python으로 키움 API 사용법

miniconda 64bit 설치

miniconda 64bit를 설치한다.

Anaconda, miniconda, 64bit, 32bit 다 해봤는데 miniconda 64bit 가 가장 좋다.

Anaconda 는 사용하지 않는 package를 너무 많이 설치한다.

키움 api는 python 32bit 를 사용하므로 miniconda 32bit를 설치해도 되는데 다른 package를 사용할 수 없다. miniconda 64bit를 설치하고 32bit environment를 만드는 게 제일 낫다.

Miniconda3-latest-Windows-x86\_64.exe

를 받아 설치한다.

32bit env 만들고 PyQt5 설치

아래 사이트를 참고 했다.

<https://howtoworld.tistory.com/36>

32bit Environment를 만들고 qt5 와 pandas를 설치한다.

set CONDA\_FORCE\_32BIT=1

conda create -n py39\_32 python=3.9

activate py39\_32

pip install pyqt5

pip install pandas

python=3.8 은 Qt5 와 호환이 안된다고 한다. python=3.7, 3.9는 된다.

Anaconda 가상환경 만들고 삭제하는 것

<https://velog.io/@tyhlife/%EC%95%84%EB%82%98%EC%BD%98%EB%8B%A4-%EA%B0%80%EC%83%81%ED%99%98%EA%B2%BD%EC%97%90%EC%84%9C-%EA%B0%9C%EB%B0%9C%ED%95%98%EA%B8%B0>

가상환경 명령어들이다.

conda create -n (project\_name) python=3.7

conda env list

conda activate (project\_name)

conda deactivate

conda remove --name (project\_name) –all

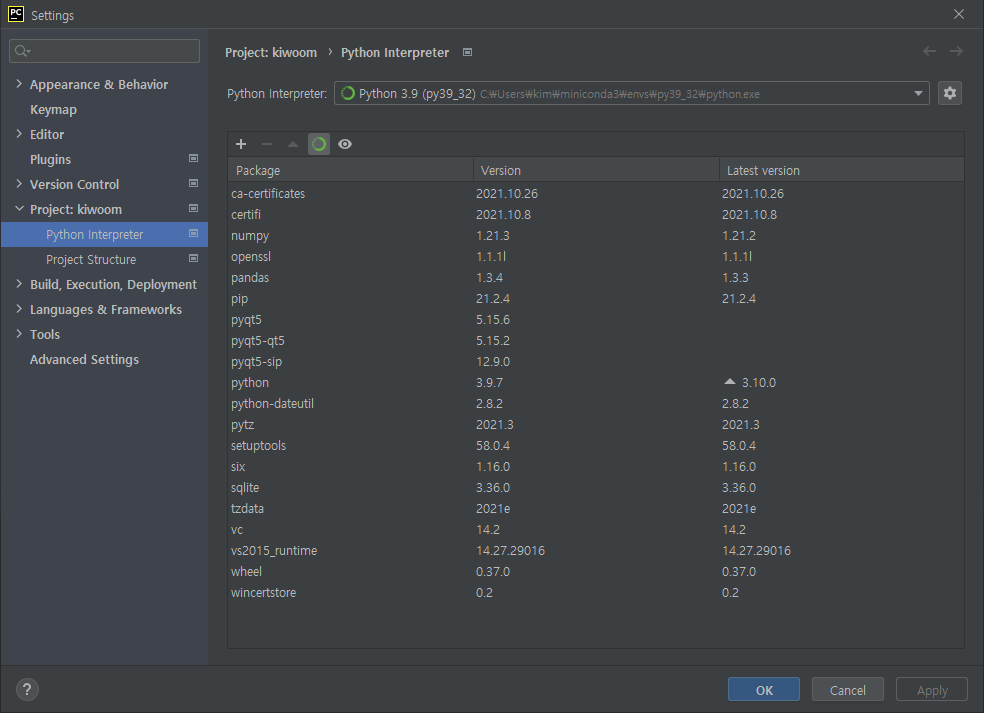
pyCharm 에서 env 사용

아래 사이트 참고

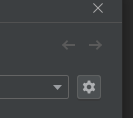
<https://www.jetbrains.com/help/pycharm/conda-support-creating-conda-virtual-environment.html>

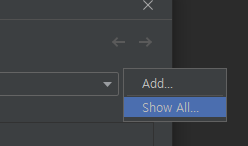
pycharm 에서 프로젝트 생성

ctrl+alt+s 로 Settings 들어 감.

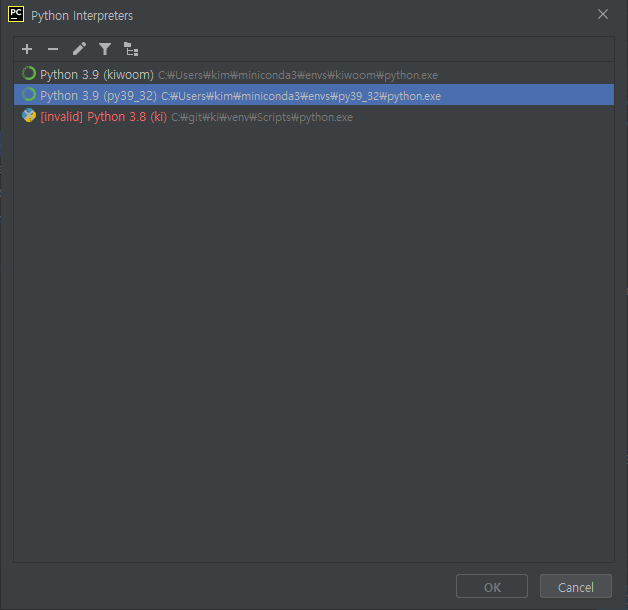


바퀴 버튼에서 Show All 했을 때

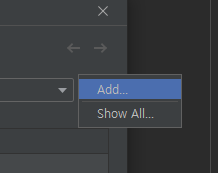




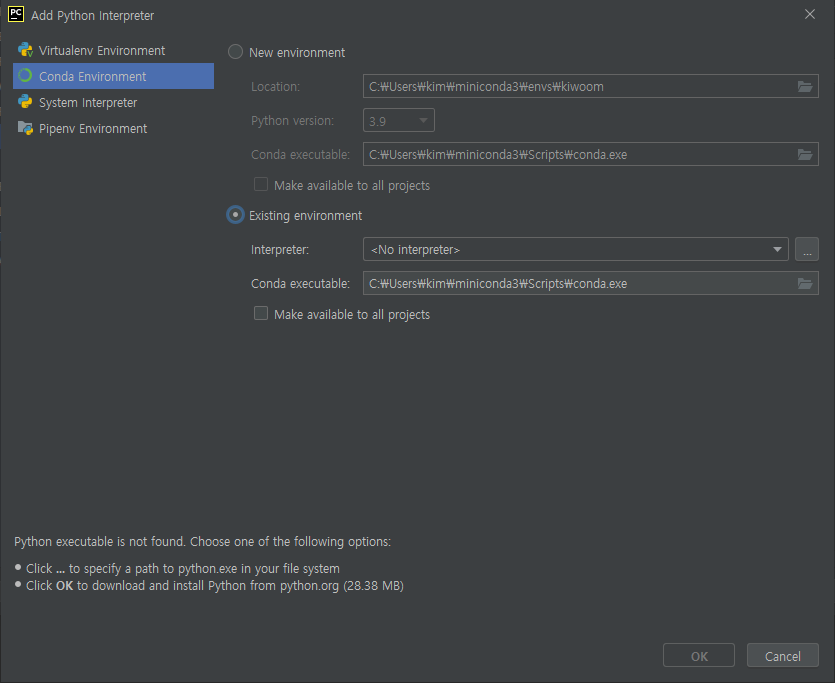
아래와 같이 conda py39\_32 가 선택되어야 함.



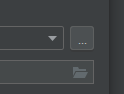
만일 선택 안되어 있으면 바퀴 버튼에서 Add를 클릭함



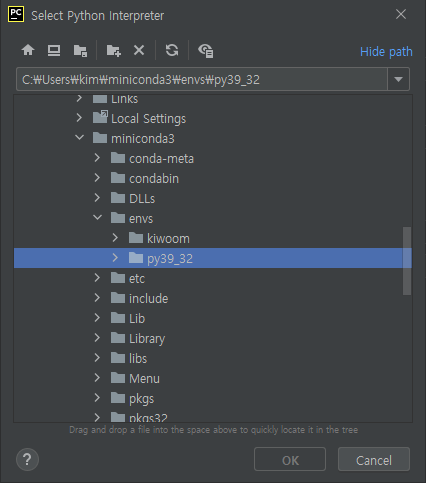
Conda Environment 에서 Existing environment 에서 interpreter 를 추가 해야 함.



… 을 누르고



새로운 environment 선택하고 OK 함



Kiwoom API 처음

아래 링크 참고

<https://lifeonroom.com/study-lab/pyqt5-kiwoom-api-use/>

아래 코드를 돌려 본다.

import sys

from PyQt5.QtWidgets import \*

from PyQt5.QtGui import \*

from PyQt5.QAxContainer import \*

class MyWindow(QMainWindow):

def \_\_init\_\_(self):

super().\_\_init\_\_()

self.setWindowTitle("PyStock")

self.setGeometry(300, 300, 300, 150)

self.kiwoom = QAxWidget("KHOPENAPI.KHOpenAPICtrl.1")

btn1 = QPushButton("Login", self)

btn1.move(20, 20)

btn1.clicked.connect(self.btn1\_clicked)

btn2 = QPushButton("Check state", self)

btn2.move(20, 70)

btn2.clicked.connect(self.btn2\_clicked)

def btn1\_clicked(self):

ret = self.kiwoom.dynamicCall("CommConnect()")

def btn2\_clicked(self):

if self.kiwoom.dynamicCall("GetConnectState()") == 0:

self.statusBar().showMessage("Not connected")

else:

self.statusBar().showMessage("Connected")

if \_\_name\_\_ == "\_\_main\_\_":

app = QApplication(sys.argv)

myWindow = MyWindow()

myWindow.show()

app.exec\_()

성공하면 아래 코드도 돌려본다.

import sys

from PyQt5.QtWidgets import \*

from PyQt5.QAxContainer import \*

import pandas as pd

class KiwoomAPIWindow(QMainWindow):

def \_\_init\_\_(self, connect=1):

super().\_\_init\_\_()

self.title = 'AutoTrader'

self.left = 50

self.top = 50

self.width = 640

self.height = 480

self.initUI()

self.kiwoom = QAxWidget("KHOPENAPI.KHOpenAPICtrl.1")

if connect == 1:

# API 연결

self.kiwoom.dynamicCall("CommConnect()")

# API 연결 되었는지를 Status Bar에 출력

self.kiwoom.OnEventConnect.connect(self.login\_event)

def initUI(self):

self.setWindowTitle(self.title)

self.setGeometry(self.left, self.top, self.width, self.height)

# 라벨 생성

label\_market = QLabel('장 선택 ', self)

label\_market.move(10, 70)

# 콤보 박스 생성

self.cbox\_market = QComboBox(self)

self.cbox\_market.setGeometry(100, 70, 150, 32)

self.cbox\_market.setObjectName(("box"))

self.cbox\_market.addItem("장내", userData=0)

self.cbox\_market.addItem("코스닥", userData=10)

self.cbox\_market.addItem("코넥스", userData=50)

# 버튼 생성

btn\_market = QPushButton('장 리스트 가져오기', self)

btn\_market.setToolTip('0: 장내, 10: 코스닥, 50: 코넥스 등등등 Spec 참조 ')

btn\_market.resize(200, 32)

btn\_market.move(300, 70)

btn\_market.clicked.connect(self.on\_click\_market)

self.show()

def login\_event(self, error):

if error == 0:

strs = '로그인 성공 Code : ' + str(error)

self.statusBar().showMessage(strs)

else:

strs = '로그인 실패 Code : ' + str(error)

self.statusBar().showMessage(strs)

def on\_click\_market(self):

print(self.cbox\_market.currentText(), ' ',self.cbox\_market.currentData())

# GetCodeListByMarket 으로 종목코드 요청

result = self.kiwoom.dynamicCall('GetCodeListByMarket(QString)', str(self.cbox\_market.currentData()))

code\_list = result.split(';')

data\_list = []

for code in code\_list:

name = self.kiwoom.dynamicCall('GetMasterCodeName(QString)', code)

data\_list.append([name, code])

# 데이터 프레임으로 만들기

df = pd.DataFrame(data\_list, columns=['회사명', '종목코드'])

print(df.head())

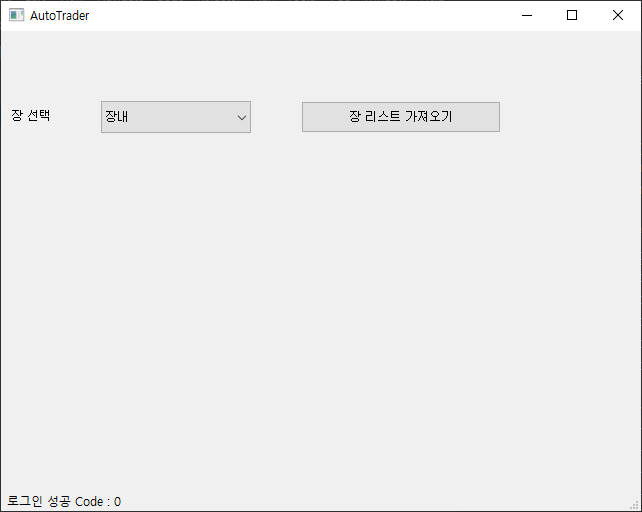
if \_\_name\_\_ == "\_\_main\_\_":

app = QApplication(sys.argv)

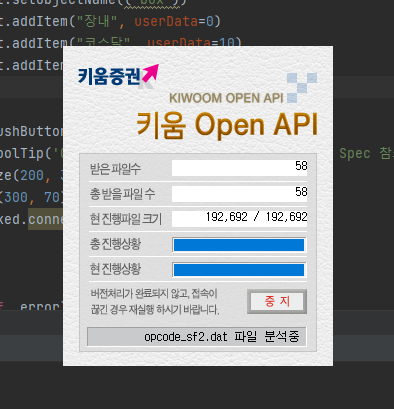
kaWindow = KiwoomAPIWindow()

app.exec\_()

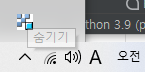
아래 창이 뜨고



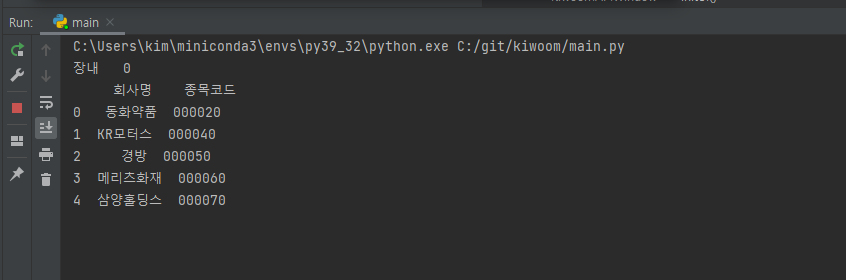
로긴 창이 뜨고



아래 아이콘이 뜬다



장 리스트 가져오기를 클릭하면 console 창에 가져오면 성공이다.



키움 API에서는 할 일이 많은데 사이트 참고해서 잘 해결해야 함

<https://lifeonroom.com/study-lab/pyqt5-kiwoom-api-use/>

주가 가져오기

<https://lifeonroom.com/study-lab/get-stock-price-kiwoom/>

Qt Designer 사용하기

Qt Designer를 다운로드한다.

아래와 같이 실행하면 설치한다.

pip install pyqt5-tools

단 designer.exe 파일이 어디 있는 지 찾아야한다.

내 경우는

C:\Users\kim\miniconda3\envs\py39\_32\Lib\site-packages\qt5\_applications\Qt\bin\designer.exe 에 있다.

이를 바탕화면에 바로가기를 만든다.

Qt Designer 에서 만들고 저장하면 tableWidget.ui 와 같이 ui 파일이 생성된다.

이것을 소스에서 그대로 가져 올 수 있다.

Form\_class, Window\_class = uic.loadUiType("tableWidget.ui")  
  
class MyWindow(Form\_class, Window\_class):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.setupUi(self)  
 self.btnStart.setEnabled(False)  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 app = QApplication(sys.argv)  
 window = MyWindow()  
 window.show()  
 app.exec\_()

이게 편하긴 한데 단점은 self.btnStart.setEnable(False) 처럼 btnStart widget를 소스에서 찾을 수 없다. 상당히 큰 단점이다.

Ui 파일을 py로 변환하여 사용하면 그런 문제가 없다.

python -m PyQt5.uic.pyuic -x .\tableWidget.ui -o .\tableWidget.py

로 tableWidget.py를 만든다.