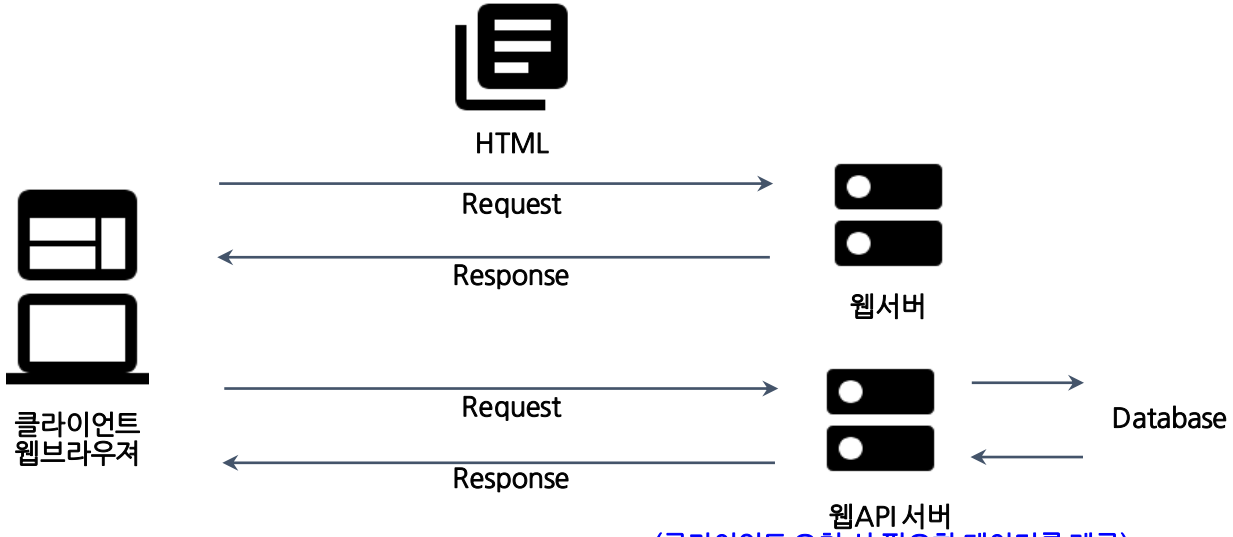
비동기 방식 크롤링

김지성 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

네이버 뉴스 : 네이버 검색

✓ **비동기 웹**

각자의 트위터에서 스크롤을 내리며 화면 전체가 다시 랜더링 되는지, 부분만 랜더링 되는지 확인 해보기

이솝 : 네이버 가격비교

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

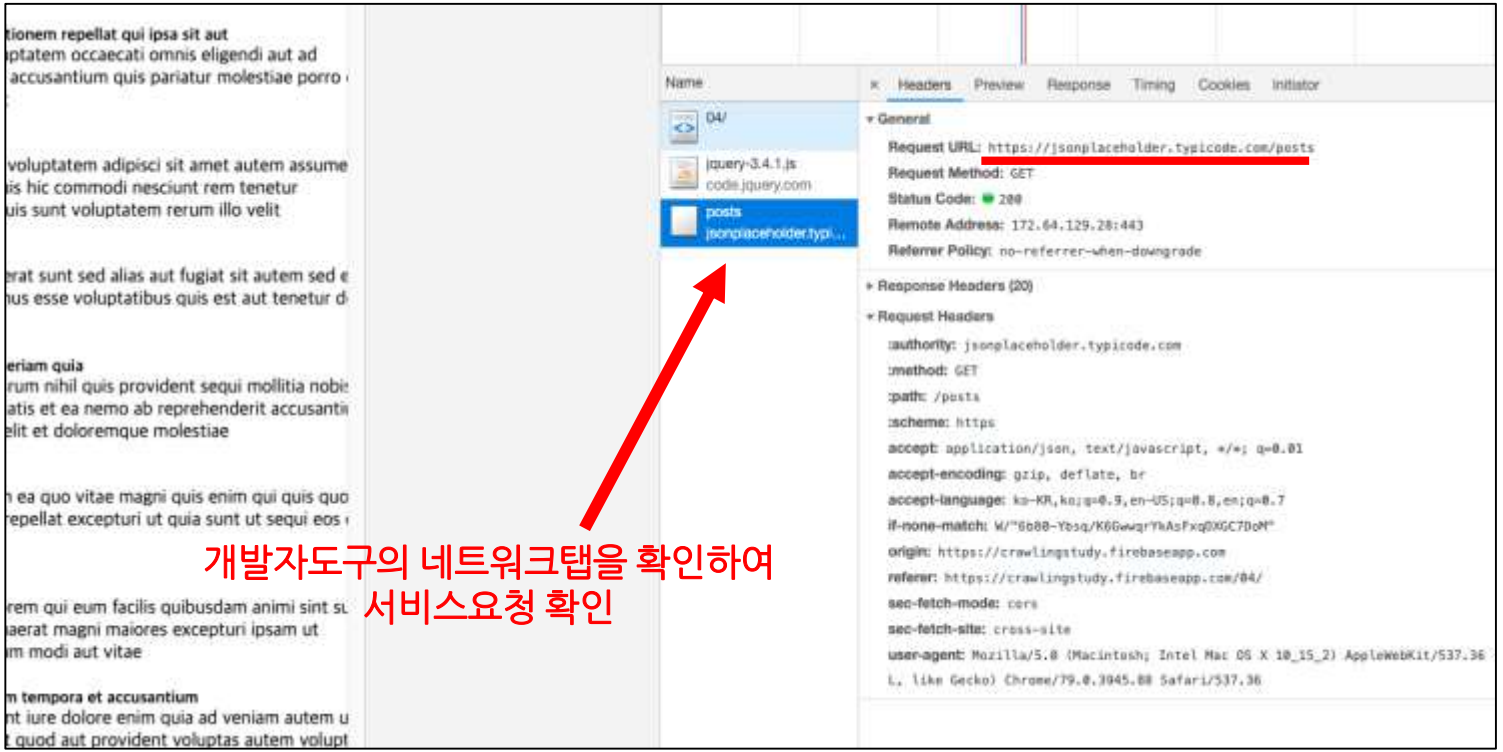
해결 방법

Selenium을 활용하여 데이터를 갖고 올 수 있음

단점

Selenium은 안정성이 떨어지고 bs4에 비해 느림

비동기 방식 웹사이트



개발자도구의 네트워크탭을 확인하여
서비스요청 확인

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

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

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

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)

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

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

김지성 강사

실습 1

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

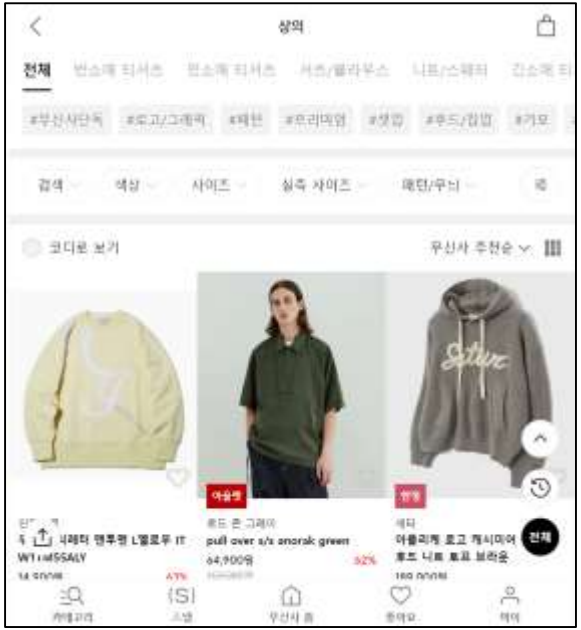
결과 :

```
[{'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



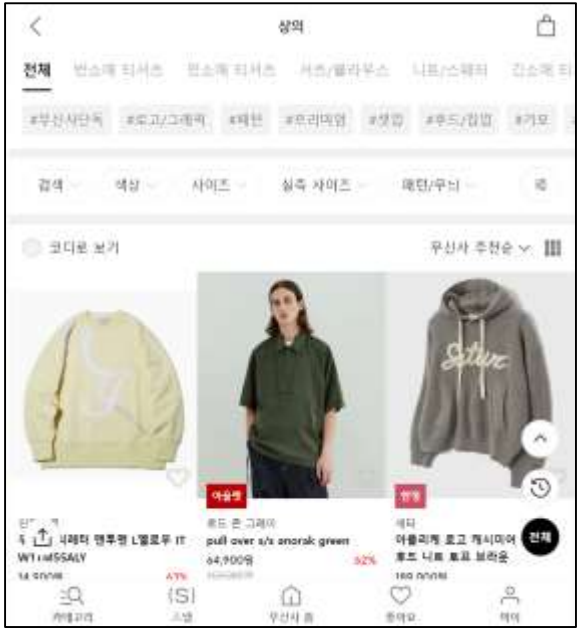
예시:

```
[{
        “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**, **할인가**를 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

✓ 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 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