Factiva Analytics

Release 0.3.12

Dow Jones - Customer Solutions Engineering (CSE) Team

Apr 10, 2023

GETTING STARTED

This package simplifies the integration process to the Dow Jones Factiva Analytics services.

Check out the Dow Jones Developer Portal for more information about all available services.

CHAPTER

ONE

OVERVIEW

`factiva-analytics` is a package that provides tools to ease the integration of Factiva Analytics APIs which is part of the Dow Jones Developer Platform family. This packages aims to simplify news data operations like estimations, extractions and real-time consumption.

CHAPTER

TWO

INSTALLATION

2.1 PIP

This package can be installed using PIP. The recommended pocedure is running:

```
pip install -u factiva-analytics
```

This will install and/or update the package to the latest official release.

Alternatively it can be installed directly from GitHub by running:

```
pip install git+https://github.com/dowjones/factiva-analytics-python.git@main
```

package guidelines establish that the `main` branch is also the latest official release. However, this method allows to install pre-release versions in any of the available branches in the repository like `dev`.

2.2 Optional Packages

The package automatically installs all prerequisites automatically. However, some custom handlers for `Streams` and other processing components in the `integration` module may require the installation of additional packages like Elasticsearch or Google Cloud BigQuery.

This is the list of optional packages. Installing them is recommended as long as these components will be used within the solution.

- elasticsearch: Used in a Streams custom handler and bulk data import.
- bigquery: Used in a Streams custom handler and bulk data import.
- MongoDB: Used in a Stream custom handler and bulk data import.

CHAPTER

THREE

ENVIRONMENT VARIABLES

When a class is instantiated, depending on the functionality some Environment Variables might be required unless a value is specified explicitly in the code.

3.1 Authentication

3.1.1 UserKey

- **FACTIVA_USERKEY: Assigned API user key. E.g. abcd1234abcd1234abcd1234abcd1234.**
  Used in all services except ArticleRetrieval.

3.1.2 OAuthUser

- **FACTIVA_CLIENTID: Assigned OAuth Client ID. E.g. 0abcd1wxyz2abcd3wxyz4abcd5wxyz6o.**
  Required for ArticleRetrieval.
- **FACTIVA_USERNAME: Assigned OAuth Username. E.g. 9ZZZ000000-svcaccount@dowjones.com.**
  Required for ArticleRetrieval.
- **FACTIVA_PASSWORD: Assigned OAuth Password. E.g. pa55WOrdpa55WOrd.**
  Required for ArticleRetrieval.

3.2 Snapshots & Streams

- **FACTIVA_WHERE: Query where statement that will be used when creating a new Snapshots**
  or Streams object with no where/query parameter.
- **FACTIVA_SUBSCRIPTIONID: Subscription ID from an existing Streaming Instance. E.g.**
  dj-synhub-stream-abcd1234abcd1234abcd1234abcd1234-1234abcxyz-filtered-abc123.

3.3 Logging

- **FACTIVA_LOGLEVEL: Level of detail for the logs.**
  Accepted values are DEBUG, INFO (*default*), WARNING, ERROR, CRITICAL.

3.4 Handlers and Data Processing

3.4.1 Elasticsearch

ENV variables used in Elasticsearch.

3.4.2 BigQuery

ENV variables used in BigQuery.

CHAPTER

FOUR

QUICK OPERATIONS

4.1 Check Account Statistics

Assumes the ENV variable `FACTIVA_USERKEY` is set.

```
from factiva.analytics import AccountInfo
u = AccountInfo()
print(u)
```

```
<'factiva.analytics.AccountInfo'>
├─user_key: <'factiva.analytics.UserKey'>
│  ├─key: **************************1234
│  └─cloud_token: **********************YKB22sJCkHXX
├─account_name: AccountName
├─account_type: account_with_contract_limits
├─active_product: DNA
├─max_allowed_concurrent_extractions: 1
├─max_allowed_extracted_documents: 2,200,000
├─max_allowed_extractions: 10
├─currently_running_extractions: 0
├─total_downloaded_bytes: 84,195,293
├─total_extracted_documents: 145,605
├─total_extractions: 3
├─total_stream_instances: 2
├─total_stream_subscriptions: 2
├─enabled_company_identifiers:
│  ├─[1]: sedol
│  ├─[3]: cusip
│  ├─[4]: isin
│  └─[5]: ticker_exchange
├─remaining_documents: 2.054,395
└─remaining_extractions: 7
```

4.2 Get Account's Historical Full Extractions

Uses the passed `key` parameter and ignores the ENV variable `FACTIVA_USERKEY`.

```
from factiva.analytics import AccountInfo
u = AccountInfo(key='abcd1234abcd1234abcd1234abcd1234')
extractions = u.get_extractions()
```

The variable `extractions` will contain a Python `list` of `SnapshotExtraction` objects.

4.3 Get Volume Estimates With Snapshot Explain

Assumes the ENV variable `FACTIVA_USERKEY` is set.

```
from factiva.analytics import SnapshotExplain
my_query = "publication_datetime >= '2020-01-01 00:00:00' AND LOWER(language_code) = 'en'
↪"
my_explain = SnapshotExplain(query=my_query)
my_explain.process_job()  # This operation can take a few minutes to complete
print(my_explain)
```

```
<'factiva.analytics.SnapshotExplain'>
├─user_key: <'factiva.analytics.UserKey'>
│   ├─key: ****************************1234
│   └─cloud_token: **********************YKB22sJCkHXX
├─query: <'factiva.analytics.SnapshotExplainQuery'>
│   ├─where: publication_datetime >= '2023-01-01 00:00:00' AND UPPER(source_code) = 'DJDN'
│   ├─includes: <NotSet>
│   ├─excludes: <NotSet>
│   ├─include_lists: <NotSet>
│   └─exclude_lists: <NotSet>
├─job_response: <'factiva.analytics.SnapshotExplainJobResponse'>
│   ├─job_id: 648075e7-b551-4bdb-b8f4-ed7f470ae6bd
│   ├─job_link: https://api.dowjones.com/alpha/extractions/documents/648075e7-b551-4bdb-
↪b8f4-ed7f470ae6bd/_explain
│   ├─job_state: JOB_STATE_DONE
│   ├─volume_estimate: 203,338
│   └─errors: <NoErrors>
└─samples: <NotRetrieved>
```

After its execution, the object `my_explain.job_results` contains details about the job itself and the estimated volume.

4.4 Get Extraction Details and Download Files

Uses the passed `key` parameter and ignores the ENV variable `FACTIVA_USERKEY`.

```
from factiva.analytics import SnapshotExtraction
se = SnapshotExtraction('zmhsvx20tl')
print(se)
```

```
<factiva.analytics.SnapshotExtraction'>
├─user_key: <'factiva.analytics.UserKey'>
│   ├─key: **************************1234
│   └─cloud_token: **********************YKB22sJCkHXX
├─query: <NotRetrieved>
└─job_response: <factiva.analytics.SnapshotExtractionJobReponse'>
    ├─job_id: dj-synhub-extraction-abcd1234abcd1234abcd1234abcd1234-zmhsvx20tl
    ├─job_link: https://api.dowjones.com/alpha/extractions/documents/dj-synhub-
↪extraction-abcd1234abcd1234abcd1234abcd1234-zmhsvx20tl
    ├─job_state: JOB_STATE_DONE
    ├─short_id: zmhsvx20tl
    ├─files: <list> - [1] elements
    └─errors: <NoErrors>
```

```
se.download_files()
```

When the operation ends, files will be available in the local folder named as the `short_id` attribute (`zmhsvx20tl`).

4.5 Create a Streaming Instance

Assumes the ENV variable `FACTIVA_USERKEY` is set.

```
from factiva.analytics import StreamingInstance
my_query = "publication_datetime >= '2020-01-01 00:00:00' AND LOWER(language_code) = 'en'
↪"
my_stream = StreamingInstance(query=my_query)
my_stream.create()
print(my_stream)
```

```
<'factiva.analytics.StreamingInstance'>
├─id: <Hidden>
├─short_id: 4doq2zigpf
├─user_key: <'factiva.analytics.UserKey'>
│   ├─key: **************************1234
│   └─cloud_token: **********************YKB22sJCkHXX
├─query: "publication_datetime >= '2020-01-01 00:00:00' AND LOWER(language_code) = 'en'"
├─subscriptions:
│   └─short_id: R4QwwB
└─status: JOB_STATE_RUNNING
```

After its execution, the object `my_explain.job_results` contains details about the job itself and the estimated volume.

CHAPTER

FIVE

AUTHENTICATION

Depending on thte service intended to be used, an operation object will need either a `UserKey` or `OAuthUser` instance. As a best practice, it is recommended to use ENV variables to store values to instantiate these objects (see Getting Started > Environment Variables > *Authentication*).

5.1 UserKey

Used by all services except the Article Retrieval Service. Usually it's not required to be instantiated independently as the creation of a *parent* object will get the value from the environment.

If using this class explicitly, the following code snippets can be found helpful:

```
from factiva.analytics import UserKey, SnapshotExplain
u = UserKey('abcd1234abcd1234abcd1234abcd1234')
se = SnapshotExplain(user_key=u)
```

5.2 OAuthUser

Used by the Article Retrieval Service only. Like `UserKey`, it is usually not required to be instantiated independently. However, below code snippets can be helpful when using this class explicitly:

```
from factiva.analytics import OAuthUser, ArticleRetrieval
c_id = "0abcd1wxyz2abcd3wxyz4abcd5wxyz6o"
uname = "9ZZZ000000-svcaccount@dowjones.com"
pwd = "pa55WOrdpa55WOrd"
ou = OAuthUser(client_id=c_id, username=uname, password=pwd)
ar = ArticleRetrieval(oauth_user=ou)
...
```

When using ENV variables, the above snippet becomes shorter.

```
from factiva.analytics import SrticleRetrieval
ar = ArticleRetrieval()
```

CHAPTER

SIX

TAXONOMIES

Taxonomy operations tutorial

CHAPTER

SEVEN

VOLUME ESTIMATES

Accurate volume estimates are based on the Snapshot Explain operation. This operation returns the exact number of matching articles in the archive.

```
from factiva.analytics import SnapshotExplain
where_str = "publication_datetime >= '2020-01-01 AND language_code = 'en' AND REGEXP_
↪CONTAINS(industry_codes, r'(?i)(^|,)(i1|i25121|i2567)($|,)')"
se = SnapshotExplain(query=where_str)
se.process_job()
print(f"The query matches {se.job_results.volume_estimate} articles")
```

```
The query matches 123456 articles
```

Using the same Snapshot Explain object, you can also get metadata samples.

```
se.get_samples()
print(se.samples)
```

```
TODO: Add samples response
```

When volume estimates are in line with your expectations, you can proceed to analyze the data using the Snapshot TimeSeries operation, or directly extract the content via the Snapshot Extract operation.

CHAPTER

EIGHT

EXTRACTIONS

Extraction operations tutorial

CHAPTER

NINE

UPDATES

Update operations tutorial

CHAPTER

TEN

STREAMS

Streaming operations tutorial

CHAPTER

ELEVEN

LISTS

List operations tutorial

CHAPTER

TWELVE

ARTICLE RETRIEVAL

Article Retrieval operations tutorial

CHAPTER

THIRTEEN

ADVANCED QUERIES

13.1 Where Attribute

The where attribute is the most used attribute to select content by applying different conditions to the available fields. Below there are some code snippets that help building a where statement, and notes that help modify the syntax according to the need.

13.1.1 Range of dates

It works with the standard SQL syntax. If updates will be collected in the future, avoid using an end-date. Alternative fields are: *modification_datetime* and *ingestion_datetime*.

```
publication_datetime >= '2010-01-01 00:00:00' AND publication_datetime <= '2020-06-30␣
↪23:59:59'
```

13.1.2 Filter by *source_codes*

The following clause is useful to select sources by their individual code.

```
UPPER(source_code) IN ('AASFNE', 'HTACCF', 'NLADLW', 'ADVTSR', 'AFNROL', 'AGEEOL', 'AGEE
↪', 'HNASNI', 'APRS', 'ASXTEX', 'AUSTOL')
```

In case sources will be selected by their category or source family, a better option is using *restrictor_codes*. This field is not in the documentation, but the CSE or Integration Team can provide more information like source family codes

```
REGEXP_CONTAINS(restrictor_codes, r'(?i)(^|,)(jpost|nytf|wp|latm|j)($|,)')
```

13.1.3 Filter by *subject_codes*

```
REGEXP_CONTAINS(subject_codes, r'(?i)(^|,
↪)(mcat|ccat|ecat|gglobe|ghea|ghnwi|gcns|gpir|gdatap|greest|grisk|gsci|gspace|gtrans)(
↪$|,)')
```

13.1.4 Filtering by the region where the source is headquarted

```
REGEXP_CONTAINS(region_of_origin, r'(?i)(aust|spain|italy|usa|uk)')
```

13.1.5 Filtering by language

```
LOWER(language_code) IN ('en', 'es', 'it')
```

13.1.6 Filtering by company codes

This is applicable to any company-related fields (about, occur or company_codes and other combinations with identifiers - ISIN, CUSIP…).

```
REGEXP_CONTAINS(company_codes, r'(?i)(^|,
↪)(agbpet|agip|agphng|agpnme|agzgi|altgaz|bbor|brnene|distrg|eenivm|egapg|enichm|enie|enimnt)(
↪$|,)')
```

In case the interest is to ensure at least one company is tagged (the field is not empty), the expressions looks like this

```
LENGTH(company_codes) > 2
```

Filtering for content with at least 1 relevant company

```
LENGTH(company_codes_about) > 0
```

13.1.7 Filtering by Industry code

```
REGEXP_CONTAINS(industry_codes, r'(?i)(^|,)(i1|i25121|i2567)($|,)')
```

13.1.8 Filtering by Executive codes

```
REGEXP_CONTAINS(LOWER(person_codes), r'(?i)(^|,)(76064380|2349856)($|,)')
```

13.1.9 Filtering by the region the article is about

```
REGEXP_CONTAINS(region_codes, r'(?i)(^|,)(aust|spain|italy|usa|uk)($|,)')
```

13.1.10 Filtering by terms in full-text (Keyword search)

```
REGEXP_CONTAINS(CONCAT(title, ' ', IFNULL(snippet, ''), ' ', IFNULL(body, '')), r'(?i)(^
↪|\b)(economic|economy|regulation|deficit|budget\W+tax|central\W+bank)($|.|\b)')
```

More examples are available in the Data Selection Samples in the Dow Jones Developer Portal (https://developer.dowjones.com/site/docs/data_selection_samples/index.gsp#)

Building the where statement. Python concatenates the strings when inside the parenthesis. Mind the extra space at the end of each string.

CHAPTER

FOURTEEN

AUTHENTICATION SERVICE

Module part of the core componenets for the Factiva Analytics python package. Contains classes and tools that allow to interact with the authentication and authorization elements of the Factiva Analytics API.

14.1 UserKey

class factiva.analytics.auth.userkey.**UserKey**(*key=None*)

Class that represents an API user and can be instantiated based on the user-key value provided by the Dow Jones Developer Support team.

get_cloud_token() → bool

Request a cloud token and stores its content in the `cloud_token` property

Returns
`True` if the operation was completed successfully. `False` otherwise.

Return type
bool

14.2 OAuthUser

class factiva.analytics.auth.oauthuser.**OAuthUser**(*client_id: str | None = None*, *username: str | None = None*, *password: str | None = None*)

Class that represents a Dow Jones OAuth user.

Parameters

- **client_id** (*str*) – Assigned Client ID and communicated via the Welcome Letter. Retrieves the value from the ENV variable `FACTIVA_CLIENTID` it not provided.
- **username** (*str*) – Assigned Username and communicated via the Welcome Letter. Retrieves the value from the ENV variable `FACTIVA_USERNAME` it not provided.
- **password** (*str*) – Assigned password and communicated via the Welcome Letter. Retrieves the value from the ENV variable `FACTIVA_PASSWORD` it not provided.

Examples

Create an `OAuthUser` instance from ENV variables and assign the JWT token to a request headers dictionary.

```
from factiva.analytics import OAuthUser
o = OAuthUser()
headers = {
    'Authorization': f'Bearer {o.current_jwt_token}'
}
```

Shows the relevant properties of a `OAuthUser` instance.

```
from factiva.analytics import OAuthUser
o = OAuthUser()
o
```

output

```
<'factiva.analytics.OAuthUser'>
|-client_id = ****************************4Cs6
|-username = 9ZZZ000000-svcaccount@dowjones.com
|-password = ************gRk3
|-token_status = not_authenticated
```

property current_jwt_token

Returns a valid token to be used in the `Authorization` HTTP header. Recalculates the JWT token automatically if needed.

get_id_token() → bool

Requests an ID token to the DJ auth service (authentication operation) and store the necessary information for furher requests in the relevant instance properties.

Returns
`True` if the operation was completed successfully. `False` otherwise.

Return type
bool

get_jwt_token() → bool

Requests a JWT Authorization token to the Factiva Auth service. The returned token is stored internally and available via the `current_jwt_token` property. Usual expiration is 1 hour (3600 seconds).

Returns
`True` if the operation was completed successfully. `False` otherwise.

Return type
bool

property token_status: str

Provides the current token status:

- `not_authenticated` (`get_id_token()` has not been executed)
- `id_token_expired` (previously obtained ID token has expired)
- `not_authorized` (`get_jwt_token()` has not been executed)
- `jwt_token_expired` (previously obtained JWT token has expired)
- `OK` (token is ready for authenticated requests)

CHAPTER

FIFTEEN

TAXONOMY SERVICE

Module that handles all Factiva Analytics Taxonomies objects within the package. Contains classes and tools that allow to interact with the taxonomies and company identifier endpoints of the Factiva Analytics API.

15.1 FactivaTaxonomyCategory

class factiva.analytics.taxonomy.factiva_taxonomies.**FactivaTaxonomyCategories**(*value*)

Class that provides a unique way to reference the different taxonomy cateories present in Factiva data. Given the fact that the API has two versions of Subjects, Regions and Industries; only the full hierarchy version is implemented. The simple version is a sub-set of the hierarchical dataset.

Examples

Use it directly when needed. Usually as param in `FactivaTaxonomy` methods.

```
from factiva.analytics import FactivaTaxonomyCategories
FactivaTaxonomyCategories.SUBJECTS
FactivaTaxonomyCategories.REGIONS
FactivaTaxonomyCategories.COMPANIES
FactivaTaxonomyCategories.INDUSTRIES
FactivaTaxonomyCategories.EXECUTIVES
```

15.2 FactivaTaxonomy

class factiva.analytics.taxonomy.factiva_taxonomies.**FactivaTaxonomy**(*user_key=None*)

Class that represents the Factiva Taxonomy endpoints in Factiva Analytics.

`Subject`, `industry` and `region` taxonomies have two separate categories in the API. However, current implementation uses only a simplified version where the dataset returns all codes with the minimum set of columns to build a hierarchy.

Parameters

user_key (*str or* UserKey) – String containing the 32-character long APi Key or UserKey instance that represents an existing user. If not provided, the constructor will try to obtain its value from the `FACTIVA_USERKEY` environment variable.

Examples

Creating a taxonomy instance with no user key. Fails if the environment variable `FACTIVA_USERKEY` is not set.

```
from factiva.analytics import FactivaTaxonomy
t = FactivaTaxonomy()
```

Creating a taxonomy instance providing the user key as string

```
from factiva.analytics import FactivaTaxonomy
t = FactivaTaxonomy(user_key='abcd1234abcd1234abcd1234abcd1234')
```

Creating a taxonomy instance with an existing `UserKey` instance

```
from factiva.analytics import UseKey, FactivaTaxonomy
u = UserKey('abcd1234abcd1234abcd1234abcd1234')
t = FactivaTaxonomy(user_key=u)
```

With the `FactivaTaxonomy` instance `t`, it's now possible to call any method. Please see below.

download_raw_category(*category:* FactivaTaxonomyCategories, *path=None*, *file_format='csv'*) → bool

Downloads a CSV or AVRO file with the specified taxonomy category. The file columns preserve their original name, thus it may not match the same column naming used in other methods in this FactivaTaxonomy class.

Parameters

- **category** (FactivaTaxonomyCategories) – Enumerator entry that specifies the taxonomy category for which the codes will be retrieved.
- **path** (*str*) – Folder path where the output file will be stored.
- **file_format** (*str {csv, avro}*) – String specifying the download format

Returns

`True` if the file is correctly downloaded. `False` otherwise.

Return type

bool

Raises

ValueError: – When the parameter `file_fomat` is invalid or not a string

Examples

Getting the raw file for the 'industries' category

```
from factiva.analytics import FactivaTaxonomy, FactivaTaxonomyCategories
f = FactivaTaxonomy()
f.download_raw_category(category=FactivaTaxonomyCategories.INDUSTRIES, path='/
↪home/user/')
```

get_category_codes(*category:* FactivaTaxonomyCategories) → DataFrame

Request for available codes in the taxonomy for the specified category.

Important: The taxonomy category `FactivaTaxonomyCategories.EXECUTIVES` is not currently supported by this operation. Use the `download_category_codes()` method instead.

Parameters

category (FactivaTaxonomyCategories) – Enumerator entry that specifies the taxonomy category for which the codes will be retrieved.

Returns

Dataframe containing the codes for the specified category

Return type

pandas.DataFrame

Examples

Getting the codes for the 'industries' category

```
from factiva.analytics import FactivaTaxonomy, FactivaTaxonomyCategories
t = FactivaTaxonomy()
industry_codes = t.get_category_codes(FactivaTaxonomyCategories.INDUSTRIES)
print(industry_codes)
```

```
             code                descriptor                                    ␣
↪    description direct_parent
code
I0              I0               Agriculture  All farming, forestry, commercial␣
↪fishing, hun...           NaN
I01001      I01001                   Farming  Agricultural crop production,␣
↪seed supply and ...            i0
I03001      I03001               Aquaculture  The farming of aquatic animals␣
↪and plants such...        i01001
I0100144  I0100144             Cocoa Growing                                  ␣
↪Growing cocoa beans.        i01001
I0100137  I0100137            Coffee Growing                                 ␣
↪Growing coffee beans.        i01001
...             ...                       ...                                   ␣
↪             ...           ...
I162          I162               Gas Utilities  Operating gas distribution and␣
↪transmission sy...           i16
IMULTI      IMULTI            Multiutilities  Utility companies with␣
↪significant presence in...         iutil
I17            I17            Water Utilities  Operating water treatment plants␣
↪and/or operat...         iutil
IDESAL      IDESAL                Desalination  Desalination is the process of␣
↪removing salt a...           i17
IDISHEA    IDISHEA  District Heating/Cooling  Heating systems that involve the␣
↪distribution ...           i17
```

lookup_code(*code: str*, *category:* FactivaTaxonomyCategories) → dict

Finds the descriptor and other details based on the provide code and category. Returns all available columns for that entry.

Parameters

- **code** (*str*) – Factiva code for lookup
- **category** (FactivaTaxonomyCategories) – Enumerator entry that specifies the taxonomy category for which the codes will be retrieved.

Returns
Dict containing the code details

Return type
dict

Important: The return dict structure can vary depending on the passed category and the enabled settings for the used account (e.g. company identifiers like ISIN, CUSIP, etc.).

Raises
ValueError – When the parameter code is not a string:

Examples

Lookup a code in the 'subjects' category

```
from factiva.analytics import FactivaTaxonomy, FactivaTaxonomyCategories
f = FactivaTaxonomy()
f.lookup_code(code='CWKDIV', category=FactivaTaxonomyCategories.SUBJECTS)
```

```
{'code': 'CWKDIV', 'descriptor': 'Workplace Diversity', 'description':
↪'Diversity and inclusion in the workplace to ensure employees encompass␣
↪varying traits such as race, gender, ethnicity, age, religion, sexual␣
↪orientation, socioeconomic background or disability.', 'direct_parent': 'C42'}
```

CHAPTER

SIXTEEN

SNAPSHOT EXPLAIN

Module that handles all Factiva Analytics Explain requests and objects. Contains classes and tools that allow to run volume estimates and obtain random samples given a predefined selection criteria (query).

16.1 SnapshotExplain

class factiva.analytics.snapshots.explain.**SnapshotExplain**(*user_key=None*, *query=None*, *job_id=None*)

Main class to interact with the Explain service from Factiva Analytics.

user_key

User representation for service authentication

Type
UserKey

query

Query object tailored for Extraction operations

Type
SnapshotExtractionQuery

job_response

Object containing job status and execution details

Type
SnapshotExtractionJobReponse

samples

Type
SnapshotExplainSamplesResponse

get_job_response() → bool

Performs a request to the API using the job ID to get its status.

If the job has been completed, results are assigned to the `job_response` object.

Returns
True if the get request was successful. An Exception otherwise.

Return type
bool

get_samples(*num_samples: int = 100*)

Performs a request to the API using the job ID to get its status.

If the job has been completed, results are assigned to the `job_response` object.

Returns
True if the get request was successful. An Exception otherwise.

Return type
bool

process_job()

Submits a new job to be processed, wait until the job is completed and then retrieves the job results.

Returns
True if the explain processing was successful. An Exception otherwise.

Return type
bool

submit_job()

Performs a POST request to the API using the assigned query to start an Explain job.

If the job is initiated succesfully, results are assigned to the `job_response` object. Otherwise any HTTP error will raise an exception.

Returns
True if the submission was successful. An Exception otherwise.

Return type
bool

16.2 SnapshotExplainQuery

class factiva.analytics.snapshots.explain.**SnapshotExplainQuery**(*where=None*, *includes: dict | None = None*, *include_lists: dict | None = None*, *excludes: dict | None = None*, *exclude_lists: dict | None = None*)

Query class used specifically for Snapshot Explain operations.

where

User representation for service authentication

Type
str

includes

Dictionary with a fixed list of codes to include

Type
dict

includes_list

Dictionary with references to Lists for inclusion

Type
dict

excludes

Dictionary with a fixed list of codes to exclude

Type
dict

excludes_list

Dictionary with references to Lists for inclusion

Type
dict

get_payload() → dict

Create the basic request payload to be used within Snapshots Explain API request.

Returns
Dictionary containing non-null query attributes.

Return type
dict

16.3 SnapshotExplainJobResponse

class factiva.analytics.snapshots.explain.**SnapshotExplainJobResponse**(*job_id: str | None = None*)

Snapshot Explain Job Response class. Essentially contains the volume of estimate documents.

job_id

Explain Job ID with a format like `abcd1234-ab12-ab12-ab12-abcdef123456`.

Type
str

job_link

Unique URL referring to the job instance

Type
str

job_state

Latest known job status. Value is self-explanatory.

Type
str

volume_estimate

Number representing the total volume of matching documents

Type
int

errors

Job execution errors returned by the API

Type
list[dict]

16.4 SnapshotExplainSamplesResponse

class factiva.analytics.snapshots.explain.**SnapshotExplainSamplesResponse**(*samples_list: list*)

Snapshot Explain Samples Response class. Essentially contains the list of metadata samples randomly selected from a previously sent criteria linked to the Job ID.

num_samples

Number with the returned number of samples

Type
int

data

Pandas DataFrame with the samples dataset

Type
pandas.DataFrame

CHAPTER

SEVENTEEN

SNAPSHOT TIME SERIES

Module that handles all Factiva Analytics Time Series requests and objects. Contains classes and tools that allow to run volume volume time series with all possible variations.

17.1 SnapshotTimeSeries

class factiva.analytics.snapshots.time_series.**SnapshotTimeSeries**(*user_key=None*, *query=None*, *job_id=None*)

Main class to interact with the Time Series service from Factiva Analytics.

user_key

User representation for service authentication

Type
UserKey

query

Query object tailored for Extraction operations

Type
SnapshotExtractionQuery

job_response

Object containing job status and execution details

Type
SnapshotExtractionJobReponse

get_job_response() → bool

Performs a request to the API using the job ID to get its status.

If the job has been completed, results are assigned to the `job_response` object.

Returns
True if the get request was successful. An Exception otherwise.

Return type
bool

process_job()

Submit a new job to be processed, wait until the job is completed and then retrieves the job results.

Returns
True if the explain processing was successful. An Exception otherwise.

Return type
bool

`submit_job()`

Performs a POST request to the API using the assigned query to start a TimeSeries job.

If the job is initiated succesfully, results are assigned to the `job_response` object. Otherwise any HTTP error will raise an exception.

Returns
True if the submission was successful. An Exception otherwise.

Return type
bool

17.2 SnapshotTimeSeriesQuery

`class factiva.analytics.snapshots.time_series.SnapshotTimeSeriesQuery(`*where=None*, *includes: dict | None = None*, *include_lists: dict | None = None*, *excludes: dict | None = None*, *exclude_lists: dict | None = None*, *frequency: str = 'MONTH'*, *date_field: str = 'publication_datetime'*, *group_dimensions: list = []*, *top: int = 10*`)`

Snapshot Query for TimeSeries operations class. Used only in the context of SnapshotTimeSeries, but can be transformed to other SnapshotQuery types when those are created using an instance of this class as parameter.

`where`

User representation for service authentication

Type
str

`includes`

Dictionary with a fixed list of codes to include

Type
dict

`includes_list`

Dictionary with references to Lists for inclusion

Type
dict

`excludes`

Dictionary with a fixed list of codes to exclude

Type
dict

excludes_list

Dictionary with references to Lists for inclusion

Type
dict

frequency

Time unit used to aggregate values in the time-series calculation

Type
str

date_field

Schema date-time field used to calculate the time-series dataset

Type
str

group_dimensions

List of fields to break-down aggregates per time period unit

Type
list[str]

top

Max entries per group_dimension per time period unit

Type
str

get_payload() → dict

Create the basic request payload to be used within Snapshots Explain API request.

Returns
Dictionary containing non-null query attributes.

Return type
dict

17.3 SnapshotTimeSeriesJobReponse

class factiva.analytics.snapshots.time_series.**SnapshotTimeSeriesJobReponse**(*job_id: str | None = None*)

Snapshot Explain Job Response class. Essentially contains the volume of estimate documents.

job_id

Explain Job ID with a format like `abcd1234-ab12-ab12-ab12-abcdef123456`.

Type
str

job_link

Unique URL referring to the job instance

Type
str

`job_state`

Latest known job status. Value is self-explanatory.

Type
str

`data`

Obtained Time-Series data from job execution

Type
pandas.DataFrame

`errors`

Job execution errors returned by the API

Type
list[dict]

CHAPTER

EIGHTEEN

SNAPSHOT EXTRACTION

Module that handles all Factiva Analytics Extraction requests and objects. Contains classes and tools that allow to run extraction jobs and download the generated files.

18.1 SnapshotExtraction

class factiva.analytics.snapshots.extraction.**SnapshotExtraction**(*job_id=None*, *query=None*, *user_key=None*)

Main class to interact with the Extractions service from Factiva Analytics.

user_key

User representation for service authentication

Type
UserKey

query

Query object tailored for Extraction operations

Type
SnapshotExtractionQuery

job_response

Object containing job status and execution details

Type
SnapshotExtractionJobReponse

download_files(*path=None*)

Download all files from a job and stores them in the given path.

If the `path` parameter is empty, files are stored in a folder with the name of the job short id.

Parameters
path (*str, Optional*) – String containing the path where to store the downloaded files. If not provided, the files are stored in a folder named after the job short_id. If such folder does not exists, it is created in the current working directory.

Returns
True if files were correctly downloaded, False if no files are available for download or the download failed.

Return type
bool

`get_job_response`() → bool

Performs a request to the API to obtain an updated status of a job execution.

If the job has been completed, result details are assigned to the `job_response` object.

Returns
True if the get request was successful. An Exception otherwise.

Return type
bool

Raises
ValueError – If the Job ID doesn't exist for the user key, or the get request is invalid.

`process_job`(*path=None*)

Submit a new job to be processed, wait until the job is completed and then retrieves the job results.

Returns
True if the extraction processing was successful. An Exception otherwise.

Return type
bool

`submit_job`()

Performs a POST request to the API using the assigned values in `user_key` and `query`.

If the job is initiated succesfully, the initial status is stored in the `job_response` object. Otherwise any HTTP error will raise an exception.

Returns
True if the submission was successful. An Exception otherwise.

Return type
bool

Raises
ValueError – When the `query` is empty or invalid.

18.2 SnapshotExtractionQuery

class factiva.analytics.snapshots.extraction.**SnapshotExtractionQuery**(*where: str | None = None*, *includes: dict | None = None*, *include_lists: dict | None = None*, *excludes: dict | None = None*, *exclude_lists: dict | None = None*, *file_format: str = 'avro'*, *limit: int = 0*)

Query class used specifically for Snapshot Extraction operations.

`where`

User representation for service authentication

Type
str

includes

Dictionary with a fixed list of codes to include

Type
dict

includes_list

Dictionary with references to Lists for inclusion

Type
dict

excludes

Dictionary with a fixed list of codes to exclude

Type
dict

excludes_list

Dictionary with references to Lists for inclusion

Type
dict

file_format

Chosen file fomat for extraction files

Type
str

limit

Max number of articles to extract

Type
int

get_payload() → dict

Create the basic request payload to be used within a Snapshots Extraction API request.

Returns
Dictionary containing non-null query attributes.

Return type
dict

18.3 SnapshotExtractionJobReponse

class factiva.analytics.snapshots.extraction.**SnapshotExtractionJobReponse**(*job_id: str | None = None*, *user_key: UserKey | None = None*)

Snapshot Explain Job Response class. Essentially contains the volume of estimate documents.

job_id

Job ID returned by Factiva Analyitcs at submission time

Type
str

`short_id`

Unique portion from the attribute `job_id`

Type
str

`job_link`

Job unique URI

Type
str

`job_state`

Job status value

Type
str

`errors`

If not empty, a list of errors during the job execution

Type
list

`files`

If the job is successful, this shows the list of files that can be downloaded with the selected content.

Type
list

CHAPTER

NINETEEN

ARTICLE RETRIEVAL SERVICE

When enabled along with all other Factiva Analytics APIs, this service allows to retrieve content for display purposes when end users need to read the underlying content from a calculated score or derived datapoint after processing Snapshots or Streams news articles.

19.1 ArticleRetrieval

class factiva.analytics.article_retrieval.article_retrieval.**ArticleRetrieval**(*oauth_user: OAuthUser | None = None*)

Allows to fetch articles against the Article Retrieval Service using the provided OAuthUser credentials.

Parameters

oauth_user (OAuthUser) – An instance of an OAuthUser with working credentials. If not provided the user instance is created automatically from ENV variables.

Examples

Create an ArticleRetrieval instance.

```
from factiva.analytics import ArticleRetrieval
ar = ArticleRetrieval()
ar
```

```
<class 'factiva.analytics.article_retrieval.article_retrieval.ArticleRetrieval'>
  |-oauth_user: <class 'factiva.analytics.auth.oauthuser.OAuthUser'>
  |   |-client_id = fbwqyORz0te484RQTt0E7qj6Tooj4Cs6
  |   |-token_status = OK
  |   |-...
```

oauth_user = None

User instance wich provides the credentials to connect to the Article Retrieval API endpoints.

retrieve_single_article(*an: str*) → dict

Method that retrieves a single article to be displayed in a user interface. The requested item is initially retrieved from the . Additionally, the retrieved data is stored in the class atttribute `last_retrieval_response`.

Parameters
an (*str*) – String containing the 32-character long article ID (AN). e.g. WSJO000020221229eict000jh

Returns
Python dict containing full articles' data. This will be replaced when the `UIArticle` class is implemented.

Return type
dict

Examples

Creating a new `ArticleRetrieval` instance which reads credentials values from environment variables and retrieves a single article:

```
from factiva.analytics import ArticleRetrieval
ar = ArticleRetrieval()
article = ar.retrieve_single_article(an='WSJO000020221229eict000jh')
print(article.txt)
```

output

```
<class 'factiva.analytics.article_retrieval.article_retrieval.UIArticle'>
  |-an: WSJO000020221229eict000jh
  |-headline: Europe Taps Tech's Power-Hungry Data Centers to Heat Homes.
  |-source_code: WSJO
  |-source_name: The Wall Street Journal Online
  |-publication_date: 2022-12-29
  |-metadata: <dict> - [4] keys
  |-content: <dict> - [19] keys
  |-included: <list> - [0] items
  |-relationships: <dict> - [0] keys
```

Raises
PermissionError – If the user doesn't have access to the requested content

19.2 UIArticle

class factiva.analytics.article_retrieval.article_retrieval.**UIArticle**(*article_dict: dict*)

Class that represents a single article for visualization purposes. Methods and attributes are tailored for front-end environments.

Parameters
article_dict (*dict*) – A python dict with the structure returned by the Dow Jones Article Retrieval service.

Examples

See ArticleRetrieval class examples.

an = None

Article unique identifier, also known as Accession Number

content = {}

Article's content dict. Full text with annotations and other UI elements.

headline = None

Article's headline, also known as title

included = []

References to objects linked to a specific article

metadata = {}

Article's metadata dict. Contains Dow Jones Intelligent Identifiers among other codes.

publication_date = None

Article's publication date in ISO format as provided by the source. e.g. '2022-12-03'

relationships = {}

References to related objects

source_code = None

Article content creator's code. e.g. WSJO

source_name = None

Article content creator's name. e.g. The Wall Street Journal Online

INDEX

T

U

V

W