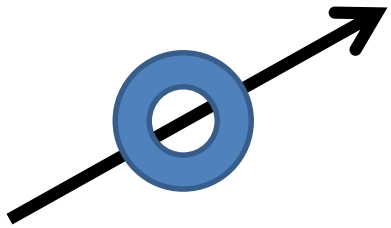
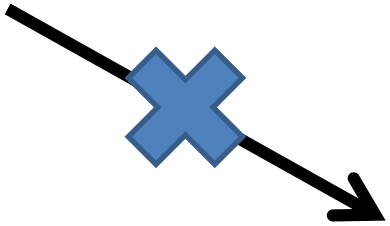
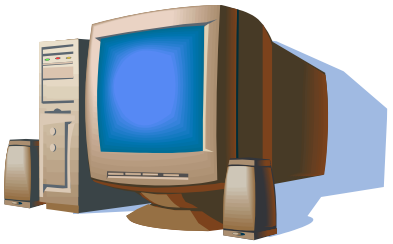
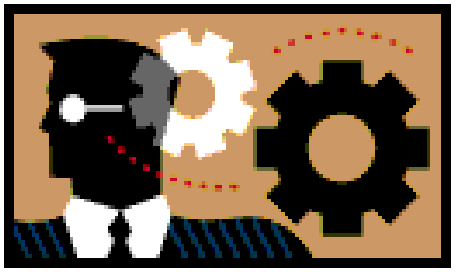
개미투자자의 시트만들기

동기 : Nevet beat the market
인간은 시장을 이길 수 없기 때문



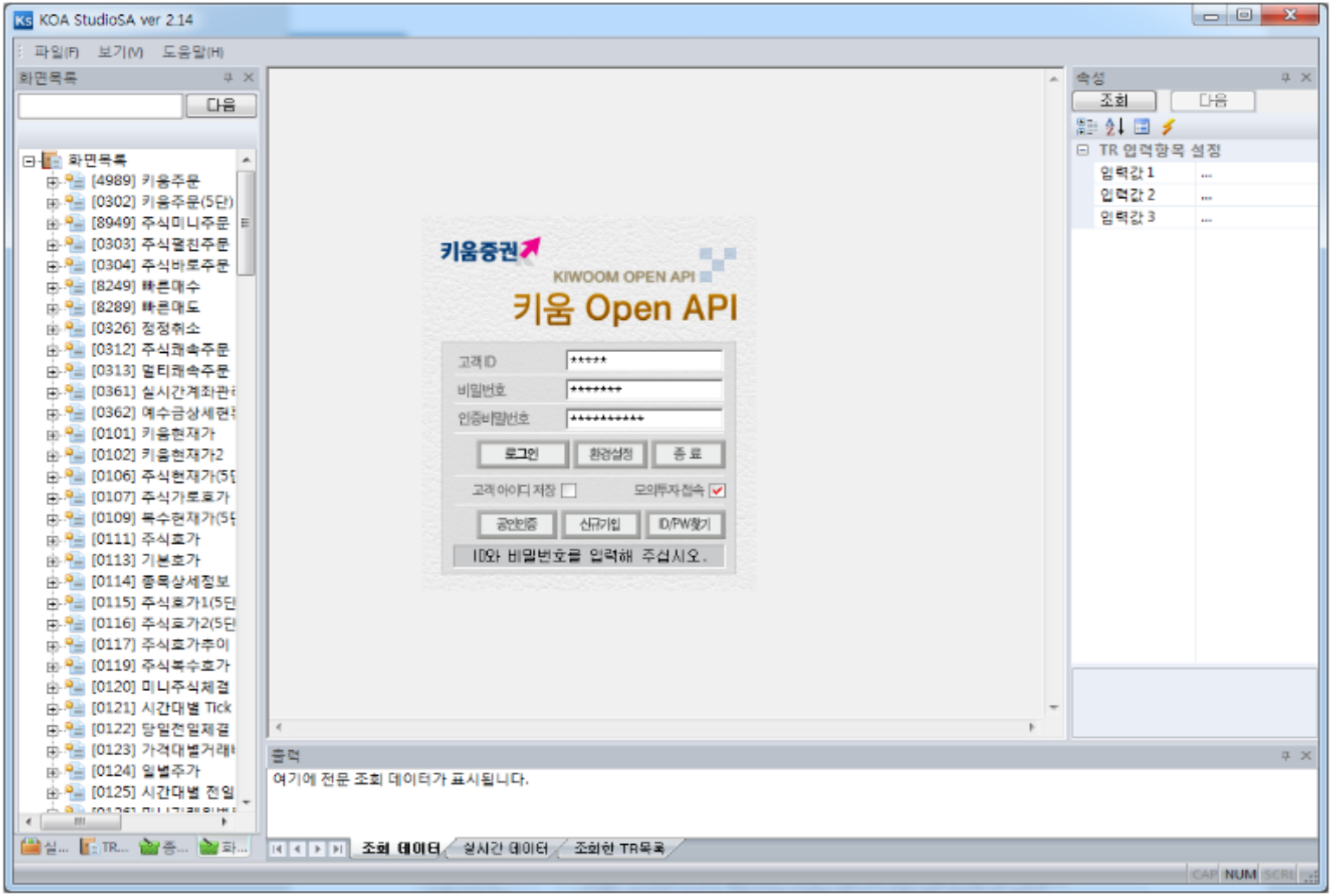






키움증권 API KEY, 설치

KOA프로그램 설치와 Python연동



키움증권 로그인 코드

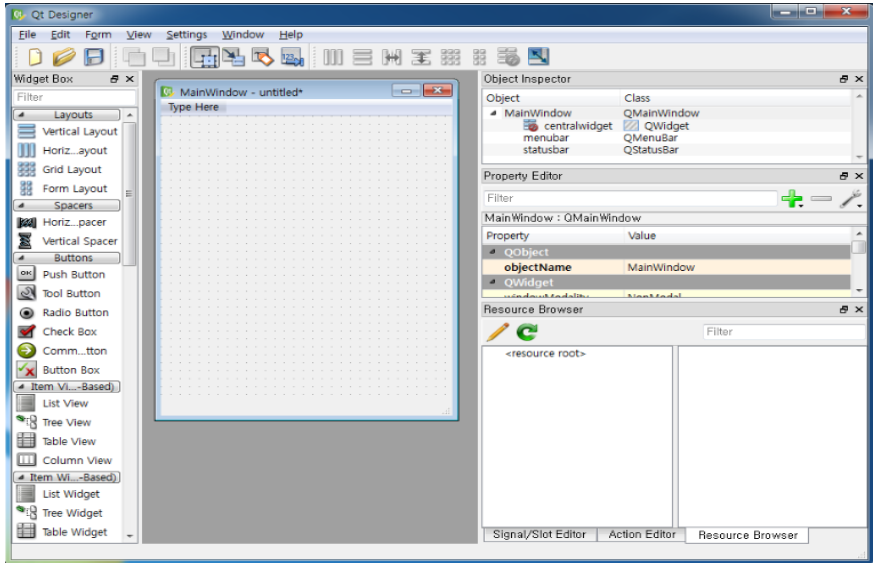
```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import sys
import os
import time
from collections import deque
import threading
from threading import Event
from threading import Lock
import logging
from logging import FileHandler

from PyQt5.QAxContainer import QAxWidget
from PyQt5.QtCore import QObject
from PyQt5.QtCore import QThread
from PyQt5.QtCore import QEventLoop
from PyQt5.QtWidgets import QApplication

import numpy as np
import pandas as pd

import util
```

```python
# 상수
종목별매수상한 = 1000000  # 종목별매수상한 백만원
매수수수료비율 = 0.00015  # 매도시 평단가에 곱해서 사용
매도수수료비율 = 0.00015 + 0.003  # 매도시 현재가에 곱해서 사용
화면번호 = "1234"


# 로그 파일 핸들러
BASE_DIR = os.path.dirname(os.path.abspath(__file__))
fh_log = FileHandler(os.path.join(BASE_DIR, 'logs/debug.log'), encoding='utf-8')
fh_log.setLevel(logging.DEBUG)

# 로거 생성 및 핸들러 등록
logger = logging.getLogger(__name__)
logger.setLevel(logging.DEBUG)
logger.addHandler(fh_log)


class SyncRequestDecorator:
    """키움 API 비동기 함수 데코레이터
    """

    @staticmethod
    def kiwoom_sync_request(func):
        def func_wrapper(self, *args, **kwargs):
            if kwargs.get('nPrevNext', 0) == 0:
                logger.debug('초기 요청 준비')
                self.params = {}
                self.result = {}
            # self.request_thread_worker.request_queue.append((func, args, kwargs))
            logger.debug("요청 실행: %s %s %s" % (func.__name__, args, kwargs))
            func(self, *args, **kwargs)
            self.event = QEventLoop()
            self.event.exec_()
            return self.result  # 콜백 결과 반환
        return func_wrapper

    @staticmethod
    def kiwoom_sync_callback(func):
        def func_wrapper(self, *args, **kwargs):
            logger.debug("요청 콜백: %s %s %s" % (func.__name__, args, kwargs))
            func(self, *args, **kwargs)  # 콜백 함수 호출
        return func_wrapper
```

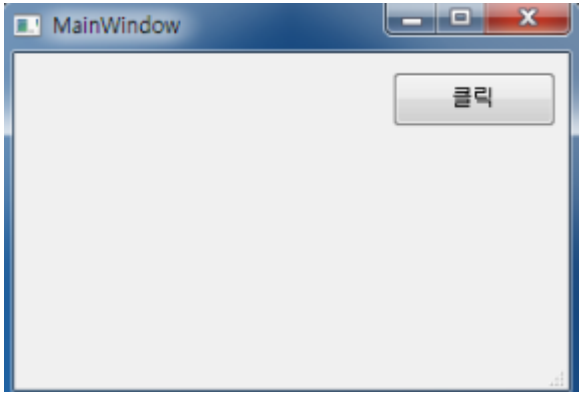
키움증권 로그인 코드2

```
@SyncRequestDecorator.kiwoom_sync_request
def kiwoom_CommConnect(self, **kwargs):
    """로그인 요청 (키움증권 로그인창 띄워줌. 자동로그인 설정시 바로 로그인 진행)
    OnEventConnect() 콜백
    :param kwargs:
    :return: 1: 로그인 요청 성공, 0: 로그인 요청 실패
    """
    lRet = self.dynamicCall("CommConnect()")
    return lRet

def kiwoom_GetConnectState(self, **kwargs):
    """로그인 상태 확인
    OnEventConnect 콜백
    :param kwargs:
    :return: 0: 연결안됨, 1: 연결됨
    """
    lRet = self.dynamicCall("GetConnectState()")
    return lRet

@SyncRequestDecorator.kiwoom_sync_callback
def kiwoom_OnEventConnect(self, nErrCode, **kwargs):
    """로그인 결과 수신
    로그인 성공시 [조건목록 요청]GetConditionLoad() 실행
    :param nErrCode: 0: 로그인 성공, 100: 사용자 정보교환 실패, 101: 서버접속 실패, 102: 버전처리 실패
    :param kwargs:
    :return:
    """
    if nErrCode == 0:
        logger.debug("로그인 성공")
    elif nErrCode == 100:
        logger.debug("사용자 정보교환 실패")
    elif nErrCode == 101:
        logger.debug("서버접속 실패")
    elif nErrCode == 102:
        logger.debug("버전처리 실패")

    self.result['result'] = nErrCode
    if self.event is not None:
        self.event.exit()
```

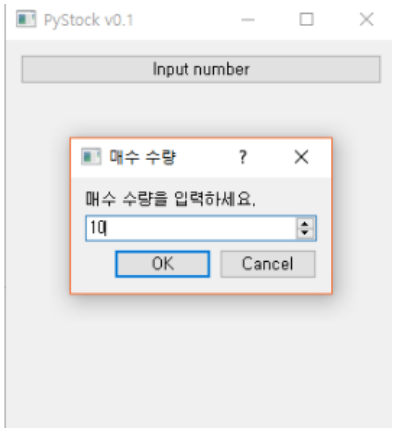
PyQt를 활용한 GUI



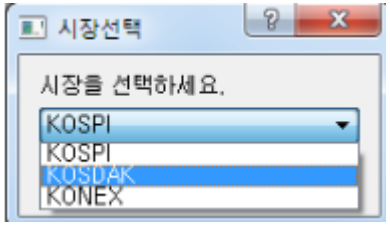

```python
import sys
from PyQt5.QtWidgets import *
from PyQt5 import uic

form_class = uic.loadUiType("main_window.ui")[0]

class MyWindow(QMainWindow, form_class):
    def __init__(self):
        super().__init__()
        self.setupUi(self)

if __name__ == "__main__":
    app = QApplication(sys.argv)
    myWindow = MyWindow()
    myWindow.show()
    app.exec_()
```



PyQt를 활용한 GUI(계속)



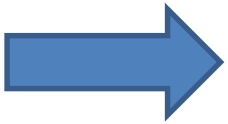

```python
import sys
from PyQt5.QtWidgets import *

class MyWindow(QWidget):
    def __init__(self):
        super().__init__()
        self.setupUI()

    def setupUI(self):
        self.setGeometry(800, 200, 300, 300)
        self.setWindowTitle("PyStock v0.1")

        self.pushButton = QPushButton("Input number")
        self.pushButton.clicked.connect(self.pushButtonClicked)
        self.label = QLabel()

        layout = QVBoxLayout()
        layout.addWidget(self.pushButton)
        layout.addWidget(self.label)

        self.setLayout(layout)

    def pushButtonClicked(self):
        text, ok = QInputDialog.getInt(self, '매수 수량', '매수 수량을 입력하세요.')
        if ok:
            self.label.setText(str(text))

if __name__ == "__main__":
    app = QApplication(sys.argv)
    window = MyWindow()
    window.show()
    app.exec_()
```




```python
def pushButtonClicked(self):
    items = ("KOSPI", "KOSDAK", "KONEX")
    item, ok = QInputDialog.getItem(self, "시장선택", "시장을 선택하세요.", items, 0, False)
    if ok and item:
        self.label.setText(item)
```

일봉 데이터 가져오기

```
import sys
from PyQt5.QtWidgets import *
import Kiwoom
import time
from pandas import DataFrame

MARKET_KOSPI   = 0
MARKET_KOSDAQ  = 10

class PyMon:
    def __init__(self):
        self.kiwoom = Kiwoom.Kiwoom()
        self.kiwoom.comm_connect()
        self.get_code_list()

    def get_code_list(self):
        self.kospi_codes = self.kiwoom.get_code_list_by_market(MARKET_KOSPI)
        self.kosdaq_codes = self.kiwoom.get_code_list_by_market(MARKET_KOSDAQ)

    def get_ohlcv(self, code, start):
        self.kiwoom.ohlcv = {'date': [], 'open': [], 'high': [], 'low': [], 'close': [], 'volume': []}

        self.kiwoom.set_input_value("종목코드", code)
        self.kiwoom.set_input_value("기준일자", start)
        self.kiwoom.set_input_value("수정주가구분", 1)
        self.kiwoom.comm_rq_data("opt10081_req", "opt10081", 0, "0101")
        time.sleep(0.2)

        df = DataFrame(self.kiwoom.ohlcv, columns=['open', 'high', 'low', 'close', 'volume'],
                       index=self.kiwoom.ohlcv['date'])
        return df

    def run(self):
        df = self.get_ohlcv("039490", "20190710")
        print(df)

if __name__ == "__main__":
    app = QApplication(sys.argv)
    pymon = PyMon()
    pymon.run()
```



	open	high	low	close	volume
83200	84000	82500	83300	47452
83500	84400	82400	83500	35650
86000	86300	83300	83500	87923
82400	86500	81700	86300	113036
80000	81500	79500	80400	33167
79700	81800	78900	80400	59060
76100	80000	75800	79200	52802
74400	77300	73300	76800	68185

자동매매 사례 : 급등주 포착

```
import sys
from PyQt5.QtWidgets import *
import Kiwoom
import time
from pandas import DataFrame
import datetime

MARKET_KOSPI   = 0
MARKET_KOSDAQ  = 10

class PyMon:
    def __init__(self):
        self.kiwoom = Kiwoom.Kiwoom()
        self.kiwoom.comm_connect()
        self.get_code_list()

    def get_code_list(self):
        self.kospi_codes = self.kiwoom.get_code_list_by_market(MARKET_KOSPI)
        self.kosdaq_codes = self.kiwoom.get_code_list_by_market(MARKET_KOSDAQ)

    def get_ohlcv(self, code, start):
        self.kiwoom.ohlcv = {'date': [], 'open': [], 'high': [], 'low': [], 'close': [], 'volume': []}

        self.kiwoom.set_input_value("종목코드", code)
        self.kiwoom.set_input_value("기준일자", start)
        self.kiwoom.set_input_value("수정주가구분", 1)
        self.kiwoom.comm_rq_data("opt10081_req", "opt10081", 0, "0101")
        time.sleep(0.2)

        df = DataFrame(self.kiwoom.ohlcv, columns=['open', 'high', 'low', 'close', 'volume'],
                       index=self.kiwoom.ohlcv['date'])
        return df

    def check_speedy_rising_volume(self, code):
        today = datetime.datetime.today().strftime("%Y%m%d")
        df = self.get_ohlcv(code, today)
        volumes = df['volume']

        if len(volumes) < 21:
            return False

        sum_vol20 = 0
        today_vol = 0

        for i, vol in enumerate(volumes):
            if i == 0:
                today_vol = vol
            elif 1 <= i <= 20:
                sum_vol20 += vol
            else:
                break

        avg_vol20 = sum_vol20 / 20
        if today_vol > avg_vol20 * 10:
            return True

    def update_buy_list(self, buy_list):
        f = open("buy_list.txt", "wt")
        for code in buy_list:
            f.writelines("매수;", code, ";시장가;10;0;매수전")
        f.close()

    def run(self):
        buy_list = []
        num = len(self.kosdaq_codes)

        for i, code in enumerate(self.kosdaq_codes):
            print(i, '/', num)
            if self.check_speedy_rising_volume(code):
                buy_list.append(code)

        self.update_buy_list(buy_list)

if __name__ == "__main__":
    app = QApplication(sys.argv)
    pymon = PyMon()
    pymon.run()
```