Requests 복습 및 심화

최희윤 강사

Requests

```
import requests

URL = 'http://httpbin.org/get'

response = requests.get(URL)

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

requests.get("http://httpbin.org/get")

requests.post("http://httpbin.org/post")

- 새로운 사용자 계정 같은 새로운 데이터 제출
- 서버에 새로운 리소스 추가

Requests.put("http://httpbin.org/put")

- 서버에 있는 기존 데이터 업데이트
- 특정 위치에 리소스 생성

requests.delete("http://httpbin.org/delete")

- url주소의 특정 리소스 삭제

Requests

```
import requests

URL = 'http://httpbin.org/get'

response = requests.get(URL)

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

웹 크롤링을 하는 이유는 **데이터를 갖고오기 위함**이기 때문에 **Get** 방식을 주로 씀

Params

```
import requests

URL = 'http://httpbin.org/get'

params = {'data1': 'data1', 'data2': 'data2'}
response = requests.get(URL, params=params)

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

URL에 파라미터 전달하는 방법

Header

```
import requests

URL = 'http://httpbin.org/get'

headers = {'Content-Type': 'application/json; charset=utf-8'}
response = requests.get(URL, headers=headers)

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

몇몇 사이트들은 header에 정보가 없어서 Response받지 못하는 경우도 있음

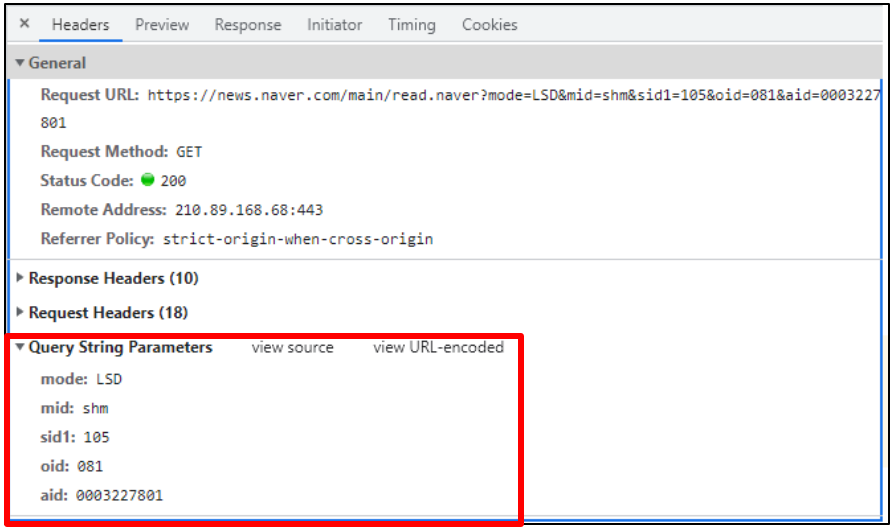
Body

```
import requests

URL = 'http://httpbin.org/post'

data = {'data1': 'data1', 'data2': 'data2'}
response = requests.post(URL, data=data)

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

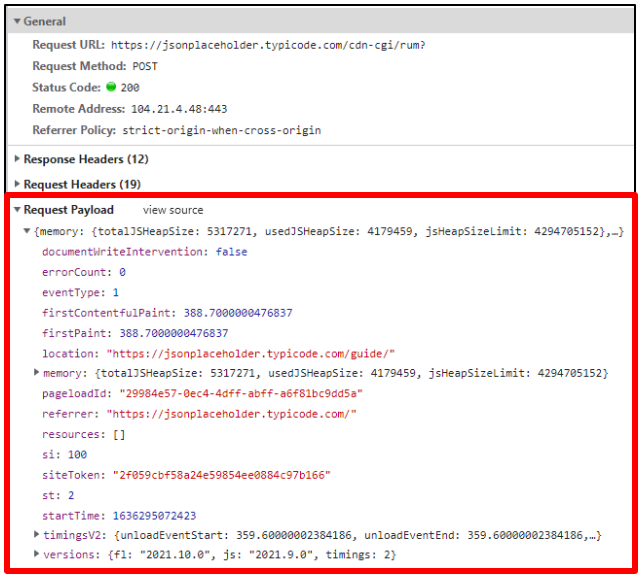
사용자 로그인 데이터, 검색폼 데이터,
제품 주문 데이터 등 post 방식에서 쓰임

get, post 구조

✓ get 방식
- 요청 시 body가 없음
- 데이터 전달 시 parameters 사용
- url?key=value&key=value



✓ post, put, delete 방식
- payload(body에 포함되 내용)에 클라이언트가 서버에 전송하고자 하는 실제 데이터



BeautifulSoup

최희윤 강사

BeautifulSoup

HTML을 분석해 필요한 데이터를 갖고 와주는 도구

라이브러리 설치 : pip install BeautifulSoup4

실습할 파이썬 스크립트/대화형 파일을 생성하고
강사님을 따라 실습을 진행해 보세요

실습 URL : https://crawlingstudy-dd3c9.web.app/01/

BeautifulSoup

✓ **원하는 태그 갖고 오는 방법**

1. html 소스를 갖고온다 : **source = requests.get(url).text**
2. BeautifulSoup으로 html을 구조화 하고 분석한다 : **BeautifulSoup(source, 'html.parser')**

   - html.parser : html 포맷의 문서를 parsing (구조화하고 분석)

   - parsing하는 이유 : html을 분석하여 원하는 태그를 갖고오기 위함

3. 특정 태그를 선택하여 원하는 데이터를 갖고 온다.

   - **soup.select_one('css 선택자') / soup.select('css 선택자')**
   - **soup.find('tag_name') / soup.find_all('tag_name')**

select로 데이터 갖고 오기

✓ 종류

1. .select('acss 선택자')

   - css선택자를 파라미터로 받음

   - 모든 일치하는 태그를 list형식으로 return -> 인덱싱, 슬라이싱 가능

   - html.select(), tag.select() 둘 다 가능

2. .select_one('css 선택자')

   - 매칭되는 태그 중 가장 첫 번째 태그를 갖고 옴

   - list에 넣지 않고 바로 return해줌

   - html.select_one(), tag.select_one() 둘 다 가능

select로 데이터 갖고 오기

✓ css 선택자로 태그 갖고 오기

1. **태그** 셀렉터 : soup.select('tag_name') / soup.select_one('tag_name')
2. **ID** 셀렉터 : soup.select('#id_name') / soup.select_one('#id_name')
3. **Class** 셀렉터 : soup.select('.class_name') / soup.select_one('.class_name')
4. **속성** 셀렉터 : soup.select('tag[속성='속성값']') / soup.select_one('tag[속성='속성값']')
5. **후손** 셀렉터 : soup.select('tag tag2') / soup.select_one('tag tag2')
6. **자식** 셀렉터 : soup.select('tag > tag2') / soup.select_one('tag > tag2')

select로 데이터 갖고 오기

✓ css 선택자로 데이터 갖고 오기

속성 셀렉터 : soup.select('tag[속성='속성값']') / soup.select_one('tag[속성='속성값']')

- tag[속성~ = "값"] : 해당 단어와 일치
- tag[속성^ = "값"] : 해당 값으로 시작
- tag[속성$ = "값"] : 해당 값으로 끝나는
- tag[속성* = "값"] : 해당 값을 포함하는

find로 데이터 갖고 오기

1. **.find_all('tag_name', limit=2)**

   - limit옵션 통해 가져올 개수 제한 할 수 있음
   - 모든 일치하는 태그를 list형식으로 return -> 인덱싱, 슬라이싱 가능
   - html.find_all(), tag.find_all() 둘 다 가능

2. **.find('tag_name')**

   - 매칭되는 태그 중 가장 첫 번째 태그를 갖고 옴
   - list에 넣지 않고 바로 return해줌
   - html.find(), tag.find() 둘 다 가능

find로 데이터 갖고 오기

1. 해당하는 **클레스 이름의 태그** 갖고 오기

   - soup.find('tag_name', 'class_name')

   - soup.find('tag_name', class_='class_name')

2. 특정 태그 중 특정 **속성 값을 가진 태그** 갖고 오기

   - soup.find('tag_name', attrs={'속성이름':'속성값'})

3. 해당하는 **아이디값을 가진 태그** 갖고오기

   - soup.find(id='id_name')

find로 데이터 갖고 오기

✓ soup.find()

- 태그의 **text** 갖고 오기
  - soup.find('h1').text
  - soup.find('h1').get_text()
  - soup.find('h1').string
- 태그의 **속성 값** 갖고 오기
  - soup.find('a').attrs['href']
- **태그에서 태그** 찾기/갖고 오기
  - result = soup.find('table') : 후손 태그까지 같이 갖고 옴 (list x)
  - result2 = result.find('tbody') : table태그의 자식태그 갖고 옴 (list x)

find, select 차이

✓ **find_all(), find()는 중첩 태그에 대한 문법이 없음**

```
<div class="some-class">
        <p>Paragraph 1</p>
        <p>Paragraph 2</p>
        <p>Paragraph 3</p>
</div>
```

- Select의 경우 : 중첩 가능

  Soup.select('div p')

  # 띄어 쓰기로(후손 셀렉트) 중첩 해결

  # 해당 태그의 list 리턴

- Find_all의 경우 : 중첩 불가능

  result = Soup.find('div')

  result.find_all('p')

  # 해당 태그의 list 리턴

데이터를 얻는 다양한 방법

✓ **soup.find_all()** 또는 **soup.select()**

- 해당하는 모든 태그가 담긴 **List를 return**

```
tags = soup.find_all('table')
tag = tags.find('tbody')
# 에러 뜸
```

→ list에는 find / find_all로 태그를 찾지 못함
For문, 인덱싱, 슬라이싱 통해
list에서 태그를 한 개씩 갖고 온 후
데이터를 얻어야 함

실습 해보기!!

실습

최희윤 강사

실습1

✓ **HTML문서 내에 ID가 cook인 태그의 내용을 출력해주세요.**

결과 :

전통적인 요리법이나 양식은 상당한 차이가 있지만, 이탈리아 요리는 다른 국가의 요리 문화에서 다양한 영감을 줄 만큼 다양하고 혁신적인 것으로 평가되고 있다. 각 지방마다 고유의 특색이 있어 그 양식도 다양하지만 크게 북부와 남부로 나눌 수 있다. 다른 나라와 국경을 맞대고 있던 북부 지방은 산업화되어 경제적으로 풍족하고 농업이 발달해 쌀이 풍부해 유제품이 다양한 반면 경제적으로 침체되었던 남부 지방은 올리브와 토마토, 모차렐라 치즈가 유명하고 특별히 해산물을 활용한 요리가 많다. 식재료와 치즈 등의 차이는 파스타의 종류와 소스와 수프 등도 다름을 의미한다.

실습2

✓ **HTML문서의 Table 내에 th와 td에 있는 값들을 크롤링해 아래와 같은 딕셔너리 형태를 만들어 보세요.**

결과 :

[{'이름': '이몽룡', '나이': '34'}, {'이름': '홍길동', '나이': '23'}]

*힌트 : 파이썬 내장함수인 zip 활용해보기

실습3

✓ **HTML문서 내에 모든 A태그에 링크된 페이지에 있는 내용을 읽어 출력해주세요.**

결과 :

크롤링 연습사이트 01-1 페이지입니다.
크롤링 연습사이트 01-2 페이지입니다.
크롤링 연습사이트 01-3 페이지입니다.
크롤링 연습사이트 01-4 페이지입니다.

실습4

✓ **사이트내에 인기검색종목과 주요해외지수를 각각 크롤링하여 종목명과 주가지수를 아래와 같이 리스트로 정리해주세요.**

결과 :

[['써니전자', '5,000'], ['삼성전자', '55,200'], ['안랩', '81,000'], ['케이엠더블..', '57,300'], ['피피아이', '12,600'], ['KT&G', '92,500'], ['삼성전자우', '45,600'], ['대양금속', '10,550'], ['SK하이닉스', '94,700'], ['SK텔레콤', '234,000']]

[['다우산업', '28,647.43'], ['나스닥', '9,015.03'], ['홍콩H', '11,320.56'], ['상해종합', '3,085.20'], ['니케이225', '23,656.62']]

실습5

✓ **사이트내에 인기검색종목과 주요해외지수를 각각 크롤링하여 종목명과 상한, 하한 여부를 아래와 같이 리스트로 정리해주세요.**

결과 :

[['써니전자', '상한'], ['삼성전자', '하한'], ['안랩', '상한'], ['케이엠더블..', '상한'], ['피피아이', '상한'], ['KT&G', '하한'], ['삼성전자우', '상한'], ['대양금속', '하한'], ['SK하이닉스', '상한'], ['SK텔레콤', '하한']]

[['다우산업', '상한'], ['나스닥', '상한'], ['홍콩H', '상한'], ['상해종합', '상한'], ['니케이225', '하한']]

실습6

✓ **사이트내에 인기검색종목과 주요해외지수를 각각 상승인 종목만 크롤링하여 종목명과 주가지수를 아래와 같이 리스트로 정리해주세요.**

결과 :

[['써니전자', '5,000'], ['안랩', '81,000'], ['케이엠더블..', '57,300'], ['피피아이', '12,600'], ['삼성전자우', '45,600'], ['SK하이닉스', '94,700']]

[['다우산업', '28,647.43'], ['나스닥', '9,015.03'], ['홍콩H', '11,320.56'], ['상해종합', '3,085.20']]

실습7

✓ **분양중인 아파트 정보를 크롤링하여 아래와 같이 딕셔너리 형태로 정리 해주세요.**

결과 :

[{'이름': 'H하우스장위', '보증금': '16000', '유형': '아파트', '분양유형': '일반민간임대', '세대수': '분양 134세대', '평형': '45㎡~65㎡'},
{'이름': '고덕리엔파크2단지 장기전세', '보증금': '38400', '유형': '아파트', '분양유형': '장기전세주택', '세대수': '분양 1세대', '평형': '149㎡'},
{'이름': '신정이펜하우스3단지 장기전세', '보증금': '39040', '유형': '아파트', '분양유형': '장기전세주택', '세대수': '분양 1세대', '평형': '148㎡'},
{'이름': '천왕이펜하우스2단지 장기전세', '보증금': '38240', '유형': '아파트', '분양유형': '장기전세주택', '세대수': '분양 1세대', '평형': '142㎡'},
{'이름': '송파파크데일2단지 장기전세', '보증금': '45600', '유형': '아파트', '분양유형': '장기전세주택', '세대수': '분양 1세대', '평형': '150㎡'}]

실습 URL : https://crawlingstudy-dd3c9.web.app/03/

실전 크롤링

최희윤 강사

영화 랭킹

박스오피스

✓ **박스오피스 1~30위 순위의 영화 정보 dictionary로 갖고 오기**

갖고 올 정보 : 순위, 제목, 관객 수

```
[{'순위': '1', '제목': '파일럿', '관객 수': '12만명'},
 {'순위': '2', '제목': '사랑의 하츄핑', '관객 수': '3.5만명'},
 {'순위': '3', '제목': '리볼버', '관객 수': '2.7만명'},
 {'순위': '4', '제목': '슈퍼배드 4', '관객 수': '2.2만명'},
 {'순위': '5', '제목': '데드풀과 울버린', '관객 수': '1.9만명'},
 ...]
```

주가 크롤링_1

https://finance.naver.com/sise/sise_quant.nhn

코스피 코스닥

N	종목명	현재가	전일비	등락률	거래량	거래대금	매수호가	매도호가	시가총액	PER	ROE
1	KODEX 200선물인버스2X	5,270	▼ 400	-7.05%	228,349,079	1,217,092	5,270	5,275	19,293	N/A	N/A
2	KODEX 레버리지	12,680	▲ 845	+7.14%	120,363,769	1,511,682	12,675	12,680	33,830	N/A	N/A
3	삼성중공업	6,970	▲ 1,080	+18.34%	110,900,770	728,045	6,970	6,980	43,911	-3.06	-21.88
4	KODEX 인버스	6,095	▼ 225	-3.56%	55,716,039	341,589	6,095	6,100	10,020	N/A	N/A
5	KODEX 코스닥150 선물인버스	6,220	▲ 20	+0.32%	50,528,871	311,701	6,220	6,225	4,715	N/A	N/A
6	삼성전자	54,500	▲ 3,100	+6.03%	48,787,603	2,629,934	54,500	54,600	3,253,531	17.39	8.69
7	두산인프라코어	6,250	▲ 710	+12.82%	43,248,149	272,059	6,250	6,260	13,010	6.37	11.59
8	KODEX 코스닥150 레버리지	10,010	▼ 80	-0.79%	37,575,643	384,722	10,005	10,010	12,172	N/A	N/A
9	미래산업	83	▲ 2	+2.47%	32,447,838	2,668	82	83	700	-16.60	-12.97
10	문배철강	3,060	▲ 140	+4.79%	29,730,396	96,750	3,060	3,065	627	37.32	2.14
11	삼성 레버리지 WTI 원유 선물 ETN	455	▲ 45	+10.98%	29,475,904	13,511	455	460	933	N/A	N/A
12	태평양물산	2,220	▲ 90	+4.23%	26,576,427	64,174	2,220	2,230	1,108	2,220.00	6.99
13	쌍방울	1,015	▼ 5	-0.49%	22,618,490	24,258	1,015	1,020	1,414	-2.08	-19.68
14	이아이디	235	▼ 7	-2.89%	20,921,183	4,979	234	235	1,289	19.58	0.44
15	신한 레버리지 WTI 원유 선물 ETN(H)	360	▲ 40	+12.50%	20,820,339	7,456	360	365	1,368	N/A	N/A
16	KODEX WTI원유선물(H)	6,095	▲ 325	+5.63%	18,983,196	114,853	6,090	6,095	14,765	N/A	N/A
17	파미셀	22,000	▼ 650	-2.87%	15,429,756	348,822	22,000	22,050	13,191	309.86	8.78
18	한화생명	1,620	▲ 60	+3.85%	15,158,508	24,384	1,620	1,625	14,070	13.97	0.51
19	화인베스틸	2,560	▲ 245	+10.58%	14,705,935	38,214	2,555	2,560	756	-10.08	-7.77
20	팬오션	3,860	▲ 235	+6.48%	14,606,724	56,803	3,855	3,860	20,634	14.46	5.49
21	신성통상	1,450	▼ 50	-3.33%	13,713,658	20,196	1,445	1,450	2,084	-55.77	2.26
22	에이프로젠 KIC	2,960	▼ 240	-7.50%	13,583,125	40,907	2,960	2,965	4,648	105.71	4.70
23	모나미	4,645	▼ 115	-2.42%	12,891,194	61,134	4,640	4,645	878	-30.36	-2.39
24	마니커	978	▼ 16	-1.61%	11,730,931	11,556	977	978	1,550	-4.51	-18.14
25	SK하이닉스	88,700	▲ 5,400	+6.48%	11,589,791	1,007,398	88,600	88,700	645,738	41.43	4.25

1. 품목명과 현재가를 크롤링해주세요.

결과 (순위, 종목명, 현재가)

1 KODEX 200선물인버스2X 5,270
2 KODEX 레버리지 12,680
3 삼성중공업 6,970
4 KODEX 인버스 6,095
5 KODEX 코스닥150선물인버스 6,220
6 삼성전자 54,500
7 두산인프라코어 6,250
8 KODEX 코스닥150 레버리지 10,010
9 미래산업 83
10 문배철강 3,060

주가 크롤링_2

https://finance.naver.com/sise/sise_quant.nhn

코스피 코스닥

N	종목명	현재가	전일비	등락률	거래량	거래대금	매수호가	매도호가	시가총액	PER	ROE
1	KODEX 200선물인버스2X	5,270	▼ 400	-7.05%	228,349,079	1,217,092	5,270	5,275	19,293	N/A	N/A
2	KODEX 레버리지	12,680	▲ 845	+7.14%	120,363,769	1,511,682	12,675	12,680	33,830	N/A	N/A
3	삼성중공업	6,970	▲ 1,080	+18.34%	110,900,770	728,045	6,970	6,980	43,911	-3.06	-21.88
4	KODEX 인버스	6,095	▼ 225	-3.56%	55,716,039	341,589	6,095	6,100	10,020	N/A	N/A
5	KODEX 코스닥150 선물인버스	6,220	▲ 20	+0.32%	50,528,871	311,701	6,220	6,225	4,715	N/A	N/A
6	삼성전자	54,500	▲ 3,100	+6.03%	48,787,603	2,629,934	54,500	54,600	3,253,531	17.39	8.69
7	두산인프라코어	6,250	▲ 710	+12.82%	43,248,149	272,059	6,250	6,260	13,010	6.37	11.59
8	KODEX 코스닥150 레버리지	10,010	▼ 80	-0.79%	37,575,643	384,722	10,005	10,010	12,172	N/A	N/A
9	미래산업	83	▲ 2	+2.47%	32,447,838	2,668	82	83	700	-16.60	-12.97
10	문배철강	3,060	▲ 140	+4.79%	29,730,396	96,750	3,060	3,065	627	37.32	2.14
11	삼성 레버리지 WTI 원유 선물 ETN	455	▲ 45	+10.98%	29,475,904	13,511	455	460	933	N/A	N/A
12	태평양물산	2,220	▲ 90	+4.23%	26,576,427	64,174	2,220	2,230	1,108	2,220.00	6.99
13	쌍방울	1,015	▼ 5	-0.49%	22,618,490	24,258	1,015	1,020	1,414	-2.08	-19.68
14	이아이디	235	▼ 7	-2.89%	20,921,183	4,979	234	235	1,289	19.58	0.44
15	신한 레버리지 WTI 원유 선물 ETN(H)	360	▲ 40	+12.50%	20,820,339	7,456	360	365	1,368	N/A	N/A
16	KODEX WTI원유선물(H)	6,095	▲ 325	+5.63%	18,983,196	114,853	6,090	6,095	14,765	N/A	N/A
17	파미셀	22,000	▼ 650	-2.87%	15,429,756	348,822	22,000	22,050	13,191	309.86	8.78
18	한화생명	1,620	▲ 60	+3.85%	15,158,508	24,384	1,620	1,625	14,070	13.97	0.51
19	화인베스틸	2,560	▲ 245	+10.58%	14,705,935	38,214	2,555	2,560	756	-10.08	-7.77
20	팬오션	3,860	▲ 235	+6.48%	14,606,724	56,803	3,855	3,860	20,634	14.46	5.49
21	신성통상	1,450	▼ 50	-3.33%	13,713,658	20,196	1,445	1,450	2,084	-55.77	2.26
22	에이프로젠 KIC	2,960	▼ 240	-7.50%	13,583,125	40,907	2,960	2,965	4,648	105.71	4.70
23	모나미	4,645	▼ 115	-2.42%	12,891,194	61,134	4,640	4,645	878	-30.36	-2.39
24	마니커	978	▼ 16	-1.61%	11,730,931	11,556	977	978	1,550	-4.51	-18.14
25	SK하이닉스	88,700	▲ 5,400	+6.48%	11,589,791	1,007,398	88,600	88,700	645,738	41.43	4.25

2. 전일대비 상승한 항목만 품목명, 현재가, 전일비를 크롤링해주세요.

결과

1 KODEX 200선물인버스2X 5,270 400
4 KODEX 인버스 6,095 225
8 KODEX 코스닥150 레버리지 10,010 80
13 쌍방울 1,015 5
14 이아이디 235 7
17 파미셀 22,000 650
21 신성통상 1,450 50
22 에이프로젠 KIC 2,960 240
23 모나미 4,645 115
24 마니커 978 16
30 삼성 인버스 2X WTI원유 선물 ETN 2,765 365
33 엔케이 1,165 30

뉴스 크롤링1

https://news.naver.com/main/list.naver?mode=LPOD&mid=sec&oid=032

✓ **네이버 뉴스에서 첫 페이지의 모든 기사 제목과 본문을 크롤링 하여 딕셔너리로 갖고 오기**

(실습 예제 3번과 같은 방식으로 실행하면 됩니다 ☺)

예시:

```
[{
        "title" : "제목 삽입",
        "body" : "본문 삽입"
},
{
        "title" : "제목 삽입",
        "body" : "본문 삽입"
}, ...
]
```

뉴스 크롤링1

https://news.naver.com/main/list.naver?mode=LPOD&mid=sec&oid=032

✓ 네이버 뉴스에서 첫 페이지의 모든 기사 제목과 본문을 크롤링 하여 딕셔너리로 갖고 오기

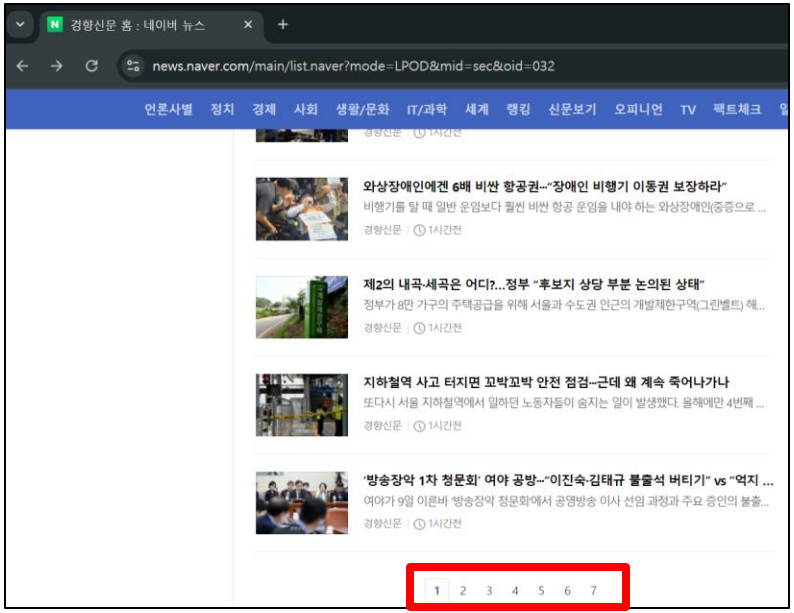
결과

```
[{'title': '[단독]“김구는 테러리스트”…이승만 미화 다큐…‘광복의 역사’가 흔들린다',
  'body': '독립기념관장에 뉴라이트 계열로 지목된 인사가 임명돼 광복회를 비롯한 독립운동 유관단체들이 반발하는 등 역사논쟁이 정국의 
{'title': '서울 12년만에 그린벨트 해제… 정부 수도권에 ‘21만호+a’ 주택 공급',
  'body': '정부가 급등하는 서울 집값을 잡기 위해 12년만에 서울의 그린벨트를 해제하고 수도권에 8만가구를 공급할 수 있는 신규 택지 후
{'title': '광복절 특사 명단에 김경수 전 지사 포함…윤 대통령 최종 결단할까',
  'body': '법무부가 윤석열 대통령에게 제출할 8·15 광복절 특별사면 및 복권 대상자에 김경수 전 경남지사가 포함됐다. 김 전 지사의 복권
{'title': '12년 만에 서울 그린벨트 전격 해제',
  'body': '정부가 급등하는 서울 집값을 잡기 위해 12년 만에 서울의 그린벨트를 해제하고 수도권에 8만가구를 공급할 수 있는 신규 택지 
{'title': '권익위 국장 사망에 국민의힘 “야당, 또 정쟁 소재 삼으려 해”',
  'body': '윤석열 대통령의 배우자 김건희 여사의 명품 가방 수수 사건을 맡았던 국민권익위원회(권익위) 간부가 사망한 것과 관련해 국민의
{'title': '큐텐, 티몬·위메프 합병법인 설립해 정상화 시도…실현 가능성 낮을 듯',
  'body': '큐텐이 티몬·위메프 미정산 사태를 해결하기 위해 신규법인을 세워 양사를 합병하겠다는 계획을 내놨다. 티몬과 위메프를 합병하
{'title': '정부 “8·8 부동산 대책 밀착 관리…지방 미분양 해소도 지원”',
  'body': '정부가 8일 발표한 ‘국민 주거안정을 위한 주택공급 확대방안’이 원활히 실행될 수 있도록 밀착 관리하겠다고 밝혔다. 내년 말까
{'title': '배드민턴협회 “임원진 비즈니스석 탑승은 사실무근”…항공권 내역 공개',
  'body': '대한배드민턴협회가 2024 파리올림픽 금메달리스트 안세영(22)의 작심 발언 이후 추가로 조명되고 있는 ‘임원진 비즈니스석 탑승
{'title': '직원 월급 강제 공제해 전주혜 전 의원 후원한 강동농협 조합장 검찰 송치',
  'body': '직원 동의 없이 월급에서 정치 후원금을 공제해 전주혜 국민의힘 전 의원의 후원회에 전달한 혐의로 서울 강동농협 조합장 등 2명
```

뉴스 크롤링2

https://news.naver.com/main/list.naver?mode=LPOD&mid=sec&oid=032

✓ 네이버 뉴스에서 **모든 페이지**의 모든 기사 제목과 본문을 크롤링 하여 딕셔너리로 갖고 오기



예시:

```
[{
        "page" : "페이지 삽입",
        "title" : "제목 삽입",
        "body" : "본문 삽입"
},
{
        "page" : "페이지 삽입",
        "title" : "제목 삽입",
        "body" : "본문 삽입"
}, …
]
```

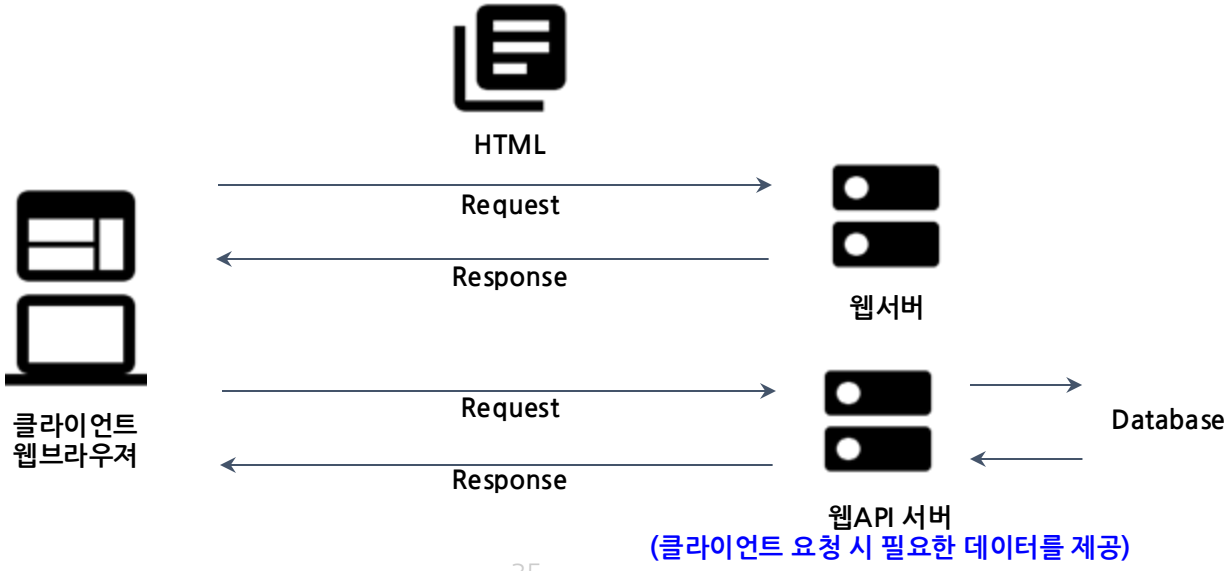
비동기 방식 크롤링

최희윤 강사

비동기 웹 (Asynchronous Web)

웹 페이지가 전체를 다시 로드하지 않고도 서버와 데이터를 주고받아 페이지의 일부만 업데이트할 수 있는 웹

Ex : 검색 자동완성 기능, 실시간 채팅 앱, 주식 가격 업데이트, 이메일 클라이언트 등



동기 VS 비동기

✓ **동기 웹**

웹 상단의 카페, 블로그, 이미지 등의 카테고리를 클릭하며 화면이 새로 랜더링 되는 걸 확인 해보기
https://search.naver.com/search.naver?ssc=tab.nx.all&where=nexearch&sm=tab_jum&query=%EB%84%A4%EC%9D%B4%EB%B2%84+%EB%89%B4%EC%8A%A4

✓ **비동기 웹**

각자의 트위터에서 스크롤을 내리며 화면 전체가 다시 랜더링 되는지, 부분만 랜더링 되는지 확인 해보기
https://x.com/Daisy2170290932/communities/explore

비동기 방식 웹사이트

✓ **실행 해보기**

```
import requests
from bs4 import BeautifulSoup
response = requests.get(" https://crawlingstudy-dd3c9.web.app/04")
print(response.text)
```

비동기 방식 웹사이트

결과 :

```
<!doctype html>
<html>
<head>
    <meta charset="utf-8">
    <script src="https://code.jquery.com/jquery-3.4.1.js" integrity="sha256-WpOohJOqMqqyKL9FccASB9O0KwACQJpFTUBLTYOVvVU=" crossorigin="anonymous"></script>
    <script>
        .....
    </script>
    <title>Demo</title>
</head>
<body>
    <div id="post" style="width:500px;">
    </div>
</body>
</html>
```

데이터가 없음

실습 URL : https://crawlingstudy-dd3c9.web.app/04/

비동기 방식 웹사이트

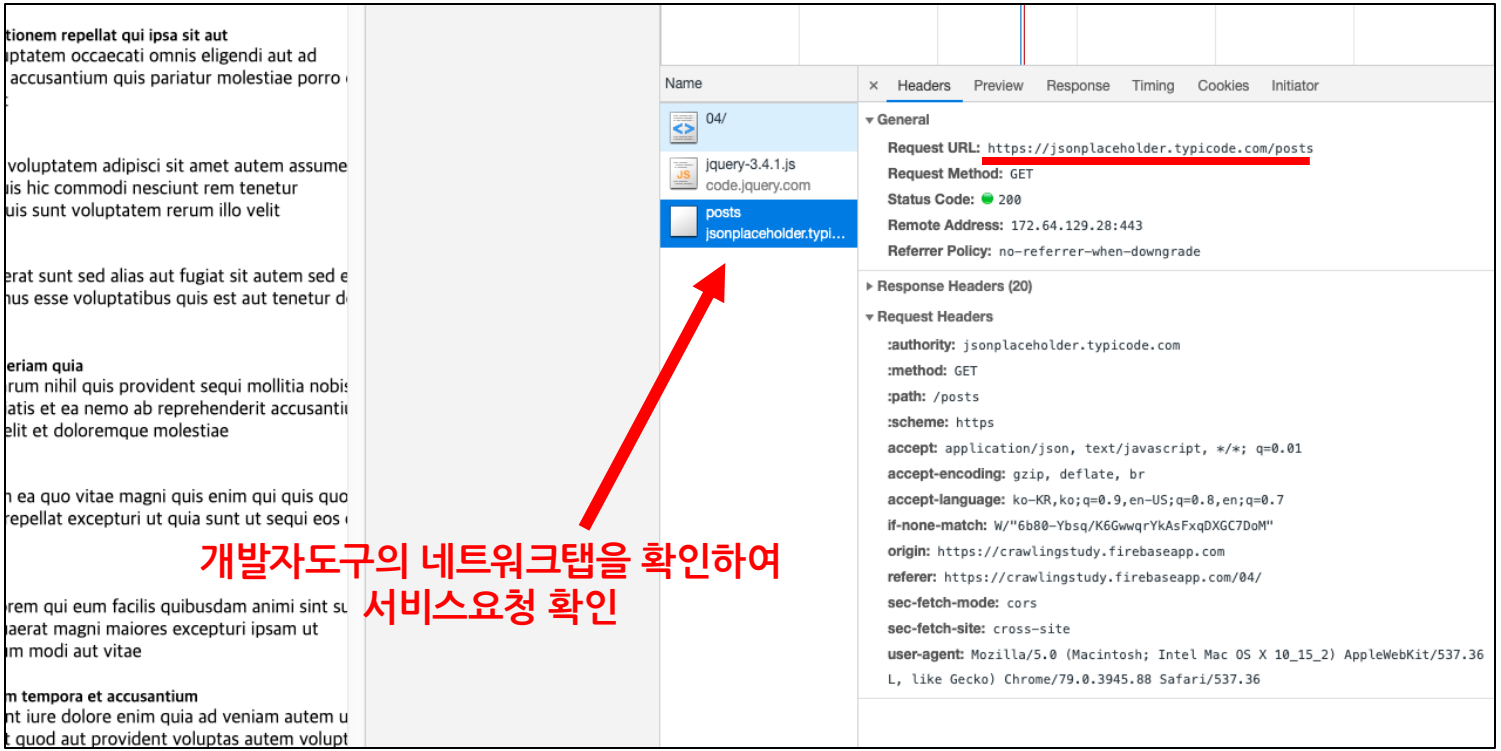
해결 방법

Selenium을 활용하여 데이터를 갖고 올 수 있음

단점

Selenium은 안정성이 떨어지고 bs4에 비해 느림

비동기 방식 웹사이트


tionem repellat qui ipsa sit aut
uptatem occaecati omnis eligendi aut ad
accusantium quis pariatur molestiae porro
voluptatem adipisci sit amet autem assume
is hic commodi nesciunt rem tenetur
uis sunt voluptatem rerum illo velit
erat sunt sed alias aut fugiat sit autem sed e
us esse voluptatibus quis est aut tenetur d
eriam quia
rum nihil quis provident sequi mollitia nobis
atis et ea nemo ab reprehenderit accusantiu
elit et doloremque molestiae
ea quo vitae magni quis enim qui quis quo
repellat excepturi ut quia sunt ut sequi eos
rem qui eum facilis quibusdam animi sint su
aerat magni maiores excepturi ipsam ut
m modi aut vitae
m tempora et accusantium
nt iure dolore enim quia ad veniam autem u
quod aut provident voluptas autem volupt
개발자도구의 네트워크탭을 확인하여
서비스요청 확인
Name
04/
jquery-3.4.1.js
code.jquery.com
posts
jsonplaceholder.typi...
Headers
Preview
Response
Timing
Cookies
Initiator
General
Request URL: https://jsonplaceholder.typicode.com/posts
Request Method: GET
Status Code: 200
Remote Address: 172.64.129.28:443
Referrer Policy: no-referrer-when-downgrade
Response Headers (20)
Request Headers
:authority: jsonplaceholder.typicode.com
:method: GET
:path: /posts
:scheme: https
accept: application/json, text/javascript, */*; q=0.01
accept-encoding: gzip, deflate, br
accept-language: ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
if-none-match: W/"6b80-Ybsq/K6GwwqrYkAsFxqDXGC7DoM"
origin: https://crawlingstudy.firebaseapp.com
referer: https://crawlingstudy.firebaseapp.com/04/
sec-fetch-mode: cors
sec-fetch-site: cross-site
user-agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_2) AppleWebKit/537.36
L, like Gecko) Chrome/79.0.3945.88 Safari/537.36

비동기 방식 웹사이트

```
import requests
from bs4 import BeautifulSoup
response = requests.get("https://jsonplaceholder.typicode.com/posts")
print(response.text)
```

결과


```
[
 {
  "userId": 1,
  "id": 1,
  "title": "sunt aut facere repellat provident occaecati excepturi optio reprehenderit",
  "body": "quia et suscipit\nsuscipit recusandae consequuntur expedita et cum\nre
   prehenderit molestiae ut ut quas totam\nnostrum rerum est autem sunt rem eveniet architecto"
 },
 {
  "userId": 1,
  "id": 2,
  "title": "qui est esse",
  "body": "est rerum tempore vitae\nsequi sint nihil reprehenderit dolor beatae ea dolores neque\
   nfugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis\nqui aperiam non debitis possimus qui neque nisi nulla"
 },
 {
  "userId": 1,
  "id": 3,
  "title": "ea molestias quasi exercitationem repellat qui ipsa sit aut",
  "body": "et iusto sed quo iure\nvoluptatem occaecati omnis eligendi aut ad\nvoluptatem doloribus
   vel accusantium quis pariatur\nmolestiae porro eius odio et labore et velit aut"
 }
```


결과가 json형식으로 나옴

JSON

JSON은 자바스크립트 문법을 따르는 문자 기반의 데이터 포맷

파이썬의 딕셔너리(사전) 형태와 비슷

결과


```
[
 {
  "userId": 1,
  "id": 1,
  "title": "sunt aut facere repellat provident occaecati excepturi optio reprehenderit",
  "body": "quia et suscipit\nsuscipit recusandae consequuntur expedita et cum\nre
   prehenderit molestiae ut ut quas totam\nnostrum rerum est autem sunt rem
   eveniet architecto"
 },
 {
  "userId": 1,
  "id": 2,
  "title": "qui est esse",
  "body": "est rerum tempore vitae\nsequi sint nihil reprehenderit dolor beatae ea
   dolores neque\nfugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis\
   nqui aperiam non debitis possimus qui neque nisi nulla"
 },
```


JSON 형태 데이터를 파이썬 딕셔너리로 바꾸기

Str형식으로는 데이터를 가져다 쓸 수 없기 때문

```
import json
json.loads(JSON형식의 텍스트)
```

JSON형태 저장

Json.loads() : text를 파이썬의 객체로 변형

Json.dump() - 파이썬 객체를 json 파일로저장

Json.load() - json 파일읽어와서 파이썬 객체로 변형

```
import requests
from bs4 import BeautifulSoup
response = requests.get("https://jsonplaceholder.typicode.com/posts")
result_dic = json.loads(response.text)

with open("data.json", "w") as json_file:
    json.dump(result_dic, json_file)
```

```
import json

with open("data.json", "r") as json_file:
    result = json.load(json_file)

print(result)
```

비동기 사이트 실습

최희윤 강사

실습 1

✓ 아래 사이트를 크롤링하여 아래와 같이 각각 글에 id와 title 그리고 글마다 코멘트내용을 리스트형식으로 담고 최종 json 파일 형태로 저장해보세요.

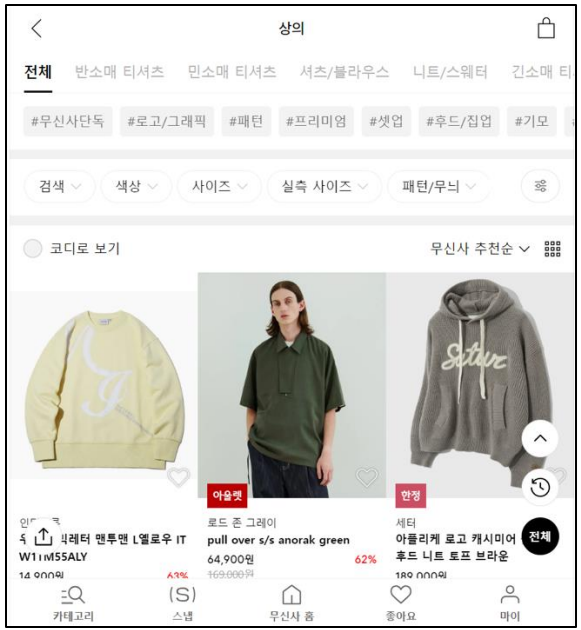
결과 :

```
[{'id': 1,
  'title': 'sunt aut facere repellat provident occaecati excepturi optio reprehenderit',
  'comment': [
                'laudantium enim quasi est quidem magnam voluptate ipsam eos\ntempora quo
                necessitatibus\ndolor quam autem quasi\nreiciendis et nam sapiente accusantium',
                'est natus enim nihil est dolore omnis voluptatem numquam\net omnis occaecati quod
                ullam at\nvoluptatem error expedita pariatur\nnihil sint nostrum voluptatem reiciendis et',
.....]},
{'id': 2
.....
]
```


실습 URL : https://crawlingstudy-dd3c9.web.app/05/

실습 2-1

✓ 아래 URL 첫 페이지 제품들의 브랜드명, 제품명, 정가, 할인가를 모두 크롤링하여 json 파일 형태로 저장해주세요



예시:

[{

“brandName” : “브랜드이름”,

“goodsName” : “제품명“,

“normalPrice”: “ 정가”,

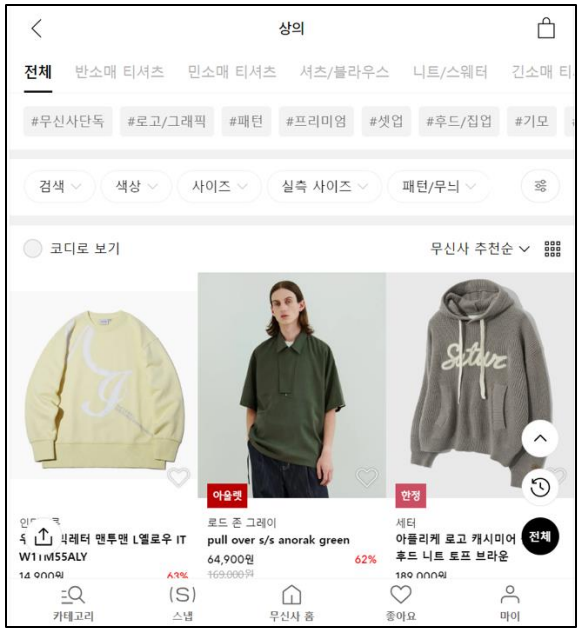
“price“ : “할인가”

}]

실습 URL : https://www.musinsa.com/categories/item/001?device=mw

실습 2-2

✓ 아래 URL 다섯 페이지 제품들의 브랜드명, 제품명, 정가, 할인가를 모두 크롤링하여 json 파일 형태로 저장해주세요



예시:

[{

“page” : “페이지”,

“brandName” : “브랜드이름”,

“goodsName” : “제품명“,

“normalPrice”: “ 정가”,

“price“ : “할인가”

}]

실습 URL : https://www.musinsa.com/categories/item/001?device=mw

실습 3-1

✓ 아래 URL 의 다음 뉴스기사의 기사제목, 기사내용, 사람들반응을 모두 크롤링하여 json 파일 형태로 저장해주세요

예시:

```
[{
    “title” :  “제목 삽입”,
    “body” : “본문 삽입“,
    “reactions”: {
        “react1”: 0,
        “react2”:0, …
    }]
```

실습 URL : https://v.daum.net/v/20240811221440047

실습 3-2

✓ 아래 URL 의 모든 페이지의 뉴스기사 기사제목, 기사내용, 사람들반응을 크롤링하여 json 파일 형태로 저장해주세요

예시:

```
[{
    “page” : 페이지 삽입,
    “title” :  “제목 삽입”,
    “body” : “본문 삽입“,
    “reactions”: {
        “react1”: 0,
        “react2”:0, …
    }]
```

실습 URL : https://news.daum.net/breakingnews/society/affair?page=1&regDate=20240811

실습 4

✓ 아래 URL 의 뉴스기사 본문에 기사제목, 기사내용, 사람들반응, 댓글을 모두 크롤링하여 json 파일 형태로 저장해주세요

실습 URL : https://n.news.naver.com/mnews/article/032/0003312694

실습 4 - 결과 형식

✓ 아래 URL 의 뉴스기사 본문에 기사제목, 기사내용, 사람들반응, 댓글을 모두 크롤링하여 json 파일 형태로 저장해주세요

예시:

```
[{
        “title” :  “제목 삽입”,
        “body” : “본문 삽입“,
        “reactions”: {
                “react1”: 0,
                “react2”:0, …
        },
        “comments” : [
                ‘댓글1’,
                ‘댓글2’, …
         ]
}]
```

각 데이터들을 딕셔너리 형태로 담기

그 중 reactions는 각 반응들을 딕셔너리로 담고

댓글은 리스트형식으로 담기

실습 URL : https://n.news.naver.com/mnews/article/032/0003312694