Installing Bloomberg APOD and API

Dr. Fluharty-Jaidee

2020-05-21

Contents

Chapter 1

Forward

1.1 Purpose

This manual is intended for use by West Virginia University active research faculty and PhD students. You must have an active Bloomberg account associated with West Virginia University in order to access the Bloomberg license.

The use of Bloomberg Anywhere is temporarily granted during the COVID-19 closure and is expected to be removed at a future date. The APOD will become unusable once the access is rescinded. Users will still be able to install the Python API and access the data by using the Terminals housed in the Department of Finance at the John Chambers College of Business and Economics.

1.2 Bloomberg User Agreement

You are responsible for reviewing the Bloomberg User Agreement. Some important points for you to note:

- There can be only 12 concurrent log-ins because West Virginia University has 12 individual Terminal licenses. If more than 12 users login simultaneously, the 13th user will be informed that she cannot log in until another user disconnects. For this reason, it is good etiquette to log out of Bloomberg when you are finished using it.
- Bloomberg limits the download amounts in each request via the API (either Python or Excel), you may need to chunk your request into components.
- You may not use the Terminal or APOD to obtain information that you trade on. The West Virginia University license does not allow for trades to be placed or made on the information obtained from the terminals.

- The license specifies that data obtained from the service may not be removed from your local drive. You may not obtain data for any individual outside of West Virginia University.
- You may not send data outside of the country without express approval from the Office of Export Control at West Virginia University. This is a matter of federal law. The Directorate of Defense Trade Controls and the Bureau of Industry and Security are responsible for regulating the export of information and research related material. Bloomberg data services is considered controlled technology. Violations range from $50,000 to $1,000,000 in fines and up to 10 years in prison. Questions on this policy can be found at Export Control.

You can review the entire Subscription Terms and Conditions on BBhub.io.

Chapter 2

Installing Bloomberg and API

2.1 Getting Started

First, you will need to have a Bloomberg account. To gain access to Bloomberg, we will first initial a Bloomberg Anywhere you need a previously authorized account. If you do not yet have an account, you will need to create one.

2.2 The Bloomberg Terminal

Next, you need to install the Bloomberg Terminal Client. Install the Terminal Client; this will restart your computer and may some time.

After installing the Terminal, you will need to activate Bloomberg. Go to Start > Programs > Bloomberg. The LaunchPad will begin, and the Terminal Screens will launch. You will be prompted with a login screen and a request to submit your Access Key.

At this point, you will need to call Bloomberg Customer Support, and you will need access to your phone and email to receive security codes.

Call Bloomberg Customer Support and inform them that you would like to activate an APOD through Bloomberg Anywhere. Your account needs to be a West Virginia University associated account in order for this to work. Additionally, your account needs to be in good standing and not expired. If your account is expired, the representative can send you a re-activation prompt to your email.

The support specialist should provide you with an Access Key which you will

type into the appropriate place on the Terminal home screen. This will link your APOD Terminal with the Bloomberg service.

2.3 Log-In And Test

After you have successfully created your APOD Terminal creation on your local, compute you will need to login to test your connection. Please be aware that if your computer does not have sufficient hardware requirements, the Terminal may significantly reduce the speed of your system. The Terminal windows will automatically be adjusted to meet your resolution specifications, but I find that minimizing 2 or 3 of the screens can reduce the demands on the system. Otherwise, you should expect your computer will freeze.

Once you have logged into the Terminal, you should test it to see if the functions are working properly. Type "CVX US Equity" in the search bar and then GP for graphics plot. This will produce a graphic image of the historical prices for Chevron, Inc. The scope of this introduction does not cover the features of Bloomberg, so I encourage you to explore.

Chapter 3

Installing the Python API

3.1 Requirements

First, you will need at least the latest version of Python, 3.7. The easiest method of installation is to download Anaconda from the Anaconda website by selecting the Individual Edition.

Once you have downloaded Anaconda, you will want to launch Anaconda Navigator. The Navigator is a GUI (graphic user interface similar to the Bloomberg LaunchPad) which launches and installs various programs housed under the Anaconda Library which was installed on your local drive. Please launch Spyder, a Python IDE. I do not suggest the use of Jupiter Notebook for programmatic coding.

Once Spyder loads, test the system by importing the pandas library.

```
# run in python script.py
import pandas as pd
```

If the Python console does not report anything and return > then it was correctly loaded. Next close the entire Spyder we will come back to it later.

3.2 Downloading and Unzipping the Bloomberg API

Download the C/C++ supported or experimental release, whichever is appropriate for your operating system. Find the blp folder in your drive directory (i.e. $C : /blp$), place the BloombergWindowsSDK (or other OS SDK) in this

directory by unzipping it into this location (you will need to have unzip, 7zip, or some other related compressor installed.)

From within the Bloomberg C++ SDK folder you have just placed into the $C:/blp$ directory locate the *bin* folder inside $BloombergWindowsSDK/C++API/.../bin/DAPI$ and find the blpapi3_32.dll and blpapi3_64.dll. Copy these driver files and paste them into the DAPI folder ($C:/blp/DAPI$), and overwrite the driver files that are in that directory already.

3.3 Setting-Up The PATH Link

Go to Start > Search > "System Variables" > Environment Variables find Path under 'System Variables' and edit this value. Add a new PATH listing and hit "Browse". Locate the $C:/blp/DAPI$ folder and add this to your root PATH.

3.4 Installing the API

Next go to Start > Search > "Anaconda Prompt". A terminal should begin which will activate conda.

Run the following commands in the terminal – you may need to type them in as copying is sometimes disabled.

```
pip install blpapi --index-url=https://bloomberg.bintray.com/pip/simple
pip install xbbg
```

Allow the installation to complete.

3.5 Running the API in Python

Launch Spyder again by going to Start > Programs > Anaconda > Anaconda Navigator then launching Spyder (alternatively search for Spyder in the search box).

Import the xbbg library for Python.

```
from xbbg import blp
```

If the console returns > it was imported without an error (cross your fingers). At this point you need to have launched and signed into the Bloomberg Terminal.

3.6 Your First Test: Download Chevron's Daily Prices

```
blp.bdh( tickers='CVX US Equity', flds=[ 'Last_Price'],
start_date='2018-10-10', end_date='2020-05-20',
)
```

If the console returns a pandas dataframe of prices you have successfully completed your first pull from Bloomberg.

Chapter 4

Python API Commands

Multiple Python API Clients connect to Bloomberg. There is the direct and strict BLP. **BLP** will return responses to you in a regular JSON style API response. You would need to know how to parse the JSON or use JSON or JSONLight to extract the information you are looking for.

pdblp and **xbbg** are wrappers for the BLP. These will provide pandas data frames ready for you to export to csv or excel or analyze directly in Python or other statistical analytics software.

I suggest you make use of **pdblp** or **xbbg**, reserve the use of BLP when the pdblp or xbbg do not provide you the functionality you require. For example, both pdblp and xbbg packages have limited field search capabilities while //blp/searchFieldRequest allows you to programmatically search fields similar to how you would with the FLDS command in the Bloomberg Terminal. For the purpose of this tutorial, I will show the most common commands in the **xbbg** as these produce results similar to those found in excel and it is the most straightforward. I encourage you to review the **pdblp** package on your own as it is very similar but provides some more features.

4.1 XBBG

The documentation for the **XBBG** package can be found in its python or anaconda library source on the internet. It also has a read-the-docs page which I find to be the simplest. Much of the code here is taken from that source.

There are three main reference commands similar to the commands we run in the Excel API. BDP – data point request, BDH – historical request, and BDS – block requests which return groups of information about the field (not all fields have this capability). Lastly, there is the BDIB request which is for intraday

trading information up-to tick level frequency. Be aware that intraday trading information is available through the Excel API for the past 140 trading days and through the Python API for the past 240 trading days (approximately 1 year).

4.1.1 A sample BDP Command

The simplest command is the BDP or Bloomberg Data Point; it will return a single point in time of data or a static data point which never changes. Let's try it out on Spyder.

```
blp.bdp(tickers='CVX US Equity', flds=['Security_Name', 'GICS_Sector_Name'])
```

4.1.2 A sample BDH Command

The most useful command for finance research will likely be the BDH or Bloomberg Data Historical, which returns you a set of at most daily values for variables which have historical data. We have already done an example of these, but there is another:

```
blp.bdh(tickers='SPX Index', flds=['High', 'Low', 'Last_Price'],
start_date='2018-10-10', end_date='2018-10-20',
)
```

Please note that while it may seem that data would not be reported under the historical feature because we tend to think of prices and returns in a time series and not much else this is not true. There are some 62,000 firm-level variables that Bloomberg allows access to which have historical data.

4.1.3 A Sample BDS Command

The BDS command can be useful in finding information that Bloomberg reports in a table or on a pre-made screen which you would like programmatic access to. Unfortunately, they are unreliable, and the syntax is complicated and not uniform by field. Since there are so many fields to deal with these are less efficient. The following example provides the history of all dividends paid from Chevron over January to June of 2018.

```
blp.bds('CVX US Equity', 'DVD_Hist_All', DVD_Start_Dt='20180101', DVD_End_Dt='20180531
```

4.1.4 A sample BDIB Command

The BDIB command provides intraday prices. You are limited to the historical value of prices you can extract and the amount of values. Generally, Bloomberg

will not allow you to extract more than three to four thousand rows of information at a given time so you may need to break this information up to get it to work correctly.

```
blp.bdib(ticker='BHP AU Equity', dt='2018-10-17')
```

4.2 Extracting Data Out of Python

I suggest you get familiar with loops, arrays, dataframes (pandas), and the csv package in Python. You can directly export data by writing it to a dataframe and them from the pandas package you can write that dataframe to a csv file in your current working directory. It would be best if you changed your working directory at the start of your Spyder session it will default to a place you typically do not want to place files.