[LAB-13] 웹 페이지 데이터 수집하기

ITWILL 이광호강사

#01. HTML 데이터 추출

라이브러리 참조

```
import requests
from bs4 import BeautifulSoup
```

웹 페이지의 모든 소스코드 가져오기

```
# 세션 객체 생성
with requests.Session() as session:
    # 세션 객체에 웹 브라우저 정보(UserAgent) 주입
    session.headers.update({
        "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/1
    })

    url = "https://data.hossam.kr/py/sample.html"    # 요청할 데이터의 URL
    r = session.get(url)            # 요청 결과 받아오기 --> HTTP GET 요청

    # HTTP 상태값이 200이 아닌 경우는 강제로 에러를 발생시켜서 코드의 진행을 중단시킴
    if r.status_code != 200:
        msg = "[%d Error] %s 에러가 발생함" % (r.status_code, r.reason)
        raise Exception(msg)

    print(r)
```

```
<Response [200]>
```

수신 결과 확인

```
# 수신된 데이터의 인코딩 설정 (한글이 깨진다면 euc-kr로 변경)
r.encoding = "utf-8"

# 수신된 결과 확인 --> 웹에서 가져온 모든 데이터는 문자열 형식임
print(type(r.text))
r.text
```

```
<class 'str'>



'<!DOCTYPE html><html lang="ko"><head><meta charset="UTF-8"><meta name="viewport" content="width=device-width, i
```

응답 결과에 대한 beautifulsoup 객체 생성

```
soup = BeautifulSoup(r.text)
print(type(soup))
soup
```

```
<class 'bs4.BeautifulSoup'>
```

```
<!DOCTYPE html>
<html lang="ko"><head><meta charset="utf-8"/><meta content="width=device-width, initial-scale=1.0" name="viewpor
```

HTML 태그에 의한 데이터 추출

`<h1>` 태그를 갖는 요소에 접근한다.

```
myselect = soup.select("h1")
print(type(myselect))
myselect
```

```
<class 'bs4.element.ResultSet'>
```

```
[<h1>Hello World</h1>]
```

리턴타입이 항상 리스트이므로 리스트의 원소에 접근하여 HTML 태그 객체를 추출한다.

```
mytag = myselect[0]
print(type(mytag))
mytag
```

```
<class 'bs4.element.Tag'>
```

```
<h1>Hello World</h1>
```

추출한 태그에서 내용만 추출

```
mytext = mytag.text.strip()
mytext
```

```
'Hello World'
```

class에 의한 데이터 추출

```
myselect = soup.select(".myclass")
myselect
```

```
[<li class="myclass">연산자</li>,
 <li class="myclass">데이터 전처리</li>,
 <ol class="myclass"><li>기초통계</li><li>데이터 시각화</li></ol>]
```

복수 요소이므로 반복 처리

```
for i, v in enumerate(myselect):
    # 추출한 요소가 하위 태그를 포함하는 경우 그 안의 텍스트만 일괄 추출
    print("%d번째 요소 : %s" % (i, v.text.strip()))
```

```
0번째 요소 : 연산자
1번째 요소 : 데이터 전처리
2번째 요소 : 기초통계데이터 시각화
```

특정 HTML 태그 객체의 하위 요소 추출하기

`select()` 메서드로 추출한 요소를 활용하여 그 하위요소를 추가적으로 추출할 수 있다.

```
myli = myselect[2].select("li")
myli
```

```
[<li>기초통계</li>, <li>데이터 시각화</li>]
```

```
for i in myli:
    print(i.text.strip())
```

```
기초통계
데이터 시각화
```

id에 의한 추출

id 속성은 페이지 내에서 고유한 요소를 의미

정상적인 경우라면 `id` 값은 해당 웹페이지 안에 단 하나만 존재하기 때문에 반복문을 적용할 필요는 없다.

```
myselect = soup.select("#myid")
myselect
```

```
[<h2 id="myid">Python</h2>]
```

```
print(myselect[0].text.strip())
```

```
Python
```

여러 요소에 동시 접근하기

다양한 선택자를 콤마(,)로 구분하여 지정한다.

```
items = soup.select("#myid, .myclass")
items
```

```
[<h2 id="myid">Python</h2>,
 <li class="myclass">연산자</li>,
 <li class="myclass">데이터 전처리</li>,
 <ol class="myclass"><li>기초통계</li><li>데이터 시각화</li></ol>]
```

```
for i in items:
    print(i.text)
```

```
Python
연산자
데이터 전처리
기초통계데이터 시각화
```

복합 선택자

자식 선택자

```
soup.select(".syllabus > .myclass")
```

```
[<li class="myclass">연산자</li>]
```

자손 선택자

```
soup.select(".part1 .myclass")
```

```
[<li class="myclass">연산자</li>]
```

속성 선택자

href 속성을 갖는 요소 추출하기

```
myselect = soup.select("a[href]")
myselect
```

```
[<a href="#">link1</a>, <a href="https://www.naver.com">link2</a>]
```

href 속성에 적용되어 있는 데이터 추출

```
# 속성값은 각 태그요소의 attrs 프로퍼티로 접근 가능 --> dict 형태
for i, v in enumerate(myselect):
    print("----[%d]----" % i)
    print(v.attrs)

    # 딕셔너리에 대한 in 연산자는 key의 존재 여부를 판별
    if "href" in v.attrs:
        print("%d번째의 href속성값 : %s" % (i, v.attrs["href"]))
```

```
----[0]----
{'href': '#'}
0번째의 href속성값 : #
----[1]----
{'href': 'https://www.naver.com'}
1번째의 href속성값 : https://www.naver.com
```