

주식투자 자동화 프로그램

돈 주고 사지 마세요!



목차

1. 시작에 앞서...
2. 주식투자 자동화 개요
3. 한국투자증권 API 소개
4. 계좌 개설 및 환경 세팅
5. 국내 주식투자 자동화
6. 해외주식 투자 자동화

1. 시작에 앞서...

주식에 투자하시는 분들이 많이 계실 겁니다.

어떤 분은 감으로, 어떤 분은 공부하여 투자하고 있으시죠?

그리고 시중에 투자 자동화 프로그램을 구매하여 투자하시는 분들도 많으실 겁니다.

최고의 선택은 주식을 깊게 공부하여 자신만의 매매법을 가지고 투자하시는 겁니다.

하지만 주식 공부를 통해 매매하기엔 귀찮음과 어려움이 따릅니다.

이런 분들을 위해 주식투자 자동화 프로그램을 개발해

유료로 배포를 하는 회사를 많이들 접하실 겁니다.

적은 노력을 투자해 꾸준한 수익을 보장한다는 광고를 많이 볼 수 있죠.

그러나 이러한 프로그램들은 사용료가 만만치 않습니다.

평균적으로 수백만 원에 달하는 사용료는 저 같은 개미에게는 부담이 될 수밖에 없죠.

사용한다고 하더라도 시드머니가 많아야 어느 정도 수익을 바라볼 수 있습니다.

이러한 분들을 위해 조금의 공부를 통해

주식투자 자동화 프로그램을 만드는 법을 알려드리겠습니다.

물론 조금 어려울 수도 있겠으나 나만의 매매 프로그램을 만든다는 생각으로 열심히 따라와 주시기 바랍니다.

2. 주식투자 자동화 개요

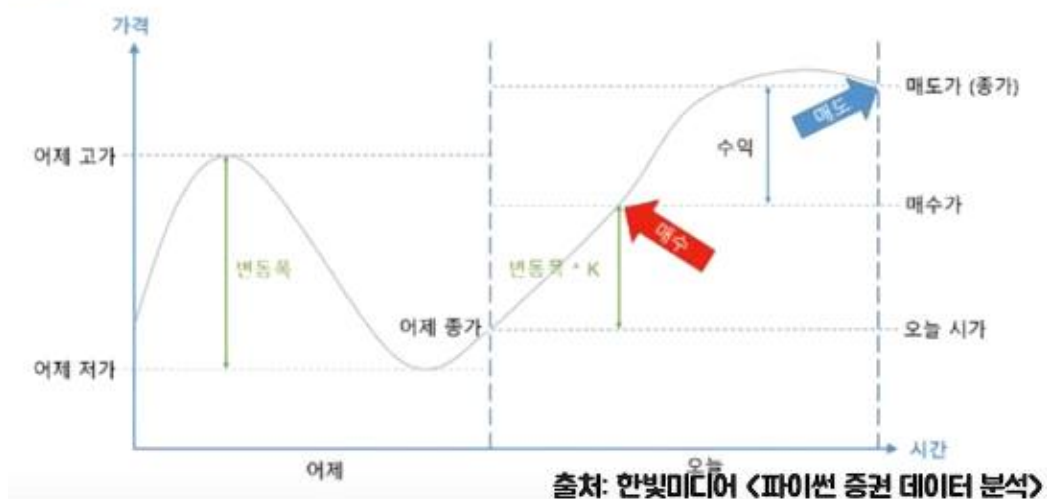
자동화 개요를 살펴보면 파이썬(Python)이라는 프로그램을 사용할 것입니다. 파이썬(Python)이란 오픈소스 고급 프로그래밍 언어 중 하나입니다. 프로그래밍 언어라는 단어를 접하는 여러분은 ‘아. 어렵겠는데…. 내가 할 수 있을까?’라는 생각을 하실 수 있습니다. 예시로 작성된 소스를 그대로 사용하면 되니까 걱정하실 필요 없습니다!.

다시 돌아와서 파이썬(Python)을 이용해 투자 전략을 구현합니다. 이 전략은 한국투자증권의 API를 이용해서 설정한 주식의 매수, 매도 타이밍을 자동으로 감지하여 실행하도록 만들 것입니다. 종목으로는 국내/해외, 주식/ETF를 대상으로 자동매매를 진행할 것입니다. 그리고 필요한 프로그램이 하나 더 있는데 바로 디스코드(discord)입니다. 디스코드(discord)란 음성, 채팅 등을 지원하는 메신저입니다. 자동화를 통한 주식의 매수, 매도를 진행할 때마다 디스코드를 이용해 알림이 뜨게 만들어 줄 것입니다.

파이썬(Python)을 이용하여 투자 전략을 구현할 때 다양한 방법이 있습니다. 예시로, 래리 윌리엄스라는 투자 전문가가 만든 변동성 돌파 전략을 이용해서 자동화 전략을 구현해 보도록 하겠습니다.

래리 윌리엄스의 변동성 돌파 전략이란 어제와 오늘의 가격이 있다고 했을 때, 어제의 저가와 고가의 변동 폭이 있다면 오늘의 시가에서 어제의 (변동 폭 \times K배)만큼의 상승이 있으면 매수를 진행하고 오늘의 종가에 매도를 진행하는 방법입니다.

그림_ 변동성 돌파 전략



3. 한국투자증권 API 소개

편하게 이용하는 오픈API

어떻게 API를 이용하는지 둘러보세요.
간단한 등록 후 이용할 수 있습니다.



- 국내 유일 'REST API, Websocket' 방식
 - 프로그램 설치 필요 없음
 - 개발 환경 상관없음 - 윈도우, 맥, 리눅스 등
- 국내 유일 API로 '해외주식 투자' 가능
 - 통합증거금 이용 시 환전 필요 없음
- KIS Developer 상세 가이드라인, 샘플 코드 제공

3. 계좌 개설 및 환경 세팅

안녕하세요!
한국투자증권 입니다.

- ▶ **한국투자 로고인>**
한국투자 앱을 이용해서 개설다만
- ▶ **모바일 이용 등록하기>**
한국투자 계좌가 있다면
모바일을 한 번도 이용하지 않았더라면
- ▶ **신규 계좌개설>**
한국투자증권이 처음이라면
회원가입 후 바로 계좌등록하면

인증센터 ID등록

스마트폰 계좌개설을 하시면
전체서비스를
바로 이용하실 수 있습니다.

주식+종가형ISA+CMA

자산관리계좌의 기본!
주식과 금융상품 투자를 한번에!

 국내, 해외 주식거래를 한번에

 전세만능 중개형 ISA계좌

 중개형ISA는 평생우대수수료&공모주 청약가능

주식+종가형ISA+CMA 계좌 개설

간편하게 동시에

주식+CMA

주식: CMA
국내·해외 주식거래와 CMA를 한 번에

**국내 • 해외주식 + CMA(발행어음) + ISA 계좌
개설을 시작합니다.**

고객님의 계좌 관리점을 선택해주세요.

 온라인 계좌(BanKIS)
언제 어디서든 온라인 계좌

 **한국투자증권 영업점 개좌**
영업점 직원을 통한 종합자산 관리
기업용 종합계좌를 개설합니다.

 영업점 직원을 통한 출입자 관리와 전문성업이 가능합니다. 주석거래, 펀드, ELS, 채권, 특판상품 가입은 종합계좌를 개설합니다.

스마트폰 계좌개설이란? >

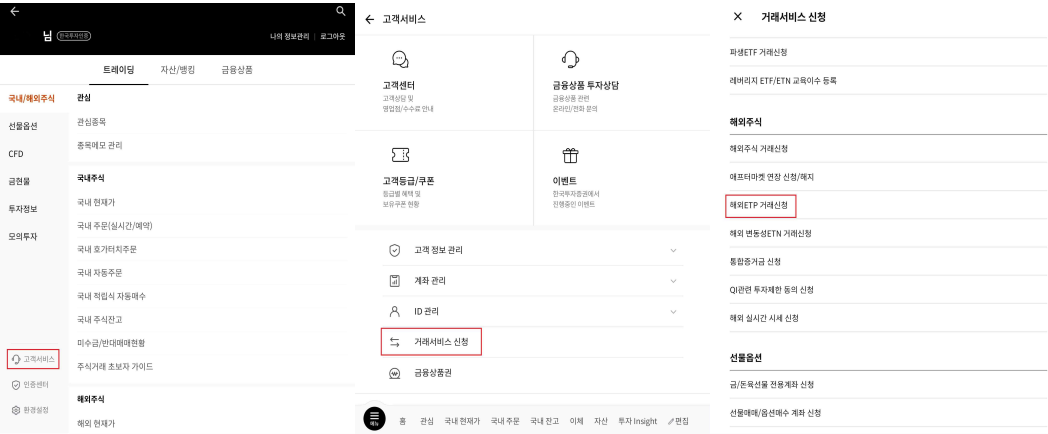
스마트폰 계좌개설

스마트폰 한국투자증권 앱을 이용해 온라인 계좌를 개설하여줍니다.

개설 후 하나의 계좌에서 국내/해외주식을 별도의 환전 없이 이용하려면 통합 증거금 신청을 하여야 합니다.

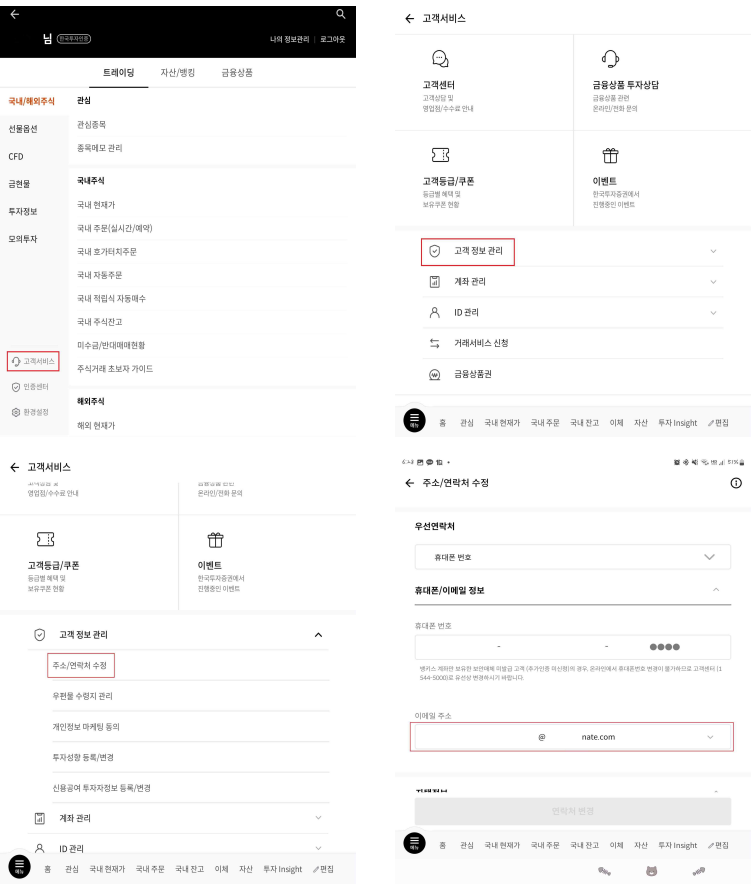
[illegible]

그리고 해외ETP 거래를 하시려면 위 순서 중 거래서비스 신청 -> 해외주식
탭의 해외ETP 거래신청을 눌러주시면 됩니다.



이제 KIS Developer 신청 방법에 대해 알아보겠습니다.

신청 전 반드시 이메일 주소가 등록되어 있어야 하므로 가입 시 등록을 안 하
신 분들은 아래 경로로 이메일 주소 등록을 해주시기 바랍니다.



신청은 <https://apiportal.koreainvestment.com/> 으로 접속 후 신청하실 수 있습니다.



접속 후 오른쪽 위의 API 신청을 눌러줍니다.

로그인

인증서 로그인

스마트폰 인증

ID로그인

한국투자 앱을 통해 QR코드 스캔 또는 인증번호를 입력해주세요.
(한국투자앱 > 전체메뉴 > 인증센터 > PC인증)

OR

090565

인증 유효시간 02:36 초기화

로그인이 안 되어 있으시다면 스마트폰 인증을 통해 간편하게 로그인하실 수 있습니다.

KIS Developers 서비스 신청하기

1. 기본정보 2 3 4

고객명	<input type="text"/>		
고객ID	@1****04		
휴대폰	<input type="text"/> ***	인증번호 요청 <input type="button" value="요청"/>	인증번호입력: <input type="text"/>

신청

유의사항

- 오픈API 서비스는 별도의 화면을 제공하지 않습니다. 따라서 고객님의 요건에 맞는 화면을 직접 제작하여 사용하시기 바랍니다.
- 과도한 서비스 요청에 의한 서버 부하 발생시 임의로 접속이 차단될 수 있습니다.

본인의 정보를 확인 후 인증요청을 눌러 인증번호를 입력한 뒤 신청을 눌러줍니다.

KIS Developers 서비스 신청하기

1 2. 유의사항확인 3 4

개인(신용)정보 수집·이용 동의서

개인(신용)정보 수집·이용 동의서
(trading 오픈API 일반고객용)

[한국투자증권과의 (금융)거래와 관련하여 [한국투자증권이 본인의 개인(신용)정보를 수집·이용·제공 하는 경우에는 「신용 정보의 이용 및 보호에 관한 법률」, 「개인정보 보호법」 등 관계 법령에 따라 본인의 동의가 필요합니다.

* 수집·이용에 관한 사항

수집·이용 목적	- 오픈 API 이용 신청, 유지 및 사후관리 2022.11.15~2023.12.31까지 수집·이용되는 개인정보 항목: 본인 식별 정보, 금융정보, 투자정보, 기타 정보
----------	---

☒ 동의 ☐ 동의하지 않음

고객 이용약관

오픈 API 서비스 이용 약관(고객)

제정 2021.12.27.

제 1 조 (목적)

한국투자증권 주식회사(이하 "회사"라 한다)가 운영하는 "KIS Developers 홈페이지(apiportal.koreainvestment.com, 이하 "포털사이트"라 한다)에서 제공하는 오픈 API 서비스를 이용함에 있어 "회사"와 오픈 API 서비스를 이용하는 회사의 고객(이하 "고객"이라 한다) 간의 권리, 의무 및 책임사항을 명확히 규정함을 목적으로 한다.

각종 동의서를 읽어 보신 후 아래쪽에 동의를 눌러줍니다.

KIS Developers 서비스 신청하기

1 2 3. 신청정보 4

고객명	<input type="text"/>
이메일	<input type="text"/> ① KIS Developers ID/PW 찾기에 사용되는 이메일입니다.
핸드폰번호	<input type="text"/>
전화번호	<input type="text"/>
계좌정보	<input checked="" type="checkbox"/> 종합계좌 <input type="checkbox"/> 모의계좌 ① 동일한 종합 계좌번호의 주식(국내, 해외), 선물/옵션 계좌로 API 이용 가능합니다. 이용하실 종합계좌번호 선택 후 비밀번호를 입력하여 주십시오.
KIS Developers 사용자 ID	<input type="text"/> ① 가입완료 후 임시비밀번호가 알림톡으로 전송됩니다. KIS Developers 사이트(apiportal.koreainvestment.com)에 접속하여 비밀번호를 변경하십시오.

그 후 다시 한번 본인 정보를 확인하시고 계좌정보란의 종합계좌에서 신청할 계좌를 선택하신 후 비밀번호를 입력합니다.

API그룹	<input checked="" type="checkbox"/> OAuth인증 <input checked="" type="checkbox"/> 해외주식현제가 <input checked="" type="checkbox"/> 국내선물옵션사세 <input checked="" type="checkbox"/> 국내선물옵션주문 <input checked="" type="checkbox"/> 국내주식주문 <input checked="" type="checkbox"/> 해외주식주문 <input checked="" type="checkbox"/> 국내주식사세
-------	--

API 그룹은 모두 선택을 해주시면 됩니다.

KIS Developers 서비스 신청하기

1 2 3 4. 실행완료

서비스 앱정보

종합계좌	APP Key	<input type="text"/>	복사
	APP Secret	<input type="text"/>	복사

① APP Key, APP Secret를 타인에게 유출을 금하여 고객님 책임하에 관리 부탁드립니다. 유출시 즉시 홈페이지에서 재발급 하시기 바랍니다.



KIS Developers 서비스가 신청 되었습니다.

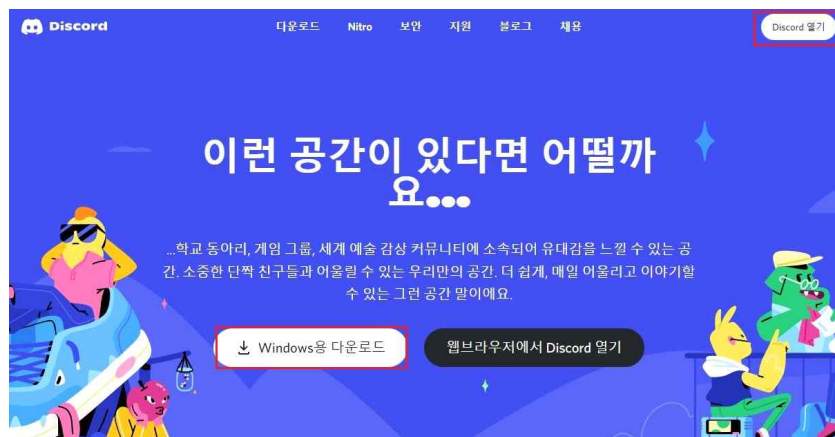
- 신청하신 기간동안 서비스 이용 가능하며, 만기 30일 이전부터 기간을 연장할 수 있습니다.
- 3개월 이상 미거래시 서비스 이용이 차단될 수 있습니다. 문의 : ☎1544-5000, 1588-0012

초기페이지로 이동

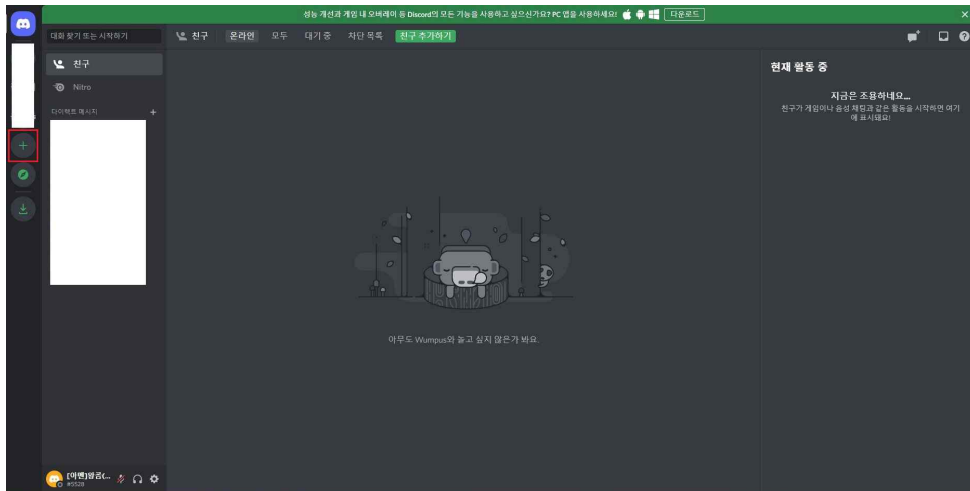
모든 절차를 완료하시면 위와 같이 APP Key, APP Secret을 메모장 등을 이용해 컴퓨터에 저장해 주시다. 이 키들은 자신의 계좌와 연동된 API를 조작할 수 있는 키이므로 외부 유출이 안 되게 보안에 신경 써주시면 됩니다.

다음으로 자동화 매매 과정에서 계속 주식 창만 보고 있을 수 없으므로 알림을 받을 수 있도록 디스코드(discord)를 이용하는 방법에 대해 설명하도록 하겠습니다.

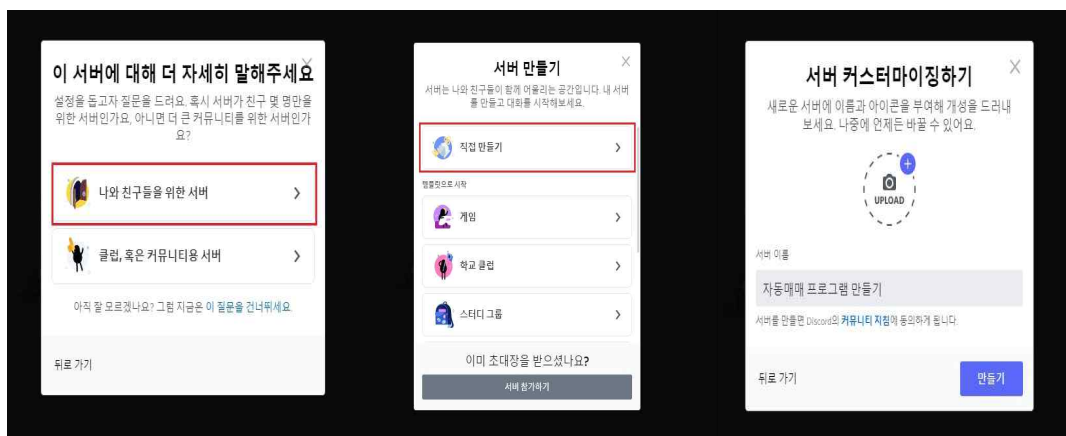
디스코드는 PC, 웹사이트, 앱 모두 지원하므로 원하시는 방법을 선택해서 세팅하시면 됩니다.



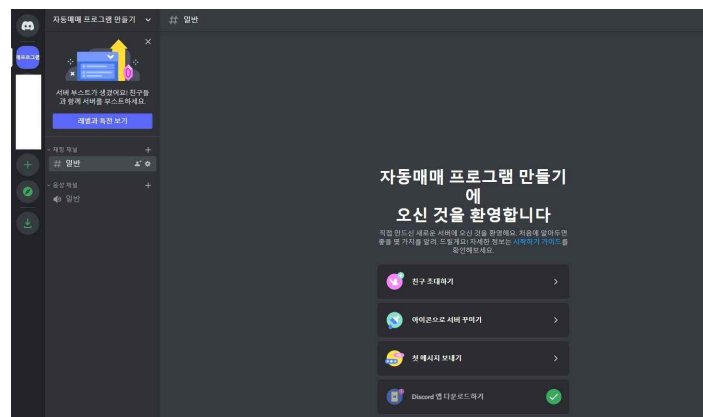
사용준비 사항으로는 <https://discord.com/> 에 접속하여 PC 프로그램으로 다운을 받거나 웹사이트에서 로그인을 통해 바로 접속할 수 있습니다.



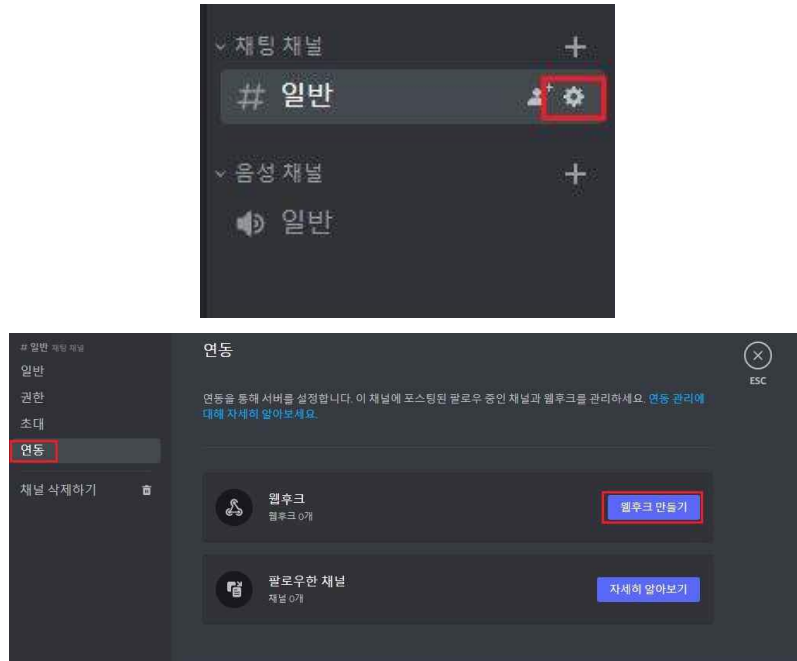
접속 후 왼쪽에 + 모양의 서버 추가하기 버튼을 눌러서 서버를 개설해 줍니다.



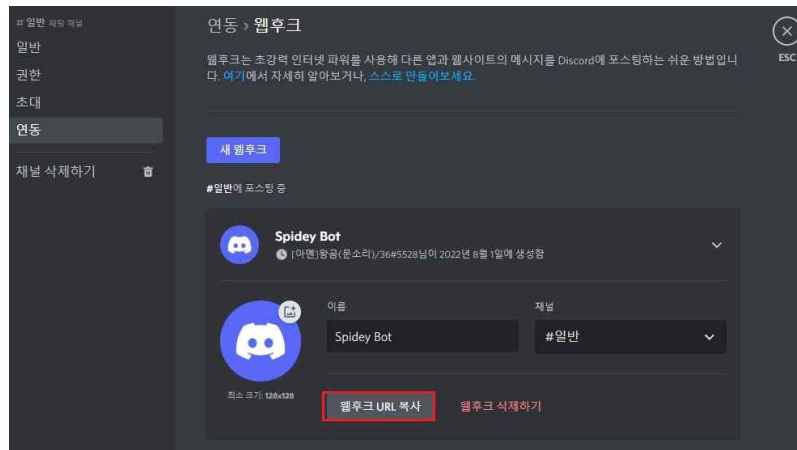
직접 만들기 -> 나와 친구들을 위한 서버 -> 서버 이름 작성 후 만들기를 하시면 기본적인 세팅 채널이 만들어집니다.



이제 이 채팅 채널에 주식투자 자동화 관련 메시지가 전송되도록 하려면 좌측 채팅 채널 탭의 # 일반 옆에 설정 버튼을 눌러줍니다.



그다음 연동 탭으로 들어가면 웹후크 만들기가 보입니다. 눌러줍니다.



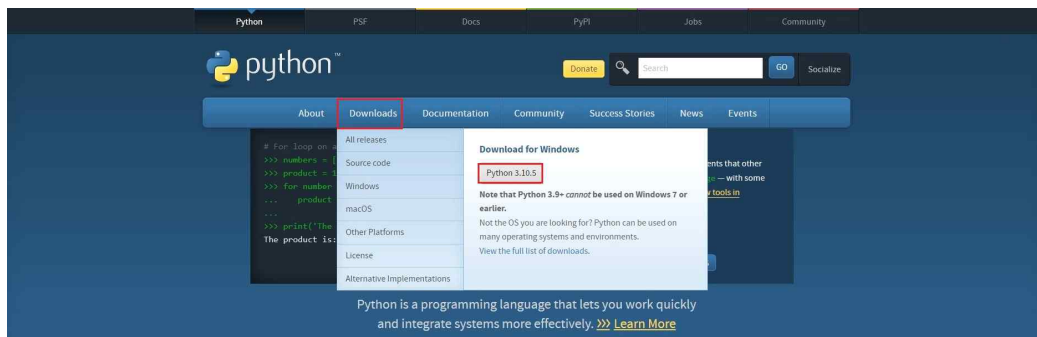
새 웹후크가 생성된 것을 확인하실 수 있습니다. 생성된 웹후크 아래 웹후크 URL 복사를 눌러 복사한 후 메모장 등에 붙여넣고 저장해줍니다.



이 URL을 통해 만들어 놓은 채팅 채널에 메시지를 보낼 수 있게 됩니다.

다음으로 로직 구현을 위한 파이썬(Python)과 코드 에디터인 비주얼 스튜디오 코드(Visual Studio Code)를 설치해 보도록 하겠습니다.

먼저 파이썬(Python)을 설치해 보도록 하겠습니다.

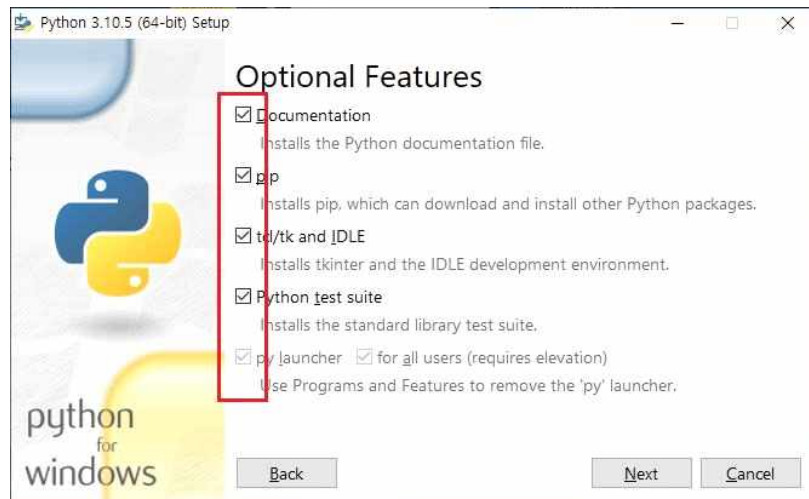


인터넷 브라우저에 접속하여 python.org로 접속하신 다음 메인 화면의 Downloads 탭에 마우스 커서를 가져다 대고 빨간 상자 안의 아이콘을 눌러 설치 파일을 다운로드 받은 후 실행시켜 줍니다.

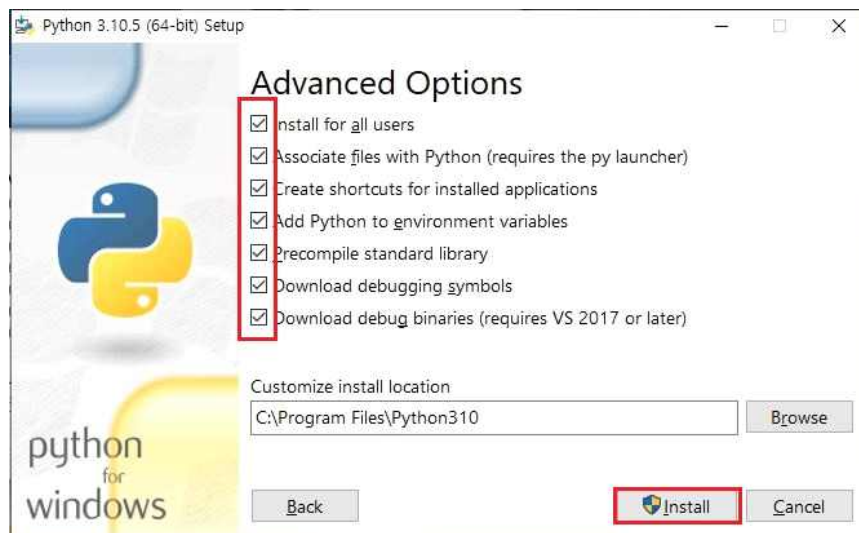


여기서 중요한 부분!! Add Python 버전 PATH 부분은 꼭 체크해주셔야 합니다.

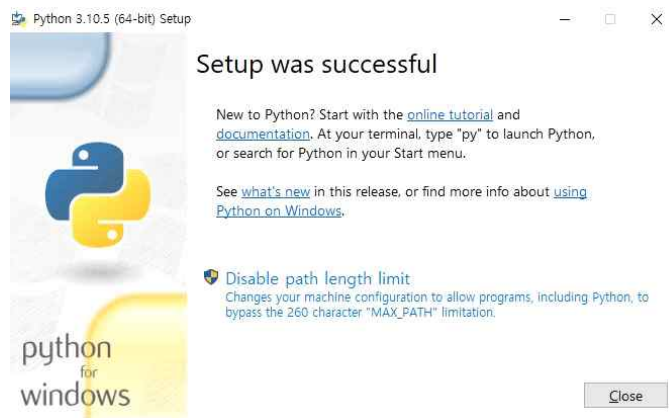
그리고 Customize installation 클릭합니다.



다음 선택 화면도 모두 체크해주세요.



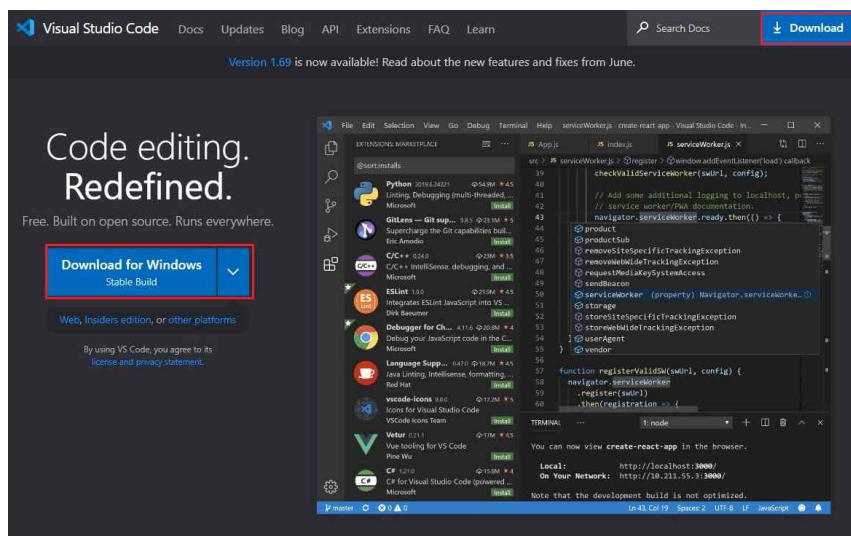
다음 옵션 화면도 모두 체크해 주시고 Install 버튼을 눌러주세요.



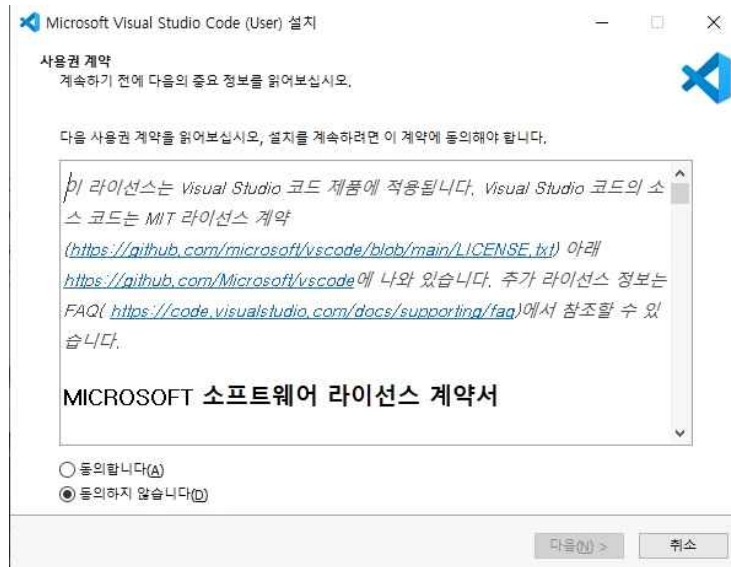
설치가 완료되었습니다.

다음은 Visual Studio Code를 설치해 보도록 하겠습니다.

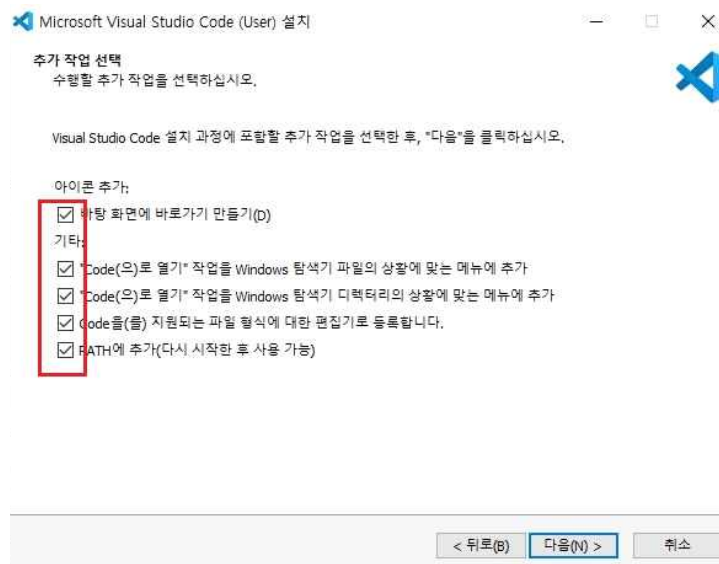
인터넷 브라우저에 접속 후 <https://code.visualstudio.com/> 에 접속합니다.



메인 화면에 Download for Windows 또는 옆에 ▾ 버튼을 눌러 Mac or Linux를 선택하여 설치 파일을 받은 뒤 실행시켜 줍니다.



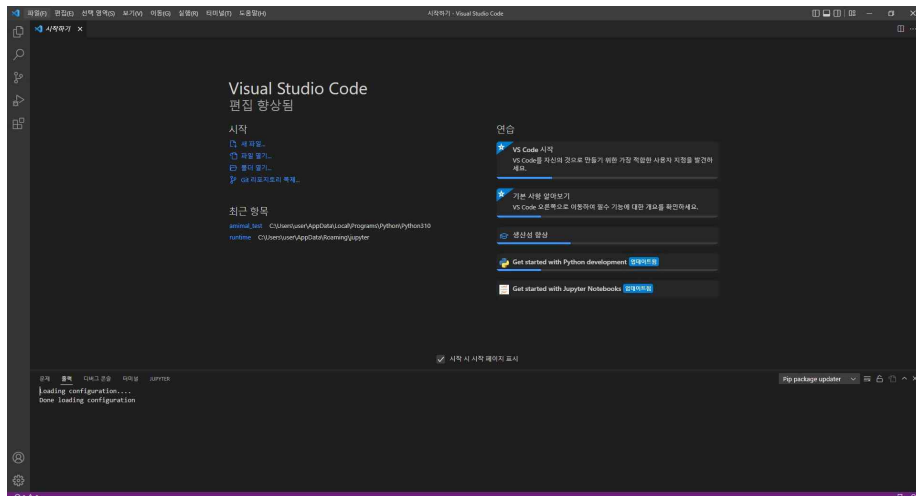
동의합니다 체크 후 다음.



체크박스 모두 체크 후 다음.

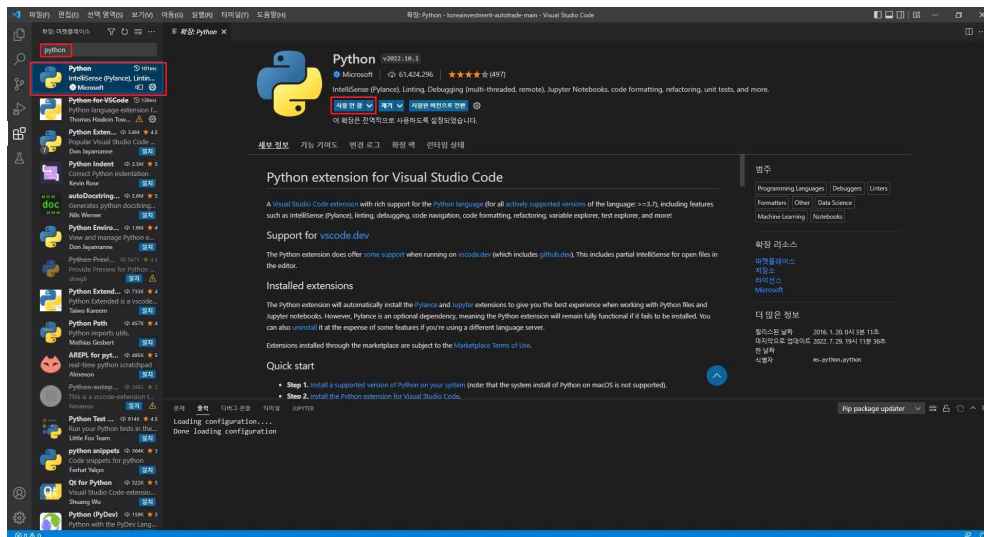


설치 완료 후 종료를 하면 Visual Studio Code가 실행됩니다.



실행된 Visual Studio Code의 모습입니다. Visual Studio Code에서 파이썬 (Python)이 잘 작동하는지 확인을 해보겠습니다.

위 화면의 좌측 아이콘 중 빨간 박스 아이콘을 클릭하여 들어가면



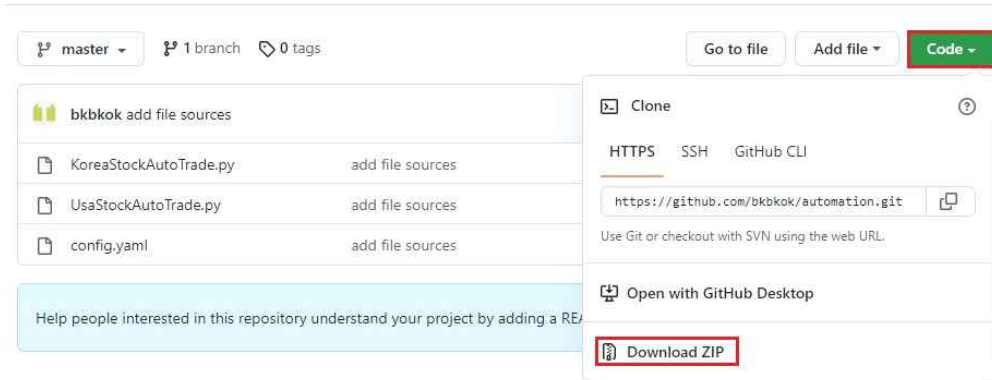
다음 그림과 같이 왼쪽 위에 검색창이 뜰 것입니다. 그곳에 python이라고 적고 엔터를 치면 여러 목록이 나오는데 첫 번째 Python을 클릭합니다. 가운데 화면에 Python에 대한 정보가 나오는데 만약 설치가 안 되었다면 빨간박스 안에 설치라고 나올 것입니다. 설치가 안 되었다면 설치를 해주시면 됩니다.

이로써 자동화 로직을 위한 준비가 모두 끝났습니다.

4. 국내 주식투자 자동화

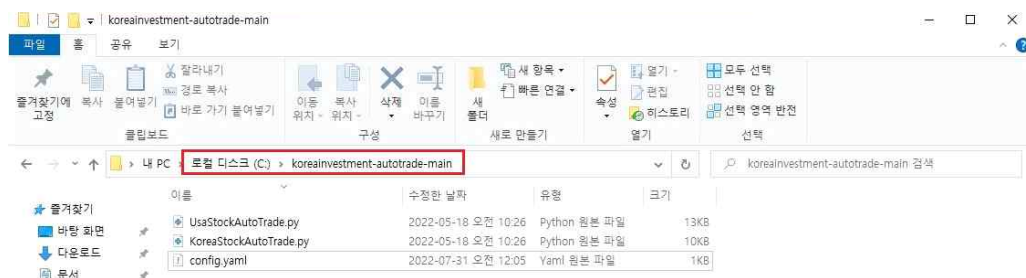
국내 주식투자 자동화를 위한 코드를 예시로 업로드하였습니다.

다운로드는 <https://github.com/bkbbkok/automation> 에서 받을 수 있습니다.

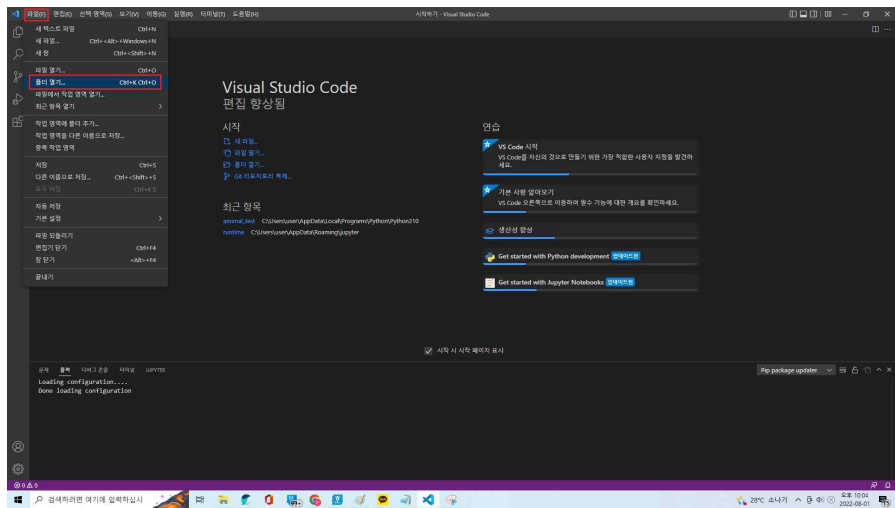


위 그림과 같이 Code를 클릭하시고 아래쪽에 Download ZIP을 누르시면 압축파일로 다운을 받을 수 있습니다.

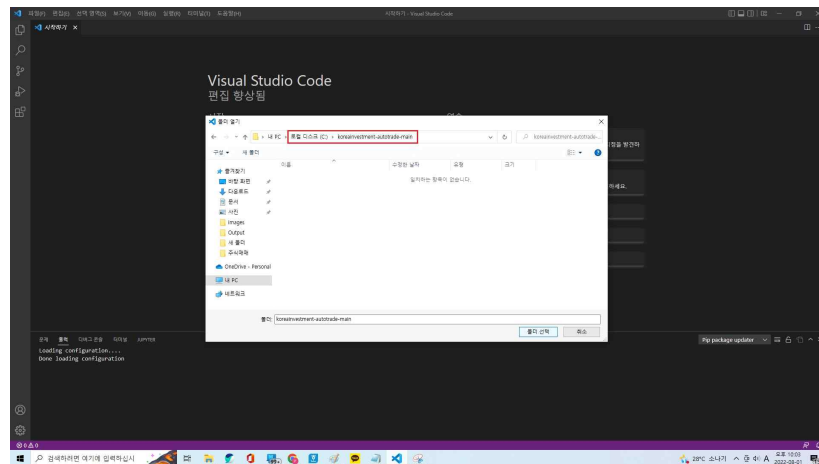
압축을 해제 후 나온 폴더를 C: 바로 아래 붙여넣기 해주세요.



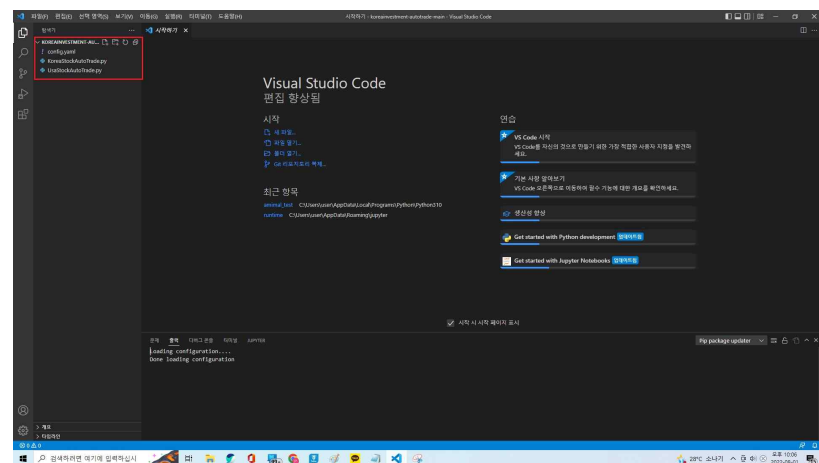
다음으로 Visual Studio Code를 실행시켜 줍니다.



파일 - 폴더열기를 클릭 후 아까 진행한 C: 아래 붙여넣은 폴더를 선택하여 줍니다.



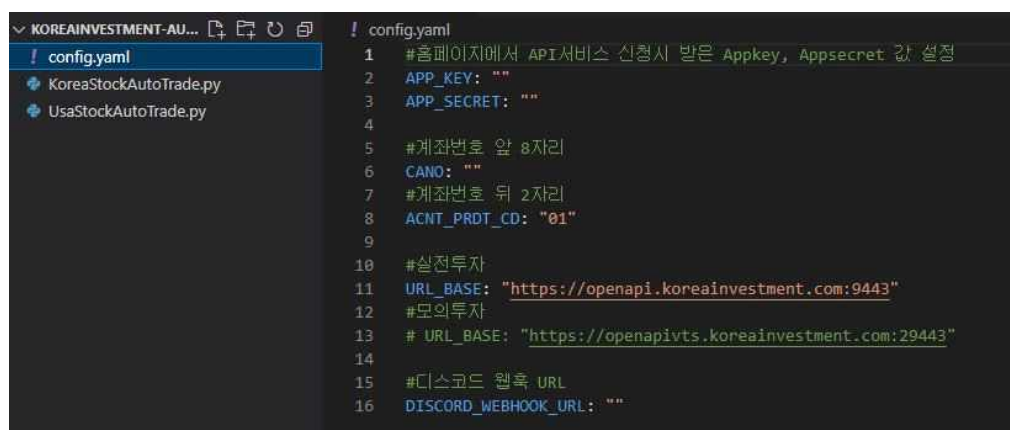
완료되면 왼쪽 위에 다운받은 코드들이 열린 것을 확인할 수 있습니다.



위 3개의 파일은 조코딩님이 만든 소스로 간단하게 만든 것이므로 오류가 있을 수 있다고 명시해놨지만, 테스트 결과 잘 작동하였습니다. 혹여 작동이 잘 안 된다면 bkbkok@gmail.com으로 문의하시면 확인 후 답변드리겠습니다.

첫 번째 파일을 살펴보면 API 키와 계좌번호 디스코드 웹훅 URL 등을 입력하는 config 파일입니다.

두 번째 파일은 국내 주식투자 자동화에 사용되는 파일과 세 번째는 해외 주식투자에 사용되는 파일로 구성되어 있습니다.

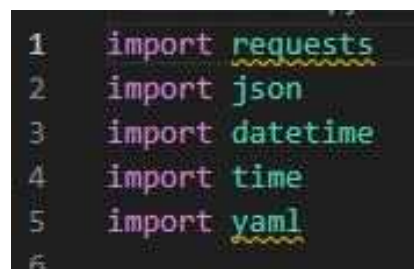


```
1 #홈페이지에서 API서비스 신청시 받은 Appkey, Appsecret 값 설정
2 APP_KEY: ""
3 APP_SECRET: ""
4
5 #계좌번호 앞 8자리
6 CANO: ""
7 #계좌번호 뒤 2자리
8 ACNT_PRDT_CD: "01"
9
10 #실전투자
11 URL_BASE: "https://openapi.koreainvestment.com:9443"
12 #모의투자
13 # URL_BASE: "https://openapivts.koreainvestment.com:29443"
14
15 #디스코드 웹훅 URL
16 DISCORD_WEBHOOK_URL: ""
```

APP_KEY와 APP_SECRET은 KIS Developer에 신청 후 발급받은 값을 입력해 줍니다.

계좌번호 앞 8자리와 뒤 2자리는 앱 또는 홈페이지에서 로그인 후 확인 가능합니다.

마지막 줄에 디스코드에서 발급받은 웹훅 URL 복사한 값을 입력해 줍니다.



```
1 import requests
2 import json
3 import datetime
4 import time
5 import yaml
6
```

두 번째 파일로 넘어가면 맨 위의 5줄은 필요한 라이브러리를 불러오는 코드입니다.

```

7   with open('config.yaml', encoding='UTF-8') as f:
8       _cfg = yaml.load(f, Loader=yaml.FullLoader)
9       APP_KEY = _cfg['APP_KEY']
10      APP_SECRET = _cfg['APP_SECRET']
11      ACCESS_TOKEN = ""
12      CANO = _cfg['CANO']
13      ACNT_PRDT_CD = _cfg['ACNT_PRDT_CD']
14      DISCORD_WEBHOOK_URL = _cfg['DISCORD_WEBHOOK_URL']
15      URL_BASE = _cfg['URL_BASE']
16

```

그다음 소스는 첫 번째 소개한 기본 정보를 넣은 config 파일을 불러오는 소스입니다.

```

17  def send_message(msg):
18      """디스코드 메시지 전송"""
19      now = datetime.datetime.now()
20      message = {"content": f"[{now.strftime('%Y-%m-%d %H:%M:%S')}] {str(msg)}"}
21      requests.post(DISCORD_WEBHOOK_URL, data=message)
22      print(message)
23
24  def get_access_token():
25      """토큰 발급"""
26      headers = {"content-type": "application/json"}
27      body = {"grant_type": "client_credentials",
28             "appkey": APP_KEY,
29             "appsecret": APP_SECRET}
30      PATH = "oauth2/tokenP"
31      URL = f"{URL_BASE}/{PATH}"
32      res = requests.post(URL, headers=headers, data=json.dumps(body))
33      ACCESS_TOKEN = res.json()["access_token"]
34      return ACCESS_TOKEN
35
26  def backkey(data):

```

그 밑으로는 자동매매에 필요한 여러 함수가 입력되어 있습니다. 디스코드 메시지 전송부터 조건부 매수, 매매에 필요한 함수들이 포함되어 있습니다.

```

211  # 자동매매 시작
212  try:
213      ACCESS_TOKEN = get_access_token()
214
215      symbol_list = ["005930", "035720", "000660", "069500"] # 매수 희망 종목 리스트
216      bought_list = [] # 매수 완료된 종목 리스트
217      total_cash = get_balance() # 보유 현금 조회
218      stock_dict = get_stock_balance() # 보유 주식 조회
219      for sym in stock_dict.keys():
220          bought_list.append(sym)
221      target_buy_count = 3 # 매수할 종목 수
222      buy_percent = 0.33 # 종목당 매수 금액 비율
223      buy_amount = total_cash * buy_percent # 종목별 주문 금액 계산
224      soldout = False
225
226      send_message("===국내 주식 자동매매 프로그램을 시작합니다===")
227      while True:
228          t_now = datetime.datetime.now()
229          t_9 = t_now.replace(hour=9, minute=0, second=0, microsecond=0)
230          t_start = t_now.replace(hour=9, minute=5, second=0, microsecond=0)

```

본격적인 매수, 매매에 필요한 로직은 212번째 줄부터 시작됩니다.

```
213 ACCESS_TOKEN = get_access_token()
```

ACCESS_TOKEN은 API를 사용하기 위해 발급하는 토큰으로 유효기간이 짧으므로 자동매매를 시작할 때마다 발급받도록 만들어 놔습니다.

```
215 symbol_list = ["005930", "035720", "000660", "069500"] # 매수 희망 종목 리스트
```

symbol_list는 매수를 원하는 종목 코드를 넣어줍니다. 추가를 원하면 콤마(,)로 구분 후 “종목 코드”를 추가하시면 됩니다.

```
216 bought_list = [] # 매수 완료된 종목 리스트
217 total_cash = get_balance() # 보유 현금 조회
218 stock_dict = get_stock_balance() # 보유 주식 조회
219 for sym in stock_dict.keys():
220     bought_list.append(sym)
221 target_buy_count = 3 # 매수할 종목 수
222 buy_percent = 0.33 # 종목당 매수 금액 비율
223 buy_amount = total_cash * buy_percent # 종목별 주문 금액 계산
224 soldout = False
```

매수가 완료된 종목은 bought_list로 표시할 수 있고 현재 보유 현금 조회, 보유 주식 조회, 등을 표시되게 코딩을 해 놓으셨습니다.

target_buy_count는 본인이 매수할 종목 수를 적어 넣으시면 됩니다. 매수 희망 종목 리스트에 4개를 넣었으면 조건에 부합하는 종목을 최대 3개까지 매수하겠다는 의미입니다.

buy_percent는 전체 보유금액의 몇 퍼센트를 한 종목당 매수에 투자할 것인가 퍼센트 비율을 적어주시면 됩니다.


```

226 send_message("==국내 주식 자동매매 프로그램을 시작합니다==")
227 while True:
228     t_now = datetime.datetime.now()
229     t_9 = t_now.replace(hour=9, minute=0, second=0, microsecond=0)
230     t_start = t_now.replace(hour=9, minute=5, second=0, microsecond=0)
231     t_sell = t_now.replace(hour=15, minute=15, second=0, microsecond=0)
232     t_exit = t_now.replace(hour=15, minute=20, second=0, microsecond=0)
233     today = datetime.datetime.today().weekday()
234     if today == 5 or today == 6: # 토요일이나 일요일이면 자동 종료
235         send_message("주말이므로 프로그램을 종료합니다.")
236         break
237     if t_9 < t_now < t_start and soldout == False: # 잔여 수량 매도
238         for sym, qty in stock_dict.items():
239             sell(sym, qty)
240             soldout = True
241             bought_list = []
242             stock_dict = get_stock_balance()
243     if t_start < t_now < t_sell: # AM 09:05 ~ PM 03:15 : 매수
244         for sym in symbol_list:
245             if len(bought_list) < target_buy_count:
246                 if sym in bought_list:
247                     continue
248                 target_price = get_target_price(sym)
249                 current_price = get_current_price(sym)
250                 if target_price < current_price:
251                     buy_qty = 0 # 매수할 수량 초기화
252                     buy_qty = int(buy_amount // current_price)
253                     if buy_qty > 0:
254                         send_message(f"{sym} 목표가 달성({target_price} < {current_price}) 매수를 시도합니다.")
255                         result = buy(sym, buy_qty)
256                         if result:
257                             soldout = False
258                             bought_list.append(sym)
259                             get_stock_balance()

```

이제 프로그램을 실행을 시작하였으니 디스코드로 메시지를 보냅니다.

datetime 모듈을 이용해 시간을 불러오고 주말엔 프로그램이 자동 종료될 수 있게 만들어 놓습니다.

잔여 수량 매도는 현재 9시부터(229줄) 프로그램 시작 시간인 9시 5분(230줄)으로 설정하였는데 9시~9시 5분 사이에 전날의 잔여 수량이 있다면 모두 매도하여 장 시작 시 보유 주식의 상태가 0으로 시작될 수 있도록 코딩되었습니다. 9시 5분부터 3시 15분까지 설정된 값으로 매수를 진행하게 되어 있습니다(243줄).

244줄 for sym - 앞서 저장한 symbol_list 즉 종목 코드를 하나씩 불러와 원하는 만큼의 주식을 사지 않았다면(245줄) 목표가격과(248줄) 현재 가격(249줄)을 불러와서 (250줄) 목표가격보다 현재 가격이 높아진 상태이면 설정한 금액에 따라 매수할 수량을 정하고(252줄) 그 수량에 맞게 매수를 진행(253줄)하는 코드입니다.

이 로직의 핵심은 target_price 즉 목표가가 얼마인지 설정하는 겁니다.

```

248 target_price = get_target_price(sym)

```

Ctrl 키를 누른 상태에서 빨간 박스 안의 get_target_price를 클릭하면 해당 함수가 어떻게 구성되어 있는지 확인할 수 있습니다.

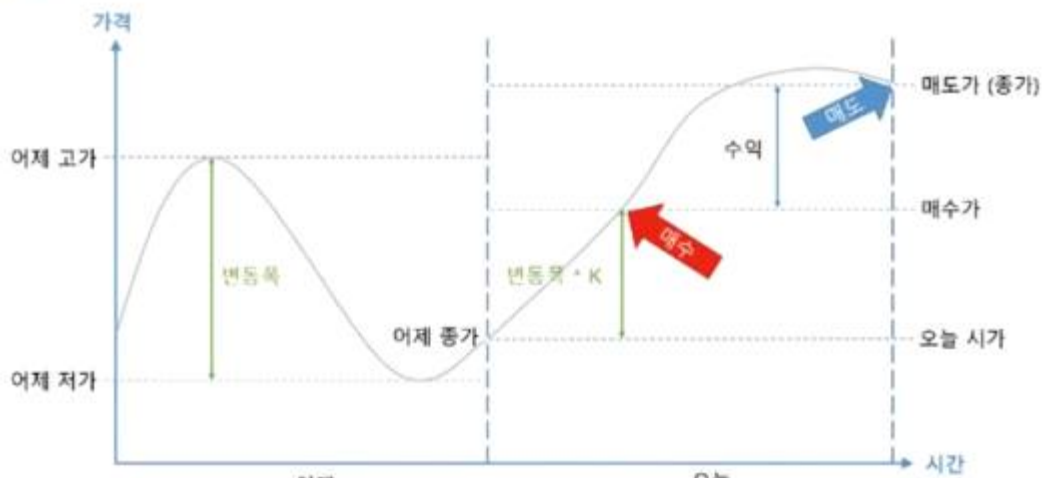

```

65 def get_target_price(code="005930"):
66     """변동성 돌파 전략으로 매수 목표가 조회"""
67     PATH = "uapi/domestic-stock/v1/quotations/inquire-daily-price"
68     URL = f"{URL_BASE}/{PATH}"
69     headers = {"Content-Type": "application/json",
70               "authorization": f"Bearer {ACCESS_TOKEN}",
71               "appKey": APP_KEY,
72               "appSecret": APP_SECRET,
73               "tr_id": "FHKST01010400"}
74     params = {
75         "fid_cond_mrkt_div_code": "J",
76         "fid_input_iscd": code,
77         "fid_org_adj_prc": "1",
78         "fid_period_div_code": "D"
79     }
80     res = requests.get(URL, headers=headers, params=params)
81     stck_oprc = int(res.json()['output'][0]['stck_oprc']) #오늘 시가
82     stck_hgpr = int(res.json()['output'][1]['stck_hgpr']) #전일 고가
83     stck_lwpr = int(res.json()['output'][1]['stck_lwpr']) #전일 저가
84     target_price = stck_oprc + (stck_hgpr - stck_lwpr) * 0.5
85     return target_price
86

```

변동성 돌파 전략을 이용한 매수 예시를 든 것입니다. API를 이용하여 오늘 시가, 전일 고가, 전일 저가를 이용한 것으로 (84줄)목표가 = 오늘 시가 + (전일 고가 - 전일 저가) * 0.5(K)

그림_변동성 돌파 전략



출처: 한빛미디어 <파이썬 증권 데이터 분석>

K값은 원하는 값으로 설정하시면 됩니다.

```

262 ✓ if t_now.minute == 30 and t_now.second <= 5:
263     get_stock_balance()
264     time.sleep(5)

```

매시간 30분마다 현재 주식 잔고(263줄)를 보여주는 코드도 들어가 있습니다.

```

265         if t_sell < t_now < t_exit: # PM 03:15 ~ PM 03:20 : 일괄 매도
266             if soldout == False:
267                 stock_dict = get_stock_balance()
268                 for sym, qty in stock_dict.items():
269                     sell(sym, qty)
270                 soldout = True
271                 bought_list = []
272                 time.sleep(1)
273         if t_exit < t_now: # PM 03:20 ~ :프로그램 종료
274             send_message("프로그램을 종료합니다.")
275             break

```

3시 15분~3시 20분 사이엔 일괄 매도하도록 코딩이 되어 있고 3시 20분엔 '프로그램을 종료합니다.'라는 메시지와 함께 종료되도록 코드가 짜여 있습니다.

이제 실행만 남겨놓고 있습니다.

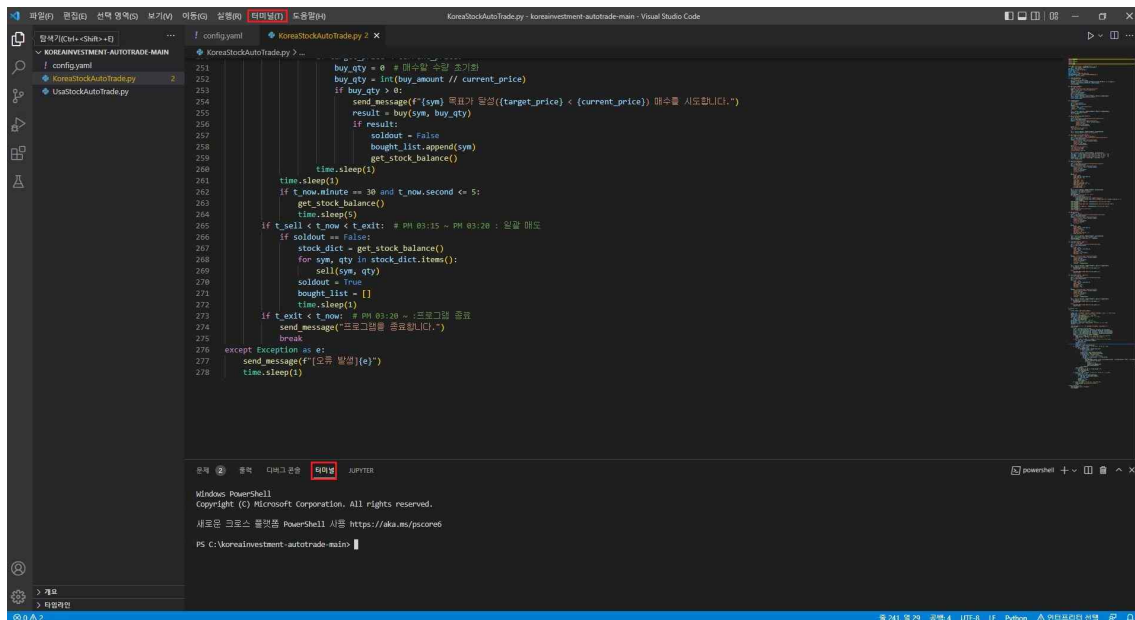
먼저 코드를 실행하려면 이 코드에 사용된 라이브러리들을 설치해야 합니다.

```

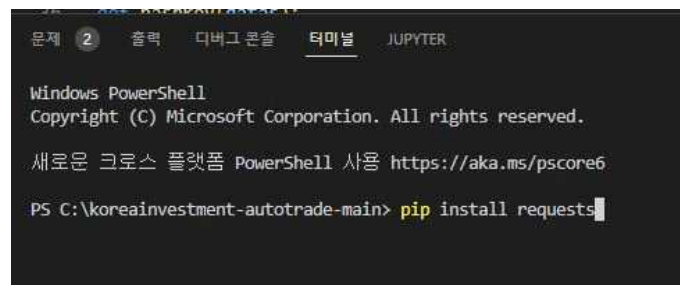
1  import requests
2  import json
3  import datetime
4  import time
5  import yaml
6

```

만약 설치가 안 되어 있다면 밑줄이 그어져 있습니다. 이럴 때 쉬운 방법으로 라이브러리를 설치할 수 있습니다.



상단 메뉴의 터미널 - 새 터미널 또는 아래 빨간 박스 안의 터미널을 클릭하면 하단에 텍스트를 입력할 수 있는 공간이 생깁니다.



위의 사진처럼 'pip install 라이브러리명'을 적어주고 엔터를 치면 설치가 됩니다.

혹시 맥을 사용 중이신 분은 'pip3 install 라이브러리명'을 입력해 주시면 됩니다.

이제 완성된 파일을 실행시켜야겠죠?

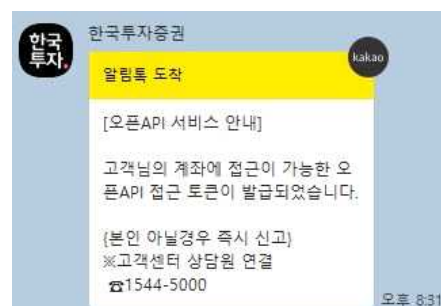
실행법은 2가지 방법이 있습니다.

```

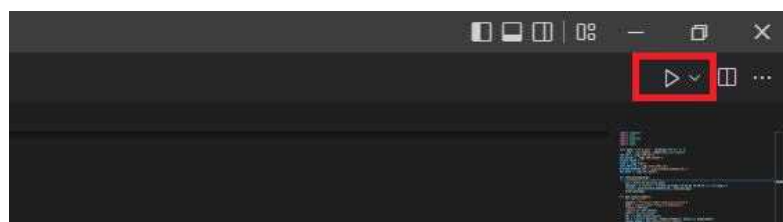
PS C:\koreainvestment-autotrade-main> python .\KoreaStockAutoTrade.py
{'content': '[2022-08-02 20:31:04] 주문 가능 현금 잔고: 0원'}
{'content': '[2022-08-02 20:31:04] ===주식 보유잔고==='}
{'content': '[2022-08-02 20:31:05] 주식 평가 금액: 0원'}
{'content': '[2022-08-02 20:31:05] 평가 손익 합계: 0원'}
{'content': '[2022-08-02 20:31:05] 총 평가 금액: 0원'}
{'content': '[2022-08-02 20:31:06] ====='}
{'content': '[2022-08-02 20:31:06] ===국내 주식 자동매매 프로그램을 시작합니다==='}
{'content': '[2022-08-02 20:31:07] 프로그램을 종료합니다.'}
PS C:\koreainvestment-autotrade-main>

```

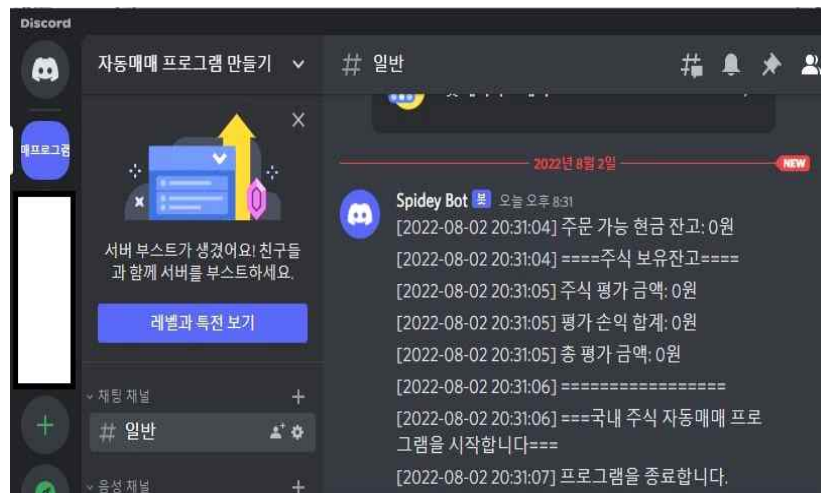
터미널을 열고 python korea까지만 작성 후 Tap 버튼을 누르면 파일명이 자동으로 입력됩니다. 그리고 엔터를 누르면 위 화면처럼 실행이 되는 것을 볼 수 있습니다. 잔고가 없고 장이 종료된 시점이므로 자동으로 프로그램을 종료하는 것을 볼 수 있습니다.



실행시킨 시점에서 API에 접근할 수 있는 토큰이 자동으로 발급되어 카카오톡으로 알림이 오는 것을 확인할 수 있었습니다.

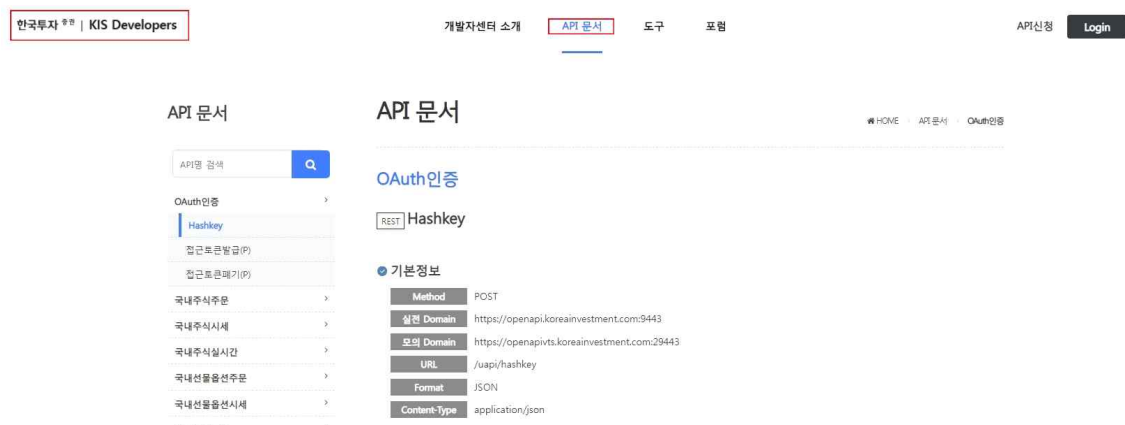


두 번째 실행 방법으로는 Visual Studio Code의 오른쪽 위를 보면 ▶ 모양의 아이콘을 클릭함으로 실행을 시킬 수도 있습니다.



그리고 디스코드 웹훅 URL을 설정해 줬기 때문에 디스코드 채팅 메시지에
도 같은 메시지가 출력되는 것을 확인할 수 있습니다. 스마트폰으로 디스코드
를 이용 중이라면 메시지 알림으로도 확인 가능합니다.
사용 중간에 프로그램을 멈추고 싶다면 Ctrl + C 단축키를 사용하면 종료가
됩니다.

위의 파일은 예시로 만들어진 프로그램이기에 자신의 투자 전략에 맞게 수정
하여 사용하시면 됩니다.



코드 변경은 KIS developer의 API 문서를 보면 상세한 내용을 확인 가능하
며,

도구

- 테스트베드
- Github
- Wikidocs

테스트베드

HOMES > 도구 > 테스트베드

KIS Developers 테스트베드

API에 접근하여 편리하게 테스트할 수 있는 환경입니다. 다만 보안을 위해 정해진 Output만을 확인할 수 있으나, 상세한 내용은 API 문서에서 확인 부탁드립니다.

API Swagger 테스트 API 서비스를 선택하세요 API개요로 이동 검색하기

도구 탭의 Wikidocs를 클릭하면 자세한 설명과 함께 예시 코드를 확인할 수 있어 다양한 전략을 구성할 수 있습니다.

WikiDocs < <위키독스> 안드로이드, 아이폰 앱 출시 > 로그인

한국투자증권

한줄 소개
위키독스 저장입니다.

첫번째 컴퓨터
기억이 안나요.

나의 URL
wikidocs.net

저서 최근댓글 최근수정

추천순 인기순 최신순

파이썬으로 배우는 오픈API 트레이딩 초급 예제

저자: 한국투자증권, 승식

Python을 활용한 한국투자증권 KIS Developers 트레이딩 오픈API 예제입니다. 서비스 신청부터 국내주식시세조회, 매수, 매도까지 조급 수준에 적합한 매매 방법을 안내합니다. 22.03.04. 기준으로 업데이트 되었으며, 문의 사항은 KIS Developers 내 Q...

수정: 2022년 7월 12일 2:57 오후

파이썬으로 배우는 한국투자증권 Websocket 사용 예제

저자: 한국투자증권

Python을 활용한 한국투자증권 KIS Developers 트레이딩 오픈API Websocket 예제입니다. 실시간으로 주식 호가, 체결가, 체결 여부 등을 확인할 수 있습니다. 22.03.31. 기준으로 업데이트 되었으며, 문의 사항은 KIS Developers 내 Q&A 개...

수정: 2022년 7월 4일 12:39 오후

이전 다음

영역어를 입력하세요.

■ 파이썬으로 배우는 오픈API 트레이딩 초급 예제

00. 한국투자증권 KIS Developers 소개

01. 개발 환경 준비

02. 서비스 신청

01) 오픈API 서비스 신청

02) KIS Developers 로그인

03. 서비스 연결 (OAuth)

01) 보안인증키 발급

02) 해쉬키 (hashkey) 발급

04. 현재가 조회

01) 주식현재가 시세

02) 주식현재가 일자별

05. 매매

01) 주식주문(현금) - 매수

02) 주식주문(정정취소)

03) 주식주문(현금) - 매도

06. 기타

07. 샘플코드

08. 전체

⑤ API로용 샘플(kis_api.py)

2022. 7. 12. 2:57

■ 파이썬으로 배우는 오픈API 트레이딩 ... / 04. 현재가 조회 / 02) 주식현재가 일자별

WikiDocs

02) 주식현재가 일자별

다음으로는 주식현재가 일자별 API 를 호출해보겠습니다. 물 가져와보겠습니다. 주식현재가 일자별 API 는 조회하고 싶은 종목의 일자별 정보를 얻는데 사용합니다. 그럼 시작해보겠습니다.

```
#URL_BASE = "https://openapi.koreainvestment.com:29443"
PATH = "/uapi/domestic-stock/v1/quotations/inquire-daily-price"
URL = f'{URL_BASE}/{PATH}'
print(URL)

>>> https://openapi.koreainvestment.com:29443/uapi/domestic-stock/v1/quotations/inquire-daily-price
```

가장 먼저, PATH 설정을 합니다. [3. 서비스 연결 > 01) 보안인증키 발급] 에 PATH에 관한 내용이 있습니다.

```
#headers = {"Content-Type": "application/json",
#           "authorization": f"Bearer {ACCESS_TOKEN}",
#           "appkey": APP_KEY,
#           "appSecret": APP_SECRET,
#           "tr_id": "FHKS701010100"}

headers["tr_id"] = "FHKS701010400"
```

다음으로는 headers 내에 tr_id 를 입력합니다. 언급했던 것과 같이 각 API 별로 다른 tr_id 를 갖고 있어 각 API 를 사용할 때마다 다른 tr_id 를 입력해주셔야 합니다. 이 때, headers의 다른 값들은 공통적으로 사용되기에, tr_id만 교체합니다. 주식현재가 시세의 tr_id는 "FHKS701010400" 이므로 headers 에 해당 값을 넣어줍니다.

```
params = {
    "fid_cond_mkt_div_code": "3",
    "fid_input_iscd": "009930",
    "fid_org_rdt_src": "1",
    "fid_period_div_code": "D"
}
```

5. 해외주식 투자 자동화

```
255     send_message("===해외 주식 자동매매 프로그램을 시작합니다===")
256     while True:
257         t_now = datetime.datetime.now(timezone('America/New_York')) # 뉴욕 기준 현재 시간
258         t_9 = t_now.replace(hour=9, minute=30, second=0, microsecond=0)
259         t_start = t_now.replace(hour=9, minute=35, second=0, microsecond=0)
260         t_sell = t_now.replace(hour=15, minute=45, second=0, microsecond=0)
261         t_exit = t_now.replace(hour=15, minute=50, second=0, microsecond=0)
262         today = t_now.weekday()
```

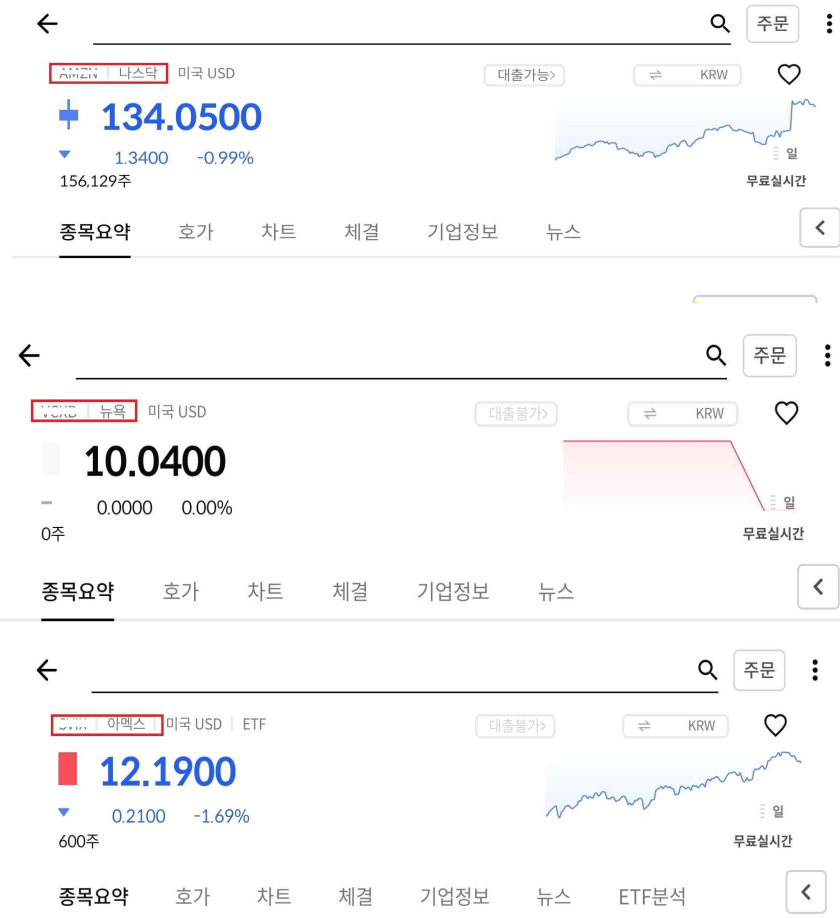
기본적인 구성은 비슷하나 해외주식이기엔 현재시간을 불러올 때 기준이 되는 뉴욕의 현재 시간을 불러올 수 있게 되어있습니다.

```
240     nasd_symbol_list = ["AAPL"] # 매수 희망 종목 리스트 (NASDAQ)
241     nyse_symbol_list = ["KO"] # 매수 희망 종목 리스트 (NYSE)
242     amex_symbol_list = ["LIT"] # 매수 희망 종목 리스트 (AMEX)
243     symbol_list = nasd_symbol_list + nyse_symbol_list + amex_symbol_list
244     bought_list = [] # 매수 완료된 종목 리스트
245     total_cash = get_balance() # 보유 현금 조회
246     exchange_rate = get_exchange_rate() # 환율 조회
247     stock_dict = get_stock_balance() # 보유 주식 조회
248     for sym in stock_dict.keys():
249         bought_list.append(sym)
250     target_buy_count = 3 # 매수할 종목 수
251     buy_percent = 0.33 # 종목당 매수 금액 비율
252     buy_amount = total_cash * buy_percent / exchange_rate # 종목별 주문 금액 계산 (달러)
253     soldout = False
254
```

해외 주식 시장은 크게 세 개의 시장으로 나누어서 등록이 필요합니다.

매수를 희망하는 종목이 나스닥에 상장되어 있는지(240줄), 뉴욕증권거래소에 상장되어 있는지(241줄), 아멕스에 상장되어 있는지(242줄)를 분류해서 작성하여 주시면 됩니다.

해외주식의 코드와 어디에 상장되어 있는지 확인하는 법을 알려드리겠습니다.



왼쪽 위의 빨간 박스 안의 앞 영어 대문자 부분이 종목 코드, 뒤에 나와 있는 나스닥, 뉴욕, 아멕스의 표시가 상장된 시장의 구분입니다.

예시로 설정된 미국 주식 자동화 매매 코드 또한 변동성 돌파 전략으로 구현되어 있습니다.

KIS developer 사이트의 API 문서 탭의 해외주식 주문 부분을 살펴보면 미국 증시 외 일본, 상해, 홍콩 등의 코드들도 제시되어 있으니 본인의 취향에 맞게 변경하여 사용하시면 됩니다.

자신만의 매매 전략법이 있으시다면 Wikidocs의 내용을 보며 로직을 변경하여 사용하시면 좋을듯합니다!