웹크롤링(Web Crawaling)1

# requests

웹사이트의 정보를 가져오기 위한 파이썬 라이브러리

## 설치

|  | pip3 install requests |
| --- | --- |

| **형식** | import requests  response = requests.get('https://www.naver.com/')  print(response.status\_code) # 응답코드를 출력  print(response.text) # HTML 코드를 출력 | |
| --- | --- | --- |

파라미터를 전달해야 할 경우 JSON형식으로 작성후 추가

| **형식** | param = {  'pageNo' : 1,  'rangeType' : 'ALL',  'orderBy' : 'sim',  'keyword' : '파이썬'  }  response = requests.get('https://section.blog.naver.com/Search/Post.nhn', params=params) | |
| --- | --- | --- |

###### 

# BeautifulSoup

HTML정보로 부터 원하는 데이터를 가져오기 쉽게, 비슷한 분류의 데이터별로 나누어주는(parsing) 파이썬 라이브러리

## 설치

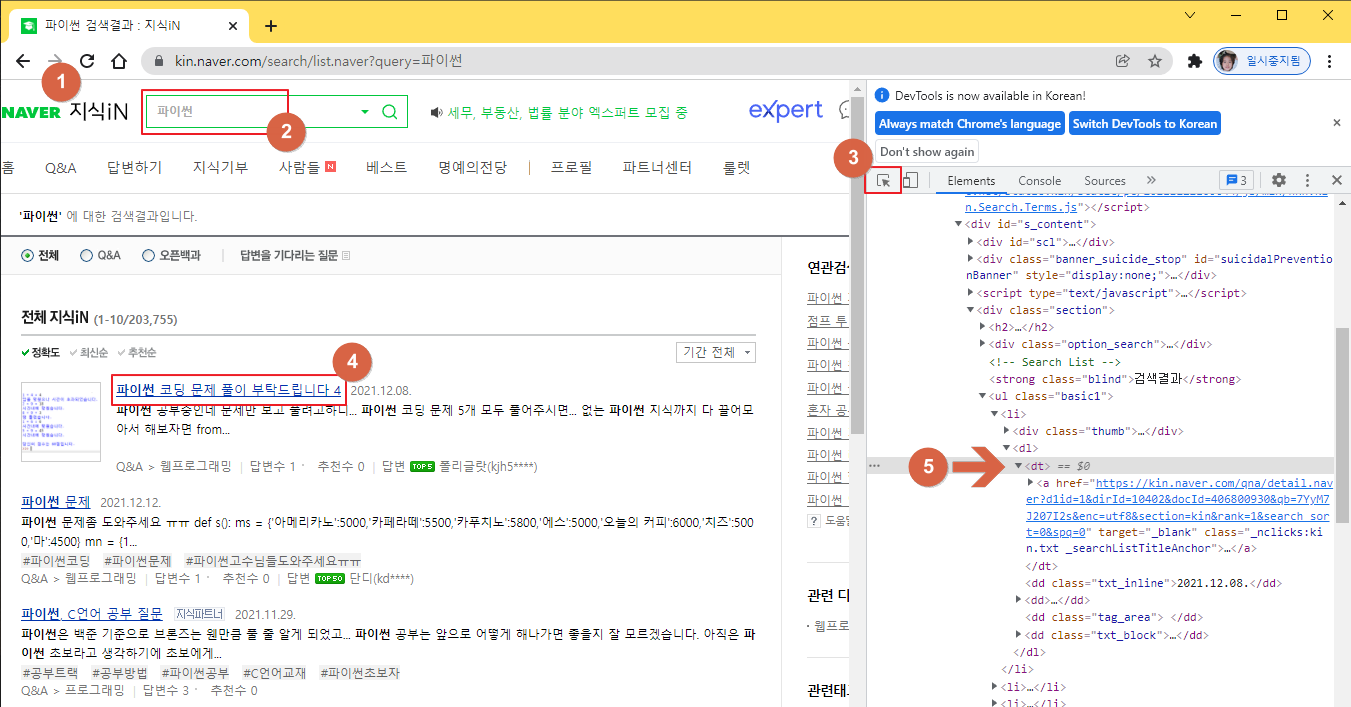
|  | pip3 install beautifulsoup4 |
| --- | --- |

| **형식** | import requests  from bs4 import BeautifulSoup  url = 'http://daum.net/'  response = requests.get(url)  if response.status\_code == 200:  html = response.text  soup = BeautifulSoup(html, 'html.parser')  print(soup)  else :  print(response.status\_code) | |
| --- | --- | --- |

from bs4 import BeautifulSoup 에러날때 [참조](https://python-forum.io/thread-11568.html) => python **-**m pip install bs4

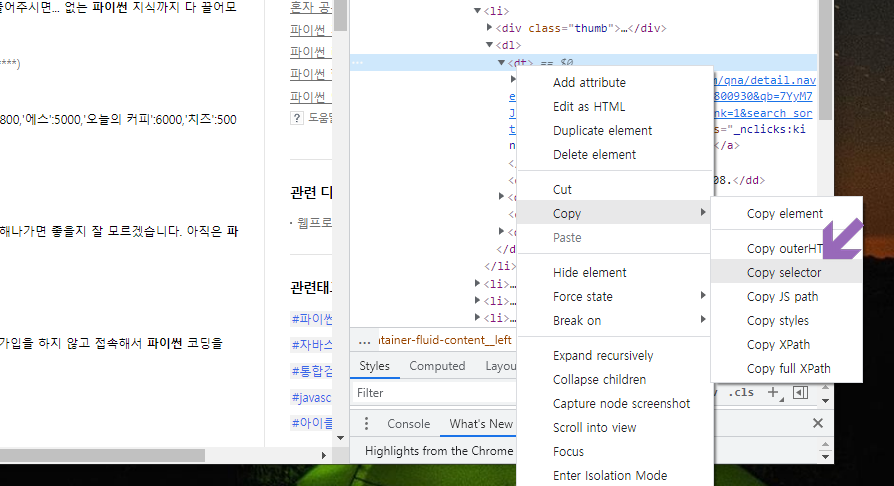
# 크롤링 실습하기

먼저 네이버 메인에서 지식in으로 이동한 후 파이썬을 검색한다.



검색 결과에서 그림의 순서대로 개발자도구의 인스펙트(Inspect)를 이용해서 첫번째 항목을 선택한다. 그러면 Elements 항목에서 <dt> 태그가 보일것이다.

이 부분을 우클릭한다.



Copy > Copy selector를 찾아 클릭한다.

그러면 CSS선택자가 복사된다.

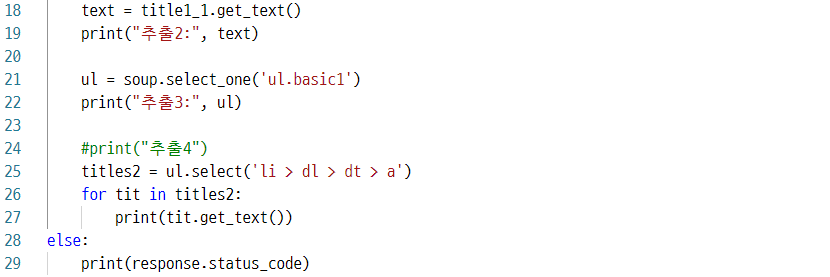
|  | **크롬**에서의 복사 결과  #s\_content > div.section > ul > li:nth-child(1) > dl > dt  **파이어폭스**에서의 복사 결과  .basic1 > li:nth-child(1) > dl:nth-child(2) > dt:nth-child(1) |
| --- | --- |

웹브라우저에 따라 복사된 결과는 조금 다르지만 크롤링한 결과는 동일하다.

### 예제] 19webCrawling01.py



네모 부분은 복사해서 작성하세요.



### 예제] 19webCrawling02.py

<https://www.koreabaseball.com/Record/Player/HitterBasic/BasicOld.aspx?sort=HRA_RT>





