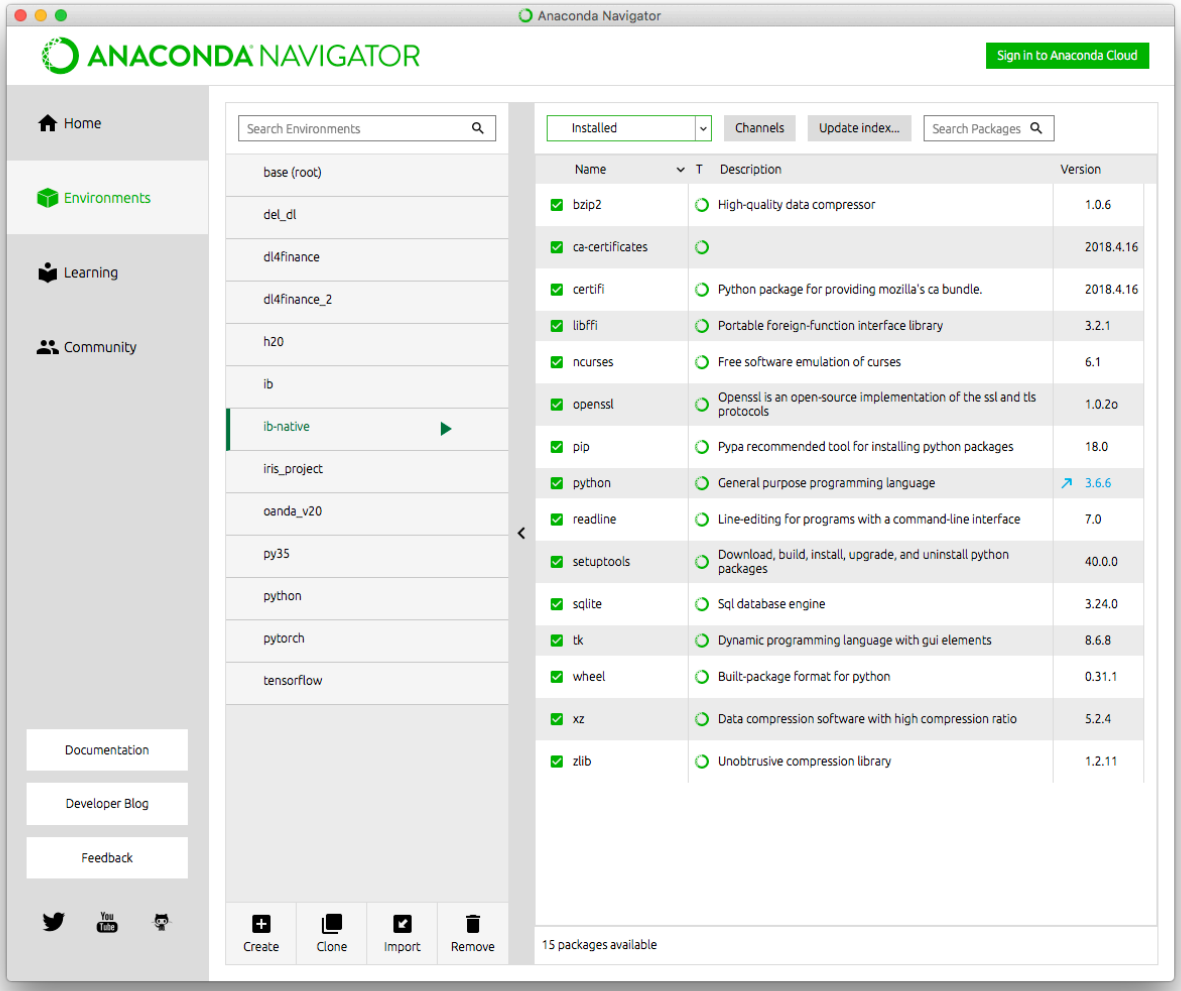
Setting up and Installing TWS API

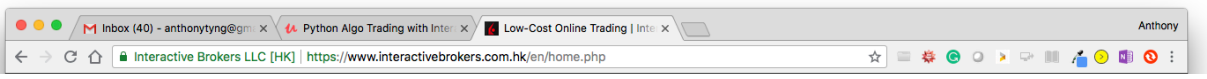
Friday, 10 August 2018 1:58 PM

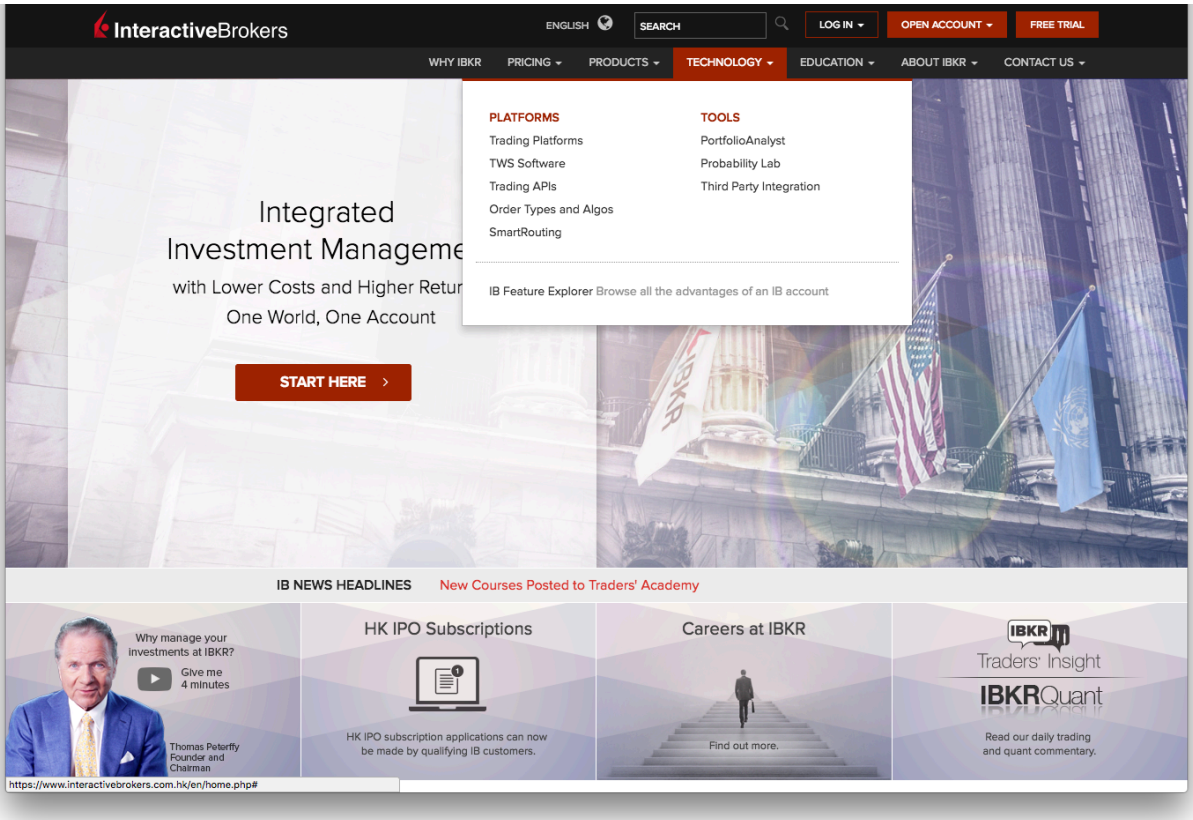
Step 1. Install new environment

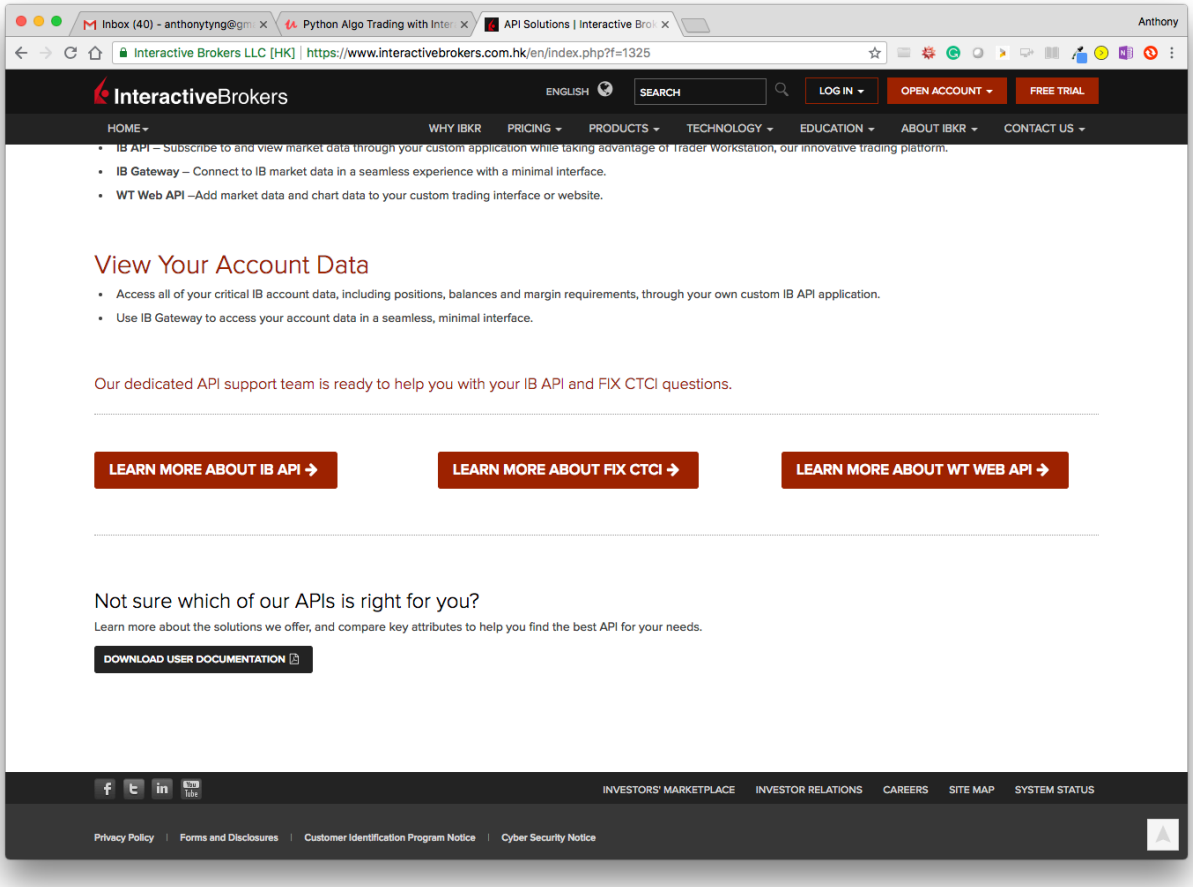


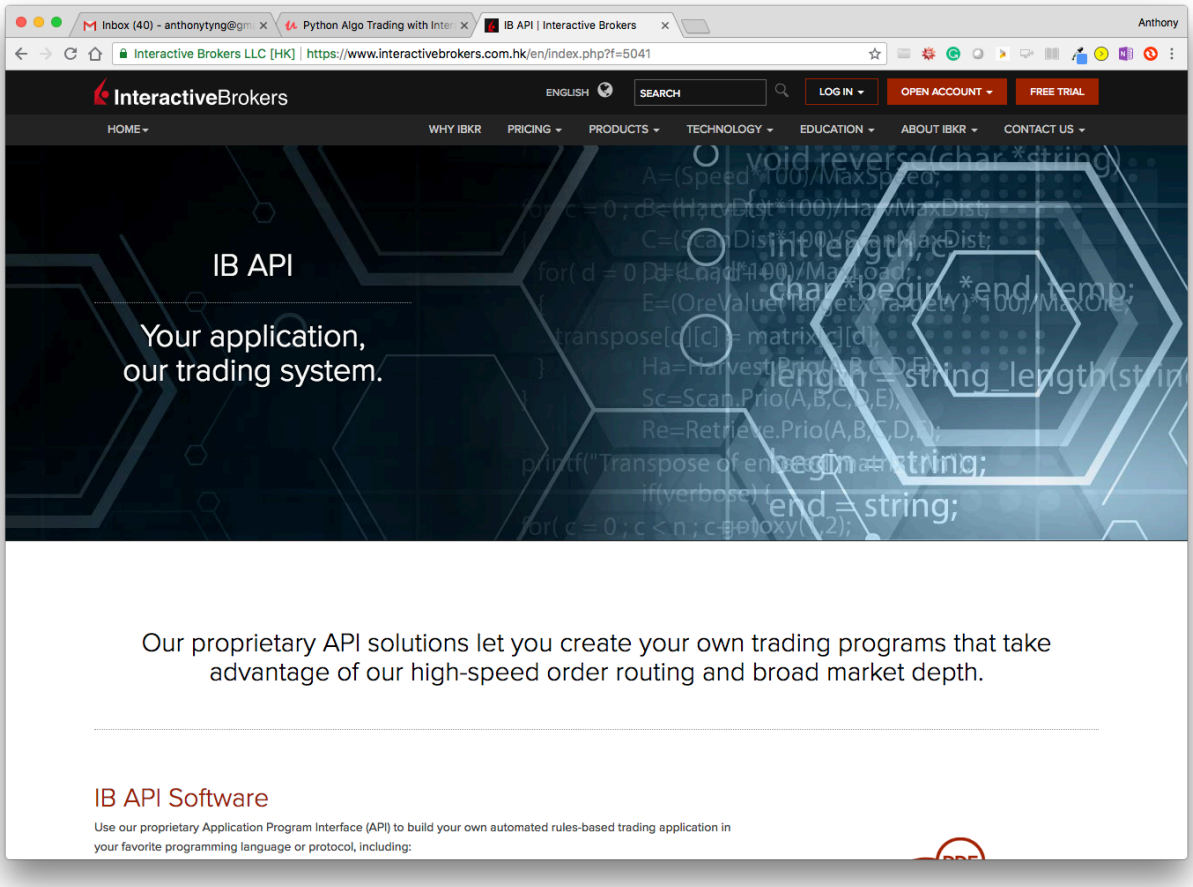
Step 2. Installing the TWS (IB Gateway is contained within) API

http://interactivebrokers.github.io/tws-api/

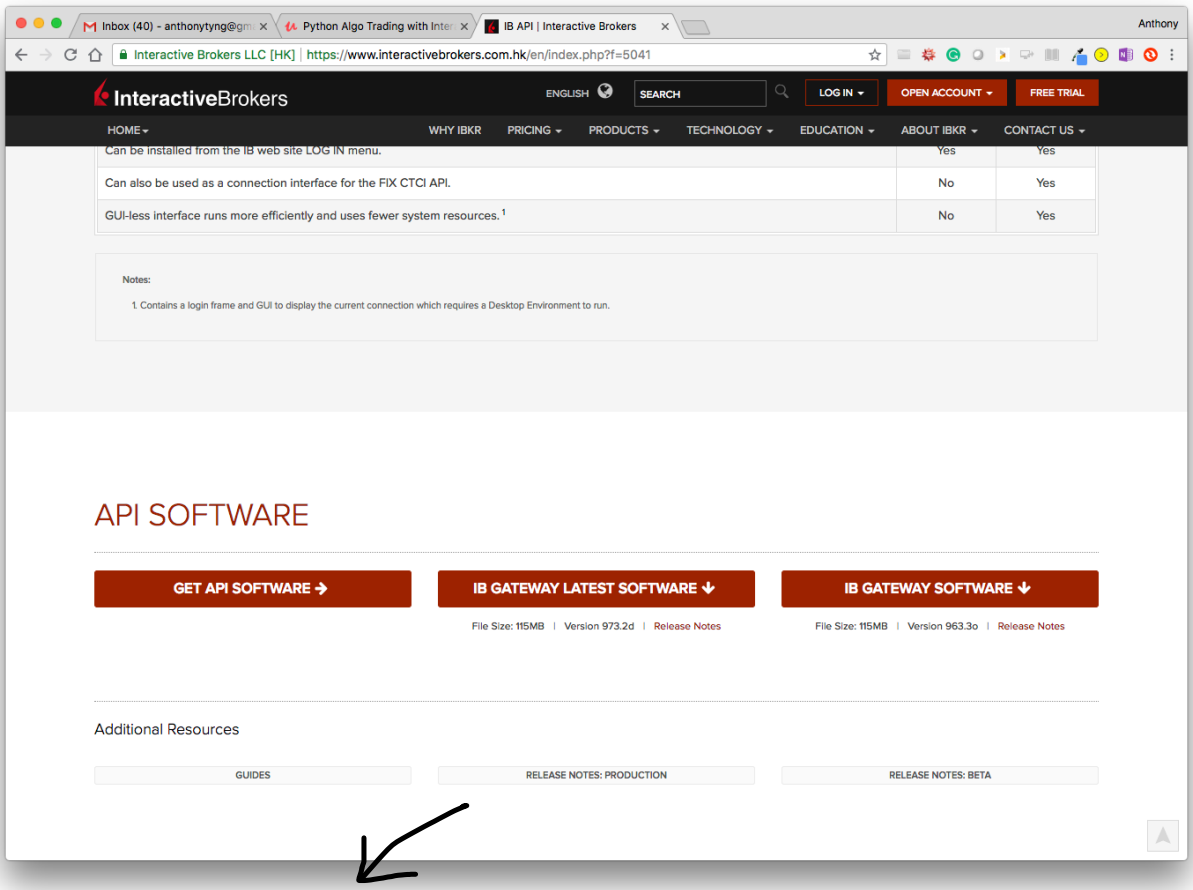


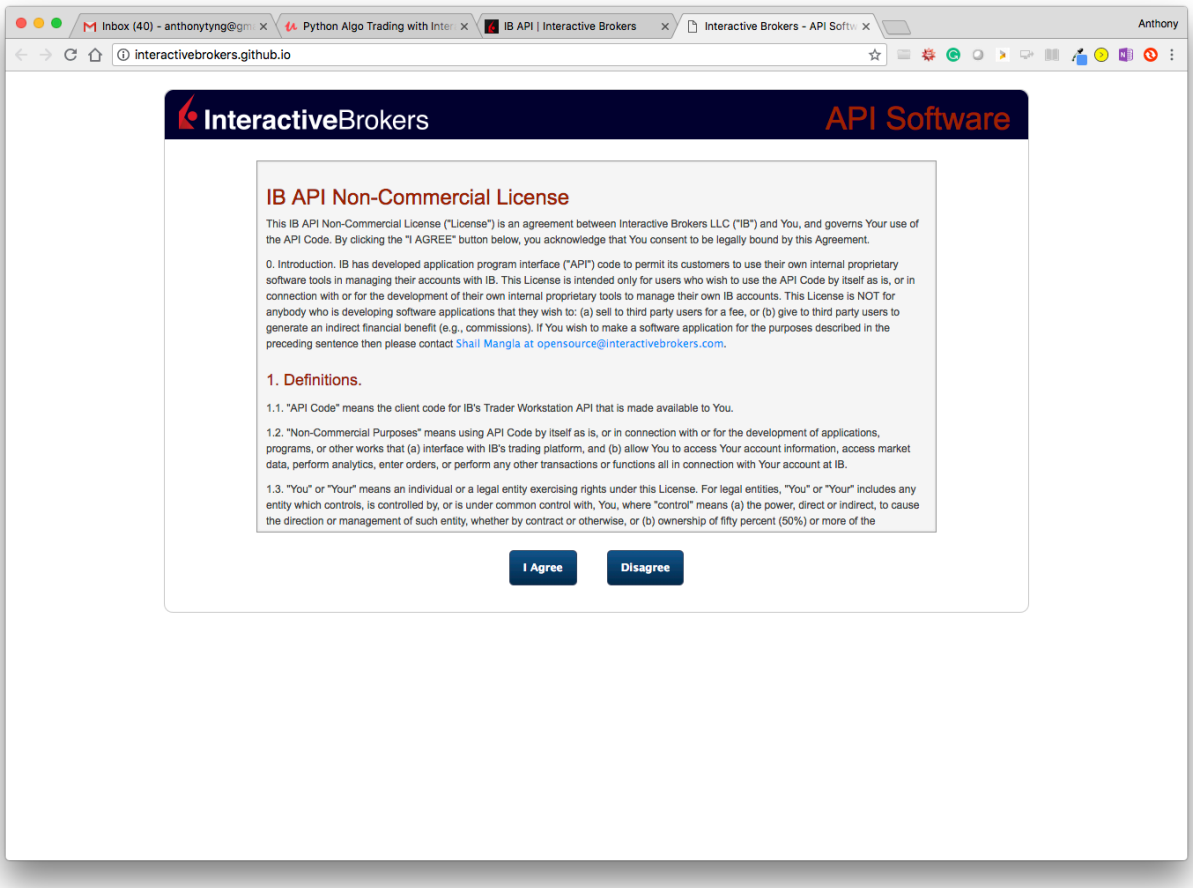

InteractiveBrokers
ENGLISH
SEARCH
LOG IN
OPEN ACCOUNT
FREE TRIAL
WHY IBKR
PRICING
PRODUCTS
TECHNOLOGY
EDUCATION
ABOUT IBKR
CONTACT US
PLATFORMS
Trading Platforms
TWS Software
Trading APIs
Order Types and Algos
SmartRouting
TOOLS
PortfolioAnalyst
Probability Lab
Third Party Integration
IB Feature Explorer Browse all the advantages of an IB account
Integrated
Investment Manageme
with Lower Costs and Higher Retur
One World, One Account
START HERE
IB NEWS HEADLINES
New Courses Posted to Traders' Academy
Why manage your investments at IBKR?
Give me 4 minutes
Thomas Peterffy
Founder and Chairman
HK IPO Subscriptions
HK IPO subscription applications can now be made by qualifying IB customers.
Careers at IBKR
Find out more.
IBKR
Traders' Insight
IBKRQuant
Read our daily trading and quant commentary.
https://www.interactivebrokers.com.hk/en/home.php#


Inbox (40) - anthonytyng@gm
Python Algo Trading with Inter
API Solutions | Interactive Brok
Anthony
Interactive Brokers LLC [HK] | https://www.interactivebrokers.com.hk/en/index.php?f=1325
InteractiveBrokers
ENGLISH
SEARCH
LOG IN
OPEN ACCOUNT
FREE TRIAL
HOME
WHY IBKR
PRICING
PRODUCTS
TECHNOLOGY
EDUCATION
ABOUT IBKR
CONTACT US
IB API – Subscribe to and view market data through your custom application while taking advantage of Trader Workstation, our innovative trading platform.
IB Gateway – Connect to IB market data in a seamless experience with a minimal interface.
WT Web API –Add market data and chart data to your custom trading interface or website.
View Your Account Data
Access all of your critical IB account data, including positions, balances and margin requirements, through your own custom IB API application.
Use IB Gateway to access your account data in a seamless, minimal interface.
Our dedicated API support team is ready to help you with your IB API and FIX CTCI questions.
LEARN MORE ABOUT IB API
LEARN MORE ABOUT FIX CTCI
LEARN MORE ABOUT WT WEB API
Not sure which of our APIs is right for you?
Learn more about the solutions we offer, and compare key attributes to help you find the best API for your needs.
DOWNLOAD USER DOCUMENTATION
INVESTORS' MARKETPLACE
INVESTOR RELATIONS
CAREERS
SITE MAP
SYSTEM STATUS
Privacy Policy
Forms and Disclosures
Customer Identification Program Notice
Cyber Security Notice

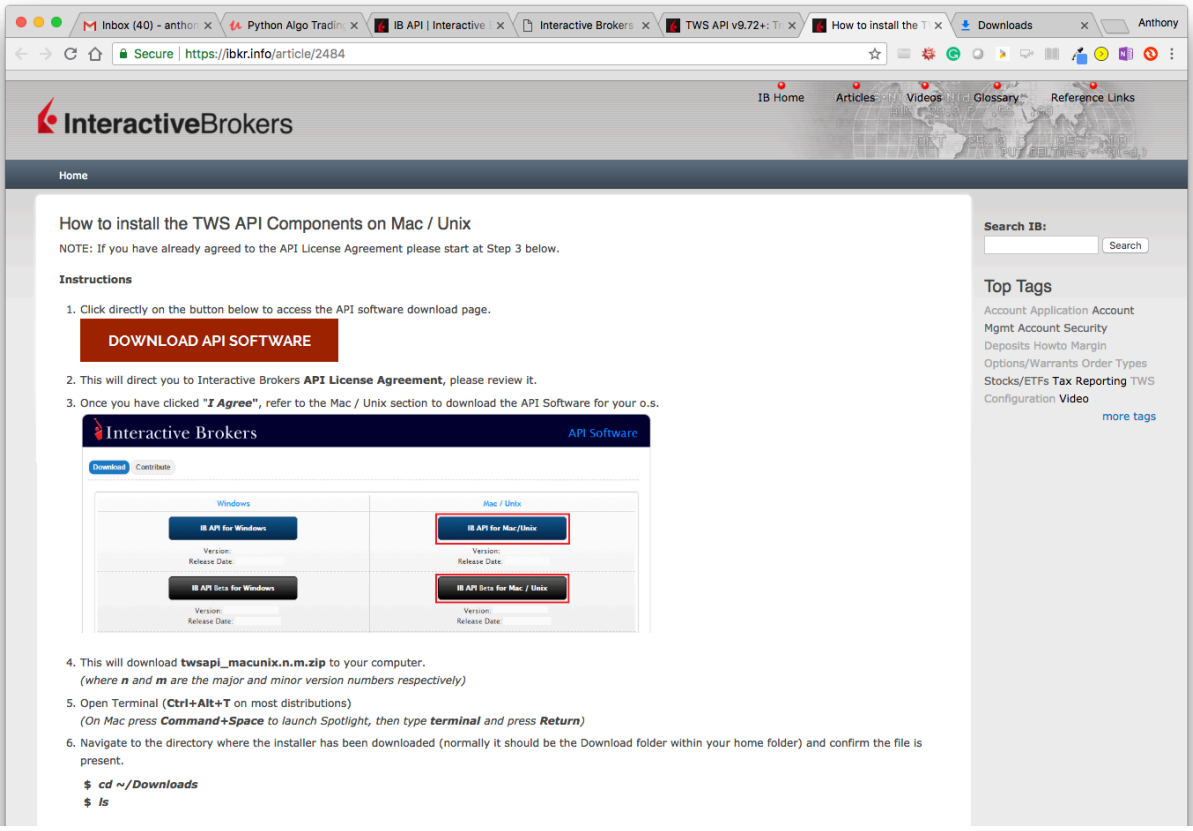


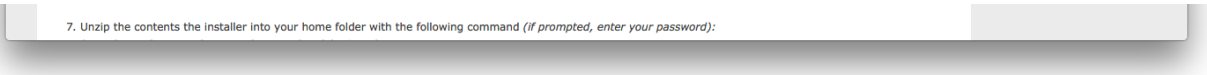
Scroll to the bottom



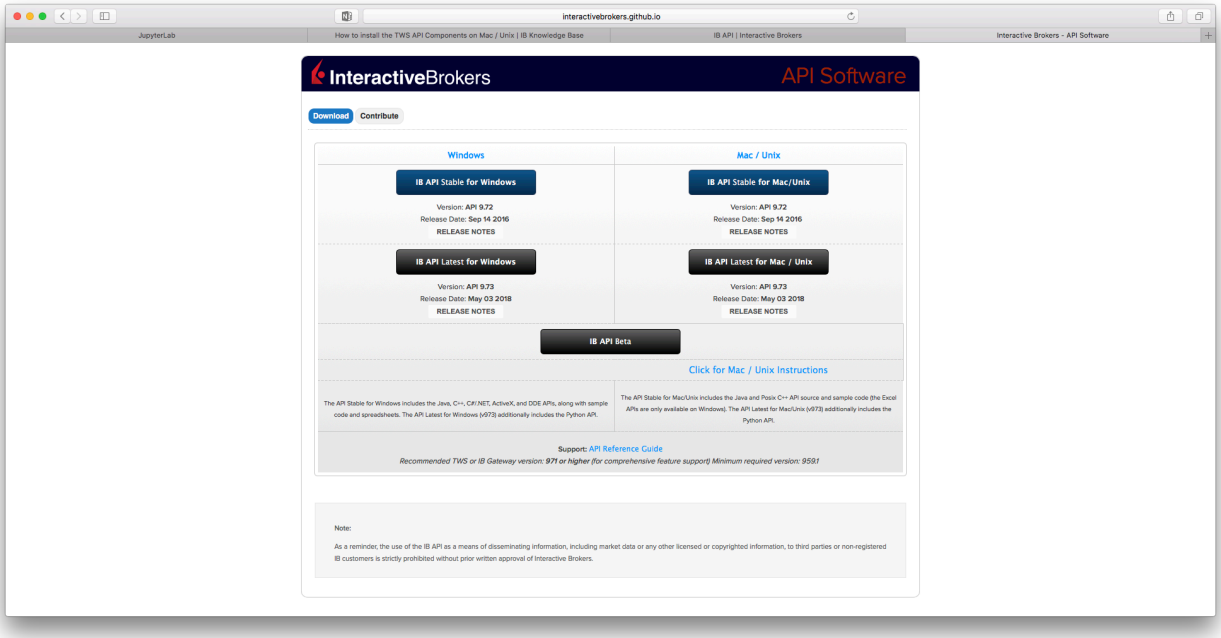


How to install the tws IB components on Mac/Unix. This applies to IB Gateway as well https://ibkr.info/article/2484



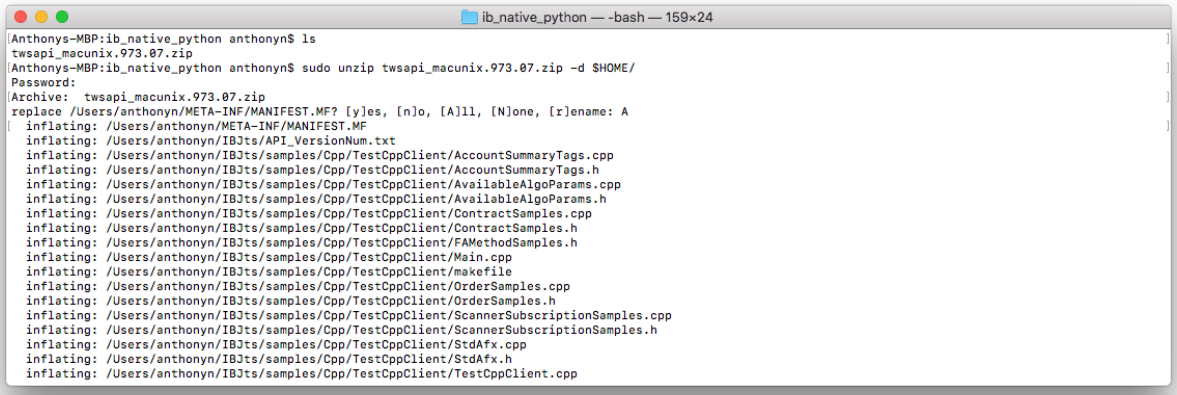


Note the "click for Mac / Unix Instructions". Make sure you click the link and follow it closely.

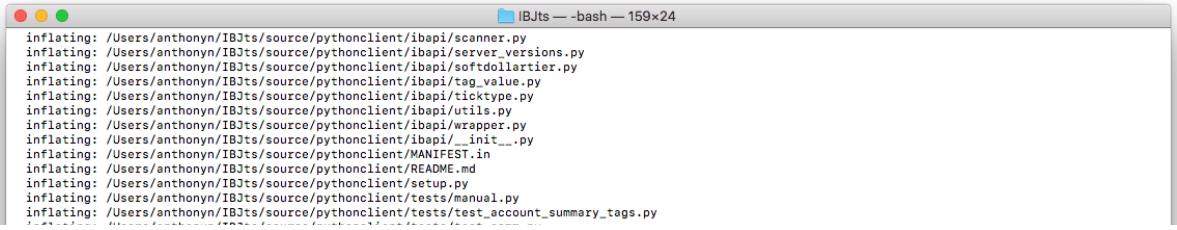


The following steps are laid out in the read.me file.

Follow the sudo unzip…

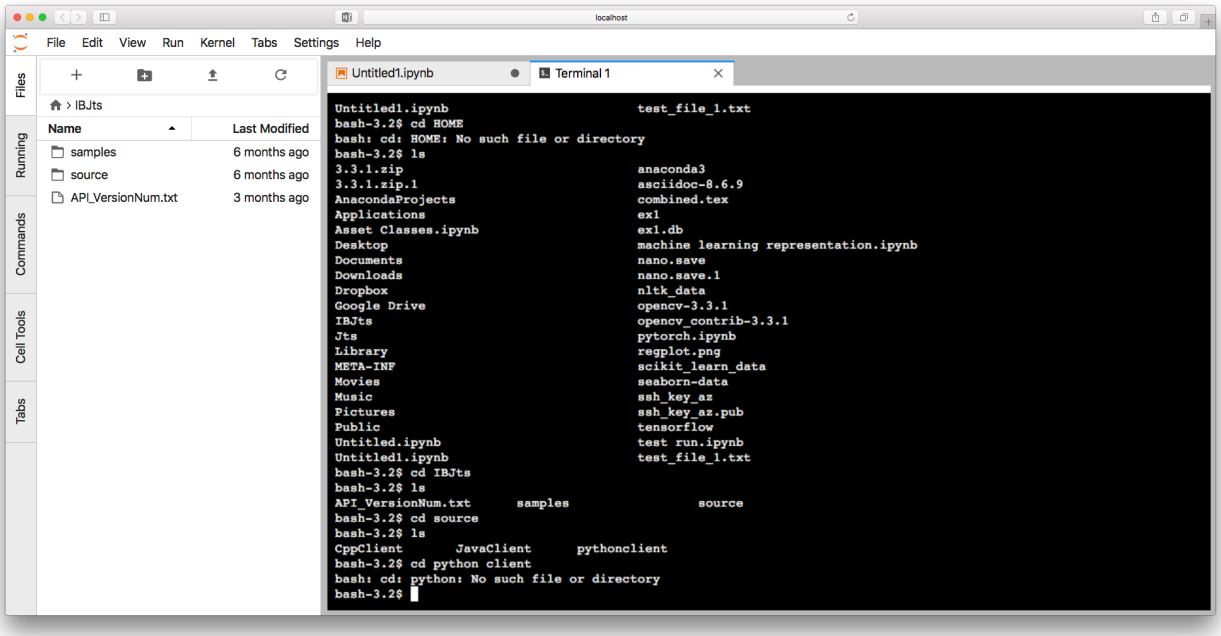


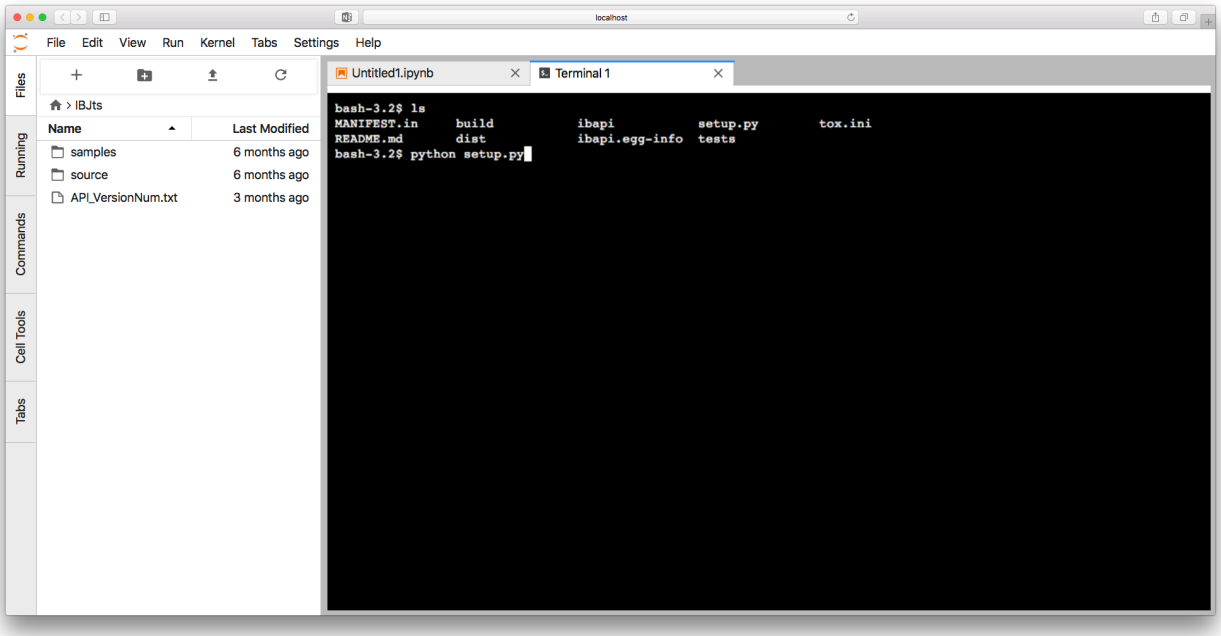
Cd ~/IBJts



```
  inflating: /Users/anthonyn/IBJts/source/pythonclient/tests/test_comm.py
  inflating: /Users/anthonyn/IBJts/source/pythonclient/tests/test_enum_implem.py
  inflating: /Users/anthonyn/IBJts/source/pythonclient/tests/test_order_conditions.py
  inflating: /Users/anthonyn/IBJts/source/pythonclient/tests/test_utils.py
  inflating: /Users/anthonyn/IBJts/source/pythonclient/tox.ini
[Anthonys-MBP:ib_native_python anthonyn$ ls
twsapi_macunix.973.07.zip
[Anthonys-MBP:ib_native_python anthonyn$ cd ~/IBJts
[Anthonys-MBP:IBJts anthonyn$ ls
API_VersionNum.txt      samples                 source
Anthonys-MBP:IBJts anthonyn$
```

Run the python setup





Open the readme.me file and read the contents of the file and follow it!!!

The read.me file:

A couple of things/definitions/conventions:
* a *low level message* is some data prefixed with its size
* a *high level message* is a list of fields separated by the NULL character; the fields are all strings; the message ID is the first field, the come others whose number and semantics depend on the message itself
* a *request* is a message from client to TWS/IBGW (IB Gateway)
* an *answer* is a message from TWS/IBGW to client

How the code is organized:
* *comm* module: has tools that know how to handle (eg: encode/decode) low and high level messages
* *Connection*: glorified socket
* *Reader*: thread that uses Connection to read packets, transform to low level messages and put in a Queue
* *Decoder*: knows how to take a low level message and decode into high level message
* *Client*:
  + knows to send requests
  + has the message loop which takes low level messages from Queue and uses Decoder to tranform into high level message with which it then calls the corresponding Wrapper method
* *Wrapper*: class that needs to be subclassed by the user so that it can get the incoming messages

The info/data flow is:

* receiving:
  + *Connection.recv_msg()* (which is essentially a socket) receives the packets
    - uses *Connection._recv_all_msgs()* which tries to combine smaller packets into bigger ones based on some trivial heuristic
  + *Reader.run()* uses *Connection.recv_msg()* to get a packet and then uses *comm.read_msg()* to try to make it a low level message. If that can't be done yet (size prefix says so) then it waits for more packets
  + if a full low level message is received then it is placed in the Queue (remember this is

a standalone thread)
  + the main thread runs the *Client.run()* loop which:
    - gets a low level message from Queue
    - uses *comm.py* to translate into high level message (fields)
    - uses *Decoder.interpret()* to act based on that message
  + *Decoder.interpret()* will translate the fields into function parameters of the correct type and call with the correct/corresponding method of *Wrapper* class

* sending:
  + *Client* class has methods that implement the _requests_. The user will call those request methods with the needed parameters and *Client* will send them to the TWS/IBGW.

Implementation notes:

* the *Decoder* has two ways of handling a message (esentially decoding the fields)
    + some message very neatly map to a function call; meaning that the number of fields and order are the same as the method parameters. For example: Wrapper.tickSize(). In this case a simple mapping is made between the incoming msg id and the Wrapper method:

    IN.TICK_SIZE: HandleInfo(wrap=Wrapper.tickSize),

    + other messages are more complex, depend on version number heavily or need field massaging. In this case the incoming message id is mapped to a processing function that will do all that and call the Wrapper method at the end. For example:

    IN.TICK_PRICE: HandleInfo(proc=processTickPriceMsg),

Instalation notes:

* you can use this to build a source distribution

python3 setup.py sdist

* you can use this to build a wheel

python3 setup.py bdist_wheel

* you can use this to install the wheel

python3 -m pip install --user --upgrade dist/ibapi-9.73.7-py3-none-any.whl

The IB API Handbook.
http://interactivebrokers.github.io/tws-api/