Search projects

Help Sponsors Log in Register

iexfinance 0.5.0

```
pip install iexfinance
```

✓ Latest version

Released: Dec 11, 2020

Python module to get stock data from IEX Cloud and IEX API 1.0

Navigation

Project description

Release history

Download files

Project links

Homepage

Download

Statistics

GitHub statistics:

Project description

build failing | codecov 72% | pypi package 0.5.0 | License Apache 2.0

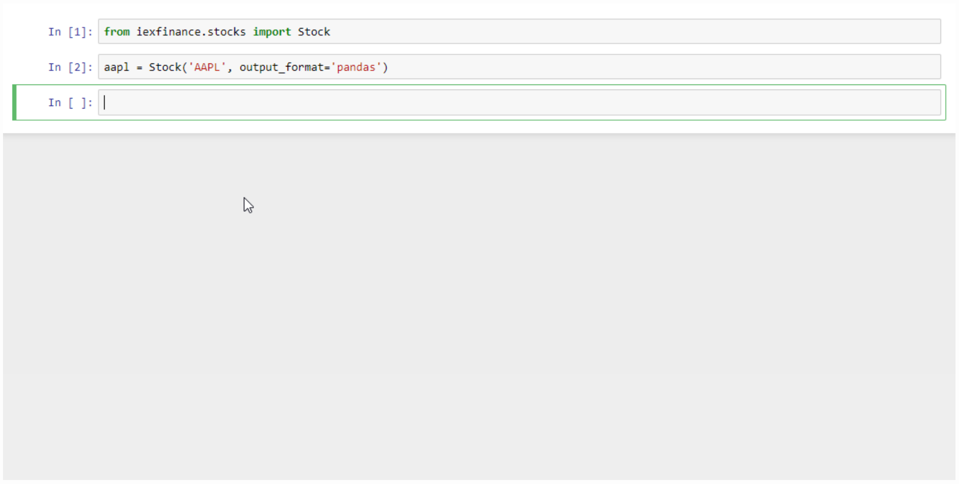
Python SDK for IEX Cloud. Architecture mirrors that of the IEX Cloud API (and its documentation).

An easy-to-use toolkit to obtain data for Stocks, ETFs, Mutual Funds, Forex/Currencies, Options, Commodities, Bonds, and Cryptocurrencies:

- Real-time and delayed quotes
- Historical data (daily and minutely)
- Financial statements (Balance Sheet, Income Statement, Cash Flow)
- End of Day Options Prices
- Institutional and Fund ownership
- Analyst estimates, Price targets
- Corporate actions (Dividends, Splits)
- Sector performance
- Market analysis (gainers, losers, volume, etc.)

- IEX market data & statistics (IEX supported/listed symbols, volume, etc)
- Social Sentiment and CEO Compensation

Example



```
In [1]: from iexfinance.stocks import Stock
In [2]: aapl = Stock('AAPL', output_format='pandas')
In [ ]: |
```

Documentation

Stable documentation is hosted on github.io.

Development documentation is also available for the latest changes in master.

Install

From PyPI with pip (latest stable release):

`$ pip3 install iexfinance`

From development repository (dev version):

```
$ git clone https://github.com/addisonlynch/iexfinance.g
$ cd iexfinance
$ python3 setup.py install
```

What’s Needed to Access IEX Cloud?

Stars: 644

Forks: 137

Open issues: 27

Open PRs: 3

View statistics for this project via Libraries.io, or by using our public dataset on Google BigQuery

Meta

License: Apache Software License (Apache)

Author: Addison Lynch

stocks, market, finance, iex, quotes, shares, currency

Maintainers

addisonlynch

Classifiers

Development Status

- 4 - Beta

Intended Audience

- Developers
- Financial and Insurance Industry

License

- OSI Approved :: Apache Software License

Operating System

- OS Independent

Programming Language

- Python
- Python :: 3
- Python :: 3.4
- Python :: 3.5
- Python :: 3.6

Topic

- Office/Business :: Financial :: Investment
- Software Development :: Libraries :: Python Modules

An IEX Cloud account is required to acecss the IEX Cloud API. Various plans are availalbe, free, paid, and pay-as-you-go.

Your IEX Cloud (secret) authentication token can be passed to any function or at the instantiation of a `Stock` object. The easiest way to store a token is in the `IEX_TOKEN` environment variable.

Passing as an Argument

The authentication token can also be passed to any function call:

```
from iexfinance.refdata import get_symbols

get_symbols(token="<YOUR-TOKEN>")
```

or at the instantiation of a `Stock` object:

```
from iexfinance.stocks import Stock

a = Stock("AAPL", token="<YOUR-TOKEN>")
a.get_quote()
```

How This Package is Structured

`iexfinance` is designed to mirror the structure of the IEX Cloud API. The following IEX Cloud endpoint groups are mapped to their respective `iexfinance` modules:

The most commonly-used endpoints are the Stocks endpoints, which allow access to various information regarding equities, including quotes, historical prices, dividends, and much more.

The `Stock` object provides access to most endpoints, and can be instantiated with a symbol or list of symbols:

```
from iexfinance.stocks import Stock
```

```
aapl = Stock("AAPL")
aapl.get_balance_sheet()
```

The rest of the package is designed as a 1:1 mirror. For example, using the Alternative Data endpoint group, obtain the Social Sentiment endpoint with `iexfinance.altdata.get_social_sentiment`:

```
from iexfinance.altdata import get_social_sentiment

get_social_sentiment("AAPL")
```

Common Usage Examples

The iex-examples repository provides a number of detailed examples of iexfinance usage. Basic examples are also provided below.

Real-time Quotes

To obtain real-time quotes for one or more symbols, use the `get_price` method of the `Stock` object:

```
from iexfinance.stocks import Stock
tsla = Stock('TSLA')
tsla.get_price()
```

or for multiple symbols, use a list or list-like object (Tuple, Pandas Series, etc.):

```
batch = Stock(["TSLA", "AAPL"])
batch.get_price()
```

Historical Data

It’s possible to obtain historical data using `get_historical_data` and `get_historical_intraday`.

Daily

To obtain daily historical price data for one or more symbols, use the `get_historical_data` function. This will return a daily time-series of the ticker requested over the desired date range (`start` and `end` passed as `datetime.datetime` objects):

```
from datetime import datetime
from iexfinance.stocks import get_historical_data

start = datetime(2017, 1, 1)
end = datetime(2018, 1, 1)

df = get_historical_data("TSLA", start, end)
```

To obtain daily closing prices only (reduces message count), set `close_only=True`:

```
df = get_historical_data("TSLA", "20190617", close_only=
```

For Pandas DataFrame output formatting, pass `output_format`:

```
df = get_historical_data("TSLA", start, end, output_form
```

It's really simple to plot this data, using matplotlib:

```
import matplotlib.pyplot as plt

df.plot()
plt.show()
```

Minutely (Intraday)

To obtain historical intraday data, use `get_historical_intraday` as follows. Pass an optional `date` to specify a date within three months prior to the current day (default is current date):

```
from datetime import datetime
from iexfinance.stocks import get_historical_intraday

date = datetime(2018, 11, 27)

get_historical_intraday("AAPL", date)
```

or for a Pandas Dataframe indexed by each minute:

```
get_historical_intraday("AAPL", output_format='pandas')
```

Fundamentals

Financial Statements

Balance Sheet

```
from iexfinance.stocks import Stock

aapl = Stock("AAPL")
aapl.get_balance_sheet()
```

Income Statement

```
aapl.get_income_statement()
```

Cash Flow

```
aapl.get_cash_flow()
```

Modeling/Valuation Tools

[Analyst Estimates](#)

```
from iexfinance.stocks import Stock

aapl = Stock("AAPL")

aapl.get_estimates()
```

[Price Target](#)

```
aapl.get_price_target()
```

Social Sentiment

```
from iexfinance.altdata import get_social_sentiment
get_social_sentiment("AAPL")
```

CEO Compensation

```
from iexfinance.altdata import get_ceo_compensation
get_ceo_compensation("AAPL")
```

Fund and Institutional Ownership

```
from iexfinance.stocks import Stock
aapl = Stock("AAPL")

# Fund ownership
aapl.get_fund_ownership()

# Institutional ownership
aapl.get_institutional_ownership()
```

Reference Data

List of Symbols IEX supports for API calls

```
from iexfinance.refdata import get_symbols

get_symbols()
```

List of Symbols IEX supports for trading

```
from iexfinance.refdata import get_iex_symbols

get_iex_symbols()
```

Account Usage

Message Count

```
from iexfinance.account import get_usage

get_usage(quota_type='messages')
```

API Status

IEX Cloud API Status

```
from iexfinance.account import get_api_status

get_api_status()
```

Configuration

Output Formatting

By default, `iexfinance` returns data for most endpoints in a pandas `DataFrame`.

Selecting `json` as the output format returns data formatted *exactly* as received from the IEX Endpoint. Configuring `iexfinance`'s output format can be done in two ways:

Environment Variable (Recommended)

For persistent configuration of a specified output format, use the environment variable `IEX_OUTPUT_FORMAT`. This value will be overridden by the `output_format` argument if it is passed.

macOS/Linux

Type the following command into your terminal:

```
$ export IEX_OUTPUT_FORMAT=pandas
```

Windows

See here for instructions on setting environment variables in Windows operating systems.

`output_format` Argument

Pass `output_format` as an argument to any function call:

```
from iexfinance.refdata import get_symbols

get_symbols(output_format='pandas').head()
```

or at the instantiation of a `Stock` object:

```
from iexfinance.stocks import Stock

aapl = Stock("AAPL", output_format='pandas')
aapl.get_quote().head()
```

Contact

Email: ahlshop@gmail.com

Twitter: alynchfc

License

Help

Installing packages

Uploading packages

User guide

Project name retention

FAQs

About PyPI

PyPI on Twitter

Infrastructure dashboard

Statistics

Logos & trademarks

Our sponsors

Contributing to PyPI

Bugs and feedback
Contribute on GitHub
Translate PyPI
Sponsor PyPI
Development credits

Using PyPI

Code of conduct
Report security issue
Privacy policy
Terms of use
Acceptable Use Policy

Status: Service Under Maintenance

Developed and maintained by the Python community, for the Python community.
Donate today!

Switch to desktop version

> English español français 日本語 português (Brasil) українська Ελληνικά Deutsch 中文 (简体) 中文 (繁體) русский עברית esperanto