S&P Global
Market Intelligence

API Developer's Guide

On-Demand Access

Client Support

Email	support.api.mi@spglobal.com
Client Support Website	support.marketplace.spglobal.com

October 15, 2025
V3.1.0

Contents

1 Introduction

The S&P Capital IQ API offers fast, secure access to our extensive data sets on demand, and freedom from the overhead of processing bulk feeds or warehousing data on site.

Using the API, developers can build client library applications that use secure web services to dynamically access our data sets or bypass the client libraries and make requests using web service direct.

1.1 Data Sets

The S&P Capital IQ API provides one of the industry's most comprehensive sets of global data—from a broad range of Fundamental, Industry Specific and Segment Data, Valuations & Pricing, S&P Global Credit Ratings and Research to Reference and Terms & Conditions data. S&P Global Market Intelligence data supports a full spectrum of investor functions.

1.1.1 *S&P Capital IQ Premium Financials*

All instances of standardized financial data on an annual, quarterly, semi-annual, last twelve month and year-to-date basis are captured in S&P Capital IQ Premium Financials. This includes all representations of a financial period including press releases, original filings and all restatements. These files contain the seven industry templates of an S&P Capital IQ presentation (Industrial, Bank, Capital Markets, Financial Services, Insurance, REITs and Utilities) as well as over 5,000 financial, supplemental and industry-specific data items for active and inactive public companies.

1.1.2 *Compustat Financials*

Compustat Financials are comprised of standardized North American fundamental and market data for active and inactive publicly-traded companies. North American Coverage includes annual fundamental data back to 1950, quarterly data and monthly market data back to 1962, and daily market data back to 1984.

1.1.3 *S&P Capital IQ Industry-Specific Data*

Our industry-specific data delivers in-depth operational global data for companies in the following 16 industries: Airline, Bank, Hotels & Gaming, Healthcare Facilities, Homebuilding, Insurance, Internet Media, Managed Care (HMO), Mining, Oil & Gas, Pharmaceutical, Real Estate, Restaurant, Retail, Semiconductor Equipment and Telecom, Cable & Wireless..

1.1.4 *Private Company Financials*

S&P Market Intelligence Private Company Financials offers financial statement and ratio data on over 750,000 companies throughout Europe, North America and Korea. The data provided has history from 2004 forward and is consistent with S&P Capital IQ's high data quality standards.

1.1.5 *Debt and Equity Capital Structure Data*

We provide detailed information about the debt and equity aspects of a company's capital structure. Debt capital structure data covers over 51,000 public and private companies and equity capital structure data on over 81,000 public and select private companies worldwide.

1.1.6 *Bank Regulatory Data*

Our bank regulatory data provides bank regulatory and financial forms data filed with federal banking agencies, including the Federal Deposit Insurance Corporation (FDIC) as well as the Federal Financial Institutions Examination Council (FFIEC). FDIC regulatory filings cover over 20,000 banks and their subsidiaries, delivering information on over 900 unique data items with history available back to 1976. FFIEC regulatory filings include over 3,500 banks and their subsidiaries, delivering information on over 800 unique data items with history available back to 1986.

1.1.7 *Estimates Data*

Estimates data provides standardized, global, real-time financial forecasting measures, such as upgrades / downgrades, target price revisions and market-moving news, as well as estimates based on projections, models, analysis and research from analysts, brokers and the companies themselves. History is available from 1996 for international companies and from 1999 for U.S. companies for annual, quarterly and semiannual time periods. Our new intra-day data gives you the most timely estimates and guidance, with updates and revisions available up to every five minutes.

1.1.8 *Pricing Data*

We provide independent evaluated prices daily on over 3 million global fixed income instruments, CDS price verification and OTC derivative valuations. Through our API service, you also gain access to over 1.8 million actual trade and indicative prices sourced from third-party providers on a daily basis. Gain insight into valuations based on trading data from over 80 exchanges around the world.

1.1.9 *Premium Company Data*

S&P Global Market Intelligence provides clients with access to one of the largest data sets of company information, including core and analytically-enhanced data for public and private companies globally. We combine proprietary research with select third-party content to provide profiles of public and private companies, investment firms, and professionals worldwide. Our Premium Company Data includes the following:

Dividends

The Dividends package provides all dividends paid in a security's history according to date and type.

General Company Details

Our General Company Details data provides profiles of public and private firms worldwide including company contact information, competitors, financial auditors and summary or detailed-level business descriptions.

Key Developments and Events

The Key Developments and Events data sets provide information on more than one million key developments from over 20,000 news sources including press releases, regulatory filings, company web sites, web mining and call transcripts.

Markit Short Interest

The Markit Short Interest data set includes stock loan availability and volume, short interest, and mnemonics reflecting the supply and demand in the securities lending market and the flow of funds driving market prices.

Multiples

S&P Global Market Intelligence Multiples package offers pre-calculated data points to accompany financial statements.

S&P Capital IQ Ownership

Our detailed equity ownership data delivers information for over 55,000 public and private companies worldwide consisting of institutional, mutual fund and insider/individual owners.

Private Equity Data

Our Private Equity data provides private equity firm profiles including investment professionals, limited partners, investment criteria, areas of interest and fund data.

Professionals

S&P Capital IQ Professionals data delivers profiles of professionals including biography, contact data, education, compensation, affiliations and corporate board memberships.

Transactions

Our Transactions data set provides profiles of M&A, private placement, venture capital, PIPE, spin-off, bankruptcies, share buyback programs and equity/debt public offerings. Each transaction has a synopsis, deal size, company participants, advisors coverage and associated filings. The global transaction database provides up to 10 years of history.

1.1.10 *Market, Commodity and Macro Data*

Our market data includes depositary receipts, equity market data, fixed income data, and foreign exchange rates.

Commodity data offers all the information you need on the commodities and futures markets at your fingertips, including details for over 300 securities with history as far back as 1970.

Our macroeconomic data includes coverage of historical and forecasted global economic indicators.

1.1.11 *Reference & Linking Data*

The Reference and Linking Master data set provides access to data at the security and entity level that help firms create customized data mapping systems, review and validate current identifiers and pinpoint the relationships that exist between standard and proprietary identifiers around the world. In addition, this data set provides a listing of data fields that are commonly used to describe a security, including (but not limited to) the security description, the issuer of the security, the currency it is quoted on, the country of issue, the symbols identifying the security and the exchange(s) that the security may be listed on, etc.

1.1.12 *Terms & Conditions Data*

Our Terms and Conditions data delivers near real-time and end of day updates for over 2 million global fixed income securities including corporates, governments, agencies, U.S. municipals and more than 1.4 million U.S. structured finance securities.

1.1.13 *S&P Global Credit Ratings*

Credit Ratings

We offer credit ratings for issuers and issues rated by S&P Global Ratings. Ratings are available for asset classes including Global Issuers, i.e., corporations, financial institutions, insurance companies, utility companies, sovereign entities, international public finance, Structured Finance, and U.S. Public Finance.

Scores & Factors

S&P Global Ratings' Scores & Factors provide transparency into the rating framework and methodology used by S&P Global Rating Analysts, which informs the issuer credit rating ("ICR") for corporations and banks. Amongst scores and assessments such as Business Risk, Financial Risk, Management & Governance for corporations, and Risk Position, Funding & Liquidity, and Capital & Earnings for banks, we also include the Stand-Alone Credit Profile ("SACP"). The SACP is S&P Global Ratings' opinion of an issue's or issuer's creditworthiness, in the absence of extraordinary intervention from its parent or affiliate or related government.

1.1.14 *Credit Related Data*

Our Credit Related data helps you identify weakening credit and fortify your surveillance process for both rated and unrated entities with an array of quantitative tools for credit risk analysis. Credit Related data includes:

Market Derived Signals (MDS)

The Market Derived Signals Model is an industry leading statistical model that evaluates different variables for e.g. CDS spreads, financials, equity prices, etc. to provide an early warning of potential credit changes and capture the market's daily view of a company's perceived risk. Data is available daily for scored entities.

Credit Default Swap (CDS)

CDS, although not credit risk indicators in themselves, are used for monitoring how the market views the credit risk across a wide range of companies and financial institutions and banks. The offering contains the 5-year CDS spread content.

CreditStats

CreditStats data allows you to gain deeper insight into S&P Global rating methodologies with access to financial statement data, as adjusted by our credit analysts for evaluating North American and European corporate and utility issuers.

1.1.15 *Global Funds*

Our Global Funds data includes a comprehensive classification system, commentary and evaluations.

1.1.16 *Industry, Sector & Security Groupings*

We provide extensive industry sector classifications linked at the company level. Industry classifications available in this service include Standard Industrial Classification (SIC) and the North American Industry Classification System (NAIC).

1.1.17 *Cross Reference Services*

The S&P Capital IQ API now offers on-demand access to our robust cross reference capabilities through the Business Entity Cross Reference Service, Global Instruments Cross Reference Service, and Industry Cross Reference Service. These products allow you to link multiple identifiers, identify relationships between instruments and issuers, and obtain a linear view of the corporate hierarchy.

For a complete list of mnemonics by packages, functions and identifiers, and other supporting documentation, please refer to the Reference column in the S&P Capital IQ API section of the Technical Guides on the Marketplace support site.

Business Entity Cross Reference Service

Our Business Entity Cross Reference Service provides immediate cross reference capabilities for 2 million public and private entities with standardized and proprietary identifiers, including the S&P Capital IQ Company ID, S&P Capital IQ Ultimate Parent ID, Global Legal Entity Identifier (GLEI) codes, Rating Agency Identifiers, CUSIP Global Services Issuer Number, and more. With the Business Entity Cross Reference Service you can use S&P Capital IQ's data mapping to systematically update and maintain the multifaceted relationships between entities, companies and issuers.

Global Instruments Cross Reference Service

Our Global Instruments Cross Reference Service resolves the securities numbering problem in a global trading and processing environment with our comprehensive database of security identifiers, cross referencing over 3 million instruments from all over the globe. Now, both middle office and back office subscribers can streamline their operations and create customized data mapping systems for securities and trading items. With Global Instruments Cross Reference Service, you can keep up with the expanding markets and rapid growth of new securities by using this automated service to assist in linking securities to issuers.

Industry Sector Cross Reference Service

The Industry Sector Cross Reference Service provides NAICS and SICS industry sector classifications linked at the company level.

1.1.18 *Credit Analytics*

PD Market Signals Data

PD Model Market Signals provides a point-in-time view of credit risk for public companies based on a statistical model. The model captures equity market sentiment to provide an early warning sign of potential default between financial reporting periods by using stock price as a primary input. The Market Signal PDs are updated daily and cover more than 64,000 public companies globally.

PD Fundamentals Data

PD Model Fundamentals provide an innovative approach to assessing potential default that separates credit risk into two components - financial risk (based on financial ratios), and business risk (based on various systemic risk components, such as country risk). The PDs are calibrated on defaults and are applicable for public and private companies of any size. We pre-score more than 648,000 corporations and banks globally and you can input your own companies for analysis.

CreditModel Data

CreditModel™ is a proprietary suite of statistical models that use financial statement information to generate credit scores that have been calibrated on S&P Global Ratings. This enables you to quickly evaluate the long-term creditworthiness of public and private companies globally by quantitatively approximating what their credit rating would be in the absence of any qualitative and non-public factors. We pre-score more than 54,000 mid- and large-cap financial institutions and corporations with S&P Capital IQ data.

Entity Benchmark Data

Credit Analytic benchmarks leverage S&P Capital IQ's vast prescored database to calculate aggregates by industry, country, and popular S&P Dow Jones Equity Indices to put an entity's absolute measure of risk in relative context, and observe broad trends in credit quality.

Entity Non-Financial Data

Our Entity Non-Financial data provides banking, corporate, industry, country, and sovereign risk scores along with economic, GDP, and population data.

1.1.19 *Investor Activism Data*

The Investor Activism data set includes a time series collection of activities launched by activist shareholders. The motivations of activist shareholders range from financial (increase of shareholder value through changes in corporate policy, financing structure, cost cutting, etc.) to non-financial (disinvestment from particular countries, adoption of environmentally friendly policies, etc.). These activist shareholders use an equity stake in a corporation to put public pressure on its management.

The Investor Activism dataset includes information on the target company, activist investor details, campaign specifics, tactics, objectives, and related transactions. Combined with other S&P Global Market Intelligence datasets, Investor Activism data creates powerful insights into activist shareholder activities.

1.1.20 *Business Relationships Data*

S&P Global Market Intelligence Business Relationships data provides a perspective on how companies are connected. The data allows you to identify dependencies among companies, view upstream suppliers as well as downstream distributors and customers, and understand how a company is positioned to compete.

2 S&P Capital IQ API Authentication

2.1 Credentials

Your credentials will be sent in a welcome email.

These credentials serve as both your Authentication as well as your Authorization credentials.

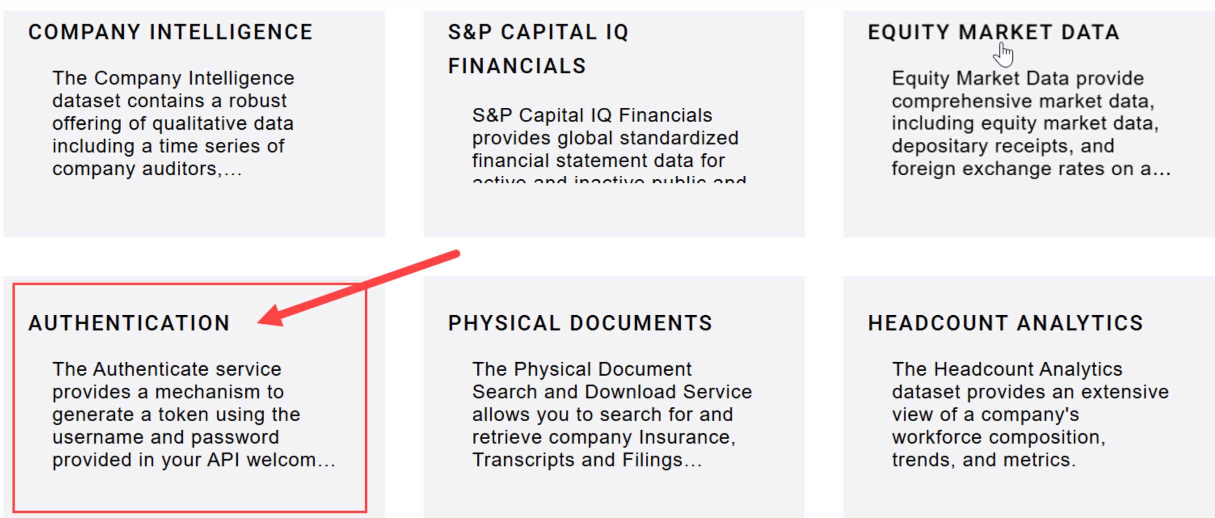
Your username will be provided along with a link to set up your password. Your username and password are confidential login credentials, please treat them as such. Please ascertain to apply your login credentials and track any login credential resets in all environments. In case you retain outdated or incorrect access credentials, you will receive a 423 - Locked Account error message after 5 unsuccessful login attempts.

To request your account to be unlocked or reset your password, reach out to our Enterprise API Support team at **support.api.mi@spglobal.com** and you will receive a password reset link via email.

2.2 Token-Based Authentication in the API Catalog Service

Token-based authentication is a protocol that generates encrypted security tokens. It enables you to verify your identity via a unique encrypted authentication token. A maximum of 4 concurrent threads can be submitted for requesting a token per second from an API account. Otherwise, you will receive a 429 – Too many requests error.

Click on the Authentication tile found on the API Catalog landing page:

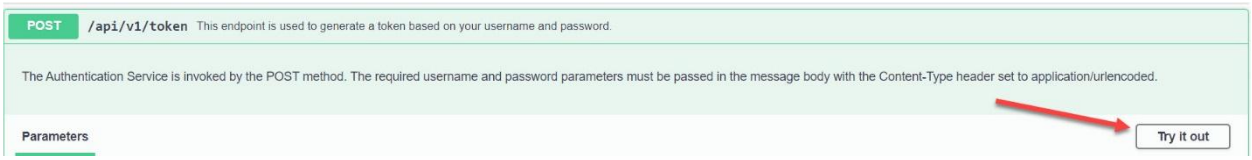


Within the Authentication tile, you will automatically be taken to the “API Definition” section on the left navigation panel. This section contains the Swagger interface for this service.

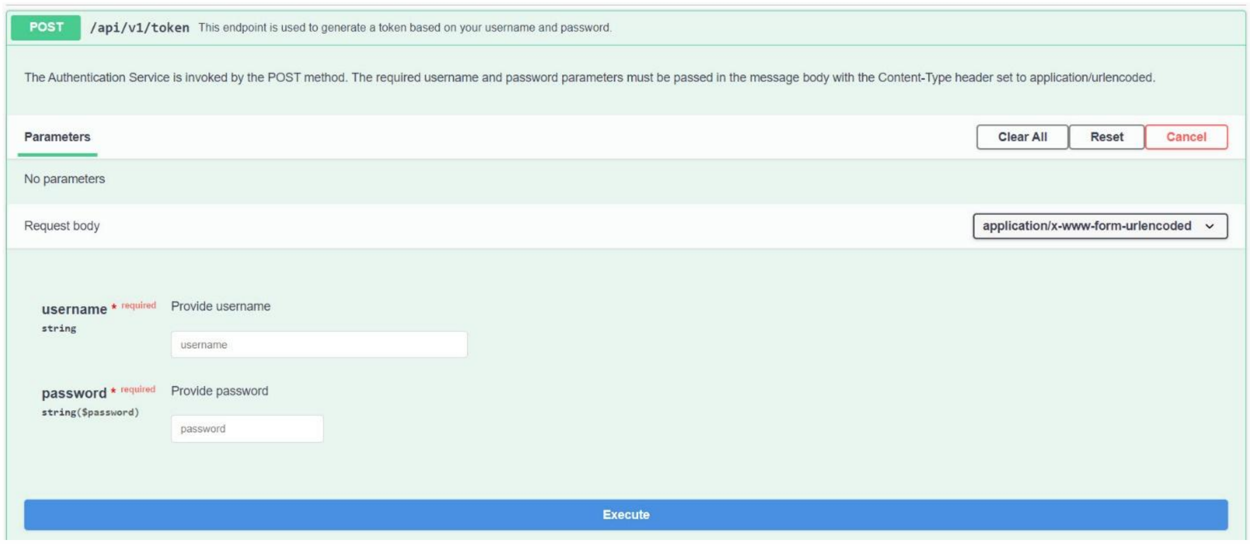
2.2.1 *Access Token*

You will need to generate an access token. The access token is valid for 3600 seconds (60 minutes).

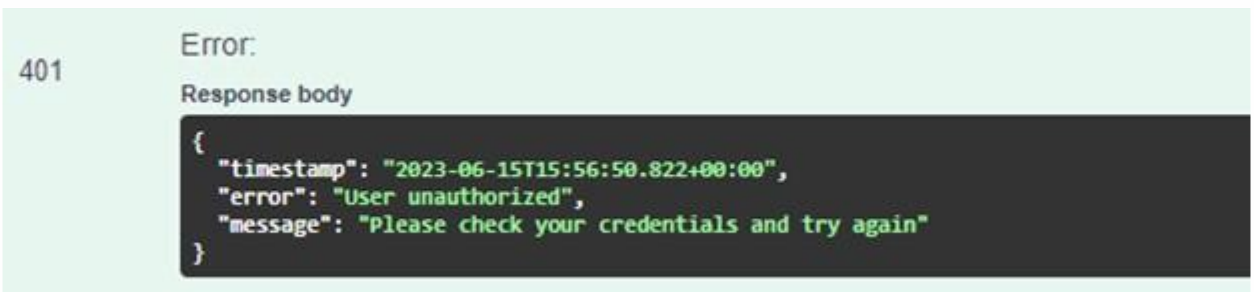
Using the POST method, first click the ‘Try it out’ button.



Then provide your username and password and then click the 'Execute' button.



If your credentials are not authenticated, you will receive a 401 Error.



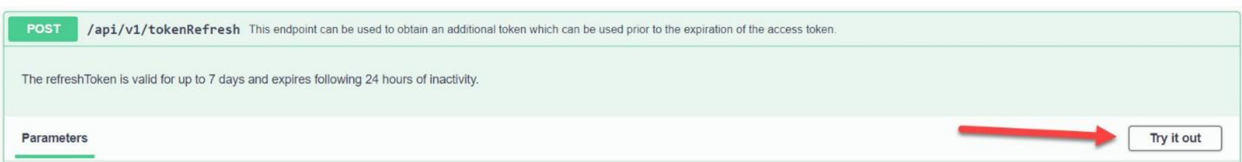
If your credentials have been authenticated, the response will contain the following:

- scope - The S&P Global platform where the token is applicable
- expires_in_seconds - Number of remaining seconds for token expiration
- token_type - Type of the token i.e., bearer
- access_token - Token used for authentication
- refresh_token - Token used for refreshing authentication

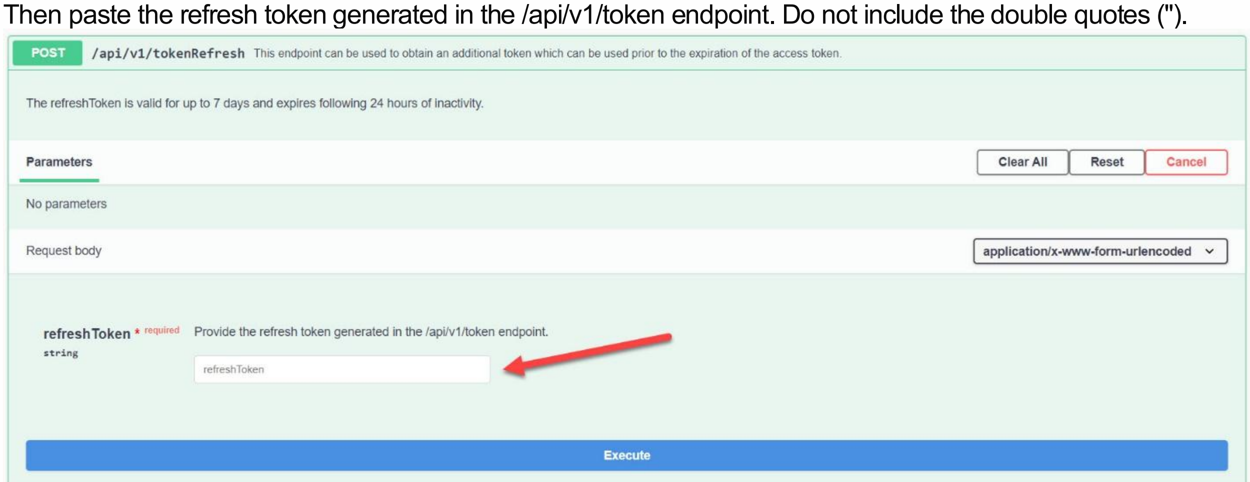
2.2.2 *Refresh Token*

A refresh token is a token that can be used to refresh your access token prior to expiration. The S&P Capital IQ API refreshToken is valid for 7 days. If it is not used within 24 hours, it will expire. If a refresh attempt is made sooner than 3,600 seconds (60 minutes), the access token will be cached.

Using the POST method, click the 'Try it out' button.



Then paste the refresh token generated in the /api/v1/token endpoint. Do not include the double quotes (").



The refreshToken is valid for 7 days. If it is not used within 24 hours, it will expire.

2.3 Token-Based Authentication Using a Rest Tool - POST Method

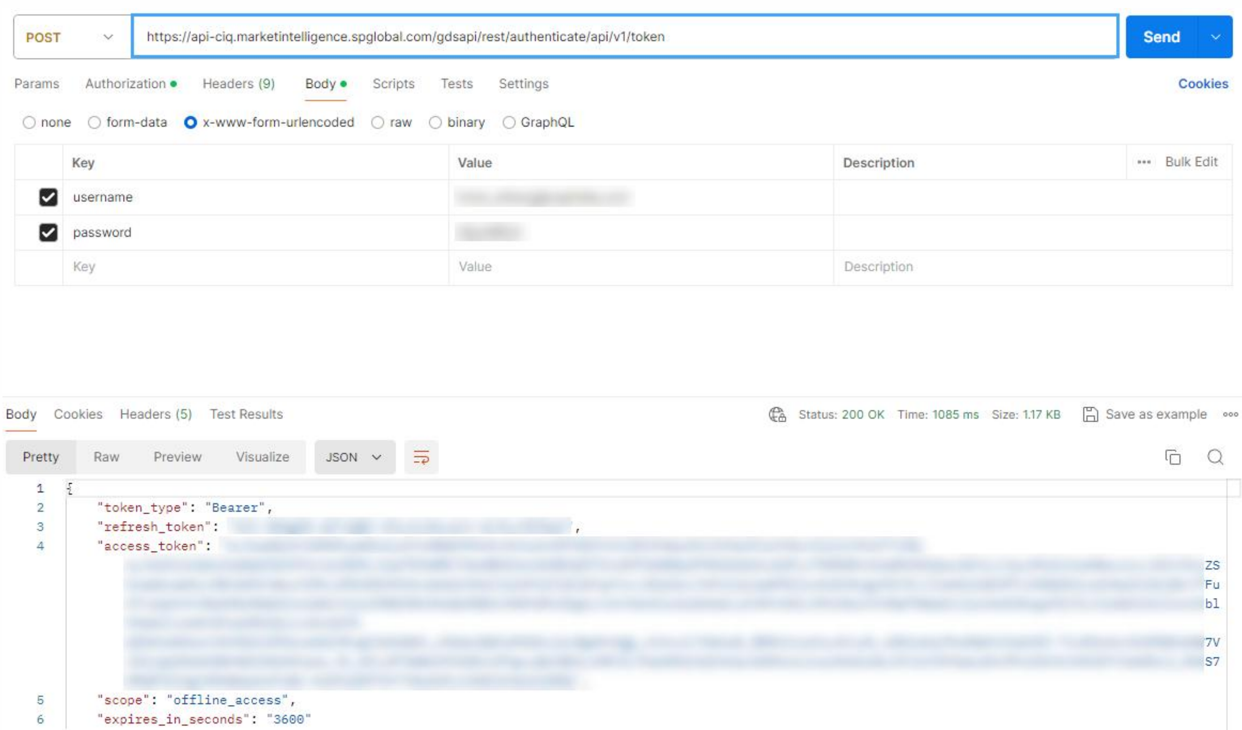
2.3.1 *Access Token*

You will need to generate an access token. The access token is valid for 3600 seconds (60 minutes). Maximum 4 concurrent threads can be submitted for requesting token per second from an API account. Else you will receive a 429 – Too many requests error.

The endpoint to generate an access token is:

https://api-ciq.marketintelligence.spglobal.com/gdsapi/rest/authenticate/api/v1/token

Calls to our endpoint must be authenticated using your username and password. Your username and password parameters must be passed in the message body with the Content-Type header set to application/urlencoded.



This will return an access token that needs to be used with all subsequent requests. The access token must be added in the Request Header as an Authorization Bearer token.

The response will contain the following:

- token_type - Type of the token i.e., bearer
- refresh_token - Token used for refreshing authentication
- access_token - Token used for authentication
- scope – The S&P Global platform where the token is applicable
- expires_in_seconds - Number of remaining seconds for token expiration
-

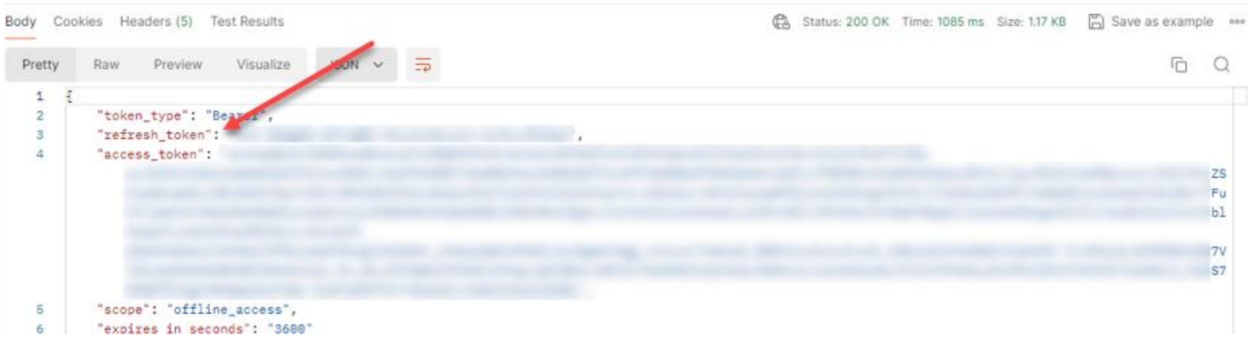
If your credentials are not authenticated, you will receive a 401 Error.

2.3.2 *Refresh Token*

A refresh token is a special token that can be used to refresh your access token prior to expiration. The S&P Capital IQ API refreshToken is valid for 7 days. If it is not used within 24 hours, it will expire. If a refresh attempt is made sooner than 3,600 seconds (60 minutes), the access token will be cached.

The endpoint to generate a refresh token is: **https://api-ciq.marketintelligence.spglobal.com/gdsapi/rest/authenticate/api/v1/tokenRefresh**

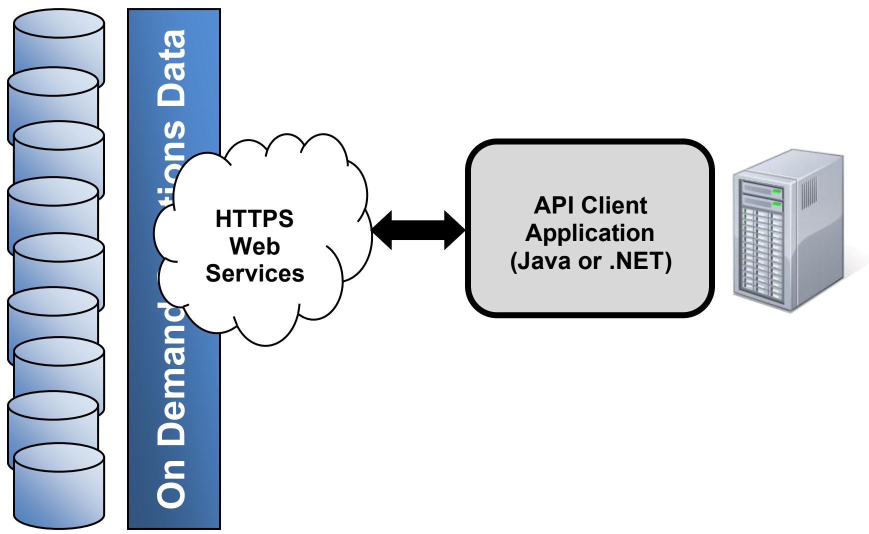
Provide the refresh_token that is provided in the /api/v1/token endpoint:



3 Communication with the Client

3.1 Communication with the Client

Communication with the API client is facilitated by web services, as shown in the following diagram:



All communication between the API client and the platform is through HTTPS protocol using 256-bit encryption on port 443. The client is authenticated using encrypted HTTPS Bearer Authentication, with the access token passed in as part of the header in the request. **NOTE:** It is a mandatory requirement to include Bearer Authentication in the request header when making API calls using the format Authorization = Bearer <token>.

API client libraries are provided for **.NET** and **Java**. In addition to using the API libraries, developers also have the option of communicating directly through the web services via REST/JSON (for more information on these web service standards, see Protocols, Standards and Message Formats).

Static IPs

Static IPs are available for API and can be provided to clients for whitelisting if required. For more information, please contact Enterprise API Support at support.api.mi@spglobal.com.

3.2 Current API Client Library Versions

Library	Version	Download
OPEN JDK 1.8	3.1.1	https://www.support.marketplace.spglobal.com/content/dam/spglobal/mi/en/documents/marketplace/api/client_libraries/spgmi_api_sdk_java_openjdk_1.8_v3.1.1.zip
Open JDK 11	3.1.1	https://www.support.marketplace.spglobal.com/content/dam/spglobal/mi/en/documents/marketplace/api/client_libraries/spgmi_api_sdk_java_openjdk_11_v3.1.1.zip
Open JDK 21	3.1.1	https://www.support.marketplace.spglobal.com/content/dam/spglobal/mi/en/documents/marketplace/api/client_libraries/spgmi_api_sdk_java_openjdk_21_v3.1.1.zip
.NET FRAMEWORK 4.8 *	3.1.1	https://www.support.marketplace.spglobal.com/content/dam/spglobal/mi/en/documents/marketplace/api/client_libraries/spgmi_api_sdk_.net_4.8.0_v3.1.1.zip
.NET CORE 6.0.0 (win-x64) *	3.1.1	https://www.support.marketplace.spglobal.com/content/dam/spglobal/mi/en/documents/marketplace/api/client_libraries/spgmi_api_sdk_.net_core_6.0.0_v3.1.1_win-x64.zip
.NET CORE 6.0.0 (win-arm64) *	3.1.1	https://www.support.marketplace.spglobal.com/content/dam/spglobal/mi/en/documents/marketplace/api/client_libraries/spgmi_api_sdk_.net_core_6.0.0_v3.1.1_win-arm64.zip

*.NET libraries are compatible exclusively with x64-bit processors.

3.3 Supported Frameworks and Language Versions

Software	Supported Version
.NET Framework	4.8
.NET Core	6.0

Software	Minimum Supported Version
Java	1.8
HTTPS	1.1
OpenJDK	1.8

3.4 Test SDKs

Test SDK	Download
.NET 4.8.0	https://www.support.marketplace.spglobal.com/en/api/reference/spgmi_api_test_sdk_.net_4.8.0_v3.1.0.zip
.NET 6.0.0	https://www.support.marketplace.spglobal.com/en/api/reference/sppgmi_api_test_sdk_.net_core_6.0.0_v3.1.0.zip
Open JDK 1.8	https://www.support.marketplace.spglobal.com/en/api/reference/spgmi_api_test_sdk_open_jdk_1.8_and_11_v3.0.0.zip
Open JDK 11	https://www.support.marketplace.spglobal.com/en/api/reference/spgmi_api_test_sdk_open_jdk_1.8_and_11_v3.0.0.zip
Java	https://www.support.marketplace.spglobal.com/content/dam/spglobal/mi/en/documents/marketplace/api/client_libraries/spgmi_api_test_sdk_3.1.0_java.zip

Below NuGet packages are required to be added for Test SDKs:

For .NET Framework 4.8.0, you must add the below NuGet packages:
log4net v3.2.0
NewtonSoft.Json v13.0.3

For .NET Core 6.0, you must add the following NuGet packages:
Microsoft.Extensions.Hosting v7.0.1
log4net v3.2.0
LogNet.Appenders.NetCore v1.0.0.6
NewtonSoft.Json v13.0.3

3.5 Protocols, Standards and Message Formats

Protocol	Website
JSON	http://www.json.org/
REST	http://en.wikipedia.org/wiki/Representational_State_Transfer
WSDL	http://www.w3.org/TR/wsdl
.NET Framework	https://dotnet.microsoft.com/en-us/platform/support/policy/dotnet-framework
.NET Core	https://dotnet.microsoft.com/en-us/platform/support/policy
Java	http://www.java.com
TLS v1.2	http://en.wikipedia.org/wiki/Transport_Layer_Security
Compression	http://en.wikipedia.org/wiki/HTTP_compression

4 Downloading and Referencing the API Libraries

The next two sections outline how to download and reference the API Libraries for .NET and Java.

4.1 API for .NET

Tools You Will Need

Microsoft .NET Framework v4.8 or Core v6.0.

Download the API for .NET

1. For Framework v4.8 go to

   https://www.support.marketplace.spglobal.com/content/dam/spglobal/mi/en/documents/marketplace/api/client_libraries/spgmi_api_sdk_.net_4.8.0_v3.1.1.zip

   https://www.support.marketplace.spglobal.com/content/dam/spglobal/mi/en/documents/marketplace/api/client_libraries/spgmi_api_sdk_.net_core_6.0.0_v3.1.1_win-x64.zip

   or

   https://www.support.marketplace.spglobal.com/content/dam/spglobal/mi/en/documents/marketplace/api/client_libraries/spgmi_api_sdk_.net_core_6.0.0_v3.1.1_win-arm64.zip

2. Your browser will prompt you to save the zip file.
3. Once saved, unzip the API into a folder of your choice.

Files Included in the Download

Included in the zip download are three client DLLs **– SDKServices.dll, SDKWSDLServices.dll** and **SDKWSDLServices.XmlSerializers.dll –** which comprise the primary libraries, as well as several other DLLs that are optional.

SDKServices.dll, SDKWSDLServices.dll and SDKWSDLServices.XmlSerializers.dll include several classes that are not used in requesting data.

One NuGet package is required to be added for Core 6.0

For .NET Core 6.0, you must add the below NuGet package:

Microsoft.Extensions.Hosting v7.0.1

Adding References to the .NET API Libraries

Within Visual Studio, references to the .NET API libraries can be added by right-clicking a given Project within Solution Explorer and adding references. The following example shows creating a new project and adding references to the appropriate API libraries.

1. Within Visual Studio menu, click Menu > New > Project.

2. Select Window or Console Application and press OK.
3. To add the API DLLs to the Project, Open Solution Explorer.
    a. Right-click the newly created Project (in this case **sdkDataSample**) and click **Add References**.
    b. Click the Browse tab. Navigate to the folder where the API libraries were unzipped. Select the SDK DLLs and click OK.
    c. Right-click the Project and go to Properties. Make sure that the Target Framework for the Project is set to .NET Framework v4.8 or .NET Core v6.0.
4. Code can now be added to the project to create a client that requests data.

Section 3.4 holds links to Test SDKs with S&P Global codes for .NET.

4.2 API for Java

Tools You Will Need

1. Java SDK 1.8 or later version
2. A Java development environment is recommended to run the API. Available Java IDEs include:

    Eclipse (http://www.eclipse.org/)

    IntelliJ by JetBrains (http://www.jetbrains.com/index.html)

    Oracle JDeveloper (http://www.oracle.com/technetwork/developer-tools/jdev/overview/index.html)

    Apache Ant (http://ant.apache.org/)

Download the API for OpenJDK 1.8

1. Go to https://www.support.marketplace.spglobal.com/content/dam/spglobal/mi/en/documents/marketplace/api/client_libraries/spgmi_api_sdk_java_openjdk_1.8_v3.1.1.zip

    Your browser will prompt you to save the zip file.
2. Once saved, unzip the API into a folder of your choice.

Download the API for OpenJDK 11

1. Go to https://www.support.marketplace.spglobal.com/content/dam/spglobal/mi/en/documents/marketplace/api/client_libraries/spgmi_api_sdk_java_openjdk_11_v3.1.1.zip

    Your browser will prompt you to save the zip file.
2. Once saved, unzip the API into a folder of your choice.

Download the API for OpenJDK 21

3. Go to

    https://www.support.marketplace.spglobal.com/content/dam/spglobal/mi/en/documents/marketplace/api/client_libraries/spgmi_api_sdk_java_openjdk_21_v3.1.1.zip

Your browser will prompt you to save the zip file.

4. Once saved, unzip the API into a folder of your choice.

Files Included in the Download

Included in the file download are **ciqapi-jdk[1.8 or 11]-sdk-3.0.0.jar** and **serviceschema-1.0.0.jar**, which comprise the primary API classes.

> **Note:** sdkjavaclient.jar and serviceschema.jar include several classes that are not used for data requests. In addition to data requests, the API libraries can be used to listen for Event Driven Alerts using a separate license; for more details, please request the Event Driven Alerts Developer's Guide.

Also included in the download are the following open source jar files that are required to run the API:

org.json-0.0.1.jar

Spring Source (http://www.springsource.org/):
spring-ws-core-3.1.3.jar
spring-xml-3.1.3.jar
spring-oxm-5.3.19.jar
spring-core-5.3.19.jar

spring-context-5.3.19.jar
spring-beans-5.3.19.jar

Apache Software Foundation (http://www.apache.org/)

Logging Services (http://logging.apache.org/log4j/1.2/)

log4j-1.2-api-2.17.2.jar

log4j-api-2.17.2.jar

log4j-core-2.17.2.jar

Commons Project (http://commons.apache.org/)

commons-logging.jar
commons-lang.jar
commons-io.jar
commons-httpclient.jar
commons-configuration-1.2.jar
commons-collections.jar
commons-codec.jar
commons-beanutils.jar

XMLBeans (http://xmlbeans.apache.org/)

xmlbeans-2.3.0.jar

Jackson - High-performance JSON processor (http://jackson.codehaus.org/)

jackson-annotations-2.13.0.jar
jackson-core-2.13.1.jar
jackson-databind-2.13.1.jar

EZMorph(http://ezmorph.sourceforge.net/)

ezmorph-1.0.4.jar

JSON-lib (http://json-lib.sourceforge.net/)

json-lib-2.4-jdk15.jar

Simple Logging Façade for Java (http://www.slf4j.org/dist/)

slf4j-api-1.7.33.jar

slf4j-jdk14-1.7.33.jar

HTTP Libraries:

httpclient-4.1.2.jar

Adding References to the Java API jar files

For development, references to the API jar files can be added to the Java build path, whereby API classes can be referenced in the code. The following example within the Eclipse IDE shows creating a new project and adding the references to the appropriate API jar files within the build path.

1. Within Eclipse, create a new Java project by clicking **New > Project** Assign a Project name (for example **SdkData**) and click **Finish**
2. To add the API jar files to the project build path, within Package Explorer
    a. Right-click the project (in this case **SdkData** > Build Path > Configure Build Path.
    b. Click the **Libraries** tab. Click the **Add External JARS** button.
    c. Navigate to the directory where the Java SDK was unzipped. All of the jar files shown are required to run the API. Select all jar files and click **Open**.
3. The API packages can now be referenced and used to create a client to request data.

5 Requesting Data on Demand

5.1 Retrieving S&P Global Data with Functions

Before coding, it is important to understand S&P Global data and how API Functions are used to retrieve the data.

Each category of data (for example, Company Data or Equity Research) includes various mnemonics pertaining to specific data, which can be queried using the **functions** provided in the API. Some functions return a single data value in time and others return a time series of data values. A description of each of the functions follows in Section 3.2.

Typical **input parameters** for the functions include a mnemonic, a Security/Entity Identifier (including CUSIP, CINS, ISIN, SEDOL, DUNS ID, GVKEY, Ticker, Capital IQ ID etc.), and a date range depending on the function type.

A complete listing of **mnemonics** that includes applicable functions, security/entity identifier types, and additional parameters is available on the Marketplace support site at https://www.support.marketplace.spglobal.com/en/delivery-resources#sec3 (use the provided credentials to log in).

5.2 Available API Functions

Following is a description of each of the functions that can be used to retrieve S&P Global data.

GDSP

This function retrieves a single data point for a point in time value for a mnemonic either current or historical. Default inputs include a Mnemonic and a Security/Entity Identifier.

Example:

Within Terms & Conditions data, this function can be used to retrieve the Coupon Rate (mnemonic COUPON_RATE) for a specific Security Identifier. A single data point is returned for the Security Identifier passed into the function.

GDSPV

This function retrieves an array (a set of values) for the most current availability of content either end of day or intraday. The retrieved values are given in header and data value output classes, and typically represent a schedule, cash flows, or other items that involve multiple data values for the most current end of day. Default inputs include a Mnemonic and a Security/Entity Identifier.

Example:

Within Terms & Conditions data, this function can be used to retrieve the most recent Call Schedule (mnemonic CALL_SCHEDULE) for a specific Security Identifier. Headers and data values are returned for Call Type, Call Date, Call Schedule Price, and other data points that represent the current call schedule for that security.

GDSG

This function retrieves a set of GDSP mnemonics that belong to a specific group. This is a convenience function—instead of making multiple calls using GDSP, this allows you to grab several mnemonics that pertain to the same group of data. The most current end of day values are returned for the set of GDSP mnemonics. Inputs include a Group Mnemonic (there are specific mnemonics available for this function) and a Security/Entity Identifier.

Example:

Within Terms and Conditions data, there are approximately 27 mnemonics that pertain to the Basic Security Description, including COUPON_RATE, COUPON_TYPE, DATED_DATE, and others. Instead of making individual GDSP calls for each of these mnemonics, a GDSG function call can be made with the mnemonic BASIC_SECURITY_DESCRIPTION to retrieve all 27 mnemonics.

GDSHE

This function retrieves historical values for a mnemonic over a range of dates. Data values are returned according to the time series of dates. Default inputs include a Mnemonic, a Security/Entity Identifier, a start date, and an end date.

Example:

Within Pricing data, this function can be used to retrieve the time series of Bid Prices (BID_PRICE) for a given security over a range of dates. In the following example, a range from 08-01-2000 to 08-31-2011 was specified for a given security:

BID_PRICE	PRICE_DATE_D
100.228	05/27/2011
100.3	05/07/2011
100.3	05/06/2011
100.42	02/18/2011
100.67	08/25/2010
100.63	07/26/2010
100.63	07/02/2010
100.63	05/25/2010

GDSHV

This function retrieves an array or set of values over a historical range of dates. Data values are returned according to the time series of dates. Default inputs include a Mnemonic, a Security/Entity Identifier, a start date, and an end date.

Example:

Within S&P Ratings data, the Standard Instrument Credit Rating (INS_STD_CREDIT_RATING) can be retrieved for an entity based on a date range. A table of credit ratings is returned where one header specifies the date.

GDST

This function retrieves historical values for a mnemonic over a range of dates, where a specific frequency can be specified—Annual (A), Quarterly (Q), Monthly (M), Weekly (W), or Daily (D). Header rows for the returned values represent the dates of the periodic values. Default inputs include a Mnemonic, a Security/Entity Identifier, a start date, an end date, and a frequency.

Example:

Within Pricing data, this function can be used to retrieve the Ask Price (ASK_PRICE) of a security based on a date range and a specified frequency of Daily (D) or Weekly (W). The following data shows what is returned for Ask Price for a given security where the frequency is specified as Weekly and the date range is from 1/1/2010 to 1/31/2010:

01/01/2010	01/08/2010	01/15/2010	01/22/2010	01/29/2010
1.62	1.59	1.60	1.69	1.66

5.3 Specialized Data Request Properties

The API Properties Matrix on the S&P Capital IQ API support site outlines the specialized data request properties available on the API. For each property, the associated functions, valid input values, and descriptions are provided.

The API Properties Matrix will help you retrieve API mnemonics with the specific input parameters you desire. You will also have a quick way to view the functions that work with each property. Along with the API Developer's Guide and API Mnemonics Library, the API Properties Matrix will be a valuable asset for your development efforts.

5.4 Requesting Data with Identifiers

With the S&P Capital IQ API, you may use the following identifiers in your requests:

5.4.1 *Securities and Instruments*

A **Security** is a financial instrument that may be traded on one or on multiple exchanges. You can designate the securities you want in your portfolio. Since API Drive can match all identifiers related to a security, you may submit any identifier including:

CUSIP9

CINS9

SEDOL

ISIN

RatingsXpress Instrument ID

5.4.2 *Trading Items*

A **Trading Item** represents the equivalent of a security + the exchange on which it trades. For example, the common stock for Teva Pharmaceuticals trades on both the NYSE and the Tel Aviv exchanges. A tradingItem differentiates between the two exchanges on which the security trades.

CIQ Ticker:Exchange

CIQ Trading Item ID

5.4.3 *Entities and Issuers*

An **Entity** uniquely identifies a legal corporate structure or legal object. An Issuer is a legal organization that issues any type of security. With API Drive, you define the entities and/or issuers you want in your portfolio. Since API Drive can match all identifiers related to an entity or issuer, you may submit any identifier including:

CIQ Company ID	CUSIP6
Compustat GVKEY	CINS6
RatingsXpress Entity ID	CABRE
D&B DUNS ID	CMA Entity ID

5.4.4 *Other Identifiers*

Other identifiers you may select include:

CIQ Professional ID – The S&P Global identifier for a professional

CIQ Person ID – The S&P Global identifier for a person

CIQ Transaction ID – The S&P Global identifier for a transaction

CIQ KeyDev ID – The S&P Global identifier for a key development or future event

CIQ Contributor ID – The S&P Global identifier for a research contributor

CIQ Currency ID – The S&P Global identifier for a currency

CIQ CountryCode – The S&P Global identifier for a country

Ticker – A symbol for a stock ticker *

Universe – This input identifier can be used with applicable mnemonics when you would like to retrieve a universe of entities and/or securities.

Company Professional Search String – This input identifier is applicable for Company and Professional quick search mnemonics. You can input a search string to look for a company or a professional.

* When using Ticker as an identifier, append a colon to the Ticker Symbol (e.g., IBM:)

5.4.5 *Use of Identifier Prefix when passing Requests*

In order to pass a request, you need to apply the identifier prefix abbreviation in addition to an underscore for some identifiers and a numeric combination.

Example:

```
"inputRequests":
[
{"function":"GDSP","identifier":"GV007647","mnemonic":"BECRS_ENTITY_BASE"},
{"function":"GDSP","identifier":"CMA_1466","mnemonic":"BECRS_ENTITY_BASE"},
{"function":"GDSP","identifier":"RX101689","mnemonic":"BECRS_ENTITY_BASE"},
```

Below is a list of prefixes to apply.

Identifier	Description	Prefix
COMPANY_ID	Company Id	IQ
DUNS	D&B business identifier	DB
CUSIP6	Issuer 6 digit	no prefix
CUSIP9	Issue 9 digit	CSP_
ISIN	12 digit	no prefix
		I_(non-IQ metrics)
SEDOL	7 digit	no prefix
RX_ENTITY_ID	RX Entity Id	RX
RX_MATURITY_ID	RX Maturity Id 9 digit	RX
TICKER	TICKER	ticker:exchangecode ticker:
GVKEY	GVKEY	GV
CURRENCY	Currency	C_
DUN_BRADSTREET	Dun & Bradstreet	DB
CABRE_ID	Cabre Id	CB
CMA_ENTITY_ID	CMA Entity Id	CMA_
LEI	LEI	LEI
SNL_ISSUER	SNL Issuer	SNL
TRADING_ITEM_ID	Trading Item Id	IQT
SECURITY_ID	Security Id	IQS
BLOOMBERG_ID	Bloomberg Id	BB_
JAPAN_SICC_NUM	Japan SICC Number	JS_
JAPAN_QUICK_NUM	Japan Quick Number	JQ_
LOANX	LOANX	LX_
EIN	EIN	EIN_
MOODYS_ISSUER	MOODYS_ISSUER	IQ_MOODYS_
VAT_ID	VAT_ID	VAT_
LOCAL_REGISTRY_ID	Local Registry Id	LR_
MARKIT_REDCODE	Markit RedCode 6-Digit	MR_
CIQ Professional ID	The S&P Global identifier for a professional	IQP
CIQ Person ID	The S&P Global identifier for a person	IQ
CIQ Transaction ID	The S&P Global identifier for a transaction	IQTR

CIQ KeyDev ID	The S&P Global identifier for a key development or future event	IQKD
CIQ Contributor ID	The S&P Global identifier for a research contributor	IQC
CIQ Currency ID	The S&P Global identifier for a currency	no prefix
CIQ CountryCode	The S&P Global identifier for a country	no prefix
UNIVERSE	Universe Vectors	no prefix
DCSID	Component identifier	no prefix
PICSCode	Primary Industry Classification Code	PICS

5.5 Mapping Company Names to Company Identifier Using the Kensho Link Service

The KL_PRO_ENTITY_ID_NAME_QUICK_MATCH mnemonic returns a list of company identifiers mapping (CIQ Company ID/MI Institution ID) based on the given keywords and other criteria like city, state etc. It also includes Company Name, City, State, URL and related metadata with score utilizing advanced machine learning.

Sample Request:

```
{
    "inputRequests": [
        {
                    "function": "GDSPV",
                    "identifier": "Facebook",
                    "mnemonic": "KL_PRO_ENTITY_ID_NAME_QUICK_MATCH",
                    "properties": {
                        "UID": "99",
                        "include_response_fields":
"city,state,countryname,year_founded,simple_industry"
                    }
        }
    ]
}
```

- **function***: This value must be GDSPV, which is the only supported function.
- **identifier*:** This value must always be the name of the entity to be searched.
- **mnemonic*:** This value must be KL_PRO_ENTITY_ID_NAME_QUICK_MATCH.
- **uid*:** This is a unique mandatory identifier for each input record. Used in the output to show which linked entity corresponds to which input record. Use any alphabetical, numeric, or alphanumeric values.
- **include_response_fields:** The fields used within this parameter must be comma-separated. Additional details on the following available fields can be found in the Properties Matrix:

- alternatecompanynames
- address
- city
- state
- zipcode
- countryname
- country_iso2
- country_iso3
- url
- phone_number
- year_founded
- response_identifier_type

* Mandatory filters.

Below is the sample response for the above request.

Sample Response:


```
{
    "GDSSDKResponse": [
        {
            "Headers": [
                "UID",
                "SCORE",
                "CIQCOMPANYID",
                "MICOMPANYID",
                "COMPANYNAME",
                "CITY",
                "STATE",
                "COUNTRYNAME",
                "YEAR_FOUNDED",
                "SIMPLE_INDUSTRY"
            ],
            "NumCols": "10",
            "NumRows": "5",
            "Rows": [
                {
                    "Row": [
                        "99",
                        "99.0",
                        "IQ20765463",
                        "MI4144156",
                        "Meta Platforms, Inc.",
                        "Menlo Park",
                        "California",
                        "United States",
```

```
            "2004",
            "Interactive Media and Services"
        ]
    },
    {
        "Row": [
            "99",
            "7.199031859636307",
            "IQ114930719",
            "MI6284253",
            "Facebook UK Ltd",
            "Egham",
            "Surrey",
            "United Kingdom",
            "",
            "Advertising"
        ]
    },
    {
        "Row": [
            "99",
            "6.772675170267329",
            "IQ241782536",
            "MI4438067",
            "Facebook Technologies, LLC",
            "Irvine",
            "California",
            "United States",
            "2012",
            "Technology Hardware, Storage and Peripherals"
        ]
    },
    {
        "Row": [
            "99",
            "6.554496825582841",
            "IQ686646314",
            "MI25114546",
            "Facebook Denmark Aps",
            "København",
            "Capital Region of Denmark",
            "Denmark",
            "2017",
            "Office Services and Supplies"
        ]
    },
    {
        "Row": [
            "99",
            "6.336318480898352",
            "IQ45978686",
            "MI5961117",
            "fbFund, L.P.",
            "Palo Alto",
```

```
                        "California",
                        "United States",
                        "2007",
                        "Asset Management and Custody Banks"
                    ]
                }
            ],
            "Properties": {
                "uid": "99",
                "include_response_fields":
"city,state,countryname,year_founded,simple_industry"
            },
            "ErrMsg": "",
            "Function": "GDSPV",
            "Mnemonic": "KL_PRO_ENTITY_ID_NAME_QUICK_MATCH",
            "Identifier": "FACEBOOK"
        }
    ]
}
```


Optional Output Parameters

See additional details for the fields you may have selected in the include_response_fields in the Properties Matrix:

- alternatecompanynames
- previouscompanynames
- ciqcompanystatus
- ciqcompanytype
- simple_industry
- address
- address1
- address2
- address3
- city
- state
- zipcode
- countryname
- country_iso2
- country_iso3
- url
- email
- phone_number
- tradingcompanyname
- year_founded
- universe

Headers: The Headers show the structure of response. Fields like UID, SCORE, CIQCOMPANYID, MICOMPANYID and COMPANYNAME are provided by default. Any optional field (in this case, `city,state,countryname,year_founded,simple_industry`) included in the "include_response_fields" section will be included here.

UID: UID stands for unique mandatory identifier for each input record. Used in the output to show which linked entity corresponds to which input record. This enables you to easily map a company match to its request when multiple requests are made.

SCORE: The link score represents the confidence score of a particular match between your input fields and the matched company identifier. A maximum of 5 matches will be returned in the response. As per Kensho specification, the first 3 matches will have the highest link score.

6 Responses from API Function Call Requests

6.1 Response Types: OBJECT, JSON and XML

Query responses are returned in native OBJECT format (the default) which returns a List of SDKDataOutput objects that can be interrogated for the response data, as well as in XML format. There is one other response type—JSON—that can be retrieved by invoking the service with the following code:

.NET Client Library

OBJECT

```
var response = _API.InvokeDataService(sdkInput) as List<SDKDataOutput>;
```

XML

```
var response = _API.InvokeDataService(sdkInput,"XML") as XDocument;
```

JSON

```
string response = (string) _API.InvokeDataService service(sdkInput,"JSON");
```

Java Client Library

JSON

```
String responseJson = (String)serviceImpl.invokeDataService(input_request,"JSON");
```

XML

```
String responseXml = (String)serviceImpl.invokeDataService(input_request,"XML");
```

6.2 Common Error Responses

It is possible to receive the following error responses when submitting a request:

Data Unavailable

This means that there is no coverage for the data request (the identifier, mnemonic, and function are correct for input).

Not Applicable

This means that an invalid identifier has been submitted that doesn't match up with any of our identifier formats (at the entity or security level). It could mean you have used an entity level identifier as an input, but made a request for a security level mnemonic (or vice versa).

Invalid Identifier

This means that an invalid identifier was passed into a data request.

Invalid Mnemonic

This means that an incorrect mnemonic was passed into a data request.

Input Arguments Missing

This means that there are one or more input arguments missing.

Function Mismatch

This means that a mnemonic has been passed in that does not work with the specific function it was assigned to (it works with another function).

Not Entitled

This means that the user is not entitled to the specific mnemonic passed in to the data request.

Error Processing Function / Error Processing Request / Invalid Request

These are general errors meaning there is an issue with the input parameters in the request – an input parameter might be missing or some of the input parameters may be invalid.

Full List of Error Responses

For a full list of responses, please refer to the Error Messages Guide under References in the S&P Capital IQ API section of the Technical Guides on the Marketplace support center website https://www.support.marketplace.spglobal.com/en/delivery-resources#sec3.

7 Web Service Direct

7.1 Understanding Web Service Direct

The Web Service Direct service is a HTTPS Web-based REST/JSON (Representational State Transfer / Java Script Object Notation) service. This allows a client or application to integrate directly with the API without deploying any client libraries within the development environment.

The REST/JSON Web Service Direct (POST method) endpoint (versioned) is:

https://api-ciq.marketintelligence.spglobal.com/gdsapi/rest/v3/clientservice.json

BEARER Authentication requires the access token to be passed in the request header.

The JSON request structure and arguments are very similar to the API arguments currently used within the API Client libraries. The actual JSON request should be passed in the request BODY rather than as part of the POST command. This will enable a larger request size to be passed to the platform.

Static IPs

Static IPs are available for API and can be provided to clients for whitelisting if required. For more information, please contact Enterprise API Support at support.api.mi@spglobal.com

8 Usage Service

The S&P Capital IQ API usage service enables you to extract usage statistics for a given day or historical date range. In addition, the usage service provides details about hourly, daily and monthly throttling limits applied to your account. This endpoint is provided to all API users with no additional entitlement.
Our usage service metrics are updated every two hours. An active account is required to access the usage service. You are authenticated using encrypted HTTPS Bearer authentication. See the Token-Based Authentication section for how to generate a token.

8.1 Generating Usage Statistics

Extract information on daily or hourly usage on the specified dates along with hourly, daily and monthly throttling limits for the authorized user with the following endpoint:

Endpoint	https://api-ciq.marketintelligence.spglobal.com/gdsapi/rest/v3/usageservice.json
Method	POST

8.2 Usage Service Sample Request

Below is an example of a Usage Service request (in JSON) which pulls the daily usage metrics of authorized user for the current day. The parameters are listed and described in the Metadata section.

```
{"inputRequests":
[
{ "mnemonic":"USAGE_METRICS"}
]}
```

Parameters

Your Usage Service request is supported by the mnemonic USAGE_METRICS in addition to the below parameters. Choose either startDate and endDate (limit 31 days between both start and end dates) or asofDate. When no date is specified, the usage metrics of the current day will be returned.

Field	Type	Description
startDate	string	The start date from which the usage data is requested. Limit: 31 days between startDate and endDate. Format: mm/dd/yyyy

Field	Type	Description
endDate	string	The end date from which the usage data is requested. Limit: 31 days between startDate and endDate. Format: mm/dd/yyyy
asofDate	string	The day-date for which the usage data is requested. Format: mm/dd/yyyy
frequency	string	Hourly or daily: Hourly: Provides data on an hourly basis i.e. 0-23 hours. Daily: Provides data on a date basis. Hourly frequency applies in UTC timezone. If no frequency is specified, it defaults to daily.

8.3 Usage Service Endpoint Response

Below is an example of a Usage Service response (in JSON). The parameters are listed and described in the Metadata section.

Response :


```
{
  "GDSSDKResponse": [
    {
      "Mnemonic": "USAGE_METRICS",
      "NumCols": 2,
      "NumRows": 1,
      "ErrMsg": "",
      "CacheExpiryTime": "0",
      "Headers": [
        "USAGE_DATE",
        "TOTAL_COUNT"
      ],
      "Rows": [
```

```
          {
            "Row": [
              "2024-06-03",
              "693"
            ]
          }
        ],
        "DailyLimit": "24000",
        "HourlyLimit": "10000",
        "MonthlyLimit": "NA"
      }
    ]
}
```


Parameters

The metadata below will be included for every search sample return in the Usage Service results set.

Field	Type	Description
USAGE_DATE	string	Date for which usage information is returned. Format: yyyy-mm-dd
USAGE_HOUR	string	Hour within a day for which usage information is returned (UTC timezone). Values should be in the range of 0-23.
TOTAL_COUNT	string	Total number of requests placed on hourly or daily basis. This includes both success and failure counts.
DailyLimit	string	User account level throttling limit set for total number of

Field	Type	Description
		requests on daily basis. If no throttling limit is set NA will be returned.
HourlyLimit	string	User account level throttling limit set for total number of requests on hourly basis. If no throttling limit is set NA will be returned.
MonthlyLimit	string	User account level throttling limit set for total number of requests on monthly basis. If no throttling limit is set NA will be returned.

Usage Service Sample Request 2

Below is example 2 of a Usage Service request (in JSON) that pulls the hourly usage metrics of authorized user as of the current day.
All request parameters are listed and described in the Metadata section of Sample Request 1.


```
{"inputRequests":
[
{ "mnemonic":"USAGE_METRICS", "properties" :{"frequency": "hourly"}}
  ]}
```


Usage Service Endpoint Response 2

Below is the Usage Service response (in JSON). The parameters are listed and described in the Metadata section of Sample Response 1.


```
{
   "GDSSDKResponse": [
      {
         "Mnemonic": "USAGE_METRICS",
         "NumCols": 3,
         "NumRows": 5,
         "ErrMsg": "",
         "CacheExpiryTime": "0",
         "Headers": [
            "USAGE_DATE",
            "USAGE_HOUR",
            "TOTAL_COUNT"
         ],
         "Rows": [
            {
               "Row": [
                  "2024-06-03",
                  "01",
                  "8"
               ]
            },
            {
               "Row": [
                  "2024-06-03",
                  "02",
                  "168"
               ]
            },
            {
               "Row": [
                  "2024-06-03",
                  "03",
                  "428"
               ]
            },
            {
               "Row": [
                  "2024-06-03",
                  "04",
                  "1"
               ]
            },
            {
               "Row": [
```

```
              "2024-06-03",
              "05",
              "88"
            ]
          }
        ],
        "DailyLimit": "24000",
        "HourlyLimit": "10000",
        "MonthlyLimit": "NA",
        "Properties": {
          "frequency": "hourly"
        }
      }
  ]
}
```


Usage Service Sample Request 3

Below is example 3 of a Usage Service request (in JSON) that pulls the daily usage information of authorized user for the date range included between start and end dates. All request parameters are listed and described in the Metadata section of Sample Request 1.


```
{"inputRequests":
[{ "mnemonic":"USAGE_METRICS", "properties":{"startdate":"05/20/2024", "enddate": "05/21/2024"}}
 ]}
```


Usage Service Endpoint Response 3

Below is an example of a Usage Service response (in JSON). The parameters are listed and described in the Metadata section of Response sample 1.


```
{
  "GDSSDKResponse": [
    {
      "Mnemonic": "USAGE_METRICS",
      "NumCols": 2,
      "NumRows": 2,
      "ErrMsg": "",
      "CacheExpiryTime": "0",
      "Headers": [
        "USAGE_DATE",
        "TOTAL_COUNT"
      ],
      "Rows": [
        {
          "Row": [
            "2024-05-20",
            "39"
          ]
        },
        {
          "Row": [
            "2024-05-21",
            "504"
          ]
        }
      ],
      "DailyLimit": "24000",
      "HourlyLimit": "10000",
```

```
            "MonthlyLimit": "NA",
            "Properties": {
                "enddate": "05/21/2024",
                "startdate": "05/20/2024"

            }
        }
    ]
}
```

9 Code Examples

For a more detailed discussion of code samples for testing user credentials and requesting data, please refer to the Reference column in the S&P Capital IQ API section on the Marketplace support website https://www.support.marketplace.spglobal.com/en/delivery-resources#sec3.

9.1 Get the Token

To communicate with the S&P Capital IQ API Service, you must get a token. This token will be valid for 60 minutes. For every 60 minutes, the token has to be refreshed using the refreshToken call.

Sample Code to get the token:

```
TokenService tokenService = new TokenServiceImpl();

public object getTokenResponse(string user, string pass)
{
   TokenServiceBaseInput tokenServiceBaseInput = new TokenServiceBaseInput();
   tokenServiceBaseInput.Username = user;
   tokenServiceBaseInput.Password = pass;

   object tokenResponse = tokenService.getToken(tokenServiceBaseInput);
   return tokenResponse;
}
```

The object tokenResponse can be deserialized to TokenResponseVO using below code.

```
string username = "---Username---";
string password = "---Password---";
Object tokenResponse = getTokenResponse(username, password);

TokenResponseVO tokenResponseVO =
      Newtonsoft.Json.JsonConvert.DeserializeObject<TokenResponseVO>(
            tokenResponse.ToString());
```

The tokenResponseVO object will have all the token data including:

access_token – token which can be used to call the ciqapi services
refresh_token – can be used to refresh the token for every 60 minutes
expires_in_seconds – token expiration time in seconds

How to Refresh the Token

Using the getRefreshToken method, you can refresh the token so that it refreshes and returns the new response once it expires.

```
object tokenResponse = tokenService.getRefreshToken(refreshToken);
```

This will return the new tokenResponse object again.

9.2 Connecting Through a Proxy

Set Proxy Example for .NET

To connect through the SDK proxy, use the following code:

```
using SDKServices.Model;

// Create Data Input
SDKDataInput dataInput = new SDKDataInput();
// proxy required
dataInput._Proxy = new SDKProxy( "proxyUser",       // proxyUser - optional (proxy dependent) or null
                                 "proxyPass",       // proxyPass - optional (proxy dependent) or null
                                 "proxy.corp.com",  // proxyHost - required (uri or ip address)
                                  1234,             // proxyPort - required (port #)
                                  ""              ); // proxyDomain - optional (proxy dependent) or null
```

Simple Request Example for .NET

To perform a simple request, use the following code:

```
using SDKServices.Model;
using SDKServices.ServicesImpl;
using SDKServices.Util;
using SDKTokenServices.Model.Input;
using SDKTokenServices.Model.Output;
using SDKTokenServices.Services;
using SDKTokenServices.ServicesImpl;
// Create Service
SDKDataServicesWrapperImpl service = new SDKDataServicesWrapperImpl();
// Create Data Request Container
List<SDKDataRequest> reqs = new List<SDKDataRequest>();
// Create Data Request
SDKDataRequest req;
// Set Data Request
req = new SDKDataRequest(SDKEnumerators.Functions.GDSP.ToString(),
    new List<string> { "IBM", "IQ313055" },
    new List<string> { "IQ_TOTAL_REV", "IQ_COST_REV" },
    new Dictionary<string, string> { { "periodType", "IQ_FY" },
                                     { "restatementTypeId", "LFR" },
                                     { "filingMode", "P" },
                                     { "currencyId", "USD" },
                              { "currencyConversionModeId", "HISTORICAL" } });
// Add Data Request to Data Request Container
reqs.Add(req);
// Create Data Request
req = new SDKDataRequest(SDKEnumerators.Functions.GDSP.ToString(),
                new List<string> { "00083BAB1", "001055102" },
                new List<string> { "SECURITY_TO_ISSUER_NAME" },
                new Dictionary<string, string>());
 // Add Data Request to Data Request Container
 reqs.Add(req);
 // Create Data Request
 req = new SDKDataRequest(SDKEnumerators.Functions.GDST.ToString(),
                new List<string> { "00083BAB1", "001055102" },
                new List<string> { "ASK_PRICE", "BID_PRICE" },
                new Dictionary<string, string> { { "startDate", "01/01/2014" },
                                                 { "endDate", "" },
                                                 { "frequency", "D" } });
 // Add Data Request to Data Request Container
 reqs.Add(req);

 // get the bearer token
 TokenService tokenService = new TokenServiceImpl();

 string apiUser = ""; // apiUser required - ciqapi accountid
```

```
 string apiPass = ""; // apiPass required - ciqapi password


 // Use below for generating access token using user credentials
 TokenServiceBaseInput tokenServiceBaseInput = new TokenServiceBaseInput();
 tokenServiceBaseInput.Username = apiUser;
 tokenServiceBaseInput.Password = apiPass;

 object tokenResponse = tokenService.getToken(tokenServiceBaseInput);
  // convert json response to TokenResponseVO model object
 TokenResponseVO tokenResponseVO =
Newtonsoft.Json.JsonConvert.DeserializeObject<TokenResponseVO>(
tokenResponse.ToString());

 string bearerToken = tokenResponseVO.access_token;
 // Create Data Input
 SDKDataInput dataInput = new SDKDataInput(
bearerToken, // bearer token
                          null, // null or SDKProxy
                          new List<SDKDataRequest>(reqs));
 // Make Data Request - Native Object (List<SDKDataOutput>) Output
 var respOBJECT = service.InvokeDataService(dataInput) as List<SDKDataOutput>;
 // Make Data Request - XML Output
 var respXML = service.InvokeDataService(dataInput,
 SDKEnumerators.OutputTypes.XML.ToString()) as XDocument;
```

... Process the Results

JSON Direct (JSON/REST) API Access Method – HTTP Post

Always use an HTTP POST (vs. HTTP GET) for API JSON Direct requests targeting the REST service UR endpoint. Use the POST body for passing the JSON API request (inputRequests=<JSON>) information.

Note: Do not issue the POST with the JSON request information as a parameter on the URI since there are restrictions on the URI size.

Simple JSON Request Examples for Python:

```
# GET THE TOKEN

import requests

api_token_url = "https://api-
ciq.marketintelligence.spglobal.com/gdsapi/rest/authenticate/api/v1/token"

HEADERS = {
'Accept': 'application/json',
'Content-Type': 'application/x-www-form-urlencoded'
}
data = {
'username': "--actual_username--",
'password': "--actual_password--"
}

session = requests.Session()
response = session.post(api_token_url, headers=HEADERS, data=data)

if response.status_code == 200:
response_data = response.json()
access_token = response_data.get('access_token')
print("Access Token:", access_token)
else:
print(f"Failed to retrieve token: {response.status_code} - {response.text}")


//CALL THE CIQ API WITH TOKEN

import requests
import json

endpoint_url = "https://api-ciq.marketintelligence.spglobal.com
/gdsapi/rest/v3/clientservice.json"

bearer_token = "--access_token--"
headers = {
"Content-Type": "application/json",
```

```
"Authorization": f"Bearer {bearer_token}"
}

req_array = [
{"function": "GDSP", "identifier": "IBM:NYSE", "mnemonic": "IQ_TOTAL_REV", "properties":
{"PeriodType": "IQ_FY"}},
{"function": "GDSHE", "identifier": "IBM:NYSE", "mnemonic": "IQ_EBITDA", "properties":
{"PeriodType": "IQ_FY-4", "restatementTypeId": "LC"}},
{"function": "GDST", "identifier": "IBM:NYSE", "mnemonic": "IQ_MARKETCAP", "properties":
{"frequency": "Monthly", "startDate": "01/01/2018"}}
]

payload = {"inputRequests": req_array}


response = requests.post(
endpoint_url,
headers=headers,
data=json.dumps(payload)
)
response.raise_for_status()
response_data = response.json()
print(response_data)
```

10 API Best Practices

This section outlines best practices to follow when using the S&P Capital IQ API with the API Client Libraries or the JSON Direct (JSON/REST) access method.

Caching Data (Locally)

Caching data locally improves overall performance. If your application uses the same data point(s) frequently and you do not need to refresh data via the API every time, caching locally avoids the overhead of redundant requests. Additionally, if your workflow requires historical data points, local caching is highly recommended as most historical data is static.

Validate API Inputs

Validate all API request input parameters (i.e. functions, identifiers, mnemonics, and property key/value pairs) prior to making any API request. This avoids the most common API errors.

Avoid Duplicate API Requests

When an API request is in progress, wait for the API response before making the same request. Multiple requests for the same identifier/mnemonic combinations can cause temporary blocking of API request processing leading to delays in API responses.

Multiple API Requests

Although concurrent API requests per IP address are supported, we advise limiting the number of concurrent API request threads to 5.

API Request Sizes (Identifiers/Mnemonics per Request)

It is good practice to group multiple functions, identifiers, and mnemonics in a single API request, although we recommend avoiding large requests. The API connection timeout is 10 minutes for a single API request.

When coding against the API, it's helpful to size the number of data points being requested. We suggest tuning individual API requests to yield approximately 1000 data points, where each request is defined as one function, mnemonic, identifier (plus other property combinations as applicable).

Time Series, Historical, and Vector Information

The Time Series, Historical, and Vector functions (GDST, GDSPV, GDSHE and GDSHV) require more processing time and return more data points than the Point-in-Time functions (GDSP and GDSG).

In addition to sizing the number of requests, when making requests for time series, historical, or vector API requests, limit the input ranges (data, rank, etc.) to return smaller API requests with better response times.

API Response Errors

It is good practice to resolve all known API errors. This avoids duplicate API requests with the same exception such as making multiple requests with an incorrect API user name and password, invalid inputs, etc.

Update to the Latest API Client Libraries

If you are using the API Client Libraries and are notified of a new library release, update the libraries to the latest version in order to leverage the latest features of the API.

Proxy Connection Issues

If your API application is behind a proxy, ensure you have necessary proxy credentials to access the API platform. If the proxy credentials change or expire, make the necessary adjustments prior to making API requests.

Multithreading

The following sample code demonstrates a multithread/asynchronous to call. You can use this loop and threading and customize your input request details and identifiers.

```
private static void testMultithreaded(int threadCount)
        {


            //List<String> identifiers = TestSDKDataInputFactory.getIdentifierList();
            IList<Thread> threads = new List<Thread>();
            for(int i= 0; i < threadCount ; i++)
            {
                Thread t = new Thread(delegate()
                {

                    try
                    {

                        ISDKDataServicesWrapper serviceImpl = new
SDKDataServicesWrapperImpl();
                        SDKDataInput inputRequest = createNewRequest();
                        Object response = serviceImpl.InvokeDataService(inputRequest,
SDKEnumerators.OutputTypes.OBJECT.ToString());
                        printOutput(response);
                        MessageBox.Show(" Successfully executed
:"+Thread.CurrentThread.Name + "- " + response.ToString());
                    }
                    catch (Exception e)
                    {
                        log.Info(e.ToString());
                        MessageBox.Show(e.Message + " ");
                    }
                 });
               t.Name = "Thread-" + i.ToString();
               t.Start();
               threads.Add(t);

            }


        }
```

```
public HashMap<Integer, Object>  processRequests() {
       // Thread pool of n threads
   ExecutorService es = Executors.newFixedThreadPool(n);
   LinkedList<Future<HashMap<String, Object>>> futures = new
LinkedList<Future<HashMap<String, Object>>>();

   int listsize = inputRequests.getDataRequests().size();
   int sublistsize =0;
   if(listsize > 1) {
      sublistsize = 1;
   } else {
      sublistsize = listsize;
   }
   int fromIndex =0;
   int endIndex =0;
   long startProcessingTime = 0;
   for (int i = 1; i <= n; i++) {

      List<SDKDataRequest> subInputRequests = new LinkedList<SDKDataRequest>();
      if(i==n){
         endIndex = listsize;
      }else {
         endIndex = fromIndex + sublistsize;
      }
       subInputRequests = inputRequests.getDataRequests().subList(fromIndex, endIndex);
      fromIndex = endIndex;

      SDKDataInput inputRequest = new SDKDataInput();
      inputRequest.setBearerToken(inputRequests.getBearerToken());
      inputRequest.setDataRequests(subInputRequests);
      Log.info(" Start worker " + i + " at time (in ms)  "+System.currentTimeMillis());
      startProcessingTime = System.currentTimeMillis();
      ParallelTask parallelTask = new ParallelTask(i, inputRequest , serviceImpl);
      futures.add(es.submit(parallelTask));

   }

   HashMap<Integer, Object> outputs = new HashMap<>();
   Log.info("Parallel requests are in progress ..");
   int ctr = 0;
   for (Future<HashMap<String, Object>> future : futures) {
      try {
            HashMap<String, Object> result = future.get();
            long endTime = System.currentTimeMillis();
            long totalProcessingTime
                  = System.currentTimeMillis() - startProcessingTime;
            int batchNo = (Integer) result.get("BATCHNO");
            String resp = (String) result.get("PARALLEL_RESPONSE");
            outputs.put(batchNo, resp);
            Log.info("** worker " + ++ctr + " is done.(end time in ms= "+endTime + " )
Time taken for the complete request/response is " + totalProcessingTime +" ms");
      } catch (Exception e) {
         Log.error("Exception in method processRequests",e);
      }
   }
   es.shutdown();
   return outputs;
}
```

```
NOTE:

inputRequest  - is type SDKDataInput which will have all the requests
(List<SDKDataRequest> )

serviceImpl – is the object of type SDKDataServicesWrapperImpl
```

11 FAQ

1) I'm getting error responses such as Not Applicable, Invalid Mnemonic, etc. What do these mean?

See the section on Error Responses.

2) How do I handle an HTTP Proxy?

For the API client libraries:

If a proxy is not required, it can be set to" null".

If a proxy is required, the ProxyUser, ProxyPassword, ProxyHost and ProxyPort are required. ProxyDomain is optional and can be "null" if not required.

For direct services access, please use the appropriate means in the language to set an HTTP Proxy while making web requests.

3) Are there any limitations on the underlying HTTP commands (GET, POST, PUT, etc.)?

Only the HTTP POST command is supported for now.

4) How do I handle connection persistence?

HTTP is a stateless protocol. However HTTP keepalive should be used to keep the connection persistent for a longer period of time. For most modern browsers, this is a built in feature.

For direct services access, please use the appropriate means in the language to set keepalive property in HTTP Header.

The client library already implements this and sets keepalive to 30 seconds.

5) Are there any restrictions on the number of connections per user?

Please refer to Section 9 Standard Account Attributes for details.

6) Are there request or response size limitations?

Please refer to Section 9 Standard Account Attributes for details.

7) Do you support HTTP compression?

To make better use of available network bandwidth and faster transmission speeds between the client and the On Demand server, we support HTTP Compression in gzip format.

http://en.wikipedia.org/wiki/HTTP_compression

8) What is required for multithreading using the client libraries?

Each thread must create a separate instance of the Data Services Wrapper class.

9) Why do I see extra log entries when I configure log4net?

Once log4net (or log4j) is configured with the project, the client libraries will utilize this to add extra log entries in the same log file.

12 Standard Account Attributes

Trial Account	
Concurrent Mnemonic Function Limit	*100*
Daily Limit	*24000 requests*

Production Account	
Concurrent Mnemonic Function Limit	*500*
Daily Limit	*1,000,000 requests*

If you have questions, please reach out to your account representative or API Support at support.api.mi@spglobal.com.

13 Appendix

Mnems	Date Type	Date Type Description	Date Format	Details	Example	Default
ALL	Absolute Date	Calendar Date	mm/dd/yyyy		06/30/2010	Current Date
n/a	Absolute Date	Calendar Date	mm-dd-yyyy		06-30-2010	n/a
n/a	Absolute Date	Calendar Date	dd-MMM-yyyy	where MMM is month abbreviation	30-JUN-2010	n/a
CSD	Absolute Y	Fiscal Year	Yyyyy		Y2010	Current Y
CSD	Absolute SY	Fiscal Semi	SsYyyyy	where s is 1 or 2	S2Y2010	Current SY
CSD	Absolute QY	Fiscal Quarter	QqYyyyy	where q is 1 thru 4	Q4Y2010	Current QY
IQ	Absolute Month	Calendar Month	MMMyyyy	where MMM is month abbreviation	JUN2010	Current Month
IQ	IQ Absolute FY	Fiscal Year	FYyyyy		FY2010	Current FY
IQ	IQ Absolute CY	Calendar Year	CYyyyy		CY2010	Current CY
IQ	IQ Absolute FH	Fiscal Half	FHhyyyy	where h is 1 or 2	FH12010	Current FH
IQ	IQ Absolute CH	Calendar Half	CHhyyyy	where h is 1 or 2	CH22010	Current CH
IQ	IQ Absolute FQ	Fiscal Quarter	FQqyyyy	where q is 1 thru 4	FQ12010	Current FQ
IQ	IQ Absolute CQ	Calendar Quarter	CQqyyyy	where q is 1 thru 4	CQ42010	Current CQ
IQ	IQ Absolute LTM	Last 12 Months	LTMqyyyy	where q is 1 thru 4	LTM42010	Current LTM
IQ	IQ Absolute NTM	Next 12 Months	NTMqyyyy	where q is 1 thru 4	NTM42010	Current NTM
IQ	IQ Absolute YTD	Year-to-Date	YTDqyyyy	where q is 1 thru 4	YTD42010	Current YTD
IQ	IQ Relative FY	Fiscal Year	IQ_FY	current FY period	IQ_FY	Current FY
IQ	IQ Relative FY	Fiscal Year –	IQ_FY–n	where n is FY periods back	IQ_FY–3	Current FY–n
IQ	IQ Relative FY	Fiscal Year +	IQ_FY+n	where n is FY periods forward	IQ_FY+3	Current FY+n
IQ	IQ Relative CY	Calendar Year	IQ_CY	current CY period	IQ_CY	Current CY
IQ	IQ Relative CY	Calendar Year –	IQ_CY–n	where n is CY periods back	IQ_CY–7	Current CY–n
IQ	IQ Relative CY	Calendar Year +	IQ_CY+n	where n is CY periods forward	IQ_CY+7	Current CY+n
IQ	IQ Relative FH	Fiscal Half	IQ_FH	current FH period	IQ_FH	Current FH
IQ	IQ Relative FH	Fiscal Half –	IQ_FH–n	where n is FH periods back	IQ_FH–5	Current FH–n
IQ	IQ Relative FH	Fiscal Half +	IQ_FH+n	where n is FH periods forward	IQ_FH+5	Current FH+n
IQ	IQ Relative CH	Calendar Half	IQ_CH	current CH period	IQ_CH	Current CH
IQ	IQ Relative CH	Calendar Half –	IQ_CH–n	where n is CH periods back	IQ_CH–3	Current CH–n
IQ	IQ Relative CH	Calendar Half +	IQ_CH+n	where n is CH periods forward	IQ_CH+3	Current CH+n
IQ	IQ Relative FQ	Fiscal Quarter	IQ_FQ	current FQ period	IQ_FQ	Current FQ
IQ	IQ Relative FQ	Fiscal Quarter –	IQ_FQ–n	where n is FQ periods back	IQ_FQ–11	Current FQ–n
IQ	IQ Relative FQ	Fiscal Quarter +	IQ_FQ+n	where n is FQ periods forward	IQ_FQ+11	Current FQ+n
IQ	IQ Relative CQ	Calendar Quarter	IQ_CQ	current CQ period	IQ_CQ	Current CY
IQ	IQ Relative CQ	Calendar Quarter –	IQ_CQ–n	where n is CQ periods back	IQ_CQ–15	Current CY–n
IQ	IQ Relative CQ	Calendar Quarter +	IQ_CQ+n	where n is CQ periods forward	IQ_CQ+15	Current CY+n

Mnems	Date Type	Date Type Description	Date Format	Details	Example	Default
IQ	IQ Relative LTM	Last 12 Months	IQ_LTM	current LTM period	IQ_LTM	Current LTM
IQ	IQ Relative LTM	Last 12 Months –	IQ_LTM–n	where n is LTM periods back	IQ_LTM–19	Current LTM–n
IQ	IQ Relative LTM	Last 12 Months +	IQ_LTM+n	where n is LTM periods forward	IQ_LTM+19	Current LTM+n
IQ	IQ Relative NTM	Next 12 Months	IQ_NTM	current NTM period	IQ_NTM	Current NTM
IQ	IQ Relative NTM	Next 12 Months –	IQ_NTM–n	where n is NTM periods back	IQ_NTM–7	Current NTM–n
IQ	IQ Relative NTM	Next 12 Months +	IQ_NTM+n	where n is NTM periods forward	IQ_NTM+7	Current NTM+n
IQ	IQ Relative YTD	Year-to-Date	IQ_YTD	current YTD period	IQ_YTD	Current YTD
IQ	IQ Relative YTD	Year-to-Date –	IQ_YTD–n	where n is YTD periods back	IQ_YTD–9	Current YTD–n
IQ	IQ Relative YTD	Year-to-Date +	IQ_YTD+n	where n is YTD periods forward	IQ_YTD+9	Current YTD+n

14 Revision History

The changes made to this document include the following:

Client Library	Date	Changes
2.0.0	December 2011	Beta Release Version 2.0.0
2.0.0.7	March 2012	Beta Release Version 2.0.0.7
2.0.0.12	June 2012	Beta Release Version 2.0.0.12
2.0.0.15	August 2012	Beta Release Version 2.0.0.15
2.0.0.16	August 2012	Beta Release Version 2.0.0.16
2.0.0.19	January 2013	Beta Release Version 2.0.0.19
2.0.0.29	May 2013	Beta Release Version 2.0.0.29
2.0.0.31	July 2013	Version 2.0.0.31
2.0.0.40	November 2013	Version 2.0.0.40
2.0.0.42	June 2014	Version 2.0.0.42
2.0.0.45	August 2014	Version 2.0.0.45
2.0.0.46	December 2014 January 2015 May 2015	Version 2.0.0.46
	June 2016	Added the Appendix and Revision History sections
	September 2016	• Reflects new company branding • Section 1.1 now includes detailed descriptions of Data Sets • Section 3.3 Specialized Data Request Properties has been updated to include additional Functions and detailed Descriptions • Added new content: Section 3.4 Requesting Data with Identifiers
	November 2016	• Added definition for tenor in Section 3.3.2 Descriptions • Minor revisions to Section 8 FAQ • Added Section 1.1.18 Cross Reference Services
	February 2017	• Updated URLs for client libraries
2.0.0.50	August 2017	• Added hyperlink to API Properties Matrix • Added Credit Analytics descriptive content • Updated links to point to v2.0.0.50 Client Libraries
2.1.1	October 2017	• Added content highlighting Investor Activism data • Updated links to point to v2.1.1 Client Libraries • Updated .NET Minimum Supported Frameworks and Language Versions from 3.5 to 4.5.2
	November 2017	• Updated section titled “JSON Direct (JSON/REST) API Access Method – HTTP Post” • Updated TLS version to v1.2
2.1.3	March 2018	• Added Business Relationships descriptive content • Updated links to point to v2.1.3 Client Libraries

Client Library	Date	Changes
2.1.4	July 2018	• Updated for new v2.1.4 Client Libraries • Updated files included in the download in section 2.2 • Updated Web Service Direct endpoint in section 5
2.1.5	September 2018	• Updated for new v2.1.5 Client Libraries
	October 2018	• Removed concurrent request limit per server under Standard Account Attributes. Changed daily limit for trial accounts to 10,000.
		• Removed macroeconomic data.
	November 2018	• Changed the “Daily Limit” of Trial Account to 24000. Removed the “Hourly Limit” attribute under Trial Account as well as Production Account sections.
	July 2019	• Updated the Cross Reference Services section to include Industry Sector Cross Reference Service • Added a link to the supported mnemonics and identifiers on the Support Site • Added 2019 copyright and included other branding updates
	November 2019	• In October 2018, only some of the macroeconomic data was removed from the API. Added back the content that describes this data set. • When using Ticker Symbol as an Identifier, added the suggestion to append a colon to the Ticker Symbol (e.g., IBM:) as a recommended practice • API Time Out Value changed to **10 Minutes**
	December 2019	• Updated section 1.1.13 S&P Global Credit Ratings
2.1.7	February 2020	Updated all references to current client libraries: o v2.1.5 (Jackson Library v1) o v2.1.7 (Jackson Library upgraded to v2) • Added content for new OpenJDK library v2.1.8 • Updated disclosure for 2020
	May 2020	• Removed Equity ETFs from the Data Sets section
	July 2020	• Updated references and hyperlinks to new client libraries
	October 2020	• Updated references and hyperlinks to new client libraries
	December 2020	• Added content for multithreading
2.1.8	January 2021	• Added log4net question to FAQs in section 8
	March 2021	• Support site and email address update
2.1.9	September 2021	• Updated references and hyperlinks to new client libraries and on new Marketplace support site
	October 2021	• Updates with references to the Marketplace support site for supporting documentation
	January 2022	• Client libraries updated for log4j issue: v2.1.6 Open JDK and 2.1.8 Open JDK Jackson

<table>
<tr><th>Client Library</th><th>Date</th><th>Changes</th></tr>
<tr><td rowspan="2">2.2.0</td><td>June 2022</td><td>Client libraries updated: Client library are upgraded to latest spring version 5.3.19 and log4j version 2.17.2 excluding</td></tr>
<tr><td>July 2022</td><td>.NET library update from 2.1.5.4 to v2.2.0 (Framework 4.8) url update in sections 1.3 and 2.1 and framework reference to 4.8
.Net sample query</td></tr>
<tr><td rowspan="2">2.2.1</td><td>November 2022</td><td>Java library updates from v2.2.0 to 2.2.1
-spgmi_api_sdk_java_openjdk_v2.2.1_jackson.zip
-spgmi_api_sdk_java_openjdk_v2.2.1_withoutjackson.zip
-spgmi_api_sdk_java_oraclejdk_v2.2.1_jackson.zip
Library updates includes changing the logging from org.apache.log4j.Logger to org.apache.logging.log4j.Logger</td></tr>
<tr><td>October 2023</td><td>Added
-Sample Library with additional required headers and
-Test SDK
for
.NET Framework 4.8.0 v2.2.1</td></tr>
<tr><td rowspan="3">2.3.0</td><td>September 2022</td><td>.NET Core Library additions
.NET Core 6.0.0 2.3.0 (win-x64)
.NET Core 6.0.0 2.3.0 (win-arm64)</td></tr>
<tr><td>April 2023</td><td>Added minimum versions and support detail for .NET libraries supported by Framework 4.8 and Core v6.0</td></tr>
<tr><td>October 2023</td><td>-Added description for the use of identifier prefixes in section 3.4.5. -Adjustment description for Compustat population.</td></tr>
<tr><td>2.3.1</td><td>October 2023</td><td>Added Sample Libraries with additional required headers:
.NET Core 6.0.0 v2.3.1 (win-x64)
.NET Core 6.0.0 v2.3.1 (win-arm64)
Added Test SDK:
.NET Core 6.0.0</td></tr>
<tr><td>3.0.0</td><td>July 2024</td><td>Added:
S&P Capital IQ API Token Authentication
Kensho Link
Static IP addresses
Usage Service
Separately, added library updates supporting Kensho Link and Bearer Token Authentication:
.NET 4.8 v3.0.0
.NET Core 6.0.0 v3.0.0 (win-arm64)
.NET Core 6.0.0 v3.0.0 (win-x64)</td></tr>
</table>

Client Library	Date	Changes
		OPEN JDK 1.8 v3.0.0 OPEN JDK 11 v3.0.0 Added Test SDKs: .NET 4.8 v3.0.0 .NET Core 6.0.0 v3.0.0 JDK 1.8 v3.0.0 JDK 11 v3.0.0
	October 2024	Addition of Unlock safety in Account Locked error 423
	January 2025	Addition of requirement to include Bearer Authentication in the request header.
	March 2025	Created a new section 12 Standard Account Attributes. Revised to indicate that the Daily Limit for Production Accounts is 1,000,000 requests.
3.1.0	May 2025	Added library updates providing 1) Proxy support changes for all Java and .NET Libraries 2) Updated .NET libraries with strict SSL certificate validation 3) New property, SCALETYPE, support for CA calls. **NOTE:** All these libraries will support only bearer authentication. New Library Versions: OPEN JDK 1.8 v3.1.0 OPEN JDK 11 v3.1.0 OPEN JDK 21 v3.1.0 .NET Framework 4.8 v3.1.0 .NET 6.0.0 v3.1.0 (win-arm64) .NET 6.0.0 v3.1.0 (win-x64)
	October 2025	Addition of Kensho Link Entity Quick match.
3.1.1	November 2025	Modified Authentication section due to release of API Catalog service. Modified links for release of new libraries: OPEN JDK 1.8 v3.1.1 OPEN JDK 11 v3.1.1 OPEN JDK 21 v3.1.1 .NET Framework 4.8 v3.1.1 .NET 6.0.0 v3.1.1 (win-x64) .NET 6.0.0 v3.1.1 (win-arm64)

Client Library	Date	Changes
		And new Test SDKs: Test SDK Java v3.1.0 Test SDK .NET Framework 4.8.0 v3.1.0 Test SDK .NET Core 6.0 v3.1.0 The new libraries and test SDKs add the following features: • The new libraries and Test SDKs support the properties for the new mnemonics: KL_PRO_ENTITY_ID_NAME_QUICK_MATCH KEY_DEV_PIT_DETAILS and EVENT_PIT_DETAILS HEADCOUNT • .NET Framework 4.8.0 and .NET Core 6.0 are supported with lognet v3.2.0
3.1.2	January 2026	Updated URLs in S&P Capital IQ Authentication section to new address for API Catalog