Extract Stock Quotes with

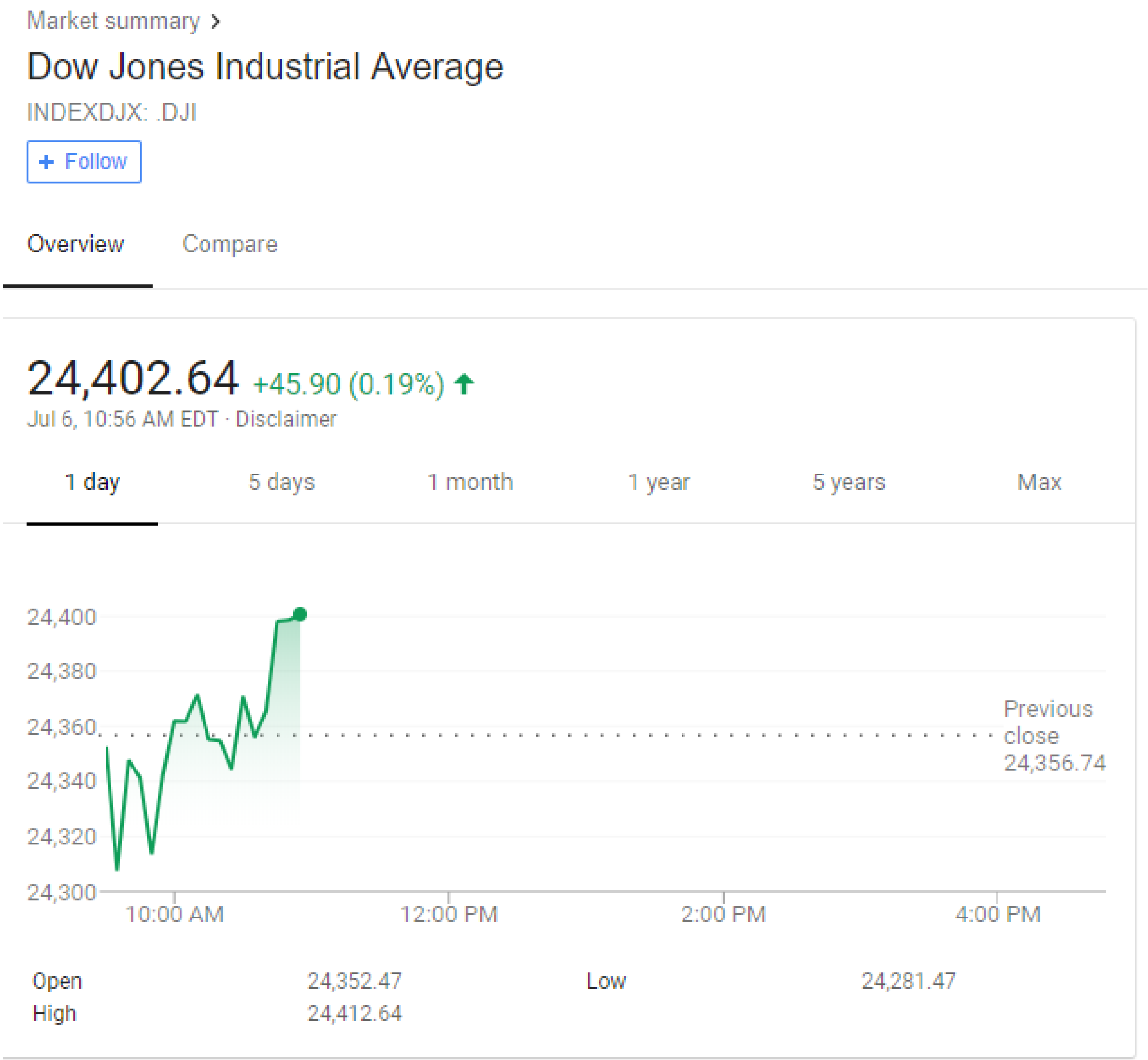
Google/Yahoo/Quandl Finance API

Report for Python Code Challenges: Extract Stock Quotes with Finance APIs

July 06, 2018

Project Object

Stocks represent fractional ownership of a company. Stocks are traded on an exchange like the New York Stock Exchange (NYSE), NASDAQ, London Stock Exchange (LSE), and more. You can buy and sell stock through brokers.



Market summary >

Dow Jones Industrial Average

INDEXDJX: .DJI

24,402.64 +45.90 (0.19%)

Jul 6, 10:56 AM EDT · Disclaimer

Previous close 24,356.74

Open	24,352.47	Low	24,281.47
High	24,412.64		

This Python project will focus on using Google Finance Client API, Yahoo Financials API and Quandl API to extract five stocks' data in real-time and dynamically visualize the stock tracking figures. The stocks are QCOM, CSCO, INTC, AAPL and GOOG.

Keywords

Python, stock quotes, google finance api, yahoo finance api, quandl api, real-time tracking stock data

Use Cases

For many, tracking stock data is a hobby. Stock price is widely considered to be an indicator of how companies are performing. If you're interested in the success of a company, you might keep an eye on their stock price to gauge performance.

For others, investing in stocks is a way to plan for retirement. Many people purchase stocks, bonds, and mutual funds expecting them to increase in value over time and fund their non-working years.

Most importantly: stock prices are always changing. If you want to monitor stock prices, you need an easy way to keep your data up to date.

No matter why you're tracking stock data, those three API are the easiest way to get data on stocks.

Datasets

Data on stocks from Google Finance, Yahoo Finance and Quandl

Google Finance Client API

The shortest interval this API provides is 1 minute, so we send our request 60s a time which the same as the shortest interval that google finance provides.

In this project, the trend line of a stock will start draw from today's stock opening time

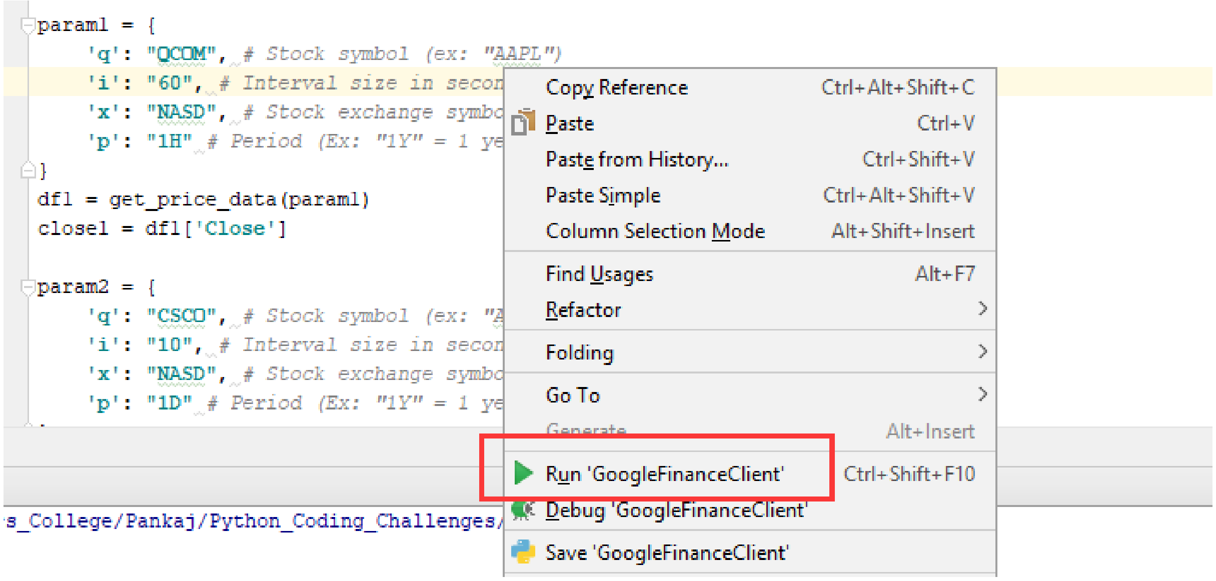
1. Step to Run the Program

1) Open a terminal, do:

$ pip install googlefinance.client

2) Suggest use Pycharm run the GoogleFinanceClient.py:

Right-click the program then click GoogleFinanceClient.py



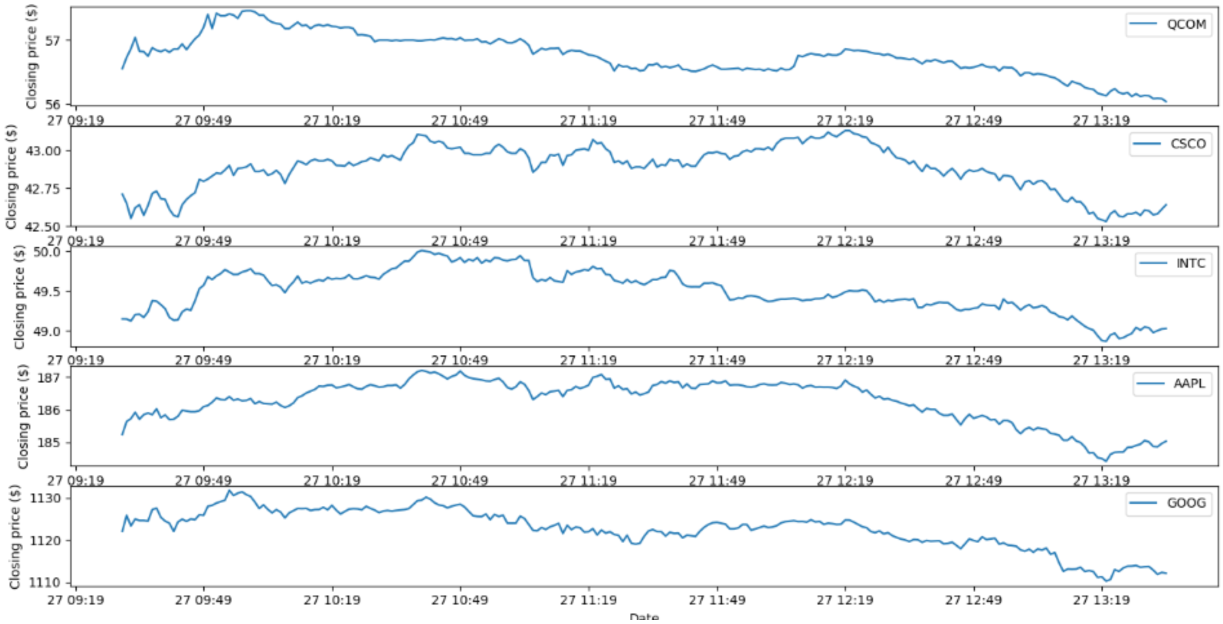
```
param1 = {
    'q': "QCOM", # Stock symbol (ex: "AAPL")
    'i': "60", # Interval size in secon
    'x': "NASD", # Stock exchange symbo
    'p': "1H" # Period (Ex: "1Y" = 1 ye
}
df1 = get_price_data(param1)
close1 = df1['Close']

param2 = {
    'q': "CSCO", # Stock symbol (ex: "A
    'i': "10", # Interval size in secon
    'x': "NASD", # Stock exchange symbo
    'p': "1D" # Period (Ex: "1Y" = 1 ye
```

```
s_College/Pankaj/Python_Coding_Challenges/
```

Copy Reference Ctrl+Alt+Shift+C
Paste Ctrl+V
Paste from History... Ctrl+Shift+V
Paste Simple Ctrl+Alt+Shift+V
Column Selection Mode Alt+Shift+Insert
Find Usages Alt+F7
Refactor
Folding
Go To
Generate Alt+Insert
Run 'GoogleFinanceClient' Ctrl+Shift+F10
Debug 'GoogleFinanceClient'
Save 'GoogleFinanceClient'

2. Final Output

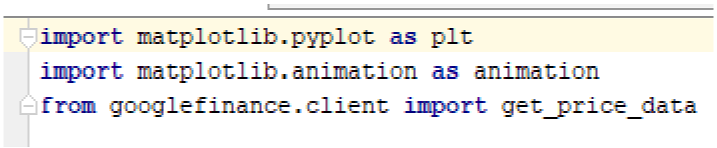
The output will be a figure that shows five stock quotes within a day.



3. The Code

“matplotlib” package is used to visualize the stock data (pyplot) and send request in real-time (animation)

“googlefinance.client” is used to get the stock data from Google Finance.



```
import matplotlib.pyplot as plt
import matplotlib.animation as animation
from googlefinance.client import get_price_data
```

Packages used in our program

Define a function named “update_stock” that used to request the stock data, “get_price_data” method is used here. “param1” is the parameters you need to provide to the API, in order to get the response stock data.

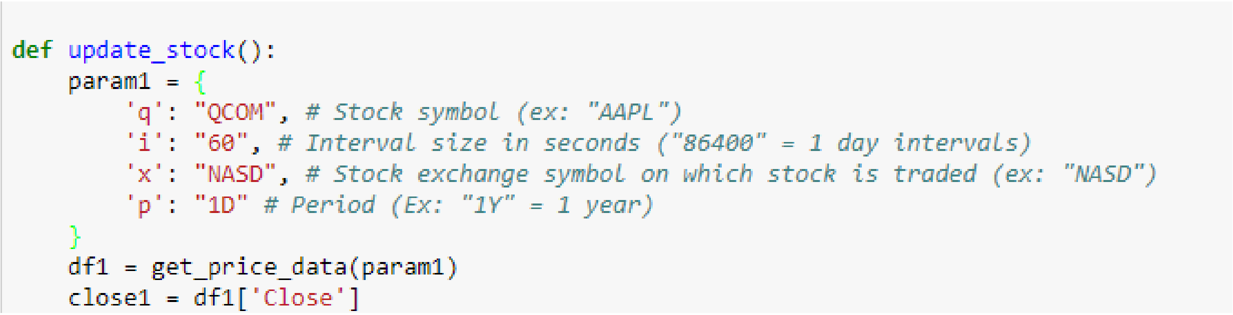
‘q’ is the stock symbol

‘i’ is the interval size in seconds, must be a multiple of 60, like 60, 120….

‘x’ is the stock exchange symbol on which stock is traded

‘p’ is the period for stock data. Can be “1M” which is one month or “1Y” which is one year

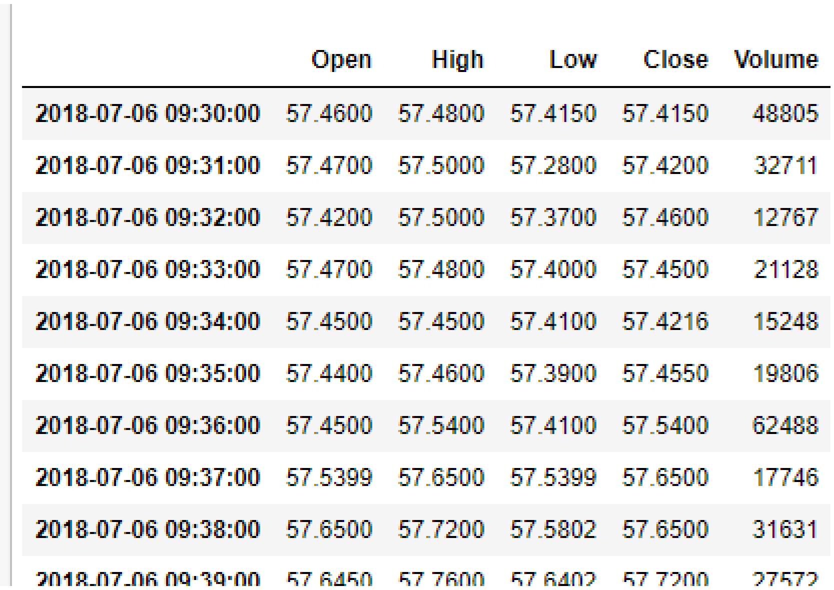
Reference Link: https://pypi.org/project/googlefinance.client/



```
def update_stock():
    param1 = {
        'q': "QCOM", # Stock symbol (ex: "AAPL")
        'i': "60", # Interval size in seconds ("86400" = 1 day intervals)
        'x': "NASD", # Stock exchange symbol on which stock is traded (ex: "NASD")
        'p': "1D" # Period (Ex: "1Y" = 1 year)
    }
    df1 = get_price_data(param1)
    close1 = df1['Close']
```

Function to get today’s stock data of QCOM

The response dataset can be put in a pandas DataFrame



	Open	High	Low	Close	Volume
2018-07-06 09:30:00	57.4600	57.4800	57.4150	57.4150	48805
2018-07-06 09:31:00	57.4700	57.5000	57.2800	57.4200	32711
2018-07-06 09:32:00	57.4200	57.5000	57.3700	57.4600	12767
2018-07-06 09:33:00	57.4700	57.4800	57.4000	57.4500	21128
2018-07-06 09:34:00	57.4500	57.4500	57.4100	57.4216	15248
2018-07-06 09:35:00	57.4400	57.4600	57.3900	57.4550	19806
2018-07-06 09:36:00	57.4500	57.5400	57.4100	57.5400	62488
2018-07-06 09:37:00	57.5399	57.6500	57.5399	57.6500	17746
2018-07-06 09:38:00	57.6500	57.7200	57.5802	57.6500	31631
2018-07-06 09:39:00	57.6450	57.7600	57.6402	57.7200	27572

Response stock data which saved in a DataFrame

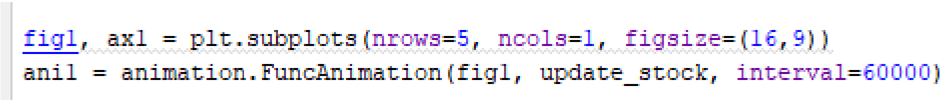
"plt.subplots" used here to draw the stock tracking figure. "animation.FuncAnimation" can allow the program to send the request every 60 seconds.

"nrows=5": five subplots in the figure

"ncols=1": only one column in the figure

"figsize=(16,9)": define the size of the figure

"interval = 60000": send the request every 60 seconds, it is in milli-second



```
fig1, ax1 = plt.subplots(nrows=5, ncols=1, figsize=(16,9))
ani1 = animation.FuncAnimation(fig1, update_stock, interval=60000)
```

Dynamically visualize the stock trend

Yahoo Financials API

Once you send a stock quote request using this API, it will take some time to receive the response. The response time for every stock is different, some are fast, some are slow. The five stocks are chosen in this program will take about 40 seconds to renew all the stock quotes every time.

The trend line of a stock will start draw from the program running point

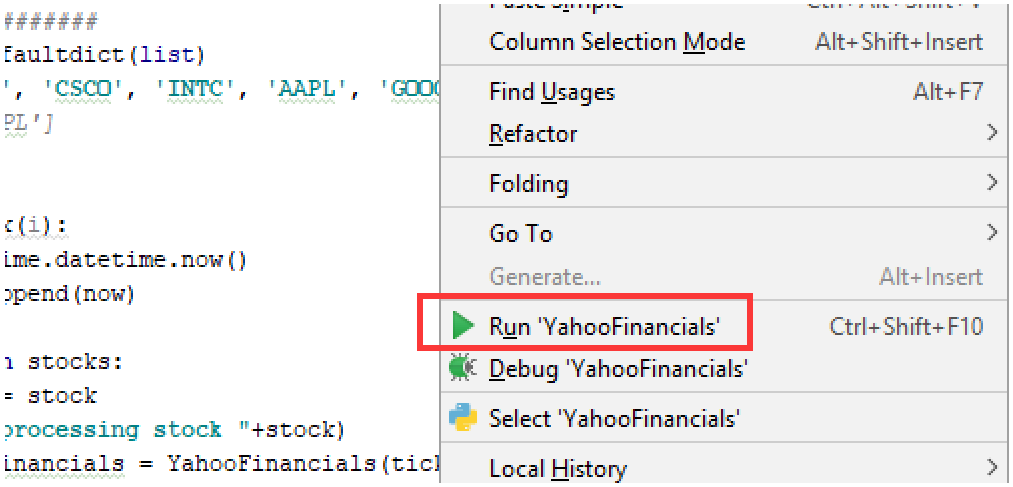
1. Step to Run the Program

1) Open a terminal, do:

$ pip install yahoofinancials

2 Suggest use Pycharm run the YahooFinancials.py:

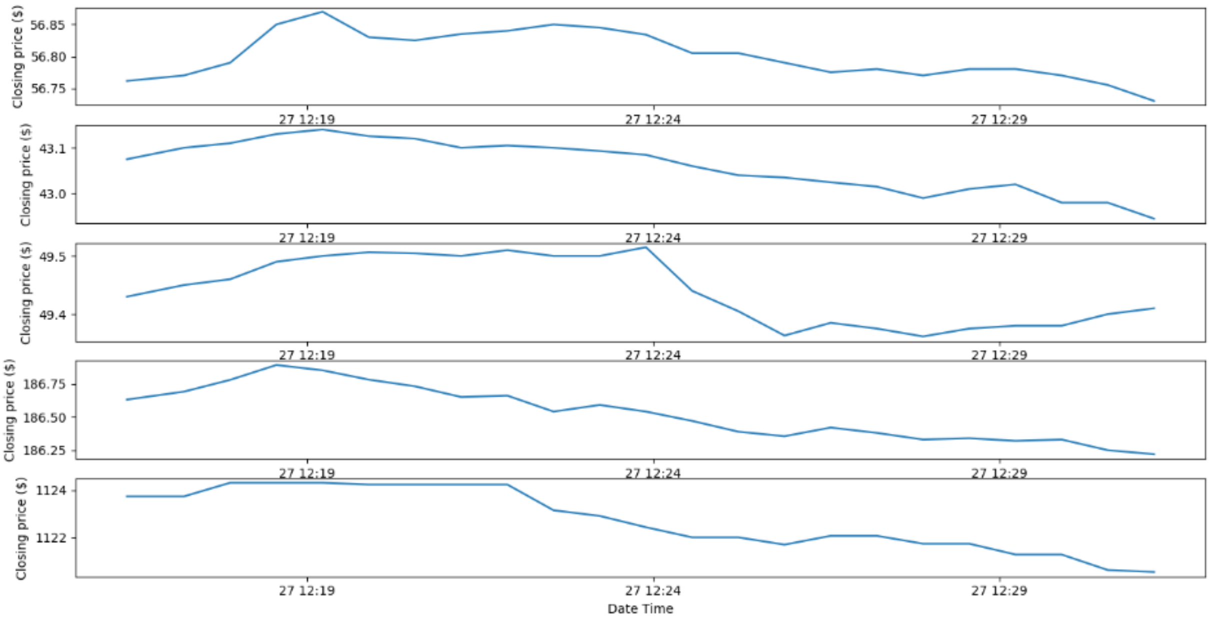
Right-click the program then click YahooFinancials.py



```
#######
faultdict(list)
', 'CSCO', 'INTC', 'AAPL', 'GOO
PL']


c(i):
ime.datetime.now()
ppend(now)

n stocks:
= stock
processing stock "+stock)
inancials = YahooFinancials(tic
```

2. Final Output

The output will be a figure that shows five stock quotes from the program start running time.



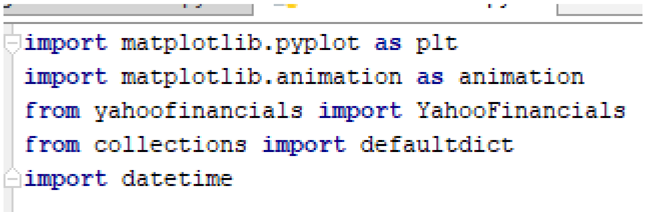
3. The Code

"matplotlib" package is used to visualize the stock data (pyplot) and send request in real-time (animation)

"yahoofinancials" is used to get the stock data from Yahoo Finance

"collections.defaultdict" is used to create a dictionary to store the quotes data of every stock

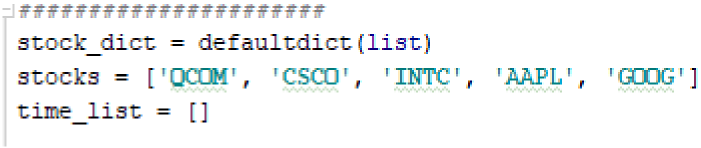
"datatime" is used to get the time of sending the request



```
import matplotlib.pyplot as plt
import matplotlib.animation as animation
from yahoofinancials import YahooFinancials
from collections import defaultdict
import datetime
```

Packages used in our program

New a dictionary named "stock_dict" which key is the stock symbol and the value is a list of the response stock quotes. "time_list" is used to store the time of the stock quotes.

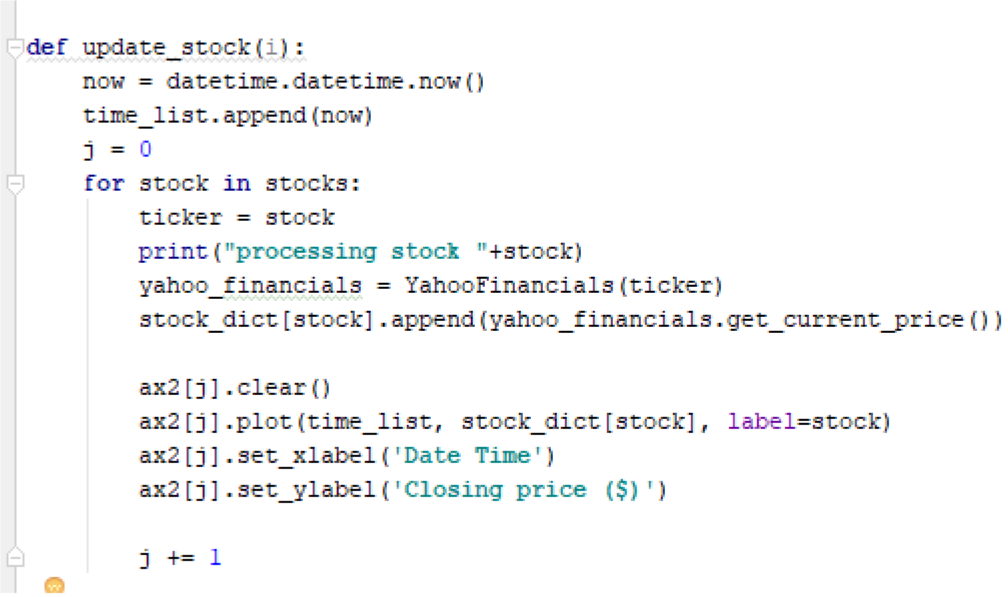


```
######################
stock_dict = defaultdict(list)
stocks = ['QCOM', 'CSCO', 'INTC', 'AAPL', 'GOOG']
time_list = []
```

Variables defined

Function "update_stock" firstly get the current time and store it in the "time_list". Then use a for loop to send the request to get the current price for every stock, save to the dictionary. "time_list" is the x axis of the output figure and stock price is the y axis.

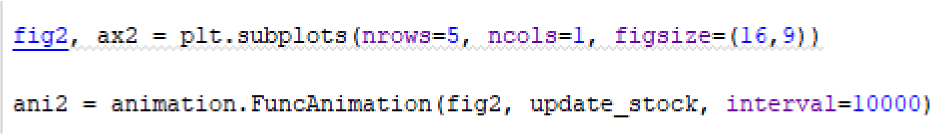
Reference Link: https://github.com/JECSand/yahoofinancials



```
def update_stock(i):
    now = datetime.datetime.now()
    time_list.append(now)
    j = 0
    for stock in stocks:
        ticker = stock
        print("processing stock "+stock)
        yahoo_financials = YahooFinancials(ticker)
        stock_dict[stock].append(yahoo_financials.get_current_price())

        ax2[j].clear()
        ax2[j].plot(time_list, stock_dict[stock], label=stock)
        ax2[j].set_xlabel('Date Time')
        ax2[j].set_ylabel('Closing price ($)')

        j += 1
```

Function to request stock quotes and visualization

“plt.subplots” used here to draw the stock tracking figure. “animation.FuncAnimation” can allow the program to send the request every 10 seconds.



```
fig2, ax2 = plt.subplots(nrows=5, ncols=1, figsize=(16,9))

ani2 = animation.FuncAnimation(fig2, update_stock, interval=10000)
```

Dynamically visualize the stock trend

Quandl API

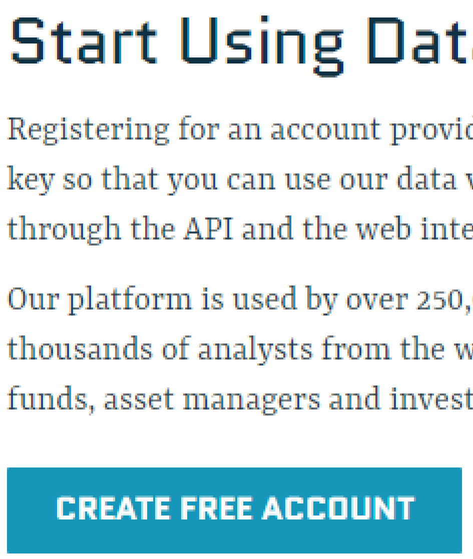
Quandl is only used for extract daily stock quotes, so this cannot update the stock quotes in real-time.

1. Step to Run the Program

1) Open a terminal, do:

$ pip install quandl

2) Go to https://www.quandl.com/ and click CREATE FREE ACCOUNT, then you will get a key that can let you to use the Quandl API:



Start Using Data

Registering for an account provides you with an API key so that you can use our data via all tools, directly through the API and the web interface.

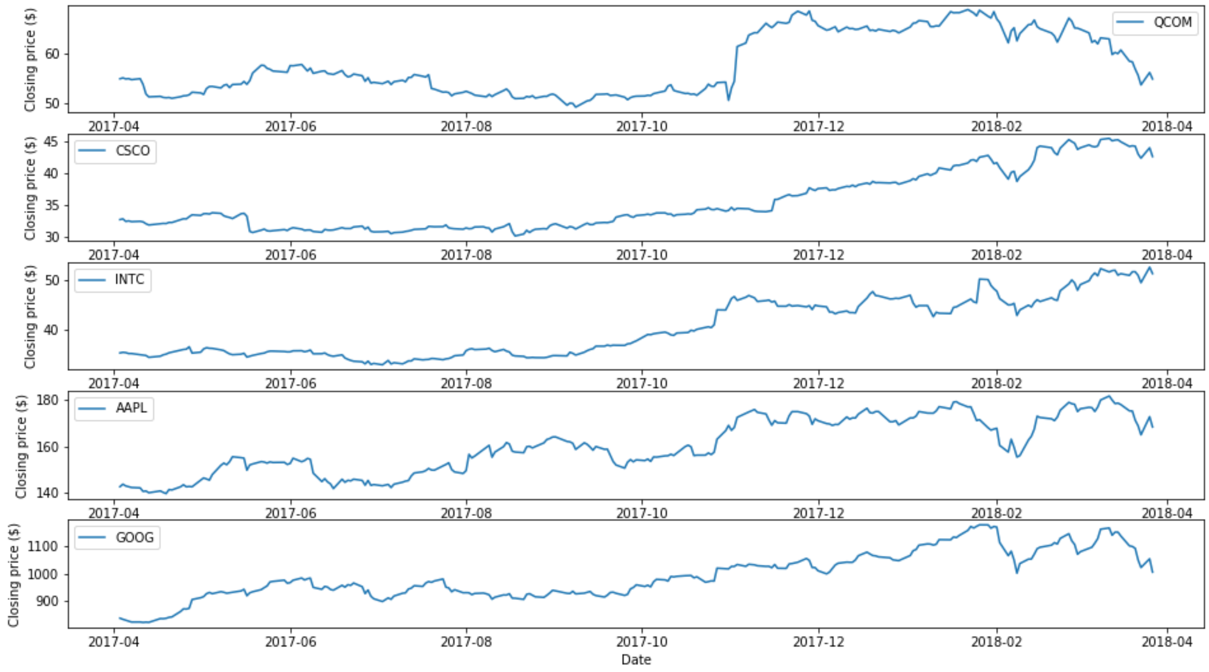
Our platform is used by over 250,000 people, including thousands of analysts from the world's top hedge funds, asset managers and investment banks.

3) In our program the key is: k9Xs_yso27irbXnFsGxz

4) You can run the program in Jupyter Notebook

2. Final Output

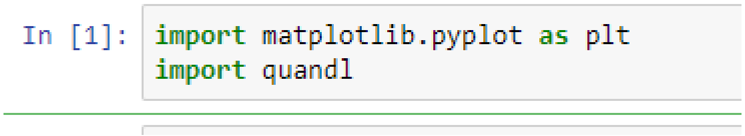
The output will be a figure that shows five stock quotes from 2017-4-1 to 2018-4-1.



3. The Code

“matplotlib” package is used to visualize the stock data (pyplot)

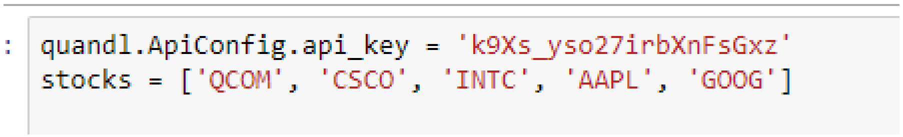
“quandl” is the API used to get the stock quotes



```
In [1]: import matplotlib.pyplot as plt
        import quandl
```

Packages used in our program

Configure our key for Quandl API

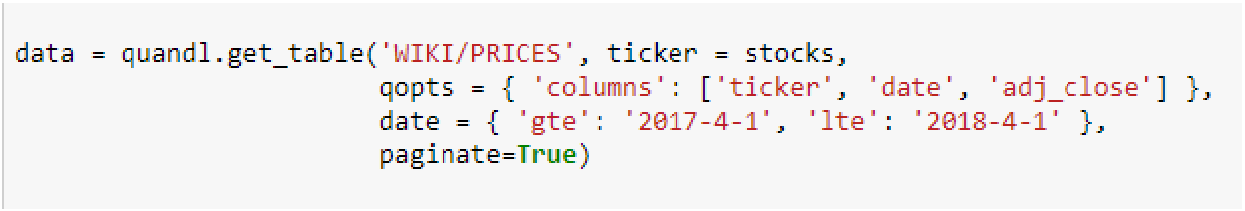


```
: quandl.ApiConfig.api_key = 'k9Xs_yso27irbXnFsGxz'
  stocks = ['QCOM', 'CSCO', 'INTC', 'AAPL', 'GOOG']
```

Key configuration

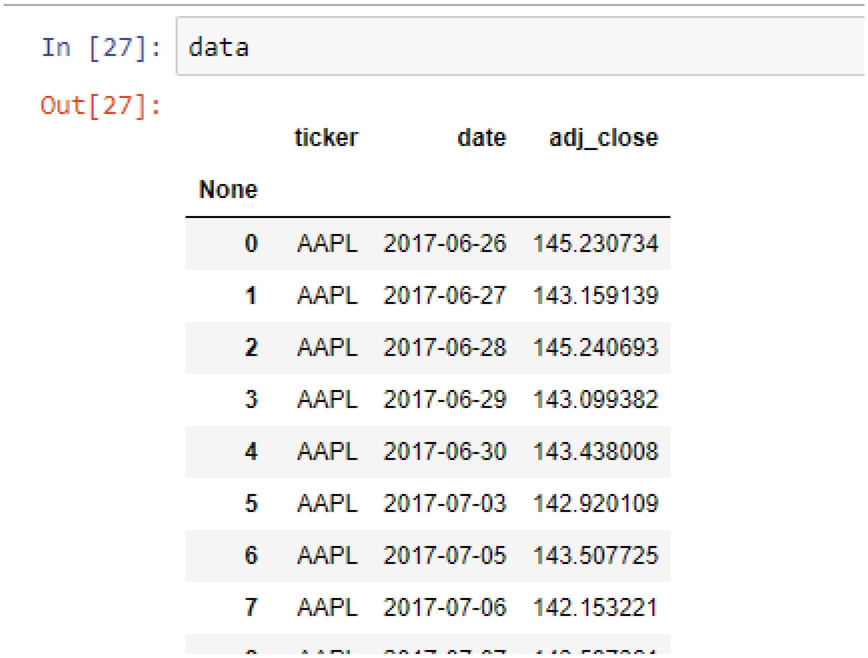
Generate a DataFrame using “quandl.get_table”. The DataFrame will store the adj_close prices of every stock within the defined days

Reference Link: https://www.quandl.com/tools/python



```
data = quandl.get_table('WIKI/PRICES', ticker = stocks,
                        qopts = { 'columns': ['ticker', 'date', 'adj_close'] },
                        date = { 'gte': '2017-4-1', 'lte': '2018-4-1' },
                        paginate=True)
```

Extract the stock data

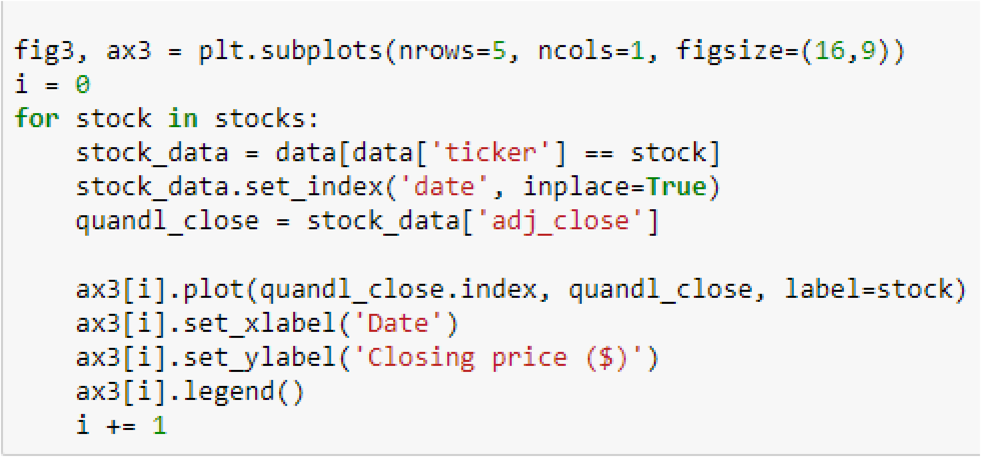


```
In [27]: data
Out[27]:
```

None	ticker	date	adj_close
0	AAPL	2017-06-26	145.230734
1	AAPL	2017-06-27	143.159139
2	AAPL	2017-06-28	145.240693
3	AAPL	2017-06-29	143.099382
4	AAPL	2017-06-30	143.438008
5	AAPL	2017-07-03	142.920109
6	AAPL	2017-07-05	143.507725
7	AAPL	2017-07-06	142.153221

Stock quotes DataFrame

Use the stock data which is stored in the DataFrame to draw the plot.



```
fig3, ax3 = plt.subplots(nrows=5, ncols=1, figsize=(16,9))
i = 0
for stock in stocks:
    stock_data = data[data['ticker'] == stock]
    stock_data.set_index('date', inplace=True)
    quandl_close = stock_data['adj_close']

    ax3[i].plot(quandl_close.index, quandl_close, label=stock)
    ax3[i].set_xlabel('Date')
    ax3[i].set_ylabel('Closing price ($)')
    ax3[i].legend()
    i += 1
```

Visualize the stock trend

Conclusion

Using a combination of these stock quotes can help you monitor those key financial ratios and research companies—whether for fun or profit.