



# sgcOpenAPI 2024.2

February 2024

Documentation for Rad Studio

Copyright © 2012-2024 eSeGeCe Software

[info@esegece.com](mailto:info@esegece.com)

[www.esegece.com](http://www.esegece.com)

# Contents

---

- OpenAPI ..... 4**
- Parser ..... 5**
  - OpenAPI Parser Pascal ..... 5
  - OpenAPI Additional Properties..... 10
- Client ..... 12**
  - OpenAPI Client ..... 12
- APIs ..... 14**
  - OpenAPI Amazon AWS ..... 14
  - OpenAPI Amazon AWS Credentials..... 20
  - OpenAPI Amazon AWS S3 ..... 22
  - OpenAPI Google Cloud ..... 24
  - OpenAPI Google Cloud OAuth2..... 29
  - OpenAPI Google Cloud Service Accounts..... 32
  - OpenAPI Google Cloud PubSub ..... 37
  - OpenAPI Google Cloud Calendar ..... 38
  - OpenAPI Microsoft..... 40
  - OpenAPI Microsoft Tenant..... 45
  - OpenAPI Microsoft Register Application ..... 46
  - OpenAPI Microsoft OAuth2 Code..... 49
  - OpenAPI Microsoft OAuth2 Credentials..... 51
  - OpenAPI Microsoft Graph ..... 53
  - APIs ..... 54
  - AbstractApi Geolocation..... 55
- License ..... 56**

License ..... 56

**Index..... 58**

**PDF-Back-Cover ..... 59**

# OpenAPI

---

## OpenAPI 3.0

The **OpenAPI Specification**, previously known as the **Swagger Specification**, is a specification for machine-readable interface files for describing, producing, consuming, and visualizing RESTful web services. Previously part of the Swagger framework, it became a separate project in 2016, overseen by the OpenAPI Initiative, an open-source collaboration project of the Linux Foundation. Swagger and some other tools can generate code, documentation, and test cases given an interface file.

Applications implemented based on OpenAPI interface files can automatically generate documentation of methods, parameters and models. This helps keep the documentation, client libraries, and source code in sync.

## Pascal Parser

**sgcOpenAPI Generator** allows generation of API client libraries (SDK generation) automatically given an OpenAPI Spec, the following OpenAPI specifications are supported:

- **OpenAPI 3.\***
- **Swagger 2.\*** (automatically converted from 2.0 to 3.0)
- **Swagger 1.\*** (automatically converted from 1.0 to 3.0)

sgcOpenAPI allows to generate **automatically** the client API interface in **Native Pascal Language** given a JSON/YAML OpenAPI or Swagger. Currently supports from Delphi 7 to latest Delphi version.

**sgcOpenAPI Generator** allows to create a documentation file from an OpenAPI / Swagger specification.

Read more about [OpenAPI Parser Pascal](#).

## OpenAPI Client

The Client Interface generated contains all the functions/methods defined in the OpenAPI specification. The constants and enumerations are created too.

The following **Authentication** methods are supported:

- Basic Authentication
- OAuth2 Code (interactive)
- OAuth2 Credentials (non-interactive)
- JWT

Read more about [OpenAPI Client](#).

## APIs

The following APIs have been compiled are supported:

- [Amazon AWS](#)
- [Google Cloud APIs](#)
- [Microsoft Azure](#)
- [Other APIs](#)

# OpenAPI | Parser Pascal

The **sgcOpenAPI Parser** reads the **OpenAPI 3.0 Specification in JSON** Format and creates automatically a **Delphi Client in Native Pascal Code**.

The **sgcOpenAPI Parser** is compatible with the following specifications:

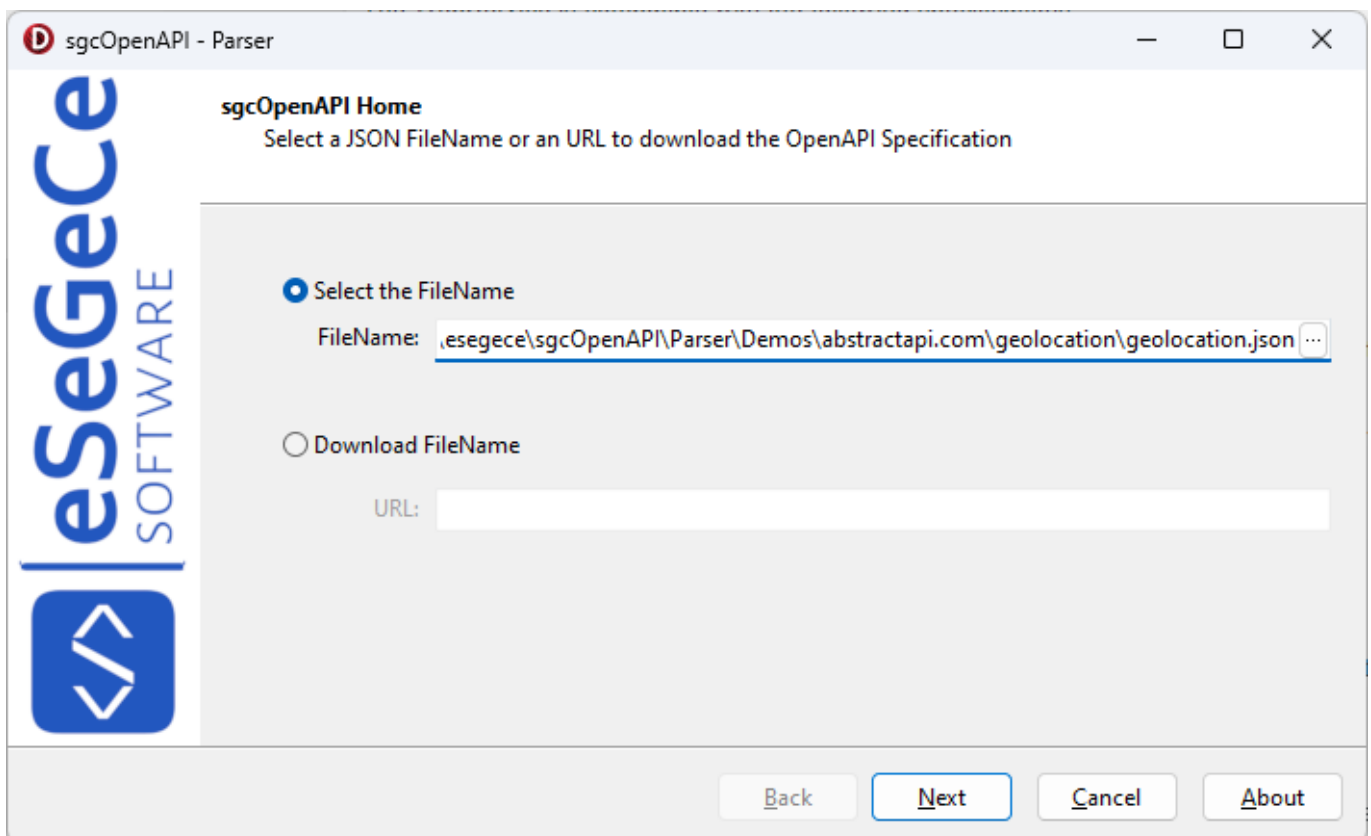
- **OpenAPI 3.\***
- **Swagger 2.\*** (automatically converted from 2.0 to 3.0)
- **Swagger 1.\*** (automatically converted from 1.0 to 3.0)

The specification file must be in **JSON** or **YAML** format.

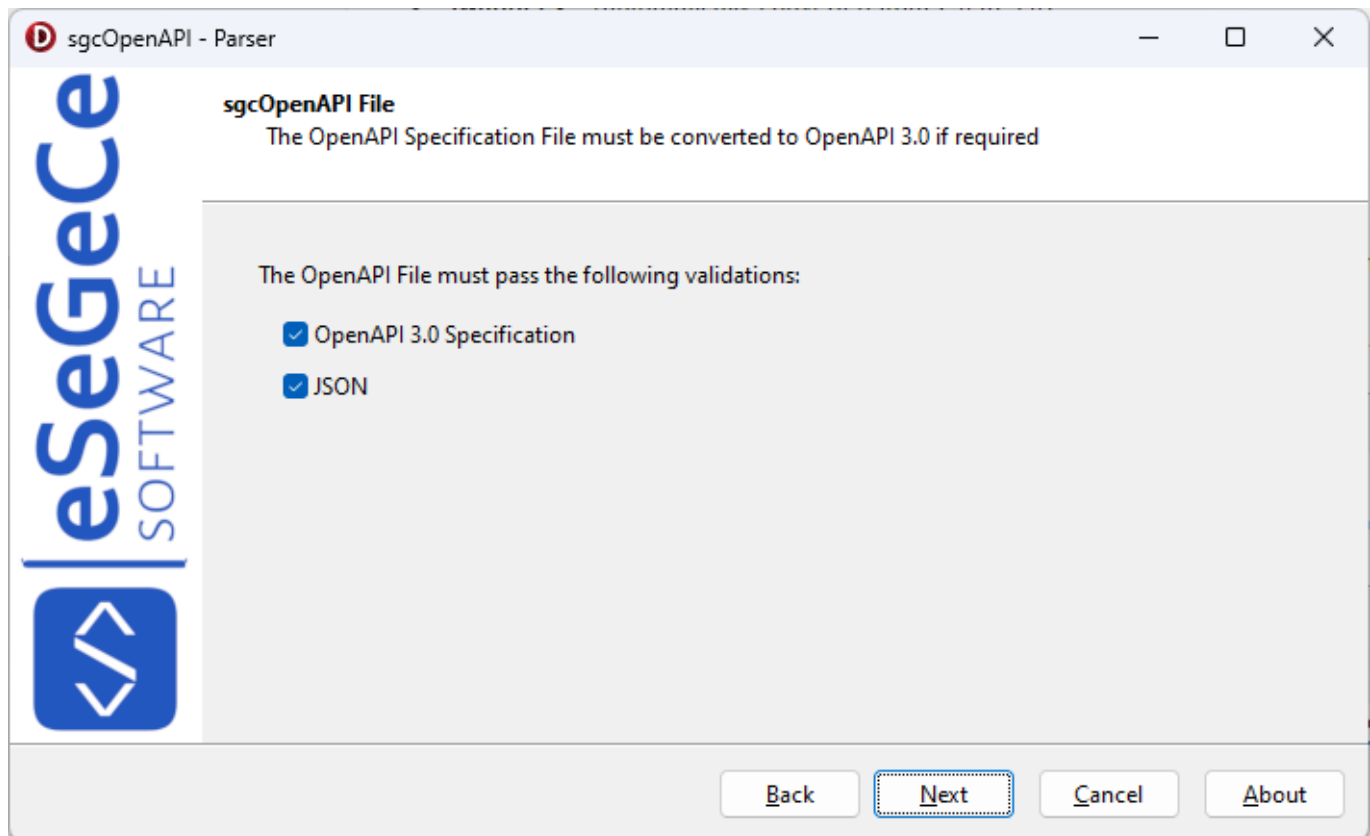
## Importing OpenAPI Specification

The first step is import the openAPI specification. Once you've the openAPI 3.0 specification in JSON format, you can generate the required Delphi files using our **OpenAPI WebService**. Follow the next steps:

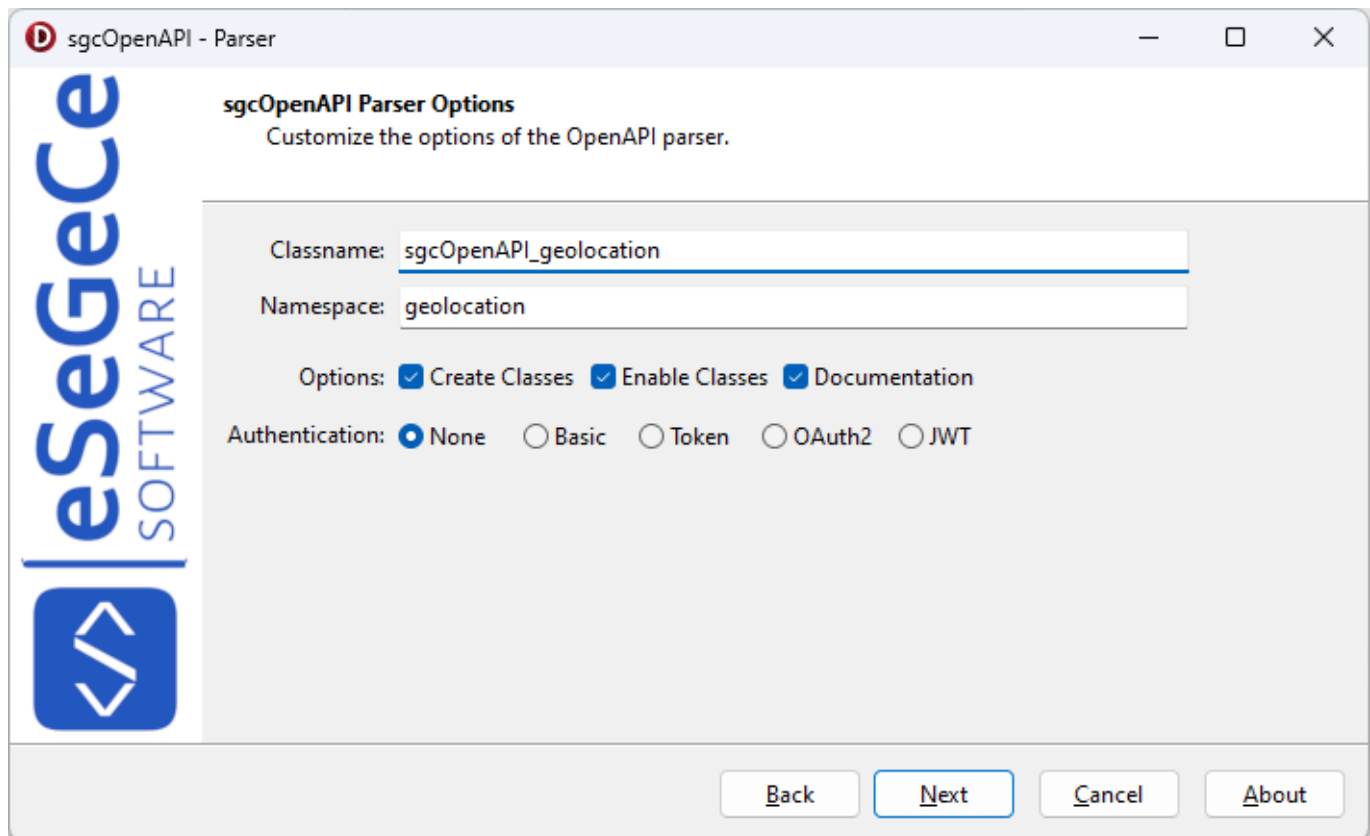
- **Execute the sgcOpenAPI Parser** and select the specification to import or the URL to download.



- The **specification is verified** and if it's compatible with the Parser you can continue to the next steps. If the specification makes use of an old version like swagger 2.0 it will be converted automatically. If there is any problem while converting the file, an error message will appear.



- Now you can **customize the following options** before parsing the document
  - **ClassName:** this is the main ClassName, by default starts with "sgcOpenAPI\_" and adds the name of the specification filename.
  - **Namespace:** this is the name of the pascal file generated, by default is the same name of the specification file with the extension ".pas".
  - **Create Classes:** if checked, it will create the classes from the specification file. Example: if a request requires to send a JSON object, and this JSON object is specified, the parser will create a class with the fields of the JSON object.
  - **Enable Classes:** if checked, enables the Classes Generated from the specification file to use with JSON objects (Requires Rad Studio XE7+).
  - **Documentation:** if checked, the parser will add comments to the fields, classes and methods.
  - **Authentication:** select any authentication if exists.
    - **None:** the API doesn't make use of any authentication method.
    - **Basic:** the API makes use of BASIC authentication.
    - **Token:** is required to send a Token as an HTTP Header. This token is obtained from any other external method.
    - **OAuth2:** the request will use OAuth2 to authenticate the HTTP Requests.
    - **JWT:** the request will use JWT to authenticate the HTTP Requests.



**sgcOpenAPI Parser Options**  
Customize the options of the OpenAPI parser.

Classname:

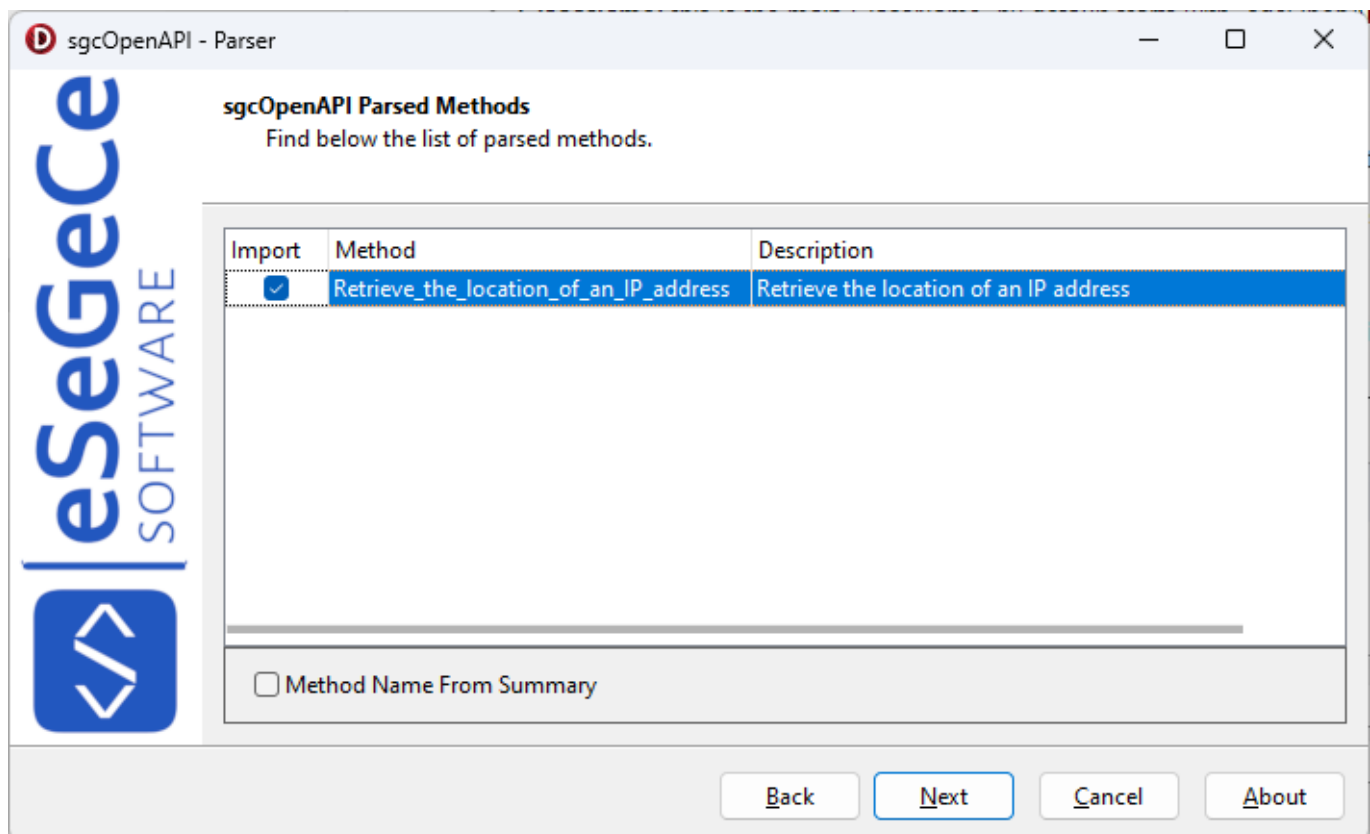
Namespace:

Options: ☒ Create Classes ☒ Enable Classes ☒ Documentation

Authentication: ☒ None ☐ Basic ☐ Token ☐ OAuth2 ☐ JWT

Buttons: Back, Next, Cancel, About

- Now a **grid with a list of methods parsed** will be shown as information. By default, the parser takes the OperationId as a name for the methods created, but here you can use the summary as method name if the OperationId is not defined or the value is not valid.



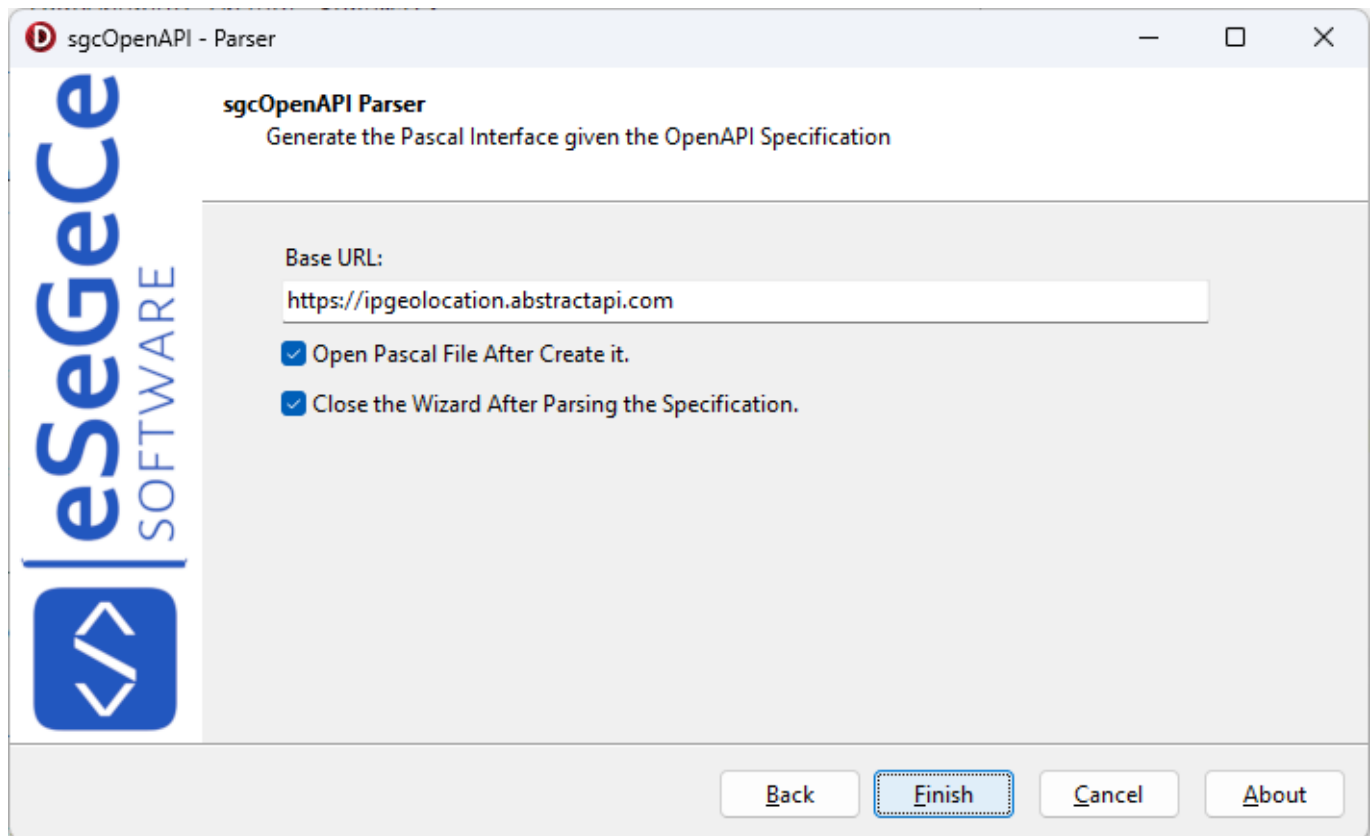
**sgcOpenAPI Parsed Methods**  
Find below the list of parsed methods.

Import	Method	Description
<input checked="" type="checkbox"/>	Retrieve_the_location_of_an_IP_address	Retrieve the location of an IP address

☐ Method Name From Summary

Buttons: Back, Next, Cancel, About

- Finally, verify the **Base URL** has the correct value and here you can customize if the generated file will be opened automatically after created and if the wizard will be closed after generate the file.



- Press **Finish** to parse the specification and generate the pascal file.

## Example


I will use a simple openAPI specification used by abstractapi.com to retrieve the location of an IP Address.

Before test the demo, you must create a free account in abstractapi.com to get an API Key.

<https://app.abstractapi.com/users/signup>

Install the [sgcOpenAPI Parser Setup](#), open the sgcOpenAPI.exe and import the OpenAPI specification that is located in the folder "Demos\abstractapi.com\geolocation". Once imported and stored in the same folder of the demo with the name "geolocation.pas", open the demo project, compile and execute it. Fill the API key obtained from the Abstractapi website and press Geolocation.



 sgcOpenAPI - Geolocation

API Key:

IP Address:

Geolocation

```
{
  "ip_address": "1.1.1.1",
  "city": "Sydney",
  "city_geoname_id": 2147714,
  "region": "New South Wales",
  "region_iso_code": "NSW",
  "region_geoname_id": 2155400,
  "postal_code": "1001",
  "country": "Australia",
  "country_code": "AU",
  "country_geoname_id": 2077456,
  "country_is_eu": false,
  "continent": "Oceania",
  "continent_code": "OC",
  "continent_geoname_id": 6255151,
  "longitude": 151.209,
  "latitude": -33.8688,
  "security": {
    "is_vpn": false
  },
  "timezone": {
    "name": "",
    "abbreviation": "",
    "gmt_offset": "",
    "current_time": "",
    "is_dst": ""
  },
  "flag": {
    "emoji": "ð\u0082\u2074",
    "unicode": "U+1F1E6 U+1F1FA",
    "png": "https://static.abstractapi.com/country-flags/AU_flag.png",
    "svg": "https://static.abstractapi.com/country-flags/AU_flag.svg"
  },
  "currency": {
    "currency_name": "Australian Dollars",
    "currency_code": "AUD"
  },
  "connection": {
    "autonomous_system_number": 13335,
    "autonomous_system_organization": "CLOUDFLARENET",
    "connection_type": "Corporate",
    "isp_name": "Cloudflare, Inc.",
    "organization_name": null
  }
}
```

# OpenAPI | Additional Properties

A dictionary (also known as a map, hashmap or associative array) is a set of key/value pairs. OpenAPI lets you define dictionaries where the keys are strings. To define a dictionary, use type: object and use the additionalProperties keyword to specify the type of values in key/value pairs. For example, a string-to-string dictionary like this:

```
{
  "en": "English",
  "fr": "French"
}
```

The OpenAPI Parser makes use of the JSON classes from Embarcadero, and converting a TDictionary to JSON, instead of creating a key/value pairs, it creates all the internal objects of the TDictionary, so the output is incorrect. The same applies when trying to convert a json string to a TDictionary object.

## Convert AdditionalProperties to JSON

The OpenAPI Parser creates the AdditionalProperties classes as type of TsgcAdditionalProperties, so if you must convert a class to json string, you must create a new class that inherits from TsgcAdditionalProperties, create all the fields you need and then assign to the property class.

**Example:** given a class with 2 properties, one is an Additional properties and the json output must be

```
{ "Property1": "value1", "AdditionalProperties": { "key1": "value1", "key2": "value" } }
TMyClass = class(TsgcOpenAPIClass)
private
  FProperty1: string;
  FAdditionalProperties: TsgcAdditionalProperties;
public
  property Property1: string read FProperty1 write FProperty1;
  property AdditionalProperties: TsgcAdditionalProperties read FAdditionalProperties write FAdditionalProperties;
end;
```

```
{ "Property1": "value1", "AdditionalProperties": { "key1": "value1", "key2": "value" } }
TMyClass = class(TsgcOpenAPIClass)
private
  FProperty1: string;
  FAdditionalProperties: TsgcAdditionalProperties;
public
  property Property1: string read FProperty1 write FProperty1;
  property AdditionalProperties: TsgcAdditionalProperties read FAdditionalProperties write FAdditionalProperties;
end;
```

Create a new class that inherits from TsgcAdditionalProperties and create 2 properties

```
TCustomAdditionalProperties = class(TsgcAdditionalProperties)
private
  FKey1: string;
  FKey2: string;
public
  property Key1: string read FKey1 write FKey1;
  property Key2: string read FKey2 write FKey2;
end;
```

```
TCustomAdditionalProperties = class(TsgcAdditionalProperties)
private
  FKey1: string;
  FKey2: string;
```

```
public
  property Key1: string read FKey1 write FKey1;
  property Key2: string read FKey2 write FKey2;
end;
```

Finally Assign this class to the AdditionalProperties property:

```
oClass := TMyClass.Create;
oClass.Property1 := 'value1';
oAdditionalProperties := TCustomAdditionalProperties.Create;
oAdditionalProperties.Key1 := 'value1';
oAdditionalProperties.Key2 := 'value2';
oClass.AdditionalProperties := oAdditionalProperties;
```

```
TMyClass *oClass = new TMyClass();
oClass->Property1 = "value1";
TCustomAdditionalProperties *oAdditionalProperties = new TCustomAdditionalProperties();
oAdditionalProperties->Key1 = "value1";
oAdditionalProperties->Key2 = "value2";
oClass->AdditionalProperties = oAdditionalProperties;
```

## Convert JSON to AdditionalProperties

Due to the limitations of the JSON classes, the OpenAPI Generator creates an additional method to load the AdditionalProperties in the TsgcAdditionalProperties.Dictionary property after the JSON string is parsed. So to access the content of the AdditionalProperties, just access to the TsgcAdditionalProperties.Dictionary property.

# OpenAPI | Client

**TsgcOpenAPI\_Client** is a non-visual component that encapsulates the main methods and properties to make HTTP requests from a OpenAPI specification.

Every OpenAPI interface created with sgcOpenAPI Parser has **2 methods**

1. **GetOpenAPIClient**: it's a singleton function that returns an instance of the main class, if not exists, it creates automatically.
2. **FreeOpenAPIClient**: frees the main class if it's created.

## Example

Use the Abstractapi to retrieve the localization of an IP Address.

```
GetOpenAPIClient.Retrieve_the_location_of_an_IP_address('your api', '80.258.15.2');
```

```
GetOpenAPIClient->Retrieve_the_location_of_an_IP_address("your api", "80.258.15.2");
```

```
GetOpenAPIClient.Retrieve_the_location_of_an_IP_address("your api", "80.258.15.2");
```

## Authentication

- **Basic**: uses a username/password as an HTTP header to authenticate.
  - **UserName**: name of the user.
  - **Password**: secret.
- **OAuth2**: authenticates using OAuth2, supports 2 types of Authorization:
  - **auth2Code**: It's used to perform authentication and authorization in the majority of application types, including single page applications, web applications, and natively installed applications. The flow enables apps to securely acquire access\_tokens that can be used to access resources secured, as well as refresh tokens to get additional access\_tokens, and ID tokens for the signed in user.
  - **auth2ClientCredentials**: This type of grant is commonly used for server-to-server interactions that must run in the background, without immediate interaction with a user. These types of applications are often referred to as daemons or service accounts.
  - Read more about OAuth2.
- **JWT**: authenticates using JWT. Read more about JWT.

## TLSOptions

Allows to configure how connect to secure SSL/TLS servers using HTTP/1 protocol

**ALPNProtocols**: list of the ALPN protocols which will be sent to server.

**RootCertFile**: path to root certificate file.

**CertFile**: path to certificate file.

**KeyFile**: path to certificate key file.

**Password**: if certificate is secured with a password, set here.

**VerifyCertificate**: if certificate must be verified, enable this property.

**VerifyDepth**: is an Integer property that represents the maximum number of links permitted when verification is performed for the X.509 certificate.

**Version**: by default uses TLS 1.0, if server requires a higher TLS version, here can be selected.

**IOHandler**: select which library you will use to connection using TLS.

**iohOpenSSL**: uses OpenSSL library and is the default for Indy components. Requires to deploy openssl libraries for win32/win64.

**iohSChannel:** uses Secure Channel which is a security protocol implemented by Microsoft for Windows, doesn't require to deploy openssl libraries. Only works in Windows 32/64 bits.

**OpenSSL\_Options:** configuration of the openssl libraries.

**APIVersion:** allows to define which OpenSSL API will be used.

**oslAPI\_1\_0:** uses API 1.0 OpenSSL, it's latest supported by Indy

**oslAPI\_1\_1:** uses API 1.1 OpenSSL, requires our custom Indy library and allows to use OpenSSL 1.1.1 libraries (with TLS 1.3 support).

**oslAPI\_3\_0:** uses API 3.0 OpenSSL, requires our custom Indy library and allows to use OpenSSL 3.0.0 libraries (with TLS 1.3 support).

**LibPath:** here you can configure where are located the openssl libraries

**oslpNone:** this is the default, the openssl libraries should be in the same folder where is the binary or in a known path.

**oslpDefaultFolder:** sets automatically the openssl path where the libraries should be located for all IDE personalities.

**oslpCustomFolder:** if this is the option selected, define the full path in the property LibPath-Custom.

**LibPathCustom:** when LibPath = oslpCustomFolder define here the full path where are located the openssl libraries.

**UnixSymLinks:** enable or disable the loading of SymLinks under Unix systems (by default is enabled, except under OSX64):

**oslsSymLinksDefault:** by default are enabled except under OSX64 (after MacOS Monterey fails trying to load the library without version.).

**oslsSymLinksLoadFirst:** Load SymLinks and do before trying to load the version libraries.

**oslsSymLinksLoad:** Load SymLinks after trying to load the version libraries.

**oslsSymLinksDontLoad:** don't load the SymLinks.

**SChannel\_Options:** allows to use a certificate from Windows Certificate Store.

**CertHash:** is the certificate Hash. You can find the certificate Hash running a dir command in powershell.

**CipherList:** here you can set which Ciphers will be used (separated by ":" ). Example: CALG\_AES\_256:CALG\_AES\_128

**CertStoreName:** the store name where is stored the certificate. Select one of below:

**scsnMY** (the default)

**scsnCA**

**scsnRoot**

**scsnTrust**

**CertStorePath:** the store path where is stored the certificate. Select one of below:

**scspStoreCurrentUser** (the default)

**scspStoreLocalMachine**

## Log

If Log property is enabled it saves socket messages to a specified log file, useful for debugging.

**Log:** enable if you want to save the HTTP requests to a text file.

**LogFileName:** full path to the filename.

## Properties

Other properties that can be used to customize the OpenAPI client:

**EncodeBodyAsUTF8:** if enabled, the JSON body is encoded as UTF8 (by default false).

# OpenAPI | Amazon AWS

---

The **sgcOpenAPI Amazon AWS Client** (TsgcOpenAPI\_Amazon\_Client) has its own OpenAPI Client which inherits from [TsgcOpenAPI\\_Client](#).

This component has a property called **AmazonOptions** that includes all required configurations to connect to Amazon AWS Servers.

## AmazonOptions

In AmazonOptions you can define the required **AccessKey** and **SecretKey** (which must be generated previously from your Amazon Account), to authenticate against the Amazon AWS Servers.

An access key grants programmatic access to your resources. This means that you must guard the access key as carefully as the AWS account root user sign-in credentials.

It's a best practice to do the following:

1. **Create an IAM user**, and then **define that user's permissions** as narrowly as possible.
2. **Create the access key** under that IAM user.

Once you've the credentials, set in the following properties:

- AmazonOptions.AccessKey
- AmazonOptions.SecretKey

The AmazonOptions.JSON property allows to define if the responses are in JSON or XML.

IAM roles, users in AWS IAM Identity Center (successor to AWS Single Sign-On), and federated users have temporary security credentials. Temporary security credentials expire after a defined period of time or when the user ends their session. You can set the token for temporary credentials in the property:

- AmazonOptions.SessionToken

## Most common uses

- **Configuration**
  - [Amazon AWS Credentials](#)
- **APIs**
  - [Amazon AWS S3](#)

## sgcOpenAPI AWS SDK

Find below a list of the currently available APIs.

- Access Analyzer
- Alexa For Business
- Amazon API Gateway
- Amazon AppConfig
- Amazon Appflow
- Amazon AppIntegrations Service
- Amazon AppStream
- Amazon Athena

- Amazon Augmented AI Runtime
- Amazon Chime
- Amazon Chime SDK Identity
- Amazon Chime SDK Messaging
- Amazon CloudDirectory
- Amazon CloudFront
- Amazon CloudHSM
- Amazon CloudSearch
- Amazon CloudSearch Domain
- Amazon CloudWatch
- Amazon CloudWatch Application Insights
- Amazon CloudWatch Events
- Amazon CloudWatch Logs
- Amazon CodeGuru Profiler
- Amazon CodeGuru Reviewer
- Amazon Cognito Identity
- Amazon Cognito Identity Provider
- Amazon Cognito Sync
- Amazon Comprehend
- Amazon Connect Contact Lens
- Amazon Connect Customer Profiles
- Amazon Connect Participant Service
- Amazon Connect Service
- Amazon Data Lifecycle Manager
- Amazon Detective
- Amazon DevOps Guru
- Amazon DocumentDB with MongoDB compatibility
- Amazon DynamoDB
- Amazon DynamoDB Accelerator (DAX)
- Amazon DynamoDB Streams
- Amazon EC2 Container Registry
- Amazon EC2 Container Service
- Amazon Elastic Inference
- Amazon Elastic Block Store
- Amazon Elastic Compute Cloud
- Amazon Elastic Container Registry Public
- Amazon Elastic File System
- Amazon Elastic Kubernetes Service
- Amazon Elastic Transcoder
- Amazon ElastiCache
- Amazon Elasticsearch Service
- Amazon EMR
- Amazon EMR Containers
- Amazon EventBridge
- Amazon Forecast Query Service
- Amazon Forecast Service
- Amazon Fraud Detector
- Amazon FSx
- Amazon GameLift
- Amazon Glacier
- Amazon GuardDuty
- Amazon HealthLake
- Amazon Honeycode
- Amazon Import/Export Snowball
- Amazon Inspector
- Amazon Interactive Video Service
- Amazon Kinesis
- Amazon Kinesis Analytics
- Amazon Kinesis Firehose
- Amazon Kinesis Video Signaling Channels
- Amazon Kinesis Video Streams
- Amazon Kinesis Video Streams Archived Media
- Amazon Kinesis Video Streams Media
- Amazon Lex Model Building Service
- Amazon Lex Model Building V2

- Amazon Lex Runtime Service
- Amazon Lex Runtime V2
- Amazon Lightsail
- Amazon Location Service
- Amazon Lookout for Equipment
- Amazon Lookout for Metrics
- Amazon Lookout for Vision
- Amazon Machine Learning
- Amazon Macie
- Amazon Macie 2
- Amazon Managed Blockchain
- Amazon Mechanical Turk
- Amazon MemoryDB
- Amazon Mobile Analytics
- Amazon Neptune
- Amazon OpenSearch Service
- Amazon Personalize
- Amazon Personalize Events
- Amazon Personalize Runtime
- Amazon Pinpoint
- Amazon Pinpoint Email Service
- Amazon Pinpoint SMS and Voice Service
- Amazon Polly
- Amazon Prometheus Service
- Amazon QLDB
- Amazon QLDB Session
- Amazon QuickSight
- Amazon Redshift
- Amazon Rekognition
- Amazon Relational Database Service
- Amazon Route 53
- Amazon Route 53 Domains
- Amazon Route 53 Resolver
- Amazon S3 on Outposts
- Amazon SageMaker Edge Manager
- Amazon SageMaker Feature Store Runtime
- Amazon SageMaker Runtime
- Amazon SageMaker Service
- Amazon Simple Email Service
- Amazon Simple Notification Service
- Amazon Simple Queue Service
- Amazon Simple Storage Service
- Amazon Simple Systems Manager (SSM)
- Amazon Simple Workflow Service
- Amazon SimpleDB
- Amazon Textract
- Amazon Timestream Query
- Amazon Timestream Write
- Amazon Transcribe Service
- Amazon Translate
- Amazon WorkDocs
- Amazon WorkLink
- Amazon WorkMail
- Amazon WorkMail Message Flow
- Amazon WorkSpaces
- AmazonApiGatewayManagementApi
- AmazonApiGatewayV2
- AmazonMQ
- AmazonMWAA
- AmazonNimbleStudio
- AmplifyBackend
- Application Auto Scaling
- Application Migration Service
- Auto Scaling
- AWS Amplify



- AWS App Mesh
- AWS App Runner
- AWS Application Cost Profiler
- AWS Application Discovery Service
- AWS AppSync
- AWS Audit Manager
- AWS Auto Scaling Plans
- AWS Backup
- AWS Batch
- AWS Budgets
- AWS Certificate Manager
- AWS Certificate Manager Private Certificate Authority
- AWS Cloud Map
- AWS Cloud9
- AWS CloudFormation
- AWS CloudHSM V2
- AWS CloudTrail
- AWS CodeBuild
- AWS CodeCommit
- AWS CodeDeploy
- AWS CodePipeline
- AWS CodeStar
- AWS CodeStar connections
- AWS CodeStar Notifications
- AWS Comprehend Medical
- AWS Compute Optimizer
- AWS Config
- AWS Cost and Usage Report Service
- AWS Cost Explorer Service
- AWS Data Exchange
- AWS Data Pipeline
- AWS Database Migration Service
- AWS DataSync
- AWS Device Farm
- AWS Direct Connect
- AWS Directory Service
- AWS EC2 Instance Connect
- AWS Elastic Beanstalk
- AWS Elemental MediaConvert
- AWS Elemental MediaLive
- AWS Elemental MediaPackage
- AWS Elemental MediaPackage VOD
- AWS Elemental MediaStore
- AWS Elemental MediaStore Data Plane
- AWS Fault Injection Simulator
- AWS Global Accelerator
- AWS Glue
- AWS Glue DataBrew
- AWS Greengrass
- AWS Ground Station
- AWS Health APIs and Notifications
- AWS Identity and Access Management
- AWS Import/Export
- AWS IoT
- AWS IoT 1-Click Devices Service
- AWS IoT 1-Click Projects Service
- AWS IoT Analytics
- AWS IoT Core Device Advisor
- AWS IoT Data Plane
- AWS IoT Events
- AWS IoT Events Data
- AWS IoT Fleet Hub
- AWS IoT Greengrass V2
- AWS IoT Jobs Data Plane
- AWS IoT Secure Tunneling

- AWS IoT SiteWise
- AWS IoT Things Graph
- AWS IoT Wireless
- AWS Key Management Service
- AWS Lake Formation
- AWS Lambda
- AWS License Manager
- AWS Marketplace Catalog Service
- AWS Marketplace Commerce Analytics
- AWS Marketplace Entitlement Service
- AWS MediaConnect
- AWS MediaTailor
- AWS Migration Hub
- AWS Migration Hub Config
- AWS Mobile
- AWS Network Firewall
- AWS Network Manager
- AWS OpsWorks
- AWS OpsWorks CM
- AWS Organizations
- AWS Outposts
- AWS Performance Insights
- AWS Price List Service
- AWS Proton
- AWS RDS DataService
- AWS Resource Access Manager
- AWS Resource Groups
- AWS Resource Groups Tagging API
- AWS RoboMaker
- AWS Route53 Recovery Control Config
- AWS Route53 Recovery Readiness
- AWS S3 Control
- AWS Savings Plans
- AWS Secrets Manager
- AWS Security Token Service
- AWS SecurityHub
- AWS Server Migration Service
- AWS Service Catalog
- AWS Service Catalog App Registry
- AWS Shield
- AWS Signer
- AWS Single Sign-On
- AWS Single Sign-On Admin
- AWS Snow Device Management
- AWS SSO Identity Store
- AWS SSO OIDC
- AWS Step Functions
- AWS Storage Gateway
- AWS Support
- AWS Systems Manager Incident Manager
- AWS Systems Manager Incident Manager Contacts
- AWS Transfer Family
- AWS WAF
- AWS WAF Regional
- AWS WAFV2
- AWS Well-Architected Tool
- AWS X-Ray
- AWSKendraFrontendService
- AWSMarketplace Metering
- AWSServerlessApplicationRepository
- Braket
- CodeArtifact
- EC2 Image Builder
- Elastic Load Balancing
- FinSpace Public API

- FinSpace User Environment Management service
- Firewall Management Service
- Managed Streaming for Kafka
- Managed Streaming for Kafka Connect
- Redshift Data API Service
- Route53 Recovery Cluster
- Schemas
- Service Quotas
- Synthetics

# OpenAPI Amazon AWS | Credentials

AWS requires different types of security credentials depending on how you access AWS. For example, you need a user name and password to sign in to the AWS Management Console and you need **access keys** to make **programmatically calls to AWS**.

## Considerations

- Be sure to save the following in a secure location: the email address associated with your AWS account, the AWS account ID, the root user password, and your account access keys. If you forget or lose your root user password, you must have access to the email address associated with your account in order to reset it. If you forget or lose your access keys, you must sign into your account to create new ones.
- We strongly recommend that you create an IAM user with administrator permissions to use for everyday AWS tasks and lock away the password and access keys for the root user. Use the root user only for the tasks that are restricted to the root user.
- Security credentials are account-specific. If you have access to multiple AWS accounts, you have separate credentials for each account.
- Do not provide your AWS credentials to a third party.

## Programmatic access

You must provide your AWS access keys to make programmatic calls to AWS.

When you create your access keys, you create the access key ID (for example, AKIAIOSFODNN7EXAMPLE) and secret access key (for example, wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY) as a set. The secret access key is available for download only when you create it. If you don't download your secret access key or if you lose it, you must create a new one.

You can assign up to two access keys per user (root user or IAM user). Having two access keys is useful when you want to rotate them. When you disable an access key, you can't use it, but it counts toward your limit of two access keys. After you delete an access key, it's gone forever and can't be restored, but it can be replaced with a new access key.

### To manage access keys when signed in as the root user

1. Sign in to the AWS Management Console as the root user.
2. In the navigation bar on the upper right, choose your account name or number and then choose **My Security Credentials**.
3. Expand the **Access keys (access key ID and secret access key)** section.
4. Do one of the following:
  - To create an access key, choose **Create New Access Key**. If you already have two access keys, this button is disabled and you must delete an access key before you can create a new one. When prompted, choose either **Show Access Key** or **Download Key File**. This is your only opportunity to save your secret access key. After you've saved your secret access key in a secure location, choose **Close**.
  - To deactivate an access key, choose **Make Inactive**. When prompted for confirmation, choose **Deactivate**. A deactivated access key still counts toward your limit of two access keys.
  - To activate an access key, choose **Make Active**.
  - To delete an access key when you no longer need it, copy the access key ID and then choose **Delete**. Before you can delete the access key, you must choose **Deactivate**. We recommend that you verify that the access key is no longer in use before you permanently delete it. To confirm deletion, paste the access key ID in the text input field and then choose **Delete**.

### To manage access keys when signed in as an IAM user

1. Sign in to the AWS Management Console as an IAM user.
2. In the navigation bar on the upper right, choose your user name and then choose **My Security Credentials**.
3. Do one of the following:

- To create an access key, choose **Create access key**. If you already have two access keys, this button is disabled and you must delete an access key before you can create a new one. When prompted, choose either **Show secret access key** or **Download .csv file**. This is your only opportunity to save your secret access key. After you've saved your secret access key in a secure location, chose **Close**.
- To deactivate an access key, choose **Make inactive**. When prompted for confirmation, choose **Deactivate**. A deactivated access key still counts toward your limit of two access keys.
- To activate an access key, choose **Make active**. When prompted for confirmation, choose **Make active**.
- To delete an access key when you no longer need it, copy the access key ID and then choose **Delete**. This deactivates the access key. We recommend that you verify that the access key is no longer in use before you permanently delete it. To confirm deletion, paste the access key ID in the text input field and then choose **Delete**.

## sgcOpenAPI Configuration

Once you have your own AWS Access Keys, you must configure in the OpenAPI Amazon Client before you do any Request to the Amazon AWS Servers.

```
GetOpenAPIClient.AmazonOptions.AccessKey := 'AKIAIOSFODNN7EXAMPLE';
GetOpenAPIClient.AmazonOptions.SecretKey := 'wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY';
```

```
GetOpenAPIClient->AmazonOptions->AccessKey = "AKIAIOSFODNN7EXAMPLE";
GetOpenAPIClient->AmazonOptions->SecretKey = "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY";
```

```
GetOpenAPIClient.AmazonOptions.AccessKey := "AKIAIOSFODNN7EXAMPLE";
GetOpenAPIClient.AmazonOptions.SecretKey := "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY";
```

# OpenAPI Amazon AWS | S3

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. Customers of all sizes and industries can use Amazon S3 to store and protect any amount of data for a range of use cases, such as data lakes, websites, mobile applications, backup and restore, archive, enterprise applications, IoT devices, and big data analytics. Amazon S3 provides management features so that you can optimize, organize, and configure access to your data to meet your specific business, organizational, and compliance requirements.

## ListBuckets

```
GetOpenAPIClient.AmazonOptions.AccessKey := 'AKIAIOSFODNN7EXAMPLE';
GetOpenAPIClient.AmazonOptions.SecretKey := 'wJalrXUtnFEMI/K7MDENG/bPxrFcYEXAMPLEKEY';
ShowMessage(GetOpenAPIClient.ListBuckets());
```

```
GetOpenAPIClient->AmazonOptions->AccessKey = "AKIAIOSFODNN7EXAMPLE";
GetOpenAPIClient->AmazonOptions->SecretKey = "wJalrXUtnFEMI/K7MDENG/bPxrFcYEXAMPLEKEY";
GetOpenAPIClient->ListBuckets();
```

```
GetOpenAPIClient.AmazonOptions.AccessKey := "AKIAIOSFODNN7EXAMPLE";
GetOpenAPIClient.AmazonOptions.SecretKey := "wJalrXUtnFEMI/K7MDENG/bPxrFcYEXAMPLEKEY";
GetOpenAPIClient.ListBuckets();
```

## GetObject

```
GetOpenAPIClient.AmazonOptions.AccessKey := 'AKIAIOSFODNN7EXAMPLE';
GetOpenAPIClient.AmazonOptions.SecretKey := 'wJalrXUtnFEMI/K7MDENG/bPxrFcYEXAMPLEKEY';
ShowMessage(GetOpenAPIClient.GetObject('bucket_name'));
```

```
GetOpenAPIClient->AmazonOptions->AccessKey = "AKIAIOSFODNN7EXAMPLE";
GetOpenAPIClient->AmazonOptions->SecretKey = "wJalrXUtnFEMI/K7MDENG/bPxrFcYEXAMPLEKEY";
GetOpenAPIClient->GetObject("bucket_name");
```

```
GetOpenAPIClient.AmazonOptions.AccessKey := "AKIAIOSFODNN7EXAMPLE";
GetOpenAPIClient.AmazonOptions.SecretKey := "wJalrXUtnFEMI/K7MDENG/bPxrFcYEXAMPLEKEY";
GetOpenAPIClient.GetObject("bucket_name");
```

## PutObject

```
GetOpenAPIClient.AmazonOptions.AccessKey := 'AKIAIOSFODNN7EXAMPLE';
GetOpenAPIClient.AmazonOptions.SecretKey := 'wJalrXUtnFEMI/K7MDENG/bPxrFcYEXAMPLEKEY';
ShowMessage(GetOpenAPIClient.PutObject('bucket_name', 'MyFile.txt', 'payload'));
```

```
GetOpenAPIClient->AmazonOptions->AccessKey = "AKIAIOSFODNN7EXAMPLE";
GetOpenAPIClient->AmazonOptions->SecretKey = "wJalrXUtnFEMI/K7MDENG/bPxrFcYEXAMPLEKEY";
GetOpenAPIClient->PutObject("bucket_name", "MyFile.txt", "payload");
```

```
GetOpenAPIClient.AmazonOptions.AccessKey := "AKIAIOSFODNN7EXAMPLE";  
GetOpenAPIClient.AmazonOptions.SecretKey := "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY";  
GetOpenAPIClient.PutObject("bucket_name", "MyFile.txt", "payload");
```

# OpenAPI | Google Cloud

The **sgcOpenAPI Google Client** (TsgcOpenAPI\_Google\_Client) has its own OpenAPI Client which inherits from [TsgcOpenAPI\\_Client](#).

This component has a property called **GoogleOptions** that includes all required configurations to connect to Google Cloud Servers.

## GoogleOptions

The OpenAPI Google client allows to authenticate using the following methods:

1. **OAuth2 Code**: is interactive, which means requires the intervention of the user.
2. **JWT (service accounts)**: is non-interactive, so can run as a service for example.

The authentication is configured in the property `GoogleOptions.Authentication`, allows the following values:

- **oagaOAuth2**: interactive.
- **oagaJWT**: non-interactive. You can import the settings from a JSON file, using the method **LoadSettingsFromFile**. This method will fill the following properties automatically:
  - **ClientEmail**
  - **PrivateKeyId**
  - **PrivateKey**
  - **ServiceAccountOptions**: when running the client using a service account, the following properties are required:
    - **TokenURI**: by default the value is "https://oauth2.googleapis.com/token", but the value is updated when using the method `LoadSettingsFromFile`.
    - **Subject**: this is the email account when using Domain-Wide delegation
    - **Scopes**: a list of the scopes.

## Most common uses

- **Configuration**
  - [Google Cloud OAuth2](#)
  - [Google Cloud Service Accounts](#)
- **APIs**
  - [Google Cloud PubSub](#)
  - [Google Cloud Calendar](#)

## sgcOpenAPI AWS SDK

Find below a list of the currently available APIs.

- Abusive Experience Report API
- Accelerated Mobile Pages (AMP) URL API
- Access Approval API
- Access Context Manager API
- Ad Exchange Buyer API
- Ad Exchange Buyer API II
- Ad Experience Report API
- Admin SDK API
- AdMob API
- AdSense Host API
- AdSense Management API
- AI Platform Training & Prediction API
- Analytics Reporting API



- Android Device Provisioning Partner API
- Android Management API
- API Discovery Service
- API Gateway API
- API Keys API
- Apigee API
- App Engine Admin API
- Apps Script API
- Area120 Tables API
- Artifact Registry API
- Assured Workloads API
- Authorized Buyers Marketplace API
- Backup for GKE API
- Bare Metal Solution API
- BigQuery API
- BigQuery Connection API
- BigQuery Data Transfer API
- BigQuery Reservation API
- Binary Authorization API
- Blogger API v3
- Books API
- Calendar API
- Campaign Manager 360 API
- Certificate Authority API
- Certificate Manager API
- Chrome Management API
- Chrome Policy API
- Chrome UX Report API
- Chrome Verified Access API
- Cloud Asset API
- Cloud AutoML API
- Cloud Bigtable Admin API
- Cloud Billing API
- Cloud Billing Budget API
- Cloud Build API
- Cloud Channel API
- Cloud Composer API
- Cloud Data Fusion API
- Cloud Data Loss Prevention (DLP) API
- Cloud Dataplex API
- Cloud Dataproc API
- Cloud Datastore API
- Cloud Debugger API
- Cloud Deployment Manager V2 API
- Cloud DNS API
- Cloud Document AI API
- Cloud Domains API
- Cloud Filestore API
- Cloud Firestore API
- Cloud Functions API
- Cloud Healthcare API
- Cloud Identity API
- Cloud Identity-Aware Proxy API
- Cloud IDS API
- Cloud IoT API
- Cloud Key Management Service (KMS) API
- Cloud Life Sciences API
- Cloud Logging API
- Cloud Memorystore for Memcached API
- Cloud Monitoring API
- Cloud Natural Language API
- Cloud OS Login API
- Cloud Private Catalog
- Cloud Private Catalog Producer
- Cloud Pub/Sub API

- Cloud Resource Manager API
- Cloud Run Admin API
- Cloud Runtime Configuration API
- Cloud Scheduler API
- Cloud Search API
- Cloud Shell API
- Cloud Source Repositories API
- Cloud Spanner API
- Cloud Speech-to-Text API
- Cloud SQL Admin API
- Cloud Storage for Firebase API
- Cloud Storage JSON API
- Cloud Talent Solution API
- Cloud Tasks API
- Cloud Testing API
- Cloud Text-to-Speech API
- Cloud Tool Results API
- Cloud TPU API
- Cloud Trace API
- Cloud Translation API
- Cloud Video Intelligence API
- Cloud Vision API
- Compute Engine API
- Connectors API
- Contact Center AI Insights API
- Container Analysis API
- Content API for Shopping
- Custom Search API
- Data Labeling API
- Data pipelines API
- Database Migration API
- Dataflow API
- Dataproc Metastore API
- Datastream API
- Dialogflow API
- Digital Asset Links API
- Display & Video 360 API
- Domains RDAP API
- DoubleClick Bid Manager API
- Drive Activity API
- Drive API
- Enterprise License Manager API
- Error Reporting API
- Essential Contacts API
- Eventarc API
- Fact Check Tools API
- Firebase App Check API
- Firebase Cloud Messaging API
- Firebase Cloud Messaging Data API
- Firebase Dynamic Links API
- Firebase Hosting API
- Firebase Management API
- Firebase ML API
- Firebase Realtime Database Management API
- Firebase Rules API
- Fitness API
- Game Services API
- Genomics API
- GKE Hub API
- Gmail API
- Gmail Postmaster Tools API
- Google Analytics Admin API
- Google Analytics API
- Google Analytics Data API
- Google Chat API

- Google Civic Information API
- Google Classroom API
- Google Cloud Data Catalog API
- Google Cloud Deploy API
- Google Cloud Memorystore for Redis API
- Google Cloud Support API
- Google Docs API
- Google Forms API
- Google Identity Toolkit API
- Google Keep API
- Google Mirror
- Google My Business API
- Google OAuth2 API
- Google Pay Passes API
- Google Play Android Developer API
- Google Play Custom App Publishing API
- Google Play Developer Reporting API
- Google Play EMM API
- Google Play Game Management
- Google Play Game Services
- Google Play Game Services Publishing API
- Google Play Integrity API
- Google Search Console API
- Google Sheets API
- Google Site Verification API
- Google Slides API
- Google Vault API
- Google Workspace Alert Center API
- Google Workspace Reseller API
- Google+ API
- Groups Migration API
- Groups Settings API
- HomeGraph API
- IAM Service Account Credentials API
- Idea Hub API
- Identity and Access Management (IAM) API
- Indexing API
- Knowledge Graph Search API
- Kubernetes Engine API
- Library Agent API
- Local Services API
- Managed Service for Microsoft Active Directory API
- Manufacturer Center API
- My Business Account Management API
- My Business Business Calls API
- My Business Business Information API
- My Business Lodging API
- My Business Notifications API
- My Business Place Actions API
- My Business Q&A API
- My Business Verifications API
- Network Connectivity API
- Network Management API
- Network Security API
- Network Services API
- Notebooks API
- On-Demand Scanning API
- Organization Policy API
- OS Config API
- PageSpeed Insights API
- Payments Reseller Subscription API
- People API
- Perspective Comment Analyzer API
- Playable Locations API
- Policy Analyzer API

- Policy Simulator API
- Policy Troubleshooter API
- Poly API
- Proximity Beacon API
- Pub/Sub Lite API
- Real-time Bidding API
- reCAPTCHA Enterprise API
- Recommendations AI (Beta)
- Recommender API
- Remote Build Execution API
- Replica Pool
- Resource Settings API
- Retail API
- Safe Browsing API
- SAS Portal API
- SAS Portal API (Testing)
- Search Ads 360 API
- Search Console API
- Secret Manager API
- Security Command Center API
- Security Token Service API
- Semantic Tile API
- Service Broker
- Service Consumer Management API
- Service Control API
- Service Directory API
- Service Management API
- Service Networking API
- Service Usage API
- Smart Device Management API
- Stackdriver Profiler API
- Storage Transfer API
- Street View Publish API
- Tag Manager API
- Tasks API
- Traffic Director API
- Transcoder API
- Version History API
- VM Migration API
- Web Fonts Developer API
- Web Risk API
- Web Security Scanner API
- Workflow Executions API
- Workflows API
- YouTube Analytics API
- YouTube Data API v3
- YouTube Reporting API

# OpenAPI Google Cloud | OAuth2

In order to use the OpenAPI Google Cloud components and Authenticate using OAuth2, first you must obtain the OAuth2 Key from Google Cloud.

Find below the steps to get Google OAuth2 Keys and how configure in our PubSub sample application.

First **login** to your **Google Cloud Account** and use an existing project or create a new one.

After that, go to **Credentials** menu and press the button **CREATE CREDENTIALS**, select the option **OAuth Client ID**.

Select your application type and set a description name

If successful, you will get your **Client Id** and **Client Secret**.


## OAuth client created

The client ID and secret can always be accessed from Credentials in APIs & Services




OAuth is limited to 100 [sensitive scope logins](#) until the [OAuth consent screen](#) is verified. This may require a verification process that can take several days.

Your Client ID

843483347040-ehcskpfsp4180r1bdfoe6mc32e3ncmn0.apps.gc 

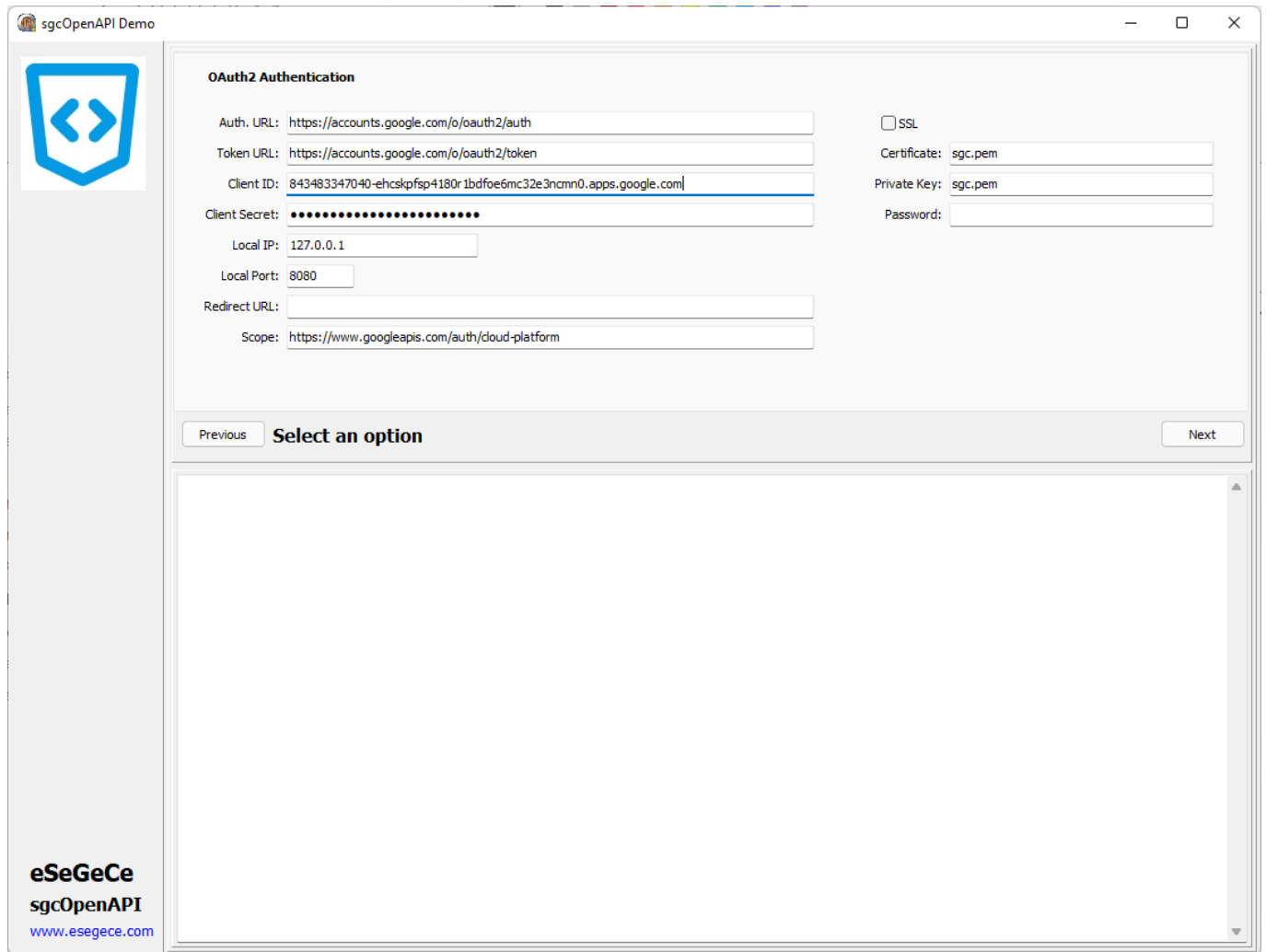
Your Client Secret

pvogD9reE0t9i1L6eR1jE60Z 

OK

**Don't share your OAuth2 data with anyone!**

Now copy to the OpenAPI Google Cloud sample



The image shows a software window titled "sgcOpenAPI Demo". On the left is a sidebar with a blue shield logo containing a white code symbol and the text "eSeGeCe sgcOpenAPI www.esegece.com". The main area is titled "OAuth2 Authentication" and contains several input fields:

- Auth. URL: `https://accounts.google.com/o/oauth2/auth`
- Token URL: `https://accounts.google.com/o/oauth2/token`
- Client ID: `843483347040-ehcspkfp4180r1bdf0e6mc32e3ncmn0.apps.google.com|`
- Client Secret: `.....`
- Local IP: `127.0.0.1`
- Local Port: `8080`
- Redirect URL: (empty)
- Scope: `https://www.googleapis.com/auth/cloud-platform`
- ☐ SSL
- Certificate: `sgc.pem`
- Private Key: `sgc.pem`
- Password: (empty)

At the bottom of the main area, there is a "Previous" button, the text "Select an option", and a "Next" button. Below this is a large empty text area with a scrollbar on the right.

Read more about OAuth2 Configuration.

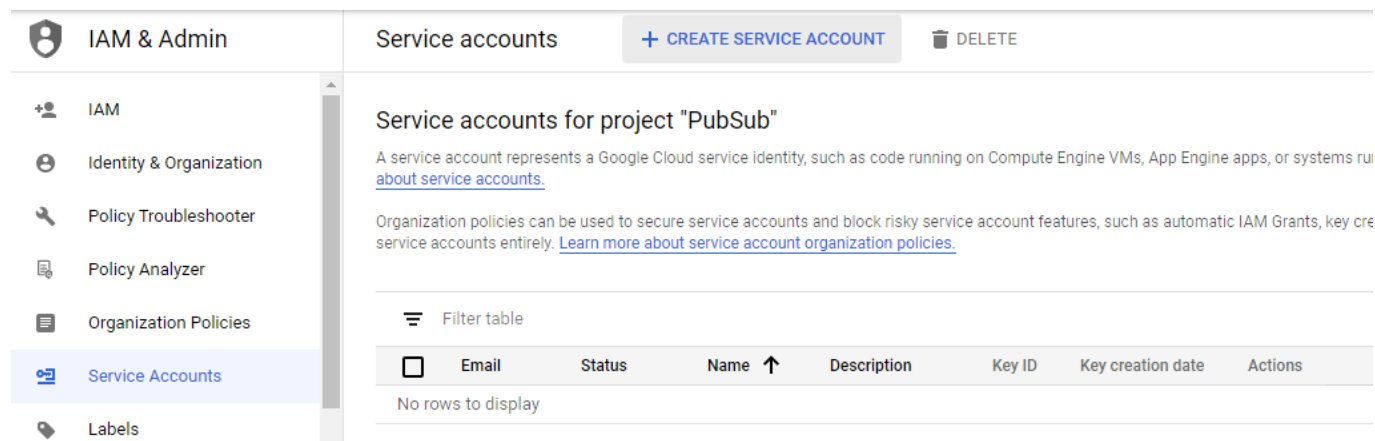
Once you are authenticated, you can **re-authenticate** calling first the method **ClearOAuth2Token** (clear all internal OAuth2 Tokens) and then call any OpenAPI requests, a new web-browser will be shown to re-authenticate against google servers.

# OpenAPI Google Cloud | Service Accounts

In order to use the **OpenAPI Google Cloud components** and **Authenticate using Service Accounts**, first you must obtain the Private Key Certificate from Google Cloud.

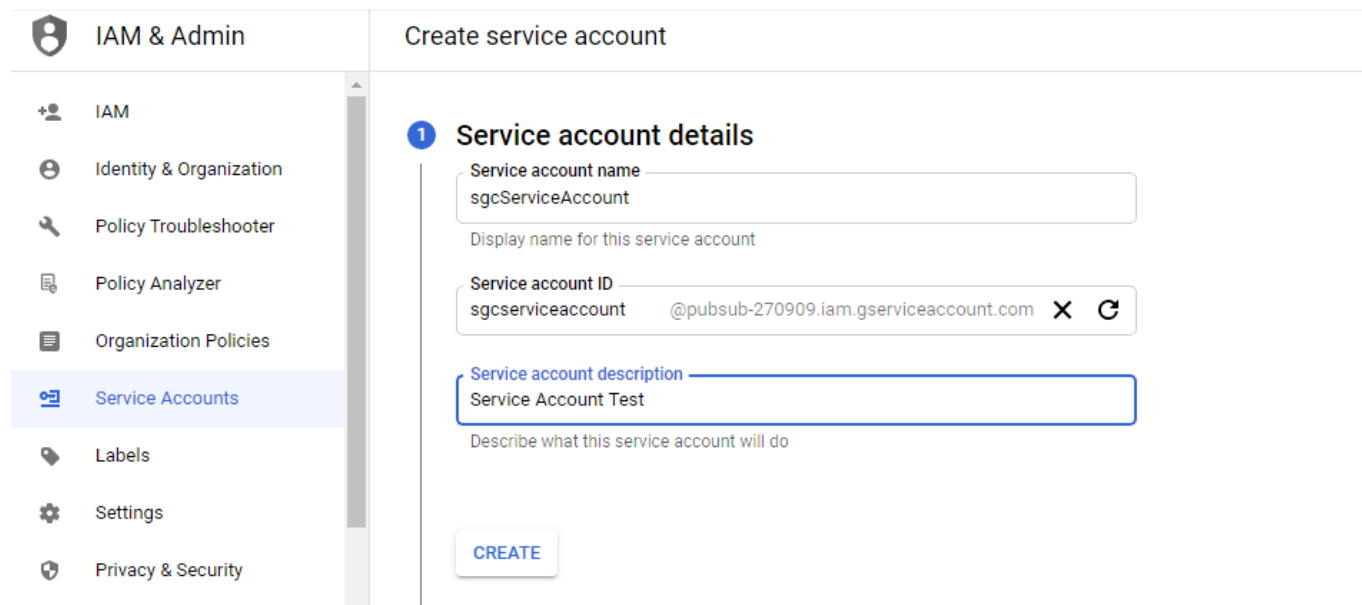
Find below the steps to get Google Private Key Certificate and how configure in our sample application.

First **login** to your **Google Cloud Account** and use an existing project or create a new one.



The screenshot shows the Google Cloud IAM & Admin console. The left sidebar lists navigation options: IAM, Identity & Organization, Policy Troubleshooter, Policy Analyzer, Organization Policies, Service Accounts (highlighted), and Labels. The main content area is titled 'Service accounts' and includes a '+ CREATE SERVICE ACCOUNT' button and a 'DELETE' button. Below this, it says 'Service accounts for project "PubSub"'. A descriptive paragraph explains that a service account represents a Google Cloud service identity. Below the text is a 'Filter table' section with a table header: Email, Status, Name (with an upward arrow), Description, Key ID, Key creation date, and Actions. The table currently shows 'No rows to display'.

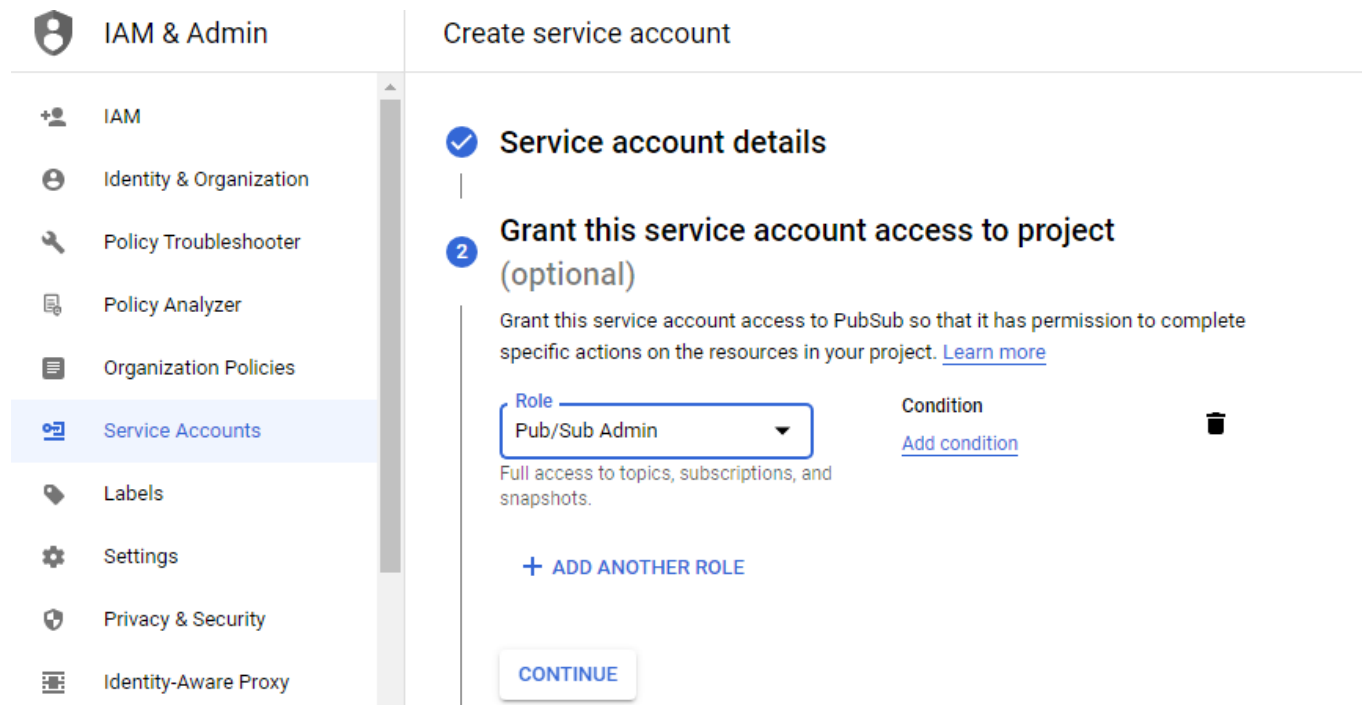
Select **CREATE SERVICE ACCOUNT** and a new page will be shown where you must set the service account name and description



The screenshot shows the 'Create service account' page in the Google Cloud IAM & Admin console. The left sidebar is the same as the previous screenshot. The main content area is titled 'Create service account' and has a step indicator '1 Service account details'. It contains three input fields: 'Service account name' with the value 'sgcServiceAccount', 'Service account ID' with the value 'sgcserviceaccount' and a domain '@pubsub-270909.iam.gserviceaccount.com', and 'Service account description' with the value 'Service Account Test'. A 'CREATE' button is at the bottom.

Then select at least one Role, I select PubSub Admin to allow the client publish and subscribe topics, but you can select other role with less privileges





**IAM & Admin**

- IAM
- Identity & Organization
- Policy Troubleshooter
- Policy Analyzer
- Organization Policies
- Service Accounts**
- Labels
- Settings
- Privacy & Security
- Identity-Aware Proxy

### Create service account

- Service account details
- Grant this service account access to project (optional)**

Grant this service account access to PubSub so that it has permission to complete specific actions on the resources in your project. [Learn more](#)

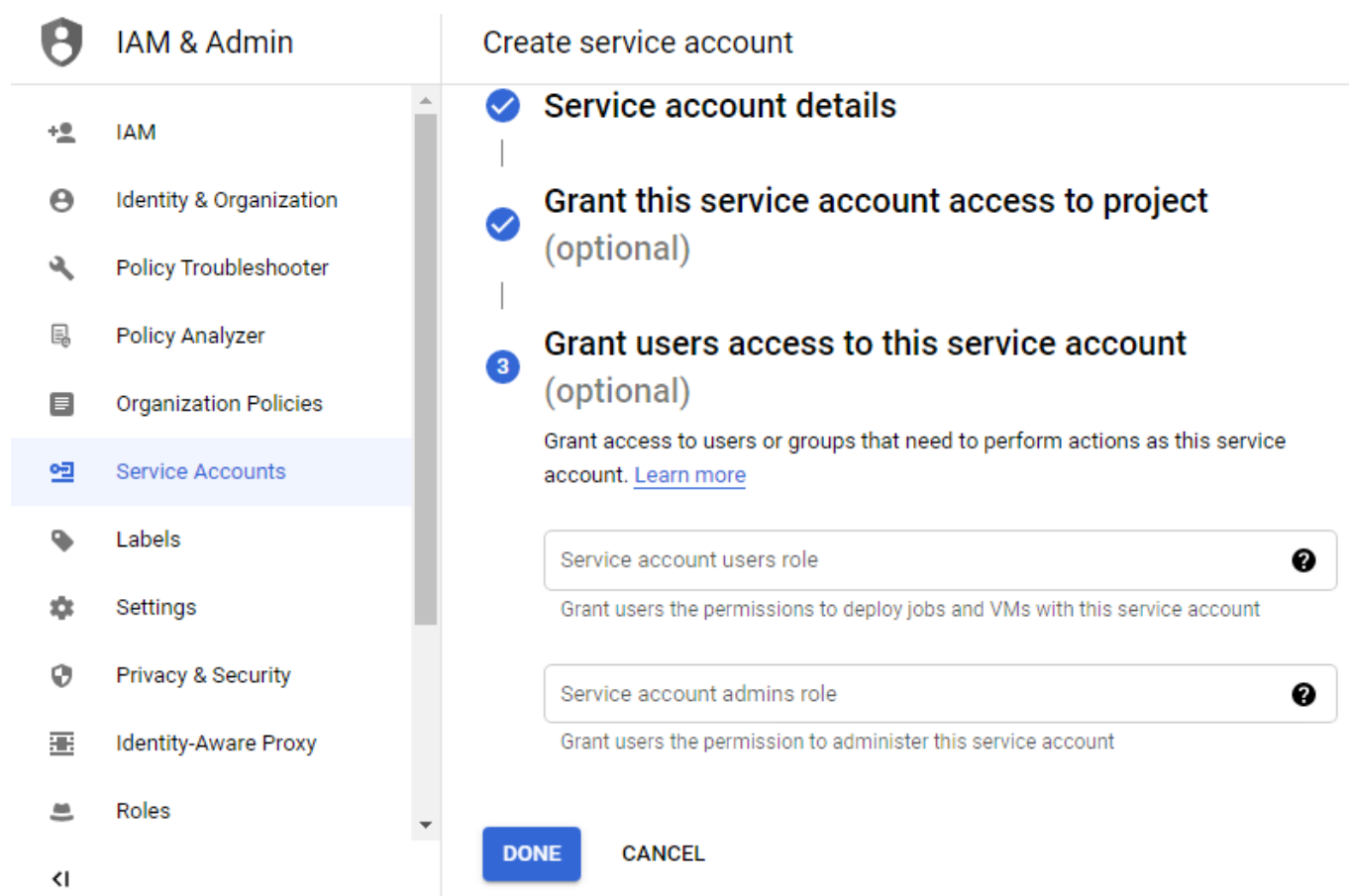
Role: **Pub/Sub Admin** Condition: [Add condition](#)

Full access to topics, subscriptions, and snapshots.

[+ ADD ANOTHER ROLE](#)

**CONTINUE**

Press CONTINUE and finally you can grant access to other users



**IAM & Admin**

- IAM
- Identity & Organization
- Policy Troubleshooter
- Policy Analyzer
- Organization Policies
- Service Accounts**
- Labels
- Settings
- Privacy & Security
- Identity-Aware Proxy
- Roles

### Create service account

- Service account details
- Grant this service account access to project (optional)
- Grant users access to this service account (optional)**

Grant access to users or groups that need to perform actions as this service account. [Learn more](#)

Service account users role ?

Grant users the permissions to deploy jobs and VMs with this service account

Service account admins role ?

Grant users the permission to administer this service account

**DONE** CANCEL

Press DONE when you finish and a new record will be shown

IAM & Admin

- IAM
- Identity & Organization
- Policy Troubleshooter
- Policy Analyzer
- Organization Policies
- Service Accounts**
- Labels


Service accounts [+ CREATE SERVICE ACCOUNT](#) [DELETE](#)

### Service accounts for project "PubSub"

A service account represents a Google Cloud service identity, such as code running on Compute Engine VMs, App Engine apps, or system [about service accounts](#).

Organization policies can be used to secure service accounts and block risky service account features, such as automatic IAM Grants, ke service accounts entirely. [Learn more about service account organization policies](#).

Filter table

<input type="checkbox"/>	Email	Status	Name ↑	Description	Key ID
<input type="checkbox"/>	 sgcserviceaccount@pubsub-270909.iam.gserviceaccount.com	✓	sgcServiceAccount	Service Account Test	No keys



The next step is create a new Key, so select the option Create Key in actions column. Select JSON to download the configuration in JSON format and a new Key will be created

#### Service accounts for project "PubSub"

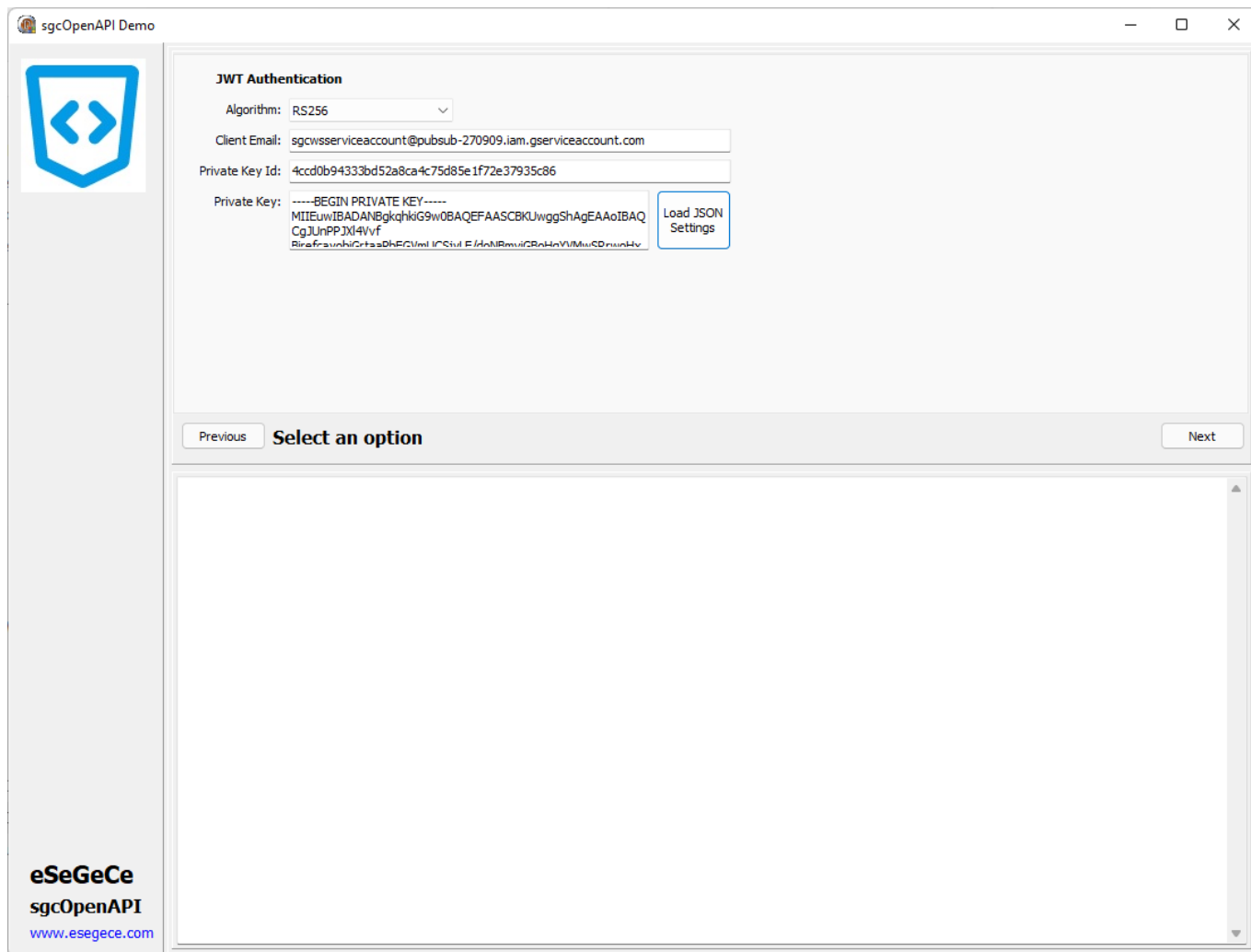
A service account represents a Google Cloud service identity, such as code running on Compute Engine VMs, App Engine apps, or systems running outside Google. [Learn more about service accounts](#).

Organization policies can be used to secure service accounts and block risky service account features, such as automatic IAM Grants, key creation/upload, or the creation of service accounts entirely. [Learn more about service account organization policies](#).

Filter table ?

<input type="checkbox"/>	Email	Status	Name ↑	Description	Key ID	Key creation date	Actions
<input type="checkbox"/>	 sgcserviceaccount@pubsub-270909.iam.gserviceaccount.com	✓	sgcServiceAccount	Service Account Test	425a876f68b8b2a66f2fcc5b2dee8e918b0eb9ab	Dec 20, 2020	

Finally you only need to fill the data provided by google in the OpenAPI PubSub client. You can use **LoadSettings-FromFile** to load the configuration JSON file.



sgcOpenAPI Demo

**JWT Authentication**

Algorithm: RS256

Client Email: sgwsserviceaccount@pubsub-270909.iam.gserviceaccount.com

Private Key Id: 4ccd0b9433bd52a8ca4c75d85e1f72e37935c86

Private Key: -----BEGIN PRIVATE KEY-----  
MIIEUwIBADANBgkqhkiG9w0BAQEFAASCCKUwggShAgEAAoIBAQCgJUnPPJX4Vvf  
BiafraunhiCrtasDhECUwIIRStut FIdnNlRmuifCnHnYUuMuSDnunkV

Load JSON Settings

Previous **Select an option** Next

**eSeGeCe**  
sgcOpenAPI  
[www.esegece.com](http://www.esegece.com)

## Domain-Wide Delegation

If you have a Google Workspace account, an administrator of the organization can authorize an application to access user data on behalf of users in the Google Workspace domain. For example, an application that uses the **Google Calendar API** to add events to the calendars of all users in a Google Workspace domain would use a service account to access the Google Calendar API on behalf of users. Authorizing a service account to access data on behalf of users in a domain is sometimes referred to as "delegating domain-wide authority" to a service account.

To delegate domain-wide authority to a service account, a super administrator of the Google Workspace domain must complete the following steps:

- From your Google Workspace domain's Admin console, go to **Main menu menu > Security > Access and data control > API Controls**.
- In the **Domain wide delegation pane**, select **Manage Domain Wide Delegation**.
- Click **Add new**.
- In the **Client ID** field, enter the service account's **Client ID**. You can find your service account's client ID in the Service accounts page.
- In the **OAuth scopes (comma-delimited)** field, enter the list of scopes that your application should be granted access to. For example, if your application needs domain-wide full access to the Google Drive API and the Google Calendar API, enter: <https://www.googleapis.com/auth/drive>, <https://www.googleapis.com/auth/calendar>.
- Click **Authorize**.

Once you've linked and authorized the workspace account, configure the property `GoogleOptions.ServiceAccountOptions` from the OpenAPI client:

- **Subject:** is the workspace email account linked to the service account. Example: `youremail@domain.com`
- **Scopes:** list of scopes. Example: `https://www.googleapis.com/auth/calendar`

- **TokenURI:** by default is <https://oauth2.googleapis.com/token>.

# OpenAPI Google Cloud | PubSub

Pub/Sub allows services to communicate asynchronously, with latencies on the order of 100 milliseconds.

Pub/Sub is used for streaming analytics and data integration pipelines to ingest and distribute data. It is equally effective as a messaging- oriented middleware for service integration or as a queue to parallelize tasks.

## List Projects by Topic (OAuth2)

```
GetOpenAPIClient.GoogleOptions.Authentication := oagaOAuth2;
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.AuthURL := 'https://accounts.google.com/o/oauth2';
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.TokenURL := 'https://accounts.google.com/o/oauth2/token';
GetOpenAPIClient.Authentication.OAuth2.OAuth2Options.ClientId := 'google client id';
GetOpenAPIClient.Authentication.OAuth2.OAuth2Options.ClientSecret := 'google client secret';
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.IP := '127.0.0.1';
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.Port := 8080;
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.Scope.Text := 'https://www.googleapis.com/auth/pubsub';
GetOpenAPIClient.ListV1TopicsByProject('projects/pubsub-270909');
```

```
GetOpenAPIClient->GoogleOptions->Authentication = oagaOAuth2;
GetOpenAPIClient->Authentication->OAuth2->AuthorizationServerOptions->AuthURL = "https://accounts.google.com/o/oauth2";
GetOpenAPIClient->Authentication->OAuth2->AuthorizationServerOptions->TokenURL = "https://accounts.google.com/o/oauth2/token";
GetOpenAPIClient->Authentication->OAuth2->OAuth2Options->ClientId = "google client id";
GetOpenAPIClient->Authentication->OAuth2->OAuth2Options->ClientSecret = "google client secret";
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->IP = "127.0.0.1";
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->Port = 8080;
GetOpenAPIClient->Authentication->OAuth2->AuthorizationServerOptions->Scope->Text = "https://www.googleapis.com/auth/pubsub";
GetOpenAPIClient->ListV1TopicsByProject("projects/pubsub-270909");
```

```
GetOpenAPIClient.GoogleOptions.Authentication = oagaOAuth2;
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.AuthURL = "https://accounts.google.com/o/oauth2";
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.TokenURL = "https://accounts.google.com/o/oauth2/token";
GetOpenAPIClient.Authentication.OAuth2.OAuth2Options.ClientId = "google client id";
GetOpenAPIClient.Authentication.OAuth2.OAuth2Options.ClientSecret = "google client secret";
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.IP = "127.0.0.1";
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.Port = 8080;
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.Scope = "https://www.googleapis.com/auth/pubsub";
GetOpenAPIClient.ListV1TopicsByProject("projects/pubsub-270909");
```

## List Projects by Topic (Service Accounts)

```
GetOpenAPIClient.GoogleOptions.Authentication := oagaJWT;
GetOpenAPIClient.LoadSettingsFromFile('google.json');
GetOpenAPIClient.ListV1TopicsByProject('projects/pubsub-270909');
```

```
GetOpenAPIClient->GoogleOptions->Authentication = oagaJWT;
GetOpenAPIClient->LoadSettingsFromFile("google.json");
GetOpenAPIClient->ListV1TopicsByProject("projects/pubsub-270909");
```

```
GetOpenAPIClient.GoogleOptions.Authentication = oagaJWT;
GetOpenAPIClient.LoadSettingsFromFile("google.json");
GetOpenAPIClient.ListV1TopicsByProject("projects/pubsub-270909");
```

# OpenAPI Google Cloud Calendar

The Calendar API lets you integrate your app with Google Calendar, creating new ways for you to engage your users.

## List Events By Calendar (OAuth2)

```
GetOpenAPIClient.GoogleOptions.Authentication := oagaOAuth2;
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.AuthURL := 'https://accounts.google.com/o/oauth2';
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.TokenURL := 'https://accounts.google.com/o/oauth2/token';
GetOpenAPIClient.Authentication.OAuth2.OAuth2Options.ClientId := 'google client id';
GetOpenAPIClient.Authentication.OAuth2.OAuth2Options.ClientSecret := 'google client secret';
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.IP := '127.0.0.1';
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.Port := 8080;
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.Scope.Text := 'https://www.googleapis.com/auth/calendar';
GetOpenAPIClient.ListCalendarsEventsByCalendarId('email@mydomain.com', true, 'json', 0, '');
```

```
GetOpenAPIClient->GoogleOptions->Authentication = oagaOAuth2;
GetOpenAPIClient->Authentication->OAuth2->AuthorizationServerOptions->AuthURL = "https://accounts.google.com/o/oauth2";
GetOpenAPIClient->Authentication->OAuth2->AuthorizationServerOptions->TokenURL = "https://accounts.google.com/o/oauth2/token";
GetOpenAPIClient->Authentication->OAuth2->OAuth2Options->ClientId = "google client id";
GetOpenAPIClient->Authentication->OAuth2->OAuth2Options->ClientSecret = "google client secret";
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->IP = "127.0.0.1";
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->Port = 8080;
GetOpenAPIClient->Authentication->OAuth2->AuthorizationServerOptions->Scope->Text = "https://www.googleapis.com/auth/calendar";
GetOpenAPIClient->ListCalendarsEventsByCalendarId("email@mydomain.com", true, "json", 0, "");
```

```
GetOpenAPIClient.GoogleOptions.Authentication = oagaOAuth2;
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.AuthURL = "https://accounts.google.com/o/oauth2";
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.TokenURL = "https://accounts.google.com/o/oauth2/token";
GetOpenAPIClient.Authentication.OAuth2.OAuth2Options.ClientId = "google client id";
GetOpenAPIClient.Authentication.OAuth2.OAuth2Options.ClientSecret = "google client secret";
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.IP = "127.0.0.1";
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.Port = 8080;
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.Scope = "https://www.googleapis.com/auth/calendar";
GetOpenAPIClient.ListCalendarsEventsByCalendarId("email@mydomain.com", true, "json", 0, "");
```

## List Events By Calendar (Service Account)

```
GetOpenAPIClient.GoogleOptions.Authentication := oagaJWT;
GetOpenAPIClient.LoadSettingsFromFile('google.json');
GetOpenAPIClient.Authentication.ServiceAccountOptions.Subject := 'email@mydomain.com';
GetOpenAPIClient.Authentication.ServiceAccountOptions.Scopes.Text := 'https://www.googleapis.com/auth/calendar';
GetOpenAPIClient.ListCalendarsEventsByCalendarId('email@mydomain.com', true, 'json', 0, '');
```

```
GetOpenAPIClient->GoogleOptions->Authentication = oagaJWT;
GetOpenAPIClient->LoadSettingsFromFile("google.json");
GetOpenAPIClient->Authentication->ServiceAccountOptions->Subject = "email@mydomain.com";
GetOpenAPIClient->Authentication->ServiceAccountOptions->Scopes->Text := "https://www.googleapis.com/auth/calendar";
GetOpenAPIClient->ListCalendarsEventsByCalendarId("email@mydomain.com", true, "json", 0, "");
```

```
GetOpenAPIClient.GoogleOptions.Authentication = oagaJWT;
GetOpenAPIClient.LoadSettingsFromFile("google.json");
GetOpenAPIClient.Authentication.ServiceAccountOptions.Subject = "email@mydomain.com";
GetOpenAPIClient.Authentication.ServiceAccountOptions.Scopes.Text := "https://www.googleapis.com/auth/calendar";
GetOpenAPIClient.ListCalendarsEventsByCalendarId("email@mydomain.com", true, "json", 0, "");
```

## Insert Calendar Event

```
GetOpenAPIClient.InsertCalendarsEventsByCalendarId('email@mydomain.com', '{"summary":"Test Event","description":'
```

```
GetOpenAPIClient->InsertCalendarsEventsByCalendarId("email@mydomain.com", "{\"summary\":\"Test Event\",\"descript
```

```
GetOpenAPIClient.InsertCalendarsEventsByCalendarId("email@mydomain.com", "{\"summary\":\"Test Event\",\"descripti
```

# OpenAPI | Microsoft

---

The **sgcOpenAPI Microsoft Client** (TsgcOpenAPI\_Microsoft\_Client) has its own OpenAPI Client which inherits from [TsgcOpenAPI\\_Client](#).

This component has a property called **MicrosoftOptions** that includes all required configurations to connect to Microsoft Servers.

## MicrosoftOptions

The OpenAPI Microsoft client allows to authenticate using the following methods:

1. **OAuth2 Code**: is interactive, which means requires the intervention of the user.
2. **OAuth2 Credentials**: is non-interactive, so can run as a service for example.

The authentication is configured in the property `MicrosoftOptions.Authentication`, allows the following values:

- **oamaOAuth2Code**: interactive.
- **oamaOAuth2Credentials**: non-interactive.

Other properties required by Microsoft are the following:

- **TenantId**: it's a value that identifies your account, you can find in your Microsoft/Azure account.

## Most common uses

- **Configuration**
  - [Microsoft Get Tenant](#)
  - [Microsoft Register Application](#)
  - [Microsoft OAuth2 Code](#)
  - [Microsoft OAuth2 Credentials](#)
- **APIs**
  - [Microsoft Graph](#)

## sgcOpenAPI Microsoft APIs

Find below a list of the currently available APIs.

- AutoSuggest Client
- Computer Vision Client
- Custom Image Search Client
- Custom Search Client
- Custom Vision Prediction Client
- Custom Vision Training Client
- Entity Search Client
- Image Search Client
- Local Search Client
- News Search Client
- Partial Graph API
- Spell Check Client
- Video Search Client
- Visual Search Client
- Web Search Client



## sgcOpenAPI Azure APIs

Find below a list of the currently available APIs.

- API Client
- ACE Provisioning ManagementPartner
- ADHybridHealthService
- AdvisorManagementClient
- Anomaly Detector Client
- Anomaly Finder Client
- ApiManagementClient
- AppConfigurationManagementClient
- Application Insights Data Plane
- ApplicationClient
- ApplicationInsightsManagementClient
- AppPlatformManagementClient
- AppServiceCertificateOrders API Client
- AppServiceEnvironments API Client
- AppServicePlans API Client
- Artifact
- AttestationClient
- AuthorizationManagementClient
- AutomationManagement
- AutomationManagementClient
- Azure Action Groups
- Azure Activity Log Alerts
- Azure Addons Resource Provider
- Azure Alerts Management Service Resource Provider
- Azure Bot Service
- Azure CDN WebApplicationFirewallManagement
- Azure Data Catalog Resource Provider
- Azure Data Lake Storage
- Azure Data Migration Service Resource Provider
- Azure Dedicated HSM Resource Provider
- Azure DevOps
- Azure Enterprise Knowledge Graph Service
- Azure IoT Central
- Azure Location Based Services Resource Provider
- Azure Log Analytics
- Azure Log Analytics - Operations Management
- Azure Log Analytics Query Packs
- Azure Machine Learning Datastore Management Client
- Azure Machine Learning Model Management Service
- Azure Machine Learning Workspaces
- Azure Maps Resource Provider
- Azure Media Services
- Azure Metrics
- Azure Migrate Hub
- Azure Migrate V2
- Azure ML Commitment Plans Management Client
- Azure ML Web Services Management Client
- Azure Monitor Private Link Scopes
- Azure Reservation
- Azure Resource Graph
- Azure Resource Graph Query
- Azure SQL Database
- Azure SQL Database API spec
- Azure SQL Database Backup
- Azure SQL Database Backup Long Term Retention Policy
- Azure SQL Database Datamasking Policies and Rules
- Azure SQL Database disaster recovery configurations
- Azure SQL Database Import/Export spec
- Azure SQL Database replication links
- Azure SQL Server API spec
- Azure SQL Server Backup Long Term Retention Vault

- Azure Stack Azure Bridge Client
- azureactivedirectory
- AzureAnalysisServices
- AzureBridgeAdminClient
- AzureDataManagementClient
- AzureDeploymentManager
- AzureDigitalTwinsManagementClient
- AzureStack Azure Bridge Client
- BackupManagementClient
- BatchAI
- BatchManagement
- BatchService
- BillingManagementClient
- BlockchainManagementClient
- BlueprintClient
- CdnManagementClient
- CertificateRegistrationProvider API Client
- Certificates API Client
- CognitiveServicesManagementClient
- CommerceManagementClient
- Compute Admin Client
- ComputeDiskAdminManagementClient
- ComputeManagementClient
- ComputeManagementConvenienceClient
- Computer Vision
- ConsumptionManagementClient
- ContainerInstanceManagementClient
- ContainerRegistryManagementClient
- ContainerServiceClient
- Content Moderator Client
- Cosmos DB
- CostManagementClient
- Customer Lockbox
- CustomerInsightsManagementClient
- customproviders
- Database Threat Detection Policy APIs
- DataBoxEdgeManagementClient
- DataBoxManagementClient
- DatabricksClient
- DataFactoryManagementClient
- DataLakeAnalyticsAccountManagementClient
- DataLakeAnalyticsCatalogManagementClient
- DataLakeAnalyticsJobManagementClient
- DataLakeStoreAccountManagementClient
- DataLakeStoreFileSystemManagementClient
- DataShareManagementClient
- DeletedWebApps API Client
- DeploymentAdminClient
- DeploymentScriptsClient
- DeviceServices
- DevSpacesManagement
- DevTestLabsClient
- Diagnostics API Client
- DiskResourceProviderClient
- DnsManagementClient
- Domain Services Resource Provider
- DomainRegistrationProvider API Client
- Domains API Client
- Dynamics Telemetry
- Engagement.ManagementClient
- EngagementFabric
- EventGridManagementClient
- EventHub2018PreviewManagementClient
- EventHubManagementClient
- Execution Service

- ExpressRouteCrossConnection REST APIs
- FabricAdminClient
- Face Client
- FeatureClient
- Form Recognizer Client
- FrontDoorManagementClient
- GalleryManagementClient
- Guest Diagnostic Settings
- Guest Diagnostic Settings Association
- GuestConfiguration
- HanaManagementClient
- HDInsightJobManagementClient
- HDInsightManagementClient
- HealthcareApisClient
- HybridComputeManagementClient
- HybridDataManagementClient
- HyperDrive
- InfrastructureInsightsManagementClient
- Ink Recognizer Client
- InstanceMetadataClient
- IntuneResourceManagementClient
- iotDpsClient
- iotHubClient
- IoTSpacesClient
- KeyVaultClient
- KeyVaultManagementClient
- KustoManagementClient
- LogicAppsManagementClient
- LogicManagementClient
- LUIS Authoring Client
- LUIS Programmatic
- Machine Learning Compute Management Client
- Machine Learning Workspaces Management Client
- MaintenanceManagementClient
- ManagedLabsClient
- ManagedNetworkManagementClient
- ManagedServiceIdentityClient
- ManagedServicesClient
- Management Groups
- ManagementLinkClient
- ManagementLockClient
- MariaDBManagementClient
- Marketplace RP Service
- MarketplaceOrdering.Agreements
- MediaServicesManagementClient
- Microsoft Insights
- Microsoft NetApp
- Microsoft Storage Sync
- Microsoft.ResourceHealth
- Microsoft.Support
- MicrosoftSerialConsoleClient
- Mixed Reality
- ML Team Account Management Client
- MonitorManagementClient
- MySQLManagementClient
- NetworkAdminManagementClient
- NetworkExperiments
- NetworkManagementClient
- NotificationHubsManagementClient
- PeeringManagementClient
- Personalizer Client
- PolicyClient
- PolicyEventsClient
- PolicyMetadataClient
- PolicyStatesClient

- PolicyTrackedResourcesClient
- portal
- PostgreSQLManagementClient
- Power BI Embedded Management Client
- PowerBIDedicated
- PrivateDnsManagementClient
- Provider API Client
- QnAMaker Client
- QnAMaker Runtime Client
- Recommendations API Client
- RecoveryServicesBackupClient
- RecoveryServicesClient
- RedisManagementClient
- Relay
- RemediationsClient
- ResourceHealthMetadata API Client
- ResourceManagementClient
- Run History APIs
- RunCommandsClient
- SchedulerManagementClient
- SeaBreezeManagementClient
- SearchIndexClient
- SearchManagementClient
- SearchServiceClient
- Security Center
- Security Insights
- ServerManagement
- Service Fabric Client APIs
- Service Map
- ServiceBusManagementClient
- ServiceFabricManagementClient
- SharedImageGalleryServiceClient
- SignalRManagementClient
- SiteRecoveryManagementClient
- Software Plan RP
- SqlManagementClient
- SqlVirtualMachineManagementClient
- Storage Cache Mgmt Client
- StorageImportExport
- StorageManagementClient
- StorSimple8000SeriesManagementClient
- StorSimpleManagementClient
- StreamAnalyticsManagementClient
- SubscriptionClient
- SubscriptionDefinitionsClient
- SubscriptionsManagementClient
- Text Analytics Client
- TimeSeriesInsightsClient
- TopLevelDomains API Client
- TrafficManagerManagementClient
- Update Management
- UpdateAdminClient
- UsageManagementClient
- VirtualMachineImageTemplate
- VirtualWANAsAServiceManagementClient
- Visual Studio Projects Resource Provider Client
- Visual Studio Resource Provider Client
- VM Insights Onboarding
- VMwareCloudSimple
- WebApplicationFirewallManagement
- WebApps API Client
- WebSite Management Client
- windowsesu
- WorkbookClient
- Workload Monitor

# OpenAPI Microsoft | Tenant

---

To build apps that use the Microsoft identity platform for identity and access management, you need access to an Azure Active Directory (Azure AD) *tenant*. It's in the Azure AD tenant that you register and manage your apps, configure their access to data in Microsoft 365 and other web APIs, and enable features like Conditional Access.

A tenant represents an organization. It's a dedicated instance of Azure AD that an organization or app developer receives at the beginning of a relationship with Microsoft. That relationship could start with signing up for Azure, Microsoft Intune, or Microsoft 365, for example.

Each Azure AD tenant is distinct and separate from other Azure AD tenants. It has its own representation of work and school identities, consumer identities (if it's an Azure AD B2C tenant), and app registrations. An app registration inside your tenant can allow authentications only from accounts within your tenant or all tenants.

## Use an existing Azure AD tenant

To check the tenant:

1. Sign in to the Azure Portal. Use the account you'll use to manage your application.
2. Check the upper-right corner. If you have a tenant, you'll automatically be signed in. You see the tenant name directly under your account name.

If you don't have a tenant associated with your account, you'll see a GUID under your account name. You won't be able to do actions like registering apps until you create an Azure AD tenant.

## Create a new Azure AD tenant

You'll provide the following information to create your new tenant:

- **Organization name**
- **Initial domain** - Initial domain <domainname>.onmicrosoft.com can't be edited or deleted. You can add a customized domain name later.
- **Country or region**

# OpenAPI Microsoft | Register Application

Registering your application establishes a trust relationship between your app and the Microsoft identity platform. The trust is unidirectional: your app trusts the Microsoft identity platform, and not the other way around.

Follow these steps to create the app registration:

1. Sign in to the Azure Portal.
2. If you have access to multiple tenants, use the **Directories + subscriptions** filter in the top menu to switch to the tenant in which you want to register the application.
3. Search for and select **Azure Active Directory**.
4. Under **Manage**, select **App registrations** > **New registration**.
5. Enter a display **Name** for your application. Users of your application might see the display name when they use the app, for example during sign-in. You can change the display name at any time and multiple app registrations can share the same name. The app registration's automatically generated Application (client) ID, not its display name, uniquely identifies your app within the identity platform.
6. Specify who can use the application, sometimes called its *sign-in audience*.
7. Select **Register** to complete the initial app registration

The screenshot shows the 'Register an application' page in the Azure Portal. The browser address bar shows 'https://portal.azure.com'. The page header includes the Microsoft Azure logo, a search bar, and the user's profile 'meganb@contoso.com'. The breadcrumb trail is 'Home > Contoso AD (dev) >'. The main heading is 'Register an application'. Below this, there is a section for '\* Name' with a text input field. A description states: 'The user-facing display name for this application (this can be changed later)'. Below the input field is a section for 'Supported account types' with the question 'Who can use this application or access this API?'. There are four radio button options: 'Accounts in this organizational directory only (Contoso AD (dev) only - Single tenant)' (selected), 'Accounts in any organizational directory (Any Azure AD directory - Multitenant)', 'Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)', and 'Personal Microsoft accounts only'. A link 'Help me choose...' is provided. Below this is a section for 'Redirect URI (optional)' with a description: 'We'll return the authentication response to this URI after successfully authenticating the user. Providing this now is optional and it can be changed later, but a value is required for most authentication scenarios.' There is a dropdown menu set to 'Web' and a text input field containing 'e.g. https://myapp.com/auth'. At the bottom, there is a link 'By proceeding, you agree to the Microsoft Platform Policies' and a blue 'Register' button.

Register an application - Microsoft

https://portal.azure.com

Microsoft Azure Search resources, services, and docs (G+/)

meganb@contoso.com CONTOSO AD (DEV)

Home > Contoso AD (dev) >

## Register an application

\* Name

The user-facing display name for this application (this can be changed later).

Supported account types

Who can use this application or access this API?

☒ Accounts in this organizational directory only (Contoso AD (dev) only - Single tenant)

☐ Accounts in any organizational directory (Any Azure AD directory - Multitenant)

☐ Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)

☐ Personal Microsoft accounts only

[Help me choose...](#)

Redirect URI (optional)

We'll return the authentication response to this URI after successfully authenticating the user. Providing this now is optional and it can be changed later, but a value is required for most authentication scenarios.

Web e.g. https://myapp.com/auth

By proceeding, you agree to the [Microsoft Platform Policies](#)

Register

## Add a redirect URI

A *redirect URI* is the location where the Microsoft identity platform redirects a user's client and sends security tokens after authentication.

In a production web application, for example, the redirect URI is often a public endpoint where your app is running, like `https://contoso.com/auth-response`. During development, it's common to also add the endpoint where you run your app locally, like `https://127.0.0.1/auth-response` or `http://localhost/auth-response`.

This **RedirectURI** will be used later to configure the **sgcOpenAPI Microsoft Client**.

## Add credentials

Credentials are used by confidential client applications that access a web API. Examples of confidential clients are web apps, other web APIs, or service-type and daemon-type applications. Credentials allow your application to authenticate as itself, requiring no interaction from a user at runtime.

You can add both certificates and client secrets (a string) as credentials to your confidential client app registration.

The screenshot shows the Microsoft Azure portal interface. At the top, the header includes the Microsoft Azure logo, a search bar, and user information. The breadcrumb trail indicates the path: Home > Fourth Coffee > Contoso App 1. The main heading is 'Contoso App 1 | Certificates & secrets'. Below this, there's a search bar and a 'Got feedback?' link. The left sidebar contains a navigation menu with sections: Overview, Quickstart, Integration assistant, Manage (with sub-items: Branding, Authentication, Certificates & secrets, Token configuration, API permissions, Expose an API, App roles, Owners, Roles and administrators | Preview, Manifest), and Support + Troubleshooting (with sub-items: Troubleshooting, New support request). The 'Certificates & secrets' item is highlighted with a red box. The main content area shows a description of credentials and a tabbed interface with 'Certificates (0)', 'Client secrets (0)', and 'Federated credentials (0)'. The 'Client secrets' tab is active, displaying a description and a '+ New client secret' button. Below this is a table with columns: Description, Expires, Value, and Secret ID. The table is currently empty, with a message stating 'No client secrets have been created for this application.'

## Add a client secret

Sometimes called an *application password*, a client secret is a string value your app can use in place of a certificate to identify itself.

Client secrets are considered less secure than certificate credentials. Application developers sometimes use client secrets during local app development because of their ease of use. However, you should use certificate credentials for any of your applications that are running in production.

1. In the Azure portal, in **App registrations**, select your application.
2. Select **Certificates & secrets > Client secrets > New client secret**.
3. Add a description for your client secret.
4. Select an expiration for the secret or specify a custom lifetime.
  - Client secret lifetime is limited to two years (24 months) or less. You can't specify a custom lifetime longer than 24 months.
  - Microsoft recommends that you set an expiration value of less than 12 months.
5. Select **Add**.
6. *Record the secret's value* for use in your client application code. This secret value is *never displayed again* after you leave this page.



# OpenAPI Microsoft | OAuth2 Code

Using OAuth2 Code Grant Flow requires interaction with the user to login and get the required privileges.

Once you have the [Tenant Id](#) and the [Credentials](#), you can configure the OAuth2 properties for Code Grant.

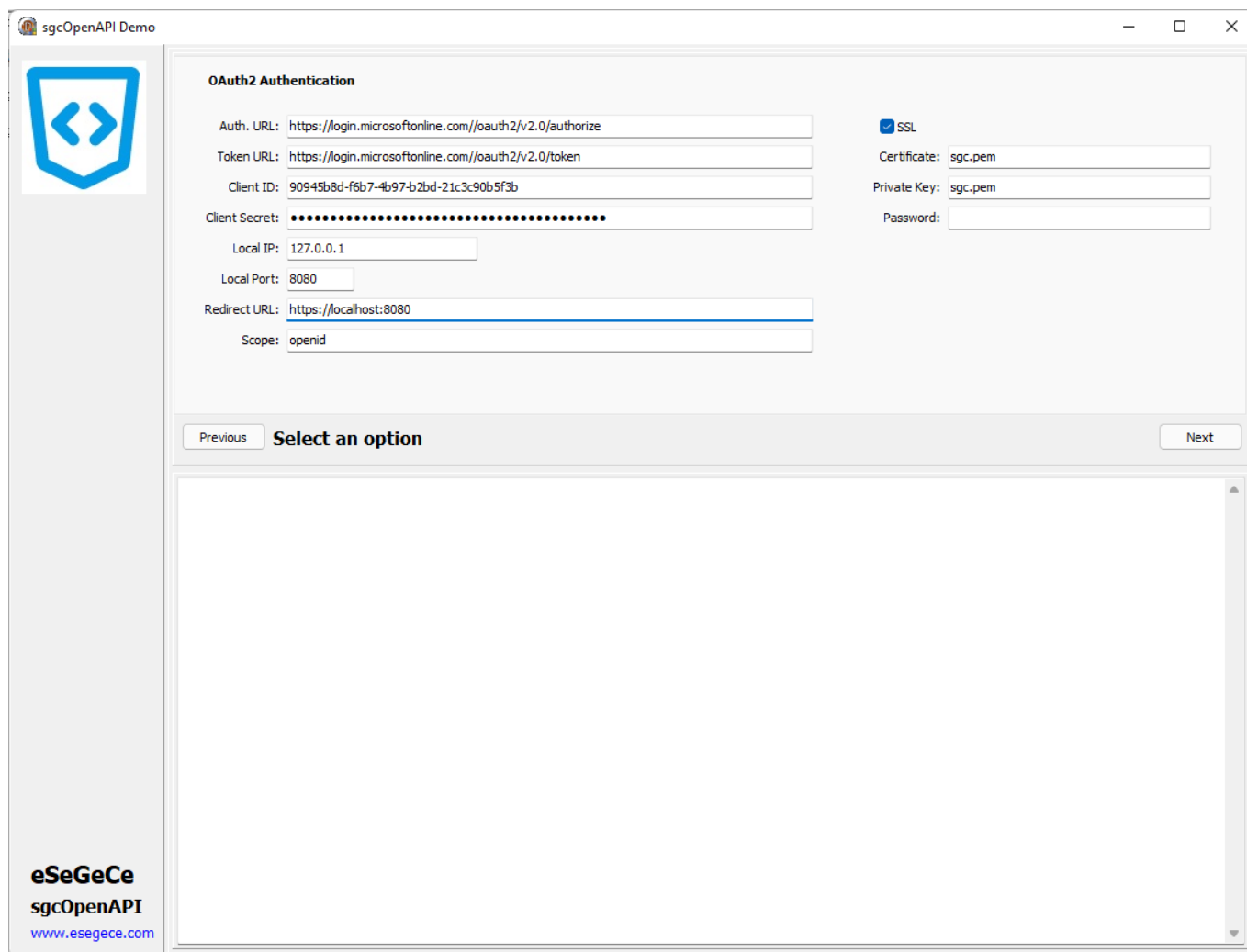
To configure the OpenAPI Client for OAuth2 Code Grant, configure the property **MicrosoftOptions.Authentication** with the following value:

```
GetOpenAPIClient.MicrosoftOptions.Authentication := oamaOAuth2Code;
```

```
GetOpenAPIClient->MicrosoftOptions->Authentication = oamaOAuth2Code;
```

```
GetOpenAPIClient.MicrosoftOptions.Authentication = oamaOAuth2Code;
```

Then you can configure the OAuth2 parameters



The screenshot shows the 'sgcOpenAPI Demo' application window. On the left is a sidebar with a blue shield icon containing a white code symbol and the text 'eSeGeCe sgcOpenAPI www.esegece.com'. The main area is titled 'OAuth2 Authentication' and contains several input fields:

- Auth. URL: `https://login.microsoftonline.com//oauth2/v2.0/authorize`
- Token URL: `https://login.microsoftonline.com//oauth2/v2.0/token`
- Client ID: `90945b8d-f6b7-4b97-b2bd-21c3c90b5f3b`
- Client Secret: A field filled with black dots.
- Local IP: `127.0.0.1`
- Local Port: `8080`
- Redirect URL: `https://localhost:8080`
- Scope: `openid`
- ☒ SSL
- Certificate: `sgc.pem`
- Private Key: `sgc.pem`
- Password: (empty field)

At the bottom, there are 'Previous' and 'Next' buttons flanking a 'Select an option' button. Below this is a large empty text area.

```
GetOpenAPIClient.MicrosoftOptions.Authentication := oamaOAuth2Code;
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.AuthURL := 'https://login.microsoftonline.com/t
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.TokenURL := 'https://login.microsoftonline.com/
GetOpenAPIClient.Authentication.OAuth2.OAuth2Options.ClientId := '90945b8d-f6b7-4b97-b2bd-21c3c90b5f3b';
```

```
GetOpenAPIClient.Authentication.OAuth2.OAuth2Options.ClientSecret := 'client_secret';
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.IP := '127.0.0.1';
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.Port := 8080;
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSL := True;
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.Port := 8080;
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.CertFile := 'sgc.pem';
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.KeyFile := 'sgc.pem';
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.Password := '';
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.RedirectURL := 'https://localhost:8080';
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.Scope.Text := '

```

openid

```
";
```

```
GetOpenAPIClient->MicrosoftOptions->Authentication = oamaOAuth2Code;
GetOpenAPIClient->Authentication->OAuth2->AuthorizationServerOptions->AuthURL = "https://login.microsoftonline.com/te
GetOpenAPIClient->Authentication->OAuth2->AuthorizationServerOptions->TokenURL = "https://login.microsoftonline.c
GetOpenAPIClient->Authentication->OAuth2->OAuth2Options->ClientId = "90945b8d-f6b7-4b97-b2bd-21c3c90b5f3b";
GetOpenAPIClient->Authentication->OAuth2->OAuth2Options->ClientSecret = "client_secret";
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->IP = "127.0.0.1";
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->Port = 8080;
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->SSL = True;
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->SSLOptions->Port = 8080;
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->SSLOptions->CertFile = "sgc.pem";
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->SSLOptions->KeyFile = "sgc.pem";
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->SSLOptions->Password = "";
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->RedirectURL = "https://localhost:8080";
GetOpenAPIClient->Authentication->OAuth2->AuthorizationServerOptions->Scope->Text = "

```

openid

```
";
```

```
GetOpenAPIClient.MicrosoftOptions.Authentication = oamaOAuth2Code;
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.AuthURL = "https://login.microsoftonline.com/te
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.TokenURL = "https://login.microsoftonline.com/te
GetOpenAPIClient.Authentication.OAuth2.OAuth2Options.ClientId = "90945b8d-f6b7-4b97-b2bd-21c3c90b5f3b";
GetOpenAPIClient.Authentication.OAuth2.OAuth2Options.ClientSecret = "client_secret";
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.IP = "127.0.0.1";
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.Port = 8080;
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSL = True;
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.Port = 8080;
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.CertFile = "sgc.pem";
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.KeyFile = "sgc.pem";
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.Password = "";
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.RedirectURL = "https://localhost:8080";
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.Scope = "openid";

```

The Tenant ID must be configured for the Authentication and Token URLs, just replace the correct Tenant Id in the url.

Microsoft only allows the URL **localhost** if you are listening in the Local IP, so set the **redirect URL** with localhost as dns name instead of configure with the IP address.

The **scope** value depends of the API, check the Microsoft / Azure documentation for every API.

The **first time** a request is made, it shows the web-browser asking the user to login to his Microsoft account. If the user already login previously, it will make the HTTP request directly.

# OpenAPI Microsoft | OAuth2 Credentials

Using OAuth2 Code Grant Flow doesn't require interaction with the user, so is suitable for services, daemons... or any application that must run without user interaction.

Once you have the [Tenant Id](#) and the [Credentials](#), you can configure the OAuth2 properties for Code Grant.

To configure the OpenAPI Client for OAuth2 Code Grant, configure the property **MicrosoftOptions.Authentication** with the following value:

```
GetOpenAPIClient.MicrosoftOptions.Authentication := oamaOAuth2Credentials;
```

```
GetOpenAPIClient->MicrosoftOptions->Authentication = oamaOAuth2Credentials;
```

```
GetOpenAPIClient.MicrosoftOptions.Authentication = oamaOAuth2Credentials;
```

The screenshot shows the 'sgcOpenAPI Demo' application window. On the left is a sidebar with the sgcOpenAPI logo and the text 'eSeGeCe sgcOpenAPI www.esegece.com'. The main area is titled 'OAuth2 Authentication' and contains the following fields:

- Auth. URL:
- Token URL:
- Client ID:
- Client Secret:
- Local IP:
- Local Port:
- Redirect URL:
- Scope:
- ☒ SSL
- Certificate:
- Private Key:
- Password:

At the bottom of the main area, there is a 'Previous' button, a 'Select an option' button (which is highlighted), and a 'Next' button.

```
GetOpenAPIClient.MicrosoftOptions.Authentication := oamaOAuth2Credentials;
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.AuthURL := 'https://login.microsoftonline.com/t';
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.TokenURL := 'https://login.microsoftonline.com/t';
GetOpenAPIClient.Authentication.OAuth2.OAuth2Options.ClientId := '90945b8d-f6b7-4b97-b2bd-21c3c90b5f3b';
GetOpenAPIClient.Authentication.OAuth2.OAuth2Options.ClientSecret := 'client_secret';
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.IP := '127.0.0.1';
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.Port := 8080;
```

```

GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSL := True;
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.Port := 8080;
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.CertFile := 'sgc.pem';
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.KeyFile := 'sgc.pem';
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.Password := '';
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.RedirectURL := 'https://localhost:8080';
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.Scope.Text := '

https://graph.microsoft.com/.default

';

```

```

GetOpenAPIClient->MicrosoftOptions->Authentication = oamaOAuth2Credentials;
GetOpenAPIClient->Authentication->OAuth2->AuthorizationServerOptions->AuthURL = "https://login.microsoftonline.com/tenant-id/oauth2/authorize";
GetOpenAPIClient->Authentication->OAuth2->AuthorizationServerOptions->TokenURL = "https://login.microsoftonline.com/tenant-id/oauth2/token";
GetOpenAPIClient->Authentication->OAuth2->OAuth2Options->ClientId = "90945b8d-f6b7-4b97-b2bd-21c3c90b5f3b";
GetOpenAPIClient->Authentication->OAuth2->OAuth2Options->ClientSecret = "client_secret";
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->IP = "127.0.0.1";
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->Port = 8080;
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->SSL = True;
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->SSLOptions->Port = 8080;
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->SSLOptions->CertFile = "sgc.pem";
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->SSLOptions->KeyFile = "sgc.pem";
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->SSLOptions->Password = "";
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->RedirectURL = "https://localhost:8080";
GetOpenAPIClient->Authentication->OAuth2->AuthorizationServerOptions->Scope->Text = "

https://graph.microsoft.com/.default

";

```

```

GetOpenAPIClient.MicrosoftOptions.Authentication = oamaOAuth2Credentials;
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.AuthURL = "https://login.microsoftonline.com/tenant-id/oauth2/authorize";
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.TokenURL = "https://login.microsoftonline.com/tenant-id/oauth2/token";
GetOpenAPIClient.Authentication.OAuth2.OAuth2Options.ClientId = "90945b8d-f6b7-4b97-b2bd-21c3c90b5f3b";
GetOpenAPIClient.Authentication.OAuth2.OAuth2Options.ClientSecret = "client_secret";
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.IP = "127.0.0.1";
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.Port = 8080;
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSL = True;
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.Port = 8080;
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.CertFile = "sgc.pem";
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.KeyFile = "sgc.pem";
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.Password = "";
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.RedirectURL = "https://localhost:8080";
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.Scope = "https://graph.microsoft.com/.default";

```

The Tenant ID must be configured for the Authentication and Token URLs, just replace the correct Tenant Id in the url.

Microsoft only allows the URL **localhost** if you are listening in the Local IP, so set the **redirect URL** with localhost as dns name instead of configure with the IP address.

The **scope** value depends of the API, check the Microsoft / Azure documentation for every API.

OAuth2 credentials doesn't require any user interaction, so no browser will be opened the first HTTP request call.

# OpenAPI Microsoft | Graph

Microsoft Graph exposes REST APIs and client libraries to access data on the following Microsoft cloud services:

- Microsoft 365 core services: Bookings, Calendar, Delve, Excel, Microsoft 365 compliance eDiscovery, Microsoft Search, OneDrive, OneNote, Outlook/Exchange, People (Outlook contacts), Planner, SharePoint, Teams, To Do, Workplace Analytics
- Enterprise Mobility + Security services: Advanced Threat Analytics, Advanced Threat Protection, Azure Active Directory, Identity Manager, and Intune
- Windows services: activities, devices, notifications, Universal Print
- Dynamics 365 Business Central

## Get Current User

```
GetOpenAPIClient.MicrosoftOptions.Authentication := oamaOAuth2Code;
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.AuthURL := 'https://login.microsoftonline.com/';
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.TokenURL := 'https://login.microsoftonline.com/';
GetOpenAPIClient.Authentication.OAuth2.OAuth2Options.ClientId := '90945b8d-f6b7-4b97-b2bd-21c3c90b5f3b';
GetOpenAPIClient.Authentication.OAuth2.OAuth2Options.ClientSecret := 'client_secret';
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.IP := '127.0.0.1';
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.Port := 8080;
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSL := True;
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.Port := 8080;
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.CertFile := 'sgc.pem';
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.KeyFile := 'sgc.pem';
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.Password := '';
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.RedirectURL := 'https://localhost:8080';
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.Scope.Text := 'openid';
GetOpenAPIClient.meuserGetUser();
```

```
GetOpenAPIClient->MicrosoftOptions->Authentication = oamaOAuth2Code;
GetOpenAPIClient->Authentication->OAuth2->AuthorizationServerOptions->AuthURL = "https://login.microsoftonline.com/";
GetOpenAPIClient->Authentication->OAuth2->AuthorizationServerOptions->TokenURL = "https://login.microsoftonline.com/";
GetOpenAPIClient->Authentication->OAuth2->OAuth2Options->ClientId = "90945b8d-f6b7-4b97-b2bd-21c3c90b5f3b";
GetOpenAPIClient->Authentication->OAuth2->OAuth2Options->ClientSecret = "client_secret";
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->IP = "127.0.0.1";
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->Port = 8080;
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->SSL = True;
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->SSLOptions->Port = 8080;
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->SSLOptions->CertFile = "sgc.pem";
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->SSLOptions->KeyFile = "sgc.pem";
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->SSLOptions->Password = "";
GetOpenAPIClient->Authentication->OAuth2->LocalServerOptions->RedirectURL = "https://localhost:8080";
GetOpenAPIClient->Authentication->OAuth2->AuthorizationServerOptions->Scope->Text = "

openid

";
GetOpenAPIClient->meuserGetUser();
```

```
GetOpenAPIClient.MicrosoftOptions.Authentication = oamaOAuth2Code;
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.AuthURL = "https://login.microsoftonline.com/";
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.TokenURL = "https://login.microsoftonline.com/";
GetOpenAPIClient.Authentication.OAuth2.OAuth2Options.ClientId = "90945b8d-f6b7-4b97-b2bd-21c3c90b5f3b";
GetOpenAPIClient.Authentication.OAuth2.OAuth2Options.ClientSecret = "client_secret";
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.IP = "127.0.0.1";
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.Port = 8080;
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSL = True;
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.Port = 8080;
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.CertFile = "sgc.pem";
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.KeyFile = "sgc.pem";
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.SSLOptions.Password = "";
GetOpenAPIClient.Authentication.OAuth2.LocalServerOptions.RedirectURL = "https://localhost:8080";
GetOpenAPIClient.Authentication.OAuth2.AuthorizationServerOptions.Scope = "

openid

";
GetOpenAPIClient.meuserGetUser();
```

# APIs

---

The following APIs have been generated using the eSeGeCe sgOpenAPI Generator and provided for free to any user. The source code of the interface is provided with the trial, so you can see what you can expect if you purchase a license of any of the private apis like Google, Amazon or Microsoft.

API	Description
<a href="#">Geolocation</a>	The Abstract IP Geolocation API takes an IP address and translates it into a location such as an address, timezone, and more.

# OpenAPI | AbstractApi Geolocation

---

## What is the IP Geolocation API?

The Abstract IP Geolocation API takes an IP address and translates it into a location, as well as many other details, such as an address, timezone, and more.

## What are some use cases for the IP Geolocation API?

There are many powerful use cases for IP geolocation API's and data. These include but are not limited to:

- Automatically redirect users to relevant sites or sub sites based on their location
- Automatically detect and displaying a user's location, country, or timezone without requiring them to explicitly make this customization
- Customize the content or experience of a website or app based on the user's location. E.g., showing a user's local weather, tax and VAT rates, currency, news, public holidays, etc.
- Filter out users based on their location, e.g., if you're unable to offer your services to users in a particular country.
- Requiring that a user accepts certain terms as required by local regulations, such as GDPR cookie banners for European Union citizens

## Where get an API Key?

Just register in [abstractapi.com](https://www.abstractapi.com) and you will get an api key for free

<https://www.abstractapi.com/api/ip-geolocation-api>

## Sample Code

The following code returns the info about the IP Address provided

```
ShowMessage(GetOpenAPIClient.Retrieve_the_location_of_an_IP_address('asdfkjlkj32i3j2liwj3es', '88.5.12.4'));
```

```
ShowMessage(GetOpenAPIClient->Retrieve_the_location_of_an_IP_address("asdfkjlkj32i3j2liwj3es", "88.5.12.4"));
```

```
MessageBox.Show(GetOpenAPIClient.Retrieve_the_location_of_an_IP_address("asdfkjlkj32i3j2liwj3es", "88.5.12.4"));
```

# License

---

## eSeGeCe Components End-User License Agreement

eSeGeCe Components ("eSeGeCe") End-User License Agreement ("EULA") is a legal agreement between you (either an individual or a single entity) and the Author of eSeGeCe for all the eSeGeCe components which may include associated software components, media, printed materials, and "online" or electronic documentation ("eSeGeCe components"). By installing, copying, or otherwise using the eSeGeCe components, you agree to be bound by the terms of this EULA. This license agreement represents the entire agreement concerning the program between you and the Author of eSeGeCe, (referred to as "LICENSER"), and it supersedes any prior proposal, representation, or understanding between the parties. If you do not agree to the terms of this EULA, do not install or use the eSeGeCe components.

The eSeGeCe components are protected by copyright laws and international copyright treaties, as well as other intellectual property laws and treaties. The eSeGeCe components are licensed, not sold.

If you want SOURCE CODE you need to pay the registration fee. You must NOT give the license keys and/or the full editions of eSeGeCe (including the DCU editions and Source editions) to any third individuals and/or entities. And you also must NOT use the license keys and/or the full editions of eSeGeCe from any third individuals' and/or entities'.

### 1. GRANT OF LICENSE

The eSeGeCe components are licensed as follows:

#### (a) Installation and Use.

LICENSER grants you the right to install and use copies of the eSeGeCe components on your computer running a validly licensed copy of the operating system for which the eSeGeCe components were designed [e.g., Windows 2000, Windows 2003, Windows XP, Windows ME, Windows Vista, Windows 7, Windows 8, Windows 10].

#### (b) Royalty Free.

You may create commercial applications based on the eSeGeCe components and distribute them with your executables, no royalties required.

#### (c) Modifications (Source editions only).

You may make modifications, enhancements, derivative works and/or extensions to the licensed SOURCE CODE provided to you under the terms set forth in this license agreement.

#### (d) Backup Copies.

You may also make copies of the eSeGeCe components as may be necessary for backup and archival purposes.

### 2. DESCRIPTION OF OTHER RIGHTS AND LIMITATIONS

#### (a) Maintenance of Copyright Notices.

You must not remove or alter any copyright notices on any and all copies of the eSeGeCe components.

#### (b) Distribution.

You may not distribute registered copies of the eSeGeCe components to third parties. Evaluation editions available for download from the eSeGeCe official websites may be freely distributed.

You may create components/ActiveX controls/libraries which include the eSeGeCe components for your applications but you must NOT distribute or publish them to third parties.

#### (c) Prohibition on Distribution of SOURCE CODE (Source editions only).

You must NOT distribute or publish the SOURCE CODE, or any modification, enhancement, derivative works and/or extensions, in SOURCE CODE form to third parties.

You must NOT make any part of the SOURCE CODE be distributed, published, disclosed or otherwise made available to third parties.

#### (d) Prohibition on Reverse Engineering, Decompilation, and Disassembly.

You may not reverse engineer, decompile, or disassemble the eSeGeCe components, except and only to the extent that such activity is expressly permitted by applicable law notwithstanding this limitation.

#### (e) Rental.

You may not rent, lease, or lend the eSeGeCe components.

#### (f) Support Services.

LICENSER may provide you with support services related to the eSeGeCe components ("Support Services"). Any supplemental software code provided to you as part of the Support Services shall be considered part of the eSeGeCe components and subject to the terms and conditions of this EULA.

eSeGeCe is licensed to be used by only one developer at a time. And the technical support will be provided to only one certain developer.

#### (g) Compliance with Applicable Laws.

You must comply with all applicable laws regarding use of the eSeGeCe components.

### 3. TERMINATION

Without prejudice to any other rights, LICENSER may terminate this EULA if you fail to comply with the terms and conditions of this EULA. In such event, you must destroy all copies of the eSeGeCe components in your possession.



## 4. COPYRIGHT

All title, including but not limited to copyrights, in and to the eSeGeCe components and any copies thereof are owned by LICENSER or its suppliers. All title and intellectual property rights in and to the content which may be accessed through use of the eSeGeCe components are the property of the respective content owner and may be protected by applicable copyright or other intellectual property laws and treaties. This EULA grants you no rights to use such content. All rights not expressly granted are reserved by LICENSER.

## 5. NO WARRANTIES

LICENSER expressly disclaims any warranty for the eSeGeCe components. The eSeGeCe components are provided "As Is" without any express or implied warranty of any kind, including but not limited to any warranties of merchantability, non-infringement, or fitness of a particular purpose. LICENSER does not warrant or assume responsibility for the accuracy or completeness of any information, text, graphics, links or other items contained within the eSeGeCe components. LICENSER makes no warranties respecting any harm that may be caused by the transmission of a computer virus, worm, time bomb, logic bomb, or other such computer program. LICENSER further expressly disclaims any warranty or representation to Authorized Users or to any third party.

## 6. LIMITATION OF LIABILITY

In no event shall LICENSER be liable for any damages (including, without limitation, lost profits, business interruption, or lost information) rising out of "Authorized Users" use of or inability to use the eSeGeCe components, even if LICENSER has been advised of the possibility of such damages. In no event will LICENSER be liable for loss of data or for indirect, special, incidental, consequential (including lost profit), or other damages based in contract, tort or otherwise. LICENSER shall have no liability with respect to the content of the eSeGeCe components or any part thereof, including but not limited to errors or omissions contained therein, libel, infringements of rights of publicity, privacy, trademark rights, business interruption, personal injury, and loss of privacy, moral rights or the disclosure of confidential information.

# Index

---

