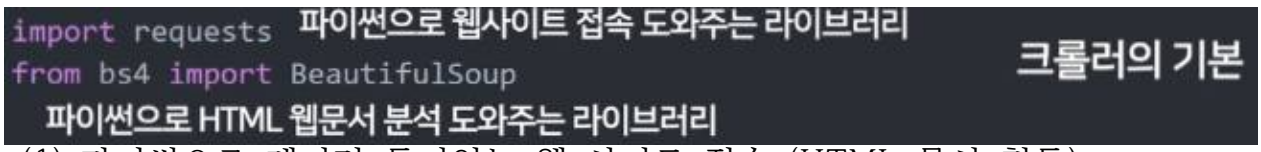
1. 웹 개발 상식 (기초 동작 방식)

[Server] <== [Client]
- GET 요청 : 서버의 페이지를 읽을 때
- POST 요청 : 서버에 데이터를 보내고 싶을 때
ㄴ(Python의 역할) "요청이 들어왔을 때 어떠한 동작을 할 것인가"
<"HTML문서" .js .css .png .jpg등 전달>

2. 웹 크롤러

크롤러란?
- 웹에 있는 데이터를 알아서 수집하고 저장해주는 프로그램
- (ex) OP.GG 사이트

• 크롤러의 기본



(1) 파이썬으로 데이터 들어있는 웹 사이트 접속 (HTML 문서 획득)
=> data = requests.get('URL')
=> 이때 발생하는 인코딩에 의한 데이터 깨짐 문제 등을 해결해주는 역할로 "BeautifulSoup"를 사용함.
=> soup = BeautifulSoup(data.content, 'html.parser')

(2) HTML 속에서 필요한 정보 추출
=> print(soup.find_all('strong', id="_nowVal"))
=> find_all('태그명', '속성명')
=> 결과 값은 항상 **'리스트'**로 저장이 됨 : indexing 중요

• 추출할 요소의 다양한 Case

(1) 추출할 글자가 해체되어 있는 경우
=> 상위 클래스의 정보를 추출하면 묶어서 추출할 수 있음.

(2) class, id가 없는 태그 요소
=> 태그 이름만으로 추출할 수는 있으나, 너무 많은 정보를 추출할 수 있음.
=> soup.select(' ') 사용 (CSS selector)
=> '.class명' / '#id명' / '태그명' / '.class명1 .class명2'
=> 마찬가지로 결과 값은 항상 **리스트**로 저장이 됨

(3) 이미지 수집
 => img 태그의 src 속성만 따로 추출
  => 이미지 URL : image['src']
 => 이미지 URL을 실제 이미지 파일로 저장하는 방법

```
import urllib.request #import 모여있는 맨위에다가 작성

urllib.request.urlretrieve(이미지URL, '파일명')
```

(4) 여러 종목을 크롤링하는 경우
 => 반복문을 통해 URL만 바꾸면서 같은 동작 반복 EZㅋㅋ

• 함수를 이용한 크롤러
 => 코드를 파라미터로 받고 변수 대입
 => 변수를 문자열 사이에 집어넣어 코딩할 때 : 'f-string' 사용

```
f'문자 {변수} 문자'
```

 => 추출값을 함수를 통해 받을 때는 return 값을 통해서.

• 무한 스크롤 데이터 수집 <-- AJAX data
(1) 위 데이터 수집 방식과의 차이
 => 위의 방식은 정적 HTML 문서를 받아 분석하고 원하는 데이터만을 따로 추출하여 쓰는 것.
 => 무한 스크롤 사이트에서의 수집방식은 원하는 데이터를 서버에게 요청하여 데이터를 동적으로 받는 것.
(2) 데이터 요청 방법
 => 사이트의 개발자 도구, Headers 탭에서 원하는 데이터를 검색해보면 추가되는 데이터의 URL과 서버 요청방식을 보여줌.
 => 이후 requests.get('URL') // GET 요청방식
 => 결론) "URL 분석이 중요하다"

3. JSON 데이터

JSON이란?
- Python의 dictionary 형태와 유사하지만 텍스트 형식으로 저장되는 데이터 저장방식을 말함.
- 형식 : {"자료이름" : "값"}

=> python에서 json을 쉽게 다루기위해서는 dictionary 형으로 바꿔줄 필요가 있다.

=> import json
json.loads(data.content)

• 방대한 내용의 json 파일 분석하기 쉽게 만드는 법
(1) 웹에서 수집한 json 파일의 경우, URL 값을 주소창에 입력하여 html 문서로 보이게 한다.
(2) vscode에 임의의 json 파일을 만들고 복붙한다.
(3) 우클릭 > Format Document

• Epoch time / UNIX
- 1970년 1월 1일부터 지금까지 몇 초 흘렀는지 초단위로 알려주는 형식
- 원래는 10자리이지만 ms 단위로 나타낼 경우 0을 세 개 더 표현하기도 함
- 시간 사이의 계산을 할 때 컴퓨터 입장에서 편하기 때문에 가끔 사용함

```
import time
time.strftime('%Y-%m-%d %H:%M:%S', time.localtime(epoch시간형식))
```

** 사람이 보는 시간형식으로 변환할 수 있다.