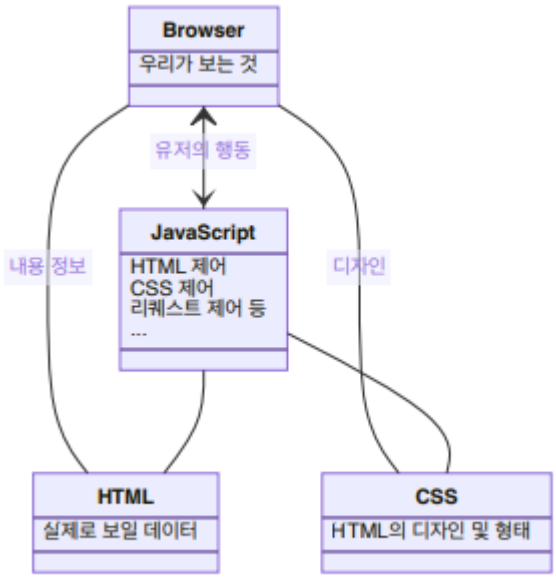
웹 크롤링

웹 크롤링: 웹 크롤링이란 웹상의 정보들을 탐색하고 수집하는 작업을 의미합니다. 인터넷에 존재하는 방대한 양의 정보를 사람이 파악하는 것은 불가능한 일입니다. 때문에 규칙에 따라 자동으로 웹 문서를 탐색하는 컴퓨터 프로그램, 웹 크롤러(Crawler)를 만들었습니다.

구조)

웹페이지란 기본적으로 사람들에게 어떠한 정보를 보여주기 위해 존재합니다. 아주 기본적인 웹페이지부터 가장 발전된 웹페이지 는 모두 아래와 같은 기본적 규격을 따르고 있습니다.



HTML:은 우리가 우리 눈에 보이는 데이터들을 모아둡니다. 문서로 치자면 문서의 글 및 내용에 해당되는 데이터 입니다.

CSS:는 HTML을 이쁘게 보일 수 있도록 해줍니다. 문서로 치자면 문서의 서식, 폰트, 색상 등 입니다.

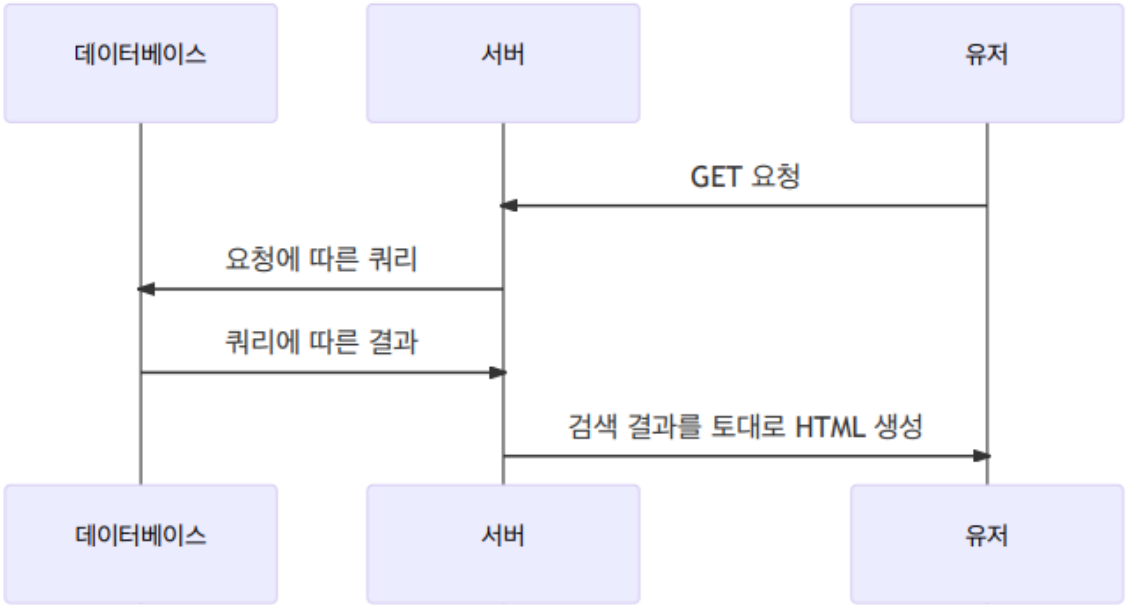
JavaScript:는 웹페이지에서 동작하는 프로그래밍 언어로, HTML의 내용을 변경하거나, CSS등을 변경하여 스타일을 변경할 수 있습니다. 또한 웹 리퀘스트 등을 통해서 추가로

정보를 가져와 유용한 정보를 표시할 수 있도록 도와줍니다.

Browser: 는 크롬, 익스플로러, 파이어 폭스 등 웹 페이지를 직접 보는 도구 입니다. 일반적 웹 브라우저들은 자바스크립트 실행 기를 가지고 있습니다. 따라서 사용자의 행동(클릭하기, 키보드 입력하기 등)에 따라 특정 자바스크립트를 실행시킬 수 있습니다.

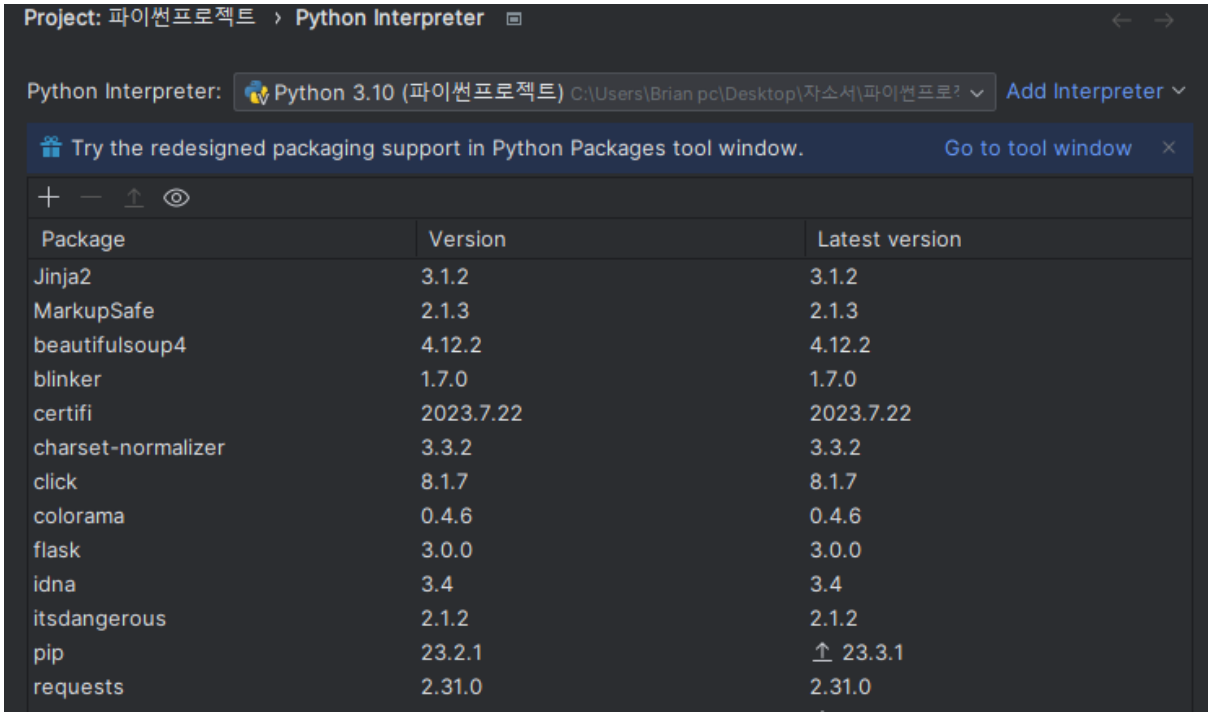
SSR 서버 크롤링

SSR 서버 크롤링이란: 원시적인 형태의 웹은 자바스크립트를 크게 사용하지 않고, 서버에서 데이터를 직접 가져와 HTML 을 수정하여 유저에게 값을 보 냅니다. 이것을 SSR (Server Side Rendering) 이라고 합니다. SSR 웹 서버의 동작은 아래와 같습니다.



위와 같이 웹 페이지에 요청을 한 경우 서버에서 HTML까지 로딩하여 결과값을 전송해줍니다.

SSR 웹 크롤링 실전



Package	Version	Latest version
Jinja2	3.1.2	3.1.2
MarkupSafe	2.1.3	2.1.3
beautifulsoup4	4.12.2	4.12.2
blinker	1.7.0	1.7.0
certifi	2023.7.22	2023.7.22
charset-normalizer	3.3.2	3.3.2
click	8.1.7	8.1.7
colorama	0.4.6	0.4.6
flask	3.0.0	3.0.0
idna	3.4	3.4
itsdangerous	2.1.2	2.1.2
pip	23.2.1	↑ 23.3.1
requests	2.31.0	2.31.0

PYCHARM을 통해 requests와 beautifulsoup4를 설치한다.

```
import requests
print(requests.get('https://ko.wikipedia.org/wiki/%EC%9C%84%ED%82%A4').text)
```

requests를 통해 위키에 있는 데이터를 가져온 결과 크롬에서 들어간 위키의 html 값이 나오는 것을 확인할 수 있다.

```
 -->
</div><!--esi <esi:include src="/esitest-fa8a495983347898/content" /> --><noscript><img src="https://login.wikimedia.org/wiki/Special:CentralAutoLogin/
<div class="printfooter" data-nosnippet="">원본 주소 "<a dir="ltr" href="https://ko.wikipedia.org/w/index.php?title=위키&amp;oldid=34784771">https://ko.w
                    <div id="catlinks" class="catlinks" data-mw="interface"><div id="mw-normal-catlinks" class="mw-normal-catlinks"><a href="/wiki/%ED%
                </div>
            </main>

        </div>
        <div class="mw-footer-container">

<footer id="footer" class="mw-footer" role="contentinfo" >
    <ul id="footer-info">
    <li id="footer-info-lastmod"> 이 문서는 2023년 4월 24일 (월) 23:22에 마지막으로 편집되었습니다.</li>
    <li id="footer-info-copyright">모든 문서는 <a href="https://creativecommons.org/licenses/by-sa/4.0/deed.ko">크리에이티브 커먼즈 저작자표시-동일조건변경허락 4
</ul>

    <ul id="footer-places">
    <li id="footer-places-privacy"><a href="https://foundation.wikimedia.org/wiki/Special:MyLanguage/Policy:Privacy_policy">개인정보처리방침</a></li>
    <li id="footer-places-about"><a href="/wiki/%EC%9C%84%ED%82%A4%EB%B0%B1%EA%B3%BC:%EC%86%8C%EA%B0%9C">위키백과 소개</a></li>
```

Beautifulsoup4: 는 웹 페이지의 정보를 읽는데 사용하는 도구 입니다. bs4.BeautifulSoup(result, 'html.parser') 를 통해 해당 문서가 html 임을 명기합니다. select 를 이용하여 원하는 위치의 값을 가져올 수 있는데, div#siteSub 은 div 태그 중 siteSub 라는 id 를 가진 것을 모두 찾아내라는 뜻입니다

```
import requests
import bs4
result = requests.get('https://ko.wikipedia.org/wiki/%EC%9C%84%ED%82%A4').text
parsed = bs4.BeautifulSoup(result, features: 'html.parser')
selected: [bs4.Tag] = parsed.select('div#siteSub')
print(selected)
print(selected[0].text)
```

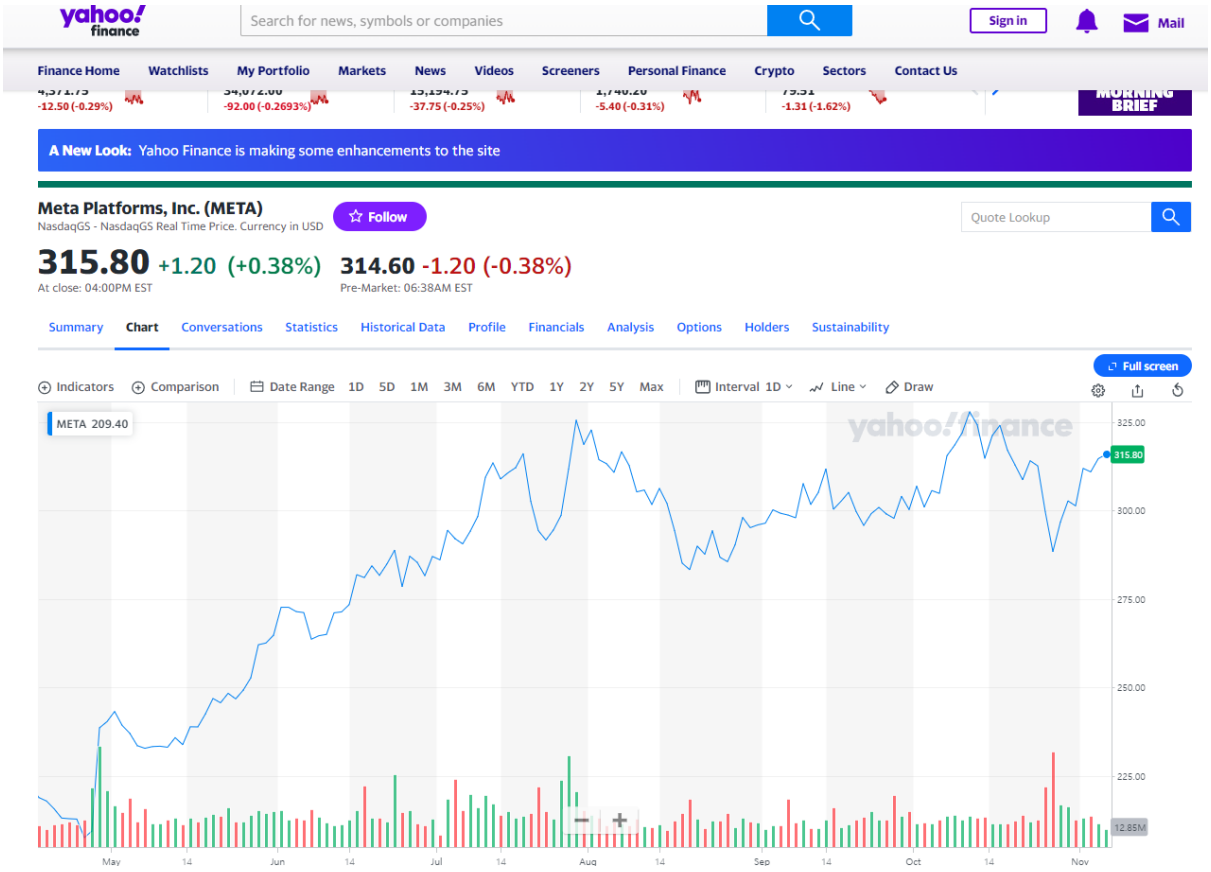
CSR 웹 크롤링

현대적인 웹 프론트엔드들은 서버에서는 별도로 값을 렌더링 하지 않음으로 부하를 줄이고, 자바스크립트등을 통해 사용자의 웹 브라우저가 직접 렌더링을 하게 하는 CSR(Client Side Rendering)을 주로 사용합니다.

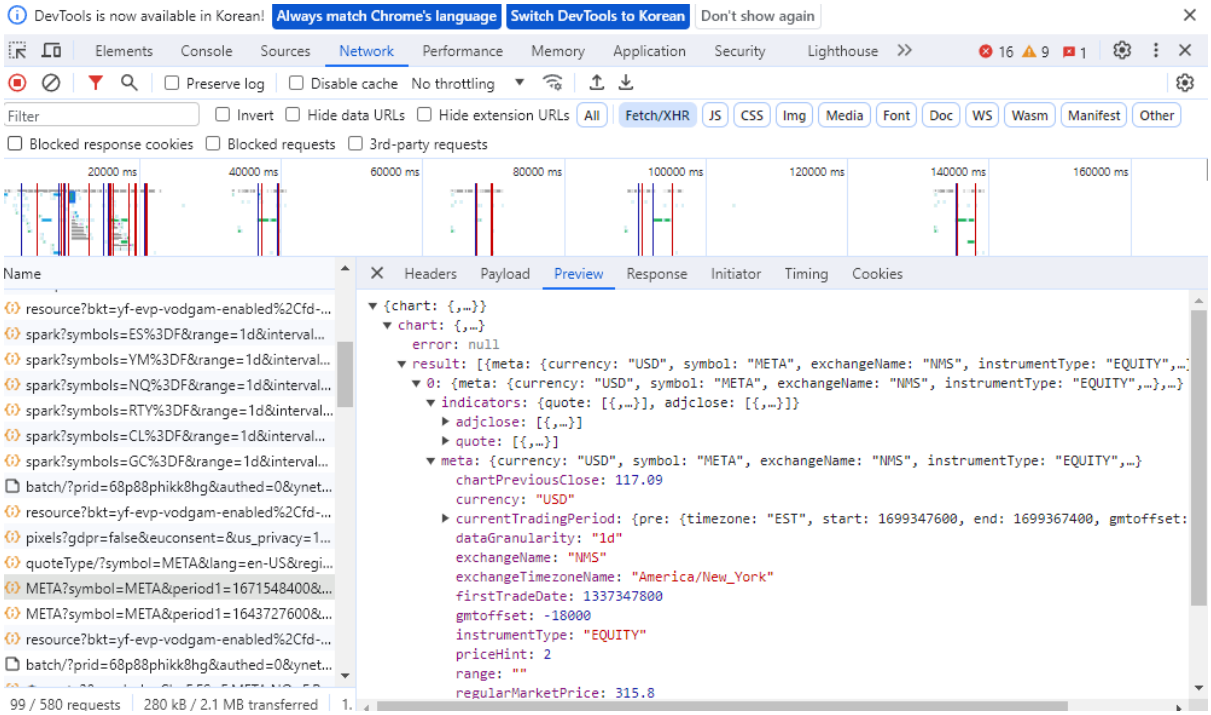
최근에 나오는 React, Vue, Anagular 등의 웹 프론트엔드 프레임워크들이 이에 해당됩니다. 이러한 코드들로 짜여 있는 웹 페이지는 대부분 위의 SSR 방식의 페이지보다 크롤링 하기 수월할 때가 많지만 그렇지 않은 경우 또한 많습니다. 일반적인 CSR 웹 페이지의 경우 builder 와 api 두가지가 존재합니다.

builder는 웹 페이지를 구성할 수 있도록 해주는 자바스크립트로써, 데이터를 파싱할 우리에게는 크게 중요하지 않은 항목입니다. api 의 경우 여러가지 방법(GET , POST , PUT , DELETE 등)으로 호출할 수 있습니다. 호출 값에 따라서 원하는 값만 JSON 형태로 제출해주는 경우가 많습니다. 이에 따라 크롤링 하기가 더욱 편리 해집니다

실전)



CSR 웹 페이지인 yahoo finance를 파싱



크롬의 개발자 도구를 통해 해당 기업의 정보등을 가지고 있는 쿼리를 가져온다.

```
https://query1.finance.yahoo.com/v8/finance/chart/META?symbol=META&period1=167
1548400&period2=1699357242&useYfid=true&interval=1d&includePrePost=true&eve
nts=div%7Csplit%7Cearn&lang=en-US&region=US&crumb=ZRx%2Fd7gF6Rx&corsDo
main=finance.yahoo.com
GET
```

파싱한 쿼리

```python
import requests
import json
result = json.loads(requests.get(
 'https://query1.finance.yahoo.com/v8/finance/chart/META?sym
print(result)
```

단순 requests를 통해 데이터를 가져올경우 접근은 제한되고 원하는 정보를 가져올 수 없습니다.

서버 속이기)

다시 크롬 개발자 모드로 가서 가져온 정보들로부터 Request Headers을 찾습니다.

Request Header란: 브라우저가 서버에게 보낸 특정 값입니다. 이를 통해 인증을 할 수 있습니다.

동일한 Request Header를 통해 전송할겨우 서버가 브라우저에서 접근했는지 아닌지 알수 없습니다.

```
import requests
import json
with requests.Session() as sess:
 sess.headers['user-agent'] = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/
 sess.headers['referer'] = 'https://finance.yahoo.com/quote/META/chart?p=META'
 result = sess.get('https://query1.finance.yahoo.com/v8/finance/chart/META?symbol=MET
 data = json.loads(result)
 ochlv = data['chart']['result'][0]['indicators']['quote'][0]
 time_data = data['chart']['result'][0]['timestamp']
 print(ochlv['open'])
 print(len(time_data))
 print(len(ochlv['open']))
```

사용자 검증을 포함한 코드

User-agent: 접속자의 브라우저가 어떤것인지 정보를 포함하고 있다.

Referer: 해당 리퀘스트가 만들어졌을 때의 사용자의 웹 페이지의 경로를 가리킵니다.

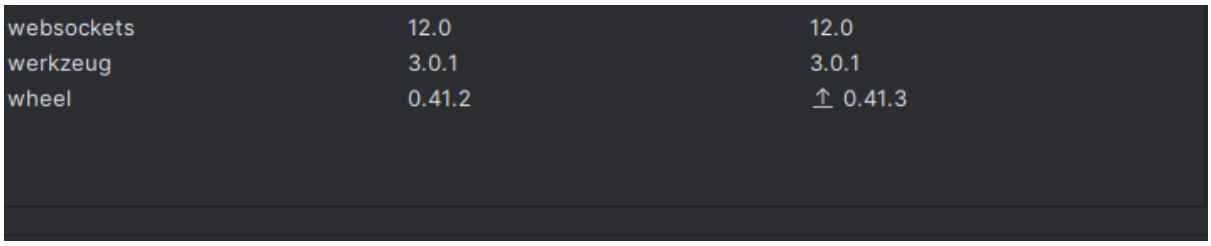
```
"C:\Users\Brian pc\Desktop\자소서\파이썬프로젝트\venv\Scripts\python.exe" "C:\Users\Brian pc\Desktop\자소서\파이썬프로젝트\main.py"
[191.1999969482422, 194.97000122070312, 195.19000244140625, 180.5500030517578, 180.39999389648438, 182.8800048828125, 191.36000
369
369

Process finished with exit code 0
```

크롤링한 데이터

Web Socket

바이낸스 등과 같은 거래소는 실시간으로 많은 데이터가 오가야 합니다. 이에 일반적인 리퀘스트가 아닌 websocket 이라는 것 을 사용하게 됩니다. 소켓의 경우 broadcast 등 많은 사용자에게 서버가 원하는 시간 원하는 데이터를 주고 받을 수 있는 장점이 있습니다. 이를 통해 실시간으로 거래 정보를 얻어올 수 있습니다

실전)



PYCHARM을 통해 웹소켓 설치

```python
import websockets
import json
import asyncio

1 usage
async def subscribe():
 uri = "wss://stream.binance.com/stream"
 async with websockets.connect(uri) as websocket:
  payload = dict(
   method="SUBSCRIBE",
   params=["!miniTicker@arr@3000ms",
           "btcbusd@aggTrade",
           "btcbusd@kline_1d",
           "btcbusd@depth"
           ],
   id=2
  )
  await websocket.send(json.dumps(payload))
  while True:
     d = await websocket.recv()
     print(d)


asyncio.get_event_loop().run_until_complete(subscribe())
```

웹소켓 라이브러리른 비동기 라이브러리이기 때문에 비동기 함수 호출을 위해 asynchio를 이용해야 합니다.

While True 와 await 을 통해 양방향 통신으로 웹 소켓 정보가 입력될때마다 갑을 출력 및 해석할 수 있습니다. 이를 통해 코인 거래 내역 및 가격 변동을 실시간으로 얻을 수 있습니다.

실시간으로 거래되고 있는 데이터