut False.
AllowEmpty	Boolean	Permet à l'utilisateur de supprimer le texte, ce qui vide la valeur. Par défaut False.
IsEmpty	Boolean	Public, en lecture seule. True tant que la valeur est vide.
Theme	TsgcUITheme	utLight (par défaut) ou utDark. Voir Thèmes.
AccentColor	TsgcUIColor	La couleur d'accentuation ARGB de la sélection, du remplissage et de l'anneau de focus. 0 (par défaut) utilise la couleur d'accentuation du thème.
TextStyle	TsgcUITextStyle	Size, Bold et Color du texte. Un Color de 0 suit le thème.

ReadOnly	Boolean	La valeur est affichée mais l'utilisateur ne peut pas la modifier. Par défaut False.
Accessible-Name	string	Le nom qu'un lecteur d'écran annonce avant la valeur. Voir Clavier et accessibilité.
FormatSettings	TFormatSettings	Public. Les noms, les formats et les séparateurs que le contrôle utilise. Il démarre avec les paramètres des paramètres régionaux de l'utilisateur. Voir Localisation.
SpinModel	TsgcSpinModel	Public, en lecture seule. Le modèle du contrôle. Model renvoie le même objet qu'un TsgcEditorModel.

Méthodes

Méthode	Renvoie	Description
Clear		Vide la valeur lorsque AllowEmpty vaut True. Sinon, il ne fait rien.
SaveToPNG(aStream ou aFileName, aScale = 1)		Dessine le contrôle à sa taille actuelle dans un PNG. aScale multiplie la taille. Voir Sortie sans interface.
SaveToSVG(aStream ou aFileName, aScale = 1)		La même chose dans un SVG.

Événements

Événement	Type	Description
OnChange	TNotifyEvent	Une nouvelle valeur : validée, avancée ou définie dans le code.
OnParseError	TsgcSpinParseErrorEvent	(Sender; const aText: string; var aValue: Double; var aAccept: Boolean). Le texte validé n'est pas un nombre. Mettez aValue et aAccept à True pour prendre une valeur. Sinon, la dernière valeur revient.
OnValidate	TsgcSpinValidateEvent	(Sender; var aValue: Double; var aAccept: Boolean). Appelé avant qu'une nouvelle valeur soit définie. aAccept arrive à True. Modifiez aValue pour l'ajuster, ou mettez aAccept à False pour garder l'ancienne valeur.

Clavier et souris

Haut et Bas avancent d'un Increment, Page préc. et Page suiv. d'un LargeIncrement, et la molette d'un Increment par cran. Chacune de ces actions valide d'abord le texte saisi. Entrée et la perte du focus valident le texte, et Échap le restaure. Un bouton maintenu enfoncé se répète : un Increment par répétition, puis 2 à partir de la 10e répétition, 5 à partir de la 20e et 10 à partir de la 30e.

Exemple : Un angle en demi-degrés

```
uses
  SysUtils, Classes, Controls,
  sgcEditors_Types, sgcEditors_VCL_Value;

type
```

```
  TAngleEvents = class
  public
    procedure Validate(Sender: TObject; var aValue: Double;
      var aAccept: Boolean);
  end;

procedure TAngleEvents.Validate(Sender: TObject; var aValue: Double;
  var aAccept: Boolean);
begin
  // aAccept comes in True; a typed angle snaps to half degrees
  aValue := Round(aValue * 2) / 2;
end;

procedure SetupAngle(aSpin: TsgcVCLSpinEdit; aEvents: TAngleEvents);
begin
  // the type and the limits before Value
  aSpin.ValueType := esvFloat;
  aSpin.Decimals := 1;
  aSpin.MinValue := 0;
  aSpin.MaxValue := 359.5;
  aSpin.Increment := 0.5;
  aSpin.LargeIncrement := 15;
  // a step past a limit stops on it, the next step goes to the other end
  aSpin.Wrap := True;
  aSpin.Units := 'deg';
  aSpin.Buttons := esbSides;
  aSpin.Value := 90;
  aSpin.OnValidate := aEvents.Validate;
end;
```

Voir aussi

- Spin edit DB
- Curseur de plage
- Localisation

Sélecteur de date DB

Classes : TsgcVCLDBDatePicker (unit sgcEditors_VCL_DB) et TsgcFMXDBDatePicker (unit sgcEditors_FMX_DB)
Le sélecteur de date DB lie la date d'un sélecteur de date à un champ de date ou de date et d'heure. Il descend de TsgcVCLCustomDatePicker et TsgcFMXCustomDatePicker et publie les propriétés du sélecteur de date sauf Date : la valeur est le champ. Les autres propriétés, méthodes et événements publiés sont ceux du sélecteur de date.
Les contrôles data-aware font partie des éditions Registered et Trial. Leurs units sont vides dans Community. Ils se lient par un TsgcEditorFieldLink, qui n'utilise que l'unit DB, de sorte que les contrôles FireMonkey n'ont pas besoin de LiveBindings.

Propriétés

Propriété	Type	Description
DataSource	TDataSource	La source de données du dataset.
DataField	string	Le nom du champ auquel la valeur est liée.
AllowEmpty	Boolean	Par défaut True, de sorte qu'un champ null affiche un texte vide et que supprimer le texte écrit null.
DisplayFormat, MinDate, MaxDate, DisabledDates, DisabledWeekDays, FirstDayOfWeek, ShowWeekNumbers, TextHint		Comme dans le sélecteur de date.
ReadOnly	Boolean	True garde le contrôle en lecture seule. Il est aussi en lecture seule tant que le champ ne peut pas être modifié : pas de dataset, un dataset ou un champ en lecture seule, ou un champ calculé. Par défaut False.
Field	TField	Public, en lecture seule. Le champ lié, nil tant que le lien n'est pas actif.

Méthodes

Méthode	Renvoie	Description
SaveToPNG, SaveToSVG		Comme dans le sélecteur de date. Les autres méthodes du contrôle de base s'appliquent aussi.

Événements

Événement	Type	Description
OnChange	TNotifyEvent	La valeur a changé, par l'utilisateur, dans le code ou parce que l'enregistrement a bougé.
OnDropDown, OnCloseUp, OnParseError, OnIsDateDisabled		Comme dans le sélecteur de date.

Comment le champ est modifié

- La valeur suit l'enregistrement. Un champ null affiche un texte vide.
- Une modification de l'utilisateur met le dataset en mode édition et marque la valeur comme modifiée. Définir la valeur dans le code est aussi une modification. Lorsque le dataset ne peut pas passer en mode édition, la valeur du champ revient.
- Entrée et la perte du focus écrivent le champ, et le dataset écrit une valeur en attente avant d'enregistrer. Un champ qui refuse la valeur lève une exception et le focus reste dans le contrôle, comme avec TDBEdit.
- Échap restaure la valeur du champ.
- Le sélecteur de date conserve la partie heure d'un TDateTimeField et ne modifie que sa date.

Exemple : Une date de naissance

```
uses
  SysUtils, Classes, Controls, DB,
  sgcEditors_VCL_DB;

procedure BindBirthDate(aPicker: TsgcVCLDBDatePicker; aSource: TDataSource);
begin
  aPicker.DataSource := aSource;
  aPicker.DataField := 'BirthDate';
  aPicker.DisplayFormat := 'dd/mm/yyyy';
  aPicker.MaxDate := Date;
  // a null field shows an empty text with this hint
  aPicker.TextHint := 'Unknown';
end;

function BirthDateKnown(aPicker: TsgcVCLDBDatePicker): Boolean;
begin
  // Field is nil while the link is not active
  Result := (aPicker.Field <> nil) and not aPicker.Field.IsNull;
end;
```

Voir aussi

- Sélecteur de date
- Sélecteur d'heure DB
- Démarrage rapide data-aware
- TsgcEditorFieldLink

Sélecteur d'heure DB

Classes : TsgcVCLDBTimePicker (unit sgcEditors_VCL_DB) et TsgcFMXDBTimePicker (unit sgcEditors_FMX_DB)
Le sélecteur d'heure DB lie l'heure d'un sélecteur d'heure à un champ d'heure ou de date et d'heure. Il descend de TsgcVCLCustomDatePicker et TsgcFMXCustomDatePicker et publie les propriétés du sélecteur d'heure sauf Time : la valeur est le champ. Les autres propriétés, méthodes et événements publiés sont ceux du sélecteur d'heure.
Les contrôles data-aware font partie des éditions Registered et Trial. Leurs units sont vides dans Community. Ils se lient par un TsgcEditorFieldLink, qui n'utilise que l'unit DB, de sorte que les contrôles FireMonkey n'ont pas besoin de LiveBindings.

Propriétés

Propriété	Type	Description
DataSource	TDataSource	La source de données du dataset.
DataField	string	Le nom du champ auquel la valeur est liée.
AllowEmpty	Boolean	Par défaut True, de sorte qu'un champ null affiche un texte vide et que supprimer le texte écrit null.
DisplayFormat, MinTime, MaxTime, ListInterval, TextHint		Comme dans le sélecteur d'heure.
ReadOnly	Boolean	True garde le contrôle en lecture seule. Il est aussi en lecture seule tant que le champ ne peut pas être modifié : pas de dataset, un dataset ou un champ en lecture seule, ou un champ calculé. Par défaut False.
Field	TField	Public, en lecture seule. Le champ lié, nil tant que le lien n'est pas actif.

Méthodes

Méthode	Renvoie	Description
SaveToPNG, SaveToSVG		Comme dans le sélecteur d'heure. Les autres méthodes du contrôle de base s'appliquent aussi.

Événements

Événement	Type	Description
OnChange	TNotifyEvent	La valeur a changé, par l'utilisateur, dans le code ou parce que l'enregistrement a bougé.
OnDropDown, OnCloseUp, OnParseError		Comme dans le sélecteur d'heure.

Comment le champ est modifié

- La valeur suit l'enregistrement. Un champ null affiche un texte vide.
- Une modification de l'utilisateur met le dataset en mode édition et marque la valeur comme modifiée. Définir la valeur dans le code est aussi une modification. Lorsque le dataset ne peut pas passer en mode édition, la valeur du champ revient.
- Entrée et la perte du focus écrivent le champ, et le dataset écrit une valeur en attente avant d'enregistrer. Un champ qui refuse la valeur lève une exception et le focus reste dans le contrôle, comme avec TDBEdit.
- Échap restaure la valeur du champ.
- Le sélecteur d'heure conserve la partie date d'un TDateTimeField et ne modifie que son heure.

Exemple : L'heure de début d'une réservation

```
uses
  SysUtils, Classes, Controls, DB,
  sgcEditors_VCL_DB;

procedure BindStartTime(aPicker: TsgcVCLDBTimePicker; aSource: TDataSource);
begin
  // a TDateTimeField keeps its date part, the picker edits the time
  aPicker.DataSource := aSource;
  aPicker.DataField := 'StartsAt';
  aPicker.DisplayFormat := 'hh:nn';
  aPicker.ListInterval := 15;
  aPicker.MinTime := EncodeTime(7, 0, 0, 0);
  aPicker.MaxTime := EncodeTime(22, 0, 0, 0);
end;
```

Voir aussi

- Sélecteur d'heure
- Sélecteur de date DB
- Démarrage rapide data-aware
- TsgcEditorFieldLink

Spin edit DB

Classes : TsgcVCLDBSpinEdit (unit sgcEditors_VCL_DB) et TsgcFMXDBSpinEdit (unit sgcEditors_FMX_DB)
Le spin edit DB lie la valeur d'un spin edit à un champ numérique. Il descend de TsgcVCLSpinEdit et TsgcFMXSpinEdit, et conserve donc les limites, les pas, les unités et les boutons. Un champ entier le fait passer à esvInteger. Les autres propriétés, méthodes et événements publiés sont ceux du spin edit.
Les contrôles data-aware font partie des éditions Registered et Trial. Leurs units sont vides dans Community. Ils se lient par un TsgcEditorFieldLink, qui n'utilise que l'unit DB, de sorte que les contrôles FireMonkey n'ont pas besoin de LiveBindings.

Propriétés

Propriété	Type	Description
DataSource	TDataSource	La source de données du dataset.
DataField	string	Le nom du champ auquel la valeur est liée.
AllowEmpty	Boolean	Par défaut True, de sorte qu'un champ null affiche un texte vide et que supprimer le texte écrit null.
Value	Double	La valeur du champ. Elle n'est pas enregistrée dans le formulaire.
ReadOnly	Boolean	True garde le contrôle en lecture seule. Il est aussi en lecture seule tant que le champ ne peut pas être modifié : pas de dataset, un dataset ou un champ en lecture seule, ou un champ calculé. Par défaut False.
Field	TField	Public, en lecture seule. Le champ lié, nil tant que le lien n'est pas actif.

Méthodes

Méthode	Renvoie	Description
SaveToPNG, SaveToSVG		Comme dans le spin edit. Les autres méthodes du contrôle de base s'appliquent aussi.

Événements

Événement	Type	Description
OnChange	TNotifyEvent	La valeur a changé, par l'utilisateur, dans le code ou parce que l'enregistrement a bougé.
OnParseError, OnValidate		Comme dans le spin edit. OnValidate s'exécute avant que la valeur aille dans le champ.

Comment le champ est modifié

- La valeur suit l'enregistrement. Un champ null affiche un texte vide.

- Une modification de l'utilisateur met le dataset en mode édition et marque la valeur comme modifiée. Définir la valeur dans le code est aussi une modification. Lorsque le dataset ne peut pas passer en mode édition, la valeur du champ revient.
- Entrée et la perte du focus écrivent le champ, et le dataset écrit une valeur en attente avant d'enregistrer. Un champ qui refuse la valeur lève une exception et le focus reste dans le contrôle, comme avec TDBEdit.
- Échap restaure la valeur du champ.
- Un champ entier fait passer ValueType à esvInteger.

Exemple : Une quantité

```
uses
  SysUtils, Classes, Controls, DB,
  sgcEditors_VCL_DB;

procedure BindQuantity(aSpin: TsgcVCLDBSpinEdit; aSource: TDataSource);
begin
  // an integer field switches the spin edit to esvInteger
  aSpin.MinValue := 1;
  aSpin.MaxValue := 999;
  aSpin.LargeIncrement := 12;
  aSpin.Units := 'pcs';
  aSpin.DataSource := aSource;
  aSpin.DataField := 'Quantity';
  // AllowEmpty is True by default, so a null field shows an empty text
  aSpin.AllowEmpty := True;
end;
```

Exemple : Une remise dans FireMonkey

```
uses
  System.SysUtils, System.Classes, Data.DB,
  sgcEditors_FMX_DB;

procedure BindDiscount(aSpin: TsgcFMXDBSpinEdit; aSource: TDataSource);
begin
  // the FireMonkey twin binds without LiveBindings
  aSpin.DataSource := aSource;
  aSpin.DataField := 'Discount';
  aSpin.MinValue := 0;
  aSpin.MaxValue := 100;
  aSpin.Units := '%';
end;
```

Voir aussi

- Spin edit
- Évaluation DB
- Démarrage rapide data-aware
- TsgcEditorFieldLink

Évaluation DB

Classes : TsgcVCLDBRating (unit sgcEditors_VCL_DB) et TsgcFMXDBRating (unit sgcEditors_FMX_DB)
L'évaluation DB lie la valeur d'une évaluation à un champ numérique. Elle descend de TsgcVCLRating et TsgcFMXRating, et conserve donc Count, Precision, les glyphes et les couleurs. Avec Precision erpExact et ReadOnly à True, elle affiche une moyenne, comme 3,7, issue d'un champ calculé ou agrégé. Les autres propriétés, méthodes et événements publiés sont ceux de l'évaluation.
Les contrôles data-aware font partie des éditions Registered et Trial. Leurs units sont vides dans Community. Ils se lient par un TsgcEditorFieldLink, qui n'utilise que l'unit DB, de sorte que les contrôles FireMonkey n'ont pas besoin de LiveBindings.

Propriétés

Propriété	Type	Description
DataSource	TDataSource	La source de données du dataset.
DataField	string	Le nom du champ auquel la valeur est liée.
Value	Double	La valeur du champ. Elle n'est pas enregistrée dans le formulaire.
ReadOnly	Boolean	True garde le contrôle en lecture seule. Il est aussi en lecture seule tant que le champ ne peut pas être modifié : pas de dataset, un dataset ou un champ en lecture seule, ou un champ calculé. Par défaut False.
Field	TField	Public, en lecture seule. Le champ lié, nil tant que le lien n'est pas actif.

Méthodes

Méthode	Renvoie	Description
SaveToPNG, SaveToSVG		Comme dans l'évaluation. Les autres méthodes du contrôle de base s'appliquent aussi.

Événements

Événement	Type	Description
OnChange	TNotifyEvent	La valeur a changé, par l'utilisateur, dans le code ou parce que l'enregistrement a bougé.
OnHover, OnDrawGlyph		Comme dans l'évaluation.

Comment le champ est modifié

- La valeur suit l'enregistrement. Un champ null ne remplit aucun glyphe.
- Une modification de l'utilisateur met le dataset en mode édition et marque la valeur comme modifiée. Définir la valeur dans le code est aussi une modification. Lorsque le dataset ne peut pas passer en mode édition, la valeur du champ revient.

- Entrée et la perte du focus écrivent le champ, et le dataset écrit une valeur en attente avant d'enregistrer. Un champ qui refuse la valeur lève une exception et le focus reste dans le contrôle, comme avec TDBEdit.
- Échap restaure la valeur du champ.
- Avec AllowClear, un clic sur la valeur actuelle écrit 0 dans le champ.

Exemple : Des étoiles et une moyenne

```
uses
  SysUtils, Classes, Controls, DB,
  sgcEditors_Types, sgcEditors_VCL_DB;

procedure BindStars(aRating: TsgcVCLDBRating; aSource: TDataSource);
begin
  aRating.Count := 5;
  aRating.Precision := erpHalf;
  // a click on the current value writes 0 to the field
  aRating.AllowClear := True;
  aRating.DataSource := aSource;
  aRating.DataField := 'Stars';
end;

procedure ShowAverage(aRating: TsgcVCLDBRating; aSource: TDataSource);
begin
  // a read-only average such as 3.7: erpExact is not rounded
  aRating.Precision := erpExact;
  aRating.ReadOnly := True;
  aRating.DataSource := aSource;
  aRating.DataField := 'AvgStars';
end;
```

Voir aussi

- Évaluation
- Spin edit DB
- Démarrage rapide data-aware
- TsgcEditorFieldLink

Curseur de plage DB

Classes : TsgcVCLDBRangeSlider (unit sgcEditors_VCL_DB) et TsgcFMXDBRangeSlider (unit sgcEditors_FMX_DB)
Le curseur de plage DB lie Value d'un curseur de plage à DataField et, dans RangeMode, EndValue à EndDataField, de sorte qu'un seul contrôle modifie une plage stockée dans deux champs. Il descend de TsgcVCLRangeSlider et TsgcFMXRangeSlider. Les autres propriétés, méthodes et événements publiés sont ceux du curseur de plage.
Les contrôles data-aware font partie des éditions Registered et Trial. Leurs units sont vides dans Community. Ils se lient par un TsgcEditorFieldLink, qui n'utilise que l'unit DB, de sorte que les contrôles FireMonkey n'ont pas besoin de LiveBindings.

Propriétés

Propriété	Type	Description
DataSource	TDataSource	La source de données du dataset.
DataField	string	Le nom du champ auquel la valeur est liée.
EndDataField	string	Le champ de EndValue dans RangeMode.
Value, EndValue	Double	Les valeurs des deux champs. Elles ne sont pas enregistrées dans le formulaire.
EndField	TField	Public, en lecture seule. Le champ de EndValue, nil tant que le lien n'est pas actif.
ReadOnly	Boolean	True garde le contrôle en lecture seule. Il est aussi en lecture seule tant que le champ ne peut pas être modifié : pas de dataset, un dataset ou un champ en lecture seule, ou un champ calculé. Par défaut False.
Field	TField	Public, en lecture seule. Le champ lié, nil tant que le lien n'est pas actif.

Méthodes

Méthode	Renvoie	Description
SaveToPNG, SaveToSVG		Comme dans le curseur de plage. Les autres méthodes du contrôle de base s'appliquent aussi.

Événements

Événement	Type	Description
OnChange	TNotifyEvent	La valeur a changé, par l'utilisateur, dans le code ou parce que l'enregistrement a bougé.
OnChangeComplete, OnFormatLabel		Comme dans le curseur de plage.

Comment le champ est modifié

- La valeur suit l'enregistrement. Un champ null place la poignée à Min, et un champ de fin null place la poignée haute à Max.
- Une modification de l'utilisateur met le dataset en mode édition et marque la valeur comme modifiée. Définir la valeur dans le code est aussi une modification. Lorsque le dataset ne peut pas passer en mode édition, la valeur du champ revient.
- Entrée et la perte du focus écrivent le champ, et le dataset écrit une valeur en attente avant d'enregistrer. Un champ qui refuse la valeur lève une exception et le focus reste dans le contrôle, comme avec TDBEdit.
- Échap restaure la valeur du champ.
- Dans RangeMode, le contrôle est en lecture seule tant que le champ de fin ne peut pas être modifié.

Exemple : Une fourchette de prix dans deux champs

```
uses
  SysUtils, Classes, Controls, DB,
  sgcEditors_VCL_DB;

procedure BindPriceRange(aSlider: TsgcVCLDBRangeSlider; aSource: TDataSource);
begin
  aSlider.Min := 0;
  aSlider.Max := 1000;
  aSlider.Step := 25;
  aSlider.ShowLabels := True;
  aSlider.TickInterval := 250;
  // Value goes to DataField and, in range mode, EndValue to EndDataField
  aSlider.RangeMode := True;
  aSlider.DataSource := aSource;
  aSlider.DataField := 'PriceFrom';
  aSlider.EndDataField := 'PriceTo';
end;

function RangeEditable(aSlider: TsgcVCLDBRangeSlider): Boolean;
begin
  // both fields must be writable for the range to change
  Result := (aSlider.Field <> nil) and aSlider.Field.CanModify and
    (aSlider.EndField <> nil) and aSlider.EndField.CanModify;
end;
```

Voir aussi

- Curseur de plage
- Évaluation DB
- Démarrage rapide data-aware
- TsgcEditorFieldLink

TsgcEditorModel

Unit : sgcEditors_Types
TsgcEditorModel est la classe de base de chaque modèle de sgcEditors. C'est un TPersistent qui n'utilise que la RTL et sgcUICore, de sorte qu'il fonctionne avec VCL, avec FireMonkey et sans aucune fenêtre, sous Windows et sous Linux64. Un modèle contient la valeur et les options, reçoit les touches et la souris, et se dessine lui-même en deux couches : Render dessine la couche de contenu et RenderOverlay le survol, l'anneau de focus et les parties glissées. Chaque contrôle VCL et FireMonkey héberge un modèle, public sous le nom Model et comme propriété typée.

Modèle	Unit	Utilisé par	Propriété du contrôle
TsgcCalendarModel	sgcEditors_Calendar	Le calendrier mensuel et la liste déroulante des sélecteurs de date.	CalendarModel, DropDownCalendar
TsgcDatePickerModel	sgcEditors_Picker	Les sélecteurs de date, d'heure et de date et d'heure.	PickerModel
TsgcDateTimeDropDownModel	sgcEditors_Picker	La liste déroulante du sélecteur de date et d'heure : un calendrier et une liste des heures côte à côte.	
TsgcEditorListModel	sgcEditors_List	La liste des heures et la liste de saisie semi-automatique du token edit.	DropDownTimeList, TokenModel.AutoCompleteList
TsgcSliderModel	sgcEditors_Slider	Le curseur de plage.	SliderModel
TsgcColorPickerModel	sgcEditors_Color	Le sélecteur de couleur.	ColorModel
TsgcTokenModel	sgcEditors_Token	Le token edit.	TokenModel
TsgcRatingModel	sgcEditors_Rating	L'évaluation.	RatingModel
TsgcSpinModel	sgcEditors_Spin	Le spin edit.	SpinModel

Un modèle publie les mêmes propriétés de valeur que son contrôle, comme Date, Value ou Tokens, plus des membres pour les hôtes, comme HitTest et les rectangles de ses parties.

Propriétés

Propriété	Type	Description
Theme	TsgcUITheme	utLight (par défaut) ou utDark. Voir Thèmes.
AccentColor	TsgcUIColor	La couleur d'accentuation ARGB. 0 (par défaut) utilise la couleur d'accentuation du thème.

TextStyle	TsgcUIText-Style	Size, Bold et Color du texte.
Enabled	Boolean	Un modèle désactivé ne reçoit aucune saisie et dessine les couleurs désactivées. Par défaut True.
ReadOnly	Boolean	Le modèle ne reçoit aucune saisie qui modifie la valeur. Par défaut False.
Focused	Boolean	Public. Dessine l'anneau de focus dans la surcouche. Les hôtes le définissent lorsque le contrôle prend et perd le focus.
HotPart	Integer	Public. La partie sous la souris, 0 s'il n'y en a pas. Chaque modèle numérote ses propres parties.
Painter	TsgcEditors-Painter	Public, en lecture seule. Les couleurs du thème et le dessin du cadre, de l'anneau de focus, du chevron et de la flèche de spin du dernier rendu.

Méthodes

Méthode	Renvoie	Description
BeginUpdate, EndUpdate		Regroupent plusieurs modifications. EndUpdate de la paire extérieure envoie une fois chaque type de modification en attente.
IsUpdating	Boolean	True entre BeginUpdate et EndUpdate.
KeyDown(var aKey: Word; aShift)	Boolean	Une touche, sous forme de code de touche virtuelle Windows. Les constantes CS_EDITORS_KEY_* contiennent les codes, et FireMonkey utilise les mêmes valeurs. True lorsque le modèle l'a traitée.
KeyPress(var aKey: Char)	Boolean	Un caractère. True lorsque le modèle l'a traité.
MouseDown, MouseUp(aButton, aShift, aX, aY)	Boolean	Un bouton de souris, embLeft, embRight ou embMiddle, à aX et aY en pixels logiques. True lorsque le modèle l'a traité.
MouseMove(aShift, aX, aY), MouseWheel(aShift, aWheelDelta, aX, aY), MouseLeave	Boolean	Le pointeur et la molette, en pixels logiques. Un cran de molette vaut 120.
Render(aCanvas, aWidth, aHeight, aScale = 1)		Dessine la couche de contenu sur n'importe quel TsgcUICanvas. aWidth et aHeight sont en pixels de l'appareil.
RenderOverlay(aCanvas, aWidth, aHeight, aScale = 1)		Dessine la surcouche par-dessus la couche de contenu.
SaveToPNG(aStream or aFileName, aWidth, aHeight, aScale = 1)		Dessine les deux couches dans un PNG. aWidth et aHeight sont en pixels logiques et aScale les multiplie.
SaveToSVG(aStream or aFileName, aWidth, aHeight, aScale = 1)		La même chose dans un SVG.
MeasureText(aText, aFont)	Double	La largeur d'un texte en pixels de l'appareil : OnMeasure lorsqu'il est affecté, sinon la police intégrée à sgcUICore.

Assign(aSource)		Copie les options et la valeur d'un autre modèle de la même classe.

Événements

Événement	Type	Description
OnModelChange	TsgcEditorModelChangeEvent	(Sender; aKind: TsgcEditorChangeKind). eckContent : redessiner la couche de contenu. eckOverlay : redessiner seulement la surcouche. eckValue : la valeur a changé. eckLayout : les parties ont bougé, donc replacer l'edit interne.
OnMeasure	TsgcEditorMeasureEvent	function (const aText: string; const aFont: TsgcUIFontStyle): Double. Mesure un texte avec la police de l'hôte, en pixels de l'appareil.

Un contrôle affecte OnModelChange et OnMeasure de son propre modèle. Ne les affectez que sur un modèle que vous créez vous-même.

Taille et saisie

Un modèle se dispose et fait son test de position en pixels logiques, à la taille du dernier Render ou à la taille donnée à SetViewSize (Width et Height sur TsgcCalendarModel). Faites un Render une fois, ou donnez la taille, avant d'envoyer des saisies de souris à un modèle que vous hébergez vous-même.

Exemple : Piloter un modèle de curseur sans fenêtre

```
uses
  SysUtils, Classes,
  sgcEditors_Types, sgcEditors_Slider;

type
  TModelLog = class
  public
    Lines: TStrings;
    procedure ModelChange(Sender: TObject; aKind: TsgcEditorChangeKind);
  end;

procedure TModelLog.ModelChange(Sender: TObject; aKind: TsgcEditorChangeKind);
const
  CKinds: array [TsgcEditorChangeKind] of string = ('content', 'overlay',
    'value', 'layout');
begin
  Lines.Add(CKinds[aKind]);
end;

procedure DriveSlider(aLog: TModelLog);
var
  oSlider: TsgcSliderModel;
  vKey: Word;
begin
  oSlider := TsgcSliderModel.Create;
  try
    oSlider.OnModelChange := aLog.ModelChange;
    // one notification per kind for the whole block
    oSlider.BeginUpdate;
    try
      oSlider.Max := 10;
      oSlider.Step := 1;
```

```
      oSlider.Value := 4;
      oSlider.ShowLabels := True;
    finally
      oSlider.EndUpdate;
    end;
    // the input a host sends: neutral key codes, logical pixels
    vKey := CS_EDITORS_KEY_RIGHT;
    if oSlider.KeyDown(vKey, []) then
      aLog.Lines.Add('Value ' + FloatToStr(oSlider.Value));
    // both layers into a PNG, 300 x 48 logical pixels at twice the size
    oSlider.Focused := True;
    oSlider.SaveToPNG('slider.png', 300, 48, 2);
  finally
    oSlider.Free;
  end;
end;
```

Voir aussi

- Architecture
- Sortie sans interface
- TsgcEditorLocale

TsgcEditorLocale

Unit : sgcEditors_Date
TsgcEditorLocale contient les noms des jours et des mois, les formats de date et d'heure, les séparateurs et le premier jour de la semaine que les modèles de date utilisent pour dessiner et pour analyser. Un contrôle garde ses propres paramètres régionaux et les remplit depuis ses FormatSettings, donc sur un formulaire vous définissez FormatSettings, voir Localisation. Utilisez TsgcEditorLocale avec un modèle que vous créez vous-même, comme un TsgcCalendarModel sans interface, et avec les fonctions de date de l'unit.
TsgcCalendarModel et TsgcDatePickerModel ont une propriété Locale. Le modèle ne possède pas les paramètres régionaux, et nil (par défaut) utilise sgcEditorDefaultLocale, des paramètres régionaux partagés capturés depuis ceux de l'utilisateur à la première utilisation. Remplissez d'abord les paramètres régionaux et affectez-les ensuite, car le modèle se redessine lorsque Locale est affecté, pas lorsqu'une propriété des paramètres régionaux change.

Propriétés

Propriété	Type	Description
ShortDayNames[aDay], LongDayNames[aDay]	string	Les noms d'un TsgcEditorWeekDay, ewdMonday à ewdSunday.
ShortMonthNames[aMonth], LongMonthNames[aMonth]	string	Les noms d'un mois, de 1 à 12.
ShortDateFormat, LongDateFormat	string	Les formats de date. Un DisplayFormat vide d'un sélecteur de date utilise ShortDateFormat.
ShortTimeFormat, LongTimeFormat	string	Les formats d'heure. Un DisplayFormat vide d'un sélecteur d'heure utilise ShortTimeFormat.
TimeAMString, TimePMString	string	Les textes AM et PM d'un format sur 12 heures.
DateSeparator, TimeSeparator	Char	Écrits là où un format contient / et :.
FirstDayOfWeek	TsgcEditorWeekDay	Le premier jour de la semaine que représente efdLocale.

Méthodes

Méthode	Renvoie	Description
Create		Crée les paramètres régionaux et appelle Capture.
Capture		Prend les noms, les formats et les séparateurs dans la RTL, et le premier jour de la semaine dans le système d'exploitation, le lundi lorsqu'il n'en a pas.
CaptureSettings(aSettings: TFormatSettings)		Prend les noms, les formats et les séparateurs dans aSettings, comme TFormatSettings.Create('de-DE'). FirstDayOfWeek est conservé.
Assign(aSource)		Copie d'autres paramètres régionaux.

Fonctions

Fonction	Renvoie	Description
sgcEditorDefaultLocale	TsgcEditorLocale	Les paramètres régionaux partagés que les modèles utilisent lorsque leur Locale vaut nil. Ne les libérez pas.
sgcEditorFormatDateTime(aDateTime, aFormat, aLocale = nil)	string	Écrit une date et une heure avec un DisplayFormat et les noms de paramètres régionaux.
sgcEditorParseDateTime(aText, aFormat, aReference, out aDateTime, aLocale = nil)	Boolean	L'analyse tolérante des sélecteurs. Les champs qui manquent au texte viennent de aReference.
sgcEditorFirstDayOf(aFirstDay, aLocale = nil)	TsgcEditorWeekDay	Résout un TsgcEditorFirstDayOfWeek, de sorte que efdLocale donne le FirstDayOfWeek des paramètres régionaux.

Événements

TsgcEditorLocale n'a pas d'événements.

Exemple : Un calendrier allemand sur un serveur

```
uses
  SysUtils, Classes,
  sgcEditors_Types, sgcEditors_Date, sgcEditors_Calendar;

procedure RenderGermanCalendar(const aFileName: string);
var
  oLocale: TsgcEditorLocale;
  oCalendar: TsgcCalendarModel;
begin
  // a locale of its own; the shared default locale stays as it is
  oLocale := TsgcEditorLocale.Create;
  try
    oLocale.CaptureSettings(TFormatSettings.Create('de-DE'));
    // CaptureSettings keeps the first day of the week, so set it
    oLocale.FirstDayOfWeek := ewdMonday;
    oCalendar := TsgcCalendarModel.Create;
    try
      // fill the locale first, then assign it; the model does not own it
      oCalendar.Locale := oLocale;
      oCalendar.ShowWeekNumbers := True;
      oCalendar.DisplayMonth := EncodeDate(2026, 10, 1);
      oCalendar.SaveToPNG(aFileName, 280, 240);
    finally
      oCalendar.Free;
    end;
  finally
    oLocale.Free;
  end;
end;

function GermanDateText(const aDate: TDateTime): string;
var
  oLocale: TsgcEditorLocale;
begin
  oLocale := TsgcEditorLocale.Create;
  try
```

```
    oLocale.CaptureSettings(TFormatSettings.Create('de-DE'));
    // the long day and month names of the locale
    Result := sgcEditorFormatDateTime(aDate, 'dddd, d. mmmm yyyy', oLocale);
  finally
    oLocale.Free;
  end;
end;
```

Voir aussi

- Localisation
- Calendrier mensuel
- TsgcEditorModel

TsgcEditorFieldLink

Unit : sgcEditors_Data
TsgcEditorFieldLink est le TDataLink des contrôles data-aware. Il n'utilise que l'unit DB, de sorte que les contrôles VCL et FireMonkey le partagent ; FireMonkey n'a pas de TFieldDataLink. Il porte la sémantique de TFieldDataLink, vous pouvez donc aussi l'utiliser pour lier un contrôle de votre cru à un champ. Il fait partie des éditions Registered et Trial, et l'unit est vide dans Community.

Propriétés

Propriété	Type	Description
DataSource	TDataSource	Hérité de TDataLink. La source de données du dataset.
FieldName	string	Le nom du champ. Il est résolu avec FieldByName lorsque le lien devient actif et lorsque les champs du dataset changent.
Field	TField	En lecture seule. Le champ, nil tant que le lien n'est pas actif ou que FieldName est vide.
ReadOnly	Boolean	Hérité de TDataLink. True empêche le lien de modifier.
CanModify	Boolean	En lecture seule. Le lien n'est pas ReadOnly, le champ existe et TField.CanModify vaut True : le champ n'est ni en lecture seule ni calculé et le dataset peut modifier.
Editing	Boolean	En lecture seule. True tant que le dataset est dans un état d'édition et que le champ peut être modifié.
IsModified	Boolean	En lecture seule. True après Modified, jusqu'à ce que UpdateRecord écrive le champ, que OnDataChange le recharge ou que Editing change.

Méthodes

Méthode	Renvoie	Description
Edit	Boolean	Met le dataset en mode édition par la source de données, lorsque le champ peut être modifié. True lorsque le lien est ensuite en édition.
Modified		Marque la valeur du contrôle comme modifiée.
Reset		Recharge le contrôle depuis le champ par OnDataChange et efface IsModified.
UpdateRecord		Hérité de TDataLink. Déclenche OnUpdateData lorsque IsModified, de sorte que le contrôle écrit le champ une fois. Le dataset l'appelle avant un post.

Événements

Événement	Type	Description

OnDataChange	TNotifyEvent	L'enregistrement ou le champ a changé : un défilement, une annulation, un post, une actualisation ou Reset. Affichez la valeur du champ. Il n'est pas déclenché pendant que Edit met le dataset en mode édition, car le contrôle détient alors la modification en attente de l'utilisateur.
OnUpdateData	TNotifyEvent	Écrivez la valeur du contrôle dans Field.
OnActiveChange	TNotifyEvent	Le dataset s'est ouvert ou fermé.
OnEditingChange	TNotifyEvent	Editing a changé. IsModified est effacé.

Exemple : Lier un contrôle de votre cru

```
uses
  SysUtils, Classes, Controls, DB,
  sgcEditors_Data, sgcEditors_VCL_Value;

type
  // binds a plain TsgcVCLRating to a numeric field, as the DB controls do
  TRatingBinder = class
  private
    FLink: TsgcEditorFieldLink;
    FRating: TsgcVCLRating;
    FLoading: Boolean;
    procedure LinkDataChange(Sender: TObject);
    procedure LinkUpdateData(Sender: TObject);
    procedure RatingChange(Sender: TObject);
    procedure RatingExit(Sender: TObject);
  public
    constructor Create(aRating: TsgcVCLRating; aSource: TDataSource;
      const aFieldName: string);
    destructor Destroy; override;
  end;

constructor TRatingBinder.Create(aRating: TsgcVCLRating; aSource: TDataSource;
  const aFieldName: string);
begin
  inherited Create;
  FRating := aRating;
  FRating.OnChange := RatingChange;
  FRating.OnExit := RatingExit;
  FLink := TsgcEditorFieldLink.Create;
  FLink.OnDataChange := LinkDataChange;
  FLink.OnUpdateData := LinkUpdateData;
  FLink.FieldName := aFieldName;
  FLink.DataSource := aSource;
end;

destructor TRatingBinder.Destroy;
begin
  FLink.Free;
  inherited;
end;

procedure TRatingBinder.LinkDataChange(Sender: TObject);
begin
  // the record moved or the field changed: show the value of the field
  FLoading := True;
  try
    if (FLink.Field = nil) or FLink.Field.IsNull then
```

```
      FRating.Value := 0
    else
      FRating.Value := FLink.Field.AsFloat;
  finally
    FLoading := False;
  end;
end;

procedure TRatingBinder.RatingChange(Sender: TObject);
begin
  // a change of the user: edit mode, then mark the value as modified;
  // when the dataset cannot be edited the field value returns
  if FLoading then
    Exit;
  if FLink.Edit then
    FLink.Modified
  else
    FLink.Reset;
end;

procedure TRatingBinder.LinkUpdateData(Sender: TObject);
begin
  // UpdateRecord and the post of the dataset call it when Modified
  FLink.Field.AsFloat := FRating.Value;
end;

procedure TRatingBinder.RatingExit(Sender: TObject);
begin
  FLink.UpdateRecord;
end;
```

Voir aussi

- Démarrage rapide data-aware
- Évaluation DB
- TsgcEditorModel

Saisie de texte et IME

Chaque champ de texte de sgcEditors est un vrai contrôle edit : le texte de date ou d'heure des sélecteurs, le nombre du spin edit, le champ hexadécimal du sélecteur de couleur et la saisie après le dernier chip du token edit. Le calendrier mensuel, le curseur de plage et l'évaluation n'ont pas de texte et sont des contrôles dessinés qui prennent eux-mêmes le focus.

Pourquoi un edit natif

L'edit interne est un enfant TEdit sans bordure en VCL et un TEdit FMX sous FireMonkey, placé là où le modèle dessine son texte. Comme c'est l'edit de la plateforme, la composition IME pour la saisie japonaise, chinoise et coréenne, le presse-papiers, l'annulation, le menu contextuel et les lecteurs d'écran fonctionnent sans code particulier. Le cadre, les boutons, les unités et les chips autour sont dessinés par le modèle. En VCL, l'edit n'est jamais un arrêt de tabulation à lui seul : le contrôle lui transmet le focus, et l'anneau de focus est dessiné tant que l'edit a le focus.
Sous Windows, le canvas raster de chaque contrôle charge une fois les fichiers de police de la plateforme et l'edit interne utilise la même police et la même taille en pixels, de sorte que le texte saisi et le texte dessiné concordent. Les autres plateformes et la sortie sans interface utilisent Roboto, la police intégrée à sgcUICore.

Saisir une date ou une heure

- Le texte suit DisplayFormat, composé des segments d, dd, ddd, dddd, m, mm, mmm, mmmm, yy, yyyy, h, hh, n, nn, s, ss, am/pm et ampm plus des littéraux. Un `/` écrit le séparateur de date des paramètres régionaux et un `:` le séparateur d'heure. Un DisplayFormat vide utilise la date courte, l'heure courte ou les deux, tirées des FormatSettings du contrôle.
- L'analyse est tolérante. L'un de `/ - . : ,` ou une espace sépare les champs, les noms des mois et des jours des paramètres régionaux sont compris, les années à deux chiffres sont pivotées à 50, et une année à quatre chiffres en tête est lue comme année, mois, jour. Un champ absent du texte prend l'année d'aujourd'hui et l'heure à 0, et un format sans champs d'heure conserve l'heure de la valeur actuelle.
- Les sélecteurs analysent le texte à chaque modification et déplacent le calendrier ouvert avec lui. La valeur est validée sur Entrée, sur la perte du focus et à la fermeture de la liste déroulante.
- Haut et Bas font avancer le segment sous le curseur et le sélectionnent de nouveau, de sorte que vous pouvez faire défiler le mois ou l'heure. Les champs rebouclent comme le sélecteur de date et d'heure de Windows, puis MinDate, MaxDate, MinTime et MaxTime bornent le résultat.
- Un texte qui ne peut pas être analysé déclenche OnParseError avec le texte, la valeur et aAccept à False. Définissez aValue et aAccept pour prendre le texte, ou laissez-les, et la dernière valeur valide revient. Une date hors des limites ou désactivée dans le calendrier est refusée à la validation.
- Avec AllowEmpty à True, un texte vide efface la valeur et IsEmpty vaut True. TextHint s'affiche tant que l'edit est vide : à partir de Delphi 2009 en VCL, et comme TextPrompt de l'edit sous FireMonkey.

Saisir un nombre

Le spin edit analyse lui aussi de façon tolérante. Les espaces et le séparateur de milliers des paramètres régionaux sont ignorés, et `.` comme `,` sont acceptés comme séparateur décimal. Le texte est formaté avec DisplayFormat, un masque FormatFloat, ou avec Decimals lorsque DisplayFormat est vide. Units est un suffixe atténué dessiné après le texte, qui n'en fait jamais partie. OnParseError fonctionne comme pour les sélecteurs, et OnValidate vous permet de modifier ou de refuser chaque nouvelle valeur, aAccept arrivant à True.

Saisir des jetons

- Chaque caractère de Delimiters, une virgule et un point-virgule par défaut, termine un jeton, et Entrée le fait toujours. Tab et la perte du focus valident aussi le texte saisi. Coller `a, b, c` produit trois chips.
- OnValidateToken reçoit le texte nettoyé de ses espaces. Modifiez-le pour normaliser le jeton, ou effacez aValid pour le refuser. Avec InvalidTokens à etiReject, le jeton n'est pas ajouté, avec etiMark il est ajouté et dessiné comme invalide.

- AllowDuplicates, CaseSensitive et MaxTokens limitent ce qui est ajouté. AddToken, RemoveToken et Clear modifient les jetons dans le code, et OnTokenAdded et OnTokenRemoved signalent chaque changement.

Exemple

```
uses
  SysUtils, Classes, Controls,
  sgcEditors_Types, sgcEditors_VCL_Date, sgcEditors_VCL_Value,
  sgcEditors_VCL_Token;

type
  TEntryRules = class(TComponent)
  public
    procedure DateParseError(Sender: TObject; const aText: string;
      var aValue: TDateTime; var aAccept: Boolean);
    procedure ValidateQuantity(Sender: TObject; var aValue: Double;
      var aAccept: Boolean);
    procedure ValidateEmail(Sender: TObject; var aText: string;
      var aValid: Boolean);
  end;

procedure TEntryRules.DateParseError(Sender: TObject; const aText: string;
  var aValue: TDateTime; var aAccept: Boolean);
begin
  // words the lenient parser does not know; aAccept comes in False
  if SameText(Trim(aText), 'today') then
  begin
    aValue := Date;
    aAccept := True;
  end
  else if SameText(Trim(aText), 'tomorrow') then
  begin
    aValue := Date + 1;
    aAccept := True;
  end;
end;

procedure TEntryRules.ValidateQuantity(Sender: TObject; var aValue: Double;
  var aAccept: Boolean);
begin
  // boxes of six: round the typed value up, aAccept comes in True
  aValue := Trunc((aValue + 5) / 6) * 6;
end;

procedure TEntryRules.ValidateEmail(Sender: TObject; var aText: string;
  var aValid: Boolean);
begin
  // aText is trimmed; normalise it, then accept or refuse it
  aText := LowerCase(aText);
  aValid := (Pos('@', aText) > 1) and (Pos('.', aText) > Pos('@', aText));
end;

procedure SetupEntry(aRules: TEntryRules; aPicker: TsgcVCLDatePicker;
  aSpin: TsgcVCLSpinEdit; aTokens: TsgcVCLTokenEdit);
begin
  aPicker.DisplayFormat := 'ddd d mmm yyyy';
  aPicker.AllowEmpty := True;
  aPicker.TextHint := 'Delivery date';
  aPicker.OnParseError := aRules.DateParseError;

  aSpin.ValueType := esvInteger;
  aSpin.Increment := 6;
  aSpin.OnValidate := aRules.ValidateQuantity;

  // a space also ends a token; refused addresses stay as marked chips
  aTokens.Delimiters := ',; ';
```

```
  aTokens.InvalidTokens := etiMark;
  aTokens.OnValidateToken := aRules.ValidateEmail;
end;
```

Voir aussi

- Clavier et accessibilité
- Localisation
- Sélecteur de date
- Token edit

Clavier et accessibilité

Aucune fonction de sgcEditors n'a besoin d'une souris. Les contrôles dessinés prennent eux-mêmes le focus, et les contrôles avec texte transmettent le focus à leur edit interne. Un contrôle ayant le focus dessine un anneau de focus de 2 pixels dans la couleur d'accentuation, sur sa couche de surcouche.

Touches

Contrôle	Touches
Calendrier mensuel	Les flèches avancent d'un jour et d'une semaine et ignorent les jours désactivés. Page préc. et Page suiv. avancent d'un mois, avec Ctrl d'une année. Entrée ou Espace sélectionne. Maj avec les flèches étend une plage. Ctrl+Haut dézoome vers la vue année et décennie, et Entrée zoome de nouveau.
Sélecteurs	F4 ou Alt+Bas ouvre et ferme la liste déroulante. Haut et Bas font avancer le segment sous le curseur, ou se déplacent dans la liste déroulante ouverte. Entrée valide, Échap ferme la liste déroulante et restaure la valeur.
Spin edit	Haut et Bas avancent par Increment, Page préc. et Page suiv. par LargeIncrement. La molette avance par Increment, et un bouton maintenu enfoncé se répète plus vite plus il est maintenu longtemps.
Curseur de plage	Les flèches avancent par Step, Page préc. et Page suiv. par LargeStep, Début et Fin vont à Min et Max. En mode plage, voir les modèles de focus ci-dessous.
Évaluation	Les flèches modifient la valeur d'un pas de Precision, Début et Fin vont à la plus petite valeur et à Count, et un chiffre définit cette valeur. Avec AllowClear, 0 ou le nouveau choix de la valeur actuelle efface la valeur.
Sélecteur de couleur	Les flèches se déplacent entre les échantillons ou déplacent le curseur du carré et des barres, Entrée ou Espace sélectionne, Début et Fin vont aux extrémités. L'edit hexadécimal accepte les textes #RGB, #RGBA, #RRGGBB et #RRGGBBAA.
Token edit	Entrée ou un délimiteur ajoute le jeton saisi. Retour arrière dans une saisie vide met le focus sur le dernier chip, et un second Retour arrière le supprime. Gauche et Droite parcourent les chips, Suppr supprime le chip ayant le focus, Échap revient à la saisie. Haut et Bas se déplacent dans la liste de saisie semi-automatique.

Modèles de focus

La plupart des contrôles sont un seul arrêt de tabulation. Deux contrôles ont deux parties, et VCL et FireMonkey les atteignent différemment, car FireMonkey déplace le focus sur Tab avant que le contrôle voie la touche.

Contrôle	VCL	FireMonkey
Curseur de plage en mode plage	Tab va de la poignée basse à la poignée haute, le Tab suivant quitte le contrôle. Maj+Tab arrive sur la poignée haute. Ctrl+Gauche ou Ctrl+Bas sélectionne la poignée basse, Ctrl+Droite ou Ctrl+Haut la poignée haute.	Un seul arrêt de tabulation. Le focus commence sur la poignée basse, et un clic garde la poignée qui a été saisie. Ctrl+Gauche ou Ctrl+Bas sélectionne la poignée basse, Ctrl+Droite ou Ctrl+Haut la poignée haute.
Sélecteur de couleur	Deux arrêts de tabulation : les parties, puis l'edit hexadécimal. Entrée sur les parties sélectionne la couleur et n'atteint pas le bouton	Un arrêt de tabulation avec deux arrêts internes. Ctrl avec les flèches passe des parties à l'edit hexadécimal.

	par défaut du formulaire. Un sélecteur en lecture seule est un seul arrêt de tabulation.	

Sous FireMonkey, le focus est mis en file d'attente par le framework, et OnEnter et OnExit se déclenchent une fois par contrôle, même lorsque le focus passe entre le contrôle et son edit interne.

Accessibilité

- Les edits internes sont des edits natifs, de sorte que les lecteurs d'écran lisent et modifient leur texte comme d'habitude.
- Les contrôles VCL dessinés répondent à WM_GETTEXT avec `AccessibleName: value` par le proxy de fenêtre MSAA, et chaque changement de valeur appelle NotifyWinEvent avec EVENT_OBJECT_VALUECHANGE, de sorte qu'un lecteur d'écran annonce une nouvelle position de curseur ou une nouvelle évaluation.
- Les contrôles FireMonkey placent le même texte dans leur Hint, en conservant un Hint de votre cru.
- Le texte des deux thèmes intégrés a un contraste d'au moins 4,5:1, et les couleurs dérivées d'une couleur d'accentuation personnalisée sont ajustées pour le conserver. La QA le vérifie.

Exemple

```
uses
  SysUtils, Classes, Controls,
  sgcEditors_VCL_Date, sgcEditors_VCL_Value, sgcEditors_VCL_Color;

procedure SetupAccess(aCalendar: TsgcVCLMonthCalendar;
  aSlider: TsgcVCLRangeSlider; aRating: TsgcVCLRating;
  aColor: TsgcVCLColorPicker);
begin
  // screen readers announce "AccessibleName: value"
  aCalendar.AccessibleName := 'Departure date';
  aSlider.AccessibleName := 'Price range';
  aRating.AccessibleName := 'Hotel rating';
  aColor.AccessibleName := 'Label colour';

  // the tab order: calendar, slider (two stops in range mode), rating,
  // colour picker (the swatches, then the hex edit)
  aCalendar.TabOrder := 0;
  aSlider.TabOrder := 1;
  aRating.TabOrder := 2;
  aColor.TabOrder := 3;

  aSlider.RangeMode := True;
  aSlider.Step := 5;
  aSlider.LargeStep := 50;
  aRating.AllowClear := True;
end;
```

Voir aussi

- Saisie de texte et IME
- Thèmes
- Curseur de plage
- Sélecteur de couleur

Thèmes

Chaque contrôle a un Theme, une AccentColor et un TextStyle. Les couleurs du cadre, du survol, de l'état appuyé, de la sélection et de l'état désactivé en sont dérivées par TsgcEditorsPainter, de sorte qu'une seule propriété change l'aspect de tout le contrôle.

Propriété	Effet
Theme	utLight (par défaut) ou utDark, les jeux de couleurs clair et sombre de sgcUICore. utCustom utilise le jeu clair.
AccentColor	La couleur de la sélection, des poignées, du remplissage et de l'anneau de focus, sous forme de TsgcUIColor ARGB comme $FF2A78D6. 0 utilise la couleur d'accentuation du thème, et une couleur sans alpha est prise comme opaque.
TextStyle	Size (11 par défaut), Bold et Color du texte. Un Color de 0 suit la couleur de texte du thème.
Enabled	Un contrôle désactivé fond sa surface, son texte, sa bordure et sa couleur d'accentuation vers l'arrière-plan.

Couleurs dérivées

Les remplissages de survol et d'état appuyé sont des mélanges clairs de la couleur d'accentuation sur la surface. Le texte sur un remplissage d'accentuation, comme le jour sélectionné, est une couleur sombre ou claire, selon ce qui se lit le mieux, et le remplissage est assombri ou éclairci jusqu'à ce que ce texte atteigne un contraste de 4,5:1. La bande de sélection d'une plage garde le même contraste avec le texte normal. Ainsi, n'importe quelle couleur d'accentuation reste lisible dans les deux thèmes.

Couleurs des contrôles

- L'évaluation a FillColor et EmptyColor. Un FillColor de 0 est l'ambre et un EmptyColor de 0 suit le thème. Glyph choisit ergStar, ergHeart, ergCircle ou ergCustom, qui dessine par OnDrawGlyph.
- Le sélecteur de couleur affiche Swatches, par défaut 24 couleurs tirées des palettes de sgcUICore et des gris.

La largeur d'un chip de jeton dépend du thème et de la police, de sorte qu'un changement de Theme ou de TextStyle mesure de nouveau chaque chip et redispose le contrôle.

Exemple

```
uses
  SysUtils, Classes, Controls, Forms,
  sgcUICore_Theme, sgcEditors_Types, sgcEditors_VCL, sgcEditors_VCL_Value;

procedure ApplyEditorsTheme(aForm: TForm; aDark: Boolean);
var
  i: Integer;
  oEditor: TsgcVCLCustomEditor;
begin
  // every sgcEditors control descends from TsgcVCLCustomEditor
  for i := 0 to aForm.ComponentCount - 1 do
    if aForm.Components[i] is TsgcVCLCustomEditor then
    begin
      oEditor := TsgcVCLCustomEditor(aForm.Components[i]);
      if aDark then
        oEditor.Theme := utDark
```

```
      else
        oEditor.Theme := utLight;
      // ARGB; 0 returns to the accent of the theme
      oEditor.AccentColor := $FF8E44AD;
      oEditor.TextStyle.Size := 12;
    end;
end;

procedure StyleRating(aRating: TsgcVCLRating);
begin
  aRating.Glyph := ergHeart;
  aRating.FillColor := $FFE0245E;
  aRating.EmptyColor := 0;
  aRating.Precision := erpHalf;
end;
```

Voir aussi

- Clavier et accessibilité
- Sortie sans interface

Localisation

Chaque contrôle a une propriété publique FormatSettings. Elle démarre avec les paramètres régionaux de l'utilisateur, et l'affecter change les noms, les formats et les séparateurs que le contrôle utilise. Chaque contrôle garde sa propre copie, de sorte que deux contrôles d'un même formulaire peuvent utiliser deux langues.

Dates et heures

- Le calendrier et les sélecteurs capturent depuis FormatSettings les noms courts et longs des jours, les noms courts et longs des mois, les formats de date et d'heure courts et longs, les chaînes AM et PM et les séparateurs de date et d'heure.
- FirstDayOfWeek vaut efdLocale par défaut, ce qui prend le premier jour de la semaine du système d'exploitation, le lundi lorsque le système n'en a pas. efdMonday, efdSaturday et efdSunday le forcent.
- WeekNumbering vaut ewnISO8601 par défaut : la semaine 1 est la semaine qui contient le premier jeudi de l'année. ewnSimple compte comme semaine 1 la semaine qui contient le 1er janvier, en commençant le premier jour de la semaine.
- Un DisplayFormat vide signifie la date courte des paramètres régionaux pour le sélecteur de date, l'heure courte pour le sélecteur d'heure, et les deux, séparées par une espace, pour le sélecteur de date et d'heure. L'analyse comprend les noms des mois et des jours des paramètres régionaux. Voir Saisie de texte et IME.

Nombres

Le spin edit ainsi que les étiquettes et l'info-bulle de valeur du curseur de plage formatent leurs nombres avec les séparateurs décimal et de milliers de FormatSettings. DisplayFormat du spin edit et LabelFormat du curseur sont des masques FormatFloat, et OnFormatLabel du curseur peut écrire n'importe quel texte, comme une devise ou une unité.

Texte des contrôles

Le titre du calendrier, les noms des jours et la liste des heures viennent des paramètres régionaux, et les unités, les indications et les noms accessibles sont vos propres propriétés.

Exemple

Un calendrier allemand avec des numéros de semaine ISO, un sélecteur de date et d'heure dans les formats courts allemands, et un spin edit et un curseur avec des séparateurs allemands :

```
uses
  SysUtils, Classes, Controls,
  sgcEditors_Types, sgcEditors_VCL_Date, sgcEditors_VCL_Value;

procedure UseGerman(aCalendar: TsgcVCLMonthCalendar;
  aPicker: TsgcVCLDateTimePicker; aSpin: TsgcVCLSpinEdit;
  aSlider: TsgcVCLRangeSlider);
var
  vSettings: TFormatSettings;
begin
  vSettings := TFormatSettings.Create('de-DE');

  // day and month names, formats and separators come from FormatSettings
  aCalendar.FormatSettings := vSettings;
  aCalendar.FirstDayOfWeek := efdMonday;
  aCalendar.ShowWeekNumbers := True;
  aCalendar.WeekNumbering := ewnISO8601;

  // empty DisplayFormat: the short date and the short time of the locale
  aPicker.FormatSettings := vSettings;
```

```
  aPicker.DisplayFormat := '';

  // 1.234,50 EUR
  aSpin.FormatSettings := vSettings;
  aSpin.ValueType := esvFloat;
  aSpin.DisplayFormat := '#,##0.00';
  aSpin.Units := 'EUR';

  aSlider.FormatSettings := vSettings;
  aSlider.LabelFormat := '#,##0';
end;
```

Voir aussi

- Saisie de texte et IME
- DPI
- TsgcEditorLocale

DPI

sgcEditors dispose chaque contrôle en pixels logiques, des pixels à 96 DPI, et le dessine à l'échelle de l'écran. Rien n'est à faire dans votre code.

VCL

- L'échelle d'un contrôle est son PPI divisé par 96. Le contenu et la surcouche sont rastérisés à cette échelle, et les positions de la souris sont divisées par elle avant d'atteindre le modèle.
- Lorsque le formulaire passe sur un moniteur avec un autre DPI, ChangeScale et WM_DPICHANGED déplacent et mettent à l'échelle l'edit interne et sa police avec le contrôle.
- La liste déroulante est dessinée au PPI du moniteur sur lequel elle s'ouvre, de sorte qu'elle est nette lorsque le formulaire s'étend sur deux moniteurs. Delphi 7 utilise Screen.PixelsPerInch.

FireMonkey

Le raster d'un contrôle est sa Width multipliée par l'échelle de la scène, de sorte qu'il est net sur les écrans Retina et les écrans Android à haut DPI. Le modèle garde sa disposition logique.

Sortie sans interface

SaveToPNG et SaveToSVG prennent la taille en pixels logiques et une échelle. Une échelle de 2 écrit un PNG de largeur et de hauteur doubles, la même disposition qu'un contrôle sur un écran à 192 DPI.

Texte mesuré

Les largeurs de texte, comme la largeur d'un chip de jeton, dépendent de l'échelle. Le token edit garde un cache de chips mesurés, indexé sur l'outil de mesure, l'échelle et le thème, de sorte qu'un changement de DPI mesure de nouveau chaque chip et redispose le contrôle.

Exemple

```
uses
  SysUtils, Classes,
  sgcEditors_Calendar;

procedure ExportCalendarIcons(const aFolder: string);
var
  oModel: TsgcCalendarModel;
begin
  oModel := TsgcCalendarModel.Create;
  try
    oModel.Date := EncodeDate(2026, 10, 14);
    oModel.ShowWeekNumbers := True;
    // the same 280 x 240 logical layout at 96, 144 and 192 DPI:
    // 280 x 240, 420 x 360 and 560 x 480 device pixels
    oModel.SaveToPNG(IncludeTrailingPathDelimiter(aFolder) + 'cal_100.png',
      280, 240, 1);
    oModel.SaveToPNG(IncludeTrailingPathDelimiter(aFolder) + 'cal_150.png',
      280, 240, 1.5);
    oModel.SaveToPNG(IncludeTrailingPathDelimiter(aFolder) + 'cal_200.png',
      280, 240, 2);
  finally
    oModel.Free;
```

```
  end;
end;
```

Voir aussi

- Sortie sans interface
- Thèmes

Sortie sans interface

Les modèles de sgcEditors s'affichent sans fenêtre. Utilisez-les pour dessiner un contrôle dans un rapport, un document ou une page web, pour faire un test d'instantané, ou pour effectuer le rendu sur un serveur, sous Windows ou sous Linux64.

Modèles

Créez un modèle, comme TsgcCalendarModel, TsgcDatePickerModel, TsgcSliderModel, TsgcColorPickerModel, TsgcTokenModel, TsgcRatingModel ou TsgcSpinModel, définissez ses propriétés et appelez l'une de ces méthodes de TsgcEditorModel. Elles dessinent la couche de contenu et la surcouche.

Méthode	Écrit
SaveToPNG(aStream, aWidth, aHeight, aScale)	Un PNG dans un flux.
SaveToPNG(aFileName, aWidth, aHeight, aScale)	Un fichier PNG.
SaveToSVG(aStream, aWidth, aHeight, aScale)	Un SVG dans un flux.
SaveToSVG(aFileName, aWidth, aHeight, aScale)	Un fichier SVG.

aWidth et aHeight sont en pixels logiques, et aScale, 1 par défaut, les multiplie. La sortie sans interface utilise Roboto, la police intégrée à sgcUICore, de sorte que le même modèle donne les mêmes pixels sur toutes les machines. Les méthodes Render et RenderOverlay dessinent sur n'importe quel TsgcUICanvas que vous possédez.

Contrôles

Chaque contrôle VCL et FireMonkey a aussi SaveToPNG et SaveToSVG avec un flux ou un nom de fichier et une échelle. Ils effectuent le rendu du contrôle à sa taille actuelle, avec le texte de son edit interne dessiné par le modèle.

Linux64

À partir de Delphi 10.2, les units indépendantes du framework compilent pour Linux64, de sorte qu'un service Linux peut afficher le même calendrier, curseur ou évaluation qu'une application de bureau.

Exemple

```
uses
  SysUtils, Classes,
  sgcUICore_Theme, sgcEditors_Types, sgcEditors_Calendar, sgcEditors_Spin,
  sgcEditors_Rating;

procedure RenderEditors(aStream: TStream);
var
  oCalendar: TsgcCalendarModel;
  oSpin: TsgcSpinModel;
  oRating: TsgcRatingModel;
begin
  // a range in a dark month calendar, as an SVG file
  oCalendar := TsgcCalendarModel.Create;
  try
    oCalendar.Theme := utDark;
    oCalendar.SelectionMode := esmRange;
```

```
    oCalendar.RangeStart := EncodeDate(2026, 10, 12);
    oCalendar.RangeEnd := EncodeDate(2026, 10, 16);
    oCalendar.DisplayMonth := EncodeDate(2026, 10, 1);
    oCalendar.SaveToSVG('calendar.svg', 280, 240);
  finally
    oCalendar.Free;
  end;

  // a spin edit with units, as a PNG at twice the size
  oSpin := TsgcSpinModel.Create;
  try
    oSpin.ValueType := esvFloat;
    oSpin.Decimals := 1;
    oSpin.Value := 21.5;
    oSpin.Units := 'C';
    oSpin.SaveToPNG('spin.png', 120, 28, 2);
  finally
    oSpin.Free;
  end;

  // three and a half stars, as a PNG into a stream
  oRating := TsgcRatingModel.Create;
  try
    oRating.Precision := erpHalf;
    oRating.Value := 3.5;
    oRating.SaveToPNG(aStream, 160, 32);
  finally
    oRating.Free;
  end;
end;
```

Voir aussi

- DPI
- Architecture
- TsgcEditorModel

Éditions

sgcEditors est construit à partir des mêmes sources pour chaque édition.

Édition	Inclut
Registered	Tout : les neuf contrôles pour VCL et FireMonkey, les cinq contrôles data-aware pour chaque framework, le carré HSV et la barre alpha du sélecteur de couleur, la liste de saisie semi-automatique du token edit et les deux poignées du curseur de plage.
Trial	Les mêmes fonctions que Registered. Un build compilé avec SGC_TRIAL affiche un message indiquant qu'il s'agit d'une version d'évaluation lorsqu'un composant est créé après la fin de la fenêtre de publication nommée dans le build. Rien d'autre ne change.
Community	Les neuf contrôles pour VCL et FireMonkey sans les fonctions avancées listées ci-dessous, et sans avis d'évaluation.

Les fonctions avancées

Les fonctions avancées sont compilées sous le define SGC_EDITORS_ADVANCED, que sgcVer.inc désactive dans l'édition Community. Voici exactement ce qui change :

Fonction	Dans Community
Sélecteur de couleur	ecpHSV et ecpAlpha dans Parts sont ignorés : le carré, la barre de teinte et la barre alpha ne sont pas dessinés, l'alpha vaut toujours $FF et le texte hexadécimal est #RRGGBB. Les échantillons et l'edit hexadécimal restent.
Token edit	AutoCompleteItems, AutoCompleteMatch, AutoCompleteMinChars et AutoCompleteMaxItems restent, de sorte qu'un formulaire enregistré par un build Registered se charge, mais la liste ne s'ouvre jamais.
Curseur de plage	RangeMode reste, mais une seule poignée est dessinée et modifiée. EndValue est conservée et n'est pas affichée.
Contrôles data-aware	Les units sgcEditors_Data, sgcEditors_VCL_DB et sgcEditors_FMX_DB sont compilées en units vides, de sorte que les dix classes data-aware et leur enregistrement n'existent pas. La palette affiche 18 composants au lieu de 28.

Community conserve le calendrier complet, les trois sélecteurs avec leurs listes déroulantes, le spin edit avec unités, l'évaluation en demi-étoiles, la validation des jetons, les thèmes, le clavier, le DPI et la sortie PNG et SVG sans interface.

Du code pour chaque édition

L'édition est un choix à la compilation dans sgcVer.inc, et SGC_COMMUNITY désactive aussi SGC_TRIAL. Une unit qui doit compiler avec Community et utilise un membre qui n'existe que dans l'édition avancée inclut sgcVer.inc et place ce code dans {$IFDEF SGC_EDITORS_ADVANCED}. Un formulaire avec un contrôle data-aware ne se charge pas dans un build Community, car la classe n'y existe pas. sgcEditors_Const contient la version et l'édition sous forme de constantes.

```
{$I sgcVer.inc}

uses
  SysUtils, Classes, Controls,
  sgcEditors_Const, sgcEditors_Types, sgcEditors_VCL_Color,
```

```
  sgcEditors_VCL_Token;

procedure SetupPickers(aColor: TsgcVCLColorPicker; aTags: TsgcVCLTokenEdit);
begin
  // the full picker: Community ignores ecpHSV and ecpAlpha
  aColor.Parts := [ecpSwatches, ecpHSV, ecpAlpha, ecpHex];
{$IFDEF SGC_EDITORS_ADVANCED}
  // the auto-complete list opens only in the Registered and Trial editions
  aTags.AutoCompleteItems.CommaText := 'Delphi,FireMonkey,VCL,Linux';
  aTags.AutoCompleteMatch := emmContains;
{$ENDIF}
end;

function EditionText: string;
begin
  // for example 'sgcEditors 2026.10.0 Registered'
  Result := CS_SGC_EDITORS_FULLNAME + ' ' + CS_SGC_EDITORS_EDITION;
end;
```

Voir aussi

- Installation
- Démarrage rapide data-aware

Vue d'ensemble de .NET

sgcEditors .NET est un portage C# managé de la bibliothèque sgcEditors de Delphi, pas une enveloppe autour d'elle. Les classes C# vivent dans l'espace de noms **esegece.sgcEditors** et gardent les noms de Delphi, de sorte que **TsgcCalendarModel**, **TsgcSpinModel**, **TsgcEditorLocale** et les énumérations se lisent de la même façon dans les deux langages, et les neuf contrôles suivent le schéma de noms de Delphi sur chaque hôte : **TsgcVCLDatePicker** est **TsgcWinFormsDatePicker**, **TsgcWpfDatePicker** et **TsgcAvaloniaDatePicker**. Les contrôles sont dessinés avec le portage .NET de **sgcUICore**, le même canevas, les mêmes polices et les mêmes thèmes clair et sombre que les autres produits de la suite visuelle d'eSeGeCe, autour d'une zone de texte hébergée du framework pour le texte saisi.

Paquets

Le portage est réparti en cinq paquets NuGet, de sorte qu'un projet n'inclut que l'hôte qu'il utilise.

PackageId	Contient	Frameworks cibles
esegece.sgcEditors	Les modèles des neuf contrôles (**TsgcCalendarModel**, **TsgcDatePickerModel**, **TsgcSliderModel**, **TsgcRatingModel**, **TsgcSpinModel**, **TsgcColorPickerModel**, **TsgcTokenModel**), les modèles de liste et de liste déroulante, le calcul de dates, les paramètres régionaux et la sortie SVG sans interface. Aucune dépendance native. Dépend de esegece.sgcUICore.	netstandard2.0, net472, net6.0, net8.0, net9.0
esegece.sgcEditors.Rendering	La sortie PNG sans interface, les méthodes d'extension **SaveToPNG** et **RenderToRaster** de **sgcEditors_Raster**, dessinées sur le canevas SkiaSharp de esegece.sgcUICore.Rendering, sous Windows, Linux et macOS.	netstandard2.0, net472, net6.0, net8.0, net9.0
esegece.sgcEditors.WinForms	Les neuf contrôles **TsgcWinForms** sur **TsgcWinFormsCustomEditor**.	net472, net8.0-windows, net9.0-windows
esegece.sgcEditors.Wpf	Les neuf contrôles **TsgcWpf** sur **TsgcWpfCustomEditor**.	net472, net8.0-windows, net9.0-windows
esegece.sgcEditors.Avalonia	Les neuf contrôles **TsgcAvalonia** sur **TsgcAvaloniaCustomEditor**, sous Windows, Linux et macOS avec Avalonia 11.	net8.0, net9.0

Les trois paquets d'hôte dépendent de esegece.sgcEditors.Rendering, qui dépend de esegece.sgcEditors, de sorte qu'une seule ligne `PackageReference` apporte tout ce dont un hôte a besoin. Voir Installation depuis NuGet.

Éditions

Édition	Identifiants de paquet	Comprend

Registered	esegece.sgcEditors, .Rendering, .WinForms, .Wpf et .Avalonia	Tout : les neuf contrôles des trois hôtes, les thèmes, la prise en charge du clavier et de l'accessibilité, la sortie PNG et SVG sans interface et les trois fonctions avancées : le carré HSV et la barre alpha du sélecteur de couleur, la liste de saisie semi-automatique du token edit et la seconde poignée du curseur de plage.
Community	Les mêmes identifiants avec le suffixe .Community, par exemple esegece.sgcEditors.WinForms.Community.	Tout sauf les trois fonctions avancées. Les propriétés restent, de sorte que le même code se compile dans les deux éditions : **Parts** ignore **ecpHSV** et **ecpAlpha** (l'alpha est toujours $FF), **AutoCompleteItems** n'ouvre jamais de liste, et **RangeMode** dessine et modifie une seule poignée tandis que **EndValue** est conservé.

C'est la même répartition que celle des éditions Delphi, voir Éditions. Les contrôles data-aware de Delphi n'ont pas d'équivalent .NET dans aucune édition : les propriétés de valeur se lient avec la liaison de données de chaque framework, voir Liaison de données dans .NET.

Hôtes

Hôte	Équivalent Delphi	Plateformes
WinForms	VCL (contrôles TsgcVCL)	Windows, .NET Framework 4.7.2 et .NET 8 et 9.
WPF	VCL et FMX	Windows, .NET Framework 4.7.2 et .NET 8 et 9. Les contrôles peuvent être déclarés en XAML.
Avalonia	FMX (contrôles TsgcFMX)	Windows, Linux et macOS, .NET 8 et 9.
Sans interface	les modèles et leurs SaveToPNG et SaveToSVG	Toute plateforme des frameworks cibles : SVG avec esegece.sgcEditors, PNG avec esegece.sgcEditors.Rendering.

Chaque hôte a la même API : les propriétés, les méthodes et les événements des contrôles Delphi avec les mêmes noms, plus **Theme**, **AccentColor**, **TextStyle**, **ReadOnly**, **FormatSettings**, **DropDown**, **CloseUp**, **SaveToPNG** et **SaveToSVG** dans la classe de base. Voir Classes .NET pour les noms de classes et Différences de .NET par rapport à Delphi pour ce qui change par rapport à Delphi.

Voir aussi

- Installation depuis NuGet
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Classes .NET
- Différences de .NET par rapport à Delphi
- Liaison de données dans .NET
- .NET Headless Rendering
- Démos .NET
- Plateformes

Installation depuis NuGet

Les paquets sont sur nuget.org. Ajoutez le paquet de l'hôte, les paquets du noyau et du rendu viennent avec lui comme dépendances.

Édition Registered

```
dotnet add package esegece.sgcEditors.WinForms
dotnet add package esegece.sgcEditors.Wpf
dotnet add package esegece.sgcEditors.Avalonia
```

Édition Community

```
dotnet add package esegece.sgcEditors.WinForms.Community
dotnet add package esegece.sgcEditors.Wpf.Community
dotnet add package esegece.sgcEditors.Avalonia.Community
```

Ou ajoutez la référence au fichier de projet à la main :

```
<ItemGroup>
  <PackageReference Include="esegece.sgcEditors.Wpf" Version="2026.10.0" />
</ItemGroup>
```

Référencez dans un projet soit les paquets Registered, soit les paquets Community, jamais les deux : ils contiennent les mêmes types dans le même espace de noms.

Paramètres du projet

- WinForms : un projet `net8.0-windows` avec `<UseWindowsForms>true</UseWindowsForms>`, ou un projet .NET Framework 4.7.2.
- WPF : un projet `net8.0-windows` avec `<UseWPF>true</UseWPF>`, ou un projet .NET Framework 4.7.2. Associez l'espace de noms en XAML avec `xmlns:sgc="clr-namespace:esegece.sgcEditors;assembly=esegece.sgcEditors.Wpf"`.
- Avalonia : un projet `net8.0` ou `net9.0` avec les paquets **Avalonia.Desktop** et **Avalonia.Themes.Fluent** d'Avalonia 11.3, comme toute application de bureau Avalonia. La zone de texte interne prend son aspect du thème Fluent. Associez l'espace de noms en XAML avec `xmlns:sgc="using:esegece.sgcEditors"`.
- Sans interface : esegece.sgcEditors seul suffit pour le SVG. Ajoutez esegece.sgcEditors.Rendering pour le PNG, il apporte SkiaSharp et ses ressources natives pour Windows et Linux.

En C#, chaque hôte a besoin de `using esegece.sgcEditors;`, et de `using esegece.sgcUICore;` pour **TsgcUITheme**, **TsgcUITextStyle** et les autres types des thèmes.

Voir aussi

- Vue d'ensemble de .NET
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Démos .NET

Démarrage rapide WinForms

Ce démarrage rapide construit en code un petit formulaire de réservation avec quatre des neuf contrôles WinForms : un sélecteur de date avec calendrier déroulant, un spin edit avec unités, un curseur de plage et un token edit.

Étapes

1. Créez un projet Windows Forms pour `net8.0-windows` et ajoutez le paquet **esegece.sgcEditors.WinForms** (**esegece.sgcEditors.WinForms.Community** dans l'édition Community).
2. Ajoutez `using esegece.sgcEditors;` et `using esegece.sgcUICore;`.
3. Créez les contrôles, définissez leurs propriétés et ajoutez-les au formulaire. Ce sont des contrôles WinForms ordinaires : Location, Size, Dock, Anchor, TabIndex et Enabled fonctionnent comme d'habitude.
4. Affectez les événements. **OnChange** est déclenché par chaque contrôle quand sa valeur change, les autres événements sont ceux du contrôle Delphi.

Exemple

Remarques

- Les valeurs de date et d'heure sont des dates OLE Automation dans un `double`, le TDateTime de Delphi. Convertissez avec `DateTime.ToOADate` et `DateTime.FromOADate`.
- Le curseur de plage attend entre **BeginInit** et **EndInit** avant de limiter **Value** et **EndValue** à **Min** et **Max**, de sorte que l'ordre des affectations n'a pas d'importance. Le concepteur Windows Forms appelle les deux, par `ISupportInitialize`.
- **Theme** est propre à chaque contrôle. Chaque contrôle descend de **TsgcWinFormsCustomEditor**, donc une boucle sur les contrôles définit le thème d'un formulaire entier.

Voir aussi

- Vue d'ensemble de .NET
- Installation depuis NuGet
- Classes .NET
- Liaison de données dans .NET
- Sélecteur de date

Démarrage rapide WPF

Les contrôles WPF peuvent être déclarés en XAML comme n'importe quel autre contrôle. Ce démarrage rapide déclare un sélecteur de date et d'heure, une évaluation à demi-étoiles et un sélecteur de couleur, et définit le reste dans le code behind.

Étapes

1. Créez un projet WPF pour `net8.0-windows` et ajoutez le paquet **esegece.sgcEditors.Wpf** (**esegece.sgcEditors.Wpf.Community** dans l'édition Community).
2. Associez l'espace de noms dans la fenêtre : `xmlns:sgc="clr-namespace:esegece.sgcEditors;assembly=esegece.sgcEditors.Wpf"`.
3. Déclarez les contrôles. Les énumérations prennent les noms de leurs valeurs, comme `Precision="erpHalf"`. Un contrôle sans Width ni Height prend une taille par défaut (160 x 28 pour la plupart des sélecteurs, 200 x 28 pour le sélecteur de date et d'heure, 280 x 240 pour le calendrier mensuel).
4. Affectez les événements dans le code behind.

Exemple

La fenêtre :

```
<Window x:Class="QuickStart.QuickStartWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:sgc="clr-namespace:esegece.sgcEditors;assembly=esegece.sgcEditors.Wpf"
        Title="Review" Width="360" Height="300">
  <StackPanel Margin="12">
    <sgc:TsgcWpfDateTimePicker x:Name="dttVisit" Width="220" Height="28" HorizontalAlignment="Left" />
    <sgc:TsgcWpfRating x:Name="rtgScore" Precision="erpHalf" Value="3.5" Margin="0,8,0,0"
                       Width="160" Height="32" HorizontalAlignment="Left" />
    <sgc:TsgcWpfColorPicker x:Name="clrTag" Margin="0,8,0,0" Width="240" Height="180"
                            HorizontalAlignment="Left" />
  </StackPanel>
</Window>
```

Le code behind :

Remarques

- Les propriétés de valeur sont des propriétés de dépendance qui se lient dans les deux sens par défaut : **Date**, **Time** et **DateTime** des sélecteurs, **Value** et **EndValue** du curseur, **Value** de l'évaluation et du spin edit, **SelectedColor** du sélecteur de couleur. Voir Liaison de données dans .NET.
- Le sélecteur de couleur a **SelectedWpfColor**, le `System.Windows.Media.Color` opaque de **SelectedColor**.
- Les listes déroulantes sont des popups WPF placés sous le contrôle, ou au-dessus près du bas de l'écran, et dessinés au DPI du moniteur du contrôle.

Voir aussi

- Vue d'ensemble de .NET
- Installation depuis NuGet
- Classes .NET
- Liaison de données dans .NET
- Évaluation

Démarrage rapide Avalonia

Les contrôles Avalonia s'exécutent sous Windows, Linux et macOS. Ce démarrage rapide est une application Avalonia complète en code : un calendrier mensuel qui sélectionne une plage de 14 jours au plus et un sélecteur d'heure avec une liste déroulante toutes les 30 minutes.

Étapes

1. Créez un projet console `net8.0`, définissez `<OutputType>WinExe</OutputType>` et ajoutez **Avalonia.Desktop** et **Avalonia.Themes.Fluent** 11.3 ainsi que le paquet **esegece.sgcEditors.Avalonia** (**esegece.sgcEditors.Avalonia.Community** dans l'édition Community).
2. Ajoutez le thème Fluent à l'application. La zone de texte interne des sélecteurs en prend l'aspect.
3. Créez les contrôles et ajoutez-les à un panneau. En XAML, associez l'espace de noms avec `xmlns:sgc="using:esegece.sgcEditors"`.

Exemple

Remarques

- Une heure est la fraction d'un jour d'une date OLE Automation, donc `TimeSpan.TotalDays` est une valeur d'heure.
- Les propriétés de valeur sont des propriétés stylées qui se lient dans les deux sens par défaut. Voir Liaison de données dans .NET.
- La sélection de plage fait partie de toutes les éditions. La seconde poignée du curseur de plage est la fonction avancée, voir Vue d'ensemble de .NET.
- **AutomationProperties.Name** nomme un contrôle pour les lecteurs d'écran, sa valeur est exposée par le pattern de valeur de son automation peer.

Voir aussi

- Vue d'ensemble de .NET
- Installation depuis NuGet
- Classes .NET
- Liaison de données dans .NET
- Calendrier mensuel

Classes .NET

Chaque classe Delphi garde son nom en .NET. Les contrôles VCL et FireMonkey deviennent les contrôles WinForms, WPF et Avalonia avec le préfixe de l'hôte, et les classes indépendantes du framework sont les mêmes sur tous les hôtes.

Classes des hôtes

Delphi	WinForms	WPF	Avalonia	Model (propriété de l'hôte)
TsgcVCLMonthCalendar, TsgcFMXMonthCalendar	TsgcWinFormsMonthCalendar	TsgcWpfMonthCalendar	TsgcAvaloniaMonthCalendar	TsgcCalendarModel (CalendarModel)
TsgcVCLDatePicker, TsgcFMXDatePicker	TsgcWinFormsDatePicker	TsgcWpfDatePicker	TsgcAvaloniaDatePicker	TsgcDatePickerModel (PickerModel)
TsgcVCLTimePicker, TsgcFMXTimePicker	TsgcWinFormsTimePicker	TsgcWpfTimePicker	TsgcAvaloniaTimePicker	TsgcDatePickerModel (PickerModel)
TsgcVCLDateTimePicker, TsgcFMXDateTimePicker	TsgcWinFormsDateTimePicker	TsgcWpfDateTimePicker	TsgcAvaloniaDateTimePicker	TsgcDatePickerModel (PickerModel)
TsgcVCLRangeSlider, TsgcFMXRangeSlider	TsgcWinFormsRangeSlider	TsgcWpfRangeSlider	TsgcAvaloniaRangeSlider	TsgcSliderModel (SliderModel)
TsgcVCLRating, TsgcFMXRating	TsgcWinFormsRating	TsgcWpfRating	TsgcAvaloniaRating	TsgcRatingModel (RatingModel)
TsgcVCLSpinEdit, TsgcFMXSpinEdit	TsgcWinFormsSpinEdit	TsgcWpfSpinEdit	TsgcAvaloniaSpinEdit	TsgcSpinModel (SpinModel)
TsgcVCLColorPicker, TsgcFMXColorPicker	TsgcWinFormsColorPicker	TsgcWpfColorPicker	TsgcAvaloniaColorPicker	TsgcColorPickerModel (ColorModel)
TsgcVCLTokenEdit, TsgcFMXTokenEdit	TsgcWinFormsTokenEdit	TsgcWpfTokenEdit	TsgcAvaloniaTokenEdit	TsgcTokenModel (TokenModel)
TsgcVCLCustomEditor, TsgcFMXCustomEditor	TsgcWinFormsCustomEditor	TsgcWpfCustomEditor	TsgcAvaloniaCustomEditor	TsgcEditorModel (Model)
TsgcVCLCustomDatePicker, TsgcFMXCustomDatePicker	TsgcWinFormsCustomDatePicker	TsgcWpfCustomDatePicker	TsgcAvaloniaCustomDatePicker	TsgcDatePickerModel (PickerModel)
le TEdit interne	TsgcWinFormsEditorInnerEdit (a TextBox)	TsgcWpfEditorInnerEdit (a TextBox)	TsgcAvaloniaEditorInnerEdit (a TextBox)	
la liste déroulante	TsgcWinFormsEditorPopup (a	TsgcWpfEditorPopup (a Popup)	TsgcAvaloniaEditorPopup (a Popup)	TsgcDateTimeDropDownModel,

	ToolStripDrop-Down)			TsgcEditorListMo-del

Les classes de base dérivent de **TsgcUIWinFormsControl**, **TsgcUIWpfControl** et **TsgcUIAvaloniaControl** de sgcUICore. Les classes d'accessibilité sont **TsgcWinFormsEditorAccessibleObject**, **TsgcWpfEditorAutomationPeer** et **TsgcAvaloniaEditorAutomationPeer** ; elles signalent le nom et la valeur du contrôle.

Membres de chaque contrôle

Membre	Description
Model	Le modèle du contrôle, du type du tableau ci-dessus. La propriété typée (CalendarModel, SpinModel...) renvoie le même objet.
InnerEdit	La zone de texte hébergée, null pour les contrôles sans texte saisi (calendrier mensuel, curseur de plage, évaluation).
Popup	La liste déroulante des sélecteurs et de la liste de saisie semi-automatique des jetons.
Theme, AccentColor, TextStyle	L'aspect : **TsgcUITheme** (utLight, utDark), la couleur d'accent sous forme de `uint` ($AARRGGBB, 0 est la couleur du thème) ainsi que la taille et la graisse de la police.
ReadOnly	La valeur peut être lue et copiée, mais pas modifiée.
FormatSettings	Une copie de **TsgcEditorFormatSettings** avec laquelle le contrôle formate et analyse, la culture actuelle par défaut. Modifiez la copie et réaffectez-la.
DroppedDown, DropDown(), CloseUp(bool)	L'état de la liste déroulante : l'ouvrir, la fermer avec ou sans validation.
OnChange	Déclenché quand la valeur change, Sender est le contrôle.
SaveToPNG, SaveToSVG	Dessinent le contrôle tel qu'il apparaît maintenant dans un flux ou un fichier, à une échelle.

Types indépendants du framework

Type	Description
TsgcEditorModel	Le modèle de base : thème, focus, activation, survol, clavier, souris, **Render**, **RendererOverlay** et **SaveToSVG**. Voir TsgcEditorModel.
TsgcEditorLocale	Les noms des jours et des mois et le premier jour de la semaine, repris des paramètres de format. Voir TsgcEditorLocale.
TsgcEditorFormatSettings	Le sous-ensemble du TFormatSettings de Delphi que lisent les éditeurs. `Create()` prend la culture actuelle, `Create(CultureInfo)` une autre.
TsgcEditorStrings	Le TStrings de Delphi : **DisabledDates**, **Swatches**, **Tokens**, **AutoCompleteItems**.
TsgcEditorListModel	La liste de la liste déroulante des heures et de la saisie semi-automatique des jetons.
TsgcDateTimeDropDownModel	La liste déroulante du sélecteur de date et d'heure : un calendrier à côté d'une liste d'heures.

sgcEditors_Types, sgcEditors_Date	Des classes statiques avec les fonctions libres et les constantes de Delphi : codes de touches (CS_EDITORS_KEY_*), formatage des nombres, conversion hexadécimale et HSV, le calcul de dates.
sgcEditors_Raster	Dans esegece.sgcEditors.Rendering : les méthodes d'extension **SaveToPNG** et **RenderToRaster** de chaque modèle.

Énumérations

Les énumérations gardent les noms et les préfixes de valeur de Delphi : **TsgcEditorSelectionMode** (esm), **TsgcEditorFirstDayOfWeek** (efd), **TsgcEditorWeekNumbering** (ewn), **TsgcEditorCalendarView** (ecv), **TsgcEditorOrientation** (eor), **TsgcEditorTickPlacement** (etp), **TsgcEditorInvalidTokens** (eti), **TsgcEditorMatchMode** (emm), **TsgcEditorRatingPrecision** (erp), **TsgcEditorRatingGlyph** (erg), **TsgcEditorSpinValueType** (esv) et **TsgcEditorSpinButtons** (esb). Les ensembles de Delphi sont des énumérations `[Flags]` combinées avec `|` : **TsgcEditorWeekDays**, **TsgcEditorColorParts**, **TsgcEditorChangeKinds** et **TsgcEditorShiftState**.

Voir aussi

- Vue d'ensemble de .NET
- Différences de .NET par rapport à Delphi
- Contrôles
- Architecture

Différences de .NET par rapport à Delphi

La version .NET garde les noms, les valeurs par défaut, le comportement du clavier et les règles de rendu de Delphi. La sortie SVG des modèles est identique octet pour octet à celle de Delphi, et la sortie PNG lui correspond dans la tolérance du rastériseur SkiaSharp. Les différences ci-dessous viennent des frameworks des hôtes et de la bibliothèque de classes .NET.

Pas de contrôles data-aware

Il n'y a ni contrôles DB ni **TsgcEditorFieldLink** en .NET, dans aucune édition. Les propriétés de valeur se lient à la place avec la liaison de données de chaque framework : `Binding` de WinForms par `INotifyPropertyChanged`, propriétés de dépendance de WPF et propriétés stylées d'Avalonia, les deux dans les deux sens par défaut. Voir Liaison de données dans .NET.

Pas de streaming de composants

Il n'y a pas de DFM en .NET, donc pas d'éditeur de liste de chaînes à la conception pour **DisabledDates** et **Swatches** : remplissez-les en code. Le curseur de plage diffère la limitation de **Value** et **EndValue** à l'intervalle entre **BeginInit** et **EndInit**, le point d'accroche qu'appellent le concepteur Windows Forms et les chargeurs XAML, là où Delphi attend Loaded.

Valeurs, événements et listes

- TDateTime est un `double`, une date OLE Automation : `DateTime.ToOADate` et `DateTime.FromOADate` la convertissent. Une heure est la fraction d'un jour.
- TsgcUIColor est un `uint`, $AARRGGBB, où 0 signifie la couleur du thème. **SelectedVCLColor** correspond à **SelectedWinFormsColor** (`System.Drawing.Color`), **SelectedWpfColor** (`System.Windows.Media.Color`) et **SelectedAvaloniaColor** (`Avalonia.Media.Color`), toutes opaques.
- Les événements sont des propriétés delegate nommées comme dans Delphi (**OnChange**, **OnSelect**, **OnTokenAdded**...). Affectez un gestionnaire avec `=`. Un paramètre `var` de Delphi est un paramètre `ref`, et une lambda avec un paramètre `ref` déclare le type de chaque paramètre. **TNotifyEvent** est **TsgcUINotifyEvent**, et Sender est le contrôle.
- Une propriété TStrings est un **TsgcEditorStrings**, avec **Add**, **Insert**, **Delete**, **Clear**, **IndexOf** (qui ignore la casse), **Assign**, **BeginUpdate** et **EndUpdate**. On accède aux objets de la liste de jetons avec **GetObject**.
- Une fonction Delphi sans paramètres reste une méthode, comme `IsRange()`, `CellCount()` ou `HexText()`. Un ensemble Delphi est une énumération `[Flags]` combinée avec `|`.
- TFormatSettings est **TsgcEditorFormatSettings**. La propriété **FormatSettings** renvoie une copie : modifiez-la et réaffectez-la, comme avec le record Delphi.

Zone de texte, clavier et focus selon l'hôte

- Le texte saisi est un `TextBox` hébergé du framework, donc le curseur, la sélection, le presse-papiers, l'annulation et l'IME sont ceux du framework. **TextHint** est le cue banner sur WinForms, une indication dessinée par l'édition interne sur WPF et le `Watermark` sur Avalonia.
- Tab, Entrée et Échap se comportent comme dans Delphi sur les trois hôtes. Sur Avalonia, une touche traitée par le contrôle n'atteint pas un gestionnaire KeyDown du contrôle lui-même.
- Le modèle travaille en pixels logiques. WPF et Avalonia placent en DIP, donc l'édition interne et la liste déroulante gardent la même taille à tout DPI ; WinForms les met à l'échelle selon le DPI du moniteur.
- Accessibilité : WinForms signale le nom et la valeur par l'`AccessibleObject`, WPF par un `AutomationPeer` avec le pattern de valeur, Avalonia par `AutomationProperties` et son automation peer. Il n'y a pas de texte de fenêtre comme dans la VCL.
- Sur WPF, **Tokens** du token edit est une propriété CLR, pas une propriété de dépendance ; **OnChange** signale ses modifications. Sur Avalonia, c'est une propriété directe notifiée à chaque modification.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- Liaison de données dans .NET
- Clavier et accessibilité

Liaison de données dans .NET

Les contrôles data-aware de Delphi relient un contrôle à un champ d'un TDataSet par **TsgcEditorFieldLink**. En .NET, les propriétés de valeur des contrôles ordinaires se lient avec la liaison de données de chaque framework, dans les deux éditions, il n'y a donc pas de contrôles DB séparés.

Contrôle	Propriétés de valeur
MonthCalendar	Date, RangeStart, RangeEnd
DatePicker, TimePicker, DateTimePicker	Date, Time, DateTime (les trois sont la même date OLE Automation vue comme une date, une heure ou les deux), IsEmpty (lecture seule)
RangeSlider	Value, EndValue
Rating	Value
SpinEdit	Value, IsEmpty (lecture seule)
ColorPicker	SelectedColor (`uint`), plus la couleur en lecture seule du type de l'hôte
TokenEdit	Tokens

WinForms

Les contrôles WinForms implémentent `INotifyPropertyChanged` et le déclenchent pour les propriétés de valeur, de sorte qu'un `Binding` avec `DataSourceUpdateMode.OnPropertyChanged` réécrit aussitôt, pendant que l'utilisateur saisit, incrémente ou fait glisser.

WPF

Les propriétés de valeur sont des propriétés de dépendance enregistrées pour se lier dans les deux sens par défaut : **DateProperty**, **TimeProperty**, **DateTimeProperty**, **RangeStartProperty**, **RangeEndProperty**, **ValueProperty**, **EndValueProperty**, **SelectedColorProperty** et la **IsEmptyProperty** en lecture seule. Le contrôle peut limiter une valeur liée à ses bornes puis réécrit la valeur limitée. **Tokens** est une propriété CLR et ne se lie pas.

Avalonia

Les propriétés de valeur sont des propriétés stylées qui se lient dans les deux sens par défaut, avec les mêmes noms que sur WPF. **IsEmptyProperty**, **SelectedAvaloniaColorProperty** et **TokensProperty** sont des propriétés directes : liez **Tokens** dans les deux sens à un **TsgcEditorStrings**, le setter copie les chaînes.
La démo BindingWpf lie un sélecteur de date, un sélecteur d'heure, un spin edit, une évaluation et un curseur de plage à une liste de réservations, voir Démos .NET.

Voir aussi

- Vue d'ensemble de .NET
- Différences de .NET par rapport à Delphi
- Démos .NET
- Démarrage rapide data-aware

Rendu sans interface dans .NET

Chaque contrôle dessine par son modèle, et les modèles n'ont pas besoin de fenêtre. Un service, un test ou un rapport peut créer un modèle, définir ses propriétés et l'enregistrer en SVG ou en PNG sur n'importe quelle plate-forme, Linux compris.

- **SaveToSVG** est une méthode de **TsgcEditorModel** dans esegece.sgcEditors, sans dépendance native.
- **SaveToPNG** est une méthode d'extension de **sgcEditors_Raster** dans esegece.sgcEditors.Rendering, dessinée avec SkiaSharp. **RenderToRaster** renvoie le canevas raster lui-même ; l'appelant le libère.

Les deux prennent le flux ou le nom du fichier, la largeur et la hauteur en pixels logiques et une échelle facultative : une échelle de 2 écrit une image de 560 x 480 pour un calendrier de 280 x 240. Les deux dessinent la couche de contenu puis la couche de superposition (survol, anneau de focus, une poignée glissée), de sorte que la sortie est ce que montre le contrôle.

Exemple

Remarques

- Les hôtes ont aussi **SaveToPNG** et **SaveToSVG**, avec la taille du contrôle.
- Les modèles formatent les nombres et les dates avec la culture actuelle. Pour obtenir la même sortie sur chaque machine, affectez `TsgcEditorFormatSettings.Create(CultureInfo.InvariantCulture)` à **FormatSettings** des modèles spin et slider, ou du contrôle de l'hôte.
- Les listes déroulantes sont aussi des modèles (**TsgcDateTimeDropDownModel**, **TsgcEditorListModel**), dessinés avec les mêmes méthodes.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- Sortie sans interface

Démos .NET

Les démos se trouvent dans `.net\demos\Editors`, un dossier par démo avec son projet, son `Program.cs` et un README.md. Elles construisent leur interface en code, sans XAML. Chaque démo a un commutateur `--test` qui exécute une autovérification hors écran, écrit des lignes PASS et FAIL et se termine avec le nombre de vérifications échouées, comme le commutateur `/test` des démos Delphi.

Démo	Hôte	Montre
Gallery	WinForms	Les neuf contrôles dans une fenêtre : un calendrier mensuel avec numéros de semaine, un sélecteur de date, un sélecteur d'heure avec une liste de 15 minutes, un sélecteur de date et d'heure, un spin edit avec unités, une évaluation, un curseur de plage avec graduations, un token edit avec liste de saisie semi-automatique et un sélecteur de couleur avec nuanciers, le carré HSV, l'alpha et un champ hexadécimal. Un commutateur de thème, un commutateur de lecture seule, l'export PNG et un journal d'événements.
GalleryWpf	WPF	La même fenêtre avec les contrôles **TsgcWpf**.
GalleryAvalonia	Avalonia	La même fenêtre avec les contrôles **TsgcAvalonia**, pour Windows, Linux et macOS.
BookingForm	WinForms	Un formulaire de réservation d'hôtel avec sept contrôles : un calendrier en mode plage pour le séjour (28 nuits au plus), des sélecteurs d'heure d'arrivée et de départ avec bornes et une liste toutes les 30 minutes, un spin edit pour les clients, un token edit pour six demandes au plus avec liste de saisie semi-automatique, une évaluation pour la catégorie de l'hôtel et un sélecteur de couleur avec nuanciers et hexadécimal pour la couleur de l'étiquette.
BindingWpf	WPF	MVVM : une liste de réservations dans un view model, et un sélecteur de date, un sélecteur d'heure, un spin edit, une évaluation et un curseur de plage liés à la réservation sélectionnée, l'équivalent .NET de la démo data-aware de Delphi.

Les démos Gallery montrent le carré HSV, la barre alpha, la seconde poignée du curseur de plage et la liste de saisie semi-automatique uniquement dans l'édition Registered, et BookingForm montre la liste de saisie semi-automatique uniquement dans celle-ci. Dans l'édition Community, le même code s'exécute sans eux.

Compiler les démos

Dans l'arborescence du produit, les démos se compilent contre les projets de la bibliothèque. Le zip des démos utilise par défaut les paquets NuGet de l'édition Community. Chaque démo se compile et s'exécute dans les deux éditions. Compilez une démo avec les paquets Registered en définissant **SgcEditorsEdition** :

```
dotnet build Gallery\Gallery.csproj -p:SgcEditorsEdition=Registered
dotnet run --project BookingForm\BookingForm.csproj -- --test
```

Voir aussi

- Vue d'ensemble de .NET
- Installation depuis NuGet
- Démarrage rapide WinForms
- Liaison de données dans .NET

Dépannage

Les problèmes courants de sgcEditors et leurs causes.

Problème	Cause et solution
Une date saisie revient à la valeur précédente.	Le texte n'a pas pu être analysé, ou la date est hors de MinDate et MaxDate, ou elle est désactivée par DisabledDates, DisabledWeekDays ou OnIsDateDisabled. Gérez OnParseError pour accepter vos propres mots, et appelez IsDateEnabled avant de définir une date dans le code.
Le sélecteur d'heure n'a pas de bouton de liste déroulante.	ListInterval vaut 0. Définissez-le sur les minutes entre les éléments de la liste, 30 par défaut. Les éléments restent entre MinTime et MaxTime.
Une valeur définie dans le code est modifiée.	La valeur est bornée aux limites. Définissez MinValue, MaxValue, ValueType et Decimals avant Value. MinValue égal à MaxValue signifie aucune limite. Le Max d'un curseur avec un Step redescend sur la grille des pas, comme dans une entrée range HTML.
Les choix dans une liste déroulante FireMonkey sont perdus.	Un clic à l'extérieur de la liste déroulante la ferme et annule les choix faits pendant qu'elle était ouverte. Fermez-la avec Entrée ou avec un choix.
Tab ne passe pas à la seconde poignée sous FireMonkey.	FireMonkey déplace le focus sur Tab. Utilisez Ctrl+Droite ou Ctrl+Haut pour la poignée haute et Ctrl+Gauche ou Ctrl+Bas pour la poignée basse.
Le sélecteur de couleur n'a pas de carré ni de barre alpha.	Parts n'inclut pas ecpHSV ou ecpAlpha, ou le build est Community, qui les ignore.
La liste de saisie semi-automatique ne s'ouvre jamais.	AutoCompleteItems est vide, moins de AutoCompleteMinChars caractères sont saisis, ou le build est Community.
Le curseur de plage affiche une seule poignée.	RangeMode vaut False, ou le build est Community.
Class TsgcVCLDBDatePicker not found when a form loads.	L'application est compilée avec l'édition Community, qui n'a pas de contrôles data-aware.
Un contrôle data-aware ne peut pas être modifié.	ReadOnly vaut True, DataSource ou son dataset est absent, le dataset ou le champ est en lecture seule, le champ est calculé ou, pour un curseur en mode plage, le champ de fin ne peut pas être modifié.
Définir Value d'un contrôle data-aware met le dataset en mode édition.	Une valeur définie dans le code compte comme une modification par l'utilisateur. Pour modifier l'enregistrement sans l'éditer, écrivez plutôt le champ.
Le PNG sans interface utilise une autre police que l'écran.	L'écran utilise la police de la plateforme, Segoe UI sous Windows, et la sortie sans interface utilise Roboto intégrée, de sorte que la sortie soit la même partout.
La saisie IME ne fonctionne pas dans le ca-	Ces contrôles n'ont pas de texte. L'IME fonctionne dans les edits internes des sélecteurs, du spin edit, du sélecteur de couleur et du token edit.

lendrier, le curseur ou l'évaluation.	
Un formulaire avec des centaines d'éditeurs s'ouvre lentement.	La plus grande partie du temps est consacrée à la création des fenêtres des contrôles et de leurs edits internes : environ 340 ms pour 270 contrôles. Créez les contrôles des pages masquées lorsque la page est affichée pour la première fois.

Exemple

```
uses
  SysUtils, Classes, Controls,
  sgcEditors_Types, sgcEditors_VCL_Date, sgcEditors_VCL_Value;

procedure SafeSetDate(aPicker: TsgcVCLDatePicker; const aDate: TDateTime);
begin
  // a date outside MinDate..MaxDate or disabled is refused on commit
  if aPicker.IsDateEnabled(aDate) then
    aPicker.Date := aDate
  else
    aPicker.Clear;
end;

procedure SetupTimeList(aPicker: TsgcVCLTimePicker);
begin
  // ListInterval 0 removes the drop-down button of a time picker
  aPicker.ListInterval := 15;
  aPicker.MinTime := EncodeTime(8, 0, 0, 0);
  aPicker.MaxTime := EncodeTime(18, 0, 0, 0);
end;

procedure SetupPercent(aSpin: TsgcVCLSpinEdit);
begin
  // set the limits and the type before Value, else Value is clamped
  // to the old limits
  aSpin.ValueType := esvFloat;
  aSpin.Decimals := 1;
  aSpin.MinValue := 0;
  aSpin.MaxValue := 100;
  aSpin.Value := 37.5;
  aSpin.Units := '%';
end;
```

Voir aussi

- Saisie de texte et IME
- Démarrage rapide data-aware
- Éditions

Introduction

sgcGauges ajoute des jauges d'instrumentation aux applications Delphi et C++Builder, pour VCL, pour FireMonkey et sans aucun framework d'interface utilisateur. Il existe dix types de jauges : radiale, linéaire, numérique, LED, barre de LED, anneau de progression, vumètre, horloge, boussole et annunciator. Chaque type existe sous forme de contrôle VCL avec le préfixe TsgcVCL, de contrôle FireMonkey avec le préfixe TsgcFMX et de classe de modèle indépendante du framework, comme TsgcRadialGauge, qui produit du PNG et du SVG sans fenêtre.
Une jauge a une ou plusieurs échelles, chacune avec ses plages de couleur, une ou plusieurs aiguilles (aiguille, repère, barre ou plage) et des seuils qui déclenchent un événement lorsqu'une valeur les franchit. Une nouvelle valeur déplace l'aiguille avec une animation et une courbe d'accélération. Les jauges ont des thèmes clair et sombre, s'adaptent à leur taille et au DPI, et la Value d'une jauge peut être écrite depuis n'importe quel thread.
Chaque jauge se dessine en deux couches. La couche statique contient la face, les échelles, les graduations, les étiquettes et les plages, et elle n'est redessinée que lorsque la configuration change. La couche dynamique contient les aiguilles et le texte de la valeur. Une nouvelle valeur ne redessine que la couche dynamique, de sorte qu'un panneau de nombreuses jauges reste rapide pendant que les valeurs changent.
sgcGauges fait partie de la suite visuelle eSeGeCe. Il est dessiné avec sgcUICore, le package d'exécution partagé avec le canevas, le rastériseur, les polices, l'écrivain SVG et les thèmes, de sorte que les jauges s'accordent avec les autres produits de la suite. sgcGauges nécessite sgcUICore, qui s'installe avec lui. La passerelle sgcGaugesWS de l'édition Registered alimente les jauges à partir de messages MQTT, WebSocket et d'événements envoyés par le serveur via sgcWebSockets.

Éditions

sgcGauges est livré en éditions Registered, Trial et Community. Community conserve les jauges radiale, linéaire, numérique (7 segments), LED, barre de LED et anneau de progression, avec plages, seuils, animation, thèmes et la sortie PNG et SVG sans interface, pour VCL et FireMonkey. Elle laisse de côté le vumètre, l'horloge, la boussole et l'annunciator, les échelles et aiguilles après la première, les afficheurs 14 segments et matrice de points et la passerelle. Voir Éditions.

Par où commencer

- Architecture : les modèles, les hôtes, les deux couches de rendu et les threads.
- Types de jauges : les dix jauges, leurs classes et leurs unités en un coup d'œil.
- Installation et Plateformes : l'installateur, les packages, les versions de Delphi et C++Builder et les plateformes.
- Démarrage rapide VCL, Démarrage rapide FMX, Démarrage rapide sans interface et Démarrage rapide C++Builder : une première jauge en quelques minutes.
- Composants : une page par contrôle, de la jauge radiale à l'annunciator, et les classes de modèle TsgcCustomGauge, TsgcGaugeScale et TsgcGaugePointer.
- Échelles, Plages, Aiguilles, Seuils, Animation, Thèmes, Sortie sans interface et Éditions.
- Passerelle : alimentez des jauges à partir de messages MQTT, WebSocket et d'événements envoyés par le serveur.
- Dépannage : problèmes courants et leurs causes.

Architecture

sgcGauges sépare ce qu'est une jauge de l'endroit où elle est affichée. Une classe de modèle contient la configuration et les valeurs et se dessine sur un canevas sgcUICore. Un contrôle hôte place un modèle sur une fiche VCL ou FireMonkey.

Modèles

Chaque type de jauge est une classe qui descend de **TsgcCustomGauge** (unit sgcGauges_Model), comme TsgcRadialGauge ou TsgcLinearGauge. Un modèle est un TPersistent sans fenêtre, il fonctionne donc aussi dans un programme console, dans un service ou sous Linux64. Il contient :

- **Scales** : une collection de TsgcGaugeScale, chacune avec une plage de valeurs, des graduations, des étiquettes et une collection de **Ranges** de couleur.
- **Pointers** : une collection de TsgcGaugePointer, chacun affichant une valeur sur une échelle.
- **Thresholds** : une collection de TsgcGaugeThreshold, des valeurs qui déclenchent OnThreshold lorsqu'une aiguille les franchit.
- **Animation** : la durée et la courbe d'accélération d'un déplacement d'aiguille.
- L'aspect partagé : Theme, Title, TitleFont, Units, ShowValue, ValueFormat, ValueFont, BackgroundColor et FaceColor.

Un nouveau modèle a toujours une échelle, de 0 à 100, et une aiguille. **Value** est un raccourci vers Pointers[0].Value, et SetValue(aPointer, aValue, aAnimate) écrit la valeur de n'importe quelle aiguille.

Hôtes

Les contrôles VCL descendent de **TsgcVCLCustomGauge** (unit sgcGauges_VCL) et les contrôles FireMonkey de **TsgcFMXCustomGauge** (unit sgcGauges_FMX). Chaque contrôle crée le modèle de son type, que la propriété publique **Gauge** renvoie, et publie les propriétés du modèle que son type dessine. Un hôte se redessine lorsque le modèle change, exécute les images de l'animation, délivre OnThreshold dans le thread principal et s'enregistre avec SaveToPNG et SaveToSVG à sa taille cliente.

Deux couches de rendu

Un modèle se dessine en deux couches :

- La **couche statique** (RenderStatic) : l'arrière-plan, la face, la lunette, les échelles, les graduations, les étiquettes, les plages, les marques de seuil et le titre. Un changement de propriété déclenche OnChange avec gckStatic et incrémente StaticStamp.
- La **couche dynamique** (RenderDynamic) : les aiguilles et le texte de la valeur. Une nouvelle valeur déclenche OnChange avec gckDynamic uniquement.

Les hôtes gardent la couche statique sous forme de pixels en cache et ne dessinent que la couche dynamique tant qu'une valeur change ou qu'une aiguille bouge. BeginUpdate et EndUpdate regroupent plusieurs changements en un seul OnChange, statique si l'un des changements était statique. Render dessine les deux couches à la fois.

Tailles, angles et couleurs

- Les tailles des échelles et des aiguilles (Radius, Offset, Thickness, Width, Length) sont des fractions de la taille de la jauge, de sorte qu'une jauge a le même aspect à toute taille. Le texte s'adapte aussi à la jauge.
- Les angles sont en degrés, 0 est 3 heures et les angles positifs tournent dans le sens horaire. L'horloge et la boussole comptent à partir de 12 heures et du nord.
- Les couleurs sont des valeurs TsgcUIColor de la forme $AARRGGBB. Une couleur de 0 utilise la couleur du thème.
- Les valeurs sont bornées à la plage de leur échelle lorsqu'elles sont écrites. NaN est ignoré et une valeur infinie va à la borne la plus proche.

Threads

Value, SetValue ainsi que Value et EndValue d'une aiguille peuvent être écrits depuis n'importe quel thread. Sur un modèle, OnChange et OnThreshold sont alors appelés dans ce thread. Les contrôles VCL et FireMonkey appellent leur OnThreshold dans le thread principal, quel que soit le thread qui a écrit la valeur. Toute autre propriété est destinée au thread principal.

Unités

Unité	Contenu
`sgcGauges_Types`	Les énumérations et les fonctions utilitaires.
`sgcGauges_Model`	TsgcCustomGauge, les échelles, les plages, les aiguilles, les seuils et l'animation.
`sgcGauges_Render`	Le peintre avec lequel chaque type de jauge se dessine.
`sgcGauges_Radial`, `_Linear`, `_Digital`, `_LED`, `_ProgressRing`, `_VUMeter`, `_Clock`, `_Compass`, `_Annunciator`	Les classes de modèle des dix types de jauges. La LED et la barre de LED partagent sgcGauges_LED.
`sgcGauges_VCL`	Les contrôles VCL.
`sgcGauges_FMX`	Les contrôles FireMonkey.
`sgcGaugesWS_Source`, `_Source_MQTT`, `_Source_WebSocket`, `_Source_SSE`	La passerelle sgcGaugesWS, édition Registered uniquement.

Voir aussi

- Types de jauges
- Sortie sans interface
- Animation

Types de jauges

sgcGauges a dix types de jauges. Chacun est une classe de modèle dans sa propre unité, un contrôle VCL dans sgcGauges_VCL et un contrôle FireMonkey dans sgcGauges_FMX. Les contrôles se trouvent sur la page de palette **SGC Gauges**.

Jauge	Classe de modèle (unité)	Contrôles VCL et FireMonkey	Édition	Ce qu'elle affiche
Radiale	TsgcRadialGauge (sgcGauges_Radial)	TsgcVCLRadialGauge, TsgcFMXRadialGauge	Community	Un cadran avec un arc complet, demi, quart ou personnalisé, une lunette et le texte de la valeur. RadialStyle, StartAngle, SweepAngle, BezelStyle, ValuePosition.
Linéaire	TsgcLinearGauge (sgcGauges_Linear)	TsgcVCLLinearGauge, TsgcFMXLinearGauge	Community	Une barre horizontale ou verticale, un thermomètre ou un graphique bullet avec une ligne cible. Orientation, LinearStyle, Target, BulbSize.
Numérique	TsgcDigitalGauge (sgcGauges_Digital)	TsgcVCLDigitalGauge, TsgcFMXDigitalGauge	Community (7 segments)	Un afficheur à segments d'une valeur ou d'un texte : 7 segments, 14 segments ou une matrice de points 5 x 7. DisplayType, Digits, Text, Skew, SegmentColor, GhostColor, ShowGhost, Alignment.
LED	TsgcLEDGauge (sgcGauges_LED)	TsgcVCLLEDGauge, TsgcFMXLEDGauge	Community	Un voyant allumé ou éteint, avec un halo et un clignotement optionnel. Active, OnColor, OffColor, Shape, Blink, BlinkInterval.
Barre de LED	TsgcLEDBarGauge (sgcGauges_LED)	TsgcVCLLEDBarGauge, TsgcFMXLEDBarGauge	Community	Une rangée ou une colonne de segments de LED avec maintien du pic. Segments, SegmentGap, Orientation, PeakHold, PeakHoldTime.
Anneau de progression	TsgcProgressRingGauge (sgcGauges_ProgressRing)	TsgcVCLProgressRingGauge, TsgcFMXProgressRingGauge	Community	Un anneau qui se remplit avec la valeur, avec un pourcentage au milieu. Thickness, TrackColor, RoundCaps, ShowPercent.
Vumètre	TsgcVUMeterGauge (sgcGauges_VUMeter)	TsgcVCLVUMeterGauge, TsgcFMXVUMeterGauge	Registered	Un indicateur de niveau audio avec une aiguille ou une barre et un maintien du pic.
Horloge	TsgcClockGauge (sgcGauges_Clock)	TsgcVCLClockGauge, TsgcFMXClockGauge	Registered	Une horloge analogique, live ou à une heure donnée, avec un décalage UTC optionnel.

Bous-sole	TsgcCompassGauge (sgcGauges_Compass)	TsgcVCLCom-passGauge, TsgcFMXCom-passGauge	Regis-tered	Une boussole avec une rose tournante ou une aiguille tour-nante et un cap cible.
Annun-ciator	TsgcAnnunciatorGauge (sgcGauges_Annunciator)	TsgcVCLAnnun-ciatorGauge, TsgcFMXAnnun-ciatorGauge	Regis-tered	Une grille de tuiles d'alarme à quatre états, qui clignotent jusqu'à leur acquittement.

Ce que partagent toutes les jauges

Chaque type descend de TsgcCustomGauge et possède donc Scales, Pointers, Thresholds, Animation, Theme et les propriétés du texte de la valeur, mais chaque contrôle ne publie que ce que son type dessine. La LED, par exemple, n'a pas d'échelle, l'horloge n'a pas de Value et la boussole publie Heading à la place de Value. Units est dessiné à côté du texte de la valeur, et le texte de la valeur utilise ValueFormat ou, lorsqu'il est vide, les décimales de l'échelle.

Voir aussi

- Architecture
- Éditions
- Démarrage rapide VCL

Installation

sgcGauges fait partie de la distribution sgcUI. Un seul installateur couvre les produits sgcUI et n'installe que les produits inclus dans votre licence.

Installation avec l'installateur sgcUI

1. Lancez l'installateur sgcUI et connectez-vous avec le nom d'utilisateur et le mot de passe de votre compte eSeGeCe.
2. Choisissez les versions de Delphi et C++Builder dans lesquelles installer.
3. Sur la page des composants, sélectionnez **sgcGauges** et les autres produits souhaités. Le serveur de licences renvoie les produits que vous possédez, et l'installateur affiche une case par produit. Les produits que votre licence n'inclut pas sont grisés. Une licence All-Access débloque tous les produits.
4. L'installateur copie les fichiers, puis compile les packages sur votre machine pour chaque IDE sélectionné, dans l'ordre des dépendances, et installe les packages de conception.

Ce qui accompagne sgcGauges

sgcUICore, la base partagée des produits visuels sgcUI, accompagne sgcGauges. Il n'a pas de case à cocher propre. L'installateur l'installe automatiquement chaque fois que sgcGauges est sélectionné.

Dossiers installés

Chaque produit s'installe dans un même dossier, par défaut `Documents\esegece\sgcUI` (`sgcUI_Trial` pour l'essai et `sgcUI_Community` pour l'édition Community).

Dossier	Contenu
`Source`	Le code source de chaque produit installé. C'est l'unique chemin de bibliothèque de sgcUI.
`Packages`	Les packages de chaque produit installé, un jeu par produit et par version d'IDE.
`Lib<V>`	Les unités compilées pour chaque version d'IDE et chaque plateforme, par exemple `LibD13` .
`Demos\Gauges`	Les démos de sgcGauges : Gallery, GalleryFMX, Headless, HMIPanel, DashboardGauges et MQTTLive, pour Delphi et C++Builder.

L'installateur ajoute `sgcUI\Source` au chemin de bibliothèque et au chemin de navigation de chaque IDE et plateforme sélectionnés, avec le dossier des unités compilées. sgcUICore se trouve dans le même dossier `Source`, il n'a donc besoin d'aucun chemin propre.

La palette de composants

Les composants se trouvent sur la page de palette **SGC Gauges**. Une fiche VCL affiche les contrôles VCL et une fiche FireMonkey les contrôles FMX. Les éditions Registered et Trial enregistrent 20 contrôles, les dix types de jauges pour chaque framework, et la passerelle ajoute ses trois sources d'alimentation (TsgcGaugeFeedSource_WebSocket, TsgcGaugeFeedSource_MQTT et TsgcGaugeFeedSource_SSE). Community enregistre 12 contrôles, les six types de jauges Community pour chaque framework.

Trial et Community

- **Trial** : l'installateur d'essai sgcUI installe chaque produit en mode d'essai, sgcGauges compris.

- **Community** : l'installateur gratuit sgcUI Community inclut sgcGauges sans ses fonctionnalités avancées. Voir Éditions.

Compiler les packages manuellement

Pour compiler vous-même les packages, ouvrez-les pour votre version d'IDE depuis `sgcUI\Packages`. `<V>` est le suffixe de version Delphi, par exemple `sgcGaugesD13.dpk`.

Package	Contenu
`sgcUICore<V>` , `sgcUICoreVCL<V>` , `sgcUICoreFMX<V>`	Le socle partagé sgcUICore et ses parties VCL et FireMonkey (exécution).
`sgcGauges<V>`	Les unités indépendantes du framework : les types, le modèle, le peintre et les dix modèles de jauge (exécution).
`sgcGaugesVCL<V>` / `dclsgcGauges<V>`	Les contrôles VCL (exécution / conception).
`sgcGaugesFMX<V>` / `dclsgcGaugesFMX<V>`	Les contrôles FireMonkey, à partir de Delphi XE6 (exécution / conception).
`sgcGaugesWS<V>` / `dclsgcGaugesWS<V>`	La passerelle vers sgcWebSockets, édition Registered uniquement. Elle nécessite le package sgcWebSockets.

Compilez-les dans l'ordre des dépendances : `sgcUICore`, `sgcUICoreVCL` et `sgcUICoreFMX`, puis `sgcGauges`, `sgcGaugesVCL` et `sgcGaugesFMX`, puis les packages de conception. Le groupe de projets `sgcGauges<V>.groupproj` contient les packages sgcGauges dans cet ordre. Installez ensuite les packages de conception et ajoutez `sgcUI\Source` au chemin de bibliothèque de votre version de Delphi (Tools > Options > Language > Delphi Options > Library > Library path). Pour C++Builder, ouvrez les projets `.cbproj` correspondant à votre version, par exemple `sgcGaugesC13.cbproj` et `sgcGaugesVCLC13.cbproj`, et compilez-les. Les en-têtes `.hpp` générés apparaissent à côté du package compilé.

Voir aussi

- Plateformes
- Démarrage rapide VCL

Plateformes

Cible	Prise en charge
Delphi	Delphi 7, 2007, 2009, 2010, XE à XE8, 10 à 10.4, 11, 12 et 13.
C++Builder	C++Builder 2007, 2010, XE à XE8, 10 à 10.4, 11, 12 et 13, avec les projets de package sgcGaugesC2007 à sgcGaugesC13. Les packages génèrent les en-têtes .hpp et les fichiers .lib qu'un projet C++Builder inclut et lie. Voir Démarrage rapide C++Builder.
VCL	Les contrôles TsgcVCL sous Windows, Win32 et Win64, pour toutes les versions de Delphi.
FireMonkey	Les contrôles TsgcFMX à partir de Delphi XE6 (SGC_GAUGES_FMX), sur les plateformes FireMonkey. Ils n'utilisent que FireMonkey et sgcUICore.
Linux64	Les unités indépendantes du framework à partir de Delphi 10.2 (SGC_GAUGES_LINUX) : le modèle, le peintre et les dix modèles de jauge, avec la sortie PNG et SVG sans interface. Il n'y a pas de contrôles Linux.
.NET	Une version C# avec la même API : les dix jauges en contrôles WinForms, WPF et Avalonia et la sortie PNG et SVG sans interface, sur .NET Framework 4.7.2 et .NET 8 et 9, voir Vue d'ensemble de .NET.

Packages

sgcGauges fournit un package d'exécution indépendant du framework et un package d'exécution par framework, de sorte qu'une application FireMonkey pour Android, iOS ou macOS ne lie jamais la VCL. Chaque nom de package se termine par le suffixe de l'IDE, par exemple sgcGaugesD13.dpk pour Delphi 13 et sgcGaugesC13.cbproj pour C++Builder 13.

Package	Type	Contenu	Requiert
sgcGauges	Exécution, indépendant du framework	Chaque unité du produit qui n'utilise ni la VCL ni FireMonkey : le modèle, le peintre et les dix modèles de jauge, avec la sortie PNG et SVG sans interface.	rtl, sgcUICore
sgcGaugesVCL	Exécution, VCL	sgcGauges_VCL (TsgcVCLRadialGauge, TsgcVCLLinearGauge, TsgcVCLDigitalGauge, TsgcVCLLEDGauge, TsgcVCLLEDBarGauge, TsgcVCLProgressRingGauge, TsgcVCLVUMeterGauge, TsgcVCLClockGauge, TsgcVCLCompassGauge et TsgcVCLAnnunciatorGauge)	rtl, vcl, sgcUICore, sgcUICoreVCL, sgcGauges
sgcGaugesFMX	Exécution, FireMonkey, à partir de XE6	sgcGauges_FMX (TsgcFMXRadialGauge, TsgcFMXLinearGauge, TsgcFMXDigitalGauge, TsgcFMXLEDGauge, TsgcFMXLEDBarGauge, TsgcFMXProgressRingGauge, TsgcFMXVUMeterGauge, TsgcFMXClockGauge, TsgcFMXCompassGauge et TsgcFMXAnnunciatorGauge)	rtl, fmx, sgcUICore, sgcUICoreFMX, sgcGauges
dclsgcGauges	Conception, VCL	Enregistre les jauges VCL dans la page de palette **SGC Gauges** (TsgcVCLVUMeterGauge, TsgcVCLClockGauge, TsgcVCLCompassGauge et TsgcVCLAnnunciatorGauge absents de l'édition Community), avec l'éditeur de collection pour Scales, Pointers, Thresholds et Tiles.	designide, rtl, vcl, sgcGauges, sgcGaugesVCL

dclsgcGaugesFMX	Conception, FireMonkey, à partir de XE6	Enregistre les jauges FireMonkey dans la page de palette **SGC Gauges**.	designide, rtl, fmx, sgcGauges, sgcGaugesFMX, dclsgcGauges
sgcGaugesWS, dclsgcGaugesWS	Exécution et conception, passerelle sgcWebSockets	Les sources de flux en direct (TsgcGaugeFeedSource_WebSocket, TsgcGaugeFeedSource_MQTT et TsgcGaugeFeedSource_SSE), indépendantes du framework, dans la page de palette **SGC Gauges**.	rtl, sgcGauges, sgcWebSockets

sgcUICore suit la même répartition : sgcUICore, sgcUICoreVCL et sgcUICoreFMX, sans package de conception. dclsgcGaugesFMX requiert dclsgcGauges, installez donc d'abord dclsgcGauges. Aucune unité sgcGauges n'utilise d'unité sgcWebSockets, sauf les unités de la passerelle optionnelle sgcGaugesWS.

Versions de l'IDE et plateformes cibles

Delphi	sgcGauges	sgcGaugesVCL	sgcGaugesFMX	dclsgcGauges, dclsgcGaugesFMX
7, 2007, 2009, 2010, XE	Win32	Win32	Aucune	Win32
XE2 à XE5	Win32, Win64	Win32, Win64	Aucune	Win32
XE6 à 10.2	Win32, Win64	Win32, Win64	Win32, Win64	Win32
10.3	Win32, Win64, Linux64, OSX64	Win32, Win64	Win32, Win64, OSX64	Win32
10.4	Win32, Win64, Linux64, OSX64, Android, Android64, iOSDevice64	Win32, Win64	Win32, Win64, OSX64, Android, Android64, iOSDevice64	Win32
11	Win32, Win64, Linux64, OSX64, OSXARM64, Android, Android64, iOSDevice64, iOSSimARM64	Win32, Win64	Win32, Win64, OSX64, OSXARM64, Android, Android64, iOSDevice64, iOSSimARM64	Win32
12, 13	Comme 11	Win32, Win64	Comme 11	Win32, Win64

De XE6 à 10.2, le package FireMonkey est compilé pour Windows uniquement. Les packages de conception sont chargés par l'IDE, ils sont donc compilés pour la plateforme de l'IDE uniquement. sgcGaugesWS est compilé pour les mêmes plateformes que sgcGauges, et dclsgcGaugesWS pour les mêmes plateformes que dclsgcGauges.

C++Builder	sgcGauges, sgcGaugesVCL	sgcGaugesFMX	dclsgcGauges, dclsgcGaugesFMX
2007, 2010, XE, XE2	Win32	Aucune	Win32 (dclsgcGauges uniquement)

XE3 à XE5	Win32, Win64	Aucune	Win32 (dclsgcGauges uniquement)
XE6 à 12	Win32, Win64	Win32, Win64	Win32
13	Win32, Win64, Win64x	Win32, Win64, Win64x	Win32, Win64x

Les projets C++Builder portent les mêmes noms avec le suffixe C, par exemple sgcGaugesC13.cbproj, sgcGaugesVCLC13.cbproj, sgcGaugesFMXC13.cbproj, dclsgcGaugesC13.cbproj et dclsgcGaugesFMXC13.cbproj. Les packages C++Builder sont compilés pour Windows uniquement. sgcGaugesWS et dclsgcGaugesWS sont compilés pour les mêmes versions et plateformes de C++Builder que sgcGauges et dclsgcGauges, sauf Win64x, car ils nécessitent une compilation Win64x du package sgcWebSockets. Les notes C++Builder qui s'appliquent à chaque produit sgcUI (packages d'exécution sous Win64, C++Builder 10.3 sous Win64, noms de fichiers des packages Win64x) se trouvent dans le sujet Plateformes et packages de l'aide de la suite sgcUI.

FireMonkey sur Android, iOS et macOS

Les contrôles FireMonkey (TsgcFMXRadialGauge, TsgcFMXLinearGauge, TsgcFMXDigitalGauge, TsgcFMXLEDGauge, TsgcFMXLEDBarGauge, TsgcFMXProgressRingGauge, TsgcFMXVUMeterGauge, TsgcFMXClockGauge, TsgcFMXCompassGauge et TsgcFMXAnnunciatorGauge) ciblent macOS à partir de Delphi 10.3, Android et iOS à partir de Delphi 10.4, et macOS ARM et le simulateur iOS à partir de Delphi 11. Dans un projet FireMonkey pour ces plateformes :

- Utilisez les contrôles FireMonkey. Les contrôles VCL et sgcGaugesVCL ne fonctionnent que sous Windows.
- Déposez le contrôle sur un formulaire, ou ajoutez son unité à la clause uses : sgcGauges_FMX.
- Sélectionnez la plateforme cible dans la fenêtre Projets. L'installateur ajoute le dossier Source de sgcUI et les unités compilées de chaque plateforme (LibD<version>\<platform>) au chemin de bibliothèque de chaque plateforme qu'il compile. Sans l'installateur, ajoutez le dossier Source de sgcUI au chemin de bibliothèque de la plateforme cible (Outils, Options, Langage, Delphi, Bibliothèque).
- Les applications Android et iOS sont liées statiquement, aucun package d'exécution n'est donc déployé.

Linux64

Le package indépendant du framework sgcGauges est compilé pour Linux64 à partir de Delphi 10.3. Une application Linux64, par exemple un programme console ou un serveur, utilise le modèle, le peintre et les dix modèles de jauge, avec la sortie PNG et SVG sans interface. Aucun package VCL ou FireMonkey n'est compilé pour Linux64, car FireMonkey sous Linux nécessite FMX pour Linux (FMXLinux), un module complémentaire tiers.

Contraintes de version

- Les unités indépendantes du framework et les unités VCL compilent sous Delphi 7 : pas de génériques, pas de méthodes anonymes, pas de variables inline et pas de record helpers.
- La sortie est la même sous Win32, Win64 et Linux64 : les fichiers PNG et SVG d'une jauge sont identiques octet pour octet sur les trois plateformes.
- Les énumérations ont des préfixes propres aux jauges (grs, gpk, gdt, gtd, ge, etc.), de sorte que les en-têtes C++Builder n'entrent jamais en conflit avec les autres produits sgcUI.

Voir aussi

- Éditions
- Architecture

Démarrage rapide VCL

Installez les packages sgcGauges pour votre version de Delphi. Les composants se trouvent sur la page de palette **SGC Gauges**.

À la conception

1. Déposez un **TsgcVCLRadialGauge** sur une fiche. Saisissez Speed dans Title et km/h dans Units.
2. Ouvrez la collection Scales, sélectionnez la première échelle et définissez Maximum à 260 et MajorInterval à 20. Ouvrez sa collection Ranges, ajoutez une plage et définissez StartValue à 200, EndValue à 260 et Color à un rouge comme $FFE34948.
3. Déposez un **TsgcVCLLEDGauge** à côté, définissez OnColor au même rouge et Blink à True.
4. Ajoutez un seuil à la collection Thresholds de la jauge radiale avec Value 200, Hysteresis 10 et Direction gtdBoth, et écrivez un gestionnaire OnThreshold qui affecte aRising à la propriété Active de la LED.
5. Définissez Value dans le code, depuis un timer ou un thread de travail, et exécutez l'application. L'aiguille se déplace vers chaque nouvelle valeur en 300 ms.

En code

Une jauge fonctionne dès qu'elle a un Parent. Définissez la plage d'une échelle avant la valeur, car une valeur est bornée à l'échelle lorsqu'elle est écrite.

```
uses
  SysUtils, Classes, Controls, Forms,
  sgcGauges_Types, sgcGauges_Model, sgcGauges_VCL;

type
  TSpeedPanel = class(TComponent)
  public
    Speed: TsgcVCLRadialGauge;
    Alarm: TsgcVCLLEDGauge;
    procedure SpeedThreshold(Sender: TObject; aThreshold: TsgcGaugeThreshold;
      aValue: Double; aRising: Boolean);
  end;

procedure TSpeedPanel.SpeedThreshold(Sender: TObject;
  aThreshold: TsgcGaugeThreshold; aValue: Double; aRising: Boolean);
begin
  // called in the main thread, whatever thread wrote the value
  Alarm.Active := aRising;
end;

function CreateSpeedPanel(aForm: TForm): TSpeedPanel;
var
  oRange: TsgcGaugeRange;
  oThreshold: TsgcGaugeThreshold;
begin
  Result := TSpeedPanel.Create(aForm);

  Result.Speed := TsgcVCLRadialGauge.Create(aForm);
  Result.Speed.Parent := aForm;
  Result.Speed.SetBounds(16, 16, 240, 240);
  Result.Speed.Title := 'Speed';
  Result.Speed.Units := 'km/h';
  // the default scale is 0..100
  Result.Speed.Scales[0].Maximum := 260;
  Result.Speed.Scales[0].MajorInterval := 20;
  // a red band at the end of the scale
  oRange := Result.Speed.Scales[0].Ranges.Add;
  oRange.StartValue := 200;
```

```
  oRange.EndValue := 260;
  oRange.Color := $FFE34948;
  // on at 200 km/h, off again below 190 km/h
  oThreshold := Result.Speed.Thresholds.Add;
  oThreshold.Value := 200;
  oThreshold.Hysteresis := 10;
  oThreshold.Direction := gtdBoth;
  Result.Speed.OnThreshold := Result.SpeedThreshold;

  Result.Alarm := TsgcVCLLEDGauge.Create(aForm);
  Result.Alarm.Parent := aForm;
  Result.Alarm.SetBounds(272, 16, 48, 48);
  Result.Alarm.OnColor := $FFE34948;
  Result.Alarm.Blink := True;

  // the needle moves from 0 to 120 in 300 ms
  Result.Speed.Value := 120;
end;
```

Voir aussi

- Types de jauges
- Seuils
- Animation

Démarrage rapide FMX

Les contrôles FireMonkey ont les mêmes propriétés que ceux de la VCL, avec le préfixe TsgcFMX, à partir de Delphi XE6. Installez les packages sgcGauges, ouvrez une fiche FireMonkey et utilisez les composants de la page de palette **SGC Gauges**.

À la conception

1. Déposez un **TsgcFMXRadialGauge** sur une fiche, définissez RadialStyle à grsHalf et Theme à utDark.
2. Définissez Title, Units et ValueFormat, ainsi que le Maximum de la première échelle dans la collection Scales.
3. Déposez un **TsgcFMXLinearGauge**, définissez Orientation à gloVertical et LinearStyle à glsThermometer.
4. Exécutez l'application sur n'importe quelle plateforme FireMonkey.

À savoir en FireMonkey

- Les contrôles dessinent à l'échelle de la scène, ils sont donc nets sur les écrans haute densité et sur les appareils mobiles.
- OnThreshold est appelé dans le thread principal via TThread.Queue, quel que soit le thread qui a écrit la valeur.

En code

```
uses
  System.SysUtils, System.Classes, FMX.Types, FMX.Forms,
  sgcUICore_Theme, sgcGauges_Types, sgcGauges_Model, sgcGauges_FMX;

type
  TBoilerPanel = class(TComponent)
  public
    Pressure: TsgcFMXRadialGauge;
    Level: TsgcFMXLinearGauge;
    procedure Update(aPressure, aLevel: Double);
  end;

procedure TBoilerPanel.Update(aPressure, aLevel: Double);
begin
  // Value may be written from any thread
  Pressure.Value := aPressure;
  Level.Value := aLevel;
end;

function CreateBoilerPanel(aForm: TForm): TBoilerPanel;
begin
  Result := TBoilerPanel.Create(aForm);

  Result.Pressure := TsgcFMXRadialGauge.Create(aForm);
  Result.Pressure.Parent := aForm;
  Result.Pressure.SetBounds(16, 16, 240, 160);
  Result.Pressure.RadialStyle := grsHalf;
  Result.Pressure.Theme := utDark;
  Result.Pressure.Title := 'Pressure';
  Result.Pressure.Units := 'bar';
  Result.Pressure.ValueFormat := '0.0';
  Result.Pressure.Scales[0].Maximum := 10;

  Result.Level := TsgcFMXLinearGauge.Create(aForm);
  Result.Level.Parent := aForm;
  Result.Level.SetBounds(272, 16, 80, 240);
```

```
  Result.Level.Orientation := gloVertical;
  Result.Level.LinearStyle := glsThermometer;
  Result.Level.Theme := utDark;
  Result.Level.Units := '%';

  Result.Update(6.5, 72);
end;
```

Voir aussi

- Démarrage rapide VCL
- Thèmes
- Plateformes

Démarrage rapide sans interface

Les classes de modèle se dessinent sans fenêtre. Un programme console, un serveur web ou un rapport peut créer une jauge, définir ses valeurs et l'enregistrer comme fichier PNG ou SVG. Les unités de modèle n'utilisent ni VCL ni FireMonkey, le même code s'exécute donc sous Windows et sous Linux64.

Étapes

1. Ajoutez sgcGauges_Model et l'unité du type de jauge à la clause uses, par exemple sgcGauges_Radial.
2. Créez le modèle, par exemple `TsgcRadialGauge.Create`, et définissez ses échelles, ses plages et sa valeur comme sur un contrôle.
3. Appelez SaveToPNG ou SaveToSVG avec la taille en pixels logiques et une échelle optionnelle, puis libérez le modèle.

En code

```
uses
  SysUtils, Classes,
  sgcGauges_Types, sgcGauges_Model, sgcGauges_Radial;

procedure SaveSpeedGauge(const aFolder: string; aSpeed: Double);
var
  oGauge: TsgcRadialGauge;
  oRange: TsgcGaugeRange;
begin
  oGauge := TsgcRadialGauge.Create;
  try
    oGauge.Title := 'Speed';
    oGauge.Units := 'km/h';
    oGauge.Scales[0].Maximum := 260;
    oGauge.Scales[0].MajorInterval := 20;
    oRange := oGauge.Scales[0].Ranges.Add;
    oRange.StartValue := 200;
    oRange.EndValue := 260;
    // a headless render always draws the target value, never a frame of a move
    oGauge.Value := aSpeed;
    // 240 x 240 logical pixels; the PNG at scale 2 is 480 x 480 pixels
    oGauge.SaveToPNG(IncludeTrailingPathDelimiter(aFolder) + 'speed.png',
      240, 240, 2);
    oGauge.SaveToSVG(IncludeTrailingPathDelimiter(aFolder) + 'speed.svg',
      240, 240);
  finally
    oGauge.Free;
  end;
end;
```

Un rendu sans interface dessine les valeurs cibles. L'animation et le minutage des images appartiennent aux contrôles. Voir Sortie sans interface.

Voir aussi

- Sortie sans interface
- Architecture

Démarrage rapide C++Builder

Installez les packages sgcGauges pour votre version de C++Builder, ajoutez `sgcUI\Source` et le dossier des unités compilées, par exemple `sgcUI\LibD13\win32` (`sgcUI\LibD13\win64` pour le 64 bits), aux chemins d'inclusion et de bibliothèque du projet, et utilisez les composants de la page de palette **SGC Gauges** exactement comme en Delphi. L'installateur ajoute les deux chemins à l'IDE pour vous. Les en-têtes .hpp et les fichiers .lib générés de sgcGauges et de sgcUICore se trouvent dans ce seul dossier.

À savoir en C++

- Incluez l'en-tête de l'unité qui déclare un contrôle : `sgcGauges_VCL.hpp` pour les contrôles VCL et `sgcGauges_FMX.hpp` pour ceux de FireMonkey. Les collections et les types d'événements se trouvent dans `sgcGauges_Model.hpp` et les énumérations dans `sgcGauges_Types.hpp`.
- Les collections n'ont pas d'indice par défaut en C++ : écrivez `Gauge->Scales->Items[0]` là où Delphi écrit `Gauge.Scales[0]`.
- Les événements sont des closures, leurs gestionnaires sont donc des fonctions membres __fastcall d'une classe.
- Les couleurs sont des valeurs non signées de 32 bits de la forme 0xAARRGGBB.
- Les classes de modèle ont des constructeurs sans argument, comme `new TsgcRadialGauge()`, et vous les supprimez vous-même. Voir Sortie sans interface.

En code

Voir aussi

- Démarrage rapide VCL
- Installation

Jauge radiale

Classes : TsgcVCLRadialGauge (unit sgcGauges_VCL), TsgcFMXRadialGauge (unit sgcGauges_FMX) et le modèle TsgcRadialGauge (unit sgcGauges_Radial)
Un cadran : les échelles sur un arc, une lunette autour de la face, les aiguilles et le texte de la valeur. L'arc et sa taille sont ajustés au contrôle, de sorte qu'une demi-jauge a un bas plat et remplit un contrôle large.
La jauge est présente dans toutes les éditions. La classe VCL descend de TsgcVCLCustomGauge et la classe FireMonkey de TsgcFMXCustomGauge. Toutes deux publient les propriétés du tableau ci-dessous, Theme, OnThreshold et MaxFPS (la limite de fréquence d'images de l'animation, 60 par défaut), ainsi que les propriétés habituelles de leur framework, comme Align, Anchors, Enabled, Visible, PopupMenu, OnClick et les événements de souris. La propriété publique Gauge renvoie le modèle, et SaveToPNG et SaveToSVG enregistrent le contrôle à sa taille cliente multipliée par aScale.

Propriétés

Propriété	Type	Description
RadialStyle	TsgcGaugeRadialStyle	L'arc : grsFull (par défaut, de 135 degrés sur 270), grsHalf (180 sur 180), grsQuarter (180 sur 90) ou grsCustom.
StartAngle	Double	Le début de l'arc avec grsCustom, en degrés, 0 est 3 heures et le sens positif est horaire. 135 par défaut.
SweepAngle	Double	La longueur de l'arc avec grsCustom, en degrés. 270 par défaut.
BezelStyle	TsgcGaugeBezelStyle	gbzNone, gbzThin (par défaut) ou gbzThick.
ValuePosition	TsgcGaugeValuePosition	L'emplacement du texte de la valeur : gvpBottom (par défaut), gvpCenter ou gvpTop.
Scales	TsgcGaugeScales	Les échelles avec leurs graduations, leurs étiquettes et leurs plages de couleur. Voir TsgcGaugeScale.
Pointers	TsgcGaugePointers	Les aiguilles. Voir TsgcGaugePointer.
Thresholds	TsgcGaugeThresholds	Les valeurs qui déclenchent OnThreshold lorsqu'une aiguille les franchit.
Animation	TsgcGaugeAnimation	Enabled, Duration et Easing d'un déplacement d'aiguille.
Title	string	La légende de la jauge.
TitleFont	TsgcUITextStyle	La police du titre. Une Color de 0 suit le thème.
Units	string	Le texte dessiné à côté du texte de la valeur, comme km/h.
ShowValue	Boolean	Dessine le texte de la valeur. True par défaut.
ValueFormat	string	Le masque FormatFloat du texte de la valeur. Vide, il utilise les décimales de l'échelle.
ValueFont	TsgcUITextStyle	La police du texte de la valeur.

Back-groundCo-lor	TsgcUIColor	La couleur derrière la jauge. 0 utilise l'arrière-plan du thème.
FaceColor	TsgcUIColor	La couleur de la face. 0 utilise le thème.
Value	Double	La valeur de l'aiguille 0. Peut être écrite depuis n'importe quel thread. Non enregistrée dans la fiche.

Événements

OnFormatValue(Sender, aPointer, aValue, var aText) remplace le texte de la valeur.

Exemple

Un demi-tachymètre avec une plage rouge :

```
uses
  Classes, Controls, Forms,
  sgcGauges_Types, sgcGauges_Model, sgcGauges_VCL;

function CreateTachometer(aForm: TForm): TsgcVCLRadialGauge;
var
  oRange: TsgcGaugeRange;
begin
  Result := TsgcVCLRadialGauge.Create(aForm);
  Result.Parent := aForm;
  Result.SetBounds(16, 16, 260, 200);
  Result.RadialStyle := grsHalf;
  Result.BezelStyle := gbzThick;
  Result.ValuePosition := gvpCenter;
  Result.Title := 'RPM';
  Result.Units := 'x1000';
  Result.ValueFormat := '0.0';
  Result.Scales[0].Maximum := 8;
  Result.Scales[0].MajorInterval := 1;
  oRange := Result.Scales[0].Ranges.Add;
  oRange.StartValue := 6.5;
  oRange.EndValue := 8;
  oRange.Color := $FFE34948;
  Result.Value := 3.2;
end;
```

Voir aussi

- Types de jauges
- TsgcCustomGauge

Jauge linéaire

Classes : TsgcVCLLinearGauge (unit sgcGauges_VCL), TsgcFMXLinearGauge (unit sgcGauges_FMX) et le modèle TsgcLinearGauge (unit sgcGauges_Linear)
Une jauge horizontale ou verticale dessinée comme une barre, un thermomètre ou un graphique bullet. L'aiguille par défaut est une barre.
La jauge est présente dans toutes les éditions. La classe VCL descend de TsgcVCLCustomGauge et la classe FireMonkey de TsgcFMXCustomGauge. Toutes deux publient les propriétés du tableau ci-dessous, Theme, OnThreshold et MaxFPS (la limite de fréquence d'images de l'animation, 60 par défaut), ainsi que les propriétés habituelles de leur framework, comme Align, Anchors, Enabled, Visible, PopupMenu, OnClick et les événements de souris. La propriété publique Gauge renvoie le modèle, et SaveToPNG et SaveToSVG enregistrent le contrôle à sa taille cliente multipliée par aScale.

Propriétés

Propriété	Type	Description
Orientation	TsgcGaugeOrientation	gloHorizontal (par défaut) va de gauche à droite, gloVertical de bas en haut.
LinearStyle	TsgcGaugeLinearStyle	glsBar (par défaut) : la piste de l'échelle et les aiguilles. glsThermometer : l'échelle 0 est un tube avec un bulbe au minimum, et l'aiguille 0 est dessinée comme le liquide. glsBullet : des bandes épaisses, les plages derrière une barre plus fine, et une ligne Target.
Target	Double	La ligne cible du style bullet, tracée en travers de l'échelle 0 lorsqu'elle se trouve dans l'échelle. 0 par défaut.
BulbSize	Double	Le rayon du bulbe du thermomètre en fraction de la taille de la jauge. 0,06 par défaut.
Scales	TsgcGaugeScales	Les échelles avec leurs graduations, leurs étiquettes et leurs plages de couleur. Voir TsgcGaugeScale.
Pointers	TsgcGaugePointers	Les aiguilles. Voir TsgcGaugePointer.
Thresholds	TsgcGaugeThresholds	Les valeurs qui déclenchent OnThreshold lorsqu'une aiguille les franchit.
Animation	TsgcGaugeAnimation	Enabled, Duration et Easing d'un déplacement d'aiguille.
Title	string	La légende de la jauge.
TitleFont	TsgcUITextStyle	La police du titre. Une Color de 0 suit le thème.
Units	string	Le texte dessiné à côté du texte de la valeur, comme km/h.
ShowValue	Boolean	Dessine le texte de la valeur. True par défaut.

Value-Format	string	Le masque FormatFloat du texte de la valeur. Vide, il utilise les décimales de l'échelle.
Value-Font	TsgcUI-TextStyle	La police du texte de la valeur.
Back-ground-Color	TsgcUICo-lor	La couleur derrière la jauge. 0 utilise l'arrière-plan du thème.
FaceCo-lor	TsgcUICo-lor	La couleur de la face. 0 utilise le thème.
Value	Double	La valeur de l'aiguille 0. Peut être écrite depuis n'importe quel thread. Non enre-gistrée dans la fiche.

Événements

OnFormatValue(Sender, aPointer, aValue, var aText) remplace le texte de la valeur.

Exemple

Un thermomètre vertical et un graphique bullet :

```
uses
  Classes, Controls, Forms,
  sgcGauges_Types, sgcGauges_Model, sgcGauges_VCL;

function CreateThermometer(aForm: TForm): TsgcVCLLinearGauge;
begin
  Result := TsgcVCLLinearGauge.Create(aForm);
  Result.Parent := aForm;
  Result.SetBounds(16, 16, 90, 280);
  Result.Orientation := gloVertical;
  Result.LinearStyle := glsThermometer;
  Result.BulbSize := 0.08;
  Result.Title := 'Oven';
  Result.Units := 'C';
  Result.Scales[0].Minimum := 0;
  Result.Scales[0].Maximum := 250;
  Result.Value := 180;
end;

function CreateBullet(aForm: TForm): TsgcVCLLinearGauge;
begin
  Result := TsgcVCLLinearGauge.Create(aForm);
  Result.Parent := aForm;
  Result.SetBounds(120, 16, 320, 90);
  Result.LinearStyle := glsBullet;
  Result.Title := 'Sales';
  Result.Target := 80; // the target line
  Result.Value := 64;
end;
```

Voir aussi

- Types de jauges
- TsgcCustomGauge

Jauge numérique

Classes : TsgcVCLDigitalGauge (unit sgcGauges_VCL), TsgcFMXDigitalGauge (unit sgcGauges_FMX) et le modèle TsgcDigitalGauge (unit sgcGauges_Digital)
Un afficheur à segments pour la valeur de l'aiguille 0, ou pour un texte. Le type 7 segments affiche les chiffres, le signe moins, la virgule décimale et les deux-points ; le type 14 segments ajoute les lettres ; la matrice de points 5 x 7 affiche l'ASCII 32 à 126. Les segments éteints peuvent être dessinés en filigrane derrière les segments allumés.
Le nombre de positions de glyphes provient de Text, ou du plus large des textes de Minimum et Maximum de l'échelle, sauf si Digits le définit. Une valeur qui ne tient pas affiche des tirets, et un Text qui ne tient pas est coupé après Digits glyphes. Dans les types à segments, un '.' ou une ',' allume le point décimal du glyphe qui le précède et n'occupe pas de position propre.
L'afficheur 7 segments est présent dans toutes les éditions ; en Community, les types 14 segments et matrice de points sont dessinés comme du 7 segments. La classe VCL descend de TsgcVCLCustomGauge et la classe FireMonkey de TsgcFMXCustomGauge. Toutes deux publient les propriétés du tableau ci-dessous, Theme, OnThreshold et MaxFPS (la limite de fréquence d'images de l'animation, 60 par défaut), ainsi que les propriétés habituelles de leur framework, comme Align, Anchors, Enabled, Visible, PopupMenu, OnClick et les événements de souris. La propriété publique Gauge renvoie le modèle, et SaveToPNG et SaveToSVG enregistrent le contrôle à sa taille cliente multipliée par aScale.

Propriétés

Propriété	Type	Description
DisplayType	TsgcGaugeDisplayType	gdt7Segment (par défaut), gdt14Segment ou gdtDotMatrix.
Digits	Integer	Le nombre de positions de glyphes. 0 (par défaut) prend le nombre depuis le texte.
Text	string	Un texte affiché à la place de la valeur. Vide (par défaut), il affiche la valeur.
Skew	Double	Incline les glyphes, en degrés de -30 à 30. Une valeur positive déplace le haut vers la droite. 0 par défaut.
SegmentColor	TsgcUIColor	La couleur des segments allumés. 0 utilise la couleur de ValueFont, sinon la couleur de texte du thème. La ValueColor d'un seuil franchi l'emporte sur elle.
GhostColor	TsgcUIColor	La couleur des segments éteints. 0 utilise la couleur allumée, estompée.
ShowGhost	Boolean	Dessine les segments éteints. True par défaut.
Alignment	TsgcUITextHAlign	La position d'un texte plus court. thaRight par défaut.
Scales	TsgcGaugeScales	Les échelles avec leurs graduations, leurs étiquettes et leurs plages de couleur. Voir TsgcGaugeScale.
Pointers	TsgcGaugePointers	Les aiguilles. Voir TsgcGaugePointer.
Thresholds	TsgcGaugeThresholds	Les valeurs qui déclenchent OnThreshold lorsqu'une aiguille les franchit.
Animation	TsgcGaugeAnimation	Enabled, Duration et Easing d'un déplacement d'aiguille.

Title	string	La légende de la jauge.
TitleFont	TsgcUIText-Style	La police du titre. Une Color de 0 suit le thème.
Units	string	Le texte dessiné à côté du texte de la valeur, comme km/h.
ValueFor-mat	string	Le masque FormatFloat du texte de la valeur. Vide, il utilise les décimales de l'échelle.
ValueFont	TsgcUIText-Style	La police du texte de la valeur.
Back-groundCo-lor	TsgcUIColor	La couleur derrière la jauge. 0 utilise l'arrière-plan du thème.
FaceColor	TsgcUIColor	La couleur de la face. 0 utilise le thème.
Value	Double	La valeur de l'aiguille 0. Peut être écrite depuis n'importe quel thread. Non enregistrée dans la fiche.

Événements

OnFormatValue(Sender, aPointer, aValue, var aText) remplace le texte de la valeur.

Exemple

Un compteur à six chiffres en vert et un texte fixe :

```
uses
  Classes, Controls, Forms,
  sgcGauges_Types, sgcGauges_Model, sgcGauges_VCL;

function CreateCounter(aForm: TForm): TsgcVCLDigitalGauge;
begin
  Result := TsgcVCLDigitalGauge.Create(aForm);
  Result.Parent := aForm;
  Result.SetBounds(16, 16, 280, 100);
  Result.DisplayType := gdt7Segment;
  Result.Digits := 6;
  Result.ValueFormat := '0.00';
  Result.Skew := 8;
  Result.SegmentColor := $FF2ECC71;
  Result.Scales[0].Maximum := 1000;
  Result.Value := 245.5;
end;

procedure ShowText(aGauge: TsgcVCLDigitalGauge);
begin
  // a Text that is not empty is shown instead of the value
  aGauge.Text := '12:45';
end;
```

Voir aussi

- Types de jauges
- TsgcCustomGauge

LED

Classes : TsgcVCLLEDGauge (unit sgcGauges_VCL), TsgcFMXLEDGauge (unit sgcGauges_FMX) et le modèle TsgcLEDGauge (unit sgcGauges_LED)
Un voyant lumineux avec halo, qui peut clignoter. Le voyant est allumé tant que Active est True. Active est une vue de Value <> 0, de sorte qu'une liaison qui écrit 0 ou 1 dans la valeur commute le voyant. La LED conserve toute valeur sauf NaN, sans borne à une échelle, et elle ne s'anime jamais.
La jauge est présente dans toutes les éditions. La classe VCL descend de TsgcVCLCustomGauge et la classe FireMonkey de TsgcFMXCustomGauge. Toutes deux publient les propriétés du tableau ci-dessous, Theme, OnThreshold et MaxFPS (la limite de fréquence d'images de l'animation, 60 par défaut), ainsi que les propriétés habituelles de leur framework, comme Align, Anchors, Enabled, Visible, PopupMenu, OnClick et les événements de souris. La propriété publique Gauge renvoie le modèle, et SaveToPNG et SaveToSVG enregistrent le contrôle à sa taille cliente multipliée par aScale.

Propriétés

Propriété	Type	Description
Active	Boolean	Allume le voyant. L'affecter écrit 1 ou 0 dans la valeur. Non enregistré dans la fiche.
OnColor	TsgcUIColor	La couleur du voyant allumé. 0 est le vert de la palette du thème.
OffColor	TsgcUIColor	La couleur du voyant éteint. 0 utilise la couleur allumée, translucide.
Shape	TsgcGaugeLEDShape	gshCircle (par défaut), gshSquare ou gshRect.
Blink	Boolean	Le voyant allumé s'allume et s'éteint à chaque BlinkInterval. False par défaut. Il n'y a pas de clignotement à la conception.
BlinkInterval	Integer	La période de clignotement en millisecondes. 500 par défaut, au moins 50.
Pointers	TsgcGaugePointers	Les aiguilles. Voir TsgcGaugePointer.
Title	string	La légende de la jauge.
TitleFont	TsgcUITextStyle	La police du titre. Une Color de 0 suit le thème.
BackgroundColor	TsgcUIColor	La couleur derrière la jauge. 0 utilise l'arrière-plan du thème.
Value	Double	La valeur de l'aiguille 0. Peut être écrite depuis n'importe quel thread. Non enregistrée dans la fiche.

Exemple

Un voyant d'alarme rouge qui clignote tant qu'il est allumé :

```
uses
  Classes, Controls, Forms,
  sgcGauges_Types, sgcGauges_Model, sgcGauges_VCL;
```

```
function CreateAlarmLamp(aForm: TForm): TsgcVCLLEDGauge;
begin
  Result := TsgcVCLLEDGauge.Create(aForm);
  Result.Parent := aForm;
  Result.SetBounds(16, 16, 80, 100);
  Result.Title := 'Alarm';
  Result.Shape := gshCircle;
  Result.OnColor := $FFE34948;
  Result.Blink := True;
  Result.BlinkInterval := 400;
end;

procedure SetAlarm(aLamp: TsgcVCLLEDGauge; aOn: Boolean);
begin
  // Active is Value <> 0, so a feed may also write 0 or 1 to Value
  aLamp.Active := aOn;
end;
```

Voir aussi

- Types de jauges
- TsgcCustomGauge
- Barre de LED

Barre de LED

Classes : TsgcVCLLEDBarGauge (unit sgcGauges_VCL), TsgcFMXLEDBarGauge (unit sgcGauges_FMX) et le modèle TsgcLEDBarGauge (unit sgcGauges_LED)
Une rangée ou une colonne de segments de LED qui divisent l'échelle de l'aiguille 0. Un segment est allumé lorsque son centre se trouve entre l'origine de l'aiguille et sa valeur. Un segment prend la couleur de la plage de l'échelle où il se trouve, sinon vert, ambre et rouge par tiers de l'échelle. Le maintien du pic garde le segment le plus haut allumé un moment après la baisse du niveau.
La jauge est présente dans toutes les éditions. La classe VCL descend de TsgcVCLCustomGauge et la classe FireMonkey de TsgcFMXCustomGauge. Toutes deux publient les propriétés du tableau ci-dessous, Theme, OnThreshold et MaxFPS (la limite de fréquence d'images de l'animation, 60 par défaut), ainsi que les propriétés habituelles de leur framework, comme Align, Anchors, Enabled, Visible, PopupMenu, OnClick et les événements de souris. La propriété publique Gauge renvoie le modèle, et SaveToPNG et SaveToSVG enregistrent le contrôle à sa taille cliente multipliée par aScale.

Propriétés

Propriété	Type	Description
Segments	Integer	Le nombre de segments, de 1 à 100. 10 par défaut.
SegmentGap	Double	L'espace entre deux segments en fraction d'un segment, jusqu'à 0,9. 0,2 par défaut.
Orientation	TsgcGaugeOrientation	gloHorizontal (par défaut) de gauche à droite, gloVertical de bas en haut.
PeakHold	Boolean	Garde le segment du pic allumé. False par défaut.
PeakHoldTime	Integer	La durée de maintien du pic, en millisecondes. 1000 par défaut.
Scales	TsgcGaugeScales	Les échelles avec leurs graduations, leurs étiquettes et leurs plages de couleur. Voir TsgcGaugeScale.
Pointers	TsgcGaugePointers	Les aiguilles. Voir TsgcGaugePointer.
Animation	TsgcGaugeAnimation	Enabled, Duration et Easing d'un déplacement d'aiguille.
Title	string	La légende de la jauge.
TitleFont	TsgcUITextStyle	La police du titre. Une Color de 0 suit le thème.
BackgroundColor	TsgcUIColor	La couleur derrière la jauge. 0 utilise l'arrière-plan du thème.
Value	Double	La valeur de l'aiguille 0. Peut être écrite depuis n'importe quel thread. Non enregistrée dans la fiche.

Exemple

Une barre de niveau verticale de 20 segments avec une plage bleue et un maintien du pic :

```
uses
  Classes, Controls, Forms,
  sgcGauges_Types, sgcGauges_Model, sgcGauges_VCL;

function CreateLevelBar(aForm: TForm): TsgcVCLLEDBarGauge;
var
  oRange: TsgcGaugeRange;
begin
  Result := TsgcVCLLEDBarGauge.Create(aForm);
  Result.Parent := aForm;
  Result.SetBounds(16, 16, 60, 260);
  Result.Orientation := gloVertical;
  Result.Segments := 20;
  Result.SegmentGap := 0.3;
  Result.PeakHold := True;
  Result.PeakHoldTime := 1500;
  // a range colours the segments inside it, the others keep the
  // green, amber and red thirds
  oRange := Result.Scales[0].Ranges.Add;
  oRange.StartValue := 0;
  oRange.EndValue := 60;
  oRange.Color := $FF2A78D6;
  Result.Value := 72;
end;
```

Voir aussi

- Types de jauges
- TsgcCustomGauge
- Plages

Anneau de progression

Classes : TsgcVCLProgressRingGauge (unit sgcGauges_VCL), TsgcFMXProgressRingGauge (unit sgcGauges_FMX) et le modèle TsgcProgressRingGauge (unit sgcGauges_ProgressRing)
Un anneau qui se remplit dans le sens horaire à partir de 12 heures avec la valeur de l'aiguille 0, avec le pourcentage au milieu. La barre prend la Color de l'aiguille, sinon la couleur de la plage qui contient la valeur, sinon la couleur d'accent du thème. L'aiguille par défaut est une barre.
La jauge est présente dans toutes les éditions. La classe VCL descend de TsgcVCLCustomGauge et la classe FireMonkey de TsgcFMXCustomGauge. Toutes deux publient les propriétés du tableau ci-dessous, Theme, OnThreshold et MaxFPS (la limite de fréquence d'images de l'animation, 60 par défaut), ainsi que les propriétés habituelles de leur framework, comme Align, Anchors, Enabled, Visible, PopupMenu, OnClick et les événements de souris. La propriété publique Gauge renvoie le modèle, et SaveToPNG et SaveToSVG enregistrent le contrôle à sa taille cliente multipliée par aScale.

Propriétés

Propriété	Type	Description
Thickness	Double	La largeur de l'anneau en fraction de la taille de la jauge. 0,15 par défaut.
TrackColor	TsgcUIColor	La couleur de l'anneau vide. 0 utilise le thème.
RoundCaps	Boolean	Arrondit les deux extrémités d'une barre plus courte que l'anneau complet. True par défaut.
ShowPercent	Boolean	True (par défaut) affiche le pourcentage de la valeur dans l'échelle, avec ValueFormat ou '0' et un signe %. False affiche le texte de la valeur et Units.
Scales	TsgcGaugeScales	Les échelles avec leurs graduations, leurs étiquettes et leurs plages de couleur. Voir TsgcGaugeScale.
Pointers	TsgcGaugePointers	Les aiguilles. Voir TsgcGaugePointer.
Thresholds	TsgcGaugeThresholds	Les valeurs qui déclenchent OnThreshold lorsqu'une aiguille les franchit.
Animation	TsgcGaugeAnimation	Enabled, Duration et Easing d'un déplacement d'aiguille.
Title	string	La légende de la jauge.
TitleFont	TsgcUITextStyle	La police du titre. Une Color de 0 suit le thème.
Units	string	Le texte dessiné à côté du texte de la valeur, comme km/h.
ShowValue	Boolean	Dessine le texte de la valeur. True par défaut.
ValueFormat	string	Le masque FormatFloat du texte de la valeur. Vide, il utilise les décimales de l'échelle.
ValueFont	TsgcUITextStyle	La police du texte de la valeur.

BackgroundColor	TsgcUIColor	La couleur derrière la jauge. 0 utilise l'arrière-plan du thème.
Value	Double	La valeur de l'aiguille 0. Peut être écrite depuis n'importe quel thread. Non enregistrée dans la fiche.

Événements

OnFormatValue(Sender, aPointer, aValue, var aText) remplace le texte de la valeur lorsque ShowPercent est False.

Exemple

La progression d'un envoi, avec le nombre d'octets comme valeur :

```
uses
  Classes, Controls, Forms,
  sgcGauges_Types, sgcGauges_Model, sgcGauges_VCL;

function CreateUploadRing(aForm: TForm): TsgcVCLProgressRingGauge;
begin
  Result := TsgcVCLProgressRingGauge.Create(aForm);
  Result.Parent := aForm;
  Result.SetBounds(16, 16, 160, 160);
  Result.Title := 'Upload';
  Result.Thickness := 0.12;
  Result.RoundCaps := True;
  Result.ShowPercent := True;
  // the bytes sent: the ring shows the percentage of the scale
  Result.Scales[0].Maximum := 5242880;
  Result.Value := 1310720; // 25%
end;
```

Voir aussi

- Types de jauges
- TsgcCustomGauge

Vumètre

Classes : TsgcVCLVUMeterGauge (unit sgcGauges_VCL), TsgcFMXVUMeterGauge (unit sgcGauges_FMX) et le modèle TsgcVUMeterGauge (unit sgcGauges_VUMeter)
Un indicateur de niveau audio. L'échelle par défaut va de -20 à +3 dB avec la correspondance en décibels, de sorte qu'une position suit l'amplitude, avec les graduations à l'extérieur et une plage rouge à partir de 0 dB. L'aiguille se déplace en amplitude, comme un vrai vumètre. Tant que MajorInterval vaut 0, l'échelle affiche les repères VU classiques.
La jauge nécessite l'édition Registered ou Trial : en Community, le modèle ne dessine rien et les contrôles ne sont pas installés. Voir Éditions. La classe VCL descend de TsgcVCLCustomGauge et la classe FireMonkey de TsgcFMXCustomGauge. Toutes deux publient les propriétés du tableau ci-dessous, Theme, OnThreshold et MaxFPS (la limite de fréquence d'images de l'animation, 60 par défaut), ainsi que les propriétés habituelles de leur framework, comme Align, Anchors, Enabled, Visible, PopupMenu, OnClick et les événements de souris. La propriété publique Gauge renvoie le modèle, et SaveToPNG et SaveToSVG enregistrent le contrôle à sa taille cliente multipliée par aScale.

Propriétés

Propriété	Type	Description
MeterStyle	TsgcGaugeVUMeterStyle	gvmNeedle (par défaut) : une aiguille sur un arc de 90 degrés. gvmBar : une barre horizontale sur une bande avec la piste et les plages.
PeakHold	Boolean	Affiche le pic du niveau. False par défaut.
PeakHoldTime	Integer	La durée de maintien du pic avant sa chute, en millisecondes. 1500 par défaut.
PeakDecay	Double	La vitesse de chute du pic après le maintien, en dB par seconde. 20 par défaut. 0 fait tomber le pic immédiatement après le maintien.
Scales	TsgcGaugeScales	Les échelles avec leurs graduations, leurs étiquettes et leurs plages de couleur. Voir TsgcGaugeScale.
Pointers	TsgcGaugePointers	Les aiguilles. Voir TsgcGaugePointer.
Thresholds	TsgcGaugeThresholds	Les valeurs qui déclenchent OnThreshold lorsqu'une aiguille les franchit.
Animation	TsgcGaugeAnimation	Enabled, Duration et Easing d'un déplacement d'aiguille.
Title	string	La légende de la jauge.
TitleFont	TsgcUITextStyle	La police du titre. Une Color de 0 suit le thème.
Units	string	Le texte dessiné à côté du texte de la valeur, comme km/h.
ShowValue	Boolean	Dessine le texte de la valeur. True par défaut.
ValueFormat	string	Le masque FormatFloat du texte de la valeur. Vide, il utilise les décimales de l'échelle.
ValueFont	TsgcUITextStyle	La police du texte de la valeur.

BackgroundColor	TsgcUIColor	La couleur derrière la jauge. 0 utilise l'arrière-plan du thème.
FaceColor	TsgcUIColor	La couleur de la face. 0 utilise le thème.
Value	Double	La valeur de l'aiguille 0. Peut être écrite depuis n'importe quel thread. Non enregistrée dans la fiche.

Méthodes

ResetPeak efface le pic.

Événements

OnFormatValue(Sender, aPointer, aValue, var aText) remplace le texte de la valeur.

Exemple

```
uses
  Classes, Controls, Forms,
  sgcGauges_Types, sgcGauges_Model, sgcGauges_VCL;

function CreateVUMeter(aForm: TForm): TsgcVCLVUMeterGauge;
begin
  Result := TsgcVCLVUMeterGauge.Create(aForm);
  Result.Parent := aForm;
  Result.SetBounds(16, 16, 280, 180);
  Result.Title := 'Left';
  Result.MeterStyle := gvmNeedle;
  Result.PeakHold := True;
  Result.PeakHoldTime := 1500;
  Result.PeakDecay := 20;
  // the default scale runs from -20 to +3 dB with a red range above 0
  Result.Value := -6;
end;

procedure NewLevel(aMeter: TsgcVCLVUMeterGauge; aDB: Double);
begin
  aMeter.Value := aDB; // any thread
end;
```

Voir aussi

- Types de jauges
- TsgcCustomGauge

Horloge

Classes : TsgcVCLClockGauge (unit sgcGauges_VCL), TsgcFMXClockGauge (unit sgcGauges_FMX) et le modèle TsgcClockGauge (unit sgcGauges_Clock)
Une horloge analogique avec 60 graduations de minutes, 12 chiffres et les aiguilles des heures, des minutes et des secondes. Elle affiche Time ou, avec Live, l'heure réelle : l'heure locale, ou l'heure UTC plus UTCOffset minutes avec UseUTCOffset. Une horloge live se redessine chaque seconde, ou chaque minute sans ShowSeconds. Les chiffres utilisent le LabelFont de l'échelle 0. L'horloge n'a ni Value ni aiguilles de valeur.
La jauge nécessite l'édition Registered ou Trial : en Community, le modèle ne dessine rien et les contrôles ne sont pas installés. Voir Éditions. La classe VCL descend de TsgcVCLCustomGauge et la classe FireMonkey de TsgcFMXCustomGauge. Toutes deux publient les propriétés du tableau ci-dessous, Theme, OnThreshold et MaxFPS (la limite de fréquence d'images de l'animation, 60 par défaut), ainsi que les propriétés habituelles de leur framework, comme Align, Anchors, Enabled, Visible, PopupMenu, OnClick et les événements de souris. La propriété publique Gauge renvoie le modèle, et SaveToPNG et SaveToSVG enregistrent le contrôle à sa taille cliente multipliée par aScale.

Propriétés

Propriété	Type	Description
Time	TDateTime	L'heure affichée tant que Live est False.
Live	Boolean	Affiche l'heure réelle. False par défaut.
UseUTCOffset	Boolean	Une horloge live affiche l'heure UTC plus UTCOffset au lieu de l'heure locale. False par défaut.
UTCOffset	Integer	Le décalage par rapport à UTC en minutes, comme 60 ou -300. 0 par défaut.
Numerals	TsgcGaugeClockNumerals	gcnArabic (par défaut), gcnRoman ou gcnNone.
ShowSeconds	Boolean	Dessine l'aiguille des secondes. True par défaut.
SmoothSeconds	Boolean	L'aiguille des secondes balaie au lieu d'avancer par saccades. False par défaut.
HourHandColor	TsgcUIColor	La couleur de l'aiguille des heures. 0 utilise le thème.
MinuteHandColor	TsgcUIColor	La couleur de l'aiguille des minutes. 0 utilise le thème.
SecondHandColor	TsgcUIColor	La couleur de l'aiguille des secondes. 0 utilise le thème.
Scales	TsgcGaugeScales	Les échelles avec leurs graduations, leurs étiquettes et leurs plages de couleur. Voir TsgcGaugeScale.
Title	string	La légende de la jauge.
TitleFont	TsgcUITextStyle	La police du titre. Une Color de 0 suit le thème.
ValueFont	TsgcUITextStyle	La police du texte de la valeur.

BackgroundColor	TsgcUIColor	La couleur derrière la jauge. 0 utilise l'arrière-plan du thème.
FaceColor	TsgcUIColor	La couleur de la face. 0 utilise le thème.

Le modèle a aussi une propriété publique TimeSource, une fonction qui renvoie l'heure réelle. Lorsqu'elle est assignée, une horloge live la lit à la place de l'horloge système, ce qui donne des images fixes dans les tests.

Exemple

Une horloge live pour UTC+5:30 avec des chiffres romains, et une heure fixe :

```
uses
  Classes, Controls, Forms, SysUtils,
  sgcGauges_Types, sgcGauges_Model, sgcGauges_VCL;

function CreateWorldClock(aForm: TForm): TsgcVCLClockGauge;
begin
  Result := TsgcVCLClockGauge.Create(aForm);
  Result.Parent := aForm;
  Result.SetBounds(16, 16, 200, 200);
  Result.Title := 'UTC+5:30';
  Result.Live := True;
  Result.UseUTCOffset := True;
  Result.UTCOffset := 330; // minutes
  Result.Numerals := gcnRoman;
  Result.ShowSeconds := True;
  Result.SmoothSeconds := True;
  Result.SecondHandColor := $FFE34948;
end;

procedure ShowFixedTime(aClock: TsgcVCLClockGauge);
begin
  aClock.Live := False;
  aClock.Time := EncodeTime(10, 10, 30, 0);
end;
```

Voir aussi

- Types de jauges
- TsgcCustomGauge

Boussole

Classes : TsgcVCLCompassGauge (unit sgcGauges_VCL), TsgcFMXCompassGauge (unit sgcGauges_FMX) et le modèle TsgcCompassGauge (unit sgcGauges_Compass)
Un indicateur de cap. Heading est la valeur de l'aiguille 0 sur une échelle de 0 à 360 degrés, dans le sens horaire à partir du nord. Il boucle au lieu d'être borné (450 vaut 90, -30 vaut 330), et un déplacement s'anime selon l'arc le plus court : de 359 à 1, la rotation est de 2 degrés. La rose a une graduation tous les 10 degrés, les points cardinaux et les étiquettes de degrés tous les 30 degrés, dans le LabelFont de l'échelle 0. Le texte du cap affiche des degrés entiers et un signe degré, sauf si ValueFormat est défini.
La jauge nécessite l'édition Registered ou Trial : en Community, le modèle ne dessine rien et les contrôles ne sont pas installés. Voir Éditions. La classe VCL descend de TsgcVCLCustomGauge et la classe FireMonkey de TsgcFMXCustomGauge. Toutes deux publient les propriétés du tableau ci-dessous, Theme, OnThreshold et MaxFPS (la limite de fréquence d'images de l'animation, 60 par défaut), ainsi que les propriétés habituelles de leur framework, comme Align, Anchors, Enabled, Visible, PopupMenu, OnClick et les événements de souris. La propriété publique Gauge renvoie le modèle, et SaveToPNG et SaveToSVG enregistrent le contrôle à sa taille cliente multipliée par aScale.

Propriétés

Propriété	Type	Description
Heading	Double	Le cap en degrés, la valeur de l'aiguille 0. Peut être écrit depuis n'importe quel thread. Non enregistré dans la fiche.
CompassStyle	TsgcGaugeCompassStyle	gcsRotatingCard (par défaut) : la rose tourne et le cap se lit sous un repère fixe en haut. gcsRotatingNeedle : la rose est fixe et une aiguille indique le cap.
ShowCardinals	Boolean	Dessine N, E, S, O et les points intercardinaux. True par défaut.
ShowDegrees	Boolean	Dessine les étiquettes de degrés de la rose. True par défaut. Le texte du cap suit ShowValue.
TargetHeading	Double	Le cap du repère cible. 0 par défaut.
ShowTarget	Boolean	Dessine le repère cible sur le bord, dans la couleur d'accent du thème. False par défaut.
Scales	TsgcGaugeScales	Les échelles avec leurs graduations, leurs étiquettes et leurs plages de couleur. Voir TsgcGaugeScale.
Pointers	TsgcGaugePointers	Les aiguilles. Voir TsgcGaugePointer.
Animation	TsgcGaugeAnimation	Enabled, Duration et Easing d'un déplacement d'aiguille.
Title	string	La légende de la jauge.
TitleFont	TsgcUITextStyle	La police du titre. Une Color de 0 suit le thème.
ShowValue	Boolean	Dessine le texte de la valeur. True par défaut.

ValueFormat	string	Le masque FormatFloat du texte de la valeur. Vide, il utilise les décimales de l'échelle.
ValueFont	TsgcUITextStyle	La police du texte de la valeur.
BackgroundColor	TsgcUIColor	La couleur derrière la jauge. 0 utilise l'arrière-plan du thème.
FaceColor	TsgcUIColor	La couleur de la face. 0 utilise le thème.

Événements

OnFormatValue(Sender, aPointer, aValue, var aText) remplace le texte du cap.

Exemple

```
uses
  Classes, Controls, Forms,
  sgcGauges_Types, sgcGauges_Model, sgcGauges_VCL;

function CreateCompass(aForm: TForm): TsgcVCLCompassGauge;
begin
  Result := TsgcVCLCompassGauge.Create(aForm);
  Result.Parent := aForm;
  Result.SetBounds(16, 16, 220, 220);
  Result.Title := 'Heading';
  Result.CompassStyle := gcsRotatingCard;
  Result.ShowCardinals := True;
  Result.ShowTarget := True;
  Result.TargetHeading := 270;
  Result.Heading := 350;
end;

procedure Turn(aCompass: TsgcVCLCompassGauge);
begin
  // from 350 the card turns 20 degrees through north, the shortest arc
  aCompass.Heading := 10;
end;
```

Voir aussi

- Types de jauges
- TsgcCustomGauge

Annunciator

Classes : TsgcVCLAnnunciatorGauge (unit sgcGauges_VCL), TsgcFMXAnnunciatorGauge (unit sgcGauges_FMX) et le modèle TsgcAnnunciatorGauge (unit sgcGauges_Annunciator)
Un panneau d'alarmes : une grille de tuiles, remplie ligne par ligne sous le titre. Chaque tuile a une Caption, un State (gasNormal, gasAdvisory, gasWarning ou gasAlarm) et un indicateur Acknowledged, et prend la couleur de son état. Un avertissement ou une alarme non acquittés clignotent ; un état informatif reste fixe. Un changement de State efface Acknowledged, de sorte que chaque nouvel état demande son propre acquittement. Les légendes utilisent la taille et la graisse de ValueFont et sont coupées à la largeur de la tuile. L'annunciator n'a ni échelles, ni aiguilles, ni seuils.
La jauge nécessite l'édition Registered ou Trial : en Community, le modèle ne dessine rien et les contrôles ne sont pas installés. Voir Éditions. La classe VCL descend de TsgcVCLCustomGauge et la classe FireMonkey de TsgcFMXCustomGauge. Toutes deux publient les propriétés du tableau ci-dessous, Theme, OnThreshold et MaxFPS (la limite de fréquence d'images de l'animation, 60 par défaut), ainsi que les propriétés habituelles de leur framework, comme Align, Anchors, Enabled, Visible, PopupMenu, OnClick et les événements de souris. La propriété publique Gauge renvoie le modèle, et SaveToPNG et SaveToSVG enregistrent le contrôle à sa taille cliente multipliée par aScale.

Propriétés

Propriété	Type	Description
Tiles	TsgcGaugeAnnunciatorTiles	Les tuiles, éléments de TsgcGaugeAnnunciatorTile avec Caption, State et Acknowledged.
Columns	Integer	Le nombre de tuiles par ligne. 4 par défaut.
FlashInterval	Integer	La période de clignotement en millisecondes. 500 par défaut, au moins 50.
NormalColor	TsgcUIColor	La couleur d'une tuile normale. 0 utilise la couleur de face du thème.
AdvisoryColor	TsgcUIColor	La couleur d'une tuile informative. 0 est bleu.
WarningColor	TsgcUIColor	La couleur d'une tuile d'avertissement. 0 est ambre.
AlarmColor	TsgcUIColor	La couleur d'une tuile d'alarme. 0 est rouge.
Title	string	La légende de la jauge.
TitleFont	TsgcUITextStyle	La police du titre. Une Color de 0 suit le thème.
ValueFont	TsgcUITextStyle	La police du texte de la valeur.
BackgroundColor	TsgcUIColor	La couleur derrière la jauge. 0 utilise l'arrière-plan du thème.
FaceColor	TsgcUIColor	La couleur de la face. 0 utilise le thème.

Méthodes

Méthode	Description
Acknowledge(aIndex)	Acquitte une tuile qui n'est pas normale : elle cesse de clignoter.
AcknowledgeAll	Acquitte toutes les tuiles qui ne sont pas normales.
TileAt(X, Y)	L'indice de la tuile à un point du contrôle, ou -1.
HoverTile	L'indice de la tuile sous la souris, ou -1.

Événements

OnTileClick(Sender, aIndex) de type TsgcGaugeTileClickEvent (unit sgcGauges_Annunciator) se déclenche lorsque le bouton gauche est relâché sur une tuile.

Exemple

Trois tuiles ; un clic acquitte une tuile :

```
uses
  Classes, Controls, Forms,
  sgcGauges_Types, sgcGauges_Model, sgcGauges_Annunciator, sgcGauges_VCL;

type
  TAlarmPanel = class(TComponent)
  public
    Panel: TsgcVCLAnnunciatorGauge;
    procedure TileClick(Sender: TObject; aIndex: Integer);
  end;

procedure TAlarmPanel.TileClick(Sender: TObject; aIndex: Integer);
begin
  // a click acknowledges the tile: it stops flashing and keeps its state
  Panel.Acknowledge(aIndex);
end;

function CreateAlarmPanel(aForm: TForm): TAlarmPanel;
var
  oTile: TsgcGaugeAnnunciatorTile;
begin
  Result := TAlarmPanel.Create(aForm);
  Result.Panel := TsgcVCLAnnunciatorGauge.Create(aForm);
  Result.Panel.Parent := aForm;
  Result.Panel.SetBounds(16, 16, 400, 120);
  Result.Panel.Columns := 3;
  Result.Panel.FlashInterval := 400;
  Result.Panel.Tiles.Add.Caption := 'PUMP 1';
  Result.Panel.Tiles.Add.Caption := 'PUMP 2';
  oTile := Result.Panel.Tiles.Add;
  oTile.Caption := 'OIL PRESS';
  oTile.State := gasAlarm;
  Result.Panel.Tiles[1].State := gasAdvisory;
  Result.Panel.OnTileClick := Result.TileClick;
end;
```

Voir aussi

- Types de jauges

- TsgcCustomGauge

TsgcCustomGauge

Unité : sgcGauges_Model
La classe de base des dix modèles de jauge, un TPersistent sans fenêtre. Un contrôle hôte crée le modèle de son type et le renvoie dans sa propriété Gauge ; un programme sans interface utilisateur crée lui-même le modèle, par exemple avec TsgcRadialGauge.Create, et le libère. Un nouveau modèle a une échelle de 0 à 100 et une aiguille.

Propriétés

Propriété	Type	Description
Scales	TsgcGaugeScales	Les échelles. Voir TsgcGaugeScale. Community n'utilise que la première.
Pointers	TsgcGaugePointers	Les aiguilles. Voir TsgcGaugePointer. Community n'utilise que la première.
Thresholds	TsgcGaugeThresholds	Les seuils. Voir TsgcGaugeThreshold.
Animation	TsgcGaugeAnimation	Enabled (True par défaut), Duration en millisecondes (300 par défaut) et Easing (geCubicOut par défaut).
Theme	TsgcUITheme	utLight (par défaut) ou utDark.
Title, TitleFont	string, TsgcUITextStyle	La légende de la jauge et sa police.
Units	string	Le texte dessiné à côté du texte de la valeur. ValueToText ne l'ajoute pas.
ShowValue	Boolean	Dessine le texte de la valeur. True par défaut.
ValueFormat	string	Le masque FormatFloat du texte de la valeur. Vide, il utilise les décimales du pas principal de l'échelle.
ValueFont	TsgcUITextStyle	La police du texte de la valeur. Une Color de 0 suit le thème.
BackgroundColor, FaceColor	TsgcUIColor	La couleur derrière la jauge et la couleur de la face. 0 utilise le thème.
Value	Double	La valeur de l'aiguille 0, identique à SetValue(0, Value).
FormatSettings	TFormatSettings	Public. Les réglages du texte de la valeur et des étiquettes, avec '.' comme séparateur décimal par défaut.
StaticStamp	Integer	Public, lecture seule. Incrémenté à chaque changement statique ; un hôte le compare pour savoir si sa couche statique en cache est à jour.
Horloge	TsgcGaugeClockFunc	Public. Une fonction qui renvoie l'heure en millisecondes pour l'animation. nil utilise l'horloge système.
Owner	TPersistent	Public. Le contrôle hôte, ou nil pour un modèle sans interface.
Designing	Boolean	Public. Défini par les hôtes à la conception, où rien ne s'anime.

Méthodes

Méthode	Description
SetValue(aPointer, aValue, aAnimate = True)	Écrit la valeur d'une aiguille, avec une animation lorsque aAnimate et Animation.Enabled sont True. Thread-safe.
BeginUpdate, EndUpdate	Regroupe les changements entre les deux en un seul OnChange.
DisplayValue(aPointer, aNowMs)	La valeur dessinée d'une aiguille à un instant de Clock, pendant une animation.
ValueToText(aPointer, aValue)	Le texte de la valeur d'une valeur, avec ValueFormat et OnFormat-Value.
IsAnimating(aNowMs), NextFrameMs(aNowMs)	Indique si une aiguille bouge, et quand la prochaine image tempori-sée (un clignotement, un tic d'horloge) est due.
ActiveScaleCount, ActivePointer-Count, PointerScale(aPointer)	Les échelles et les aiguilles que l'édition dessine, et l'échelle d'une aiguille.
Render, RenderStatic, RenderDyna-mic	Dessinent les deux couches, ou une seule, sur un TsgcUICanvas de aWidth x aHeight pixels logiques à l'échelle aScale.
SaveToPNG, SaveToSVG	Enregistrent la jauge dans un flux ou un fichier à aWidth x aHeight pixels logiques, à l'échelle aScale.

Événements

Événement	Description
OnChange(Sender, aKind)	Un changement : gckStatic pour la couche statique, gckDynamic pour une valeur ou un déplacement d'aiguille. Appelé dans le thread du changement.
OnThreshold(Sender, aThre-shold, aValue, aRising)	Une aiguille a franchi un seuil. Sur un modèle, il est appelé dans le thread qui a écrit la valeur ; les contrôles l'appellent dans le thread principal.
OnFormatValue(Sender, aPointer, aValue, var aText)	Remplace le texte de la valeur.

Exemple

Une jauge sans interface avec deux aiguilles qui compte ses changements :

```
uses
  SysUtils, Classes,
  sgcGauges_Types, sgcGauges_Model, sgcGauges_Radial;

type
  TGaugeWatcher = class
  public
    StaticChanges, DynamicChanges: Integer;
    procedure GaugeChange(Sender: TObject; aKind: TsgcGaugeChangeKind);
  end;

procedure TGaugeWatcher.GaugeChange(Sender: TObject;
  aKind: TsgcGaugeChangeKind);
begin
  if aKind = gckStatic then
```

```
    Inc(StaticChanges)
  else
    Inc(DynamicChanges);
end;

procedure TwoPointers(aWatcher: TGaugeWatcher);
var
  oGauge: TsgcRadialGauge;
begin
  oGauge := TsgcRadialGauge.Create;
  try
    oGauge.OnChange := aWatcher.GaugeChange;
    oGauge.BeginUpdate;
    try
      // one static OnChange for the whole block
      oGauge.Title := 'Pressure';
      oGauge.Units := 'bar';
      oGauge.Scales[0].Maximum := 10;
      oGauge.Pointers.Add.Kind := gpkMarker;
    finally
      oGauge.EndUpdate;
    end;
    oGauge.SetValue(0, 6.5, False); // dynamic only
    oGauge.SetValue(1, 8, False);
    WriteLn(oGauge.ValueToText(0, oGauge.Value)); // 6.5
    oGauge.SaveToSVG('pressure.svg', 200, 200);
  finally
    oGauge.Free;
  end;
end;
```

Voir aussi

- Architecture
- Sortie sans interface
- Animation

TsgcGaugeScale et TsgcGaugeRange

Unité : sgcGauges_Model
TsgcGaugeScale est un élément de la collection Scales (TsgcGaugeScales) d'une jauge. Une échelle fait correspondre des valeurs à des positions sur l'arc ou l'axe et dessine sa piste, ses graduations, ses étiquettes et ses plages de couleur. TsgcGaugeRange est un élément de la collection Ranges (TsgcGaugeRanges) d'une échelle. Les tailles sont des fractions de la taille de la jauge.

TsgcGaugeScale

Propriété	Type	Description
Minimum, Maximum	Double	La plage de valeurs. 0 et 100 par défaut. Définissez-les avant les valeurs, qui sont bornées à l'écriture.
Mapping	TsgcGaugeScale-Mapping	gsmLinear (par défaut) ou gsmDecibel, où une position suit l'amplitude.
Reversed	Boolean	Parcourt l'échelle de Maximum à Minimum. False par défaut.
Radius	Double	Le rayon extérieur de l'échelle sur un cadran. 1 par défaut.
Offset	Double	La position de la bande en travers de l'axe d'une jauge linéaire. 0 par défaut.
Thickness	Double	La largeur de la piste. 0,05 par défaut.
MajorInterval	Double	Le pas entre les graduations étiquetées. 0 (par défaut) choisit un pas.
MinorCount	Integer	Les graduations secondaires entre deux graduations principales. 4 par défaut.
TickPlacement	TsgcGaugeTick-Placement	gtpInside (par défaut), gtpOutside ou gtpCross.
TickColor	TsgcUIColor	La couleur des graduations. 0 utilise le thème.
ShowLabels	Boolean	Dessine les étiquettes des graduations principales. True par défaut.
LabelFormat	string	Le masque FormatFloat des étiquettes. Vide, il utilise les décimales du pas.
LabelFont	TsgcUITextStyle	La police des étiquettes.
Ranges	TsgcGauge-Ranges	Les plages de couleur.
Visible	Boolean	Dessine l'échelle. True par défaut.

MajorStep renvoie le pas utilisé et ValueToRatio(aValue) la position d'une valeur de 0 à 1.

TsgcGaugeRange

Propriété	Type	Description
StartValue, EndValue	Double	Les valeurs que la plage couvre.
Color	TsgcUIColor	La couleur de la bande.
Thickness	Double	La largeur de la bande. 0 utilise l'épaisseur de l'échelle.
Caption	string	Un nom pour la plage. Il n'est pas dessiné.

Exemple

```
uses
  SysUtils, Classes,
  sgcGauges_Types, sgcGauges_Model, sgcGauges_Radial;

procedure ConfigureScales(aGauge: TsgcRadialGauge);
var
  oScale: TsgcGaugeScale;
  oRange: TsgcGaugeRange;
begin
  aGauge.BeginUpdate;
  try
    oScale := aGauge.Scales[0];
    oScale.Minimum := 0;
    oScale.Maximum := 220;
    oScale.MajorInterval := 20;
    oScale.MinorCount := 3;
    oScale.TickPlacement := gtpInside;
    oScale.LabelFormat := '0';
    oRange := oScale.Ranges.Add;
    oRange.StartValue := 180;
    oRange.EndValue := 220;
    oRange.Color := $FFE34948;
    oRange.Thickness := 0.08;
    // an inner scale in miles per hour (Registered: more than one scale)
    oScale := aGauge.Scales.Add;
    oScale.Maximum := 140;
    oScale.Radius := 0.7;
    oScale.TickPlacement := gtpOutside;
    aGauge.Pointers.Add.ScaleIndex := 1;
  finally
    aGauge.EndUpdate;
  end;
end;
```

Voir aussi

- Échelles
- Plages
- TsgcCustomGauge

TsgcGaugePointer, TsgcGaugeThreshold et TsgcGaugeAnimation

Unité : sgcGauges_Model
TsgcGaugePointer est un élément de la collection Pointers (TsgcGaugePointers), TsgcGaugeThreshold un élément de la collection Thresholds (TsgcGaugeThresholds), et TsgcGaugeAnimation la propriété Animation d'une jauge.

TsgcGaugePointer

Propriété	Type	Description
Kind	TsgcGaugePointerKind	gpkNeedle (par défaut), gpkMarker, gpkBar ou gpkRange (une bande de Value à EndValue).
ScaleIndex	Integer	L'échelle utilisée par l'aiguille. 0 par défaut.
Value	Double	La valeur. Thread-safe ; l'écrire déclenche une animation.
EndValue	Double	La fin d'une aiguille gpkRange. Thread-safe.
Origin	Double	Le point de départ d'une barre. Non défini (NaN) signifie le Minimum de l'échelle. OriginValue renvoie la valeur utilisée.
Color	TsgcUIColor	La couleur. 0 utilise le thème.
Width, Length, Offset	Double	La taille et la position. Valeurs par défaut 0,04, 0,9 et 0.
NeedleStyle	TsgcGaugeNeedleStyle	gndTapered (par défaut), gndLine ou gndArrow.
MarkerShape	TsgcGaugeMarkerShape	gmsTriangle (par défaut), gmsCircle, gmsDiamond ou gmsLine.
CapVisible	Boolean	Dessine le capuchon sur le pivot d'une aiguille. True par défaut.
Visible	Boolean	Dessine l'aiguille. True par défaut.

TsgcGaugeThreshold

Propriété	Type	Description
Value	Double	La valeur surveillée.
PointerIndex	Integer	L'aiguille surveillée. 0 par défaut.
Direction	TsgcGaugeThresholdDirection	gtdRising (par défaut), gtdFalling ou gtdBoth.
Hysteresis	Double	De combien la valeur doit revenir en arrière avant que le seuil puisse se déclencher à nouveau. 0 par défaut.

Color	TsgcUIColor	La couleur de la marque du seuil sur l'échelle. 0 (par défaut) ne dessine aucune marque.
ValueColor	TsgcUIColor	La couleur du texte de la valeur tant que le seuil est franchi. Elle s'applique aux seuils de l'aiguille 0.
Enabled	Boolean	True par défaut.
Active	Boolean	Public, lecture seule. True tant que la valeur est au-delà du seuil.

TsgcGaugeAnimation

Propriété	Type	Description
Enabled	Boolean	Anime les déplacements d'aiguille. True par défaut.
Duration	Integer	La durée d'un déplacement en millisecondes. 300 par défaut.
Easing	TsgcGaugeEasing	geLinear, geQuadOut, geCubicOut (par défaut), geCubicInOut ou geBackOut.

Exemple

```
uses
  SysUtils, Classes,
  sgcGauges_Types, sgcGauges_Model, sgcGauges_Linear;

procedure ConfigurePointers(aGauge: TsgcLinearGauge);
var
  oPointer: TsgcGaugePointer;
  oThreshold: TsgcGaugeThreshold;
begin
  aGauge.BeginUpdate;
  try
    // pointer 0 is the bar of the linear gauge
    aGauge.Pointers[0].Color := $FF2A78D6;
    // a range pointer: a band from Value to EndValue
    oPointer := aGauge.Pointers.Add;
    oPointer.Kind := gpkRange;
    oPointer.Value := 40;
    oPointer.EndValue := 60;
    // a threshold on pointer 0 with a hysteresis of 2
    oThreshold := aGauge.Thresholds.Add;
    oThreshold.Value := 90;
    oThreshold.Direction := gtdBoth;
    oThreshold.Hysteresis := 2;
    oThreshold.Color := $FFE34948;
    oThreshold.ValueColor := $FFE34948;
    aGauge.Animation.Duration := 500;
    aGauge.Animation.Easing := geBackOut;
  finally
    aGauge.EndUpdate;
  end;
end;
```

Voir aussi

- Aiguilles
- Seuils

- Animation

Échelles

Une échelle fait correspondre des valeurs à une position sur la jauge : un arc sur une jauge radiale, une ligne sur une jauge linéaire. La collection Scales d'une jauge contient toujours au moins une échelle, de 0 à 100. Une aiguille affiche sa valeur sur l'échelle donnée par son ScaleIndex.

Propriétés de TsgcGaugeScale

Propriété	Description
Minimum, Maximum	La plage de l'échelle, 0 et 100 par défaut. Un changement de la plage borne à nouveau les valeurs des aiguilles sur l'échelle.
Mapping	gsmLinear répartit les valeurs de façon uniforme. gsmDecibel les répartit sur une courbe en décibels, comme sur un vumètre.
Reversed	L'échelle va de Maximum à Minimum.
Radius, Offset, Thickness	La position et la largeur de la bande de l'échelle, en fractions de la taille de la jauge. Radius vaut 1 par défaut, le bord extérieur de l'arc, et Thickness 0,05. Une seconde échelle avec un Radius de 0,7 se place à l'intérieur de la première.
MajorInterval, MinorCount	Le pas entre deux graduations étiquetées, et le nombre de graduations secondaires entre elles (4 par défaut). Un MajorInterval de 0 choisit un pas rond pour environ 5 intervalles. MajorStep renvoie le pas utilisé.
TickPlacement, TickColor	Graduations à l'intérieur de la bande (gtpInside), à l'extérieur (gtpOutside) ou en travers (gtpCross), et leur couleur (0 = thème).
ShowLabels, LabelFormat, LabelFont	Les étiquettes des graduations principales. LabelFormat est un masque FormatFloat ; lorsqu'il est vide, les étiquettes ont les décimales du pas principal. Les étiquettes qui se chevaucheraient sont éclaircies.
Ranges	Les bandes de couleur de l'échelle. Voir Plages.
Visible	Dessine l'échelle, True par défaut.

ValueToRatio renvoie la position d'une valeur sur l'échelle, de 0 au début à 1 à la fin, avec la correspondance et Reversed appliqués.

Plusieurs échelles

Dans l'édition Registered, une jauge dessine chaque échelle de la collection, par exemple km/h à l'extérieur et mph à l'intérieur. En Community, seule la première échelle est dessinée. Les autres échelles restent dans la collection, de sorte qu'une fiche créée avec l'édition Registered se charge toujours.

Exemple

```
uses
  SysUtils, Classes,
  sgcGauges_Types, sgcGauges_Model, sgcGauges_Radial;

procedure SetupDualScale(aGauge: TsgcRadialGauge);
var
  oOuter, oInner: TsgcGaugeScale;
```

```
begin
  aGauge.BeginUpdate;
  try
    // scale 0, the outer one: km/h
    oOuter := aGauge.Scales[0];
    oOuter.Minimum := 0;
    oOuter.Maximum := 240;
    oOuter.MajorInterval := 20;
    oOuter.MinorCount := 3;
    oOuter.TickPlacement := gtpInside;
    oOuter.LabelFormat := '0';
    // scale 1, inside it: mph (drawn in the Registered edition only)
    oInner := aGauge.Scales.Add;
    oInner.Minimum := 0;
    oInner.Maximum := 150;
    oInner.Radius := 0.7;
    oInner.MajorInterval := 25;
    oInner.MinorCount := 4;
    oInner.TickPlacement := gtpOutside;
    oInner.LabelFont.Size := 8;
    // the needle stays on scale 0
    aGauge.Pointers[0].ScaleIndex := 0;
  finally
    aGauge.EndUpdate;
  end;
end;
```

Voir aussi

- Plages
- Aiguilles
- Éditions

Plages

Une plage colore une partie d'une échelle, comme une bande verte normale et une bande rouge critique. Chaque échelle a sa propre collection Ranges d'éléments TsgcGaugeRange.

Propriétés de TsgcGaugeRange

Propriété	Description
StartValue, EndValue	Les valeurs où la bande commence et finit, en unités de l'échelle. Elles peuvent être données dans n'importe quel ordre.
Color	La couleur de la bande. 0 prend une couleur de la palette du thème, selon l'indice de la plage.
Thickness	La largeur de la bande en fraction de la taille de la jauge, dessinée à partir du bord extérieur de l'échelle. 0 remplit toute la bande de l'échelle.
Caption	Un nom pour la plage, conservé avec elle. La jauge ne le dessine pas.

Plages et aiguilles

Une aiguille de type barre sans couleur propre (Color 0) prend la couleur de la plage où se trouve sa valeur, de sorte que la barre d'une jauge linéaire devient rouge lorsque la valeur entre dans la bande rouge. Un seuil peut aussi colorer le texte de la valeur, voir Seuils.

Exemple

```
uses
  SysUtils, Classes,
  sgcUICore_Canvas, sgcGauges_Types, sgcGauges_Model, sgcGauges_Linear;

procedure AddBand(aScale: TsgcGaugeScale; aStart, aEnd: Double;
  aColor: TsgcUIColor; const aCaption: string);
var
  oRange: TsgcGaugeRange;
begin
  oRange := aScale.Ranges.Add;
  oRange.StartValue := aStart;
  oRange.EndValue := aEnd;
  oRange.Color := aColor;
  oRange.Caption := aCaption;
  // the band is a fraction of the gauge size; 0 = the full scale band
  oRange.Thickness := 0.03;
end;

procedure SetupTemperature(aGauge: TsgcLinearGauge);
begin
  aGauge.LinearStyle := glsBullet;
  aGauge.Scales[0].Minimum := -20;
  aGauge.Scales[0].Maximum := 120;
  AddBand(aGauge.Scales[0], -20, 60, $FF2EAD6B, 'Normal');
  AddBand(aGauge.Scales[0], 60, 90, $FFF0A030, 'High');
  AddBand(aGauge.Scales[0], 90, 120, $FFE34948, 'Critical');
  // the bullet target line
  aGauge.Target := 75;
```

```
  aGauge.Value := 68;
end;
```

Voir aussi

- Échelles
- Seuils
- Thèmes

Aiguilles

Une aiguille affiche une valeur sur une échelle. La collection Pointers d'une jauge contient toujours au moins une aiguille, et Value est un raccourci vers la valeur de la première.

Types d'aiguilles

Type	Description
gpkNeedle	Une aiguille partant du centre d'une jauge radiale, effilée, une ligne ou une flèche (NeedleStyle), avec un capuchon sur le pivot (CapVisible). Le type par défaut.
gpkMarker	Une petite forme sur l'échelle : un triangle, un cercle, un losange ou une ligne (MarkerShape). Utile pour une consigne ou une cible.
gpkBar	Une bande qui remplit l'échelle de Origin à la valeur.
gpkRange	Une bande de Value à EndValue, par exemple une fenêtre de tolérance.

Propriétés de TsgcGaugePointer

Propriété	Description
Kind, ScaleIndex	Le type d'aiguille et l'échelle sur laquelle elle affiche sa valeur.
Value, EndValue	La valeur, et la fin de la bande d'une aiguille de type plage. Toutes deux sont bornées à l'échelle et peuvent être écrites depuis n'importe quel thread.
Origin	Le point de départ d'une barre, borné à l'échelle. Lorsqu'il n'est pas défini, la barre part du Minimum de l'échelle. Un Origin de 0 sur une échelle de -50 à 50 dessine une barre qui croît vers la gauche ou la droite à partir du milieu.
Color	La couleur de l'aiguille. Avec 0, une aiguille ou un repère prend une couleur de la palette du thème, une barre la couleur de la plage où se trouve sa valeur (ou la couleur d'accent) et une aiguille de type plage une couleur d'accent translucide.
Width, Length, Offset	La taille et la position de l'aiguille, en fractions de la taille de la jauge (Width 0,04 et Length 0,9 par défaut).
NeedleStyle, MarkerShape, CapVisible, Visible	La forme d'une aiguille (gndTapered, gndLine, gndArrow) et d'un repère, le capuchon d'une aiguille, et si l'aiguille est dessinée.

Plusieurs aiguilles

Dans l'édition Registered, chaque aiguille de la collection est dessinée, animée et évaluée pour les seuils, par exemple une aiguille pour la valeur courante et un repère pour la consigne. En Community, seule la première aiguille est active ; les autres restent dans la collection.

Exemple

```
uses
  SysUtils, Classes,
  sgcGauges_Types, sgcGauges_Model, sgcGauges_Radial;

procedure SetupPointers(aGauge: TsgcRadialGauge);
var
  oSetpoint, oBand: TsgcGaugePointer;
begin
  aGauge.BeginUpdate;
  try
    // pointer 0: the default needle
    aGauge.Pointers[0].NeedleStyle := gndArrow;
    aGauge.Pointers[0].Length := 0.8;
    // pointer 1: a marker for the setpoint (Registered edition)
    oSetpoint := aGauge.Pointers.Add;
    oSetpoint.Kind := gpkMarker;
    oSetpoint.MarkerShape := gmsTriangle;
    oSetpoint.Color := $FF2EAD6B;
    // pointer 2: a band from Value to EndValue (Registered edition)
    oBand := aGauge.Pointers.Add;
    oBand.Kind := gpkRange;
    oBand.Width := 0.06;
  finally
    aGauge.EndUpdate;
  end;
  aGauge.Pointers[0].Value := 64;
  aGauge.Pointers[1].Value := 70;
  aGauge.Pointers[2].Value := 55;
  aGauge.Pointers[2].EndValue := 85;
end;

procedure SetupDeviation(aGauge: TsgcRadialGauge);
begin
  // a bar that grows from 0, left or right, on a -50..50 scale
  aGauge.Scales[0].Minimum := -50;
  aGauge.Scales[0].Maximum := 50;
  aGauge.Pointers[0].Kind := gpkBar;
  aGauge.Pointers[0].Origin := 0;
  aGauge.Value := -18;
end;
```

Voir aussi

- Échelles
- Animation
- Seuils

Seuils

Un seuil surveille la valeur d'une aiguille et déclenche OnThreshold lorsque la valeur le franchit. Utilisez-le pour des alarmes, des avertissements ou un changement de couleur du texte de la valeur.

Fonctionnement d'un seuil

- Un seuil est évalué lorsqu'une valeur est écrite, sur la nouvelle valeur cible, jamais sur les valeurs dessinées pendant une animation.
- gtdRising et gtdBoth sont actifs à partir de Value et cessent de l'être sous Value moins Hysteresis. gtdFalling est actif jusqu'à Value et cesse de l'être au-dessus de Value plus Hysteresis. L'hystérésis évite qu'une valeur qui oscille autour de la limite ne déclenche le seuil sans cesse.
- Avec gtdRising et gtdFalling, OnThreshold se déclenche lorsque le seuil devient actif. Avec gtdBoth, il se déclenche aux deux franchissements, et aRising indique la direction. Pour une alarme qui a aussi besoin d'un événement de retour à la normale, utilisez gtdBoth avec une hystérésis.
- Un changement de la configuration, comme une nouvelle Value du seuil, met à jour Active sans événement.

Propriétés de TsgcGaugeThreshold

Propriété	Description
Value, Hysteresis, Direction	La limite, la bande morte et la direction (gtdRising par défaut).
PointerIndex	L'aiguille que le seuil surveille, 0 par défaut.
Color	Lorsqu'il n'est pas 0, une marque de cette couleur est dessinée sur l'échelle à Value.
ValueColor	Lorsqu'il n'est pas 0, le texte de la valeur prend cette couleur tant que le seuil est actif (seuils de la première aiguille).
Enabled, Active	Désactive le seuil, et indique s'il est actif maintenant (lecture seule).

OnThreshold

`procedure(Sender: TObject; aThreshold: TsgcGaugeThreshold; aValue: Double; aRising: Boolean)`. Les contrôles VCL et FireMonkey l'appellent dans le thread principal, dans l'ordre des franchissements, quel que soit le thread qui a écrit la valeur. Un modèle sans hôte l'appelle dans le thread qui a écrit la valeur.

Exemple

```
uses
  SysUtils, Classes, Controls, Forms, StdCtrls,
  sgcGauges_Types, sgcGauges_Model, sgcGauges_VCL;

type
  TTankAlarms = class(TComponent)
  public
    Level: TsgcVCLLinearGauge;
    Status: TLabel;
    procedure LevelThreshold(Sender: TObject; aThreshold: TsgcGaugeThreshold;
      aValue: Double; aRising: Boolean);
  end;
```

```
procedure TTankAlarms.LevelThreshold(Sender: TObject;
  aThreshold: TsgcGaugeThreshold; aValue: Double; aRising: Boolean);
begin
  case aThreshold.Index of
    0:
      if aRising then
        Status.Caption := Format('High level: %.1f %%', [aValue])
      else
        Status.Caption := 'Level back to normal';
    1:
      Status.Caption := Format('Low level: %.1f %%', [aValue]);
  end;
end;

procedure SetupTankAlarms(aAlarms: TTankAlarms);
var
  oHigh, oLow: TsgcGaugeThreshold;
begin
  // threshold 0: active at 90 %, clears below 85 %, an event both ways
  oHigh := aAlarms.Level.Thresholds.Add;
  oHigh.Value := 90;
  oHigh.Hysteresis := 5;
  oHigh.Direction := gtdBoth;
  oHigh.Color := $FFE34948;
  oHigh.ValueColor := $FFE34948;
  // threshold 1: active at 10 % or below, an event on entering only
  oLow := aAlarms.Level.Thresholds.Add;
  oLow.Value := 10;
  oLow.Hysteresis := 5;
  oLow.Direction := gtdFalling;
  oLow.Color := $FFF0A030;
  aAlarms.Level.OnThreshold := aAlarms.LevelThreshold;
end;
```

Voir aussi

- Plages
- Aiguilles
- Démarrage rapide VCL

Animation

Lorsqu'une valeur change, l'aiguille passe de la valeur qu'elle affiche à la nouvelle en un court instant, avec une courbe d'accélération. La valeur elle-même, les seuils et les événements changent immédiatement ; seul le dessin bouge.

Propriétés de TsgcGaugeAnimation

Propriété	Description
Enabled	Anime les changements de valeur, True par défaut.
Duration	La durée d'un déplacement en millisecondes, 300 par défaut. 0 désactive l'animation.
Easing	La courbe du déplacement : geLinear, geQuadOut, geCubicOut (par défaut), geCubicInOut ou geBackOut, qui dépasse légèrement à la fin.

Quand une valeur ne s'anime pas

- À la conception, avec Enabled à False ou avec une Duration de 0.
- Lorsque la valeur est écrite avec `Gauge.SetValue(aPointer, aValue, False)`.
- Lorsque la valeur est chargée depuis un fichier de fiche, pendant la mise à jour de la collection Pointers.
- Dans un rendu sans interface, qui dessine toujours la valeur cible.

Une nouvelle valeur qui arrive pendant un déplacement part de la valeur dessinée à cet instant, de sorte que l'aiguille ne saute jamais.

Images

Les contrôles VCL et FireMonkey dessinent les images d'un déplacement sur la couche dynamique uniquement, et au plus MaxFPS fois par seconde (60 par défaut). Lorsque rien ne bouge, une jauge ne dessine aucune image. Le temps de la jauge provient de la fonction Clock du modèle ; DisplayValue renvoie la valeur dessinée à un instant donné et IsAnimating indique si une aiguille bouge encore.

Exemple

```
uses
  SysUtils, Classes, Controls, Forms,
  sgcGauges_Types, sgcGauges_Model, sgcGauges_VCL;

procedure SetupAnimation(aGauge: TsgcVCLRadialGauge);
begin
  // half a second, with a small overshoot at the end
  aGauge.Animation.Enabled := True;
  aGauge.Animation.Duration := 500;
  aGauge.Animation.Easing := geBackOut;
  // at most 30 frames per second while a pointer moves
  aGauge.MaxFPS := 30;
end;

procedure ShowReading(aGauge: TsgcVCLRadialGauge; aValue: Double);
begin
  // animated, as the Animation settings say
  aGauge.Value := aValue;
end;
```

```
procedure ResetReading(aGauge: TsgcVCLRadialGauge);
begin
  // no animation: the needle jumps to 0
  aGauge.Gauge.SetValue(0, 0, False);
end;

procedure ShowMinMax(aGauge: TsgcVCLRadialGauge; aMin, aMax: Double);
begin
  // one change and one repaint for both pointers
  aGauge.Gauge.BeginUpdate;
  try
    aGauge.Gauge.SetValue(1, aMin);
    aGauge.Gauge.SetValue(2, aMax);
  finally
    aGauge.Gauge.EndUpdate;
  end;
end;
```

Voir aussi

- Aiguilles
- Architecture
- Sortie sans interface

Thèmes

Une jauge prend ses couleurs du thème sgcUICore défini dans **Theme** : utLight (par défaut) ou utDark. Chaque propriété de couleur d'une jauge vaut 0 par défaut, ce qui signifie la couleur du thème, de sorte qu'une jauge sans couleurs propres suit entièrement le thème.

Couleurs

Propriété	Avec 0
BackgroundColor	L'arrière-plan du thème.
FaceColor	La couleur de face du thème, derrière l'échelle d'un cadran.
Scales[i].TickColor	La couleur de texte atténuée du thème.
Ranges[i].Color, Pointers[i].Color	Couleurs de la palette du thème, voir Plages et Aiguilles.
TitleFont.Color, ValueFont.Color, LabelFont.Color	La couleur de texte du thème.

Les couleurs s'écrivent $AARRGGBB, par exemple $FFE34948 pour un rouge opaque.

Texte

- TitleFont, ValueFont et le LabelFont de chaque échelle sont des valeurs TsgcUITextStyle avec Size, Bold et Color. Les tailles sont relatives à la jauge : le texte grandit et rétrécit avec le contrôle. Le texte de la valeur est dessiné au double de la taille de ValueFont.
- ValueFormat est un masque FormatFloat pour le texte de la valeur. Lorsqu'il est vide, la valeur a les décimales de l'échelle. Units est dessiné à côté de la valeur.
- Les nombres utilisent les FormatSettings du modèle, avec '.' comme séparateur décimal par défaut, de sorte qu'une jauge a le même aspect dans toutes les locales tant que vous ne le changez pas.
- OnFormatValue permet de remplacer le texte de la valeur de n'importe quelle aiguille.

Exemple

```
uses
  SysUtils, Classes, Controls, Forms,
  sgcUICore_Theme, sgcGauges_Types, sgcGauges_Model, sgcGauges_VCL;

type
  TGaugeStyler = class(TComponent)
  public
    procedure FormatPower(Sender: TObject; aPointer: Integer; aValue: Double;
      var aText: string);
  end;

procedure TGaugeStyler.FormatPower(Sender: TObject; aPointer: Integer;
  aValue: Double; var aText: string);
begin
  if aValue >= 1000 then
    aText := FormatFloat('0.00', aValue / 1000) + ' MW';
end;

procedure StyleGauge(aGauge: TsgcVCLRadialGauge; aStyler: TGaugeStyler);
var
```

```
  vSettings: TFormatSettings;
begin
  aGauge.Theme := utDark;
  // 0 means the theme colour, so only these two change
  aGauge.FaceColor := $FF1E2A38;
  aGauge.Pointers[0].Color := $FFF0A030;
  aGauge.Scales[0].TickColor := $FF8899AA;
  aGauge.TitleFont.Size := 11;
  aGauge.TitleFont.Bold := True;
  aGauge.ValueFormat := '0.0';
  // labels and values with a decimal comma
  vSettings := aGauge.Gauge.FormatSettings;
  vSettings.DecimalSeparator := ',';
  aGauge.Gauge.FormatSettings := vSettings;
  aGauge.OnFormatValue := aStyler.FormatPower;
end;
```

Voir aussi

- Plages
- Démarrage rapide FMX
- Sortie sans interface

Sortie sans interface

Chaque jauge se dessine sans fenêtre. Les classes de modèle dessinent sur un canevas sgcUICore et enregistrent des fichiers PNG et SVG, de sorte qu'un serveur, un rapport ou un programme console sous Windows ou Linux64 peut produire les mêmes images que les contrôles.

Méthodes d'un modèle

Méthode	Description
SaveToPNG(aStream or aFileName, aWidth, aHeight, aScale = 1)	Une image PNG de aWidth x aHeight pixels logiques, multipliée par aScale. Une échelle de 2 donne une image deux fois plus grande pour un écran haute densité ou une impression.
SaveToSVG(aStream or aFileName, aWidth, aHeight, aScale = 1)	Un document SVG du même dessin. Les nombres du SVG ne dépendent jamais de la locale.
Render(aCanvas, aWidth, aHeight, aScale = 1, aNowMs = -1)	Dessine les deux couches sur un TsgcUICanvas. Avec aNowMs inférieur à 0, les aiguilles sont dessinées à leurs valeurs cibles.
RenderStatic, RenderDynamic	Dessinent uniquement la couche statique ou uniquement la couche dynamique.

Contrôles

Les contrôles VCL et FireMonkey ont SaveToPNG(aStream or aFileName, aScale = 1) et SaveToSVG avec les mêmes arguments. Ils enregistrent la jauge à la taille cliente du contrôle, de sorte qu'une capture d'un panneau n'a pas besoin de capture de fenêtre.

Même sortie partout

La sortie sans interface est déterministe : la même jauge donne des fichiers PNG et SVG identiques octet pour octet sur Win32, Win64 et Linux64. Les unités de modèle n'utilisent ni VCL ni FireMonkey, et sur Linux64 elles compilent à partir de Delphi 10.2.

Exemple

```
uses
  SysUtils, Classes,
  sgcGauges_Types, sgcGauges_Model, sgcGauges_Linear;

function TankLevelSVG(aLevel: Double): string;
var
  oGauge: TsgcLinearGauge;
  oStream: TStringStream;
begin
  oGauge := TsgcLinearGauge.Create;
  try
    oGauge.Orientation := gloVertical;
    oGauge.Units := '%';
    oGauge.Value := aLevel;
    // Delphi 2009 and later; the SVG is UTF-8 text
    oStream := TStringStream.Create('', TEncoding.UTF8);
    try
      // 80 x 240 logical pixels
```

```
      oGauge.SaveToSVG(oStream, 80, 240);
      Result := oStream.DataString;
    finally
      oStream.Free;
    end;
  finally
    oGauge.Free;
  end;
end;

procedure SaveTankLevelPNG(aStream: TStream; aLevel: Double);
var
  oGauge: TsgcLinearGauge;
begin
  oGauge := TsgcLinearGauge.Create;
  try
    oGauge.Orientation := gloVertical;
    oGauge.Value := aLevel;
    // twice the size, for a high DPI page or a report
    oGauge.SaveToPNG(aStream, 80, 240, 2);
  finally
    oGauge.Free;
  end;
end;
```

Voir aussi

- Démarrage rapide sans interface
- Plateformes
- Thèmes

Éditions

sgcGauges est construit à partir du même code source pour chaque édition.

Édition	Inclut
Registered	Tout : les dix types de jauges pour VCL et FireMonkey, plusieurs échelles et aiguilles par jauge, les afficheurs 14 segments et matrice de points, et la passerelle sgcGaugesWS vers sgcWebSockets.
Trial	Les mêmes fonctionnalités que Registered. Un build compilé avec SGC_TRIAL affiche un message indiquant qu'il s'agit d'une version d'essai lorsqu'un composant est créé après la fin de la période de validité nommée dans le build. Rien d'autre ne change.
Community	Six types de jauges pour VCL et FireMonkey sans les fonctionnalités avancées listées ci-dessous, et sans message d'essai.

Ce que Community inclut

La jauge radiale (tous les styles), la jauge linéaire (barre, thermomètre et bullet), l'afficheur numérique (7 segments), la LED, la barre de LED et l'anneau de progression, avec plages, seuils, animation, thèmes clair et sombre, les contrôles VCL et FireMonkey et la sortie PNG et SVG sans interface. La cible bullet de la jauge linéaire et le pic de la barre de LED sont des propriétés, et non des aiguilles supplémentaires, ils fonctionnent donc en Community.

Les fonctionnalités avancées

La directive SGC_GAUGES_ADVANCED de sgcVer.inc les active. Elle est désactivée dans l'édition Community.

- **Plusieurs échelles et aiguilles** : les échelles et aiguilles après la première sont conservées dans leurs collections, de sorte qu'une fiche créée avec l'édition Registered se charge toujours, mais Community ne les dessine pas, ne les anime pas et n'évalue pas leurs seuils.
- **Quatre types de jauges** : le vumètre, l'horloge, la boussole et l'annunciator, leurs unités (sgcGauges_VUMeter, sgcGauges_Clock, sgcGauges_Compass, sgcGauges_Annunciator), leurs huit contrôles et leur enregistrement.
- **Afficheurs numériques** : le dessin 14 segments et matrice de points. gdt14Segment et gdtDotMatrix restent dans l'énumération et sont dessinés comme du 7 segments.
- **La passerelle** : les packages sgcGaugesWS ne sont ni compilés ni installés en Community.

Code pour toutes les éditions

Le code qui utilise un type avancé peut tester la directive, de sorte que la même unité compile dans les deux éditions.

```
{$I sgcVer.inc}

uses
  SysUtils, Classes, Controls, Forms,
  sgcGauges_Types, sgcGauges_Model, sgcGauges_VCL;

function CreateHeadingGauge(aForm: TForm): TsgcVCLCustomGauge;
begin
{$IFDEF SGC_GAUGES_ADVANCED}
  // Registered and Trial: the compass
  Result := TsgcVCLCompassGauge.Create(aForm);
  TsgcVCLCompassGauge(Result).Heading := 270;
{$ELSE}
  // Community: a full circle radial gauge from 0 to 360
  Result := TsgcVCLRadialGauge.Create(aForm);
  TsgcVCLRadialGauge(Result).Scales[0].Maximum := 360;
```

```
  TsgcVCLRadialGauge(Result).Scales[0].MajorInterval := 45;
  TsgcVCLRadialGauge(Result).Value := 270;
{$ENDIF}
  Result.Parent := aForm;
  Result.SetBounds(16, 16, 200, 200);
end;
```

Voir aussi

- Types de jauges
- Installation
- Dépannage

Vue d'ensemble de la passerelle

sgcGaugesWS est la passerelle entre sgcGauges et sgcWebSockets. Ses sources lisent les messages JSON qu'un client sgcWebSockets reçoit et écrivent les valeurs dans les jauges, sans code dans les événements de message. La passerelle fait partie de l'édition Registered et nécessite sgcWebSockets ; elle possède ses propres packages, sgcGaugesWS et dclsgcGaugesWS, et ses composants se trouvent sur la page de palette **SGC Gauges**.

Composant	Unité	Client
TsgcGaugeFeedSource_MQTT	sgcGaugesWS_Source_MQTT	TsgcWSPClient_MQTT
TsgcGaugeFeedSource_WebSocket	sgcGaugesWS_Source_WebSocket	TsgcWebSocketClient
TsgcGaugeFeedSource_SSE	sgcGaugesWS_Source_SSE	TsgcHTTP1Client

Fonctionnement d'une source

Tant qu'une source est Active et que son Client est assigné, à l'exécution, elle installe son propre gestionnaire de messages sur le client et conserve le gestionnaire précédent, de sorte que le gestionnaire de l'application se déclenche toujours après la source. Mettre Active à False, effacer Client ou libérer la source rétablit le gestionnaire précédent. La source ne se connecte pas, ne s'abonne pas et ne se reconnecte pas : c'est l'application qui le fait avec le client.

Liaisons

Chaque élément de Bindings (TsgcGaugeFeedBinding, unit sgcGaugesWS_Source) écrit une aiguille d'une jauge. Un message est analysé une seule fois comme JSON et lu par chaque liaison active.

Propriété	Type	Description
Gauge	TComponent	Un contrôle de jauge VCL ou FireMonkey. Les autres composants sont refusés.
GaugeModel	TsgcCustomGauge	Public. Un modèle sans interface ; il l'emporte sur Gauge. Effacez-le avant de libérer le modèle.
PointerIndex	Integer	L'aiguille écrite. 0 par défaut.
ValuePath	string	Le chemin JSON de la valeur, comme price ou data.items[0].v. Vide, il lit le message entier.
FilterPath	string	Le chemin d'un champ qui sélectionne les messages. La liaison ne lit un message que si le chemin existe.
FilterValue	string	Si non vide, le texte de FilterPath doit lui être égal, comme ticker.
Topic	string	MQTT uniquement : un filtre de topic avec + et #. Vide, il lit chaque publication.
EventName	string	SSE uniquement : le type d'événement. Vide, il lit chaque événement.
Active	Boolean	False fait ignorer tous les messages à la liaison. True par défaut.

Un nombre JSON, une chaîne numérique avec '.' comme séparateur décimal et un booléen (true vaut 1, false vaut 0) sont acceptés. Un message sans le chemin d'une liaison est ignoré par cette liaison, de sorte que les mises à jour partielles sont possibles. Un JSON invalide, une valeur d'un autre type ou un mauvais indice d'aiguille vont à OnError.

Propriétés et événements de la source

Membre	Description
Client	Le client sgcWebSockets que la source lit.
Active	False fait ignorer tous les messages à la source. True par défaut.
Bindings	Les liaisons.
OnValue(Sender, aBinding, aValue)	Une liaison a écrit une valeur.
OnError(Sender, aError)	Un message n'a pas pu être appliqué.

Threads

Les valeurs sont écrites avec le SetValue thread-safe de la jauge, dans le thread d'arrivée du message, et les contrôles de jauge se redessinent et appellent OnThreshold dans le thread principal. Les écritures d'un message vers une jauge deviennent un seul changement. OnValue et OnError s'exécutent dans le thread d'arrivée sous le verrou de la source : utilisez queue (TThread.Queue) plutôt que synchronize depuis ces événements avant de toucher un contrôle. Modifiez Bindings depuis le thread principal.

Compilation

Gardez séparés dans le chemin de recherche les dossiers source de sgcGauges et de sgcWebSockets. Chaque dossier a son propre sgcVer.inc, et une unité lit celui de son propre dossier.

Exemple

Chaque source a une méthode publique qui applique un message comme si le client l'avait reçu : ProcessPublish (MQTT), ProcessMessage (WebSocket) et ProcessChunk (SSE). Elles renvoient le nombre de valeurs écrites. Cela alimente un modèle sans interface sans client :

```
uses
  SysUtils, Classes,
  sgcGauges_Model, sgcGauges_Radial,
  sgcGaugesWS_Source, sgcGaugesWS_Source_WebSocket;

procedure FeedWithoutClient;
var
  oGauge: TsgcRadialGauge;
  oFeed: TsgcGaugeFeedSource_WebSocket;
  oBinding: TsgcGaugeFeedBinding;
begin
  oGauge := TsgcRadialGauge.Create;
  oFeed := TsgcGaugeFeedSource_WebSocket.Create(nil);
  try
    oGauge.Scales[0].Maximum := 200;
    oBinding := oFeed.Bindings.Add;
    oBinding.GaugeModel := oGauge; // a headless model
    oBinding.ValuePath := 'data.price';
    oBinding.FilterPath := 'type';
    oBinding.FilterValue := 'ticker';
    // the messages a client would deliver
```

```
    oFeed.ProcessMessage('{"type":"ticker","data":{"price":101.5}}');
    oFeed.ProcessMessage('{"type":"trade","data":{"price":99}}'); // filtered
    WriteLn(FloatToStr(oGauge.Value)); // 101.5
  finally
    oFeed.Free; // free the source before the model it writes to
    oGauge.Free;
  end;
end;
```

Voir aussi

- Source MQTT
- Source WebSocket
- Source d'événements envoyés par le serveur
- TsgcCustomGauge

Source MQTT

Classe : TsgcGaugeFeedSource_MQTT (unit sgcGaugesWS_Source_MQTT)
Alimente les jauges avec les publications qu'un TsgcWSPClient_MQTT reçoit. Client est le client MQTT. La source installe son gestionnaire OnMQTTPublish et conserve le précédent, et le crochet continue de fonctionner après les reconnexions.
Le Topic de chaque liaison est un filtre de topic MQTT : + correspond à un niveau et # à ce niveau et à tous les niveaux inférieurs. Une liaison ne lit que les publications dont le topic lui correspond, et un Topic vide lit chaque publication. La source ne s'abonne pas : c'est l'application qui s'abonne sur le client, avec un filtre qui couvre les topics des liaisons. Chaque payload est lu comme du JSON.
ProcessPublish(aTopic, aText) applique une publication sans client et renvoie le nombre de valeurs écrites. La fonction sgcGaugeMQTTTopicMatch(aFilter, aTopic) de la même unité teste un topic par rapport à un filtre.

Exemple

Deux jauges radiales alimentées par les topics sensors/<id>/temperature et sensors/<id>/humidity, avec des payloads comme {"value": 21.5} :

```
uses
  SysUtils, Classes, Controls, Forms,
  sgcWebSocket, sgcWebSocket_Protocols,
  sgcGauges_VCL, sgcGaugesWS_Source, sgcGaugesWS_Source_MQTT;

type
  TSensorFeed = class(TComponent)
  public
    WSClient: TsgcWebSocketClient;
    MQTT: TsgcWSPClient_MQTT;
    Feed: TsgcGaugeFeedSource_MQTT;
    procedure FeedError(Sender: TObject; const aError: string);
  end;

procedure TSensorFeed.FeedError(Sender: TObject; const aError: string);
begin
  // the thread of the client: queue before touching a control
end;

function CreateSensorFeed(aForm: TForm; aTemperature,
  aHumidity: TsgcVCLRadialGauge): TSensorFeed;
var
  oBinding: TsgcGaugeFeedBinding;
begin
  Result := TSensorFeed.Create(aForm);
  Result.WSClient := TsgcWebSocketClient.Create(Result);
  Result.WSClient.Host := 'broker.example.com';
  Result.WSClient.Port := 1883;
  Result.MQTT := TsgcWSPClient_MQTT.Create(Result);
  Result.MQTT.Client := Result.WSClient;

  Result.Feed := TsgcGaugeFeedSource_MQTT.Create(Result);
  Result.Feed.OnError := Result.FeedError;
  oBinding := Result.Feed.Bindings.Add;
  oBinding.Topic := 'sensors/+/temperature';
  oBinding.ValuePath := 'value';
  oBinding.Gauge := aTemperature;
  oBinding := Result.Feed.Bindings.Add;
  oBinding.Topic := 'sensors/+/humidity';
  oBinding.ValuePath := 'value';
  oBinding.Gauge := aHumidity;
  Result.Feed.Client := Result.MQTT;

  Result.WSClient.Active := True;
```

```
  // the source does not subscribe: the application does
  Result.MQTT.Subscribe('sensors/#');
end;
```

Voir aussi

- Vue d'ensemble de la passerelle
- Source WebSocket
- Source d'événements envoyés par le serveur

Source WebSocket

Classe : TsgcGaugeFeedSource_WebSocket (unit sgcGaugesWS_Source_WebSocket)
Alimente les jauges avec les messages texte qu'un TsgcWebSocketClient reçoit. Client est le client WebSocket. La source installe son gestionnaire OnMessage et conserve le précédent, et le crochet continue de fonctionner après les reconnexions, comme celles du WatchDog. Chaque message texte est lu comme du JSON avec les ValuePath, FilterPath et FilterValue des liaisons, dans le thread depuis lequel le client notifie, de sorte que tout mode NotifyEvents du client est sûr.
FilterPath et FilterValue sélectionnent les messages d'un flux qui en transporte plusieurs types, comme un canal de cotation et un canal de transactions sur une même connexion. ProcessMessage(aText) applique un message sans client et renvoie le nombre de valeurs écrites.

Exemple

Un afficheur de prix et une barre de variation alimentés par les messages de cotation d'un flux :

```
uses
  SysUtils, Classes, Controls, Forms,
  sgcWebSocket,
  sgcGauges_VCL, sgcGaugesWS_Source, sgcGaugesWS_Source_WebSocket;

type
  TTickerFeed = class(TComponent)
  public
    WSClient: TsgcWebSocketClient;
    Feed: TsgcGaugeFeedSource_WebSocket;
    procedure FeedValue(Sender: TObject; aBinding: TsgcGaugeFeedBinding;
      aValue: Double);
  end;

procedure TTickerFeed.FeedValue(Sender: TObject;
  aBinding: TsgcGaugeFeedBinding; aValue: Double);
begin
  // the thread of the client, under the source lock: keep it short
end;

function CreateTickerFeed(aForm: TForm; aPrice: TsgcVCLDigitalGauge;
  aChange: TsgcVCLLinearGauge): TTickerFeed;
var
  oBinding: TsgcGaugeFeedBinding;
begin
  Result := TTickerFeed.Create(aForm);
  Result.WSClient := TsgcWebSocketClient.Create(Result);
  Result.WSClient.URL := 'wss://feed.example.com/ticker';

  Result.Feed := TsgcGaugeFeedSource_WebSocket.Create(Result);
  Result.Feed.Client := Result.WSClient;
  Result.Feed.OnValue := Result.FeedValue;
  // {"type":"ticker","symbol":"ABC","last":"12.50","change":0.8}
  oBinding := Result.Feed.Bindings.Add;
  oBinding.FilterPath := 'type';
  oBinding.FilterValue := 'ticker';
  oBinding.ValuePath := 'last';
  oBinding.Gauge := aPrice;
  oBinding := Result.Feed.Bindings.Add;
  oBinding.FilterPath := 'type';
  oBinding.FilterValue := 'ticker';
  oBinding.ValuePath := 'change';
  oBinding.Gauge := aChange;
```

```
  Result.WSClient.Active := True;
end;
```

Voir aussi

- Vue d'ensemble de la passerelle
- Source MQTT
- Source d'événements envoyés par le serveur

Source d'événements envoyés par le serveur

Classe : TsgcGaugeFeedSource_SSE (unit sgcGaugesWS_Source_SSE)
Alimente les jauges avec les événements envoyés par le serveur d'un TsgcHTTP1Client. Client est le client HTTP. La source installe son gestionnaire OnSSEMessage et conserve le précédent. L'application démarre le flux avec Client.GetSSE.
La source découpe les fragments du flux en événements : les lignes data: sont jointes avec un saut de ligne, une ligne event: définit le type d'événement et une ligne vide termine l'événement. Les fins de ligne LF, CRLF et CR sont acceptées, et un événement peut être réparti sur plusieurs fragments. L'EventName de chaque liaison sélectionne le type d'événement qu'elle lit. Un événement sans ligne event: a le type message, et un EventName vide lit chaque événement. Les données de chaque événement sont lues comme du JSON.

Méthode	Description
ResetStream	Abandonne un événement qu'un flux a laissé à moitié lu. Appelez-la avant un nouveau flux après une reconnexion.
ProcessChunk(aText)	Applique un fragment de flux sans client et renvoie le nombre de valeurs écrites.

Exemple

Un anneau de progression et une barre de LED alimentés par les événements cpu et queue d'un flux de métriques :

```
uses
  SysUtils, Classes, Controls, Forms,
  sgcHTTP,
  sgcGauges_VCL, sgcGaugesWS_Source, sgcGaugesWS_Source_SSE;

type
  TMetricsFeed = class(TComponent)
  public
    HTTP: TsgcHTTP1Client;
    Feed: TsgcGaugeFeedSource_SSE;
  end;

function CreateMetricsFeed(aForm: TForm; aCPU: TsgcVCLProgressRingGauge;
  aQueue: TsgcVCLLEDBarGauge): TMetricsFeed;
var
  oBinding: TsgcGaugeFeedBinding;
begin
  Result := TMetricsFeed.Create(aForm);
  Result.HTTP := TsgcHTTP1Client.Create(Result);
  Result.Feed := TsgcGaugeFeedSource_SSE.Create(Result);
  Result.Feed.Client := Result.HTTP;
  // event: cpu       data: {"load":42.5}
  oBinding := Result.Feed.Bindings.Add;
  oBinding.EventName := 'cpu';
  oBinding.ValuePath := 'load';
  oBinding.Gauge := aCPU;
  // event: queue     data: {"pending":17}
  oBinding := Result.Feed.Bindings.Add;
  oBinding.EventName := 'queue';
  oBinding.ValuePath := 'pending';
  oBinding.Gauge := aQueue;
end;

procedure StartMetrics(aFeed: TMetricsFeed);
```

```
begin
  aFeed.HTTP.GetSSE('https://metrics.example.com/stream');
end;

procedure RestartMetrics(aFeed: TMetricsFeed);
begin
  // a new stream: drop an event that the old stream left half read
  aFeed.Feed.ResetStream;
  aFeed.HTTP.GetSSE('https://metrics.example.com/stream');
end;
```

Voir aussi

- Vue d'ensemble de la passerelle
- Source MQTT
- Source WebSocket

Vue d'ensemble de .NET

sgcGauges .NET est un portage C# managé de la bibliothèque sgcGauges de Delphi, et non une enveloppe autour d'elle. Les classes se trouvent dans l'espace de noms **esegece.sgcGauges** et gardent les noms de Delphi : **TsgcCustomGauge**, les dix modèles de jauge comme **TsgcRadialGauge** et **TsgcLEDBarGauge**, **TsgcGaugeScale**, **TsgcGaugeRange**, **TsgcGaugePointer**, **TsgcGaugeThreshold**, **TsgcGaugeAnimation** et les énumérations avec leurs préfixes de jauge (**grsHalf**, **gtdBoth**, **glsBullet**). Les contrôles suivent le schéma de nommage de Delphi sur chaque hôte : **TsgcVCLRadialGauge** devient **TsgcWinFormsRadialGauge**, **TsgcWpfRadialGauge** et **TsgcAvaloniaRadialGauge**. Les jauges sont dessinées avec le portage .NET de **sgcUICore**, le même canevas, les mêmes polices et les mêmes thèmes clair et sombre que les autres produits de la suite visuelle d'eSeGeCe.
Comme dans Delphi, chaque contrôle possède un modèle indépendant du framework, exposé sous le nom **Gauge**. Le modèle contient les échelles, les plages, les pointeurs, les seuils et l'animation et se dessine sur n'importe quel canevas de sgcUICore, de sorte que le même code de modèle fonctionne dans un formulaire WinForms, une fenêtre WPF ou Avalonia et un programme console sans interface, voir Rendu sans interface dans .NET.

Paquets

Le portage est découpé en six paquets NuGet, de sorte qu'un projet n'embarque que ce qu'il utilise.

PackageId	Contient	Frameworks cibles
esegece.sgcGauges	Les dix modèles de jauge, les classes d'échelle, de plage, de pointeur, de seuil et d'animation, le peintre et la sortie SVG (**SaveToSVG**). Aucune dépendance native. Dépend de esegece.sgcUICore.	netstandard2.0, net472, net6.0, net8.0, net9.0
esegece.sgcGauges.Rendering	La sortie PNG sans interface : les méthodes d'extension **SaveToPNG** et **RenderToRaster** de **sgcGauges_Raster**, dessinées sur le canevas SkiaSharp de esegece.sgcUICore.Rendering, sous Windows, Linux et macOS.	netstandard2.0, net472, net6.0, net8.0, net9.0
esegece.sgcGauges.WinForms	Les dix contrôles **TsgcWinForms<Type>Gauge** sur **TsgcWinFormsCustomGauge**.	net472, net8.0-windows, net9.0-windows
esegece.sgcGauges.Wpf	Les dix contrôles **TsgcWpf<Type>Gauge** sur **TsgcWpfCustomGauge**.	net472, net8.0-windows, net9.0-windows
esegece.sgcGauges.Avalonia	Les dix contrôles **TsgcAvalonia<Type>Gauge** sur **TsgcAvaloniaCustomGauge**, sous Windows, Linux et macOS avec Avalonia 11.	net8.0, net9.0
esegece.sgcGauges.sgcWebSockets	La passerelle vers sgcWebSockets : **TsgcGaugeFeedSource_WebSocket**, **TsgcGaugeFeedSource_MQTT** et **TsgcGaugeFeedSource_SSE**. Édition Registered uniquement.	netstandard2.0, net8.0, net9.0

Une seule ligne `PackageReference` apporte tout ce dont un hôte a besoin : les paquets WinForms et Avalonia dépendent de esegece.sgcGauges.Rendering, qui dépend de esegece.sgcGauges ; le paquet WPF dépend de esegece.sgcGauges et dessine via esegece.sgcUICore.Wpf. Voir Installation depuis NuGet.

Éditions

Édition	Identifiants de paquet	Comprend
Registered	esegece.sgcGauges, .Rendering, .WinForms, .Wpf, .Avalonia et .sgcWebSockets	Tout : les dix types de jauge sur les trois hôtes et sans interface, plusieurs échelles et pointeurs par jauge, les 14 afficheurs à segments et à matrice de points, et la passerelle sgcWebSockets.
Community	Les mêmes ids avec le suffixe .Community, par exemple esegece.sgcGauges.WinForms.Community. Il n'y a pas de paquet de passerelle Community.	Les jauges radiale, linéaire, numérique, LED, barre de LED et anneau de progression. Le VU meter, l'horloge, le compas et l'annunciator ne figurent pas dans les assemblages Community ; seules la première échelle et la première aiguille d'une jauge sont dessinées, animées et contrôlées pour les seuils ; **gdt14Segment** et **gdtDotMatrix** se dessinent comme du 7 segments.

C'est le même découpage que les éditions Delphi, voir Éditions.

Hôtes

Hôte	Équivalent Delphi	Plateformes
WinForms	VCL (contrôles TsgcVCL)	Windows, .NET Framework 4.7.2 et .NET 8 et 9.
WPF	VCL et FMX	Windows, .NET Framework 4.7.2 et .NET 8 et 9. Les contrôles peuvent être déclarés en XAML.
Avalonia	FMX (contrôles TsgcFMX)	Windows, Linux et macOS, .NET 8 et 9.
Sans interface	Les modèles de jauge	Toute plateforme des frameworks cibles : SVG avec esegece.sgcGauges, PNG avec esegece.sgcGauges.Rendering.

Chaque contrôle publie les propriétés de son équivalent Delphi avec les mêmes noms, plus **Gauge**, **Theme**, **OnThreshold**, **MaxFPS**, **SaveToPNG** et **SaveToSVG** sur la classe de base. Voir Classes .NET pour les noms de classes et Différences de .NET par rapport à Delphi pour ce qui change par rapport à Delphi.

Voir aussi

- Installation depuis NuGet
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia

- Classes .NET
- Différences de .NET par rapport à Delphi
- Liaison de données dans .NET
- Rendu sans interface dans .NET
- Passerelle sgcWebSockets pour .NET
- Démos .NET
- Plateformes

Installation depuis NuGet

Les paquets sont sur nuget.org. Ajoutez le paquet de l'hôte, les paquets du noyau et du rendu viennent avec lui comme dépendances.

Édition Registered

```
dotnet add package esegece.sgcGauges.WinForms
dotnet add package esegece.sgcGauges.Wpf
dotnet add package esegece.sgcGauges.Avalonia
```

Édition Community

```
dotnet add package esegece.sgcGauges.WinForms.Community
dotnet add package esegece.sgcGauges.Wpf.Community
dotnet add package esegece.sgcGauges.Avalonia.Community
```

Ou ajoutez la référence au fichier de projet à la main :

```
<ItemGroup>
  <PackageReference Include="esegece.sgcGauges.Wpf" Version="2026.10.0" />
</ItemGroup>
```

Référencez dans un projet soit les paquets Registered, soit les paquets Community, jamais les deux : ils contiennent les mêmes types dans le même espace de noms.

Paramètres du projet

- WinForms : un projet `net8.0-windows` avec `<UseWindowsForms>true</UseWindowsForms>`, ou un projet .NET Framework 4.7.2.
- WPF : un projet `net8.0-windows` avec `<UseWPF>true</UseWPF>`, ou un projet .NET Framework 4.7.2. Associez l'espace de noms en XAML avec `xmlns:sgc="clr-namespace:esegece.sgcGauges;assembly=esegece.sgcGauges.Wpf"`.
- Avalonia : un projet `net8.0` ou `net9.0` avec les paquets **Avalonia.Desktop** et **Avalonia.Themes.Fluent** d'Avalonia 11.3, comme toute application de bureau Avalonia. Associez l'espace de noms en XAML avec `xmlns:sgc="using:esegece.sgcGauges"`.
- Sans interface : esegece.sgcGauges seul suffit pour les modèles et le SVG. Ajoutez esegece.sgcGauges.Rendering pour le PNG, qui apporte SkiaSharp et ses assets natifs pour Windows et Linux.
- Données en direct de sgcWebSockets : ajoutez **esegece.sgcGauges.sgcWebSockets** (édition Registered), qui dépend de esegece.sgcGauges et de esegece.sgcWebSockets, voir Passerelle sgcWebSockets pour .NET.

En C#, chaque hôte a besoin de `using esegece.sgcGauges;`, et de `using esegece.sgcUICore;` pour **TsgcUITheme**, **TsgcUITextStyle** et les autres types de sgcUICore.

Voir aussi

- Vue d'ensemble de .NET
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Démos .NET

Démarrage rapide WinForms

Ce démarrage rapide construit en code un formulaire avec trois jauges WinForms : un compteur de vitesse avec une bande rouge et un seuil à 200 km/h, une barre de carburant et un voyant d'excès de vitesse que le seuil allume et éteint.

Étapes

1. Créez un projet Windows Forms pour `net8.0-windows` et ajoutez le paquet **esegece.sgcGauges.WinForms** (**esegece.sgcGauges.WinForms.Community** dans l'édition Community).
2. Ajoutez `using esegece.sgcGauges;` et `using esegece.sgcUICore;`.
3. Créez les contrôles et ajoutez-les au formulaire. Ce sont des contrôles WinForms ordinaires : Bounds, Dock, Anchor et Enabled fonctionnent comme d'habitude.
4. Définissez d'abord l'étendue de l'échelle, puis les plages, les seuils et la valeur : une valeur est bornée à l'échelle quand elle est écrite.
5. Affectez les événements. Ils portent les noms de Delphi et s'affectent avec `=`.

Exemple

Remarques

- **Value** est **Pointers[0].Value**. N'importe quel thread peut l'écrire : le contrôle se redessine dans le thread de l'interface et y déclenche **OnThreshold**.
- Avec **Animation.Enabled** (la valeur par défaut), l'aiguille se déplace vers la nouvelle valeur pendant **Animation.Duration** millisecondes ; la valeur, les seuils et les événements changent immédiatement. Seul le pointeur en mouvement est redessiné, au plus **MaxFPS** images par seconde.
- **Theme** est propre à chaque contrôle ; une propriété de couleur laissée à 0 prend la couleur du thème.
- **SaveToPNG** et **SaveToSVG** écrivent ce que le contrôle affiche à sa taille, avec une échelle facultative.

Voir aussi

- Vue d'ensemble de .NET
- Installation depuis NuGet
- Classes .NET
- Liaison de données dans .NET
- Démarrage rapide VCL

Démarrage rapide WPF

Ce démarrage rapide déclare en XAML un compteur de vitesse, un afficheur numérique de tr/min et un voyant d'excès de vitesse, et lie les deux valeurs à un view model.

Étapes

1. Créez un projet WPF pour `net8.0-windows` et ajoutez le paquet **esegece.sgcGauges.Wpf** (**esegece.sgcGauges.Wpf.Community** dans l'édition Community).
2. Associez l'espace de noms dans la fenêtre : `xmlns:sgc="clr-namespace:esegece.sgcGauges;assembly=esegece.sgcGauges.Wpf"`.
3. Déclarez les contrôles. Les propriétés simples comme **Title**, **Units**, **ValueFormat** et **Digits** se définissent comme des attributs ; **Value** est une propriété de dépendance et accepte une liaison.
4. Remplissez les collections (échelles, plages, seuils) et affectez les événements dans le code behind.

XAML

```
<Window x:Class="QuickStart.QuickStartWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:sgc="clr-namespace:esegece.sgcGauges;assembly=esegece.sgcGauges.Wpf"
        Title="sgcGauges" Width="680" Height="320">
  <StackPanel Orientation="Horizontal" Margin="8">
    <sgc:TsgcWpfRadialGauge x:Name="gauSpeed" Width="260" Height="260"
                            Title="Speed" Units="km/h" ValueFormat="0" Value="{Binding Speed}" />
    <StackPanel Margin="8,0,0,0">
      <sgc:TsgcWpfDigitalGauge x:Name="gauRpm" Width="340" Height="110"
                               Title="Engine" Units="rpm" ValueFormat="0" Digits="5" Value="{Binding Rpm}" />
      <sgc:TsgcWpfLEDGauge x:Name="ledAlarm" Width="160" Height="110" Title="Overspeed" Blink="True" />
    </StackPanel>
  </StackPanel>
</Window>
```

Code behind

Remarques

- Chaque contrôle a une propriété de dépendance : **ValueProperty** sur les jauges radiale, linéaire, numérique, LED, barre de LED, anneau de progression et VU meter, et **HeadingProperty** sur le compas. L'horloge et l'annunciator n'en ont aucune. Les autres propriétés sont des propriétés CLR. Voir Liaison de données dans .NET.
- **OnThreshold** s'exécute dans le thread de l'interface, y compris lorsque la valeur liée est définie depuis un autre thread.
- **Theme** est propre à chaque contrôle.

Voir aussi

- Vue d'ensemble de .NET
- Installation depuis NuGet
- Classes .NET
- Liaison de données dans .NET
- Démarrage rapide FMX

Démarrage rapide Avalonia

Ce démarrage rapide construit en code une fenêtre Avalonia avec un cadran en demi-cercle pour une pression, un anneau de progression et une barre de LED avec des bandes verte, ambre et rouge, le tout en thème sombre. Le même code fonctionne sous Windows, Linux et macOS.

Étapes

1. Créez un projet `net8.0` avec les paquets **Avalonia.Desktop** et **Avalonia.Themes.Fluent** (Avalonia 11.3) et ajoutez **esegece.sgcGauges.Avalonia** (**esegece.sgcGauges.Avalonia.Community** dans l'édition Community).
2. Ajoutez `using esegece.sgcGauges;` et `using esegece.sgcUICore;`.
3. Créez les contrôles, définissez l'échelle et la valeur, et ajoutez-les à un panneau de la fenêtre.

Exemple

Remarques

- Les contrôles peuvent aussi être déclarés en XAML avec `xmlns:sgc="using:esegece.sgcGauges"`, par exemple `<sgc:TsgcAvaloniaRadialGauge Title="Pressure" Value="{Binding Pressure}" />`.
- **ValueProperty** (tous les contrôles), **ActiveProperty** (LED) et **HeadingProperty** (compas) sont des propriétés stylées qui se lient dans les deux sens par défaut, voir Liaison de données dans .NET.
- Le minuteur d'animation d'un contrôle s'arrête quand il quitte l'arbre visuel ; les contrôles n'ont pas besoin de Dispose.

Voir aussi

- Vue d'ensemble de .NET
- Installation depuis NuGet
- Classes .NET
- Liaison de données dans .NET
- Démarrage rapide FMX

Classes .NET

Les classes de modèle portent les noms de Delphi et se trouvent dans l'espace de noms **esegece.sgcGauges**. Les contrôles remplacent le préfixe VCL et FMX par le nom de l'hôte.

Jauges et contrôles

Modèle	Contrôles Delphi	WinForms	WPF	Avalonia
TsgcRadialGauge	TsgcVCLRadialGauge, TsgcFMXRadialGauge	TsgcWinFormsRadialGauge	TsgcWpfRadialGauge	TsgcAvaloniaRadialGauge
TsgcLinearGauge	TsgcVCLLinearGauge, TsgcFMXLinearGauge	TsgcWinFormsLinearGauge	TsgcWpfLinearGauge	TsgcAvaloniaLinearGauge
TsgcDigitalGauge	TsgcVCLDigitalGauge, TsgcFMXDigitalGauge	TsgcWinFormsDigitalGauge	TsgcWpfDigitalGauge	TsgcAvaloniaDigitalGauge
TsgcLEDGauge	TsgcVCLLEDGauge, TsgcFMXLEDGauge	TsgcWinFormsLEDGauge	TsgcWpfLEDGauge	TsgcAvaloniaLEDGauge
TsgcLEDBarGauge	TsgcVCLLEDBarGauge, TsgcFMXLEDBarGauge	TsgcWinFormsLEDBarGauge	TsgcWpfLEDBarGauge	TsgcAvaloniaLEDBarGauge
TsgcProgressRingGauge	TsgcVCLProgressRingGauge, TsgcFMXProgressRingGauge	TsgcWinFormsProgressRingGauge	TsgcWpfProgressRingGauge	TsgcAvaloniaProgressRingGauge
TsgcVUMeterGauge (Registered)	TsgcVCLVUMeterGauge, TsgcFMXVUMeterGauge	TsgcWinFormsVUMeterGauge	TsgcWpfVUMeterGauge	TsgcAvaloniaVUMeterGauge
TsgcClockGauge (Registered)	TsgcVCLClockGauge, TsgcFMXClockGauge	TsgcWinFormsClockGauge	TsgcWpfClockGauge	TsgcAvaloniaClockGauge
TsgcCompassGauge (Registered)	TsgcVCLCompassGauge, TsgcFMXCompassGauge	TsgcWinFormsCompassGauge	TsgcWpfCompassGauge	TsgcAvaloniaCompassGauge
TsgcAnnunciatorGauge (Registered)	TsgcVCLAnnunciatorGauge, TsgcFMXAnnunciatorGauge	TsgcWinFormsAnnunciatorGauge	TsgcWpfAnnunciatorGauge	TsgcAvaloniaAnnunciatorGauge

Les quatre types Registered et leurs contrôles ne figurent pas dans les assemblages Community. Voir Types de jauges pour ce que dessine chaque type.

Classes de base

Classe	Paquet	Description
TsgcCustomGauge	esegece.sgcGauges	La base du modèle : **Scales**, **Pointers**, **Thresholds**, **Animation**, **Theme**, **Title**, **Units**, **ValueFormat**, **FormatSettings**, **Value**, **SetValue**, **ValueToText**, **BeginUpdate** et **EndUpdate**, **Render**,

		RenderStatic, **RenderDynamic**, **SaveToSVG** et les événements **OnChange**, **OnThreshold** et **OnFormatValue**.
TsgcWinFormsCustomGauge	esegece.sgcGauges.WinForms	La base des contrôles WinForms, sur **TsgcUIWinFormsControl**.
TsgcWpfCustomGauge	esegece.sgcGauges.Wpf	La base des contrôles WPF, sur **TsgcUIWpfControl**.
TsgcAvaloniaCustomGauge	esegece.sgcGauges.Avalonia	La base des contrôles Avalonia, sur **TsgcUIAvaloniaControl**, avec **ValueProperty**, **LogicalWidth()** et **LogicalHeight()**.
IsgcGaugeHost	esegece.sgcGauges	Implémentée par chaque contrôle : **GetGauge()** renvoie le modèle. La passerelle pilote tout objet qui l'implémente.
sgcGauges_Raster	esegece.sgcGauges.Rendering	Les méthodes d'extension **SaveToPNG** et **RenderToRaster** de **TsgcCustomGauge**.

Éléments du modèle

Classe	Description
TsgcGaugeScale, TsgcGaugeScales	Une échelle : **Minimum**, **Maximum**, **Mapping**, **MajorInterval**, **MinorCount**, **TickPlacement**, **LabelFormat**, **Ranges**.
TsgcGaugeRange, TsgcGaugeRanges	Une bande colorée d'une échelle : **StartValue**, **EndValue**, **Color**, **Thickness**.
TsgcGaugePointer, TsgcGaugePointers	Un pointeur de type aiguille, repère, barre ou plage : **Kind**, **ScaleIndex**, **Value**, **EndValue**, **Origin**, **Color**, **NeedleStyle**, **MarkerShape**.
TsgcGaugeThreshold, TsgcGaugeThresholds	Une limite qui déclenche **OnThreshold** : **Value**, **PointerIndex**, **Direction**, **Hysteresis**, **Color**, **ValueColor**, **Enabled**, **Active**.
TsgcGaugeAnimation	L'animation des pointeurs : **Enabled**, **Duration** en millisecondes et **Easing**.
TsgcGaugeAnnunciatorTile, TsgcGaugeAnnunciatorTiles	Une tuile de l'annunciator : **Caption**, **State**, **Acknowledged**.
TsgcGaugeFormatSettings	Une struct avec le **DecimalSeparator** et le **ThousandSeparator** des libellés, '.' et ',' par défaut.

Délégués

Délégué	Signature
TsgcGaugeChangeEvent	`(object Sender, TsgcGaugeChangeKind aKind)`
TsgcGaugeThresholdEvent	`(object Sender, TsgcGaugeThreshold aThreshold, double aValue, bool aRising)`
TsgcGaugeFormatValueEvent	`(object Sender, int aPointer, double aValue, ref string aText)`
TsgcGaugeTileClickEvent	`(object Sender, int aIndex)`, le **OnTileClick** des contrôles annunciator

Les classes de la passerelle sont listées dans Passerelle sgcWebSockets pour .NET.

Voir aussi

- Vue d'ensemble de .NET
- Différences de .NET par rapport à Delphi
- TsgcCustomGauge
- Types de jauges

Différences de .NET par rapport à Delphi

Le portage conserve les noms, les valeurs par défaut et le comportement de Delphi, les rubriques Delphi de cette aide valent donc aussi pour .NET. Voici les différences qui viennent du langage et des frameworks.

Langage

- Les **événements** sont des propriétés delegate avec les noms de Delphi (**OnThreshold**, **OnFormatValue**, **OnChange**, **OnTileClick**). Affectez-les avec `=` : un gestionnaire par événement, comme dans Delphi. Le paramètre `var` de **OnFormatValue** est un paramètre `ref`. Dans les événements d'un contrôle, Sender est le contrôle.
- Les **fonctions sans paramètre** restent des méthodes : **ActiveScaleCount()**, **ActivePointerCount()**, **NowMs()**, **GetGauge()**, **TsgcGaugeAnnunciatorTile.Flashing()**.
- **FormatSettings** est la struct **TsgcGaugeFormatSettings** : lisez-la, modifiez-la et réaffectez-la.
- Les **couleurs** sont des valeurs `uint` dans l'ordre ARGB (`0xFFE34948`). 0 signifie la couleur du thème, comme dans Delphi.
- Les **collections** dérivent de **TsgcGaugeCollection<T>** : un indexeur (`Scales[0]`), **Add**, **Delete**, **Clear**, **Count**, **BeginUpdate** et **EndUpdate**, et `foreach`. **Assign(object)** copie une collection ou un élément.
- Les fonctions libres des unités Delphi sont des méthodes statiques : **sgcGauges_Types** (mathématiques et easing), **sgcGauges_Render** (géométrie) et **sgcGauges_Raster** (PNG).
- Il n'y a pas de **Free** : les modèles sont des classes ordinaires. Les contrôles WinForms et WPF suivent leur framework ; un contrôle Avalonia arrête son minuteur quand il quitte l'arbre visuel.

Exemple

Sortie

- **SaveToSVG** est une méthode du modèle dans esegece.sgcGauges. **SaveToPNG** d'un modèle est une méthode d'extension dans esegece.sgcGauges.Rendering, car le PNG a besoin du raster SkiaSharp. Les contrôles ont les deux, à leur propre taille.
- Le SVG d'une jauge a les mêmes octets que le SVG Delphi de la même jauge. Le PNG est dessiné par SkiaSharp et peut différer du PNG Delphi par l'anticrénelage des bords.

Threads et temps

- **SetValue** et **Value** sont thread safe, comme dans Delphi. Les contrôles se redessinent et déclenchent **OnThreshold** dans le thread de l'interface via le **SynchronizationContext** capturé à leur création, à la place de **TThread.Queue**. Sur un modèle sans contrôle, **OnThreshold** s'exécute dans le thread qui a écrit la valeur.
- L'horloge de l'animation est un **Stopwatch** à la place de **sgcGaugeNowMs** ; **Clock** peut la remplacer par n'importe quel **TsgcGaugeClockFunc**.

Non porté

- Les aides de streaming des .dfm (les fonctions **Is*Stored**, les propriétés définies ZeroValues et **GetNamePath**) n'ont pas d'équivalent .NET. Les collections se remplissent en code.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- Rendu sans interface dans .NET
- TsgcCustomGauge

- Éditions

Liaison de données dans .NET

Une jauge affiche une valeur, la liaison de données consiste donc à amener une valeur d'un objet de l'application vers le contrôle. Chaque hôte le fait avec son propre système de liaison, sur la propriété de valeur du contrôle. Les échelles, les plages, les pointeurs et les seuils sont des collections et se définissent en code.

Hôte	Propriétés pouvant être liées	Sens par défaut
Win-Forms	**Value** (et toute propriété publique) via **Control.DataBindings**	De la source vers le contrôle avec **DataSourceUpdateMode.Never** ; les contrôles ne déclenchent aucun événement de changement.
WPF	Propriétés de dépendance **ValueProperty** (radiale, linéaire, numérique, LED, barre de LED, anneau de progression, VU meter) et **HeadingProperty** (compas)	Dans un seul sens. Avec **BindingMode.TwoWay**, une valeur que le modèle borne ou ramène en boucle est réécrite dans la source.
Avalo-nia	Propriétés stylées **ValueProperty** (tous les contrôles), **ActiveProperty** (LED) et **HeadingProperty** (compas)	Dans les deux sens : une valeur que le modèle borne ou ramène en boucle est réécrite dans la source.

La source peut changer depuis n'importe quel thread : le modèle est thread safe et le contrôle se redessine dans le thread de l'interface. Les liaisons WinForms elles-mêmes doivent être déclenchées dans le thread de l'interface, comme pour tout contrôle WinForms.

WinForms

WPF

En XAML : `<sgc:TsgcWpfRadialGauge Value="{Binding Speed}" />`, voir Démarrage rapide WPF. En code :

Avalonia

Flux

Pour piloter des jauges avec des messages WebSocket, MQTT ou des événements envoyés par le serveur sans view model, liez-les à une source de données de la passerelle, voir Passerelle sgcWebSockets pour .NET.

Voir aussi

- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Passerelle sgcWebSockets pour .NET

Rendu sans interface dans .NET

Chaque modèle de jauge se dessine sans contrôle, de sorte qu'un programme console, un service ou un serveur web peut écrire des images PNG et SVG de jauges, sous Windows, Linux et macOS. C'est la forme .NET de Sortie sans interface.

Méthodes

Méthode	Paquet	Description
SaveToSVG(Stream or string, int aWidth, int aHeight, double aScale = 1)	esegece.sgcGauges	Écrit un document SVG de la jauge. Aucune dépendance native.
SaveToPNG(Stream or string, int aWidth, int aHeight, double aScale = 1)	esegece.sgcGauges.Rendering	Méthode d'extension de **sgcGauges_Raster** : dessine sur un raster SkiaSharp vierge de Round(aWidth * aScale) sur Round(aHeight * aScale) pixels et écrit un PNG.
RenderToRaster(int aWidth, int aHeight, double aScale = 1, double aNowMs = -1)	esegece.sgcGauges.Rendering	Renvoie le **TsgcUICanvas_Raster** avec la jauge dessinée, pour dessiner davantage ou lire des pixels. L'appelant le libère.
Render(TsgcUICanvas aCanvas, int aWidth, int aHeight, double aScale = 1, double aNowMs = -1)	esegece.sgcGauges	Dessine sur n'importe quel canevas de sgcUICore.

Exemple

Remarques

- Un rendu sans interface dessine les valeurs cibles, jamais une image de l'animation, la sortie d'une même jauge a donc toujours les mêmes octets. Une horloge avec **Live** à false affiche son **Time**.
- aScale multiplie les pixels et conserve la mise en page : 200 sur 200 à l'échelle 2 donne un PNG de 400 sur 400 du même dessin.
- Sous Linux, esegece.sgcGauges.Rendering apporte les assets natifs de SkiaSharp ; aucun serveur X n'est nécessaire.

La démo ConsoleRender écrit un fichier PNG et un fichier SVG de chaque type de jauge, voir Démos .NET.

Voir aussi

- Vue d'ensemble de .NET
- Différences de .NET par rapport à Delphi
- Démos .NET
- Sortie sans interface
- Démarrage rapide sans interface

Passerelle sgcWebSockets pour .NET

Le paquet **esegece.sgcGauges.sgcWebSockets** (édition Registered) est le portage .NET de la passerelle sgcGaugesWS. Une source de données se branche sur un client de sgcWebSockets .NET, lit chaque message comme du JSON et écrit les valeurs de ses liaisons dans les jauges. Le comportement est celui décrit dans Vue d'ensemble de la passerelle.

Classe	Client	Lit
TsgcGaugeFeedSource_WebSocket	**Client** : un **TsgcWebSocketClient**, son événement **OnMessage**	Chaque message texte. **ProcessMessage(string)** applique un message manuellement.
TsgcGaugeFeedSource_MQTT	**Client** : un **TsgcWSProtocol_MQTT_Client**, son événement de publication	Le contenu des publications dont le sujet correspond au filtre **Topic** d'une liaison (jokers `+` et `#`). **ProcessPublish(topic, text)** en applique une manuellement.
TsgcGaugeFeedSource_SSE	**Client** : un **TsgcHTTP1Client**, son événement **OnSSEMessage**	Les données des événements dont le type est égal au **EventName** d'une liaison. **ProcessChunk(string)** et **ResetStream()** appliquent un flux manuellement.

Chaque source possède **Active**, **Bindings**, **OnValue** et **OnError**. Un **TsgcGaugeFeedBinding** possède **Gauge** (n'importe quel contrôle, ou tout objet qui implémente **IsgcGaugeHost**), **GaugeModel** (un modèle sans contrôle), **PointerIndex**, **ValuePath**, **FilterPath** et **FilterValue**, **Topic**, **EventName** et **Active**.

Exemple

Remarques

- Les sources ne se connectent ni ne s'abonnent jamais : l'application connecte le client, abonne les sujets MQTT et appelle **GetSSE** du client HTTP.
- Les événements du client sont des événements .NET : une source ajoute son propre gestionnaire avec `+=` et ne retire que celui-là, si bien que les gestionnaires de l'application continuent de fonctionner et s'exécutent dans l'ordre où ils ont été ajoutés.
- Les valeurs sont écrites dans le thread du client. Les jauges sont thread safe et se redessinent dans le thread de l'interface ; **OnValue** et **OnError** s'exécutent dans le thread du client, il faut donc poster vers le thread de l'interface avant de toucher un contrôle.
- Une liaison surveille l'événement **Disposed** d'un contrôle WinForms et efface **Gauge** quand le contrôle disparaît. Pour les contrôles WPF et Avalonia et pour **GaugeModel**, effacez la liaison avant la disparition de la jauge.
- Une chaîne numérique est lue avec '.' comme séparateur décimal, quelle que soit la culture ; true et false sont lus comme 1 et 0.

La démo MQTTLive exécute six jauges sur MQTT avec un petit broker dans le processus, voir Démos .NET.

Voir aussi

- Vue d'ensemble de la passerelle
- Source MQTT

- Source WebSocket
- Source d'événements envoyés par le serveur
- Liaison de données dans .NET
- Démos .NET

Démos .NET

Les démos .NET se trouvent dans `demos\Gauges` du dossier .NET de sgcUI, un dossier par démo avec un fichier de projet, Program.cs et un README.md. L'interface est construite en code.

Démo	Hôte	Montre
Gallery	WinForms	Les dix contrôles de jauge sur un formulaire : changement de thème, animation activée et désactivée, une simulation de valeur et un export PNG au double de la taille. Le seuil de vitesse à 200 km/h commute un voyant d'alarme et la tuile HIGH SPEED ; un clic sur une tuile l'acquitte.
GalleryWpf	WPF	La même galerie avec les contrôles TsgcWpf.
GalleryAvalonia	Avalonia	La même galerie avec les contrôles TsgcAvalonia.
HMIPanel	WinForms	Un procédé simulé avec deux cuves et deux pompes : jauges linéaires de niveau, jauges radiales de pression, afficheurs numériques de débit, LED des pompes et un annunciator déclenché par un seul gestionnaire **OnThreshold** partagé.
MQTTLive	WinForms	Six jauges alimentées par **TsgcGaugeFeedSource_MQTT**. Avec l'option loopback, Connect démarre un petit broker MQTT dans le processus, la démo n'a donc besoin ni de broker ni de réseau. Édition Registered uniquement.
ConsoleRender	Console	Un modèle de chaque type écrit dans des fichiers PNG et SVG, sans framework d'interface ; le pendant de la démo Headless de Delphi.

Compiler et exécuter

```
dotnet run --project Gallery\Gallery.csproj
```

Dans l'arborescence des sources de sgcUI, une démo se compile contre les projets de la bibliothèque dans l'édition Registered ; ajoutez `-p:SgcCommunity=true` pour l'édition Community. Dans le zip de la démo, elle restaure les paquets depuis nuget.org, par défaut les paquets Community ; définissez `-p:SgcGaugesEdition=Registered` pour les Registered.
Dans l'édition Community, les galeries montrent six jauges, HMIPanel liste les alarmes dans une liste à la place de l'annunciator et ConsoleRender écrit les six types Community. MQTTLive nécessite l'édition Registered et le paquet esegece.sgcWebSockets.

Autotest

Chaque démo accepte `--test result.txt` : elle exécute un autocontrôle hors écran, écrit une ligne PASS ou FAIL par contrôle et se termine avec le nombre de contrôles échoués. `--snapshot file.png` enregistre une image PNG et se termine.

Voir aussi

- Vue d'ensemble de .NET
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Rendu sans interface dans .NET

- Passerelle sgcWebSockets pour .NET

Dépannage

La valeur s'arrête à 100

Une valeur est bornée à la plage de son échelle lorsqu'elle est écrite, et l'échelle par défaut va de 0 à 100. Définissez Minimum et Maximum de l'échelle avant la valeur. Un changement ultérieur de la plage borne à nouveau les valeurs stockées, il ne restaure pas une valeur qui a été coupée.

La deuxième échelle ou aiguille n'est pas dessinée

L'édition Community ne dessine que la première échelle et la première aiguille. Les autres sont conservées dans les collections. Voir Éditions.

TsgcVCLClockGauge ou TsgcCompassGauge n'est pas déclaré

Le vumètre, l'horloge, la boussole et l'annunciator font partie de l'édition Registered. En Community, leurs unités sont vides et leurs contrôles ne sont pas enregistrés.

Un afficheur 14 segments affiche 7 segments

Les afficheurs 14 segments et matrice de points font partie de l'édition Registered. En Community, ils sont dessinés comme du 7 segments.

L'aiguille saute au lieu de se déplacer

Vérifiez Animation.Enabled et Animation.Duration. Il n'y a pas d'animation à la conception, pour les valeurs chargées depuis le fichier de fiche, pour `SetValue(aPointer, aValue, False)` et dans un rendu sans interface, qui dessine la valeur cible.

OnThreshold ne se déclenche pas

- Avec gtdRising et gtdFalling, l'événement ne se déclenche que lorsque le seuil devient actif. Utilisez gtdBoth pour obtenir aussi l'événement de retour à la normale.
- Après un événement, la valeur doit quitter la bande donnée par Hysteresis avant que le seuil puisse se déclencher à nouveau.
- Un changement du seuil lui-même met à jour Active sans événement.
- Vérifiez Enabled et PointerIndex. En Community, les seuils des aiguilles après la première ne sont pas évalués.

Aucune marque de seuil sur l'échelle

Un seuil ne dessine une marque que si sa Color n'est pas 0.

Les étiquettes affichent un '.' dans une locale à virgule décimale

Les jauges utilisent leurs propres FormatSettings, avec '.' par défaut, de sorte que la sortie ne dépend pas de la machine. Affectez FormatSettings du modèle, via la propriété Gauge d'un contrôle, pour le changer. Voir Thèmes.

Le texte est trop petit ou trop grand

Les tailles de police sont relatives à la taille de la jauge, le texte suit donc la taille du contrôle. Modifiez la Size de TitleFont, ValueFont ou LabelFont pour changer la proportion.

Écrire des valeurs depuis un thread de travail

Value, SetValue ainsi que Value et EndValue de l'aiguille sont thread-safe, et les contrôles se redessinent et appellent OnThreshold dans le thread principal. Toute autre propriété appartient au thread principal.

Exemple

```
uses
  SysUtils, Classes, Controls, Forms,
  sgcGauges_Types, sgcGauges_Model, sgcGauges_VCL;

type
  TSensorThread = class(TThread)
  private
    FGauge: TsgcVCLRadialGauge;
  protected
    procedure Execute; override;
  public
    constructor Create(aGauge: TsgcVCLRadialGauge);
  end;

constructor TSensorThread.Create(aGauge: TsgcVCLRadialGauge);
begin
  FGauge := aGauge;
  inherited Create(False);
end;

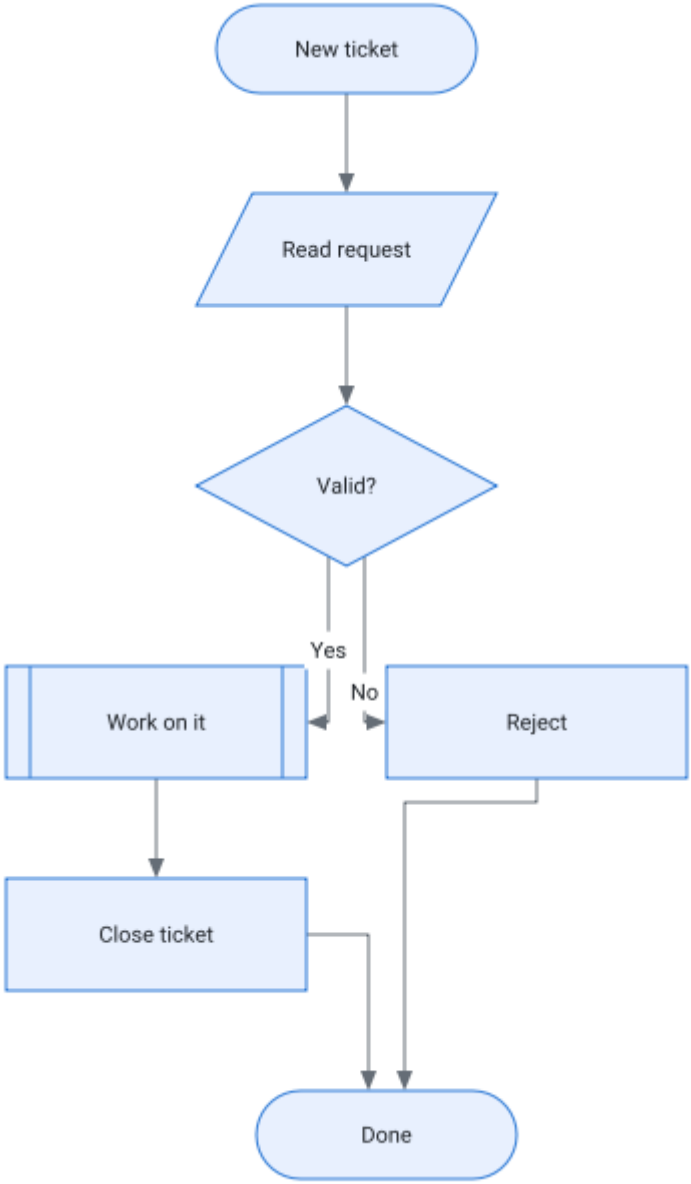
procedure TSensorThread.Execute;
begin
  while not Terminated do
  begin
    // Value is thread safe: no Synchronize needed
    FGauge.Value := Random(100);
    Sleep(250);
  end;
end;
```

Voir aussi

- Éditions
- Animation
- Seuils

Introduction

sgcDiagram ajoute un éditeur et une visionneuse de diagrammes aux applications Delphi et C++Builder, pour VCL et FireMonkey : organigrammes, diagrammes de flux et graphes de nœuds. L'utilisateur fait glisser des formes depuis une palette, trace des connecteurs entre elles, déplace, redimensionne et fait pivoter les nœuds, modifie leur texte sur place et annule chaque étape. Le même diagramme peut être construit, organisé et exporté par code, sans fenêtre, sur un serveur ou sous Linux64.
Tout se trouve dans un seul objet document, **TsgcDiagramDocument** : les nœuds, les connecteurs avec leurs ports et leurs étiquettes, les calques, les images et les styles par défaut. **TsgcVCLDiagramView** et **TsgcFMXDiagramView** affichent et modifient un document, **TsgcVCLDiagramPalette** et **TsgcFMXDiagramPalette** proposent les formes à faire glisser sur la vue, et **TsgcDiagramRenderer** écrit un document en PNG et en SVG sans vue. Un document est enregistré et chargé en JSON, et une vue enregistre son diagramme dans le fichier de la fiche.
sgcDiagram fait partie de la suite visuelle eSeGeCe. Il est dessiné avec **sgcUICore**, le paquet d'exécution partagé avec le canevas, le rastériseur, les polices, l'écrivain SVG et les thèmes clair et sombre, si bien qu'un diagramme a le même aspect sous VCL, sous FireMonkey et dans les fichiers exportés. sgcDiagram requiert sgcUICore, qui s'installe avec lui.



Éditions

sgcDiagram est fourni en Registered, Trial et Community. L'édition Community est un éditeur manuel complet : toutes les formes, les tracés personnalisés, les images, les trois types de connecteurs, les ports et le collage, les groupes et les conteneurs, l'annulation, le presse-papiers, JSON, PNG et SVG, les thèmes, le magnétisme et les repères, la palette et la visionneuse en lecture seule. Elle exclut les dispositions automatiques, le routage orthogonal qui évite les obstacles, la liaison de données et plus d'un calque. Voir Éditions.

Par où commencer

- Qu'est-ce que sgcDiagram : les types de diagrammes qu'il dessine et ce que l'utilisateur peut en faire.
- Architecture : le document, les vues, le contrôleur et les unités.
- Démarrage rapide VCL, Démarrage rapide FMX, Démarrage rapide C++Builder et Démarrage rapide Headless : un premier diagramme en quelques minutes.
- Les composants : TsgcVCLDiagramView, TsgcFMXDiagramView, TsgcVCLDiagramPalette, TsgcFMXDiagramPalette, TsgcDiagramDataBinding et leurs Événements.
- Les classes : TsgcDiagramDocument, TsgcDiagramNode, TsgcDiagramConnector, TsgcDiagramPort, TsgcDiagramStyle, TsgcDiagramLayer, TsgcDiagramController, TsgcDiagramRenderer et le Format JSON.
- Disposition en couches, Disposition en arbre et Liaison de données : organiser un diagramme automatiquement ou le construire à partir d'un jeu de données.
- Les fonctionnalités : Formes, Connecteurs et routage, Ports et collage, Édition et clavier, Magnétisme et repères, Annuler et presse-papiers, Calques, Sérialisation, Export, Thèmes et Mode visionneuse.
- Plateformes : les IDE, frameworks et systèmes d'exploitation pris en charge.
- Dépannage : les problèmes courants et leurs causes.

Qu'est-ce que sgcDiagram

sgcDiagram est un ensemble de composants de diagramme : un document de nœuds et de connecteurs, une vue interactive qui le modifie, une palette de formes et un exportateur. Il est destiné aux diagrammes que les applications métier affichent et laissent leurs utilisateurs modifier.

Ce que vous pouvez dessiner

- **Diagrammes de flux.** Les formes processus, décision, terminateur, données, document, processus prédéfini et cercle de connexion, des connecteurs avec des flèches et des étiquettes Oui et Non.
- **Organigrammes et arbres.** Des cadres avec un nom et un rôle, reliés par des connecteurs orthogonaux. L'édition Registered les organise avec une disposition en arbre et les construit à partir d'un jeu de données.
- **Graphes de nœuds.** Des nœuds avec des ports d'entrée et de sortie typés, des connecteurs courbes et votre propre dessin dans OnDrawNode, comme dans un éditeur de nœuds ou un concepteur de pipelines.
- **Croquis libres.** Rectangles, rectangles arrondis, ellipses, texte libre, images, formes personnalisées issues de données de chemin SVG et conteneurs qui contiennent d'autres nœuds.

Ce que l'utilisateur peut faire

- Faire glisser une forme de la palette vers la vue, ou la choisir et la dessiner avec l'outil de forme.
- Sélectionner par un clic, des clics avec Shift et Ctrl ou un cadre de sélection, puis déplacer, redimensionner, faire pivoter, ajuster avec les touches fléchées, grouper, supprimer, copier et coller.
- Tracer un connecteur depuis un port, coller ses extrémités à des ports ou au contour d'un nœud, déplacer ses étiquettes et ses waypoints.
- Modifier le texte d'un nœud sur place : double-clic, F2 ou simplement commencer à taper.
- Annuler et rétablir chaque action, faire défiler avec la molette, le bouton du milieu ou Space, zoomer avec Ctrl et la molette ou un pincement.

Ce que le code peut faire

- Construire un diagramme avec AddNode et AddConnector, modifier n'importe quelle propriété et regrouper les modifications en une seule étape d'annulation avec BeginTransaction et EndTransaction.
- L'enregistrer et le charger en JSON, comme fichier, flux ou chaîne.
- L'exporter en PNG et en SVG avec TsgcDiagramRenderer, y compris dans une application console ou sous Linux64.
- Écouter OnChange, OnSelectionChange, OnItemClick, OnConnecting et les autres événements pour réagir à l'utilisateur.

Ce qu'il ne fait pas

La version 1 n'a pas de pochoirs BPMN, UML ou réseau, pas de bassins de couloirs (swimlanes), pas de dispositions par forces ou circulaires, pas d'export PDF, pas d'échange de presse-papiers avec d'autres applications, pas de minicarte, de règles, de barres de défilement ni d'impression, pas d'import SVG ou Visio, et pas de texte enrichi dans les nœuds.

Voir aussi

- Architecture
- Éditions
- Démarrage rapide VCL

Architecture

sgcDiagram sépare le document, l'interaction et le dessin, si bien que le même code modifie un diagramme sous VCL, sous FireMonkey et dans un test sans fenêtre.

Partie	Rôle
TsgcDiagramDocument	Le modèle : Nodes, Connectors, Layers, Images, Selection, les NodeStyle et ConnectorStyle par défaut, Routing, GridSize, Theme et la pile d'annulation. Chaque modification est signalée une fois par paire BeginUpdate et EndUpdate via OnChange.
TsgcDiagramNode	Un cadre avec Shape, Left, Top, Width, Height, Angle, Text, Style, Ports, Layer, ContainerId, GroupId, ImageKey, PathData, Locked, Visible, Data et Tag. L'ordre Z est l'index dans Nodes.
TsgcDiagramConnector	Un lien entre deux nœuds ou deux points libres : Kind, SourceNodeId, SourcePortId, TargetNodeId, TargetPortId, SourcePoint, TargetPoint, Waypoints, StartArrow, EndArrow, ArrowSize, Labels et Style. Points contient le tracé calculé.
Router	Calcule, à la fin de chaque lot de modifications, le tracé de chaque connecteur dont les extrémités, les nœuds ou le type ont changé lorsque Routing.AutoRoute vaut True.
TsgcDiagramController	La machine à états d'interaction : test de survol, sélection, glissement, redimensionnement, rotation, connexion, magnétisme, repères, édition du texte, zoom et défilement. Elle est neutre vis-à-vis du framework et travaille en pixels de l'appareil à l'extérieur et en unités monde à l'intérieur.
TsgcDiagramPainter et le cache de la vue	Dessinent le document sur un TsgcUICanvas de sgcUICore. La vue conserve une image en cache du contenu, donc un défilement est une copie de pixels et une modification ne redessine que son rectangle sale.
TsgcVCLDiagramView, TsgcFMXDiagramView	Les contrôles. Ils possèdent un document (la propriété Diagram) et un Controller, et traduisent la souris, le clavier, les gestes et le glisser-déposer en appels au contrôleur.
TsgcVCLDiagramPalette, TsgcFMXDiagramPalette	Une grille d'éléments de forme groupés par catégorie. Un clic choisit l'outil de forme de la vue liée, un glissement dépose la forme dessus.
TsgcDiagramRenderer	Exporte un document en PNG et en SVG sans vue.

Coordonnées

Les unités monde sont des pixels au zoom 1 et à 96 PPI, avec y croissant vers le bas. Les angles des nœuds sont en degrés, dans le sens horaire, autour du centre du nœud. Une couleur est un TsgcUIColor, et 0 signifie la couleur du thème. ScreenToDiagram et DiagramToScreen de la vue convertissent entre les pixels de l'appareil du contrôle et les unités monde.

Notifications de modification

Le document collecte les modifications entre le BeginUpdate et le EndUpdate les plus externes et les signale une seule fois : d'abord à la vue qui le possède, qui redessine le rectangle sale, puis via OnChange avec un ensemble de valeurs TsgcDiagramChange (dgkGeometry, dgkStyle, dgkStructure, dgkSelection, dgkView) et OnSelectionChange. Une modification faite par code en dehors de BeginUpdate et EndUpdate est signalée immédiatement.

Unités

Unité	Contenu
sgcDiagram_Types	Les énumérations, TsgcDiagramPoint, TsgcDiagramRect, les aides géométriques comme sgcDiagramPoint et sgcDiagramBounds, et EsgcDiagramError.
sgcDiagram_Shapes	Le contour, le rectangle de texte et les ports par défaut de chaque forme, et l'analyseur de données de chemin SVG.
sgcDiagram_Model	TsgcDiagramDocument, les nœuds, connecteurs, ports, étiquettes, calques, images, styles et la sélection.
sgcDiagram_Undo, sgcDiagram_JSON, sgcDiagram_Router	La pile d'annulation, le format JSON et le presse-papiers, le routage des connecteurs.
sgcDiagram_Render	Le peintre, le cache de la vue et TsgcDiagramRenderer.
sgcDiagram_Palette, sgcDiagram_Controller	Le modèle de palette et le contrôleur d'interaction.
sgcDiagram_Layout, sgcDiagram_Data	Les dispositions en couches et en arbre et la liaison de données. Édition Registered ; unités vides dans Community.
sgcDiagram_VCL, sgcDiagram_FMX	La vue et la palette VCL et FireMonkey.

Voir aussi

- Qu'est-ce que sgcDiagram
- Plateformes
- Annuler et presse-papiers
- TsgcDiagramDocument
- TsgcDiagramController
- TsgcDiagramRenderer

Éditions

sgcDiagram est construit à partir de la même source pour chaque édition.

Édition	Contient
Registered	Tout : les vues et palettes pour VCL et FireMonkey, toutes les formes, les trois types de connecteurs, les ports, les groupes et conteneurs, l'annulation, le presse-papiers, JSON, PNG et SVG, les thèmes, le magnétisme, les repères et les fonctionnalités avancées listées ci-dessous. La page de palette SGC Diagram montre 5 composants.
Trial	Les mêmes fonctionnalités que Registered. Une build compilée avec SGC_TRIAL affiche un message indiquant qu'il s'agit d'une version d'essai lorsqu'un composant est créé après la fin de la fenêtre de publication indiquée dans la build. Rien d'autre ne change.
Community	Tout sauf les fonctionnalités avancées : un éditeur manuel complet de diagrammes de flux et d'organigrammes pour VCL, FireMonkey et l'usage sans fenêtre. La page de palette SGC Diagram montre 4 composants (les deux vues et les deux palettes).

Les fonctionnalités avancées

Les fonctionnalités avancées sont compilées sous le define SGC_DIAGRAM_ADVANCED, que sgcVer.inc désactive dans l'édition Community. Les listes d'unités des paquets sont les mêmes dans chaque édition : sgcDiagram_Layout et sgcDiagram_Data se compilent en unités vides dans Community.

Fonctionnalité	Dans l'édition Community
Disposition automatique : TsgcDiagramLayeredLayout et TsgcDiagramTreeLayout	Non disponible. Les nœuds restent là où l'utilisateur ou le code les place.
Routage orthogonal qui évite les obstacles, avec écartement des segments parallèles	Les connecteurs dgcOrthogonal sont routés en simples coudes qui n'évitent pas les nœuds, et RouteFailed reste False.
Liaison de données : TsgcDiagramDataBinding construit un diagramme à partir d'un jeu de données	Non disponible.
Calques au-delà du premier	Layers.Add lève EsgcDiagramError lorsqu'un calque existe déjà. Un fichier enregistré par l'édition Registered garde ses noms de calques et se charge et s'enregistre sans changement, mais chaque calque est dessiné, exporté et modifié comme visible et non verrouillé.

Un fichier de diagramme est le même dans chaque édition, donc un document créé avec l'édition Registered s'ouvre dans une build Community et inversement.

Voir aussi

- Connecteurs et routage
- Calques
- Plateformes
- Disposition en couches
- Disposition en arbre

- TsgcDiagramDataBinding

Plateformes

sgcDiagram est une seule arborescence de sources pour chaque IDE et plateforme. Le document, le routeur, le contrôleur, le peintre et le moteur de rendu n'utilisent que la RTL et sgcUICore, ils sont donc le même code sur chaque cible ; seules les vues et les palettes sont liées à un framework.

Cible	Prise en charge
Delphi VCL	Delphi 7 à Delphi 13, Win32 et Win64 : TsgcVCLDiagramView et TsgcVCLDiagramPalette.
Delphi FireMonkey	Delphi XE6 à Delphi 13 pour Windows, macOS, iOS et Android : TsgcFMXDiagramView et TsgcFMXDiagramPalette. Compilé sous SGC_DIAGRAM_FMX.
Linux64	Delphi 10.2 et versions ultérieures, compilé sous SGC_DIAGRAM_LINUX : le document, JSON, le routage, les dispositions et TsgcDiagramRenderer, de sorte qu'un serveur peut construire un diagramme et produire des fichiers PNG et SVG. Voir Démarrage rapide Headless.
C++Builder	Les mêmes composants avec les en-têtes .hpp générés. Voir Démarrage rapide C++Builder.
.NET	Un portage C# avec la même API pour WinForms, WPF et Avalonia, la vue et la palette sur chaque hôte et la sortie JSON, PNG et SVG sans fenêtre, sur .NET Framework 4.7.2 et .NET 8 et 9, voir Vue d'ensemble de .NET.

Unités par framework

Unités	Framework
sgcDiagram_Const, sgcDiagram_Types, sgcDiagram_Shapes, sgcDiagram_Model, sgcDiagram_Undo, sgcDiagram_JSON, sgcDiagram_Router, sgcDiagram_Layout, sgcDiagram_Render, sgcDiagram_Palette, sgcDiagram_Controller	Aucun. Le modèle, les moteurs, le peintre et le format de fichier.
sgcDiagram_Data	Seulement l'unité DB (dbrtl), aucune unité VCL ou FireMonkey.
sgcDiagram_VCL	VCL, Windows uniquement.
sgcDiagram_FMX	FireMonkey, Delphi XE6 et versions ultérieures.

Packages

sgcDiagram fournit un package d'exécution indépendant du framework et un package d'exécution par framework, de sorte qu'une application FireMonkey pour Android, iOS ou macOS ne lie jamais la VCL. Chaque nom de package se termine par le suffixe de l'IDE, par exemple sgcDiagramD13.dpk pour Delphi 13 et sgcDiagramC13.cbproj pour C++Builder 13.

Package	Type	Contenu	Requiert

sgcDiagram	Exécution, indépendant du framework	Chaque unité du produit qui n'utilise ni la VCL ni FireMonkey : le document, les formes, l'annulation, JSON, le routeur, les dispositions, le moteur de rendu, le modèle de palette, le contrôleur et, hors de l'édition Community, TsgcDiagramDataBinding (sgcDiagram_Data).	rtl, dbrtl, sgcUICore
sgcDiagramVCL	Exécution, VCL	sgcDiagram_VCL (TsgcVCLDiagramView, TsgcVCLDiagramPalette)	rtl, vcl, sgcUICore, sgcUICoreVCL, sgcDiagram
sgcDiagramFMX	Exécution, FireMonkey, à partir de XE6	sgcDiagram_FMX (TsgcFMXDiagramView, TsgcFMXDiagramPalette)	rtl, fmx, sgcUICore, sgcUICoreFMX, sgcDiagram
dclsgcDiagram	Conception, VCL	Enregistre TsgcVCLDiagramView, TsgcVCLDiagramPalette et, hors de l'édition Community, TsgcDiagramDataBinding dans la page de palette **SGC Diagram**, et ajoute les verbes "Load Diagram...", "Save Diagram..." et "Clear Diagram" à l'éditeur de composant de la vue.	designide, rtl, vcl, sgcDiagram, sgcDiagramVCL
dclsgcDiagramFMX	Conception, FireMonkey, à partir de XE6	Enregistre TsgcFMXDiagramView et TsgcFMXDiagramPalette dans la page de palette **SGC Diagram**, avec les mêmes verbes.	designide, rtl, fmx, sgcDiagram, sgcDiagramFMX, dclsgcDiagram

sgcUICore suit la même répartition : sgcUICore, sgcUICoreVCL et sgcUICoreFMX, sans package de conception. dclsgcDiagramFMX requiert dclsgcDiagram, installez donc d'abord dclsgcDiagram. sgcDiagram n'a pas de package passerelle vers sgcWebSockets.

Versions de l'IDE et plateformes cibles

Delphi	sgcDiagram	sgcDiagramVCL	sgcDiagramFMX	dclsgcDiagram, dclsgcDiagramFMX
7, 2007, 2009, 2010, XE	Win32	Win32	Aucune	Win32
XE2 à XE5	Win32, Win64	Win32, Win64	Aucune	Win32
XE6 à 10.2	Win32, Win64	Win32, Win64	Win32, Win64	Win32
10.3	Win32, Win64, Linux64, OSX64	Win32, Win64	Win32, Win64, OSX64	Win32
10.4	Win32, Win64, Linux64, OSX64, Android, Android64, iOSDevice64	Win32, Win64	Win32, Win64, OSX64, Android, Android64, iOSDevice64	Win32
11	Win32, Win64, Linux64, OSX64, OSXARM64, Android, Android64, iOSDevice64, iOSSimARM64	Win32, Win64	Win32, Win64, OSX64, OSXARM64, Android, Android64, iOSDevice64, iOSSimARM64	Win32

12, 13	Comme 11	Win32, Win64	Comme 11	Win32, Win64

De XE6 à 10.2, le package FireMonkey est compilé pour Windows uniquement. Les packages de conception sont chargés par l'IDE, ils sont donc compilés pour la plateforme de l'IDE uniquement.

C++Builder	sgcDiagram, sgcDiagramVCL	sgcDiagramFMX	dclsgcDiagram, dclsgcDiagramFMX
2007, 2010, XE, XE2	Win32	Aucune	Win32 (dclsgcDiagram uniquement)
XE3 à XE5	Win32, Win64	Aucune	Win32 (dclsgcDiagram uniquement)
XE6 à 12	Win32, Win64	Win32, Win64	Win32
13	Win32, Win64, Win64x	Win32, Win64, Win64x	Win32, Win64x

Les projets C++Builder portent les mêmes noms avec le suffixe C, par exemple sgcDiagramC13.cbproj, sgcDiagramVCLC13.cbproj, sgcDiagramFMXC13.cbproj, dclsgcDiagramC13.cbproj et dclsgcDiagramFMXC13.cbproj. Les packages C++Builder sont compilés pour Windows uniquement. Les notes C++Builder qui s'appliquent à chaque produit sgcUI (packages d'exécution sous Win64, C++Builder 10.3 sous Win64, noms de fichiers des packages Win64x) se trouvent dans le sujet Plateformes et packages de l'aide de la suite sgcUI.

FireMonkey sur Android, iOS et macOS

Les contrôles FireMonkey (TsgcFMXDiagramView et TsgcFMXDiagramPalette) ciblent macOS à partir de Delphi 10.3, Android et iOS à partir de Delphi 10.4, et macOS ARM et le simulateur iOS à partir de Delphi 11. Dans un projet FireMonkey pour ces plateformes :

- Utilisez les contrôles FireMonkey. Les contrôles VCL et sgcDiagramVCL ne fonctionnent que sous Windows.
- Déposez le contrôle sur un formulaire, ou ajoutez son unité à la clause uses : sgcDiagram_FMX.
- Sélectionnez la plateforme cible dans la fenêtre Projets. L'installateur ajoute le dossier Source de sgcUI et les unités compilées de chaque plateforme (LibD<version>\<platform>) au chemin de bibliothèque de chaque plateforme qu'il compile. Sans l'installateur, ajoutez le dossier Source de sgcUI au chemin de bibliothèque de la plateforme cible (Outils, Options, Langage, Delphi, Bibliothèque).
- Les applications Android et iOS sont liées statiquement, aucun package d'exécution n'est donc déployé.

Linux64

Le package indépendant du framework sgcDiagram est compilé pour Linux64 à partir de Delphi 10.3. Une application Linux64, par exemple un programme console ou un serveur, utilise le document, JSON, le routage, les dispositions et TsgcDiagramRenderer pour construire un diagramme et produire des fichiers PNG et SVG. Aucun package VCL ou FireMonkey n'est compilé pour Linux64, car FireMonkey sous Linux nécessite FMX pour Linux (FMXLinux), un module complémentaire tiers.

Remarques par plateforme

- **Haute résolution (DPI).** La vue VCL s'adapte au PPI du moniteur et la vue FireMonkey à l'échelle de la fiche. Le zoom 1 affiche une unité monde par pixel indépendant de l'appareil.
- **Tactile.** Sous FireMonkey, un pincement zoome autour de son centre et un doigt sur un espace vide fait défiler la vue. Un doigt sur un nœud le fait glisser, y compris sur un écran tactile Windows.
- **Glisser-déposer depuis la palette.** Sous iOS et Android, il n'y a pas de service de glisser-déposer : appuyez sur un élément de la palette, puis sur la vue pour placer la forme.
- **Touche Tab.** Tab et Shift+Tab font parcourir les éléments à la sélection sous VCL. Une fiche FireMonkey traite Tab avant le contrôle, donc ces touches y déplacent le focus.

- **Delphi 7 et Delphi 2007.** Ces versions utilisent des chaînes Ansi, donc le texte en dehors de la page de codes du système est perdu lors du chargement d'un fichier JSON.
- **Éditions.** SGC_DIAGRAM_ADVANCED, que sgcVer.inc désactive dans l'édition Community, décide des fonctionnalités avancées sur chaque plateforme. Voir Éditions.

Voir aussi

- Architecture
- Export
- Dépannage

Démarrage rapide VCL

Installez les paquets sgcDiagram pour votre version de Delphi. Les composants apparaissent sur la page de palette **SGC Diagram**.

À la conception

- Déposez une **TsgcVCLDiagramPalette** sur la fiche et mettez Align à alLeft.
- Déposez une **TsgcVCLDiagramView** et mettez Align à alClient.
- Définissez la propriété View de la palette sur la vue.
- Lancez l'application. Faites glisser des formes depuis la palette, tracez des connecteurs depuis les ports qui apparaissent lorsque le pointeur est au-dessus d'un nœud, double-cliquez sur un nœud pour modifier son texte.

Le menu contextuel de la vue à la conception a des commandes pour charger un diagramme depuis un fichier JSON, l'enregistrer et l'effacer. Le diagramme est stocké dans le dfm comme propriété DiagramJSON.

Dans le code

```
uses
  Forms, Controls,
  sgcDiagram_Types, sgcDiagram_Model, sgcDiagram_VCL;

type
  TFormMain = class(TForm)
  private
    FPalette: TsgcVCLDiagramPalette;
    FView: TsgcVCLDiagramView;
  public
    procedure CreateDiagram;
  end;

procedure TFormMain.CreateDiagram;
var
  oStart, oCheck, oDone: TsgcDiagramNode;
begin
  FPalette := TsgcVCLDiagramPalette.Create(Self);
  FPalette.Parent := Self;
  FPalette.Align := alLeft;
  FPalette.Width := 180;
  FView := TsgcVCLDiagramView.Create(Self);
  FView.Parent := Self;
  FView.Align := alClient;
  FPalette.View := FView;
  // three nodes and two connectors, one undo step
  FView.Diagram.BeginTransaction('Create');
  try
    oStart := FView.Diagram.AddNode(dgsTerminator, 40, 40, 140, 50, 'Start');
    oCheck := FView.Diagram.AddNode(dgsDecision, 30, 140, 160, 90, 'Valid?');
    oDone := FView.Diagram.AddNode(dgsProcess, 40, 290, 140, 60, 'Save');
    FView.Diagram.AddConnector(oStart, oCheck);
    FView.Diagram.AddConnector(oCheck, oDone).Labels.Add.Text := 'Yes';
  finally
    FView.Diagram.EndTransaction;
  end;
  FView.ZoomToFit;
end;
```

Étapes suivantes

- Enregistrez le diagramme avec FView.Diagram.SaveToFile, voir Sérialisation.
- Exportez-le avec FView.SaveToPNG et FView.SaveToSVG, voir Export.
- Basculez la vue en visionneuse en lecture seule avec ReadOnly, voir Mode visionneuse.

Voir aussi

- Démarrage rapide FMX
- Démarrage rapide Headless
- Édition et clavier
- TsgcVCLDiagramView
- TsgcVCLDiagramPalette

Démarrage rapide FMX

Les composants FireMonkey sont **TsgcFMXDiagramView** et **TsgcFMXDiagramPalette**, sur la page de palette **SGC Diagram** de Delphi XE6 et ultérieur. Ils ont les mêmes propriétés, méthodes et événements que les composants VCL, et la même classe de document, de sorte que le code qui travaille sur la propriété Diagram est partagé.

À la conception

- Déposez une TsgcFMXDiagramPalette avec Align Left et une TsgcFMXDiagramView avec Align Client.
- Définissez la propriété View de la palette sur la vue.
- Lancez l'application sous Windows ou macOS et faites glisser des formes sur la vue. Sous iOS et Android, appuyez sur un élément de la palette, puis sur la vue.

Dans le code

```
uses
  FMX.Forms, FMX.Types,
  sgcDiagram_Types, sgcDiagram_Model, sgcDiagram_FMX;

type
  TFormMain = class(TForm)
  private
    FPalette: TsgcFMXDiagramPalette;
    FView: TsgcFMXDiagramView;
  public
    procedure CreateDiagram;
  end;

procedure TFormMain.CreateDiagram;
var
  oA, oB: TsgcDiagramNode;
begin
  FPalette := TsgcFMXDiagramPalette.Create(Self);
  FPalette.Parent := Self;
  FPalette.Align := TAlignLayout.Left;
  FPalette.Width := 180;
  FView := TsgcFMXDiagramView.Create(Self);
  FView.Parent := Self;
  FView.Align := TAlignLayout.Client;
  FPalette.View := FView;
  oA := FView.Diagram.AddNode(dgsRoundRect, 40, 40, 140, 60, 'Order');
  oB := FView.Diagram.AddNode(dgsRoundRect, 260, 40, 140, 60, 'Invoice');
  FView.Diagram.AddConnector(oA, oB);
  FView.ZoomToFit;
end;
```

Tactile

- Un pincement zoome autour de son centre.
- Un doigt sur un espace vide fait défiler la vue, un doigt sur un nœud le fait glisser.

Voir aussi

- Démarrage rapide VCL
- Plateformes
- TsgcFMXDiagramView
- TsgcFMXDiagramPalette

Démarrage rapide C++Builder

Installez les paquets sgcDiagram pour votre version de C++Builder, ajoutez `sgcUI\Source` et le dossier des unités compilées, par exemple `sgcUI\LibD13\win32` (`sgcUI\LibD13\win64` pour 64 bits), aux chemins d'inclusion et de bibliothèque du projet, et utilisez les composants de la page de palette **SGC Diagram** exactement comme dans Delphi.

À savoir en C++

- Les énumérations n'ont pas de portée : écrivez `dgsDecision`, `dgcCurved` ou `dgtSelect`.
- Les exemples de cette aide passent explicitement chaque argument d'une méthode, par exemple `AddConnector(Start, Check, "", "")`, de sorte qu'ils se compilent avec n'importe quelle version des en-têtes générés.
- Les événements sont des closures, leurs gestionnaires sont donc des fonctions membres __fastcall d'une classe. Un paramètre var tel que aAccept est une référence bool.
- TsgcDiagramRenderer est une classe simple : créez-la avec `new TsgcDiagramRenderer()` et supprimez-la.

Dans le code

Voir aussi

- Démarrage rapide VCL
- Plateformes

Démarrage rapide Headless

Un diagramme n'a pas besoin de vue. TsgcDiagramDocument et TsgcDiagramRenderer n'utilisent que la RTL et sgcUICore, si bien qu'une application console, un service ou un serveur Linux64 peut construire un diagramme, calculer son tracé et écrire des fichiers PNG, SVG et JSON.

- Créez le document avec `TsgcDiagramDocument.Create` (sans propriétaire) et libérez-le vous-même.
- Le tracé des connecteurs est calculé à la fin de chaque lot de modifications tant que Routing.AutoRoute vaut True (la valeur par défaut). Appelez RouteAll pour recalculer chaque connecteur.
- Le moteur de rendu exporte les limites du contenu plus une marge, 20 unités monde par défaut, à l'échelle que vous passez.

Exemple

```
uses
  SysUtils, Classes,
  sgcDiagram_Types, sgcDiagram_Model, sgcDiagram_Render;

procedure ExportOrderFlow(const aFolder: string);
var
  oDocument: TsgcDiagramDocument;
  oRenderer: TsgcDiagramRenderer;
  oStart, oPay, oShip: TsgcDiagramNode;
begin
  oDocument := TsgcDiagramDocument.Create;
  try
    oDocument.BeginUpdate;
    try
      oStart := oDocument.AddNode(dgsTerminator, 0, 0, 140, 50, 'New order');
      oPay := oDocument.AddNode(dgsProcess, 0, 100, 140, 60, 'Take payment');
      oShip := oDocument.AddNode(dgsProcess, 0, 210, 140, 60, 'Ship');
      oDocument.AddConnector(oStart, oPay);
      oDocument.AddConnector(oPay, oShip);
    finally
      oDocument.EndUpdate; // the connectors are routed here
    end;
    oRenderer := TsgcDiagramRenderer.Create;
    try
      oRenderer.SaveToPNGFile(oDocument, IncludeTrailingPathDelimiter(aFolder) + 'order.png', 2);
      oRenderer.SaveToSVGFile(oDocument, IncludeTrailingPathDelimiter(aFolder) + 'order.svg');
    finally
      oRenderer.Free;
    end;
    oDocument.SaveToFile(IncludeTrailingPathDelimiter(aFolder) + 'order.json');
  finally
    oDocument.Free;
  end;
end;
```

Voir aussi

- Export
- Sérialisation
- Plateformes
- TsgcDiagramDocument
- TsgcDiagramRenderer
- Disposition en couches

TsgcVCLDiagramView

TsgcVCLDiagramView (unité sgcDiagram_VCL) est le contrôle VCL qui affiche et modifie un diagramme. Il possède un TsgcDiagramDocument, publié comme Diagram, et un TsgcDiagramController qui gère la souris, le clavier et le glisser-déposer.

Propriétés

Propriété	Signification
Diagram	Le TsgcDiagramDocument affiché et modifié par la vue. Il est sérialisé dans le fichier de la fiche comme propriété DiagramJSON, écrite seulement lorsque le diagramme n'est pas vide.
Tool	L'outil du pointeur : dgtSelect (par défaut), dgtPan, dgtConnector ou dgtShape.
ShapeTool	La forme dessinée par l'outil dgtShape, dgsRectangle par défaut.
ReadOnly	True transforme la vue en visionneuse : la sélection, le survol, le zoom et le défilement fonctionnent, la modification non.
Zoom, MinZoom, MaxZoom	Le facteur de zoom et ses limites (0.05 et 8 par défaut).
ShowGrid	Dessine la grille de Diagram.GridSize (True par défaut).
SnapToGrid, SnapToGuides	Aimante les nœuds déplacés et redimensionnés sur la grille et sur les bords, les centres et les écarts égaux des nœuds voisins (les deux à True par défaut).
ShowPorts	Quand les ports des nœuds sont dessinés : dgvNever, dgvHover (par défaut) ou dgvAlways.
Theme	Le thème du diagramme, la même valeur que Diagram.Theme (non stocké séparément).
FontFile	Un fichier de police utilisé pour dessiner le texte, la même police dans la vue et dans l'export.
MaxFPS	La fréquence de redessin maximale du contrôle.
Controller	Public. Le TsgcDiagramController qui transforme le pointeur et le clavier en modifications.

Méthodes

Méthode	Effet
ZoomToFit, ZoomToSelection	Zoomer pour que tout le diagramme, ou la sélection, remplisse la vue.
CenterOn(aPoint)	Fait défiler un point monde vers le centre de la vue.
ScreenToDiagram, DiagramToScreen	Convertissent entre les pixels de l'appareil de la vue et les unités monde.

Undo, Redo	Annulent ou rétablissent la dernière étape du document.
CopyToClipboard, CutToClipboard, PasteFromClipboard	Le presse-papiers du diagramme, voir Annuler et presse-papiers.
DeleteSelection	Supprime les éléments sélectionnés en une seule étape d'annulation.
EditText(aNode)	Ouvre l'éditeur sur place sur le texte d'un nœud ; renvoie False lorsque la modification n'est pas autorisée.
SaveToPNG, SaveToSVG	Exportent vers un fichier ou un flux, avec une échelle facultative (1 par défaut).

Événements

La vue publie OnChange, OnSelectionChange, OnItemClick, OnItemDblClick, OnHover, OnConnecting, OnConnected, OnDrawNode, OnTextEditing, OnTextEdited et OnZoomChange, décrits dans Événements.

Remarques

- La vue n'a pas de barres de défilement. La molette fait défiler, Shift et la molette font défiler horizontalement, Ctrl et la molette zooment autour du pointeur, et le bouton du milieu, Space ou l'outil dgtPan déplacent la vue, voir Édition et clavier.
- L'éditeur de texte est un mémo sur place : Enter ou la sortie du mémo valide, Esc annule et Shift+Enter ajoute une ligne.
- Le dépôt d'un élément de la TsgcVCLDiagramPalette crée le nœud, ou le fragment, au point de dépôt.

Exemple

```
uses
  Forms,
  sgcDiagram_Types, sgcDiagram_Model, sgcDiagram_VCL;

procedure AddStepAtCentre(aView: TsgcVCLDiagramView);
var
  vPoint: TsgcDiagramPoint;
  oNode: TsgcDiagramNode;
begin
  // the world point under the centre of the control
  vPoint := aView.ScreenToDiagram(sgcDiagramPoint(aView.ClientWidth / 2,
    aView.ClientHeight / 2));
  oNode := aView.Diagram.AddNode(dgsProcess, vPoint.X - 60, vPoint.Y - 30,
    120, 60, 'New step');
  aView.Diagram.Selection.Clear;
  aView.Diagram.Selection.Add(oNode);
  aView.EditText(oNode);
end;

procedure ExportView(aView: TsgcVCLDiagramView);
begin
  aView.SaveToPNG('diagram.png', 2);
  aView.SaveToSVG('diagram.svg');
end;
```

Voir aussi

- TsgcFMXDiagramView
- TsgcVCLDiagramPalette

- TsgcDiagramDocument
- Démarrage rapide VCL

TsgcFMXDiagramView

TsgcFMXDiagramView (unité sgcDiagram_FMX) est le contrôle FireMonkey qui affiche et modifie un diagramme sous Windows, macOS, Linux, iOS et Android. Il a les mêmes propriétés, méthodes et événements que TsgcVCL-DiagramView : Diagram, Tool, ShapeTool, ReadOnly, Zoom, MinZoom, MaxZoom, ShowGrid, SnapToGrid, SnapTo-Guides, ShowPorts, Theme, FontFile, MaxFPS, Controller, les méthodes de zoom, de presse-papiers, d'annulation et d'export, ainsi que OnChange, OnSelectionChange, OnItemClick, OnItemDblClick, OnHover, OnConnecting, On-Connected, OnDrawNode, OnTextEditing, OnTextEdited et OnZoomChange, décrits dans Événements.

Différences avec la vue VCL

- Le diagramme est sérialisé dans le fichier fmx comme propriété DiagramJSON, écrite seulement lorsque le diagramme n'est pas vide.
- Les positions sont en pixels de l'appareil : les coordonnées locales du contrôle multipliées par son échelle.
- Un pincement zoome autour du centre du pincement. Un doigt qui commence sur un élément le fait glisser ; un doigt qui commence sur un espace vide fait défiler la vue.
- La molette FMX n'a pas de position, donc elle zoome autour de la dernière position du pointeur.
- iOS et Android n'ont pas de service de glisser-déposer : appuyez sur un élément de la palette pour l'armer, puis appuyez sur la vue pour placer la forme.
- La fiche FMX traite Tab avant le contrôle ayant le focus, donc Tab et Shift+Tab ne parcourent pas les éléments.
- L'éditeur de texte est un simple TMemo FMX qui suit le style de la fiche.

Exemple

```
uses
  FMX.Forms, FMX.Types,
  sgcDiagram_Types, sgcDiagram_Model, sgcDiagram_FMX;

function CreateTouchViewer(aForm: TForm; const aFileName: string)
  : TsgcFMXDiagramView;
begin
  Result := TsgcFMXDiagramView.Create(aForm);
  Result.Parent := aForm;
  Result.Align := TAlignLayout.Client;
  Result.ReadOnly := True;
  Result.ShowGrid := False;
  Result.ShowPorts := dgvNever;
  Result.MinZoom := 0.25;
  Result.Diagram.LoadFromFile(aFileName);
  Result.ZoomToFit;
end;
```

Voir aussi

- TsgcVCLDiagramView
- TsgcFMXDiagramPalette
- Démarrage rapide FMX
- Mode visionneuse

TsgcVCLDiagramPalette

TsgcVCLDiagramPalette (unité sgcDiagram_VCL) affiche les formes que l'utilisateur peut ajouter à une TsgcVCL-DiagramView, par clic ou par glisser-déposer.

Propriétés

Propriété	Signification
View	La vue de diagramme qui reçoit les éléments cliqués ou glissés.
Items	Les éléments, une collection de TsgcDiagramPaletteItem affichée dans une grille, groupée par Category.
IconSize	La taille de l'icône d'un élément en pixels, 32 par défaut.
ShowCap-tions	Dessine la légende sous chaque icône (True par défaut).
Theme	utLight (par défaut), utDark ou utCustom.
MaxFPS	La fréquence de redessin maximale du contrôle.
Model	Public. Le modèle de palette derrière le contrôle : disposition, test de survol et charge utile du glissement.
DragPay-load	Public. La charge utile du glissement en cours : le JSON de l'élément glissé.

Éléments

TsgcDiagramPaletteItem	Signification
Category, Caption	L'en-tête du groupe et le texte sous l'icône.
Shape, Width, Height, Text, Style, PathData	Le nœud que l'élément crée (PathData pour dgsPath, dans un cadre 0..100).
Fragment	Un fragment JSON, tel que renvoyé par Diagram.CopySelection. Un élément avec un fragment dépose le fragment entier.
Payload	En lecture seule. Le JSON déposé par l'élément.

Événements

Événement	Quand
OnItemClick(Sender; aItem)	Un élément est cliqué.

Remarques

- Un clic sur un élément met le Tool de la vue à dgtShape et son ShapeTool à la forme de l'élément ; un clic ou un glissement sur la vue crée alors le nœud, et Esc revient à dgtSelect. Un élément fragment n'est placé que par glisser-déposer.
- Faire glisser un élément sur la vue affiche un aperçu et dépose le nœud, ou le fragment, au pointeur en une seule étape d'annulation. La charge utile ne passe jamais par le presse-papiers du système.
- Items.AddShape(aCategory, aCaption, aShape, aWidth, aHeight, aText) et Items.AddFragment(aCategory, aCaption, aFragment) ajoutent des éléments par code. LoadDefaultItems restaure les éléments par défaut.

Exemple

```
uses
  Controls, Forms,
  sgcUICore_Theme, sgcDiagram_Types, sgcDiagram_Palette, sgcDiagram_VCL;

procedure CreateFlowPalette(aForm: TForm; aView: TsgcVCLDiagramView);
var
  oPalette: TsgcVCLDiagramPalette;
  oItem: TsgcDiagramPaletteItem;
begin
  oPalette := TsgcVCLDiagramPalette.Create(aForm);
  oPalette.Parent := aForm;
  oPalette.Align := alLeft;
  oPalette.Width := 200;
  oPalette.View := aView;
  oPalette.Items.Clear;
  oPalette.Items.AddShape('Flow', 'Step', dgsProcess, 120, 60, 'Step');
  oPalette.Items.AddShape('Flow', 'Choice', dgsDecision, 140, 80, '?');
  oItem := oPalette.Items.AddShape('Notes', 'Note', dgsText, 160, 40, 'Note');
  oItem.Style.Fill := sgcUIARGB($FF, $FF, $F5, $9D);
  // the current selection of the view as a reusable block
  if aView.Diagram.Selection.Count > 0 then
    oPalette.Items.AddFragment('Blocks', 'My block',
      aView.Diagram.CopySelection);
end;
```

Voir aussi

- TsgcFMXDiagramPalette
- TsgcVCLDiagramView
- Formes

TsgcFMXDiagramPalette

TsgcFMXDiagramPalette (unité sgcDiagram_FMX) est la palette FireMonkey pour une TsgcFMXDiagramView. Elle a les propriétés, éléments et événements de TsgcVCLDiagramPalette : View, Items, IconSize, ShowCaptions, Theme, MaxFPS, Model, DragPayload, OnItemClick et LoadDefaultItems.

Différences avec la palette VCL

- Sur les plateformes de bureau, un glissement démarre le glisser-déposer FMX de la fiche ; la vue affiche l'aperçu tant que le pointeur la survole.
- Sous iOS et Android, il n'y a pas de service de glisser-déposer : un appui sur un élément arme l'outil de forme de la vue, et un appui sur la vue place la forme.

Remarques

- Un clic sur un élément met le Tool de la vue à dgtShape et son ShapeTool à la forme de l'élément ; un clic ou un glissement sur la vue crée alors le nœud, et Esc revient à dgtSelect. Un élément fragment n'est placé que par glisser-déposer.
- Faire glisser un élément sur la vue affiche un aperçu et dépose le nœud, ou le fragment, au pointeur en une seule étape d'annulation. La charge utile ne passe jamais par le presse-papiers du système.
- Items.AddShape(aCategory, aCaption, aShape, aWidth, aHeight, aText) et Items.AddFragment(aCategory, aCaption, aFragment) ajoutent des éléments par code. LoadDefaultItems restaure les éléments par défaut.

Exemple

```
uses
  FMX.Forms, FMX.Types,
  sgcDiagram_Types, sgcDiagram_Palette, sgcDiagram_FMX;

procedure CreateOrgPalette(aForm: TForm; aView: TsgcFMXDiagramView);
var
  oPalette: TsgcFMXDiagramPalette;
begin
  oPalette := TsgcFMXDiagramPalette.Create(aForm);
  oPalette.Parent := aForm;
  oPalette.Align := TAlignLayout.Left;
  oPalette.Width := 160;
  oPalette.IconSize := 40;
  oPalette.View := aView;
  oPalette.Items.Clear;
  oPalette.Items.AddShape('People', 'Manager', dgsRoundRect, 160, 60, 'Manager');
  oPalette.Items.AddShape('People', 'Employee', dgsRectangle, 140, 50, 'Employee');
end;
```

Voir aussi

- TsgcFMXDiagramView
- TsgcVCLDiagramPalette

TsgcDiagramDataBinding

TsgcDiagramDataBinding (unité sgcDiagram_Data) construit un diagramme à partir de lignes, typiquement un organigramme ou un diagramme de flux : chaque ligne devient un nœud et chaque clé parente un connecteur du parent vers l'enfant. Il est neutre vis-à-vis du framework et fonctionne avec la vue VCL, la vue FMX ou un document seul. Les noms de champs et de colonnes ne sont pas sensibles à la casse.

Propriétés

Propriété	Signification
DataSource	La TDataSource dont le DataSet fournit les lignes.
Table	Public. Une IsgcUITable qui fournit les lignes ; lorsqu'elle est assignée, elle est lue à la place de DataSource. Mettez-la à nil avant que l'objet table soit libéré.
KeyField	Le champ contenant la clé unique d'une ligne. La clé devient l'Id du nœud, donc Document.FindNode(key) trouve le nœud.
ParentField	Le champ contenant la clé du parent, ou plusieurs clés séparées par ParentSeparator.
ParentSeparator	Le séparateur de plusieurs clés parentes, un point-virgule par défaut. Plusieurs parents donnent un diagramme de flux au lieu d'un arbre.
LabelField	Le champ affiché comme texte du nœud ; la clé lorsqu'il est vide.
Target	La vue qui affiche le diagramme : tout composant qui implémente IsgcDiagramDocumentOwner, comme TsgcVCLDiagramView et TsgcFMXDiagramView.
Document	Public. Un document à remplir directement ; lorsqu'il est assigné, il est utilisé à la place du document de Target. Il n'est pas possédé : mettez-le à nil avant que le document soit libéré.
Layout	dgbAuto (par défaut) applique la disposition en arbre lorsqu'aucun nœud n'a deux parents, sinon la disposition en couches ; dgbTree et dgbLayered en imposent une ; dgbNone garde les nœuds en lignes, dans l'ordre des lignes.
Direction, TreeStyle	Transmis à la disposition : dglTopToBottom et dgoStandard par défaut.
NodeShape, NodeWidth, NodeHeight	La forme et la taille des nœuds : dgsRoundRect, 140 par 60 par défaut.
ConnectorKind	Le type des connecteurs, dgcOrthogonal par défaut.
LiveUpdates	True (par défaut) reconstruit le diagramme après les changements de données.
Active	True construit le diagramme et commence à suivre les données.

Méthodes et événements

Membre	Signification

Refresh	Reconstruit le diagramme maintenant. Il lève EsgcDiagramError dans une transaction du document.
OnCreateNode(Sender; aNode; aRow)	S'exécute pour chaque nœud après que son texte, sa forme et sa taille sont définis. aRow est la ligne en base 0, et le DataSet est positionné sur elle.
OnCreateConnector(Sender; aConnector; aRow)	S'exécute pour chaque connecteur une fois que chaque nœud existe. aRow est la ligne du nœud enfant ; le DataSet n'est pas positionné sur elle.

Règles

- Une ligne dont la clé est vide, ou dont la clé a déjà été utilisée par une ligne précédente, ne donne aucun nœud. Une clé parente vide, inconnue ou répétée, ou la clé propre de la ligne, ne donne aucun connecteur, donc une ligne dont tous les parents sont inconnus est une racine.
- Une reconstruction vide le document (éléments, calques et images), crée les éléments, applique la disposition et calcule le tracé des connecteurs dans un seul BeginUpdate, si bien que la vue n'est redessinée qu'une fois. Ce n'est pas une étape d'annulation et elle vide les deux piles d'annulation.
- La liaison est à sens unique : elle n'écrit jamais dans les données, et les modifications apportées au diagramme sont remplacées par la prochaine reconstruction.
- Aucune reconstruction n'a lieu à la conception, donc les vues n'enregistrent jamais un diagramme généré dans le fichier de la fiche.
- Voir Liaison de données pour les mises à jour en direct et un exemple complet.

Édition Community

TsgcDiagramDataBinding requiert SGC_DIAGRAM_ADVANCED : dans l'édition Community, l'unité sgcDiagram_Data est vide et le composant n'est pas enregistré.

Exemple

```
uses
  DB,
  sgcDiagram_Types, sgcDiagram_Model, sgcDiagram_Data, sgcDiagram_VCL;

function BindStaff(aView: TsgcVCLDiagramView; aDataSource: TDataSource)
  : TsgcDiagramDataBinding;
begin
  Result := TsgcDiagramDataBinding.Create(aView.Owner);
  Result.KeyField := 'EmpNo';
  Result.ParentField := 'ManagerNo';
  Result.LabelField := 'FullName';
  Result.Layout := dgbTree;
  Result.TreeStyle := dgoCompactLeaves;
  Result.Target := aView;
  Result.DataSource := aDataSource;
  Result.Active := True;
end;
```

Voir aussi

- Liaison de données
- Disposition en arbre
- Disposition en couches
- Éditions

Événements

Les événements des vues viennent de leur contrôleur ; le document et la liaison ont les leurs. Les vues VCL et FMX publient les mêmes événements avec les mêmes paramètres.

Événements de la vue

Événement	Quand
OnChange(Sender; aChanges)	Le document a changé. aChanges est un ensemble de dgkGeometry, dgkStyle, dgkStructure, dgkSelection et dgkView.
OnSelectionChange(Sender)	La sélection a changé.
OnItemClick(Sender; aItem), OnItemDblClick(Sender; aItem)	Un nœud ou un connecteur est cliqué ou double-cliqué.
OnHover(Sender; aItem)	Le pointeur entre dans un élément ou le quitte (aItem vaut nil).
OnConnecting(Sender; aSource, aSourcePort, aTarget, aTargetPort; var aAccept)	Un glissement de connecteur demande avant de coller une extrémité. Un port vide est le contour du nœud, un nœud nil une extrémité libre. Mettez aAccept à False pour refuser.
OnConnected(Sender; aItem)	Un connecteur a été créé ou reconnecté par l'utilisateur.
OnDrawNode(Sender; aCanvas; aNode; aRect; aZoom; aScale; var aDefault)	Avant qu'un nœud soit peint. aRect est le cadre non pivoté en unités monde, et la transformation du canevas convertit les unités monde en pixels de l'appareil avec la rotation du nœud appliquée ; une unité monde vaut aZoom x aScale pixels de l'appareil. Dessinez sur aCanvas, un TsgcUICanvas, et mettez aDefault à False pour ignorer le dessin par défaut de la forme, de l'image et du texte. Les légendes des ports sont peintes dans tous les cas.
OnTextEditing(Sender; aNode; var aAllow)	L'éditeur de texte va s'ouvrir. Mettez aAllow à False pour garder le texte en lecture seule.
OnTextEdited(Sender; aNode)	Le texte d'un nœud a été modifié par l'éditeur.
OnZoomChange(Sender)	Le zoom a changé.

Autres événements

Événement	Quand
TsgcDiagramDocument.OnChange, OnSelectionChange	Les mêmes notifications, pour un document sans vue.
TsgcDiagramDocument.OnDecodeImage(Sender; aKey; aStream; var aImage; var aHandled)	Une image dans un format autre que PNG est chargée : décodez aStream dans aImage et mettez aHandled.
TsgcDiagramDataBinding.OnCreateNode, OnCreateConnector	Voir TsgcDiagramDataBinding.

TsgcVCLDiagramPalette.OnItemClick, TsgcFMXDiagramPalette.OnItemClick	Un élément de la palette est cliqué.

Exemple

```
uses
  sgcDiagram_Types, sgcDiagram_Model, sgcDiagram_VCL;

type
  TDiagramRules = class
  private
    FDirty: Boolean;
    procedure DoConnecting(Sender: TObject; aSource: TsgcDiagramNode;
      const aSourcePort: string; aTarget: TsgcDiagramNode;
      const aTargetPort: string; var aAccept: Boolean);
    procedure DoTextEditing(Sender: TObject; aNode: TsgcDiagramNode;
      var aAllow: Boolean);
    procedure DoChange(Sender: TObject; aChanges: TsgcDiagramChanges);
  public
    procedure Attach(aView: TsgcVCLDiagramView);
    property Dirty: Boolean read FDirty write FDirty;
  end;

procedure TDiagramRules.Attach(aView: TsgcVCLDiagramView);
begin
  aView.OnConnecting := DoConnecting;
  aView.OnTextEditing := DoTextEditing;
  aView.OnChange := DoChange;
end;

procedure TDiagramRules.DoConnecting(Sender: TObject; aSource: TsgcDiagramNode;
  const aSourcePort: string; aTarget: TsgcDiagramNode;
  const aTargetPort: string; var aAccept: Boolean);
begin
  // no loops, and nothing goes back into a terminator
  if aTarget <> nil then
    aAccept := (aTarget <> aSource) and (aTarget.Shape <> dgsTerminator);
end;

procedure TDiagramRules.DoTextEditing(Sender: TObject; aNode: TsgcDiagramNode;
  var aAllow: Boolean);
begin
  aAllow := not aNode.Locked;
end;

procedure TDiagramRules.DoChange(Sender: TObject; aChanges: TsgcDiagramChanges);
begin
  if aChanges * [dgkGeometry, dgkStyle, dgkStructure] <> [] then
    FDirty := True;
end;
```

Voir aussi

- TsgcVCLDiagramView
- TsgcDiagramController
- Ports et collage

TsgcDiagramDocument

TsgcDiagramDocument (unité sgcDiagram_Model) contient un diagramme : ses nœuds, connecteurs, calques et images, la sélection et l'historique d'annulation. Chaque vue affiche un document, publié comme sa propriété Diagram, et un document fonctionne aussi seul, sans vue, dans une application console ou un service.

Propriétés

Propriété	Signification
Nodes, Connectors	Les éléments dans l'ordre Z (le dernier au-dessus).
Layers, Images	Les calques et les images affichées par les nœuds image, par clé (Images.Add, AddPNG, Find, Delete).
Selection	Les éléments sélectionnés : Add, Remove, Toggle, Clear, SelectAll, Contains, Count et Items.
Theme	utLight (par défaut), utDark ou utCustom.
GridSize	Le pas de la grille en unités monde, 10 par défaut.
NodeStyle, ConnectorStyle	Les styles utilisés lorsqu'un élément laisse une valeur à 0.
DefaultConnectorKind	Le type des nouveaux connecteurs, dgcOrthogonal par défaut.
Routing	Les paramètres du routeur : Margin (12), BendPenalty (40), Spacing (6) et AutoRoute (True).
UndoLimit	Le nombre d'étapes d'annulation conservées, 100 par défaut.
InTransaction, Loading	Indicateurs d'état en lecture seule.

Méthodes

Méthode	Effet
AddNode(aShape, aX, aY, aWidth, aHeight, aText)	Ajoute un nœud et le renvoie.
AddConnector(aSource, aTarget, aSourcePort, aTargetPort)	Ajoute un connecteur entre deux nœuds ; un port vide colle l'extrémité au contour.
FindNode(aId), FindConnector(aId)	Trouve un élément par son Id, nil s'il est introuvable.
ConnectorsOf(aNode, aList), ChildrenOf(aContainer, aList)	Remplit une TList avec les connecteurs d'un nœud ou les enfants d'un conteneur.
ItemsInRect(aRect, aList, aFullyInside)	Remplit une TList avec les éléments qui touchent un rectangle ou se trouvent à l'intérieur.
ContentBounds	Les limites de tous les éléments visibles.

DeleteSelection, Clear	Suppriment les éléments sélectionnés ou tout.
Group, Ungroup, BringToFront, SendToBack	Agissent sur la sélection. Group renvoie le nouvel Id de groupe.
BeginUpdate, EndUpdate	Un seul redessin et une seule notification de modification pour de nombreux changements.
BeginTransaction(aCaption), EndTransaction, Rollback	Plusieurs modifications en une seule étape d'annulation, ou aucune.
BeginUndoSuspend, EndUndoSuspend	Les modifications intermédiaires ne sont pas enregistrées et vident les deux piles d'annulation ; pour les chargements en masse.
Undo, Redo, CanUndo, CanRedo, UndoCaption, RedoCaption	L'historique d'annulation.
RouteAll	Recalcule le tracé de chaque connecteur.
SaveToJSON, LoadFromJSON, SaveToFile, LoadFromFile, SaveToStream, LoadFromStream	Le format JSON.
CopySelection, Paste(aJSON)	Un fragment du document en JSON et son collage.
ResolveNodeStyle(aNode), ResolveConnectorStyle(aConnector)	Le style avec lequel un élément est dessiné, couleurs du thème résolues.
LayerVisible(aName), LayerLocked(aName)	La façon dont un calque est traité.

Événements

OnChange(Sender; aChanges), OnSelectionChange et OnDecodeImage, voir Événements.

Threads

Un document n'est modifié que dans le thread qui le possède, le thread principal lorsqu'il est affiché dans une vue.

Exemple

```
uses
  Classes, SysUtils,
  sgcDiagram_Types, sgcDiagram_Model;

procedure ImportGrid(aDocument: TsgcDiagramDocument; aCount: Integer);
var
  i: Integer;
begin
  // a bulk load: no undo history, one repaint
  aDocument.BeginUndoSuspend;
  aDocument.BeginUpdate;
  try
    for i := 0 to aCount - 1 do
      aDocument.AddNode(dgsRectangle, (i mod 10) * 160, (i div 10) * 100,
        120, 60, 'Item ' + IntToStr(i + 1));
  finally
    aDocument.EndUpdate;
    aDocument.EndUndoSuspend;
  end;
end;
```

```
function CountIncoming(aDocument: TsgcDiagramDocument;
  const aId: string): Integer;
var
  oNode: TsgcDiagramNode;
  oList: TList;
  i: Integer;
begin
  Result := 0;
  oNode := aDocument.FindNode(aId);
  if oNode = nil then
    Exit;
  oList := TList.Create;
  try
    aDocument.ConnectorsOf(oNode, oList);
    for i := 0 to oList.Count - 1 do
      if TsgcDiagramConnector(oList[i]).TargetNodeId = oNode.Id then
        Inc(Result);
  finally
    oList.Free;
  end;
end;
```

Voir aussi

- TsgcDiagramNode
- TsgcDiagramConnector
- Annuler et presse-papiers
- Architecture

TsgcDiagramNode

Un nœud est une forme avec un texte : une étape de processus, une décision, une personne d'un organigramme ou un conteneur d'autres nœuds. Les nœuds sont créés avec Document.AddNode et résident dans Document.Nodes.

Propriétés

Propriété	Signification
Shape	La forme, dgsRectangle par défaut.
Left, Top, Width, Height	Le cadre en unités monde (120 par 60 par défaut).
Angle	La rotation en degrés, dans le sens horaire, autour du centre.
Text, TextAlign, TextVAlign	Le texte et son alignement (centré par défaut).
Ports	Les ports ; vide, les ports par défaut de la forme sont utilisés.
ContainerId	L'Id du nœud conteneur qui contient ce nœud.
GroupId	Le groupe du nœud ; les nœuds d'un groupe sont sélectionnés et déplacés ensemble.
ImageKey	La clé dans Document.Images de l'image dessinée par un nœud dgsImage.
PathData	Le contour d'un nœud dgsPath, en syntaxe de chemin SVG dans un cadre 0..100.
Visible	False masque le nœud.
Id	L'Id unique de l'élément dans le document ; AddNode et AddConnector en génèrent un.
Style	Le style de l'élément ; les valeurs à 0 viennent du style du document.
Layer	Le nom du calque ; vide pour le calque par défaut.
Locked	True empêche de sélectionner et de modifier l'élément.
Data, Tag	Données d'application : une chaîne (enregistrée dans le fichier) et un entier.
Selected, Document	Public, en lecture seule.

Méthodes

Méthode	Effet
SetBounds(aLeft, aTop, aWidth, aHeight), MoveBy(aDX, aDY)	Déplacent ou redimensionnent le nœud en une seule modification.
BoundsRect, Center	Le cadre et son centre.

PortPoint(aPortId)	La position monde d'un port.

Exemple

```
uses
  SysUtils,
  sgcUICore_Canvas, sgcDiagram_Types, sgcDiagram_Model;

function AddOrderNode(aDocument: TsgcDiagramDocument;
  aOrderNo: Integer): TsgcDiagramNode;
begin
  Result := aDocument.AddNode(dgsDocument, 40, 40, 160, 80, 'Order');
  Result.TextAlign := thaLeft;
  Result.Tag := aOrderNo;
  Result.Data := '{"order":' + IntToStr(aOrderNo) + '}';
  Result.Angle := -5;
end;

procedure PutInContainer(aContainer, aNode: TsgcDiagramNode);
begin
  aNode.ContainerId := aContainer.Id;
  aNode.SetBounds(aContainer.Left + 20, aContainer.Top + 40, aNode.Width,
    aNode.Height);
end;
```

Voir aussi

- TsgcDiagramPort
- TsgcDiagramConnector
- Formes

TsgcDiagramConnector

Un connecteur relie deux nœuds, ou un nœud et un point libre. Son tracé est calculé par le routeur à partir de son type. Les connecteurs sont créés avec Document.AddConnector et résident dans Document.Connectors.

Propriétés

Propriété	Signification
Kind	dgcStraight, dgcOrthogonal ou dgcCurved.
SourceNodeId, SourcePortId, Target-NodeId, TargetPortId	Les extrémités collées. Un port vide colle l'extrémité au contour du nœud.
SourcePoint, TargetPoint	Les extrémités libres, utilisées lorsque l'Id du nœud est vide.
Waypoints	Les points par lesquels passe un connecteur droit ou courbe.
StartArrow, EndArrow, ArrowSize	Les pointes de flèche (dgaNone et dgaTriangle par défaut) et leur taille (10).
Labels	Les étiquettes : Text, Position le long du chemin de 0 à 1 (0.5 par défaut) et Offset par rapport au chemin.
Points	Public, en lecture seule. Le tracé actuel.
RouteFailed	Public, en lecture seule. True lorsque le routeur n'a trouvé aucun chemin autour des nœuds.
Id	L'Id unique de l'élément dans le document ; AddNode et AddConnector en génèrent un.
Style	Le style de l'élément ; les valeurs à 0 viennent du style du document.
Layer	Le nom du calque ; vide pour le calque par défaut.
Locked	True empêche de sélectionner et de modifier l'élément.
Data, Tag	Données d'application : une chaîne (enregistrée dans le fichier) et un entier.
Selected, Document	Public, en lecture seule.

Méthodes

Méthode	Effet
SetRoute(aPoints, aFailed)	Définit le tracé ; utilisé par le routeur et les dispositions.

Édition Community

Sans SGC_DIAGRAM_ADVANCED, un connecteur dgcOrthogonal est routé en coude : pas d'évitement d'obstacles et pas d'écartement, et RouteFailed reste False.

Exemple

```
uses
  sgcDiagram_Types, sgcDiagram_Model;

procedure ConnectWithLabel(aDocument: TsgcDiagramDocument;
  aFrom, aTo: TsgcDiagramNode; const aText: string);
var
  oConnector: TsgcDiagramConnector;
begin
  oConnector := aDocument.AddConnector(aFrom, aTo);
  oConnector.Kind := dgcCurved;
  oConnector.StartArrow := dgaCircle;
  oConnector.EndArrow := dgaOpenTriangle;
  oConnector.Style.Dash := dgdDash;
  with oConnector.Labels.Add do
  begin
    Text := aText;
    Position := 0.25;
    Offset := -12;
  end;
end;
```

Voir aussi

- Connecteurs et routage
- TsgcDiagramPort
- TsgcDiagramNode

TsgcDiagramPort

Un port est un point nommé d'un nœud auquel les extrémités des connecteurs sont collées. Un nœud dont la collection Ports est vide utilise les ports par défaut de sa forme : les milieux des côtés, nommés left, top, right et bottom.

Propriétés

Propriété	Signification
Id	Le nom utilisé par SourcePortId et TargetPortId.
X, Y	La position en fraction du cadre du nœud, de 0 à 1 (0.5 par défaut).
Side	Le côté d'où part un connecteur : dgpAuto (par défaut), dgpLeft, dgpTop, dgpRight ou dgpBottom.
Flow	dgfBoth (par défaut), dgfIn ou dgfOut : quelles extrémités peuvent se coller au port.
Caption	Un texte dessiné à côté du port.

Node.Ports.Add ajoute un port et Node.Ports.Find(aId) en trouve un. Node.PortPoint(aId) renvoie sa position monde.

Exemple

```
uses
  sgcDiagram_Types, sgcDiagram_Model;

procedure AddTypedPorts(aNode: TsgcDiagramNode);
begin
  with aNode.Ports.Add do
  begin
    Id := 'in';
    X := 0;
    Y := 0.5;
    Side := dgpLeft;
    Flow := dgfIn;
    Caption := 'Input';
  end;
  with aNode.Ports.Add do
  begin
    Id := 'out';
    X := 1;
    Y := 0.5;
    Side := dgpRight;
    Flow := dgfOut;
    Caption := 'Output';
  end;
end;
```

Voir aussi

- Ports et collage
- TsgcDiagramNode

TsgcDiagramStyle

Chaque nœud et chaque connecteur a un Style, et le document a un NodeStyle et un ConnectorStyle. Une couleur ou une taille laissée à 0 vient du style du document, et une couleur encore à 0 à ce niveau vient du thème, de sorte qu'un diagramme suit un changement de thème tant que ses couleurs ne sont pas définies.

Propriétés

Propriété	Signification
Fill	La couleur de remplissage, un TsgcUIColor (en construire un avec sgcUIARGB).
Stroke	La couleur du contour ou de la ligne.
StrokeWidth	L'épaisseur du contour ou de la ligne en unités monde, 1 par défaut.
Dash	dgdSolid (par défaut), dgdDash ou dgdDot.
CornerRadius	Le rayon des coins des rectangles et des rectangles arrondis.
Text	Le style du texte : Size, Bold et Color.

Document.ResolveNodeStyle(aNode) et ResolveConnectorStyle(aConnector) renvoient les valeurs avec lesquelles un élément est dessiné.

Exemple

```
uses
  sgcUICore_Theme, sgcDiagram_Types, sgcDiagram_Model;

procedure ApplyHouseStyle(aDocument: TsgcDiagramDocument);
begin
  aDocument.NodeStyle.Fill := sgcUIARGB($FF, $E3, $F2, $FD);
  aDocument.NodeStyle.Stroke := sgcUIARGB($FF, $15, $65, $C0);
  aDocument.NodeStyle.CornerRadius := 6;
  aDocument.NodeStyle.Text.Bold := True;
  aDocument.ConnectorStyle.StrokeWidth := 2;
end;

procedure MarkAsError(aNode: TsgcDiagramNode);
begin
  aNode.Style.Fill := sgcUIARGB($FF, $E5, $39, $35);
  aNode.Style.Text.Color := sgcUIARGB($FF, $FF, $FF, $FF);
end;
```

Voir aussi

- Thèmes
- TsgcDiagramDocument

TsgcDiagramLayer

Un calque regroupe des éléments par nom. Document.Layers contient les calques ; chaque nœud et chaque connecteur nomme son calque dans sa propriété Layer, et un nom vide désigne le calque par défaut.

Propriétés

Propriété	Signification
Name	Le nom utilisé par la propriété Layer des éléments.
Visible	False masque les éléments du calque dans la vue et dans l'export.
Locked	True empêche de sélectionner et de modifier les éléments du calque.

Layers.Add ajoute un calque et Layers.Find(aName) en trouve un. Ajouter, supprimer et renommer un calque sont des étapes d'annulation ; Visible et Locked n'en sont pas.

Édition Community

Sans SGC_DIAGRAM_ADVANCED, un second Layers.Add lève EsgcDiagramError. Un fichier avec plus de calques se charge et s'enregistre sans changement, mais chaque calque est dessiné et modifié comme visible et non verrouillé.

Exemple

```
uses
  sgcDiagram_Types, sgcDiagram_Model;

procedure ToggleLayer(aDocument: TsgcDiagramDocument; const aName: string);
var
  oLayer: TsgcDiagramLayer;
begin
  oLayer := aDocument.Layers.Find(aName);
  if oLayer <> nil then
    oLayer.Visible := not oLayer.Visible;
end;
```

Voir aussi

- Calques
- Éditions

TsgcDiagramController

TsgcDiagramController (unité sgcDiagram_Controller) est le moteur d'édition des vues, neutre vis-à-vis du framework : il transforme les entrées du pointeur, du clavier, de la molette et du glisser-déposer en modifications du document, conserve le zoom et la position de défilement, et peint via un TsgcDiagramPainter. Chaque vue en crée un et l'expose comme propriété Controller. Les positions sont en pixels de l'appareil ; le contrôleur travaille en unités monde à l'intérieur.

Propriétés

Propriété	Signification
Document, Host, Painter	Le document, le contrôle hôte (IsgcDiagramHost) et le peintre.
Tool, ShapeTool, ShapeWidth, ShapeHeight	L'outil actif, ainsi que la forme et la taille par défaut que l'outil de forme dessine.
ReadOnly, ShowGrid, SnapToGrid, SnapToGuides, ShowPorts	Les options publiées par les vues.
Zoom, MinZoom, MaxZoom, OffsetX, OffsetY, Viewport	La transformation de la vue.
State	L'interaction en cours : dgiIdle, dgiMoving, dgiResizing, etc.
HoverItem, EditingNode	L'élément sous le pointeur et le nœud dont le texte est en cours de modification.

Méthodes

Méthode	Effet
ZoomAt(aZoom, aX, aY), ScrollBy(aDX, aDY)	Zoomer autour d'un point de l'appareil, défiler en pixels de l'appareil.
ZoomToFit, ZoomToSelection, CenterOn	Comme dans les vues.
SelectAll, Duplicate, Nudge(aDX, aDY)	Sélectionner tout, dupliquer la sélection, la déplacer en unités monde.
GroupSelection, UngroupSelection	Grouper ou dissocier les nœuds sélectionnés.
Undo, Redo, CopyToClipboard, CutToClipboard, PasteFromClipboard, DeleteSelection	Les commandes de modification.
EditText(aNode), EndTextEdit	Ouvrir et fermer l'éditeur de texte.
Cancel	Annule l'interaction en cours, comme le fait Esc.

Les événements du contrôleur sont publiés par les vues, voir Événements. Le clavier est décrit dans Édition et clavier.

Exemple

```
uses
  sgcDiagram_Types, sgcDiagram_VCL;

procedure DuplicateAndShift(aView: TsgcVCLDiagramView);
begin
  if aView.Diagram.Selection.Count = 0 then
    Exit;
  aView.Controller.Duplicate;
  aView.Controller.Nudge(aView.Diagram.GridSize, aView.Diagram.GridSize);
end;

procedure SelectAndGroup(aView: TsgcVCLDiagramView);
begin
  aView.Controller.SelectAll;
  aView.Controller.GroupSelection;
end;
```

Voir aussi

- TsgcVCLDiagramView
- Édition et clavier

TsgcDiagramRenderer

TsgcDiagramRenderer (unité sgcDiagram_Render) dessine un document sans vue : vers des fichiers ou des flux PNG et SVG, ou vers n'importe quel TsgcUICanvas. Les vues l'utilisent pour SaveToPNG et SaveToSVG, et il fonctionne dans les applications console, les services et sous Linux.

Méthodes

Méthode	Effet
SaveToPNGFile(aDocument, aFileName, aScale, aMargin), SaveToPNGStream	Un PNG de tout le diagramme. aScale vaut 1 par défaut, aMargin 20 unités monde.
SaveToSVGFile(aDocument, aFileName, aScale, aMargin), SaveToSVGStream	La même chose en SVG.
ExportViewport(aDocument, aScale, aMargin)	Le viewport utilisé par un export : décalage, zoom, échelle et taille d'image.
RenderTo(aCanvas, aDocument, aViewport)	Dessine le document sur un canevas à travers un viewport.
Painter	Le TsgcDiagramPainter qui dessine ; son événement OnDrawNode personnalise aussi les nœuds dans l'export.

Exemple

```
uses
  sgcDiagram_Model, sgcDiagram_Render;

procedure ExportDiagram(aDocument: TsgcDiagramDocument);
var
  oRenderer: TsgcDiagramRenderer;
begin
  oRenderer := TsgcDiagramRenderer.Create;
  try
    oRenderer.SaveToPNGFile(aDocument, 'diagram.png');
    oRenderer.SaveToPNGFile(aDocument, 'diagram@2x.png', 2);
    oRenderer.SaveToSVGFile(aDocument, 'diagram.svg', 1, 40);
  finally
    oRenderer.Free;
  end;
end;
```

Voir aussi

- Export
- Démarrage rapide Headless

Format JSON

SaveToJSON écrit un seul objet JSON. La sortie est canonique : un ordre de clés fixe, aucune indentation, et chaque valeur par défaut est omise, de sorte qu'une clé absente signifie toujours la valeur par défaut. Les énumérations sont écrites sous forme de noms, les couleurs sous forme de chaînes hexadécimales, les points sous forme de tableaux [x, y]. Le lecteur ignore les clés inconnues et les noms inconnus.

Document

Clé	Valeur
`format`	Toujours "sgcDiagram".
`version`	La version du format, 1. Un fichier d'une version plus récente lève EsgcDiagramError.
`gridSize`	GridSize (10).
`theme`	"light" (par défaut), "dark" ou "custom".
`layers`	Tableau de { name, visible (true), locked (false) }.
`nodes`	Tableau des nœuds, dans l'ordre Z.
`connectors`	Tableau des connecteurs, dans l'ordre Z.
`images`	Tableau de { key, png }, les données PNG en Base64.

NodeStyle, ConnectorStyle, DefaultConnectorKind, Routing et UndoLimit sont des propriétés de l'objet document et ne sont pas écrites dans le fichier.

Nœud

Clé	Propriété
`id`	Id
`shape`	Shape : "rectangle" (par défaut), "roundRect", "ellipse", "text", "image", "path", "process", "decision", "terminator", "data", "document", "predefinedProcess", "connectorCircle", "container"
`x` , `y` , `w` , `h`	Left, Top, Width (120), Height (60)
`angle`	Angle
`text`	Text
`textAlign` , `textVAlign`	TextAlign ("center"), TextVAlign ("middle")
`layer` , `container` , `group`	Layer, ContainerId, GroupId
`image` , `path`	ImageKey, PathData

`locked`, `visible`	Locked (false), Visible (true)
`data`, `tag`	Data, Tag
`style`	Style, voir ci-dessous
`ports`	Tableau de { id, x (0.5), y (0.5), side ("auto"), flow ("both"), caption }

Connecteur

Clé	Propriété
`id`	Id
`kind`	Kind : "straight", "orthogonal" (par défaut) ou "curved"
`source`, `target`	{ node, port } : SourceNodeId et SourcePortId, TargetNodeId et TargetPortId
`sourcePoint`, `targetPoint`	SourcePoint, TargetPoint des extrémités libres
`waypoints`	Waypoints
`startArrow`, `endArrow`	StartArrow ("none"), EndArrow ("triangle"): "none", "triangle", "openTriangle", "diamond", "openDiamond", "circle"
`arrowSize`	ArrowSize (10)
`labels`	Tableau de { text, pos (0.5), offset (0) }
`layer`, `locked`, `data`, `tag`	Layer, Locked, Data, Tag
`style`	Style, voir ci-dessous
`points`	Le tracé enregistré, conservé au chargement

Style

Clé	Propriété
`fill`, `stroke`	Fill, Stroke
`strokeWidth`	StrokeWidth (1)
`dash`	Dash : "solid" (par défaut), "dash" ou "dot"
`radius`	CornerRadius
`text`	{ size, bold, color } : le style du texte

Un petit fichier complet est montré dans Sérialisation.

Exemple

```
uses
  SysUtils,
  sgcDiagram_Types, sgcDiagram_Model;

function DiagramFromJSON(const aJSON: string): TsgcDiagramDocument;
begin
  Result := TsgcDiagramDocument.Create;
  try
    Result.LoadFromJSON(aJSON);
  except
    on E: EsgcDiagramError do
    begin
      Result.Free;
      raise Exception.Create('Not a valid diagram: ' + E.Message);
    end;
  end;
end;
```

Voir aussi

- Sérialisation
- TsgcDiagramDocument

Disposition en couches

TsgcDiagramLayeredLayout (unité sgcDiagram_Layout) organise un graphe orienté en couches, selon la méthode de Sugiyama : diagrammes de flux, diagrammes de processus, machines à états et graphes de dépendances. Les connecteurs vont d'une couche à la suivante, et la disposition limite leurs croisements.

Propriétés

Propriété	Signification
Direction	dglTopToBottom (par défaut), dglLeftToRight, dglBottomToTop ou dglRightToLeft.
LayerSpacing	L'écart entre les couches, 60 unités monde par défaut, ajouté au nœud le plus profond de la couche.
NodeSpacing	L'écart entre nœuds voisins, 30 par défaut.
SelectionOnly	Ne dispose que les nœuds sélectionnés.
Crossings	En lecture seule. Les croisements de connecteurs comptés après Execute.

Fonctionnement

- Les cycles sont rompus en inversant les arêtes de retour d'un parcours en profondeur dans l'ordre de la collection.
- Chaque nœud reçoit la couche de son plus long chemin depuis une source ; un long connecteur reçoit un nœud factice sur chaque couche qu'il traverse.
- Des balayages barycentriques réduisent les croisements et le meilleur ordre est conservé.
- Les coordonnées sont placées vers la médiane des voisins, les longs connecteurs droits, sans chevauchements.
- Les positions des nœuds factices deviennent les waypoints des connecteurs droits et courbes ; les connecteurs orthogonaux sont routés par le routeur.
- Les parties non connectées sont disposées côte à côte, espacées de NodeSpacing.

Remarques

- La disposition travaille sur les nœuds de niveau supérieur, ou sur les nœuds sélectionnés lorsque SelectionOnly vaut True (par exemple les enfants d'un conteneur). Un conteneur compte comme un seul nœud de sa taille et porte ses enfants.
- Les nœuds gardent le coin supérieur gauche de leurs limites. Le résultat est appliqué en une seule transaction, donc une disposition est une seule étape d'annulation, et les connecteurs déplacés voient leur tracé recalculé, même lorsque Routing.AutoRoute vaut False.
- Chaque coordonnée est calculée en centièmes entiers et chaque tri est stable, si bien qu'une disposition donne le même résultat sous Win32, Win64 et Linux64.

Édition Community

Les dispositions requièrent SGC_DIAGRAM_ADVANCED : dans l'édition Community, l'unité sgcDiagram_Layout est vide.

Exemple

```
uses
  sgcDiagram_Types, sgcDiagram_Model, sgcDiagram_Layout;

function ArrangeFlowchart(aDocument: TsgcDiagramDocument): Integer;
var
  oLayout: TsgcDiagramLayeredLayout;
begin
  oLayout := TsgcDiagramLayeredLayout.Create(aDocument);
  try
    oLayout.Direction := dglLeftToRight;
    oLayout.LayerSpacing := 80;
    oLayout.NodeSpacing := 40;
    oLayout.Execute;
    Result := oLayout.Crossings;
  finally
    oLayout.Free;
  end;
end;
```

Voir aussi

- Disposition en arbre
- TsgcDiagramDataBinding
- Éditions

Disposition en arbre

TsgcDiagramTreeLayout (unité sgcDiagram_Layout) organise les hiérarchies : organigrammes, arbres de catégories et cartes mentales. Les parents sont centrés au-dessus de leurs enfants et les sous-arbres voisins sont tassés à une distance NodeSpacing, en temps linéaire.

Propriétés

Propriété	Signification
Direction	dglTopToBottom (par défaut), dglLeftToRight, dglBottomToTop ou dglRightToLeft.
LayerSpacing	L'écart entre les niveaux, 60 unités monde par défaut, ajouté au nœud le plus profond du niveau.
NodeSpacing	L'écart entre sous-arbres voisins et entre feuilles empilées, 30 par défaut.
TreeStyle	dgoStandard (par défaut) ou dgoCompactLeaves.
SelectionOnly	Ne dispose que les nœuds sélectionnés.

Fonctionnement

- Les racines sont les nœuds sans connecteur entrant, dans l'ordre de la collection. Un connecteur va du parent à l'enfant ; un connecteur qui n'est pas une arête de l'arbre est dessiné mais ignoré pour le placement.
- Les enfants d'un nœud gardent leur ordre dans la collection.
- dgoCompactLeaves empile les enfants d'un nœud dans une colonne à droite du centre du parent lorsqu'il a deux enfants ou plus et que tous sont des feuilles, l'organigramme classique. Les connecteurs droits et courbes reçoivent un waypoint en coude sous le parent.
- Plusieurs arbres sont placés côte à côte dans l'ordre des racines, espacés de NodeSpacing.

Remarques

- La disposition travaille sur les nœuds de niveau supérieur, ou sur les nœuds sélectionnés lorsque SelectionOnly vaut True (par exemple les enfants d'un conteneur). Un conteneur compte comme un seul nœud de sa taille et porte ses enfants.
- Les nœuds gardent le coin supérieur gauche de leurs limites. Le résultat est appliqué en une seule transaction, donc une disposition est une seule étape d'annulation, et les connecteurs déplacés voient leur tracé recalculé, même lorsque Routing.AutoRoute vaut False.
- Chaque coordonnée est calculée en centièmes entiers et chaque tri est stable, si bien qu'une disposition donne le même résultat sous Win32, Win64 et Linux64.

Édition Community

Les dispositions requièrent SGC_DIAGRAM_ADVANCED : dans l'édition Community, l'unité sgcDiagram_Layout est vide.

Exemple

```
uses
  sgcDiagram_Types, sgcDiagram_Model, sgcDiagram_Layout;

procedure ArrangeOrgChart(aDocument: TsgcDiagramDocument; aCompact: Boolean);
var
  oLayout: TsgcDiagramTreeLayout;
begin
  oLayout := TsgcDiagramTreeLayout.Create(aDocument);
  try
    if aCompact then
      oLayout.TreeStyle := dgoCompactLeaves;
    oLayout.NodeSpacing := 20;
    oLayout.Execute;
  finally
    oLayout.Free;
  end;
end;
```

Voir aussi

- Disposition en couches
- Liaison de données

Liaison de données

Une TsgcDiagramDataBinding transforme les lignes d'un jeu de données en diagramme et le tient à jour. Cette page montre la configuration habituelle d'un organigramme pas à pas.

Étapes

1. Donnez aux données un champ clé et un champ parent : chaque ligne est un nœud, et la clé parente donne le connecteur depuis le parent. Les lignes sans parent connu sont des racines.
2. Déposez une TsgcDiagramDataBinding sur la fiche, définissez DataSource, KeyField, ParentField et LabelField, et mettez Target sur la vue du diagramme.
3. Choisissez la disposition : dgbAuto choisit la disposition en arbre pour un arbre et la disposition en couches lorsqu'une ligne a plusieurs parents (clés séparées par ParentSeparator). Définissez Direction et TreeStyle selon les besoins.
4. Stylez les nœuds dans OnCreateNode : le DataSet est positionné sur la ligne du nœud, donc n'importe quel champ peut définir le texte, les couleurs ou le Data du nœud.
5. Mettez Active à True. Le diagramme est construit immédiatement et reconstruit lorsque les données changent.

Mises à jour en direct

- Avec LiveUpdates à True, l'ouverture du DataSet, une modification validée, une insertion ou une suppression, un changement du nombre de lignes ou du filtre, ou une notification de la table reconstruit le diagramme. Les demandes sont fusionnées en une seule reconstruction par tour de la boucle de messages.
- La navigation ne reconstruit pas, donc la sélection est conservée. Un rafraîchissement qui ne change que des lignes autres que la ligne courante, et pas le nombre de lignes, n'est pas détecté : appelez Refresh ensuite.
- Une reconstruction attend tant que le DataSet est en cours de modification et tant que l'utilisateur fait glisser dans la vue. La fermeture du DataSet conserve le diagramme.
- À partir de Delphi 10.2, la reconstruction est mise en file avec TThread.ForceQueue : les applications console et les services doivent appeler CheckSynchronize.
- Une reconstruction vide l'historique d'annulation : les modifications faites dans le diagramme ne sont pas réécrites dans les données.

Édition Community

La liaison de données requiert SGC_DIAGRAM_ADVANCED. Dans l'édition Community, construisez le diagramme par code avec AddNode et AddConnector.

Exemple

```
uses
  DB, SysUtils,
  sgcUICore_Theme, sgcDiagram_Types, sgcDiagram_Model, sgcDiagram_Data,
  sgcDiagram_VCL;

type
  TOrgChart = class
  private
    FBinding: TsgcDiagramDataBinding;
    procedure DoCreateNode(Sender: TObject; aNode: TsgcDiagramNode;
      aRow: Integer);
  public
    constructor Create(aView: TsgcVCLDiagramView; aDataSource: TDataSource);
    destructor Destroy; override;
  end;
```

```
constructor TOrgChart.Create(aView: TsgcVCLDiagramView;
  aDataSource: TDataSource);
begin
  inherited Create;
  FBinding := TsgcDiagramDataBinding.Create(nil);
  FBinding.KeyField := 'EmpNo';
  FBinding.ParentField := 'ManagerNo';
  FBinding.LabelField := 'FullName';
  FBinding.Layout := dgbTree;
  FBinding.TreeStyle := dgoCompactLeaves;
  FBinding.NodeWidth := 180;
  FBinding.OnCreateNode := DoCreateNode;
  FBinding.Target := aView;
  FBinding.DataSource := aDataSource;
  FBinding.Active := True;
end;

destructor TOrgChart.Destroy;
begin
  FBinding.Free;
  inherited;
end;

procedure TOrgChart.DoCreateNode(Sender: TObject; aNode: TsgcDiagramNode;
  aRow: Integer);
var
  oDataSet: TDataSet;
begin
  oDataSet := FBinding.DataSource.DataSet;
  aNode.Text := oDataSet.FieldByName('FullName').AsString + #13#10 +
    oDataSet.FieldByName('JobTitle').AsString;
  if SameText(oDataSet.FieldByName('Dept').AsString, 'Sales') then
    aNode.Style.Fill := sgcUIARGB($FF, $C8, $E6, $C9);
end;
```

Voir aussi

- TsgcDiagramDataBinding
- Disposition en arbre
- Disposition en couches

Formes

Chaque nœud a une Shape de type TsgcDiagramShape. La forme détermine le contour, la zone qui reçoit le texte, les ports par défaut et les couleurs du thème. Un nœud est placé avec Left, Top, Width et Height en unités monde et pivoté avec Angle, en degrés dans le sens horaire autour de son centre.

Forme	Usage
dgsRectangle, dgsRoundRect, dgsEllipse	Des cadres simples. dgsRoundRect utilise un rayon de coin de 8 lorsque Style.CornerRadius vaut 0 ; les autres cadres n'arrondissent leurs coins que lorsque CornerRadius est supérieur à 0.
dgsProcess, dgsDecision, dgsTerminator, dgsData, dgsDocument, dgsPredefinedProcess, dgsConnectorCircle	Les symboles de diagramme de flux. Un terminateur est toujours arrondi de la moitié de son plus petit côté ; une décision est un losange dont les ports par défaut sont ses coins.
dgsText	Texte libre sans remplissage ni bordure.
dgsImage	Une image du document, nommée par ImageKey. Voir ci-dessous.
dgsPath	Un contour personnalisé issu de PathData, en syntaxe de chemin SVG dans un cadre de 0 à 100 étiré au nœud.
dgsContainer	Un cadre avec une bande d'en-tête qui contient d'autres nœuds. Un nœud lui appartient par ContainerId, et déplacer le conteneur déplace ses enfants.

Texte

Le texte est dessiné dans le rectangle de texte de la forme, avec retour à la ligne, avec TextAlign (thaCenter par défaut) et TextVAlign (tvaMiddle par défaut). La taille, la graisse et la couleur de la police viennent de Style.Text. Dans la vue, un double-clic, F2 ou un caractère tapé modifie le texte sur place.

Images

Images.AddPNG(aKey, aStream) stocke les octets PNG d'un flux et les décode ; Images.Add(aKey, aImage) ajoute une TsgcUIImage décodée. Un nœud avec Shape dgsImage et ImageKey défini sur la clé dessine l'image étirée à son cadre. Les autres formats sont décodés par votre gestionnaire de l'événement OnDecodeImage du document. Supprimer une image qu'un nœud utilise encore lève EsgcDiagramError.

Groupes et conteneurs

Document.Group place les nœuds sélectionnés dans un groupe et renvoie son GroupId ; un clic sélectionne alors tout le groupe, et Alt avec un clic sélectionne un membre. Ungroup supprime le groupe de la sélection. Un conteneur possède ses enfants : ils se déplacent avec lui, un conteneur supprimé les supprime, et un nœud déposé sur un conteneur dans la vue devient son enfant.

Exemple

```
uses
  Classes,
  sgcDiagram_Types, sgcDiagram_Model;
```

```
procedure AddShapes(aDocument: TsgcDiagramDocument; aLogo: TStream);
var
  oBox, oStar, oLogo, oChild: TsgcDiagramNode;
begin
  oBox := aDocument.AddNode(dgsContainer, 0, 0, 400, 260, 'Warehouse');
  oChild := aDocument.AddNode(dgsProcess, 20, 60, 140, 60, 'Pick');
  oChild.ContainerId := oBox.Id;
  // a five pointed star in the 0..100 path box
  oStar := aDocument.AddNode(dgsPath, 220, 60, 80, 80, '');
  oStar.PathData := 'M 50 0 L 61 35 L 98 35 L 68 57 L 79 91 L 50 70 L 21 91 L 32 57 L 2 35 L 39 35 Z';
  oStar.Angle := 15;
  // an image node
  aDocument.Images.AddPNG('logo', aLogo);
  oLogo := aDocument.AddNode(dgsImage, 420, 0, 96, 96, '');
  oLogo.ImageKey := 'logo';
end;
```

Voir aussi

- Ports et collage
- Thèmes
- Connecteurs et routage
- TsgcDiagramNode

Connecteurs et routage

Un TsgcDiagramConnector relie une source et une cible. Chaque extrémité est soit collée à un nœud, via SourceNodeId et TargetNodeId, soit libre à SourcePoint et TargetPoint. Document.AddConnector(aSource, aTarget) crée un connecteur du type DefaultConnectorKind (dgcOrthogonal par défaut) et calcule son tracé à la fin du lot de modifications.

Type	Tracé
dgcStraight	Une ligne droite, ou une polyligne passant par les Waypoints.
dgcOrthogonal	Des segments horizontaux et verticaux. Dans l'édition Registered, le routeur évite les nœuds avec une marge et écarte les segments parallèles ; dans Community, le tracé est un simple coude.
dgcCurved	Une courbe lisse, passant par les Waypoints s'il y en a.

Flèches, ligne et étiquettes

- StartArrow (dgaNone par défaut) et EndArrow (dgaTriangle par défaut) acceptent dgaNone, dgaTriangle, dgaOpenTriangle, dgaDiamond, dgaOpenDiamond ou dgaCircle. ArrowSize définit leur taille.
- Style.Stroke, Style.StrokeWidth et Style.Dash (dgdSolid, dgdDash, dgdDot) définissent la ligne.
- Labels contient un nombre quelconque de TsgcDiagramLabel. Position va de 0 à la source jusqu'à 1 à la cible, Offset éloigne l'étiquette de la ligne. L'utilisateur fait glisser une étiquette le long de son connecteur.

Routage

Le document calcule le tracé de chaque connecteur dont les nœuds, les extrémités, le type ou les waypoints ont changé lorsque le EndUpdate le plus externe s'exécute, tant que Routing.AutoRoute vaut True. RouteAll recalcule chaque connecteur. Le tracé se trouve dans Points ; RouteFailed vaut True lorsque le routeur orthogonal n'a trouvé aucun chemin autour des nœuds et a dessiné un tracé de secours.

Propriété de routage	Défaut	Signification
Margin	12	Distance qu'un tracé orthogonal garde avec les nœuds.
BendPenalty	40	Coût d'un coude, de sorte que le routeur préfère moins de coudes à un chemin légèrement plus court.
Spacing	6	Distance entre des segments parallèles qui partagent un couloir.
AutoRoute	True	Calcule le tracé à la fin de chaque lot de modifications. Désactivez-le pendant que vous construisez un grand diagramme par code, puis appelez RouteAll.

Exemple

```
uses
  sgcDiagram_Types, sgcDiagram_Model;

procedure AddConnectors(aDocument: TsgcDiagramDocument; aA, aB, aC: TsgcDiagramNode);
var
```

```
  oLink: TsgcDiagramConnector;
begin
  aDocument.Routing.Margin := 16;
  aDocument.BeginUpdate;
  try
    oLink := aDocument.AddConnector(aA, aB);
    oLink.Labels.Add.Text := 'Yes';
    oLink := aDocument.AddConnector(aA, aC);
    oLink.Kind := dgcCurved;
    oLink.StartArrow := dgaDiamond;
    oLink.EndArrow := dgaOpenTriangle;
    oLink.Style.Dash := dgdDash;
    with oLink.Labels.Add do
    begin
      Text := 'No';
      Position := 0.25;
    end;
  finally
    aDocument.EndUpdate;
  end;
end;
```

Voir aussi

- Ports et collage
- Éditions
- Formes
- TsgcDiagramConnector

Ports et collage

Un port est un point d'un nœud auquel s'attache l'extrémité d'un connecteur. Un nœud dont la collection Ports est vide utilise les ports par défaut de sa forme : `left`, `top`, `right` et `bottom`, les milieux des côtés (les coins d'une décision). Ajoutez vos propres ports pour donner à un nœud des entrées et des sorties typées.

TsgcDiagram-Port	Signification
Id	Nom du port, unique dans son nœud. Un connecteur y fait référence dans SourcePortId ou TargetPortId.
X, Y	Position en fraction du cadre du nœud, de 0 à 1.
Side	dgpAuto, dgpLeft, dgpTop, dgpRight ou dgpBottom : la direction dans laquelle un tracé quitte le port.
Flow	dgfBoth, dgfIn ou dgfOut.
Caption	Texte dessiné dans le nœud à côté du port. Les légendes sont masquées en dessous du zoom 0.4.

Collage

- Une extrémité de connecteur avec un Id de port est collée à ce port.
- Une extrémité de connecteur avec un Id de port vide est collée au contour du nœud : l'extrémité suit le nœud et rejoint le contour sur la ligne vers le centre.
- Une extrémité de connecteur sans nœud est libre à SourcePoint ou TargetPoint.
- Node.PortPoint(aPortId) renvoie la position monde d'un port.

Dans la vue

ShowPorts décide quand les ports sont dessinés : dgvNever, dgvHover (la valeur par défaut, les ports du nœud sous le pointeur) ou dgvAlways. Un glissement depuis un port trace un nouveau connecteur, et pendant son tracé chaque port est une cible. Sur un nœud sélectionné, les poignées de redimensionnement recouvrent les ports latéraux, donc mettez Tool à dgtConnector pour tracer depuis eux. Avant qu'une extrémité soit collée, la vue déclenche OnConnecting, où vous pouvez la refuser ; une fois le connecteur créé, elle déclenche OnConnected.

Exemple

```
uses
  sgcDiagram_Types, sgcDiagram_Model, sgcDiagram_VCL;

type
  TNodeEditor = class
  public
    procedure DoConnecting(Sender: TObject; aSource: TsgcDiagramNode;
      const aSourcePort: string; aTarget: TsgcDiagramNode;
      const aTargetPort: string; var aAccept: Boolean);
  end;

function AddFilter(aDocument: TsgcDiagramDocument): TsgcDiagramNode;
begin
  Result := aDocument.AddNode(dgsRoundRect, 0, 0, 160, 80, 'Filter');
  with Result.Ports.Add do
  begin
```

```
    Id := 'in';
    X := 0;
    Y := 0.5;
    Side := dgpLeft;
    Flow := dgfIn;
    Caption := 'rows';
  end;
  with Result.Ports.Add do
  begin
    Id := 'out';
    X := 1;
    Y := 0.5;
    Side := dgpRight;
    Flow := dgfOut;
    Caption := 'result';
  end;
end;

procedure TNodeEditor.DoConnecting(Sender: TObject; aSource: TsgcDiagramNode;
  const aSourcePort: string; aTarget: TsgcDiagramNode;
  const aTargetPort: string; var aAccept: Boolean);
begin
  // an output may only connect to an input
  aAccept := (aSourcePort = 'out') and (aTargetPort = 'in');
end;

procedure Connect(aView: TsgcVCLDiagramView; aEditor: TNodeEditor);
var
  oA, oB: TsgcDiagramNode;
begin
  aView.ShowPorts := dgvAlways;
  aView.OnConnecting := aEditor.DoConnecting;
  oA := AddFilter(aView.Diagram);
  oB := AddFilter(aView.Diagram);
  oB.Left := 260;
  aView.Diagram.AddConnector(oA, oB, 'out', 'in');
end;
```

Voir aussi

- Connecteurs et routage
- Édition et clavier
- TsgcDiagramPort

Édition et clavier

La vue modifie le document par son Controller. Chaque action de l'utilisateur qui modifie le document est une étape d'annulation, et Esc pendant un glissement l'annule.

Outils

Outil	Effet
dgtSelect	La valeur par défaut. Cliquer pour sélectionner, faire glisser pour déplacer, les poignées redimensionnent et font pivoter, un glissement sur un espace vide trace un cadre de sélection, un glissement depuis un port trace un connecteur.
dgtPan	Un glissement avec le bouton gauche fait défiler la vue.
dgtConnector	Un glissement depuis n'importe quel nœud trace un connecteur.
dgtShape	Un clic ou un glissement crée un nœud de la forme de ShapeTool. L'outil reste actif après chaque nœud ; Esc revient à dgtSelect. Un clic sur un élément de la palette active cet outil.

Souris

- Shift et un clic ajoutent à la sélection, Ctrl et un clic basculent un élément, Alt et un clic sélectionnent un membre d'un groupe.
- Shift pendant le redimensionnement conserve les proportions ; Shift pendant la rotation aimante l'angle par pas de 15 degrés.
- La molette fait défiler verticalement, Shift et la molette horizontalement, Ctrl et la molette zooment autour du pointeur.
- Le bouton du milieu, ou le bouton gauche avec Space maintenu, fait défiler la vue.
- Le bouton droit sélectionne l'élément sous le pointeur, prêt pour un menu contextuel.
- Un double-clic sur un nœud modifie son texte et déclenche OnItemDblClick.

Clavier

Touches	Action
Ctrl+Plus, Ctrl+Minus	Zoomer avant et arrière autour du centre de la vue.
Ctrl+0, Ctrl+1	Zoom 1, ajuster à la vue.
Ctrl+A, Ctrl+C	Tout sélectionner, copier.
Ctrl+X, Ctrl+V, Ctrl+D	Couper, coller, dupliquer.
Ctrl+Z, Ctrl+Y ou Ctrl+Shift+Z	Annuler, rétablir.
Ctrl+G, Ctrl+Shift+G	Grouper, dissocier.
Delete	Supprime la sélection (les éléments verrouillés restent).
Touches fléchées	Déplacent la sélection de GridSize ; avec Alt d'une unité monde.

F2, ou n'importe quel caractère	Modifie le texte du nœud sélectionné.
Tab, Shift+Tab	Sélectionne l'élément suivant ou précédent.
Esc	Annule un glissement, quitte l'outil courant ou efface la sélection.

En mode ReadOnly, seuls les touches de zoom, Ctrl+A, Ctrl+C, Tab et Esc fonctionnent.

Modification du texte

Le texte est modifié dans un mémo placé sur le nœud. Enter ou un clic ailleurs valide, Esc annule, Shift+Enter commence une nouvelle ligne. OnTextEditing peut refuser la modification en mettant aAllow à False, et OnTextEdited se déclenche après une validation. EditText(aNode) démarre l'éditeur par code.

Exemple

```
uses
  sgcDiagram_Types, sgcDiagram_Model, sgcDiagram_VCL;

type
  TEditorEvents = class
  public
    procedure DoTextEditing(Sender: TObject; aNode: TsgcDiagramNode;
      var aAllow: Boolean);
  end;

procedure TEditorEvents.DoTextEditing(Sender: TObject; aNode: TsgcDiagramNode;
  var aAllow: Boolean);
begin
  // the terminators keep their text
  aAllow := aNode.Shape <> dgsTerminator;
end;

procedure DrawDecisions(aView: TsgcVCLDiagramView; aEvents: TEditorEvents);
begin
  aView.OnTextEditing := aEvents.DoTextEditing;
  aView.ShapeTool := dgsDecision;
  aView.Tool := dgtShape;
end;
```

Voir aussi

- Magnétisme et repères
- Annuler et presse-papiers
- Mode visionneuse
- TsgcDiagramController
- Événements

Magnétisme et repères

Deux aides alignent les nœuds pendant que l'utilisateur les déplace, les redimensionne ou les crée.

Propriété	Effet
Diagram.GridSize	Pas de la grille en unités monde, 10 par défaut. Les touches fléchées déplacent la sélection de cette valeur.
ShowGrid	Dessine la grille dans la vue.
SnapToGrid	Aimante sur la grille les nœuds déplacés, redimensionnés et créés, ainsi que les formes déposées.
SnapToGuides	Affiche des repères d'alignement : pendant qu'un nœud est déplacé ou redimensionné, ses bords et ses lignes médianes s'aimantent aux bords et aux lignes médianes des nœuds voisins et un repère est dessiné.

- Un repère capte un bord ou un centre à 5 pixels de l'appareil près, par rapport aux nœuds situés à moins de 400 pixels de l'appareil.
- Les repères s'appliquent aux nœuds qui ne sont pas pivotés.
- Shift pendant le redimensionnement désactive les repères pour ce glissement.
- Alt avec une touche fléchée déplace la sélection d'une unité monde, hors de la grille.

Exemple

```
uses
  sgcDiagram_VCL;

procedure SetupSnapping(aView: TsgcVCLDiagramView);
begin
  aView.Diagram.GridSize := 20;
  aView.ShowGrid := True;
  aView.SnapToGrid := True;
  aView.SnapToGuides := True;
end;
```

Voir aussi

- Édition et clavier
- Formes

Annuler et presse-papiers

Le document tient une pile d'annulation et une pile de rétablissement. Chaque action de l'utilisateur dans la vue est une étape. Dans le code, BeginTransaction(aCaption) et EndTransaction regroupent un nombre quelconque de modifications en une seule étape, et Rollback annule les modifications de la transaction ouverte.

- Undo, Redo, CanUndo, CanRedo, UndoCaption et RedoCaption agissent sur les piles ; la vue et le contrôleur ont eux aussi Undo et Redo.
- UndoLimit (100 par défaut) est le nombre d'étapes conservées.
- Une modification faite par code hors d'une transaction vide les deux piles, si bien qu'une annulation ne restaure jamais un état périmé.
- BeginUndoSuspend et EndUndoSuspend encadrent une reconstruction en masse qui ne doit pas être enregistrée : ils vident les piles et ne donnent aucune étape, et les vues sont quand même notifiées. Ils peuvent être imbriqués.
- Document.Clear n'est pas une étape d'annulation.

Presse-papiers

Le presse-papiers est interne à l'application. CopySelection renvoie les nœuds sélectionnés, les enfants des conteneurs sélectionnés, les connecteurs dont les deux extrémités sont copiées et les images qu'ils utilisent, sous forme de fragment JSON, et le conserve dans le presse-papiers interne de l'application. Paste(aJSON) insère un fragment en une seule étape d'annulation et sélectionne les éléments collés ; Paste sans argument colle le presse-papiers interne, 20 unités monde plus loin à chaque collage répété. La vue a CopyToClipboard, CutToClipboard, PasteFromClipboard et DeleteSelection, et Ctrl+D duplique la sélection.

Exemple

```
uses
  sgcDiagram_Types, sgcDiagram_Model;

procedure AddApproval(aDocument: TsgcDiagramDocument);
begin
  aDocument.BeginTransaction('Add approval');
  try
    aDocument.AddNode(dgsDecision, 0, 0, 160, 90, 'Approved?');
    aDocument.AddNode(dgsProcess, 0, 140, 160, 60, 'Notify');
    aDocument.EndTransaction;
  except
    aDocument.Rollback;
    raise;
  end;
  // one step: undo removes both nodes
  if aDocument.CanUndo then
    aDocument.Undo;
end;

procedure DuplicateSelection(aDocument: TsgcDiagramDocument);
var
  vFragment: string;
begin
  vFragment := aDocument.CopySelection;
  aDocument.Paste(vFragment);
end;
```

Voir aussi

- Sérialisation
- Édition et clavier

Calques

Les calques organisent les éléments d'un diagramme, par exemple un calque d'arrière-plan avec des cadres et des notes sous le calque de travail. Chaque nœud et chaque connecteur nomme son calque dans sa propriété Layer ; un nom vide désigne le calque par défaut.

TsgcDiagramLayer	Signification
Name	Nom utilisé par la propriété Layer des éléments.
Visible	False masque les éléments du calque dans la vue et dans l'export.
Locked	True empêche de sélectionner et de modifier les éléments du calque.

- Layers.Add, Layers.Find(aName) et les Items de la collection gèrent les calques. Ajouter, supprimer et renommer un calque sont des étapes d'annulation ; Visible et Locked n'en sont pas.
- Document.LayerVisible(aName) et LayerLocked(aName) indiquent comment un calque est traité.
- Un élément isolé peut aussi être verrouillé avec sa propre propriété Locked.

Édition Community

Sans SGC_DIAGRAM_ADVANCED, un seul calque peut être ajouté : un second Layers.Add lève EsgcDiagramError. Un fichier avec plus de calques se charge et s'enregistre toujours sans changement, mais chaque calque est dessiné, exporté et modifié comme visible et non verrouillé.

Exemple

```
uses
  sgcDiagram_Types, sgcDiagram_Model;

procedure AddBackground(aDocument: TsgcDiagramDocument);
var
  oLayer: TsgcDiagramLayer;
  oNote: TsgcDiagramNode;
begin
  oLayer := aDocument.Layers.Add;
  oLayer.Name := 'Background';
  oLayer.Locked := True;
  oNote := aDocument.AddNode(dgsText, 0, -60, 300, 40, 'Draft, not approved');
  oNote.Layer := 'Background';
end;
```

Voir aussi

- Éditions
- Mode visionneuse
- TsgcDiagramLayer

Sérialisation

Un document est enregistré et chargé en JSON. Le format est le même sur chaque plateforme et dans chaque édition.

Méthode	Effet
SaveToJSON, LoadFromJSON	Vers et depuis une chaîne.
SaveToFile, LoadFromFile	Vers et depuis un fichier.
SaveToStream, LoadFromStream	Vers et depuis un flux.
CopySelection, Paste	Un fragment du document, voir Annuler et presse-papiers.

- La sortie est canonique : un ordre de clés fixe, les propriétés à leur valeur par défaut sont omises, aucune indentation. Le même document donne toujours le même texte.
- Les images du document sont stockées sous forme de données PNG dans le fichier.
- Le lecteur est tolérant : les clés inconnues et les noms d'énumération inconnus sont ignorés. Un fichier écrit par une version de format plus récente lève EsgcDiagramError, et une erreur d'analyse indique sa position.
- Le chargement remplace le contenu du document. Les tracés de connecteurs enregistrés dans le fichier sont conservés.
- Une vue enregistre son diagramme dans le fichier dfm ou fmx comme propriété DiagramJSON, seulement lorsque le diagramme n'est pas vide.

Exemple

```
uses
  SysUtils,
  sgcDiagram_Model;

procedure SaveAndReload(aDocument: TsgcDiagramDocument; const aFileName: string);
var
  vJSON: string;
begin
  aDocument.SaveToFile(aFileName);
  aDocument.LoadFromFile(aFileName);
  // or through a string, for a database field
  vJSON := aDocument.SaveToJSON;
  aDocument.LoadFromJSON(vJSON);
end;
```

Un petit fichier

Deux nœuds et un connecteur, indentés ici pour la lecture :

```
{
  "format": "sgcDiagram",
  "version": 1,
  "nodes": [
    {
      "id": "n1",
      "shape": "terminator",
      "x": 40,
      "y": 40,
      "w": 140,
      "h": 50,
      "text": "Start"
```

```
    },
    {
      "id": "n2",
      "shape": "process",
      "x": 40,
      "y": 140,
      "w": 140,
      "text": "Save"
    }
  ],
  "connectors": [
    {
      "id": "c3",
      "source": {
        "node": "n1"
      },
      "target": {
        "node": "n2"
      },
      "labels": [
        {
          "text": "Go"
        }
      ],
      "points": [[110, 90], [110, 140]]
    }
  ]
}
```

Voir aussi

- Annuler et presse-papiers
- Export
- Format JSON

Export

Un diagramme est exporté en PNG et en SVG. L'image couvre les limites du contenu du document plus une marge.

Méthode	Remarques
View.SaveToPNG(aFileName or aStream, aScale)	Exporte le diagramme de la vue avec FontFile et OnDrawNode appliqués. aScale est le nombre de pixels par unité monde, 1 par défaut.
View.SaveToSVG(aFileName or aStream, aScale)	La même chose en SVG.
TsgcDiagramRenderer.SaveToPNGFile, SaveToPNGStream	(aDocument, aFileName ou aStream, aScale = 1, aMargin = 20). Aucune vue nécessaire.
TsgcDiagramRenderer.SaveToSVGFile, SaveToSVGStream	La même chose en SVG.
TsgcDiagramRenderer.RenderTo	(aCanvas, aDocument, aViewport) peint une partie du document sur un TsgcUICanvas de sgcUICore.

- L'export utilise le thème du document, donc un diagramme sombre est exporté en sombre.
- Le moteur de rendu fonctionne dans une application console, un service et sous Linux64 ; voir Démarrage rapide Headless.
- L'export PDF et l'impression ne font pas partie de la version 1.

Exemple

```
uses
  SysUtils,
  sgcDiagram_VCL;

procedure ExportView(aView: TsgcVCLDiagramView; const aFolder: string);
begin
  // 2 pixels per world unit, for a sharp image on a slide
  aView.SaveToPNG(IncludeTrailingPathDelimiter(aFolder) + 'diagram.png', 2);
  aView.SaveToSVG(IncludeTrailingPathDelimiter(aFolder) + 'diagram.svg');
end;
```

Voir aussi

- Thèmes
- Sérialisation
- TsgcDiagramRenderer

Thèmes

Les vues, les palettes et l'export sont dessinés avec les thèmes de sgcUICore, si bien qu'un diagramme s'accorde avec les autres produits sgcUI sur la même fiche.

- **Theme** de la vue, de la palette et du document vaut utLight (par défaut), utDark ou utCustom. Le Theme d'une vue définit le Theme de son Diagram.
- **Style** de chaque nœud et connecteur : Fill, Stroke, StrokeWidth, Dash, CornerRadius et Text (Size, Bold et Color). Une couleur est un TsgcUIColor ARGB, par exemple sgcUIARGB($FF, $00, $96, $88) ; 0 garde la couleur du thème, qui dépend de la forme.
- **NodeStyle** et **ConnectorStyle** du document sont les modèles copiés dans chaque nouveau nœud et connecteur.
- **FontFile** de la vue dessine le texte avec un fichier TrueType au lieu de la police intégrée à sgcUICore.
- **OnDrawNode** de la vue vous permet de peindre un nœud vous-même ou de compléter le dessin par défaut : mettez aDefault à False pour l'ignorer.

Exemple

```
uses
  sgcUICore_Theme, sgcDiagram_Types, sgcDiagram_Model, sgcDiagram_VCL;

procedure ApplyCorporateStyle(aView: TsgcVCLDiagramView);
var
  oNode: TsgcDiagramNode;
begin
  aView.Theme := utDark;
  // new nodes get a teal fill and bold white text
  aView.Diagram.NodeStyle.Fill := sgcUIARGB($FF, $00, $96, $88);
  aView.Diagram.NodeStyle.Text.Bold := True;
  aView.Diagram.NodeStyle.Text.Color := sgcUIARGB($FF, $FF, $FF, $FF);
  aView.Diagram.ConnectorStyle.StrokeWidth := 2;
  // one node with its own style
  oNode := aView.Diagram.AddNode(dgsRoundRect, 0, 0, 160, 60, 'Alert');
  oNode.Style.Fill := sgcUIARGB($FF, $E5, $39, $35);
  oNode.Style.Dash := dgdDash;
  // 0 hands the colour back to the theme
  oNode.Style.Stroke := 0;
end;
```

Voir aussi

- Export
- Formes
- TsgcDiagramStyle

Mode visionneuse

Mettez ReadOnly à True pour afficher un diagramme que l'utilisateur peut explorer mais pas modifier, comme une carte de processus dans un tableau de bord ou un organigramme généré.

- L'utilisateur peut sélectionner des éléments, zoomer avec Ctrl et la molette ou les touches, faire défiler avec la molette, le bouton du milieu, Space ou l'outil de défilement, et copier la sélection.
- Rien n'est déplacé, redimensionné, connecté, supprimé, collé ou modifié, et la palette ne peut pas déposer de formes.
- Le pointeur devient une main au-dessus d'un élément, et OnItemClick, OnItemDblClick et OnHover signalent l'élément, de sorte que l'application peut ouvrir une page de détail ou afficher une info-bulle.
- ZoomToFit, ZoomToSelection et CenterOn déplacent la vue par code ; MinZoom (0.05) et MaxZoom (8) bornent le zoom.

Pour ne protéger que certains éléments, laissez ReadOnly à False et mettez Locked sur ces éléments ou sur leur calque.

Exemple

```
uses
  SysUtils,
  sgcDiagram_Types, sgcDiagram_Model, sgcDiagram_VCL;

type
  TViewerEvents = class
  public
    procedure DoItemClick(Sender: TObject; aItem: TsgcDiagramItem);
  end;

procedure TViewerEvents.DoItemClick(Sender: TObject; aItem: TsgcDiagramItem);
begin
  if aItem is TsgcDiagramNode then
    // Data holds an application key, for example a record id
    Writeln('open ' + TsgcDiagramNode(aItem).Data);
end;

procedure ShowProcess(aView: TsgcVCLDiagramView; aEvents: TViewerEvents;
  const aFileName: string);
begin
  aView.Diagram.LoadFromFile(aFileName);
  aView.ReadOnly := True;
  aView.OnItemClick := aEvents.DoItemClick;
  aView.ZoomToFit;
end;
```

Voir aussi

- Édition et clavier
- Calques
- TsgcVCLDiagramView
- TsgcFMXDiagramView

Vue d'ensemble de .NET

sgcDiagram .NET est un portage C# managé de la bibliothèque sgcDiagram de Delphi, pas un wrapper autour d'elle. Les classes C# vivent dans l'espace de noms **esegece.sgcDiagram** et gardent les noms Delphi, si bien que **TsgcDiagramDocument**, **TsgcDiagramNode**, **TsgcDiagramConnector**, **TsgcDiagramController**, **TsgcDiagramRenderer** et les énumérations se lisent de la même façon dans les deux langages. La vue et la palette suivent le modèle de nommage Delphi sur chaque hôte : **TsgcVCLDiagramView** devient **TsgcWinFormsDiagramView**, **TsgcWpfDiagramView** et **TsgcAvaloniaDiagramView**, et **TsgcVCLDiagramPalette** devient **TsgcWinFormsDiagramPalette**, **TsgcWpfDiagramPalette** et **TsgcAvaloniaDiagramPalette**. Les contrôles sont dessinés avec le portage .NET de **sgcUICore**, le même canevas, les mêmes polices et les mêmes thèmes clair et sombre que les autres produits de la suite visuelle d'eSeGeCe.
Le document, la pile d'annulation, le format JSON, le routeur, les dispositions, le peintre et le contrôleur sont le code Delphi porté ligne à ligne. Un diagramme enregistré par une application .NET s'ouvre dans une application Delphi et inversement, le texte JSON et l'export SVG sont les mêmes qu'en Delphi, et les dispositions et les routes donnent les mêmes coordonnées.

Paquets

Le portage est réparti en cinq paquets NuGet, de sorte qu'un projet n'inclut que l'hôte qu'il utilise.

PackageId	Contient	Frameworks cibles
esegece.sgcDiagram	Le document avec ses nœuds, connecteurs, ports, libellés, calques, images et styles, annuler et rétablir, le format JSON et le presse-papiers interne, le routeur, les dispositions en couches et en arbre, le modèle de la palette, le contrôleur, le peintre, l'export SVG de **TsgcDiagramRenderer** et **TsgcDiagramDataBinding**. Aucune dépendance native. Dépend de esegece.sgcUICore.	netstandard2.0, net472, net6.0, net8.0, net9.0
esegece.sgcDiagram.Rendering	La sortie PNG : les méthodes d'extension **SaveToPNGStream** et **SaveToPNGFile** de **TsgcDiagramRenderer** et le raster du cache de la vue, dessinés sur le canevas SkiaSharp de esegece.sgcUICore.Rendering, sous Windows, Linux et macOS.	netstandard2.0, net472, net6.0, net8.0, net9.0
esegece.sgcDiagram.WinForms	**TsgcWinFormsDiagramView** et **TsgcWinFormsDiagramPalette**.	net472, net8.0-windows, net9.0-windows
esegece.sgcDiagram.Wpf	**TsgcWpfDiagramView** et **TsgcWpfDiagramPalette**.	net472, net8.0-windows, net9.0-windows
esegece.sgcDiagram.Avalonia	**TsgcAvaloniaDiagramView** et **TsgcAvaloniaDiagramPalette**, sous Windows, Linux et macOS avec Avalonia 11.	net8.0, net9.0

Les trois paquets d'hôte dépendent de esegece.sgcDiagram.Rendering, qui dépend de esegece.sgcDiagram, donc une seule ligne `PackageReference` apporte tout ce dont un hôte a besoin. Voir Installation depuis NuGet.

Éditions

Édition	Identifiants de paquet	Comprend
Registered	esegece.sgcDiagram, .Rendering, .WinForms, .Wpf et .Avalonia	Tout : formes, connecteurs, ports et collage, édition, aimantation et repères, annulation et presse-papiers, calques, JSON, export PNG et SVG, thèmes, le mode visionneuse et les fonctions avancées : les dispositions en couches et en arbre, le routage orthogonal avec évitement d'obstacles et ajustement, la liaison de données et plus d'un calque.
Community	Les mêmes identifiants avec le suffixe .Community, par exemple esegece.sgcDiagram.WinForms.Community.	Tout sauf les fonctions avancées. **TsgcDiagramLayeredLayout**, **TsgcDiagramTreeLayout** et **TsgcDiagramDataBinding** ne sont pas dans les paquets Community, les connecteurs **dgcOrthogonal** sont routés comme de simples coudes, et **Layers.Add** lève **EsgcDiagramError** lorsqu'un calque existe déjà.

C'est le même découpage que les éditions Delphi, et un fichier de diagramme est le même dans les deux, voir Éditions.

Hôtes

Hôte	Équivalent Delphi	Plateformes
WinForms	VCL (TsgcVCLDiagramView, TsgcVCLDiagramPalette)	Windows, .NET Framework 4.7.2 et .NET 8 et 9.
WPF	VCL et FMX	Windows, .NET Framework 4.7.2 et .NET 8 et 9. Les contrôles peuvent être déclarés en XAML.
Avalonia	FMX (TsgcFMXDiagramView, TsgcFMXDiagramPalette), avec zoom par pincement et défilement à un doigt sur un écran tactile	Windows, Linux et macOS, .NET 8 et 9.
Sans interface	TsgcDiagramDocument and TsgcDiagramRenderer	N'importe quelle plateforme des frameworks cibles : JSON et SVG avec esegece.sgcDiagram, PNG avec esegece.sgcDiagram.Rendering.

Chaque hôte a la même API : les propriétés, méthodes et événements de la vue et de la palette Delphi avec les mêmes noms. Voir Classes .NET pour les noms des classes et Différences de .NET par rapport à Delphi pour ce qui change par rapport à Delphi.

Voir aussi

- Installation depuis NuGet
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Classes .NET
- Différences de .NET par rapport à Delphi

- Liaison de données dans .NET
- Rendu sans interface dans .NET
- Démos .NET
- Plateformes

Installation depuis NuGet

Les paquets sont sur nuget.org. Ajoutez le paquet de l'hôte, les paquets du noyau et du rendu viennent avec lui comme dépendances.

Édition Registered

```
dotnet add package esegece.sgcDiagram.WinForms
dotnet add package esegece.sgcDiagram.Wpf
dotnet add package esegece.sgcDiagram.Avalonia
```

Édition Community

```
dotnet add package esegece.sgcDiagram.WinForms.Community
dotnet add package esegece.sgcDiagram.Wpf.Community
dotnet add package esegece.sgcDiagram.Avalonia.Community
```

Ou ajoutez la référence au fichier de projet à la main :

```
<ItemGroup>
  <PackageReference Include="esegece.sgcDiagram.Wpf" Version="2026.10.0" />
</ItemGroup>
```

Référencez dans un projet soit les paquets Registered, soit les paquets Community, jamais les deux : ils contiennent les mêmes types dans le même espace de noms.

Paramètres du projet

- WinForms : un projet `net8.0-windows` avec `<UseWindowsForms>true</UseWindowsForms>`, ou un projet .NET Framework 4.7.2.
- WPF : un projet `net8.0-windows` avec `<UseWPF>true</UseWPF>`, ou un projet .NET Framework 4.7.2. Associez l'espace de noms en XAML avec `xmlns:sgc="clr-namespace:esegece.sgcDiagram;assembly=esegece.sgcDiagram.Wpf"`.
- Avalonia : un projet `net8.0` ou `net9.0` avec les paquets **Avalonia.Desktop** et **Avalonia.Themes.Fluent** d'Avalonia 11.3, comme toute application de bureau Avalonia. L'éditeur de texte en place est une **TextBox** Avalonia et tire son aspect du thème Fluent.
- Sans fenêtre : esegece.sgcDiagram seul suffit pour le document, le JSON et le SVG. Ajoutez esegece.sgcDiagram.Rendering pour le PNG, il apporte SkiaSharp et ses ressources natives pour Windows et Linux.

En C#, chaque hôte a besoin de `using esegece.sgcDiagram;`, et de `using esegece.sgcUICore;` pour **TsgcUITheme**, **TsgcUINotifyEvent** et les autres types de sgcUICore.

Voir aussi

- Vue d'ensemble de .NET
- Démarrage rapide WinForms
- Démarrage rapide WPF
- Démarrage rapide Avalonia
- Démos .NET

Démarrage rapide WinForms

Ce démarrage rapide construit un formulaire en code avec une palette de formes à gauche, une vue de diagramme occupant le reste et un petit organigramme dans la vue. Glissez une forme de la palette vers la vue, ou cliquez dessus puis sur la vue, et connectez deux nœuds depuis un port.

Étapes

1. Créez un projet Windows Forms pour `net8.0-windows` et ajoutez le paquet **esegece.sgcDiagram.WinForms** (**esegece.sgcDiagram.WinForms.Community** dans l'édition Community).
2. Ajoutez `using esegece.sgcDiagram;` et `using esegece.sgcUICore;`.
3. Créez une **TsgcWinFormsDiagramView** et une **TsgcWinFormsDiagramPalette** et ajoutez-les au formulaire. Ce sont des contrôles WinForms ordinaires : Dock, Anchor, TabIndex et Enabled fonctionnent comme d'habitude.
4. Définissez **View** de la palette sur la vue. La palette charge les formes Flowchart et Basic lors de sa création.
5. Remplissez **Diagram**, le **TsgcDiagramDocument** de la vue, dans une transaction pour que toute la mise en place soit une seule étape d'annulation, et affectez les événements.

Exemple

Remarques

- **Tool** choisit ce que fait un appui (**dgtSelect**, **dgtPan**, **dgtConnector**, **dgtShape** avec **ShapeTool**), et **ReadOnly** transforme la vue en visionneuse, comme en Delphi.
- Le clavier est celui de Delphi, voir Édition et clavier. L'éditeur en place est une **TextBox** multiligne, exposée comme **TextEditor** : Entrée valide, Maj+Entrée est un retour à la ligne, Échap annule.
- **Theme** est propre à chaque contrôle, définissez-le donc sur la vue et sur la palette. La vue le transmet à son document.
- Le diagramme est stocké dans la propriété masquée **DiagramJSON**, de sorte qu'une vue placée avec le concepteur conserve le diagramme qui y est dessiné.
- **SaveToPNG** et **SaveToSVG** écrivent le diagramme entier dans un fichier ou un flux, à une échelle.

Voir aussi

- Vue d'ensemble de .NET
- Installation depuis NuGet
- Classes .NET
- Différences de .NET par rapport à Delphi
- Démarrage rapide VCL

Démarrage rapide WPF

Les contrôles WPF peuvent être déclarés en XAML. Cette fenêtre a la palette à gauche et la vue occupant le reste ; le code-behind lie la palette à la vue et ajoute un petit organigramme.

Étapes

1. Créez un projet WPF pour `net8.0-windows` et ajoutez le paquet **esegece.sgcDiagram.Wpf** (**esegece.sgcDiagram.Wpf.Community** dans l'édition Community).
2. Associez l'espace de noms dans la fenêtre : `xmlns:sgc="clr-namespace:esegece.sgcDiagram;assembly=esegece.sgcDiagram.Wpf"`.
3. Déclarez une **TsgcWpfDiagramPalette** et une **TsgcWpfDiagramView** et définissez leurs propriétés comme attributs.
4. Dans le code-behind, définissez **View** de la palette, remplissez **Diagram** et affectez les événements.

XAML

```
<Window x:Class="QuickStart.QuickStartWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:sgc="clr-namespace:esegece.sgcDiagram;assembly=esegece.sgcDiagram.Wpf"
        Title="sgcDiagram" Width="900" Height="600">
  <DockPanel>
    <sgc:TsgcWpfDiagramPalette x:Name="palShapes" DockPanel.Dock="Left" Width="180" />
    <sgc:TsgcWpfDiagramView x:Name="dgmFlow" ShowGrid="True" SnapToGrid="True" ShowPorts="dgvHover" />
  </DockPanel>
</Window>
```

Code behind

Remarques

- Les propriétés sont de simples propriétés CLR, donc XAML les définit comme valeurs d'attribut. **View** de la palette prend l'objet vue, définissez-le en code.
- Un diagramme peut être donné en XAML avec l'attribut **DiagramJSON**. Le texte JSON commence par une accolade, écrivez-le donc après l'échappement `{}`.
- Les coordonnées de l'API, y compris **ScreenToDiagram** et **DiagramToScreen**, sont des pixels de périphérique du contrôle, c'est-à-dire des pixels indépendants du périphérique fois l'échelle DPI.
- Appelez **Dispose** de la vue à la fermeture de la fenêtre : il arrête le minuteur de la vue et libère son contrôleur.

Voir aussi

- Vue d'ensemble de .NET
- Installation depuis NuGet
- Classes .NET
- Différences de .NET par rapport à Delphi
- Démarrage rapide VCL

Démarrage rapide Avalonia

Les contrôles Avalonia s'exécutent sous Windows, Linux et macOS. Ce démarrage rapide construit une fenêtre en code avec la palette à gauche et la vue occupant le reste, dans le thème sombre.

Étapes

1. Créez un projet de bureau Avalonia pour `net8.0` avec Avalonia 11.3 et le thème Fluent, et ajoutez le paquet **esegece.sgcDiagram.Avalonia** (**esegece.sgcDiagram.Avalonia.Community** dans l'édition Community).
2. Créez une **TsgcAvaloniaDiagramView** et une **TsgcAvaloniaDiagramPalette**, définissez **View** de la palette et placez les deux dans un panneau.
3. Remplissez **Diagram** et affectez les événements.

Exemple

Remarques

- Les événements des contrôles Avalonia sont des propriétés délégué, affectées avec `=`, alors que les contrôles WinForms et WPF ont des événements C#, affectés avec `+=`.
- **Theme** des contrôles Avalonia masque la propriété **Theme** d'Avalonia (le thème du contrôle) et prend un **TsgcUITheme**.
- Sur un écran tactile, un pincement zoome et un doigt sur un espace vide fait défiler ; un doigt sur un nœud le déplace, comme dans la vue FMX.
- Un glisser-déposer depuis la palette utilise `DragDrop.DoDragDropAsync` avec le format d'application **DragFormat** de la vue (**CS_DIAGRAM_AVALONIA_DRAG_FORMAT**). **DragPayloadOf** et **CreateDragData** lisent et construisent ces données pour une source de glisser-déposer personnelle.
- Avalonia ne libère pas les contrôles, appelez donc **Dispose** de la vue à la fermeture de la fenêtre.

Voir aussi

- Vue d'ensemble de .NET
- Installation depuis NuGet
- Classes .NET
- Différences de .NET par rapport à Delphi
- Démarrage rapide FMX

Classes .NET

Les classes C# portent les noms des classes Delphi. Les classes du modèle, des moteurs et du peintre sont les mêmes sur chaque hôte ; les contrôles ont une classe par hôte.

Contrôles

Delphi	WinForms	WPF	Avalonia
TsgcVCLDiagramView, TsgcFMXDiagramView	TsgcWinFormsDiagramView	TsgcWpfDiagramView	TsgcAvaloniaDiagramView
TsgcVCLDiagramPalette, TsgcFMXDiagramPalette	TsgcWinFormsDiagramPalette	TsgcWpfDiagramPalette	TsgcAvaloniaDiagramPalette
Classe de base	TsgcUIWinFormsControl	TsgcUIWpfControl	TsgcUIAvaloniaControl

La vue a les propriétés Delphi **Diagram**, **Controller**, **Tool**, **ShapeTool**, **ReadOnly**, **Zoom**, **MinZoom**, **MaxZoom**, **ShowGrid**, **SnapToGrid**, **SnapToGuides**, **ShowPorts**, **Theme** et **FontFile**, ainsi que **DiagramJSON** et **TextEditor** ; les méthodes **ZoomToFit**, **ZoomToSelection**, **CenterOn**, **ScreenToDiagram**, **DiagramToScreen**, **Undo**, **Redo**, **CopyToClipboard**, **CutToClipboard**, **PasteFromClipboard**, **DeleteSelection**, **EditText**, **SaveToPNG** et **SaveToSVG** ; et les événements **OnChange**, **OnSelectionChange**, **OnItemClick**, **OnItemDblClick**, **OnHover**, **OnConnecting**, **OnConnected**, **OnDrawNode**, **OnTextEditing**, **OnTextEdited** et **OnZoomChange**. La palette a **Model**, **View**, **Items**, **IconSize**, **ShowCaptions**, **Theme**, **DragPayload**, **LoadDefaultItems** et **OnItemClick**.

Modèle et moteurs

Dans le paquet esegece.sgcDiagram :

Classe	Description
TsgcDiagramDocument	Le diagramme : **Nodes**, **Connectors**, **Layers**, **Images**, **Selection**, **NodeStyle**, **ConnectorStyle**, **Routing**, **Theme**, **GridSize** ; **AddNode**, **AddConnector**, **BeginTransaction**, **EndTransaction**, **Undo**, **Redo**, **RouteAll**, **SaveToJSON**, **LoadFromJSON**, **SaveToFile**, **LoadFromFile**, **CopySelection**, **Paste**. Une vue crée le sien, un programme sans interface en crée un avec `new TsgcDiagramDocument()` .
TsgcDiagramNode, TsgcDiagramConnector	Les éléments, avec **Style**, **Layer** et **Data** ; un nœud a des **Ports**, un connecteur a des **Labels**, comme en Delphi.
TsgcDiagramPort, TsgcDiagramLabel, TsgcDiagramLayer, TsgcDiagramImage, TsgcDiagramStyle	Les parties des éléments et du document. Les collections sont **TsgcDiagramNodes**, **TsgcDiagramConnectors**, **TsgcDiagramPorts**, **TsgcDiagramLabels** et **TsgcDiagramLayers**, avec `Add()` , `Count` et un indexeur.
TsgcDiagramUndoStack	La pile d'annulation d'un document, accessible par **UndoStack**.
TsgcDiagramController	La logique d'édition partagée par les hôtes : test de sélection, les états de glissement, la table des touches, l'aimantation, les repères et les dépôts. Une vue l'expose comme **Controller**.

TsgcDiagramPaletteModel, TsgcDiagramPaletteItem	La palette sans hôte : éléments, catégories, disposition, dessin, test de sélection et la charge utile de glisser-déposer.
TsgcDiagramPainter, TsgcDiagramViewCache	Le peintre du document et le raster en cache d'une vue.
TsgcDiagramRenderer	Export sans interface. **SaveToSVGStream** et **SaveToSVGFile** sont dans esegece.sgcDiagram, **SaveToPNGStream** et **SaveToPNGFile** sont des méthodes d'extension dans esegece.sgcDiagram.Rendering.
TsgcDiagramLayeredLayout, TsgcDiagramTreeLayout	Les dispositions automatiques. Édition Registered.
TsgcDiagramDataBinding	Construit un diagramme à partir d'une `DataTable`, voir Liaison de données dans .NET. Édition Registered.
EsgcDiagramError	L'exception de la bibliothèque.

Classes statiques

Les constantes et fonctions de niveau unité de Delphi sont des membres statiques d'une classe portant le nom de l'unité : **sgcDiagram_JSON** (**sgcDiagramSaveJSON**, **sgcDiagramLoadJSON**, **sgcDiagramCopy**, **sgcDiagramPaste**), **sgcDiagram_Render**, **sgcDiagram_Controller** (les constantes **CS_DIAGRAM_*** et les codes de touche **CS_DIAGRAM_KEY_***), **sgcDiagram_Palette** et **sgcDiagram_Types**. **sgcDiagram_Render_Rendering.RegisterRaster** installe le raster Skia ; les contrôles l'appellent dans leur constructeur. Le format de glisser-déposer de la palette est **sgcDiagram_WinForms.CS_DIAGRAM_DRAG_FORMAT**, **sgcDiagram_Wpf.CS_DIAGRAM_DRAG_FORMAT** et **TsgcAvaloniaDiagramView.CS_DIAGRAM_AVALONIA_DRAG_FORMAT**, tous `esegece.sgcDiagram.Shape`.

Énumérations

Les énumérations et leurs valeurs gardent les noms Delphi : **TsgcDiagramShape** (**dgsRectangle**, **dgsDecision** et les autres formes), **TsgcDiagramConnectorKind**, **TsgcDiagramArrow**, **TsgcDiagramTool**, **TsgcDiagramShowPorts**, **TsgcDiagramDirection**, **TsgcDiagramTreeStyle**, **TsgcDiagramBindingLayout**. Une valeur s'écrit avec son type, par exemple `TsgcDiagramShape.dgsProcess`. Les ensembles Delphi sont des énumérations `[Flags]` : **TsgcDiagramChanges** et **TsgcDiagramShiftState**.

Voir aussi

- Vue d'ensemble de .NET
- Différences de .NET par rapport à Delphi
- TsgcDiagramDocument
- TsgcDiagramController
- TsgcVCLDiagramView

Différences de .NET par rapport à Delphi

L'API C# suit l'API Delphi nom pour nom. Ce qui change vient du langage et des frameworks :

Delphi	.NET
Les événements sont des pointeurs de méthode, Sender est le composant.	Les contrôles WinForms et WPF ont des événements C# (`+=`), Sender est le contrôle. Les contrôles Avalonia, le document, le contrôleur, le modèle de la palette et la liaison de données ont des propriétés délégué (`=`).
Paramètres `var` des événements (**OnTextEditing**, **OnConnecting**, **OnDrawNode**).	Paramètres `ref` . Une lambda avec un paramètre `ref` nomme les types de tous ses paramètres.
Propriétés indexées et valeurs lues par une fonction.	Méthodes : **ContentBounds()**, **UndoCaption()**, **CanUndo()**, **Viewport()** du contrôleur, **GetCollapsed**, **SetCollapsed**, **GetCategory**, **MaxScroll()** et **Payload(i)** du modèle de la palette. Un élément de collection s'ajoute avec `Add()` .
Arguments **TList**.	`List<TsgcDiagramItem>` , `List<TsgcDiagramNode>` and `List<TsgcDiagramConnector>` .
UndoStack est un **TObject**.	C'est toujours un `object` : faites un cast vers **TsgcDiagramUndoStack**.
Free.	Le ramasse-miettes. Les vues WPF et Avalonia sont `IDisposable` , appelez **Dispose** à la fermeture de la fenêtre ; une vue WinForms est libérée avec son formulaire. **TsgcDiagramController** et **TsgcDiagramDataBinding** sont `IDisposable` .
Le diagramme enregistré dans le fichier DFM ou FMX.	La propriété masquée **DiagramJSON**, écrite par le concepteur WinForms seulement lorsque le diagramme n'est pas vide, et utilisable comme attribut XAML.
Codes de touche et **TShiftState**.	Codes de touche virtuels `int` (**CS_DIAGRAM_KEY_*** ; les valeurs **Keys** de WinForms sont les mêmes) et l'énumération `[Flags]` **TsgcDiagramShiftState**.
Les couleurs sont des **TsgcUIColor**.	Valeurs ARGB `uint` , par exemple `0xFFDBEAFE` ; 0 est la couleur du thème.
Export PNG du **TsgcDiagramRenderer**.	Méthodes d'extension de esegece.sgcDiagram.Rendering, dessinées avec Skia. Les fichiers PNG ne sont pas identiques octet par octet à ceux de Delphi ; le texte SVG et JSON l'est.
TsgcDiagramDataBinding est un composant avec une **TDataSource**.	Une classe simple avec une `DataTable` , voir Liaison de données dans .NET.

Exemple

Des événements avec des paramètres `ref`, un ensemble de flags et les méthodes qui remplacent des propriétés Delphi :

Comportement identique

- Unités monde, l'axe y vers le bas, angles en degrés dans le sens horaire, la grille, l'aimantation et l'ordre de sélection.
- Les libellés d'annulation, une étape d'annulation par transaction, le format du presse-papiers interne et la charge utile de glisser-déposer de la palette.
- Le format JSON, les routes, les dispositions et l'export SVG, avec les mêmes nombres et le même texte que Delphi.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- Événements
- Format JSON

Liaison de données dans .NET

Le **TsgcDiagramDataBinding** de Delphi construit un diagramme à partir d'un TDataSet : un nœud par enregistrement et un connecteur de chaque enregistrement vers son parent. Dans .NET la même classe lit une `System.Data.DataTable`, la table qu'un `DataAdapter` de n'importe quel fournisseur ADO.NET remplit, ou une **IsgcUITable** de sgcUICore. C'est une fonctionnalité de l'édition Registered.

Propriété	Description
DataSource	La `DataTable` des enregistrements.
Table	Une **IsgcUITable**, lue avant **DataSource** lorsqu'elle est affectée.
KeyField, ParentField, ParentSeparator, LabelField	La colonne de la clé, la colonne de la clé du parent (plusieurs parents séparés par **ParentSeparator**, un point-virgule par défaut) et la colonne du texte du nœud.
Target, Document	**Target** est une vue de diagramme, ou tout objet qui implémente **IsgcDiagramDocumentOwner** ; **Document** est un document dans lequel écrire directement, pour un programme sans vue.
Layout, Direction, TreeStyle	La disposition du diagramme reconstruit (**dgbAuto**, **dgbTree**, **dgbLayered**, **dgbNone**) et ses options.
NodeShape, NodeWidth, NodeHeight, ConnectorKind	La forme et la taille des nœuds et le type des connecteurs.
LiveUpdates, Active	Le diagramme est reconstruit après une modification de la table tant que **LiveUpdates** est true ; définir **Active** le construit.
OnCreateNode, OnCreateConnector	Déclenché pour chaque nœud et connecteur créé, avec l'index de la ligne. **CurrentRow** est la `DataRow` du nœud pendant l'exécution de **OnCreateNode**.

Exemple

Différences par rapport à TDataSet

- Une `DataTable` n'a pas de curseur : **OnCreateNode** lit donc la ligne par **CurrentRow** au lieu des champs du dataset.
- La liaison écoute les événements de la table : une ligne ajoutée, modifiée ou supprimée, une modification rejetée et une table vidée reconstruisent le diagramme une fois. `AcceptChanges` et le déplacement dans les lignes ne reconstruisent rien. Une ligne en cours d'édition (`BeginEdit`) est utilisée lorsque son `EndEdit` s'exécute. Les lignes supprimées et détachées sont ignorées.
- La reconstruction est postée, comme le **ForceQueue** de Delphi. Dans une application WinForms, WPF ou Avalonia, elle est postée sur le **SynchronizationContext** courant lors de la création de la liaison, donc rien n'est à appeler. Un programme console ou un service n'a pas de contexte : appelez **CheckQueued** dans le thread du document, le **CheckSynchronize** de Delphi, ou **Refresh** pour reconstruire immédiatement.
- La liaison n'est pas un composant : elle est `IDisposable`, et une table ou une cible libérée se délie d'elle-même.

Lier les contrôles

Les propriétés des contrôles sont de simples propriétés CLR, pas des propriétés de dépendance ni stylisées, car un diagramme est un modèle entier et non une valeur unique. Dans une application MVVM, gardez le **TsgcDia-**

gramDocument ou son JSON dans le modèle de vue et réagissez à **OnChange** et **OnSelectionChange** de la vue.

Voir aussi

- Vue d'ensemble de .NET
- Différences de .NET par rapport à Delphi
- TsgcDiagramDataBinding
- Liaison de données

Rendu sans interface dans .NET

Le document n'a pas besoin de fenêtre. Un service, un test ou un rapport peut construire ou charger un diagramme et l'enregistrer en JSON, SVG et PNG sur toute plateforme, Linux compris.

- **TsgcDiagramDocument** se crée avec `new TsgcDiagramDocument()`. Les connecteurs sont routés à la fin de la transaction ou de la mise à jour la plus externe.
- **TsgcDiagramRenderer** écrit le diagramme entier avec une marge (20 par défaut) à une échelle : **SaveToSVGFile** et **SaveToSVGStream** avec esegece.sgcDiagram, **SaveToPNGFile** et **SaveToPNGStream** avec esegece.sgcDiagram.Rendering.
- **SaveToJSON**, **LoadFromJSON**, **SaveToFile**, **LoadFromFile**, **SaveToStream** et **LoadFromStream** du document lisent et écrivent le format de fichier Delphi.

Exemple

Disposition automatique

Dans l'édition Registered, un diagramme construit en code peut être disposé avant d'être enregistré. **Execute** est une étape d'annulation et route de nouveau les connecteurs.

Remarques

- La taille de l'image est l'étendue du contenu plus la marge, fois l'échelle ; une échelle de 2 écrit une image deux fois plus large et deux fois plus haute.
- **Theme** du document choisit les couleurs claires ou sombres de l'export.
- **Painter.SetFontFile** du renderer prend un fichier de police TrueType normal et un en gras, pour un serveur sans la police par défaut.
- Les contrôles ont aussi **SaveToPNG** et **SaveToSVG**.

Voir aussi

- Vue d'ensemble de .NET
- Classes .NET
- Démarrage rapide Headless
- TsgcDiagramRenderer
- Disposition en couches
- Disposition en arbre

Démos .NET

Les démos se trouvent dans `.net\demos\Diagram`, un dossier par démo avec son projet, son `Program.cs` et un README.md. Elles construisent leur interface en code, sans XAML. Chaque démo a une option `--test` qui exécute un auto-contrôle hors écran, écrit des lignes PASS et FAIL et se termine avec le nombre de contrôles échoués, et une option `--snapshot` qui écrit le diagramme dans un fichier PNG.

Démo	Hôte	Montre
FlowchartEditor	WinForms	Une palette et une vue, la jumelle de la démo FlowchartEditor de Delphi : glissez des formes depuis la palette, connectez-les avec l'outil connecteur ou depuis un port, annuler et rétablir, zoom, thèmes, la disposition en couches et des fichiers JSON, PNG et SVG.
FlowchartWpf	WPF	Le même éditeur avec **TsgcWpfDiagramPalette** et **TsgcWpfDiagramView**.
FlowchartAvalonia	Avalonia	Le même éditeur avec **TsgcAvaloniaDiagramPalette** et **TsgcAvaloniaDiagramView**, pour Windows, Linux et macOS, avec zoom par pincement et défilement à un doigt.
OrgChart	WinForms	Une `DataTable` de 60 employés dans un `DataGridView` et, par un **TsgcDiagramDataBinding**, un organigramme dans une vue en lecture seule avec la disposition en arbre ; une modification dans la grille met à jour l'organigramme. Édition Registered.
NodeEditor	WinForms	Un éditeur de graphes de nœuds : des blocs avec des ports d'entrée et de sortie typés, connectés seulement lorsque les types correspondent.
ConsoleRender	Console	Sans fenêtre : construit en code un organigramme de tickets de support, le dispose avec la disposition en couches et écrit flowchart.png, flowchart.svg et flowchart.json, sous Windows et Linux. La jumelle de la démo Headless de Delphi.

La partie sans interface des trois démos Flowchart (l'organigramme d'exemple, les actions de la barre d'outils et l'auto-contrôle) se trouve dans `FlowchartEditor_Shared.cs`, que les projets WinForms, WPF et Avalonia compilent, si bien que les trois hôtes exécutent les mêmes contrôles.
Dans l'édition Community, les démos Flowchart gardent les positions approximatives de l'exemple et masquent le bouton de disposition, ConsoleRender place les lignes en code, et OrgChart affiche la table et indique que l'organigramme nécessite l'édition Registered.

Compiler les démos

Dans l'arborescence du produit, les démos se compilent contre les projets de la bibliothèque. Le zip des démos utilise par défaut les paquets NuGet de l'édition Community (OrgChart utilise les Registered). Compilez une démo avec les paquets Registered en définissant **SgcDiagramEdition** :

```
dotnet build FlowchartEditor\FlowchartEditor.csproj -p:SgcDiagramEdition=Registered
dotnet run --project ConsoleRender\ConsoleRender.csproj -- --test
```

Voir aussi

- Vue d'ensemble de .NET
- Installation depuis NuGet
- Démarrage rapide WinForms
- Rendu sans interface dans .NET

Dépannage

Problème	Cause et solution
Les connecteurs créés par code n'ont pas de tracé, ou le tracé ne suit pas un nœud déplacé.	Les tracés sont calculés à la fin de la paire BeginUpdate et EndUpdate la plus externe, et seulement tant que Routing.AutoRoute vaut True. Fermez chaque BeginUpdate, ou appelez RouteAll après une modification en masse faite avec AutoRoute désactivé.
Les connecteurs orthogonaux traversent les nœuds.	Dans l'édition Community, les connecteurs dgcOrthogonal sont de simples coudes sans évitement d'obstacles. L'édition Registered calcule le tracé autour des nœuds.
Layers.Add lève EsgcDiagramError.	L'édition Community autorise un calque. Voir Calques.
Un nœud image affiche un cadre barré.	La clé d'image n'est pas dans Images, ou les données ne sont pas un PNG et aucun gestionnaire OnDecodeImage ne les a décodées.
Supprimer une image lève EsgcDiagramError.	Un nœud l'utilise encore dans ImageKey. Modifiez ou supprimez d'abord ces nœuds.
Un glissement depuis le côté d'un nœud sélectionné le redimensionne au lieu de tracer un connecteur.	Les poignées de redimensionnement d'un nœud sélectionné recouvrent ses ports latéraux. Cliquez d'abord sur un espace vide, ou mettez Tool à dgtConnector.
L'éditeur de texte ne s'ouvre pas.	La vue est ReadOnly, le nœud ou son calque est Locked, ou un gestionnaire OnTextEditing a mis aAllow à False.
Coller ne fait rien après une copie dans une autre application.	Le presse-papiers de sgcDiagram est interne à l'application ; il n'échange pas de données avec d'autres programmes.
Annuler ne fait rien après une modification faite par code.	Une modification en dehors de BeginTransaction et EndTransaction vide les piles d'annulation et de rétablissement. Enveloppez les modifications du code dans une transaction pour pouvoir les annuler.
Les formes ne peuvent pas être glissées depuis la palette sous iOS ou Android.	Les plateformes mobiles n'ont pas de service de glisser-déposer. Appuyez sur un élément de la palette, puis sur la vue.
Tab ne parcourt pas les éléments dans une vue FireMonkey.	Une fiche FireMonkey traite Tab avant le contrôle ayant le focus et déplace le focus.
Définir l'Id d'un élément lève EsgcDiagramError.	Les Id sont uniques dans un document. Choisissez un Id qu'aucun autre nœud ni connecteur n'utilise.
Les classes de disposition et de liaison de données sont absentes.	Elles font partie de l'édition Registered. Avec l'édition Community, sgcDiagram_Layout et sgcDiagram_Data sont des unités vides.

Voir aussi

- Éditions
- Plateformes

Index

Copyright © 2012-2026 eSeGeCe Software
info@esegece.com
www.esegece.com