

# Introduction to 'Derivatives and Cryptocurrency Automated Trading using API' for Non-programmers

Editor : 말과 의미  
(파생인의심터 카페 매니저)



파생인의심터 카페 회원에게 제공되는 국내최저 협의수수료  
<http://cafe.naver.com/fo24>  
 파생인의심터 국내파생, 해외파생 협의수수료(제휴수수료) 정리 2024년 기준, 온라인 거래 기준

상품	구분	타증권사 협의수수료	파생인의심터 협의수수료	적용조건	비고
코스피선물	코스피200 선물(주간, 야간, 미니, 위클리)	0.0015%	0.0007%	조건없음	타증권사 협의수수료 대비 30% 이상 추가 할인
	코스피200 야간선물(EUREX)	0.0025%	0.0015%		
	결제수수료 및 오프수수료	0.01%	0.01%		
코스피옵션	코스피200 옵션(미니, 위클리, 개별주식)	0.04%	0.025%		
	코스피200 야간옵션(EUREX)	0.1%	0.08%		
	결제수수료 및 오프수수료	0.3%	0.3%		
코스닥선물	코스닥150 선물	0.006%	0.002%		
주식선물	주식선물	0.006%	0.002%		
채권선물, 달러선물			별도 요청		
해외파생 (해외선물, 해외옵션)	CME 거래소 등 USD 기준	2.59달러	2.29\$		
	마이크로 상품(Mirco USD)	1달러	0.9\$		
	EUREX 거래소 등 유로(EUR) 기준	2.59유로	2.29€		
	항셱 등 홍콩달러(HKD) 기준	25.9HK	22.9HK		
	일본 마이크로 엔화상품(Micro JPY)	150JPY	100¥		
	일본 엔화상품(JPY)	300JPY	300¥		
주식	국내주식	0.015%	0.01%	조건 : 주식평가액 1천만원 이상	
	해외주식	0.1%	0.07%		
	환전우대	90%	100%		
CMA	CMA 금리는 수시 변동됨(2024년 2월 기준)	금리와 이자는 수시로 변동 별도 문의 요청!		잔고 5천만원 이상, 지점 내방	
대출이자	국내주식대출, 해외주식대출			3개월 후 재협의	

\* 본 협의수수료는 개인은 DB금융투자 비대면 계좌개설의 경우에만 적용, 법인인 가까운 지점에 방문해서 계좌 개설한 경우로 한정  
 \* 법인은 자본금 5천만원 이상 법인에게만 위 협의수수료가 적용됩니다  
 \* 본 협의수수료는 거래가 지속되고 있는 한 계속해서 적용받게 되나, 거래가 1년 이상 없는 경우 협의수수료는 재적용되지 않으며, 이 경우 협의수수료 재적용 요청시 변경된 협의수수료가 적용될 수 있습니다.  
 \* 국내 코스피200옵션, 미니옵션 거래시 옵션가격 0.42미만(미니는 0.48미만) 거래시 13원 수수료 추가 부과  
 \* 옵션매도하려면 파생 계좌에 자산 2천만원 이상 있어야 함  
 \* 온라인 최저수수료 적용으로 국내파생 만기결제수수료는 선물 0.01%, 옵션 0.3%로 적용.  
 \* 국제선물, 달러선물도 협의수수료 적용 가능(최근 6개월간 월 평균 100만원 이상 수수료 발생한 경우 추가 신청. 국제선물 계약당 1천원, 달러선물 계약당 2백원 협의수수료)

# Contents

- 1) Non-Programmer를 위한 API 자동매매 기초과정 Overview (1)
  - 1-1) Non-Programmer를 위한 API 자동매매 기초과정 Overview (2)
  - 1-2) Automated Trading using API with ChatGPT
- 2) What API is & Differences between HTS API and Rest API
  - 2-1) Exchange의 API 구조의 이해 (Component API와 Rest API)
  - 2-2) TR and RTD Approach in HTS-Based API vs. Web-based API
- 3) Terms you need to know when using API
  - 3-1) 증권사 vs. Upbit의 시세 조회 API Coding 비교
- 4) Understanding of Coding and Programming
  - 4-1) Selection Program language as a non-Programmer
  - 4-2) How to Read a code written by AI

- 5) Most Popular Python Lib used in Automated Trading
  - 5-1) Comparison between Python and C#
- 6) Deep Learning, AI, ChatGPT
  - 6-1) How to use ChatGPT for a Trader
  - 6-2) ChatGPT as a Coding Assistant(1)
  - 6-3) ChatGPT as a Coding Assistant(2)
  - 6-4) Neo Automated Trading Algorithm made by AI
- 7) Negative Sum Game (Zero Sum Game)의 본질적 이해가 필요
  - 7-1) How to get a winning edge in Zero-Sum Game
  - 7-2) Positive Sum Game with Leverage (Debt)
- 8) Briefing Schedule of Automated Trading using API with ChatPGT'

# 1) Non-Programmer를 위한 API 자동매매 Overview (1)

Everybody can do automated trading! But, 자동매매한다고 꼭 수익이 나는 건 아니다! 그전에 Zero-sum game의 이해가 선행

## a) Why must you do automated trading?

왜 암호화폐와 선물옵션시장에서 자동매매를 해야 하는지?

자동매매와 손매매의 장단점 비교 / API의 기초개념 등 <https://cafe.naver.com/fo24/34146>

## b) Automated Trading vs. System Trading

- System Trading(기계적 매매, entry-exit-losscut)\_1980년대 이후 컴퓨터의 발달로 각종 보조지표가 개발되고(John Bollinger 1980) 통계적 데이터로 시스템적 매매 시작. **매매 규칙을 잘 지키면 손매매도 해당!**

- Automated Trading(**핵심은 주문의 자동화.** '시장 데이터 분석, **매매 알고리즘대로 주문 자동 실행**, 실행된 거래의 모니터링 과 전략의 최적화' 등이 포함되는 자동매매 시스템 구축이 최종 목표임). 1989년 자동주문이 가능해진 자동매매 tool인 Tradestation이 시장에 출시. 과체적화, Forward Testing 등 이론 발전.

## c) 자동매매해도 수익이 지속적으로 나지는 않는다. Zero-Sum Game의 이해가 선행

왜 Individual Trader가 선물옵션과 같은 시장에서는 장기적으로 수익을 내기 불가능한지 이론적 근거를 먼저 충분히 습득하고, **Zero-Sum game(In face, Negative Game)에서 일시적 Luck이 아닌 장기적, 지속적 수익을 내기 위해서는 어떻게 해야 하는지 공부를 충분히 하고 '자동매매'의 세계에 입문하자!**



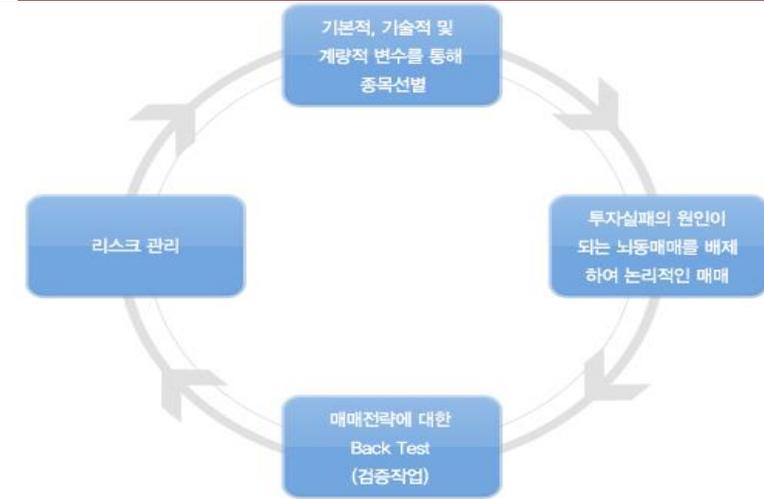
# 1-1) Non-Programmer를 위한 API 자동매매 Overview (2)

If Close crosses above Average(Close, 20) then Buy this bar at close; If Close crosses below Average(Close, 20) then Sell this bar at close;

## a) Why must you do Automated Trading?

대부분의 Derivatives Individual Traders는 역추세 매매자로, 'Counter-trend Trading'은 승률이 높아서 평상시에는 자주 작은 수익을 보지만, **One-way trend 발생시 Stop-loss를 하지 못하고 오히려 몰타기(Averaging Down)를 하다, 큰 손해를 입게 된다.**

선물옵션을 3년 이상 거래한 전업 트레이더들도 몇달간 수익을 잘 내다가도, Mind control의 실패로 매매규칙을 어기고, 99번 손절을 잘 하다가도 1번의 손절을 못해서 큰 손해를 입는데, 역시 이렇게 심리적으로 무너져서 큰 손실이 나는 것이 **자동매매가 유일한 대안!**



## b) API와 Coding에 대한 두려움이 있다면 자동매매 플랫폼 또는 Bot 활용하기

HTS 내의 조건검색 식이나 주식관리를 이용하거나, DDE(RTD real time date) 등에서 데이터를 받아서 활용하는 방법부터 시작

비프로그래머를 위한 다양한 자동매매 Platform이나 Bot을 활용해보자.

간단한 랭귀지로 매매알고리즘을 만들어서 매매하기(Yestrader, Signalmaker, **Tradingview**, **POAbot** 등) 다만, 이런 Platform 역시 Yes language나 Pine script 등의 간단한 Coding이 필요함. (Yes language는 주식지왕 블로그 참고, **Pine Script는 ChatGPT가 코딩해줌**)

특히 'Cryptocurrency market'은 다양한 자동매매Bot과 Platform이 존재

**Tradingview (해외선물, 암호화폐) <https://www.tradingview.com/>**

**POAbot(암호화폐, 해외주식, 한국투자증권) <https://poabot.com/>**

```
pinestript
Copy code

//@version=5
strategy("My Custom Strategy", overlay=true)

// 매수 조건을 계산
// 직전 3개의 봉이 모두 음봉인지 확인
isPrev3Bearish = close[1] < open[1] and close[2] < open[2] and close[3] < open[3]

// 현재 봉이 양봉인지 확인
isCurrentBullish = close > open

// 매수 신호: 직전 3개의 봉이 음봉이고 현재 봉이 양봉일 때
buySignal = isPrev3Bearish and isCurrentBullish

// 매수 신호에 따른 전략 실행
if (buySignal)
    strategy.entry("Buy", strategy.long)

// 매수 신호를 차트에 표시 (선택 사항)
plotshape(series=buySignal, location=location.belowbar, color=color.green, style=shap
```

# 1-2) Automated Trading using API with ChatGPT

API와 Coding에 대한 접근이 엄청 쉽게 되었다. 자동매매, AI 라는 거인의 어깨를 잘 활용하면 누구나 할 수 있다.

**a) 코딩에 대한 부담감을 버려라. API나 Coding은 ChatGPT or Gemini에 任せ!**

- Coding with AI Chatbot 코드 작성은 외주(AI에게 시키거나 coder에게 시키거나)를 준다. Coder가 되는 것을 목표로 하는게 아니라 **Programmer as a Non-coder 가 되는 것을 지향하자!**

- Learning how to read a code 코드 작성을 하기 전에 ChatGPT나 타인이 만든 code 를 읽는 법부터 배우자! 코드를 작성할 지는 몰라도. 읽을 줄 알고 일부 코드는 수정할 수 있는 level까지를 목표로 하자.

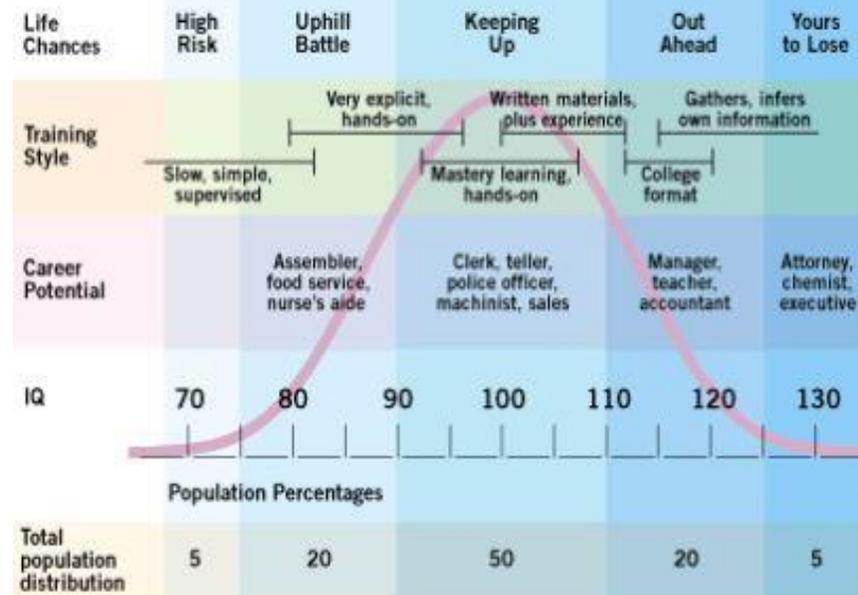
**a) API & Coding 어려움 극복하기=>자꾸 반복해서 설명을 듣고, 자주 실행을 해본다**

- API, Coding, Deep Learning에 대한 설명이 어렵다 : 보통 어렵다고 하면 낯설어서 어렵다. 처음 접하는 분야이기에 당연히 낯설어서 어렵게 느껴진다.

- 추상적인 개념이 많다. 인간은 원래 구체적이지 않은 Abstract Concept을 이해하기 어려워 한다(수학, 이론 물리학, 형이상학 등). **가급적 구체적인 사례로 접근하길!**

- 그래도 어렵고(including English), 이해가 안된다면 일찍 포기하는 것도 현명한 방법 (IQ의 한계, IQ가 130 이상이어야 고도의 추상적 추론 가능, IQ란 배우는 속도를 의미)

**- But, At least IQ가 표준편차 범위 내라면, API 자동매매 할 수 있다! With ChatGPT**



# 2) What API is & Differences between HTS API and Rest API

Advantage : Easy Access(API, coding), Emotionless trading, Back-testing Capability, Time-saving, Consistency, 24/7 Trading( 특히 암호화폐 시장)

## a) What is API?

API(Application Programming Interface)는 복수의 Computer 간에 또는 복수의 프로그램(App) 간에 통신(Network)을 통해서 서로 데이터를 주고 받는 규약, 행위 등을 일컫는 말(Inter-face)

원래 API란 (Application Programming Interface) 복수의 APP(프로그램)이 서로 통신으로 연결이 되어서 데이터를 주고 받는 것을 말한다.

따라서 증권사나 암호화폐 거래소 API라고 하면, '고객이 직접 만든 자동매매 프로그램'이라는 APP와 '시세 데이터를 요청하면 주고, 주문을 하면 처리해주는 서버'가 서로 연결되어서 시세, 주문 등의 데이터를 주고 받는 것을 말한다.

## b) Exchange API 에 대한 개념 이해 필수

API 서비스는 크게 대부분의 국내 증권사에서 제공하는 HTS 기반 API 서비스와 해외 증권사나 암호화폐 거래소(Ubit, Binance 등)에서 제공하는 Web 기반 REST API 서비스가 있다.

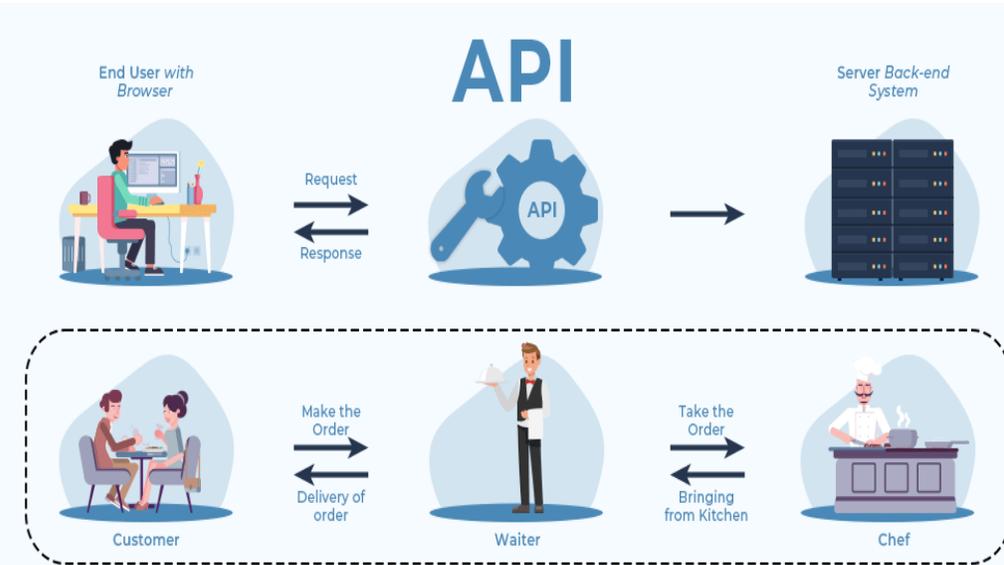
해외는 국내외 달리 2000년 이후 HTS 기반이 아니라 Web trading 기반으로 발전됨. 따라서 REST API 서비스가 보편적

두 API 방식 모두 이해는 하고 있어야 하나, 굳이 이 중 하나만 선택하자면 Rest API만 사용하자.

### ※ 시스템 트레이딩(System Trading)

자신의 자의적 판단이나 편견을 배제하고 일정한 매매규칙을 사용해 일관성 있게 매매를 수행함으로써 투자수익률을 높이는 매매방법 (출처 : 시사상식사전)

매체	시스템 트레이딩		
	HTS	DDE	API
방법	자동주문 및 수식관리자 화면 등	엑셀 DDE	사용자 작성 프로그램
특징	HTS내의 화면으로 제공되어 사용하기 편함	실시간 데이터를 이용하여 자기만의 매매 신호 발생	HTS와 동일한 데이터 이용하여 나만의 HTS를 작성할 수 있음
한계	나만의 매매 조건을 설정하는 데 한계가 있음	실시간 시세 데이터만 사용 가능	사용자가 직접 프로그램을 작성해야 함
난이도	쉬움	보통	어려움



# 2-1) Exchange의 API 구조의 이해(Component API와 Rest API)

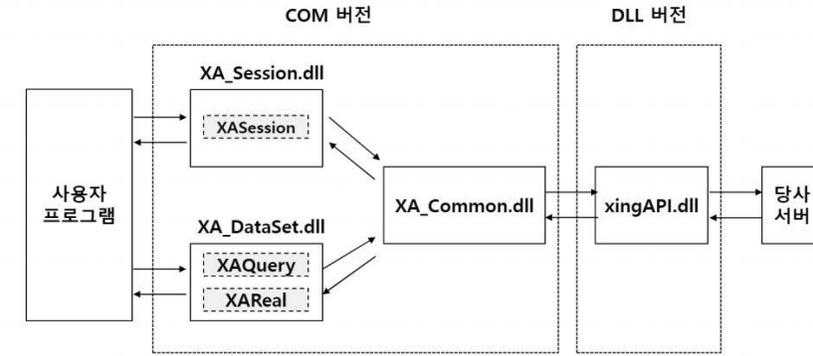
거래소 API를 이용한 Automated Trading이란 Stocks, Futures & Options, Cryptocurrency 등 다양한 Asset(Products)의 Trading을 자동화 시스템으로 만드는 기술

## a) Component API(HTS-based API) vs. Rest API(Web-based API)

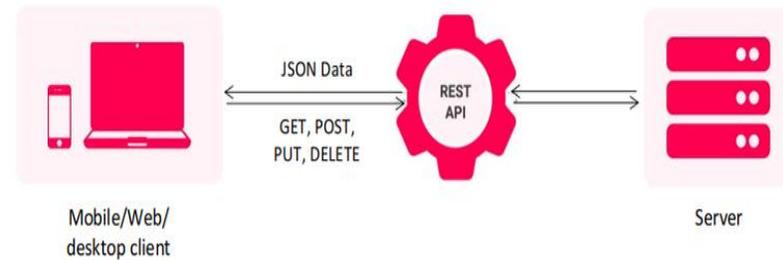
- 국내 증권사: Windows 기반에서 작동되며, API 통신 규약(함수)을 담은 파일(Dll, OCX 등)을 설치하고, XML형식의 데이터(시세, 주문 등)를 TCP/IP 형태의 네트워크로 Client(HTS)와 Server가 로그인, 시세, 주문, 잔고 등의 Data를 주고 받는 방식임. 1997년 대신증권 사이보스가 HTS 원조, 이후 대부분의 증권사가 HTS 기반의 트레이딩 Platform을 제공하면서, HTS 기반의 API Service 제공 중.

- 해외 증권사와 암호화폐 거래소 : Rest API 라는 개념은 2000년 등장했고, 2000년 중반부터 대중화면서 웹서비스와 APP간의 통신 수단으로 널리 채택. HTTP 및 Websocket 이라는 두가지 Network protocol을 이용해서 JSON 형태의 데이터를 주고 받는 형태의 API(API 이용을 위해서 별도의 파일 DLL, OCX, RES 파일 등 설치 불필요). 상대적으로 쉬운 API 코딩과 거래소간 공통 Lib 이용으로 API의 대세가 됨. Rest API(Web API)라고 함.

※ COM 구성도



## REST API Model

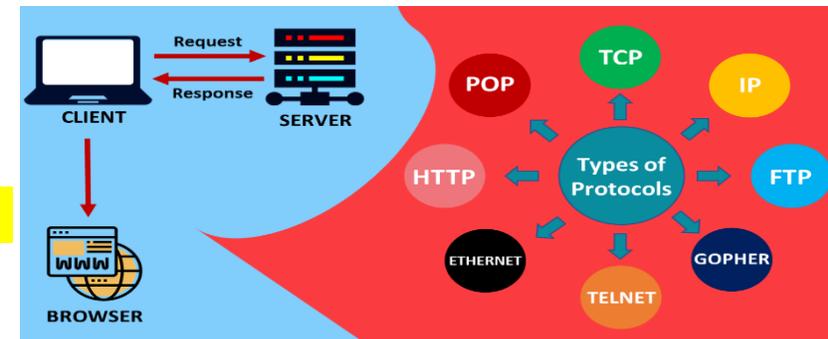


## b) Rest API is recommended.

- 국내 증권사 API는 DLL 파일을 이용해야 해서 원래 C++만 가능했지만, 2010년 이후 DLL파일을 어떤 프로그래밍 언어에서도 접근할 수 있도록 OCX 파일로 가공해서 제공. 그러나 공인인증 등 관련 파일이 32bit 기반이라서, 해당 파일을 이용하려면 32Bit 기반 프로그래밍 언어를 사용해야 함. 이 경우 엑셀(64bit), Python(64bit)과는 충돌 및 error 발생

- 코딩면에서도 증권사 방식 API는 DLL(OCX)내 API 관련 내장 함수를 이용해서 코딩해야 하며, 이 함수는 또 증권사마다 다 달라서 호환성 없음. 코딩도 복잡하고 오타로 인한 Error로 많고, 호환성도 떨어지고, 총체적으로 문제점이 많음!

- RestAPI는 OS 독립성, 개발언어 독립성, HTTP method(CRUD) 이용, 코딩이 상대적으로 쉬움, 거래소마다 공통의 Library 사용 가능(Python의 경우 CCXT lib은 업비트부터 바이낸스 등 100개 거래소에서 사용됨)



# 2-2) TR and RTD Approach in HTS-Based API vs. Web-based API

<https://api.upbit.com/v1/ticker?markets=KRW-BTC> (TR 조회, 현재가 조회, Endpoint, 데이터에 접근하는 경로) / <wss://api.upbit.com/websocket/v1> (실시간 시세)

오픈 Trading API vs eFriend expert		
주요 평가 항목	(신) Open API	(구) HTS기반 API (eFriend expert)
서비스	한국투자증권(증권사 유일)	증권사 다수(한국투자증권 포함)
사용 가능 환경(OS)	제한 없음	Windows, 스마트폰
구동 방식	Rest API(request), websocket(실시간)	Com, ocx, dll
인증 방식	대체 인증 토큰(Oauth 2.0)	HTS 접속 인증
서비스 대상 (당시 계좌보유 고객)	개인고객 계후사 (계후사 앱 연동)	HTS 접속 회원고객
연결상태(세션)	Stateless(무상태) 세션 정보가 없어 구현 단순 단, websocket은 세션 방식	세션 연결 상태이어야 사용가능

## <TR와 Real Time Data의 처리 방식의 차이>

### a) HTS 기반 API\_Platform 종속적, 호환성 없음

클라이언트가 **OCX(Dll)** 파일을 다운로드해서 이를 이용하여 **API**를 사용하는 경우  
여기서 **“TR”** 은 거래(**Transaction**) 또는 특정 요청(**Request**)을 의미하며, **“real time data”**은 실시간으로 처리되어야 하는 실시간 데이터를 말함

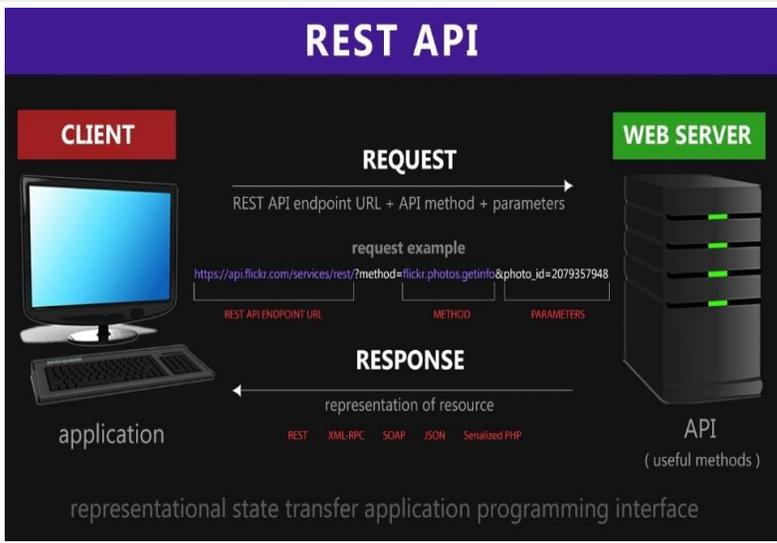
Tr과 Real은 **DLL(OCX)**파일에서 처리

### b) Web 기반 API\_Platform 독립성, 호환성 높음

**TR**과 **Real time data** 모두 **Get** 하나로 조회(**Real time data** 제공 안하는 곳도 있음, 이 경우 **TR**를 폴링하는 방식으로 실시간처럼 보이게 처리함)

**HTTP (Hypertext Transfer Protocol):** 웹 서버와 클라이언트 간의 문서 교환을 위한 프로토콜, **조회성 TR**를 처리

**WebSocket(Network Protocol),**은 웹 서버와 클라이언트 간에 양방향 통신과 지속적인 연결을 유지하는 기술. 이를 통해 데이터를 신속하게 교환할 수 있어, 채팅, 온라인 게임, 등에서 널리 사용. **실시간 데이터를 처리**



# 3) Terms you need to know when using API

<https://api.upbit.com/v1/ticker?markets=KRW-BTC> (TR 조회, 현재가 조회, Endpoint, 데이터에 접근하는 경로) <wss://api.upbit.com/websocket/v1> (실시간 시세)

## a) Tr(Transaction, 조회성 TR) vs. RTD(Real Time Data, 실시간 TR)

API에서 필수 핵심 개념으로, 일회성으로 Client가 Server에 한번 Data Request(요청)하면, Server는 한번 Data Response(응답) 해준다. 이것을 Tr이라고 함 : 현재가 시세 조회

반면에, 한번 요청하면 계속해서 데이터를 보내주어야 하는 경우가 있다. 이런 데이터를 Real Data라고 부른다. 이걸 Request라고 하지 않고 Subscribe(Advise)라고 함 : 실시간 시세 등록

## b) HTS 기반 API에서의 TR 호출 방식\_Inblock(input) vs OutBlock(output)

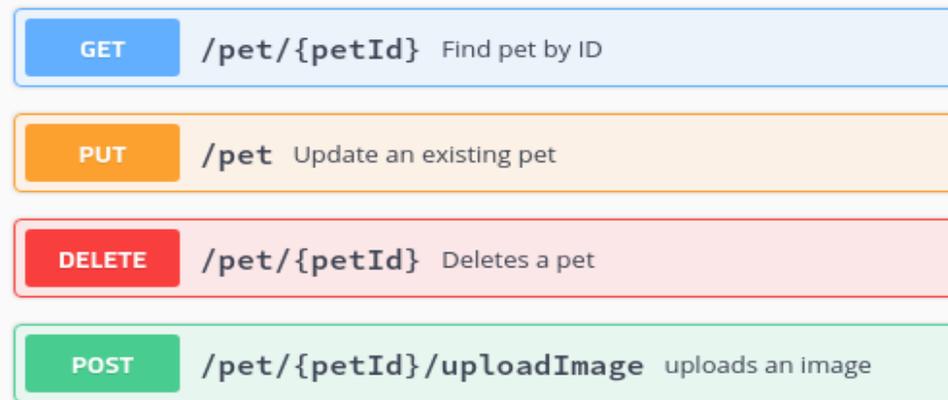
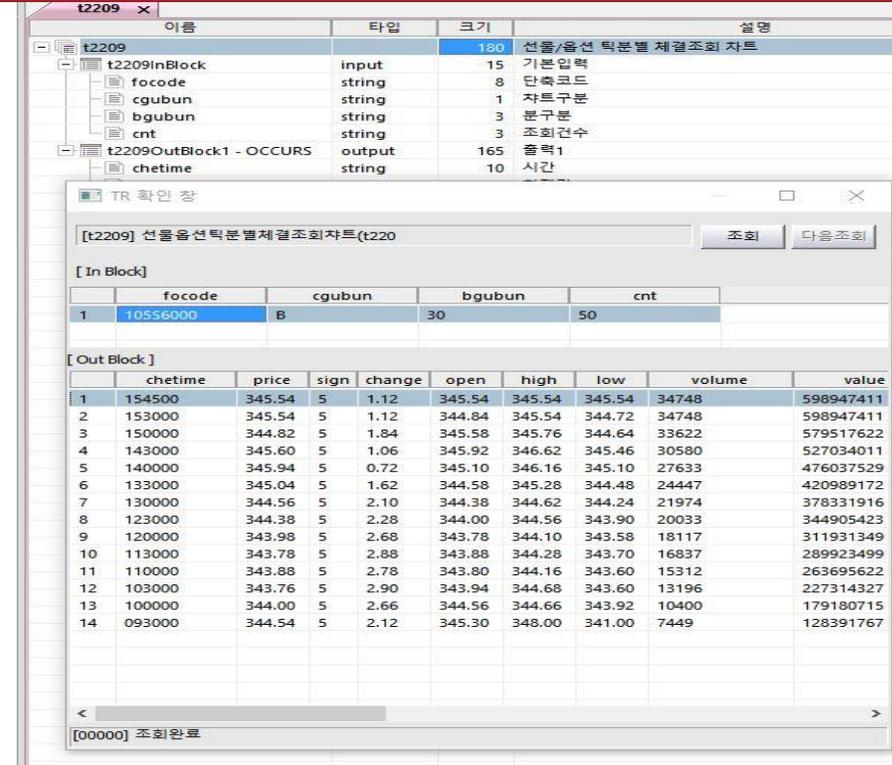
선물현재가, 주식현재가, 옵션현재가 라는 각각 데이터를 조회하고자 하면 각각의 TR/RTD 번호가 있으며(오른쪽 t2209 처럼) 해당 TR번호에 Inblock에 Parameter를 채워서(선물 3월물을 매개변수(파라미터) 해당 코드를 넣어서 request 하면, Outblock 에 요청한 선물 3월물 시세가 response 된다. 그리고 Outblock까지 코딩을 해줘야 한다.(상당히 귀찮고 오류가 많음)

c) 인증 방법 : HTS 기반 API 에서는 아이디, 비밀번호, 공인인증번호, Rest API를 사용하기 위한 인증 절차와 필요한 키(예: API Access Key, Secret Key Key)

d) HTTP Methods: REST API의 경우 요청시 사용해야 하는 HTTP 메소드(GET, POST, PUT, DELETE)를 이용해서 데이터 조회, 등록, 변경, 삭제한다. The type of request being made, such as GET (retrieve data), POST (send data), PUT (update data), or DELETE (remove data)

e) 엔드포인트 : Endpoint는 HTS 기반 API의 TR 번호와 같은 역할. Rest API 에서는 사용 가능한 API 엔드포인트(URL 경로)와 각 엔드포인트가 수행하는 작업에 대한 설명이 제공됨(Endpoint: The URL where your API can be accessed by a client application.)

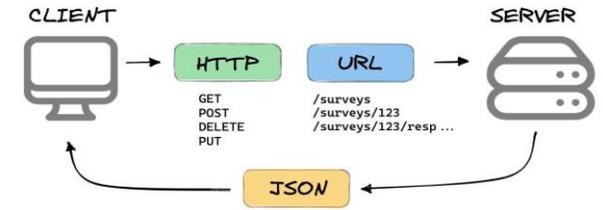
f) RestAPI\_Status Codes: Numbers that indicate the outcome of the HTTP request, such as 200 OK for success or 404 Not Found for an error.



# 3-1) 증권사 vs. Upbit의 시세 조회 API Coding 비교

<시세 조회(Tr request) API Coding 비교\_Python Coding>

WHAT IS A REST API?



a) 증권사 API 방식의 조회\_선물 현재가

```
import win32com.client <- 32비트 OCX 파일을 사용하기 위한 library 호출
import pythoncom <- Python에서 Com 방식을 사용을 위한 library 호출
```

Session = win32com.client.DispatchWithEvents("XA\_Session.XASession", login) -> 접속/로그인  
관련 기능을 담당하는 OCX 파일 내 로그인/서버접속 해주는 내장함수 XA\_Session.기능을 변수 선언

```
아이디 = "" # 내 아이디
비밀번호 = "" # 내 비밀번호
공인인증번호 = "" # 내 공인인증번호
```

```
session.ConnectServer("demo.ebestsec.co.kr", 20001) -서버접속을 하게 하는 API 내장함수
session.Login(아이디, 비밀번호, 공인인증번호, 0, 0) -> 로그인을 하게 하는 API 내장함수
```

# t2101 (선물 현재가 시세 조회 TR), HTS 화면번호와 유사하다

```
t2101 = win32com.client.DispatchWithEvents("XA_DataSet.XAQuery", 선물현재가) -> TR
데이터를 주고 받는 OCX 파일 이름
```

```
t2101.ResFileName = "C:\\eBEST\\xingAPI\\Res\\t2101.res" -> 데이터가 xml 형태
```

```
t2101.SetFieldData("t2101InBlock", "focode", 0, " ") # 종목코드 -> 요청 형식 inblock,
set으로 요청
```

Class 선물현재가: 응답 형식이 outblock, get으로 가져온다.

```
def OnReceiveData(self, code):
    선물현재가.수신여부 = True # 현재가 수신을 했다면 화면에 출력
```

```
print("종목명 : " + t2101.GetFieldData("t2101OutBlock", "hname", 0))
print("현재가 : " + t2101.GetFieldData("t2101OutBlock", "price", 0))
print("시가 : " + t2101.GetFieldData("t2101OutBlock", "open", 0))
print("고가 : " + t2101.GetFieldData("t2101OutBlock", "high", 0))
print("저가 : " + t2101.GetFieldData("t2101OutBlock", "low", 0))
```

b) REST API 방식의 조회\_BTC 시세 조회(Upbit)

Requests 는 웹서버에 요청, API에 호출, Http 프로토콜 작업 위한 Lib(Library)

```
C: > Users > COMZO > Desktop > getbitcoinpriceinupbit.py > ...
1 import requests
2
3 # access_key = os.environ.get('upbit_access_key')
4 # secret_key = os.environ.get('upbit_secret_key')
5 # upbit = pyupbit.Upbit(access_key, secret_key)
6
7 def get_bitcoin_price():
8     url = "https://api.upbit.com/v1/ticker?markets=KRW-BTC"
9     response = requests.get(url)
10    price_info = response.json()
11    return price_info[0]['trade_price']
12
13 if __name__ == "__main__":
14    price = get_bitcoin_price()
15    print(f"Current Bitcoin Price: {price} KRW")
```

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS

Windows PowerShell  
Copyright (C) Microsoft Corporation. All rights reserved.

새로운 크로스 플랫폼 PowerShell 사용 <https://aka.ms/pscore6>

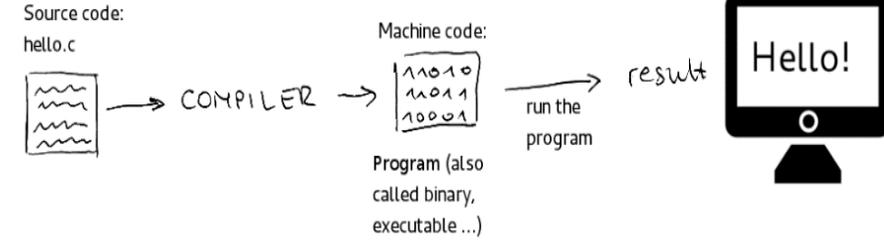
```
PS C:\Users\COMZO\Desktop> & 'c:\Users\COMZO\anaconda3\python.exe' 'c:\Use
Current Bitcoin Price: 88329000.0 KRW
PS C:\Users\COMZO\Desktop> []
```

# 4) Understanding of Coding and Programming

Coding is the conversation with a Computer / 컴퓨터가 특정한 작업을 수행하도록 지시하는 것 = 코딩  
Programming Language란 인간이 이해할 수 있는 언어(English), 따라서 Computer가 이해할 수 있도록 다시 번역/해석의 과정이 필요함

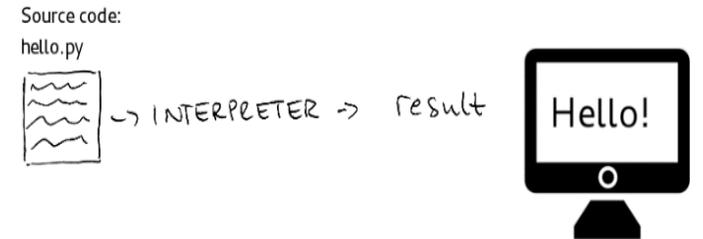
## a) What is coding? Programming Language vs. Machine Language

코딩이란 **Computer**에게 일을 시키는 것으로, 컴퓨터에게 일을 시키려면 컴퓨터가 이해할 수 있는 언어(**Machine Language**)로 명령을 내려야 한다. 그러나 컴퓨터가 이해하는 언어는 **2진법(Binary)**이기에 인간은 이해할 수 없다. 따라서 인간이 이해할 수 있는 언어(**Programming Language**)로 일단 코드를 작성 한 후 **Machine Language**(기계어)로 변형시켜야 줘야 한다.



## b) Coding vs. Programming

**Coding**은 기본적으로 인간이 이해할 수 있는 언어(**Programming Language**)를 이용해서 **Computer**에게 명령을 내리는 일, **Programming**은 **Coding**을 포함해서 하나의 **Project**를 완성하는 일.

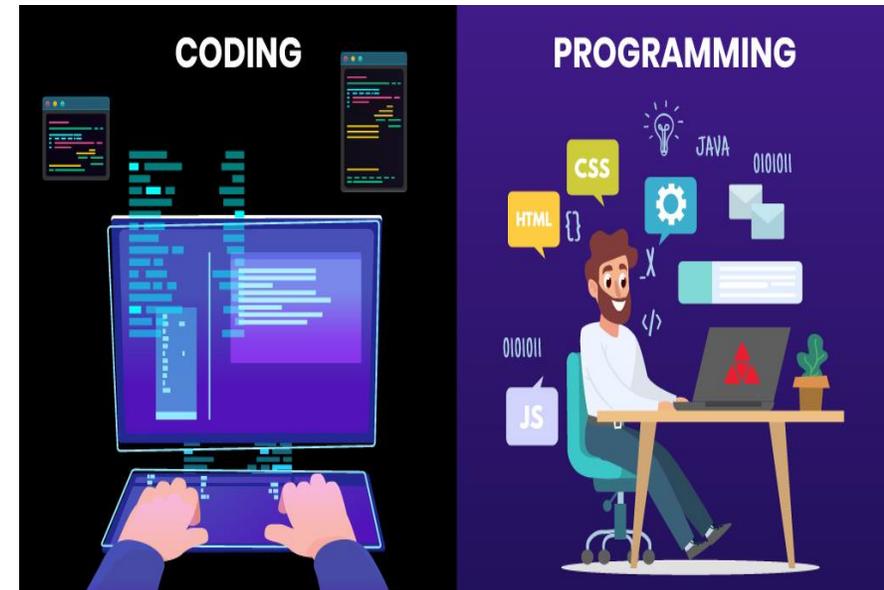


**Coding**이란 코드를 작성하는 것. **Programming**은 프로그램을 만들과 관리하는 일련의 **Process**를 모두 하는 것(기획->설계->**Coding**->Implementation->Test->Debugging->Review->Maintenance)

코딩 자체는 **ChatGPT**나 타인에게 외주를 주자! **We become a Programmer as a non-coder!**

## c) Code는 어떻게 Computer(CPU와 RAM)에게 일을 시키는가?

코드 작성 -> 기계(즉 컴퓨터)가 이해할 수 있게 **Compile**(C#의 경우) 또는 **Interpret** (Python의 경우) 해야 한다 -> 기계어로 변환된 코드는 **CPU**에서 실행 -> **CPU**에서 메모리 관리, 데이터 처리, 입출력 관리 -> **OS**(운영체제, 대표적 운영체제는 **Windows, macOS, Linux**, 컴퓨터의 여러 **hard ware**를 관리, 다른 프로그램을 실행할 수 있도록 해주는 **Platform** 역할, **OS가 없으면 파일 접근도 불가, 응용 프로그램 실행도 불가**) 위에서 프로그램 작동(네트워크, 키보드와 마우스, 모니터를 이용가능)



# 4-1) Selection Program language as a non-Programmer

Python, C#, Javascript 중에서 Python을 추천! 단순한 App 개발부터 Web page 제작, 데이터 분석부터 인공지능(AI) 까지 Must-Learn Skill

## a) Natural Language vs. Machine Language

자연어\_안녕하세요? How are you doing? Vs. 기계어\_01001001111

자연어(English)와 기계어(Machine language) 사이에서 인간과 컴퓨터가 대화할 수 있는 **Programming Language**가 탄생. 다양한 프로그래밍 언어가 개발되어져 왔다(C++은 1970년대, Python은 1991년 출시, C#은 2000년에 MS에서 출시)

## b) API를 이용한 Trading Program(Application)을 만드는 데 가장 적합한 프로그래밍 랭귀지는?

- **Python** \_ 가장 대중적으로 사용되는개발언어, **Finance and Trading Community**에서 가장 널리 사용됨. **Data Analysis, Deep learning, Back Testing**에 필요한 다양한 **Library**가 풍부함. 단점은 국내 증권사의 **component API** 방식과 안 맞는다. **장점은 Non-programmer such as Traders** 도 가장 쉽게 배울 수 있고, **RestAPI**와 암호화폐 거래소를 이용할 때 **최대 장점 발휘 (32비트 ocx 파일을 이용해 API 사용해야 하는 경우, 32비트 Python 인터프리터를 설치)**

- **C#** \_ MS의 .net 플랫폼을 이용해서 프로그램 개발, **Windows OS** 체제에서 잘 작동되는 프로그램 개발에 용이. 단점은 금융이나 트레이딩 관련 **Library**가 부족. 장점은 국내 증권사의 **API**를 이용하는데 무리가 없고, 화면(**UI**) 등을 만드는 데 편리하다.

- **Javascript** \_ 웹페이지 개발할 때 사용(**Html, Css, Javascript** 3개 언어로 웹페이지를 만듦), 따라서 웹 기반 트레이딩 프로그램을 만들 때 사용함

- **C++** \_ 국내 증권사 **HTS**는 다 **C++**로 개발됨. **C++**은 고성능의 금융 거래 시스템 개발에 유리, 컴파일 언어라서 실행 속도가 빠름(**DMA** 를 이용한 **Arbitrage trading** 등은 **C++**로 프로그램 개발). **단점은 비프로그래머가 접근하기 어려운 프로그램 랭귀지, [White House urges developers to dump C and C++\(2024\)](#)**

## c) 영어를 잘해야 Coding을 잘 할 수 있다(Python = English). English is another Must-learn Skill for us such as a individual investor and trader

Python was written with the goal of "**Computer Programming for Everybody.**" Many of us have come to learn and love the intuitive nature of this language; however, that "everybody" should have an asterisk because Python's readability is really only specific to **English sentences.**

# 4-2) How to Read a code written by Someone or AI

<https://www.programiz.com/python-programming/online-compiler/> (Python Online Compiler)

## a) Coding Basics

쉬운 개발언어로 시작 -> 각 개발언어마다 문법과 명령어가 다르지만, 기본적으로 명령어 중에서 **Variable(변수), Operators(연산자), IF statement(조건문), Loop(반복문), Function 또는 Method(함수)**는 거의 비슷한 문법과, 기능, 구조를 가지고 있다.

```
현재가 = 50
시가 = 3
if 현재가 > 시가:
    print("매수해라")
else:
    print("매도해라")
```

매수해라

## b) Python 기본문법 이해하기

Python에서는 변수를 통해 데이터를 메모리에 저장, 다양한 데이터 자료형(문자, 정수 등) 함수(Function)은 **def** 키워드로 정의(함수를 사용하면 코드의 재사용 가능)  
Python은 **Object Oriented Programming Language** 이기에 **Object**를 만들기 위해 **Class, Instance** 를 사용함(**Def, Class, instance** 를 사용하면 코드의 가독성이 높고, 공동 코딩에 유리)  
**#** 뒤에 내용은 실행되는게 아니라 주석이라고 해서, 코드를 설명해 줌  
**Import xxxx** 라는 건 라이브러리, 패키지, 파일, 모듈을 불러올 때 사용함

## c) 코드 분석 연습

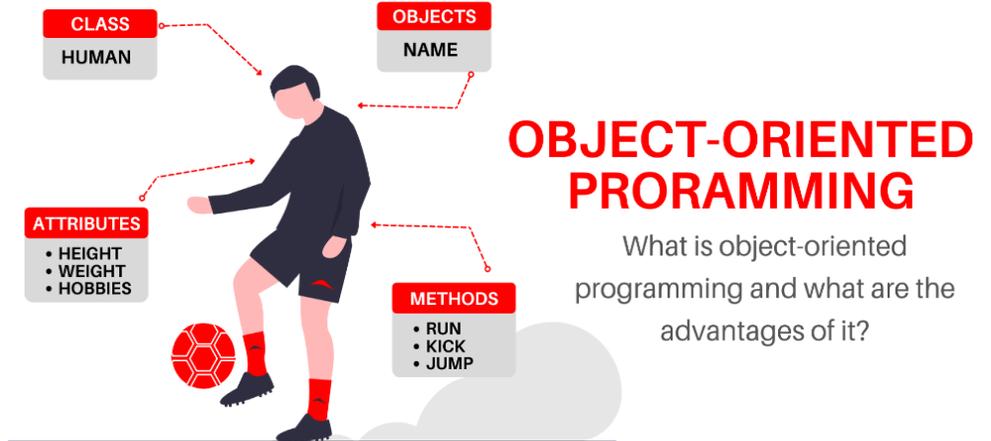
Github 와 같은 Platform에 올려져 있는 **Open source code**를 찾아서 **ChatGPT** 등에게 코드 분석을 의뢰

- 쉬운 코딩 언어로 기본만 익히고 시작하기
- 변수, 조건문, 반복문, 함수만 알아도 웬만한 건 만들 수 있음
  1. 변수 : ~는 ~다의 개념 ex) ID = 'jocoding'
  2. 조건문 : 만약 ~라면 ex) ID/PW가 일치하면 로그인 성공표시
  3. 반복문 : ~를 n번 반복해라 ex) 앞으로 이동 10번 반복
  4. 함수 : a를 넣으면 b가 나옴 ex) 공격력이 50이면 데미지가 50

<객체 지향 코딩(OOP)을 하는 이유>

모듈성, 재사용성, 유지보수성, 가독성 -> 타인도 이 **code**를 쉽게 이해하고 수정하고 관리할 수 있어야! API 함수도 **OOP**로 되어 있음

초보자나 혼자 사용하는 프로그램이라면 아무렇게나 코딩을 해도 문제될 건 없다. **If it works, No problems!**



# 5) Most Popular Python Lib used in Automated Trading.

Install library : CMD -> 관리자권한 실행 -> pip install requests / Import libraryname in code

## 1) HTS-based API

a) **pythoncom**과 **pyWin32** : Windows의 COM(Component Object Model) 기술을 파이썬에서 사용할 수 있도록 해주는 라이브러리, 파이썬에서 **OCX** 컨트롤을 직접 로드하고 조작할 수 있다. 증권사 **API**의 이벤트 핸들링, 데이터 요청 및 수신 등을 구현할 때 필수적

b) **PyQt** 또는 **PySide** : **QAxContainer** 모듈을 사용하면 **PyQt** 또는 **PySide**에서 **OCX** 컨트롤을 직접 호스팅하고 조작할 수 있다. 화면(UI)을 구현할 때 주로 사용.

## 2) Web-Based API

a) **Requests**: HTTP 요청(TR 요청)을 보내기 위한 라이브러리로, **RESTful API**와 상호작용할 때 필수적

b) **Websockets**: 실시간으로 암호화폐 거래소의 시장 데이터를 스트리밍할 때(**RTD** 요청) 사용되는 **Python** 라이브러리

a) **CCXT (CryptoCurrency eXchange Trading Library)**: 다양한 암호화폐 거래소의 **API**를 지원하는 가장 인기 있는 라이브러리 중 하나. 이 라이브러리를 통해 **100개** 이상의 암호화폐 거래소(from **Upbit** to **Binance**)에서 자동매매가 가능함.

## 3) Commonly used Library

d) **Pandas**: 데이터 분석을 위한 라이브러리로, 시장 데이터를 분석하고 처리하는 데 매우 유용

e) **NumPy**: 가격 계산, 차트 보조 지표 계산을 기본 패키지로, 대규모 다차원 배열과 행렬 연산에 유용하며, 고급 수학 함수도 포함

c) **ta-lib (Technical Analysis Library)**: 기술적 분석 지표들을 쉽게 계산할 수 있게 해주는 라이브러리임. 이동 평균, **RSI**, **MACD**와 같은 다양한 기술적 지표를 제공

f) **Plotly / Matplotlib**: 이 라이브러리들은 데이터 시각화에 사용되며, 시세의 가격 변동, 거래량 등을 차트로 표현하는 데 도움을 준다.

h) **Flask / Django**: 웹 프레임워크는 자동매매 봇을 웹 애플리케이션을 구축할 때 사용

```
Microsoft Windows [Version 10.0.18362.592]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\Wikhil>pip install librosa
Collecting librosa
  Using cached https://files.pythonhosted.org/packages/77/b5/1817862d64a7c231afd15419d8418ae1f000742cac275e85c74b219cbcb/b/librosa-0.7.2.tar.gz
Collecting audioread=>2.0.0
  Using cached https://files.pythonhosted.org/packages/2e/0b/940ea7861e0e9049f09cdf72a90c9ae55f697c17c299a323f0148f9130/2/audioread-2.1.8.tar.gz
Collecting numpy=>1.15.0
  Downloading https://files.pythonhosted.org/packages/53/74/b997e4c7b4abc68e99f4c3dba87ee2c6f7559319af756cc1ede37665a80/numpy-1.18.1-cp36-cp36m-win_amd64.whl (12.8MB)
  |#####| 12.8MB 126kB/s
Collecting scipy=>1.0.0
  Downloading https://files.pythonhosted.org/packages/8d/2f/fcb6150813b89d628749784370132e431f687ebab5a1063eb298cc941f70/scipy-1.4.1-cp36-cp36m-win_amd64.whl (30.8MB)
  |#####| 26.4MB 79kB/s eta 0:00:56
```

```
import numpy as np
import matplotlib.pyplot as plt

# 가정 설정
initial_capital = 10000 # 초기 자본
annual_return_rate = 0.10 # 연간 수익률 10%
years = 30 # 투자 기간 30년

# 시간 배열 생성
years_array = np.arange(0, years + 1, 1)

# 수익률 계산 (복리 계산)
final_capital = initial_capital * (1 + annual_return_rate) ** years_array

# 그래프 생성
plt.figure(figsize=(10, 6))
plt.plot(years_array, final_capital, label='Investment Growth', color='blue')
plt.title('Long-Term Returns with a Winning Edge Algorithm')
plt.xlabel('Years')
plt.ylabel('Capital ($)')
plt.legend()
plt.grid(True)
plt.show()
```

# 5-1) Comparison between Python and C#

Form을 중심으로 한다면 C#, 빠른 개발 테스트가 필요하다면 Python /각각의 장단점이 있기에 둘 다 잘 활용할 수 있어야 한다!  
Python은 매우 가독성 높고 이해하기 쉬운 Programming Language / Python 코드를 읽는 능력은 시간과 연습을 통해 점차 향상

## a) Python (대표적인 script language, Compile 과정 없이, 코딩을 run 하면 한 줄씩 interpreting 방식으로 기계어로 변환되어 실행)

- 스크립트 랭귀지 특성상 문법이 쉬운 편. 매매전략 아이디어가 떠올랐을 때 빠르게 코딩해서 확인하는데 제격(한줄 코드 작성후 바로 실행해서 결과 확인 가능). GUI에서 form 등을 중요하게 생각하지 않고 console에서 처리하는데 제격인 언어. PyQt를 이용해서 form을 만들 수 있음.

- 그러나 스크립트 랭귀지는 컴파일드 랭귀지 대비 실행 속도가 상대적으로 느린 언어, 코드 상의 에러는 실행시 에 발견됨. 실행하면서 코드 수정 요함. 또한 공식적인 코드 작성 및 수정, 실행시키는 Editor가 따로 없음. 개발환경 설정하는 다소 어려움이 있음. **Library 설치치 관리자권한으로 실행하면 대부분 문제 해결**

- Non-Programmer에게 최대 장점은 ChatGPT에게 거의 완벽하게 Coding을 해 줌. 또한 금융, 주식, 투자 관련 참고할 Sample code and Library가 풍부. 머신러닝 등을 하는데 있어 필수 프로그램 랭귀지

\*\* 외부 Library나 py 파일 불러올 때 Python import xxxxx / C#에서는 Using xxxxx

## b) C# (대표적인 compiled language, 코딩 후 컴파일 과정을 거쳐 기계어로 변환 후 실행됨)

- Visual Studio 라는 강력한 IDE(코딩, 디버깅, 컴파일 등) 를 사용. 기본 라이브러리 지원이 잘 되어 있어 기능들을 쓰기가 편하다. 특히 Form 등을 만들어서 사용할 때는 C#을 추천함.

- Visual Studio 라는 개발툴 사용(개발툴 숙련에 시간이 필요). 코딩이 Python보다 복잡(다양한 변수 타입, 규모가 큰 개발 환경에 필요한 모듈 활용)

- 데이터 분석, 차트나 전략 구현을 지원해주는 라이브러리 부재로, 직접 코딩해야 하는 수고로움, Python은 기존에 만들어진 라이브러리를 불러와서 사용하면 그만

### Python 예시:

```
python
import math
print(math.sqrt(4))
```

### C# 예시:

```
csharp
using System;
class Program
{
    static void Main()
    {
        Console.WriteLine(Math.Sqrt(4));
    }
}
```

## Python vs. C# code 비교

```
import requests
```

```
url =
"https://api.upbit.com/v1/ticker?markets=KRW-BTC"
```

```
headers = {"accept":
"application/json"}
```

```
response =
requests.get(url,
headers=headers)
```

```
print(response.text)
```

```
using System;
using System.Net.Http;
using System.Threading.Tasks;
namespace UpbitApiConsoleApp
{
    class Program
    {
        static async Task Main(string[] args)
        {
            // HttpClient 인스턴스 생성
            using (HttpClient client = new HttpClient())
            {
                // 요청할 URL 생성
                string url = "https://api.upbit.com/v1/ticker?markets=KRW-BTC";

                // 요청 헤더 추가
                client.DefaultRequestHeaders.Accept.Clear();
                client.DefaultRequestHeaders.Accept.Add(new System.Net.Http.Headers.MediaTypeHeaderValue("application/json"));

                try
                {
                    // HTTP GET 요청 실행 및 응답 대기
                    HttpResponseMessage response = await client.GetAsync(url);
                    if (response.IsSuccessStatusCode)
                    {
                        // 응답 본문(응답체) 읽기
                        string responseBody = await response.Content.ReadAsStringAsync();
                        Console.WriteLine(responseBody);
                    }
                    else
                    {
                        Console.WriteLine($"Error: {response.StatusCode}");
                    }
                }
                catch (HttpRequestException e)
                {
                    // HTTP 요청 오류 처리
                    Console.WriteLine($"Request exception: {e.Message}");
                }
            }
        }
    }
}
```

# 6) Deep Learning, AI, ChatGPT

프롬프트 엔지니어링은 AI와 인간 사이의 소통을 최적화하는 예술과 과학/

## a) Deep Learning, AlphaGo, ChatGPT

딥 러닝이란 인간의 뇌가 학습하는 방식을 모방해서 만든 컴퓨터를 위한 학습법으로 **Big data**를 입력 후 **미세조정과 최적화, 지도학습과 강화학습** 등을 통해서 인간 두뇌의 'Learning'과 비슷한 효과를 얻는 최신 컴퓨터 기술이다.

2013년 'Dogs vs. Cats' 사진의 구분에서 성공하면서 학계에서 주목받기 시작함. 대중에게 **Deep learning** 기술이 알려지게 된 계기는 2016년 **Alphago vs. 이세돌** 대결이다. 알파고는 위의 기술로 최선의 '다음 수'를 찾는데 특화된 **Model**. 딥러닝, 지도학습, 강화학습 등의 기술이 꽃을 피운 건 바로 **ChatGPT**. 2022년 12월 세상을 놀라게 했다.

**ChatGPT**는 알파고와 비슷하게 최선의 '**Next Words or Sentence**'를 예측하는 모델(아버\*, Give me \*). 그리고 예측을 잘 하기 위해서는 문법, 맥락, 정보, 상식, 지식, 수학, 과학, 외국어, 코딩 등을 모두 알고 있어야 가장 높은 확률로 다음 단어나 구를 제시할 수 있다. 그래서 자연스럽게 척척박사가 되버린 것이다. 원래는 **Text-to-Text model** 이나 **Multimodal**로 점점 **AGI(artificial general intelligence)**로 가고 있는 중  
(If you know How ChatGPT works, You might be surprised)

## b) What is Prompt Engineering?

프롬프트 엔지니어링이란, **ChatGPT**와 같은 대화형 **AI chatbot(Multimodal AI)**에 적절한 **Prompt(입력, Input)**을 입력해서, 본인이 원하는 결과값(**Output**)을 효과적으로 도출해내는 기술(**Tech**)

## c) Comparison between Chat GPT 4 Paid version(20 dollar per month) and Free version

**ChatGPT 4 paid version(Plus version)** : 더 빠른 응답속도, 더 정확한 대답, 다양한 **ChatGPT App** 사용 가능, 대량의 데이터 분석 가능 (무료는 **ChatGPT 3-5 version** 이용?)

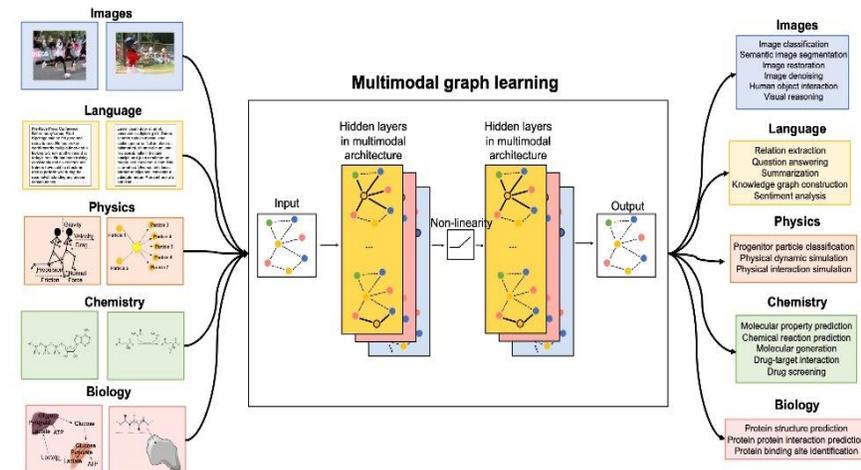
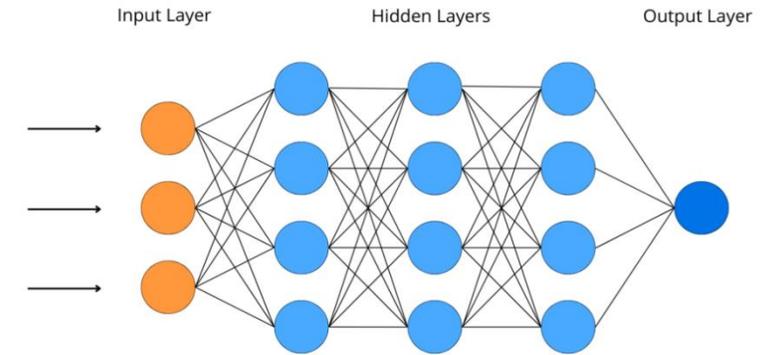
## d) ChatGPT 4 (20 dollar/month) vs. Google Gemini advanced(29000won/month)

<https://chat.openai.com/>

<https://gemini.google.com/advanced>



ChatGPT'S Neural Network Architecture



# 6-1) How to use ChatGPT for a Trader

## a) Many Ways to Use ChatGPT Effectively as a Trader

- 경제 및 금융 뉴스와 중요 시장 이벤트에 대한 정보 획득(Webpilot으로 실시간 정보 획득)
- **Trading strategy and Back testing** : 다양한 트레이딩 전략에 대해서 AI와 논의하고, 관련 정보를 수집하고, 매매 전략의 백테스팅도 진행(Data Analyst와 Python library 이용)
- 매매에 관한 조언 얻기, 문서 번역 및 논문 요약 등 깊이 있는 분석도 가능(PDF ai PDF)
- 거래 자동화 및 프로그래밍 지원, 특히 Python을 이용한 API 연동 Automated Trading Program 개발에 특화되어 있음

ChatGPT의 중요한 특징 중 하나가 '프로그래밍 언어'가 아닌 '자연어'(한국어, English)로 컴퓨터에게 일을 시킬 수 있다는 것이다. 코딩을 전혀 몰라도 원하는 APP을 만들 수 있는 등 놀라운 일을 가능케 해주는 세상이 왔다.

누구나 Prompt Engineer가 되어서, ChatGPT(OpenAI) 나 Gemini(Google) 등 AI Chat bot을 Your Smart Assistant로 잘 활용한다면 비약적인 생산성 증가로 이어진다.

즉, AI를 잘 활용하는 사람(기업)과 그렇지 못한 사람(기업)의 격차는 점점 더 비교불가능 정도로 커질 것이다.

## b) Making a Strategy or Algorithm using Data Analyst(ChatGPT)

<https://kr.investing.com/crypto/bitcoin/historical-data>

비트코인 일별시세를 CSV로 Download 받아서 매매전략 수익 Back-Testing

## c) Coding from API document using ChatGPT

'API 관련 Reference를 ChatGPT에 Upload 하고, 해당 문서의 API 함수를 이용해서 자동매매하는 Coding을 요청하기

## d) Ask about '파생인의쉽터와 협의수수료 ' to ChatGPT

Check the response how the Chat know about 파생인의쉽터와 협의수수료

**You**  
워노피라는 암호화폐 트레이더가 암호화폐 매매로 3천억을 벌었다고 그의 매매법을 인터넷에서 최대한 많이 수집해서 설명해주세요

**WebPilot**

Talked to gpts.webpilot.ai

워노피는 한국 코인 시장에서 주목받는 암호화폐 트레이더로, 비트코인 마진거래 및 선물거래를 주로 하며 큰 수익을 올린 것으로 유명합니다. 그의 매매법은 많은 투자자들에게 관심의 대상이 되었습니다. 워노피가 공개한 주요 매매법을 요약하면 다음과 같습니다:

**You**

비트코인 과거 데이터.csv  
Spreadsheet

증가가 5일 이동평균선 위에 있을 때 매수하고, 증가가 5일 이동평균선 아래에 있을 때 매도하는 매매 전략을 사용하고자 한다. 위 비트코인의 시세 데이터를 가지고 본 매매 전략의 수익과 손실을 분석해주세요

**Data Analyst**



위 그래프는 비트코인 시세 데이터에 기반한 매매 전략의 누적 수익률과 시장의 누적 수익률을 비교한 것입니다. 파란색 선은 시장의 누적 수익률을, 녹색 선은 전략에 따른 누적 수익률을 나타냅니다. 그래프에서 볼 수 있듯이, 전략의 수익률이 시장 수익률에 비해 안정적으로 유지되며, 특정 구간에서는 시장 대비 더 나은 성과를 보여줍니다. 이러한 분석은 투자 전략의 효율성을 평가하는 데 도움이 될 수 있습니다. [-]

# 6-2) ChatGPT as a Coding Assistant (1)

Write code using ChatGPT(코딩을 시킬 때는, 구체적으로 기술하라! 예를 들어 어떤 librar를 이용해달라 or 구체적 매매 전략 지시)

**Let the ChatGPT write a code for you :** ChatGPT에게 코딩을 시켜라. 일단 Python이나 C#으로 코딩을 시킨 후 VS code (IDE for Python) or Visual Studio 2022 (IDE for C#) 등에서 실행시켜 본다. Error가 발생하면 Error 부분을 ChatGPT에게 복사해서 문의하면서 수정을 계속 한다.

**You**  
pyupbit library를 이용해서 비트코인 자동매매 app을 만들려고 한다. programming language는 python으로 하고, 매매전략은 현재가가 직전 고가보다 높으면 buy, 직전 저가보다 낮으면 sell하는 전략이다. 최대한 많은 주석을 붙여서 코딩해주세요

**ChatGPT**  
아래는 `pyupbit` 라이브러리를 이용하여 비트코인 자동 매매 앱을 만드는 기본적인 Python 코드 예시입니다. 이 예제는 지정된 매매 전략(현재가가 직전 고가보다 높으면 매수, 직전 저가보다 낮으면 매도)을 따릅니다. 실제 투자 전략에 맞게 조정하고, API 키를 안전하게 관리하십시오.

python Copy code

```
import pyupbit
import time

# Upbit API Key를 설정합니다. (안전한 저장 방법을 사용하세요)
access_key = "your_access_key"
secret_key = "your_secret_key"

# Upbit 객체를 생성합니다.
upbit = pyupbit.Upbit(access_key, secret_key)

# 사용할 티커와 기준 시간 간격을 설정합니다.
ticker = "KRW-BTC" # 비트코인 대한 원화 시장
interval = "minute60" # 60분봉 데이터를 사용

def get_target_price():
    """
    직전 고가와 저가를 기준으로 매수 및 매도 목표 가격을 계산합니다.
    """
    # 최근 60분봉 데이터를 가져옵니다.
    df = pyupbit.get_ohlcv(ticker, interval=interval, count=2)

    # 직전 봉의 고가와 저가를 가져옵니다.
    last_high = df['high'][-2]
    last_low = df['low'][-2]

    return last_high, last_low
```

```
def trade():
    """
    매매 전략에 따라 매수 또는 매도를 수행합니다.
    """
    last_high, last_low = get_target_price()
    current_price = pyupbit.get_current_price(ticker)

    # 현재 가격이 직전 고가보다 높으면 매수
    if current_price > last_high:
        # 잔고 조회
        krw_balance = upbit.get_balance("KRW")
        if krw_balance > 5000: # 최소 거래 금액 확인
            # 전체 잔고로 매수
            upbit.buy_market_order(ticker, krw_balance)
            print("매수 실행:", current_price)

    # 현재 가격이 직전 저가보다 낮으면 매도
    elif current_price < last_low:
        # 해당 코인의 잔고 조회
        btc_balance = upbit.get_balance(ticker)
        if btc_balance > 0.00008: # 최소 거래 단위 확인
            # 전체 잔고로 매도
            upbit.sell_market_order(ticker, btc_balance)
            print("매도 실행:", current_price)

if __name__ == "__main__":
    while True:
        try:
            trade()
        except Exception as e:
            print("Error:", e)
        time.sleep(60) # 60초 마다 반복 실행
```

주의사항 및 안내:

- 이 스크립트는 실제 거래에 앞서 테스트 목적으로만 사용해야 합니다.
- `your\_access\_key`와 `your\_secret\_key`는 본인의 Upbit API 키로 대체해야 합니다. API 키는 매우 민감한 정보이므로 안전하게 관리해야 합니다.
- 실제 투자 시, 시장 상황, 수수료, 슬리피지(↓ 가격과 체결 가격의 차이) 등 여러 요인을 고려해야 합니다.

# 6-3) ChatGPT as a Coding Assistant (2)

Code analysis using ChatGPT(Text가 아니라 Image나 URL로도 코드 분석 및 설명 가능)

For instance : DB API.NET용 nuget Package 를 Open Source로 Github에 올렸다는 카페 글 참고 <https://cafe.naver.com/fo24/34158>

Click the link below for visiting the Github <https://github.com/teranum/DBHappyPlus>

**You**  
아래 URL에 나온 코드 및 라이브러리를 최대한 쉽게 이해할 수 있도록 설명해주세요.  
<https://github.com/teranum/DBHappyPlus>

**WebPilot**

Talked to gpts.webpilot.ai

DBHappyPlus는 DB금융투자의 OpenApi를 C# 언어로 래핑한 프로젝트입니다. 이 프로젝트를 통해 DB금융투자의 OpenApi를 C# 애플리케이션에서 쉽게 사용할 수 있게 해줍니다. 프로젝트는 GitHub에서 오픈 소스로 제공되며, MIT 라이선스 하에 배포됩니다.

프로젝트의 구조는 다음과 같습니다.

- **DBCommAgent.NET**: DB금융투자의 통신 에이전트를 관리하는 모듈입니다. 최신 버전은 1.0.3으로 업데이트되었습니다.
- **DBOpenApiW.NET**: DB금융투자 OpenApi를 직접 호출하는 모듈입니다. 이 모듈을 통해 개발자는 금융 데이터를 조회하거나 거래 명령을 실행할 수 있습니다. 최신 버전은 1.0.2입니다.
- **Samples**: 예제 코드가 포함된 폴더입니다. 이 폴더에는 DBCommAgent와 DBOpenApiW를 사용하는 방법을 보여주는 샘플 애플리케이션 코드가 들어 있습니다. 이를 통해 개발자는 라이브러리의 사용 방법을 빠르게 이해하고 자신의 프로젝트에 적용할 수 있습니다.
- **기타 파일들**: 프로젝트 설정 파일(`DBHappyPlus.sln`), 아이콘 파일(`DBHappyPlus.ico`), 그리고 프로젝트의 설명과 라이선스 정보를 담은 문서(`README.md`, `LICENSE.txt`) 등이 포함됩니다.

이 프로젝트는 C# 개발자들이 DB금융투자의 금융 서비스를 자신의 애플리케이션에 통합할 수 있도록 돕기 위해 만들어졌습니다. 개발자는 이 라이브러리를 사용하여 금융 데이터를 조회하거나, 주식, 채권 등의 거래 명령을 실행할 수 있습니다. 이러한 기능은 금융 관련 소프트웨어나 애플리케이션 개발에 매우 유용할 수 있습니다.

프로젝트에 대해 더 자세히 알아보고 싶거나, 코드의 구체적인 사용 방법을 이해하고 싶다면, 프로젝트의 [Github 페이지](#)를 방문하거나, `Samples` 폴더 내의 예제 코드를 참조하는 것이 좋습니다.

**You**

DB API .NET용 nuget package

소스파일 다운로드

DB종류 com버전을 쉽게 이용할 수 있습니다  
일부vs버전에서 생기는 폴더자이 오류 없고, console, winform, wpf, winu3 모두 지원합니다.  
.NET 및 .NET Core : 2.0, 2.1, 2.2, 3.0, 3.1, 5.0, 6.0, 7.0, 8.0  
.NET Framework: 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Nuget명목: DBCommAgent.NET(국내), DBOpenApiW.NET(해외)  
이름 간단히 개인 플러스에서 선언 후 이용하시면 됩니다.(폴더자이 추가 없습니다)  
오른소스로 되어있어, 기능 추가 하일분론 깃헙에서 다운받아 추가 하시면 됩니다

GitHub - teranum/DBHappyPlus: DB금융투자 OpenApi C# wrapper  
DB금융투자 OpenApi C# wrapper. Contribute to teranum/DBHappyPlus development by c...  
github.com

위에 이미지에 있는 내용을 코딩을 전혀 모르는 사람에게도 최대한 쉽게 설명해주세요

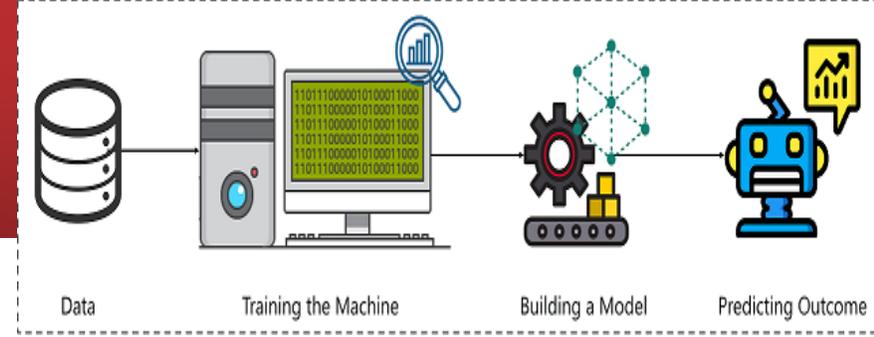
**Data Analyst**

이 이미지에 있는 내용은 DB API와 관련된 .NET 프로그래밍 라이브러리에 대한 정보를 담고 있습니다. 코딩을 전혀 모르는 사람에게 설명하자면, 여기서 말하는 라이브러리는 소프트웨어를 만드는 데 필요한 도구 모음과 같습니다. 마치 요리를 할 때 미리 준비된 소스를 사용하는 것처럼, 개발자들이 더 쉽게 프로그램을 만들 수 있도록 도와주는 코드 조각이나 기능들을 제공합니다.

이미지의 설명을 간단히 요약하면 다음과 같습니다.

- **제품 이름**: DB API .NET용 nuget package
- **목적**: DB와 관련된 작업을 더 쉽게 할 수 있도록 돕는 도구들을 제공합니다.
- **지원 환경**: 이 도구들은 다양한 유형의 .NET 프로그램에서 사용할 수 있습니다. 여기서 .NET은 마이크로소프트에서 개발한 프로그래밍 방식의 하나로, 윈도우 컴퓨터에서 주로 사용됩니다.
- **.NET 지원 버전**: 이 도구는 .NET의 여러 버전에서 사용할 수 있습니다. 버전은 기본적으로 소프트웨어의 다양한 출시본을 의미합니다. 이 도구는 새로운 버전뿐만 아니라 좀 더 오래된 버전에서도 작동합니다.
- **Nuget 패키지**: Nuget은 .NET에서 사용되는 도구 모음을 관리하는 시스템입니다. 이 시스템을 통해 개발자들은 필요한 도구를 쉽게 찾아서 설치할 수 있습니다. 이 이미지에 나와 있는 'DBCommAgent.NET'과 'DBOpenApiW.NET'은 그러한 도구 모음의 예입니다.

# 6-4) Neo Automated Trading Algorithm made by AI

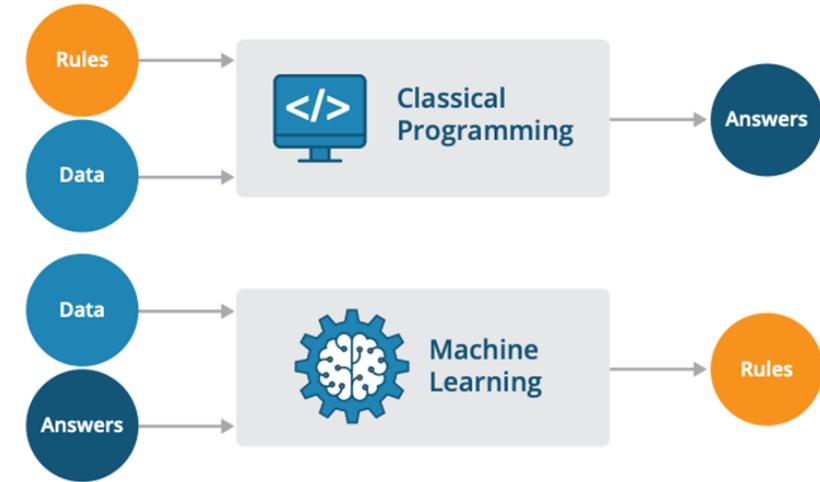


## a) Traditional Programming vs. Machine Learning(Including Deep Learning)

그동안은 Human Trader가 Price, Indicator, 다양한 Data 등을 가지고 Data 등을 가지고 Trading Algorithm을 만들고 그것을 coding을 하는 형태로 Automated Trading App 을 만들었다.

그러나 Deep Learning technology 가 등장하면서 완전히 새로운 형태의 Automated Trading이 등장하게 되었다.

즉 매매 알고리즘조차 Deep learning(AI)에 의해 만들어지고, 자동매매는 전적으로 AI에 의해서 관리 운영되는 형태다.



## b) How can I make a trading Algorithm using Machine Learning

AI 에게 최대의 수익을 내는 것이 너의 목표라고 임무를 주고, 수백만개의 price Data나 Chart를 Input으로 넣어주고, 미세조정 및 강화학습, 지도학습을 시키면 AI는 특정 '매매 알고리즘' 을 만들게 되고, 실제 매매를 하면서 스스로 알고리즘을 수정해 간다. 이제 더 이상 인간이 매매 알고리즘을 만들 이유도 필요도 없어지고 있는 시대에 접어들었다.

### 기존 컴퓨터와 양자컴퓨터 비교

구분	기존 컴퓨터	양자컴퓨터
연산 개념도	<p>0 1 정보를 0이나 1로 표현</p> <p>0 1 0 0 0 1 0 0 0 0 1 1 0 0 0</p> <p>0 1 0 0 0 1 0 0 0 0 1 1 0 0 0</p> <p>1 0 0 0 0 1 0 0 0 0 0 0 0 0 0</p> <p>↓</p> <p>많은 시간 소요</p> <p>1개씩 순차적으로 계산</p>	<p>0 1 0과 1을 중첩</p> <p>0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</p> <p>0 0 0 0 0 0 0 0 0 0 0 0 0 0 0</p> <p>⋮</p> <p>1 0 0 0 0 1 0 0 0 1 0 0 0 0 0</p> <p>↓</p> <p>순식간에 계산</p> <p>합쳐서 한번에 계산</p>
기본 단위	Bit(0 또는 1)	Qubit
연산 방법	논리 표에 의한 계산	행렬 함수에 의한 계산

## c) Quantum Computer and AI

기존의 Computer가 Binary 방식으로 Computing 한다면, 양자 컴퓨터는 양자이론(Quantum Theory)에 근거한 Q-bit라는 방식(양자중첩, 양자얽힘 등을 계산식으로 사용함)으로 계산. 이론적으로 GPU보다 연산속도가 1만배 빠름.

**성공한다면 우리가 상상만 하던 GAI & Robot 세상이 드디어 열림!**

# 7) Negative Sum Game(Zero Sum Game)의 본질적 이해

참가자들이 자신의 수익을 극대화하기 위해 경쟁하면, 게임은 점점 더 치열해진다. 이 과정에서 승리하기 위한 전략이나 기술의 발전은 일시적인 이득을 가져올 수 있지만, 결국 다른 참가자들도 비슷한 전략이나 기술을 개발하여 경쟁 우위를 상쇄시키게 된다.

## a) Negative Sum Game vs. Positive Sum Game

제로섬 게임에서는 한 참가자의 이득이 반드시 다른 참가자의 손실로 이어진다. 선물옵션 시장은 본질적으로 Zero Sum Game(수수료와 파생양도세를 고려하면 Negative Sum Game)이 된다.

## b) Why Most Individual Derivatives Traders Lose Money? 매매전략이 문제가 아니라 게임의 Rule이 문제

제로섬 게임에서는 지속적인 수익을 내기 위해서는 상대방을 압도하는 데 초점을 맞추어야 한다. 하지만 모든 참가자가 다른 참가자를 압도하기 위해서 계속 경쟁하기에 어떤 매매전략도 누구도 지속적인 이득을 보장받을 수 없게 된다. 이게 게임의 본질적인 속성이다.

파생은 절대평가가 아니라 상대평가다. 제로섬 파생시장에서 외국계, 기관이 수익을 늘 독점, 개인들은 항상 큰 손실만 보는 존재임. 앞으로는 인간보다 몇 백배 더 똑똑한 AI들이 만들어내는 Algorithm과 싸워서 이겨야 하는 상황이라 향후 파생시장에서 개인 트레이더가 수익낼 가능성은 더 줄어들고 있다.

누구와 싸우고 있는지 부터 알고, 과연 그들을 이길 수 있는지도 고민해보자. 일정 수준에 도달하면 작은 수익을 얻을 수 있지만, 그것은 단지 행운일 뿐, 게임에 참여하는 한 참여자들보다 늘 우위에 있도록 부단한 노력과 공부(English, Coding, Analysis Macro issues, Making Valid algorithms, Getting Insight, Mind control, Discipline continuously)를 해야 한다.

한 순간이라도 뒤쳐지면 바로 시장에서 퇴출된다.

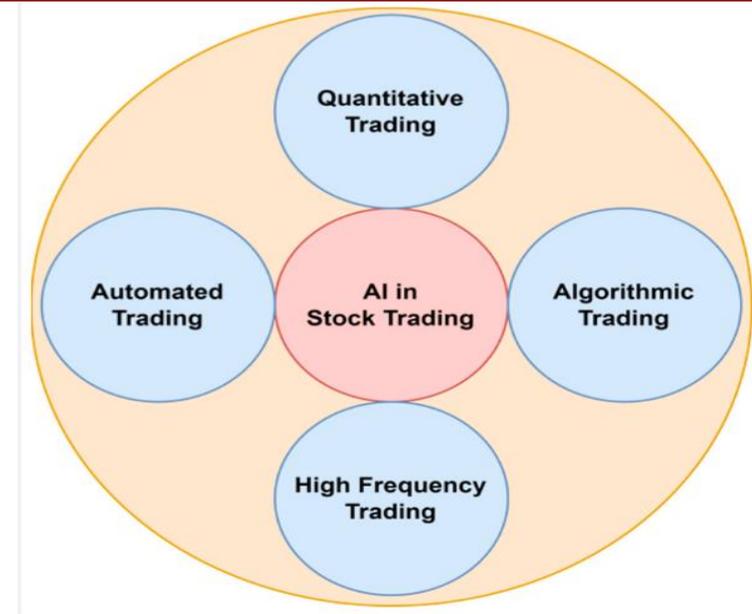
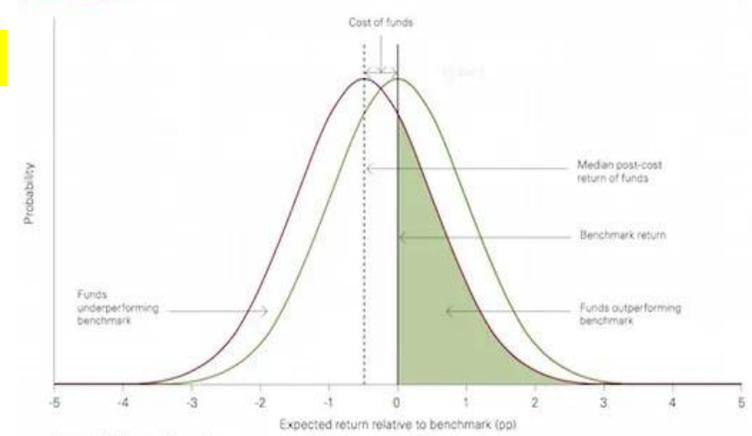


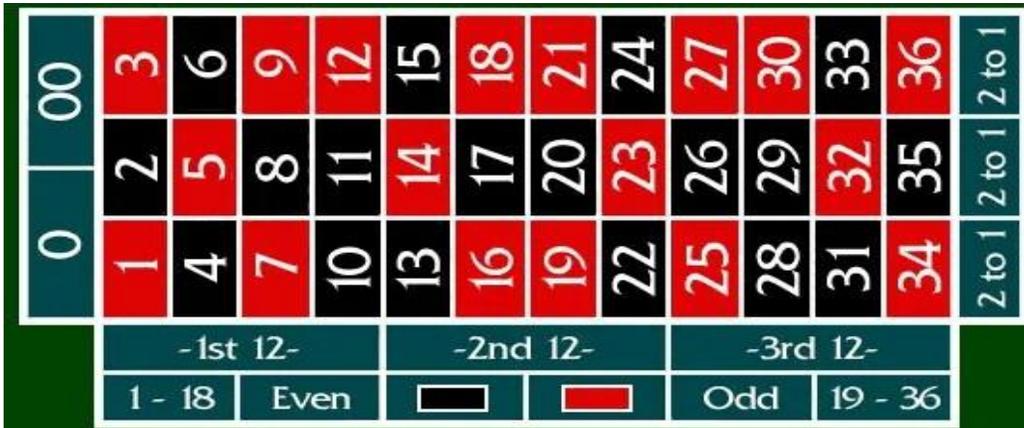
Figure 1. The zero-sum game



# 7-1) How to get a Winning Edge in Zero-Sum Game.

Study permanently, Develop Algorithms continuously, Discipline till you die!

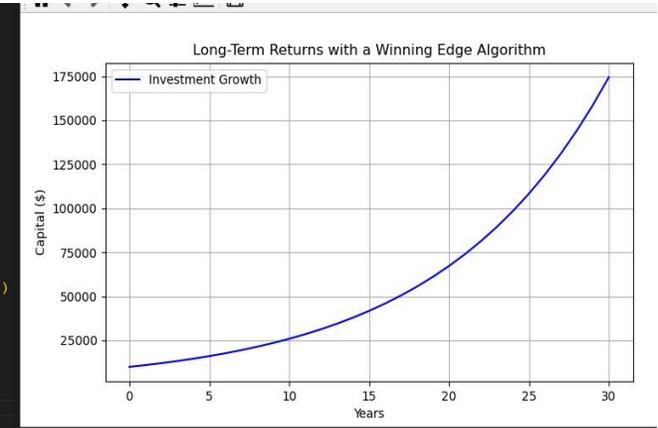
- a) Winning Edge란 말 그래도 승리하는 끝자리란 뜻으로, 경쟁이 치열한 분야에서 성공할 수 있는 결정적인 우위나 장점을 의미
- b) Casino는 승률 1% 더 유리하게 만든 구조적인 winning edge를 가지고 있기에 꾸준히 수익이 나는 반면, 도박꾼(Negative sum game player)는 꾸준히 손실이 남(Roulette Game에서 주사위가 0과 00이 들어가면 항상 카지노 승리 이게 바로 winning edge)



```

1 import numpy as np
2 import matplotlib.pyplot as plt
3
4 # 가정 설정
5 initial_capital = 10000 # 초기 자본
6 annual_return_rate = 0.10 # 연간 수익률 10%
7 years = 30 # 투자 기간 30년
8
9 # 시간 배열 생성
10 years_array = np.arange(0, years + 1, 1)
11
12 # 수익률 계산 (복리 계산)
13 final_capital = initial_capital * (1 + annual_return_rate) ** years_array
14
15 # 그래프 생성
16 plt.figure(figsize=(10, 6))
17 plt.plot(years_array, final_capital, label='Investment Growth', color='blue')
18 plt.title('Long-Term Returns with a Winning Edge Algorithm')
19 plt.xlabel('Years')
20 plt.ylabel('Capital ($)')
21 plt.legend()
22 plt.grid(True)
23 plt.show()
24

```



- c) Trading이나 Investment에서의 winning edge는 단순히 한두 번의 거래에서 운 좋게 수익을 내는 것이 아니라, 장기간에 걸쳐 지속적으로 수익을 창출할 수 있는 능력 -> 시장에서 Individual Trader가 AI가 만든 Algorithm, 외국계 헤지펀드와 수십억 연봉의 기관투자자들과 싸워서 Winning Edge를 얻을 수 있을까?
- d) 혹시나 일시적 Gambler's luck or beginner's luck 을 우리는 Winning Edge를 얻었다고 착각하고 있는 것은 아닐까? 3년간 수익낸 것을 하루만에 다 날렸다(이것은 Winning Edge 즉 수익나는 algorithm이 애초에 없었다는 이야기임. 3년간의 수익은 그저 단순한 행운이었음. 카지노가 어떤 한 명의 갬블러가 크게 수익이 났다고 해서 망하는가? 내부적 외부적 어떠한 event가 발생해도 수익률 그래프가 우상향되어야 winnig Edge)
- e) Zero-sum game 에서 winning edge 를 얻으려면, '지속적인 정보의 우위, 지속적인 리스크 관리의 우위, 지속적인 자기통제의 우위, Deep learning의 의해 꾸준히 Updated Algorithm Trading을 해야함(인간의 의해 코딩된 매매 알고리즘 아니라, AI에 의해 꾸준히 자동적으로 수정되는 매매 알고리즘에 기반한 자동화된 시스템을 사용하여, 인간의 감정 없이 시장 기회를 식별하고 활용하는 능력)

# 7-2) Positive Sum game with Leverage(Debt)

a) The result will be different even though the strategy(Algorithm) is the same

암호화폐 시장(현물시장 기준)도 본질적으로 Negative Sum Game(Miner와 Exchange fee), Bitcoin은 장기적으로 계속 시가총액이 늘어나고 있음. **Bitcoin Trading Market might be Positive Sum Game**

같은 매매 전략이라도 선물옵션과 같은 Zero sum game에서는 수익내기 어렵지만, 계속 우상향하는 Positive sum game에서 수익 내기 훨씬 수월하다. **가격이 하락하는 차트에서는 Short 외에는 어떤 매수전략도 수익이 나지 않는다. You may find the winning edge in the positive sum game!**

b) What do you want? To be rich? Vs To be a successful trader?

트레이더로서 성공해서 돈 버는 길 vs. 투자, 사업, Youtuber, AI 작곡가 등

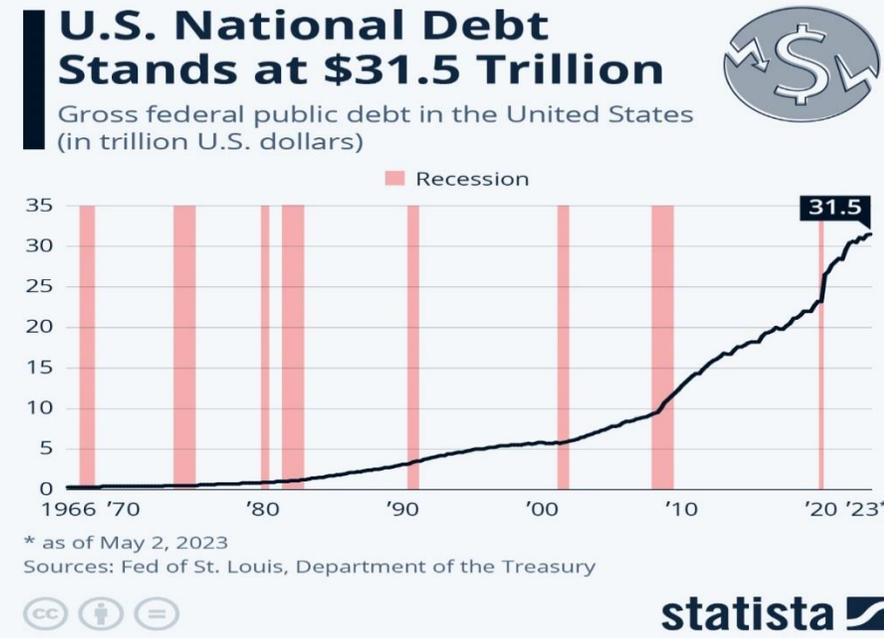
파생 트레이딩은 돈 버는 수단 중 하나! 파생 트레이딩에 쏟을 에너지와 시간을 다른 분야에 투자한다면 훨씬 더 부자가 될 수 있다! **A simple way to be rich -> "Invest Index(Market) and Enjoy your life" By John Bogle.**

c) Rich people hide this secret(Debt made me rich), 레버리지는 Positive Sum Game에서나 사용하자.

한국에서 가장 쉽게 부자가 되는 방법 : 빚을 많이 내서 부동산을 사두면, 처음 몇 년은 이자 등으로 힘들지만, 5년 정도 지나면 가격이 오름(Inflation effect) -> 다시 빚을 내서 또 Apartment를 삼, 몇 년은 힘들지만 전세주고 버티면 가격이 오름(Printing Money) -> **그렇게 10년 전에 강남에 아파트 3채 사두었으면 100억 부자**

45% Winners @ 2:1 R:R		VS	65% Winners @ 1:2 R:R	
2	-1		0.5	-1
2	2		0.5	0.5
-1	-1		-1	-1
-1	-1		0.5	0.5
2	2		-1	0.5
-1	-1		0.5	-1
2	2		0.5	0.5
-1	-1		-1	0.5
2	2		0.5	-1
-1	-1		0.5	0.5
<b>Positive 7 Units</b>			<b>Minus 0.5 Units</b>	

\*Transaction costs and slippage not included.



# 8) Briefing Schedule of 'Automated Trading using API with ChatPGT'

March ~ May  
(Beginner course)

Overview 'API & Automated Trading' For Beginners

Understanding of Zero-sum Game

English Study

Price theory Study

June ~ August  
(Intermediate)

증권사 Rest API 함수의 이해 및 선물옵션 자동매매 coding

Upbit 거래소 API를 이용한 암호화폐 자동매매 Coding

September ~ October  
(Advanced)

Making trading algorithm using Deep learning

증권사 Rest API를 이용한 자동매매 test

Upbit, Binance 등 Exchange 에서 자동매매 coding and 알고리즘 테스트

- We will use ChatGPT paid version for this coding.
- Python is required for coding and C# is optional
- English is Must-learn Skill. English is Essential for your future. Prepare Shrinking Society

