REFINITIV EIKON

DATA API FOR PYTHON

Eikon 4.0

REFINITIV®

DATA IS JUST
THE BEGINNING®

Contents

About this document

The Eikon Data API for Python allows your Python applications to access data directly from Eikon, powering in-house or third-party desktop apps with Refinitiv data. It provides seamless workflow with the same data across all applications running on the desktop. It leverages Eikon data and entitlements to simplify market data management and reporting.

The Eikon Data API for Python is a software library that works in conjunction with the Eikon Data API Proxy. To learn more about this Python library and the Eikon Data API Proxy, refer to the Quick Start guides, Tutorials, Documentation and other learning materials available on the Refinitiv Developer Community.

Eikon functions

TR_Field

This is a helper function to build the field for the `get_data` function.

Structure

```
TR_Field(field_name, params=None, sort_dir=None, sort_priority=None)
```

Parameters

Name	Type	Description
`field_name`	string	Field name to request. You can find the list in Data Item Browser.
`params`	dict	Dictionary containing the parameters for the field passed in the argument `field_name`
`sort_dir`	string	Indicate the sort direction. Possible values are `asc` or `desc`. The default value is `asc`.
`sort_priority`	integer	Gives a priority to the field for the sorting, where the highest priority is `0` (zero). The default value is None.

Return

Returns a dictionary that can directly passed to `get_data`.

Example

```
TR_Field('tr.revenue') TR_Field('tr.open','asc',1) TR_Field('TR.GrossProfit',{'Scale': 6, 'Curn': 'EUR'},'asc',0)
```

get_app_id

Returns the app key previously set with `set_app_id` function.

Structure

```
get_app_id()
```

Notes

The app key identifies your application on Refinitiv Platform. You can get an app key using the App Key Generator (this App is available in Eikon Desktop).

ⓘ Deprecated:: 0.1.12
This will be removed in future releases. Use get_app_key function instead

get_app_key

Returns the app key previously set.

Structure

```
get_app_key()
```

Notes

The app key identifies your application on Refinitiv Platform. You can get an application ID using the App Key Generator (this App is available in Eikon Desktop).

get_data

Returns a `pandas.DataFrame` with fields in columns and instruments as row index

Structure

```
get_data(instruments, fields, parameters=None, field_name=False, raw_output=False, debug=False)
```

Parameters

Name	Type	Description
`instruments`	string or list	Single instrument or list of instruments to request.

`fields`	string, dictionary or list of strings and/or dictionaries.	List of fields to request.
		Examples:
		`'TR.PriceClose'`
		`{'TR.GrossProfit': { 'params':{ 'Scale': 6, 'Curn': 'EUR' }}`
		`{'TR.GrossProfit': { 'params':{ 'Scale': 6, 'Curn': 'EUR' },sort_dir:'desc'}`
		`['TR.PriceClose','TR.PriceOpen']`
		`[{'TR.PriceClose': {'sort_dir':asc,sort_priority:1}},{'TR.PriceOpen': {'sort_dir':asc,sort_priority:0}}`
		You can use the function TR_Field to build the fields:
		`>>> fields = [ek.TR_Field('tr.revenue'),ek.TR_Field('tr.open','asc',1),ek.TR_Field('TR.GrossProfit',{'Scale': 6, 'Curn': 'EUR'},'asc',0)]`
		`>>> data, err = ek.get_data(["IBM","MSFT.O"],fields)`
		ⓘ You can launch the Data Item Browser to discover fields and parameters, or copy field names and parameters from TR Eikon - MS Office formulas

`parameters`	string or dictionary, optional	Single global parameter key=value or dictionary of global parameters to request. The default is None.
`field_name`	boolean, optional	Define if column headers are filled with field name or display names. If `True` value, field names will be used as column headers. Otherwise, the full display name will be used. The default is `False`.
`raw_output`	boolean, optional	By default, the output is a `pandas.DataFrame`. Set `raw_output=True` to get data in JSON format. The default is `False`.
`debug`	bool	When set to `True`, the JSON request and response are printed. The default value is `False`.

Returns

pandas.DataFrame

Returns `pandas.DataFrame` with fields in columns and instruments as row index

errors

Returns a list of errors

Raises

Exception

If http request fails or if server returns an error.

ValueError

If a parameter type or value is wrong.

Examples

```
>>> import eikon as ek
>>> ek.set_app_key('set your app key here')
>>> data, err = ek.get_data(["IBM", "GOOG.O", "MSFT.O"], ["TR.PriceClose", "TR.Volume",
"TR.PriceLow"])
>>> data, err = ek.get_data("IBM", ['TR.Employees', {'TR.GrossProfit':{'params':{'Scale': 6,
'Curn': 'EUR'},'sort_dir':'asc'}}])
>>> fields =
[ek.TR_Field('tr.revenue'),ek.TR_Field('tr.open',None,'asc',1),ek.TR_Field('TR.GrossProfit',{'Sca
le': 6, 'Curn': 'EUR'},'asc',0)]
>>> data, err = ek.get_data(["IBM","MSFT.O"],fields)
```

get_news_headlines

Returns a list of news headlines

Structure

```
get_news_headlines(query='Topic:TOPALL and Language:LEN', count=10,
date_from=None, date_to=None, raw_output=False, debug=False)
```

Parameters

Name	Type	Description
`query`	string, optional	News headlines search criteria. The text can contain instrument codes, company names, country names, and operators (AND, OR, NOT, IN, parentheses and quotes for explicit search). ⓘ Prefix instrument names with **R** to improve performance. The default is Top News written in English
`count`	int, optional	Max number of headlines retrieved. Value Range: `1-100`. The default is `10`.
`date_from`	string or datetime, optional	Beginning of date range. The string format is `%Y-%m-%dT%H:%M:%S`, for example, `2016-01-20T15:04:05`.

Name	Type	Description
date_to	string or datetime, optional	End of date range. The string format is %Y-%m-%dT%H:%M:%S, for example, 2016-01-20T15:04:05.
raw_output	boolean, optional	Set this parameter to True to get the data in JSON format. If set to False, the function returns a data frame. The default is False.
debug	bool, optional	When set to True, the json request and response are printed. The default is False.

Returns

pandas.DataFrame

- Returns a DataFrame of news headlines with the following columns:
- Index: Timestamp of the publication time
- version_created: Date of the latest update on the news
- text: Text of the Headline
- story_id: Identifier to be used to retrieve the full story using the get_news_story function
- source_code: Second news identifier

Raises

Exception

If http request fails or if server returns an error.

AttributeError

If a parameter type is wrong.

Examples

```
>>> import eikon as ek
>>> ek.set_app_key('set your app key here')
>>> headlines = ek.get_news_headlines("R:MSFT.O", 2)
>>> headlines
versionCreated                                    text
2016-04-13 18:28:57.000 2.016-04-13 18:28:59.001 RBC Applies Blockchain as a Loyalty
Boost...
2016-04-13 17:28:21.577 2016-04-13 17:28:21.671 UPDATE 2-Long-stalled email privacy bill
...
                          storyId
2016-04-13 18:28:57.000    urn:newsml:reuters.com:20160413:nNRA1uxh03:1
2016-04-13 17:28:21.577    urn:newsml:reuters.com:20160413:nL2N17G16Q:2
```

```
>>> headlines = ek.get_news_headlines("R:MSFT.O IN FRANCE")
>>> headlines = ek.get_news_headlines("R:MSFT.O IN FRANCE IN ENGLISH", count=5)
>>> headlines = ek.get_news_headlines("OBA* OR CLINTON IN ENGLISH", count=5)
```

get_news_story

Return a single news story corresponding to the identifier provided in `story_id`

Structure

```
get_news_story(story_id, raw_output=False, debug=False)
```

Parameters

Name	Type	Description
`story_id`	string	The story ID is a field you find in every headline you retrieved using the `get_news_headlines` function.
`raw_output`	boolean	Set this parameter to `True` to get the data in JSON format. If set to `False`, the function returns the story content. The default value is `False`.
`debug`	bool	When set to `True`, the JSON request and response are printed.

Raises

Exception

If http request fails or if Refinitiv Services return an error.

ValueError

If a parameter type or value is wrong.

Examples

```
>>> import eikon as ek
>>> ek.set_app_key('set your app key here')
>>> headlines = ek.get_news_headlines('IBM')
>>> for index, headline_row in headlines.iterrows():
       story = ek.get_news_story(headline_row['storyId'])
       print (story)
```

get_port_number

Returns the port number used to communicate with the Eikon Data API Proxy

Structure

```
get_port_number()
```

get_symbology

Returns a list of instrument names converted into another instrument code. For example: convert SEDOL instrument names to RIC names

Structure

```
get_symbology(symbol, from_symbol_type='RIC', to_symbol_type=None,
raw_output=False, debug=False, bestMatch=True)
```

Parameters

Name	Type	Description
`symbol`	string or list of strings	Single instrument or list of instruments to convert.
`from_symbol_type`	string	Instrument code to convert from. Possible values: `CUSIP`, `ISIN`, `SEDOL`, `RIC`, `ticker`, `lipperID`, and `IMO`. The default is `RIC`.
`to_symbol_type`	string or list	Instrument code to convert to. Possible values: `CUSIP`, `ISIN`, `SEDOL`, `RIC`, `ticker`, `lipperID`, `IMO`, and `OAPermID`. The default is None (means all symbol types are requested)
`raw_output`	boolean	Set this parameter to `True` to get the data in JSON format. If set to `False`, the function returns a data frame. The default is `False`.
`debug`	bool	When set to `True`, the JSON request and response are printed.
`bestMatch`	bool	When set to `True`, only primary symbol is requested. When set to `False`, all symbols are requested.

Returns

If `raw_output` is set to `True`, the data is returned in the JSON format. If `raw_output` is `False` (default value) the data is returned as a `pandas.DataFrame`.

pandas.DataFrame content:

- columns : Symbol types
- rows : Symbol requested
- cells : the symbols (None if not found)
- symbol : The requested symbol

Raises

Exception

If request fails or if server returns an error

ValueError

If a parameter type or value is wrong

Examples

```
>>> import eikon as ek
>>> ek.set_app_key('set your app key here')
>>> ISIN_codes = ek.get_symbology(["MSFT.O", "GOOG.O", "IBM.N"], from_symbol_type="RIC",
to_symbol_type="ISIN")
>>> ISIN_codes
                ISIN
MSFT.O  US5949181045
GOOG.O  US02079K1079
IBM.N   US4592001014
```

get_timeout

Returns the request timeout in seconds

Structure

```
get_timeout()
```

get_timeseries

Returns historical data on one or several instruments

Structure

```
get_timeseries(rics, fields='*', start_date=None, end_date=None, interval='daily',
count=None, calendar=None, corax=None, normalize=False, raw_output=False,
debug=False)
```

Parameters

Name	Type	Description
`rics`	string or list of strings	Single RIC or list of RICs to retrieve historical data for
`start_date`	string or datetime.datetime or datetime.timedelta	Starting date and time of the historical range. The string format is `%Y-%m-%dT%H:%M:%S`, for example, `2016-01-20T15:04:05`. `datetime.timedelta` is negative number of day relative to `datetime.now()`. The default is `datetime.now()` + `timedelta(-100)`. You can use the helper function `get_date_from_today`. See the Examples section for usage.

`end_date`	string or datetime.datetime or datetime.timedelta	End date and time of the historical range.
		Possible string formats:
		`%Y-%m-%d`, for example, `2017-01-20`
		`%Y-%m-%dT%H:%M:%S`, for example, `2017-01-20T15:04:05`
		`datetime.timedelta` is negative number of day relative to `datetime.now()`.
		The default is `datetime.now()`.
		You can use the helper function `get_date_from_today`. See the Examples section for usage.

`interval`	string	Data interval. Possible values are `tick`, `minute`, `hour`, `daily`, `weekly`, `monthly`, `quarterly`, `yearly`. The default is `daily`.
`fields`	string or list of strings	Use this parameter to filter the returned fields set. Available fields: `TIMESTAMP`, `VALUE`, `VOLUME`, `HIGH`, `LOW`, `OPEN`, `CLOSE`, `COUNT`. By default, all fields are returned.
`count`	int, optional	Maximum number of data points retrieved.
`calendar`	string, optional	Possible values: `native`, `tradingdays`, `calendardays`.
`corax`	string, optional	Possible values: `adjusted`, `unadjusted`

Name	Type	Description
normalize	boolean	If set to True, the function returns a normalized data frame with the following columns Date, Security, Field. If the value of this parameter is False, the returned data frame shape depends on the number of RICs and the number of fields in the response There are three different shapes: • One RIC and many fields • Many RICs and one field • Many RICs and many fields The default is False. ⓘ This parameter has less precedence than the parameter rawOutput. That is, if rawOutput is set to True, the returned data is the raw data and this parameter is ignored.
raw_output	boolean	Set this parameter to True to get the data in JSON format. If set to False, the function returns a data frame which shape is defined by the normalize parameter. The default is False.
debug	bool	When set to True, the JSON request and response are printed. The default is False.

Raises

Exception

If request fails or if server returns an error.

ValueError

If a parameter type or value is wrong.

Examples

```
>>> import eikon as ek
>>> ek.set_app_key('set your app key here')
>>> req = ek.get_timeseries(["MSFT.O"], start_date = "2017-02-01T15:04:05",
>>>                         end_date = "2017-02-05T15:04:05", interval="tick")
>>> req = ek.get_timeseries(["MSFT.O"], start_date = "2017-03-01",
>>>                         end_date = "2017-03-10", interval="daily")
>>> req = ek.get_timeseries(["MSFT.O"], start_date = get_date_from_today(150),
>>>                         end_date = get_date_from_today(100), interval="daily")
```

send_json_request

Returns the JSON response. This function can be used for advanced usage or early access to new features.

Structure

```
send_json_request(entity, payload, debug=False)
```

Parameters

Name	Type	Description
entity	string	A string containing a service name
payload	string	A string containing a JSON request

Name	Type	Description
debug	bool, optional	When set to `True`, the json request and response are printed. The default is `False`.

Returns

string

The JSON response as a string

Raises

EikonError

If daemon is disconnected

requests.Timeout

If request times out

Exception

If request fails (HTTP code other than 200)

EikonError

If daemon is disconnected

set_app_id

Set the app key.

Structure

```
set_app_id(app_key)
```

Parameters

Name	Type	Description
app_key	string	the app key

Notes

The app key identifies your application on Refinitiv Platform. You can get an app key using the App Key Generator (this App is available in Eikon Desktop).

ⓘ Deprecated:: 0.1.12
This will be removed in 1.1.0. Use set_app_key function instead.

set_app_key

Set the app key.

Structure

```
set_app_key(app_key)
```

Parameters

Name	Type	Description
app_key	string	the app key

Notes

The app key identifies your application on Refinitiv Platform. You can get an app key using the App Key Generator (this App is available in Eikon Desktop).

set_log_level

Set the log level. When logs are activated (`log_level != logging.NOTSET`), log files are created in the current directory. To change directory for log files, set log path with `set_log_path()` function.

Structure

```
set_log_level(level)
```

Parameters

Name	Type	Description
level	int	Possible values from logging module: CRITICAL, FATAL, ERROR, WARNING, WARN, INFO, DEBUG, NOTSET

Example

```
ek.set_log_level(logging.DEBUG)
```

set_log_path

Set the filepath of the log file.

Structure

```
set_log_path(path)
```

Parameters

Name	Type	Description
path	string	File path location for log files

Example

```
ek.set_log_path("c:\my_directory")
```

set_port_number

Set the port number to communicate with the Eikon Data API proxy. This port number is detected automatically but you can call this function to force it manually for troubleshooting issues.

Structure

```
set_port_number(port_number)
```

Parameters

Name	Type	Description
port_number	int	the port number

set_timeout

Set the timeout for each request.

Structure

```
set_timeout(timeout)
```

Parameters

Name	Type	Description
timeout	int	the request timeout in sec Default value: 30 sec

Eikon classes

EikonError

Base class for exceptions specific to Eikon platform.

Constructor

```
__init__(self, code, message)
```

Parameters

Name	Type	Description
code	int	
message	string	Indicate the sort direction. Possible values are asc or desc. The default value is asc.