6. 웹 스크래핑 사례연구

6.1. 다음 뉴스

6.1.1. 검색결과 가져오기

다음(http://daum.net)에서 "인공지능"으로 검색을 한다.

검색 결과에서 '뉴스', '최신순'을 클릭한다.

이제 해당 검색결과의 URL을 `requests`로 가져온다.

```
import requests
res = requests.get('http://search.daum.net/search?w=news&cluster=n&q=%EC%9D%B8%EA%B3%B5%EC%A7%80%EB%8A%A5&sort=recency')
```

6.1.2. 기사 링크 모으기

다음 뉴스 검색 결과에서 기사 제목을 클릭하면 언론사 홈페이지로 이동한다. 언론사마다 홈페이지 디자인이 다르기 때문에 자동으로 기사 수집이 어렵다. 대신 주요 언론사의 기사는 기사 제목 옆에 "다음 뉴스"를 클릭하면 다음 내에서 확인할 수 있다.

```
import lxml.html
root = lxml.html.fromstring(res.text)

urls = []
for link in root.cssselect('a.f_nb'):
    urls.append(link.attrib['href'])
```

6.1.3. 기사 본문 수집하기

다음 뉴스의 기사 페이지에서 본문은 다음과 같이 수집한다.

```
articles = []
for u in urls:
    if not u.startswith('http'):
        continue
    res = requests.get(u)
    root = lxml.html.fromstring(res.text)
    body = root.cssselect('.article_view').pop()
    content = body.text_content()
    articles.append(content)
```

6.2. 네이버 카페

이번 강의에서는 네이버 카페의 게시판을 크롤링 해보자.

이번 예시에서는 의자 관련 카페인 '의자를 사용하는 사람들의 모임 - 의사모' 를 크롤링 해보자.

(http://cafe.naver.com/duoin)

많은 카페의 경우, 독립적으로 게시글에 접근하면 비회원에게 게시물을 공개하지 않는다.

하지만, 이러한 비공개 게시글도 네이버 카페 검색으로 유입한다면 게시글을 열람할 수 있다.

이는 게시글을 올릴 때 게시자가 카페 회원에게만 공개라는 항목을 체크하지만 동시에 검색 유입 허용을 체크했기 때문에 발생하는 현상이다.

따라서, 이번 시간에는 이러한 네이버 카페 게시글의 특징을 이용해 크롤링을 해보자.

먼저 필요한 패키지를 불러온다.

```
import requests
import lxml.html
from urllib.parse import urljoin
```

네이버 카페는 페이지가 복잡하여 조금 더 크롤링하기 수월한 모바일 주소로 접속한다.

```
'https://m.cafe.naver.com/ArticleList.nhn?search.clubid=19773565&search.menuid=142&search.boardtype=L'
```

모바일 주소는 앞에 m 을 붙이면 된다.

모바일 버전의 의사모 게시판 주소를 `url` 이라는 변수에 저장한다.

```
url = 'https://m.cafe.naver.com/ArticleList.nhn?search.clubid=19773565&search.menuid=142&search.boardtype=L'
```

후에 크롤링 결과를 확인하면, 최대 20개의 게시물만 크롤링해온다.

더 많은 양의 게시물을 가져오고 싶다면 `url` 주소에 페이지를 다르게 해서 가져오는 방법이 있다.

`url`에 `param`이라는 변수에 저장하면 다른 페이지의 정보도 가져올 수 있다.

```
naver_param = {'search.page': 1}
```

현재 있는 페이지가 1 페이지이다. `search.page` 의 숫자를 다르게하여 다른 페이지의 게시물을 가져올 수 있다.

`requests.get()`에 url 주소와 `param`이라는 옵션을 넘겨주면 된다.

```
res = requests.get(url, param=naver_param)
```

6.2.1. 게시판 url 가져오기

이제 게시판의 각 게시물의 url 주소를 가져와보자.

```
elements = lxml.html.fromstring(res.text)
```

요소검사를 해서 게시글 요소를 찾아보자.

`articleListArea`안에 게시물들이 들어있는 것을 확인할 수 있다.

각 게시물이 `li class "board-box"`라는 태그 안에 `class="txt_area"` 안에 url 이 들어있는 것을 확인할 수 있다.

url 이 들어있는 부분을 선택하도록 하자.

```
postings = elements.cssselect('li.board_box .txt_area')
```

`postings`에 있는 각 항목의 `href` 에 url 주소가 저장되어있으니, `.attrib` 사용하여 주소를 뽑아온다.

```
href_list = [a.attrib['href'] for a in postings]
```

```
href_list   # href_list 를 확인한다
```

```
['/ArticleRead.nhn?clubid=19773565&articleid=75127&page=1&boardtype=L&menuid=142',
     '/ArticleRead.nhn?clubid=19773565&articleid=75125&page=1&boardtype=L&menuid=142',
    ...]
```

각 게시물의 url 주소가 저장되어있다.

주소를 확인하면, `https://m.cafe.naver.com` 이 제외된 미완성 주소인 것을 확인할 수 있다.

`urljoin`을 사용하여 `https://m.cafe.naver.com` 을 `href_list` 의 각 항목에 추가한다.

```
base_url = 'https://m.cafe.naver.com'
```

```
new_urls = [urljoin(base_url, i) for i in href]
```

`len`를 통해 `new_urls`의 개수를 세준다.

```
len(new_urls)
```

```
20
```

총, 20개로 주소가 잘 뽑힌 것을 확인할 수 있다.

6.2.2. 제목, 본문 뽑기

이제 각 게시물의 제목, 본문을 크롤링하도록 하자.

Referer

앞서 말한대로, 그냥 접속하게 된다면 자료를 가져올 수가 없다.

그래서 네이버 검색창을 통해 접근한다고 생각하도록 해줘야한다.

그러기 위해선, `referer` 라는 옵션을 설정하면 된다.

먼저 네이버에서 아무 글자로 검색을 한다. 여기서는 숫자 1을 검색하였다.

주소를 확인하면 `https://m.search.naver.com` 뒤에 `search.naver?query=1` 부분을 `referer`에 넘겨주면, 네이버 검색을 통한 유입이라고 흉내내면서 자료를 요청을 할 수 있다. `referer`를 `headers`라는 변수에 dictionary 형태로 저장한다.

```
headers = {'referer': 'https://m.search.naver.com/search.naver?query=1'}
```

`requests.get()`에 url 과 `headers` 를 같이 넘겨주면 된다.

제목과 본문을 크롤링하기 위해 게시물을 하나 들어가보자.

```
url2 = new_urls[1]  # new_urls 목록 중, 두 번째 게시글에 접속한다.
# requests.get() 에 url 주소와 headers 를 같이 넘겨준다.
res2 = requests.get(url2, headers=headers)
element2 = lxml.html.fromstring(res2.text)
```

제목

제목 요소 검사를 하여 내용을 뽑는다. 만약, 마우스 우클릭이 되지 않는다면, 크롬의 경우, 설정 밑에 도구를 클릭하면 개발자 도구를 통해 확인이 가능하다.

제목은 `h2 class="tit"` 아래 있는 것을 확인할 수 있다.

```
element2.cssselect('h2.tit')[0].text_content()  # 해당 태그의 텍스트 정보를 가져온다.
```

```
'등업요청 부탁합니다'
```

제목이 잘 뽑힌 것을 확인할 수 있다.

본문

이제 내용을 가져오자.

다시 내용부분 검사를 해보면, 내용은 `div id="postContent"`에 들어있는 것을 확인할 수 있다.

`.cssselect()`를 사용하여 본문을 가져오자.

```
body = element2.cssselect('div#postContent')[0].text_content()
body
```

```
'\n\n\t\t \n\t\t \n\t     \n\t\n\t\t\n\t\t\n\t\t\n\t\t\t\n\t\t\t\n\t\t\t\t\n\t\t\t\t\n\t\t\t\t\t\n\t\t\t\t\t\n\t\t\t\t\t\n\t\t\t\t\t\t최근에 허리가 안 좋아서
```

`body`를 확인하면 공백을 뜻하는 `\n`, `\t`과 같은 불필요한 단어가 같이 추출된것을 확인할 수 있다.

```
body.strip()  # .strip() 을 통하여 불필요한 단어를 정리한다
```

```
'최근에 허리가 안 좋아서 의자에 관심이 많아졌네요 잘부탁드리고 등업요청합니다'
```

본문 내용만 잘 추출된 것을 확인할 수 있다.

6.2.3. 반복문

기존에 `new_urls` 에 저장한 게시물들의 제목과, 본문을 뽑아보도록 하자.

우선, 제목과 본문을 저장할 수 있는 빈 리스트를 생성한다.

```
titles = []   # 제목을 저장해둘 빈 리스트를 생성한다.
contents = []   # 본문을 저장할 빈 리스트를 생성한다.
```

그 후에 `for`문을 통해, 위에서와 같은 방식으로 제목과 본문을 크롤링하면 된다.

간혹 크롤링이 안되는 게시물이 있는데, 그것은 작성자가 글을 올릴 때, 검색 설정을 허용하지 않았기 때문이다.

그럴때는 `referer`로 흉내를 내도 정보를 가져올 수가 없다.

따라서 접속했을 때, 기존의 코드대로 제목을 뽑으면 빈 리스트가 추출이 된다.

그렇게 되면, `for`문을 돌 때, 제목을 가져오는 부분에서 `IndexError`가 발생한다.

그래서, `try`, `except` 문을 사용하여 제목과 본문을 추출한다..

`for`문을 한번 확인하자.

```
for i in new_urls:
    res = requests.get(i, headers=headers)
    element = lxml.html.fromstring(res.text)

    try:   # try 후에, 제목을 가져오는 코드를 실행한다.
        title = element.cssselect('h2.tit')[0].text_content()
    except IndexError:   # 만약 인덱스 에러가 발생한다면, except 부분으로 넘어간다.
        continue   # continue 를 하여, 해당 게시물을 건너 뛴다.
    titles.append(title)  # 제목을 titles 에 append 한다.

    # 본문
    body = element.cssselect('div#postContent')[0].text_content().strip()
    contents.append(body)  # 추출한 본문을 contents 에 append 한다.
```

`titles` 와 `contents`에 제목과 본문들이 저장된 것을 확인할 수 있다. 이제 추출한 결과를 `pandas.DataFrame`으로 만들어보자.

```
import pandas as pd
```

`pd.DataFrame`에 `titles`와 `contents`를 넘겨주고 컬럼 이름으로 '제목'과 '본문'으로 지정한다.

```
pd.DataFrame({'제목': titles, '본문': contents})
```

	제목	본문
0	등업요청~	일반회원 등업 요청입니다~
1	등업요청이에요	등업부탁드려요~!~
2	등업요청 드립니다.	정말 좋은 의자 찾기 너무 어렵네요. 도와주세요ㅠㅠ
3	등업요청 부탁합니다	최근에 허리가 안 좋아서 의자에 관심이 많아졌네요 잘부탁드리고 등업요청합니다
4	등업신청합니다^^	최근들어 의자에 관심이 많이 생겨 검색하는 도중에 우연히 의사모를 알게되어 가입까지...
5	가입인사	안녕하세요 가입했습니다.
6	등업요청	등업요청이요
7	등업 신청합니다	의자 구매가 쉽지가 않네요정보 공유를 위해 등업 부탁 드립니다~
8	등업요청합니다^^	문의글 드리고 싶습니다등업부탁드려요^^
9	등업부탁드립니다	등업부탁드려요
10	반가워요	등업

11	등업부탁드립니다	ㅎ
12	등업 부탁드려요	등업 부탁드립니다~^^
13	등업요청 드립니다.	부탁드려요~
14	등업 부탁드립니다	등업 요청합니다^^..

게시물의 제목과, 본문이 데이터 프레임 형태로 잘 추출된 것을 확인할 수 있다.