파이썬 프로그래밍

(Web Scraping)

01. 웹 스크랩핑 소개

웹 스크래핑(Web Scraping)은 웹 페이지로부터 원하는 정보를 추출하는 기법이다. 어떤 서비스에서 API가 별도로 제공되고 있지 않지만 웹 페이지로는 정보가 제공되고 있을 때, 웹 스크래핑 기법을 이용하면 원하는 정보를 획득할 수 있다.

웹 스크래핑은 흔히 웹 크롤링(Web Crawling)이라고도 많이 불린다. 물론 엄밀하게 두 단어는 서로 다른 의미이다. 크롤링은 여러 웹 페이지를 기계적으로 탐색하는 일을 말합니다. 한편 웹 스크래핑은 특정한 하나의 웹 페이지를 탐색하고, 소스코드 작성자가 원하는 정보를 콕 집어 얻어낸다는 점에서 크롤링과 차이가 있다. 그럼에도 크롤링과 스크래핑은 구현방법이 거의 같기 때문에, 실무에서는 구분 없이 불린다.

- [스크랩핑 페이지]-[F12]-[오른쪽마우스]-[Copy]-[Copy XPath] 또는 [Copy full XPath] 값을 얻을 수 있다.

[c:]-[data]-[python]-[scraping] 01_xpath.txt

```
/학교/학년/반/학생[2] 강감찬  /한 단계 아래
//*[@학번="1-1-5"]   //*문서전체 [@학번="1-1-5"] 학번속성이 "1-1-5"검색

<학교 이름="나도고등학교">
    <학년 value="1학년">
        <반 value="1반">
            <학생 value="1번" 학번='1-1-1">홍길동</학생>
            <학생 value="2번" 학번='1-1-2">심청이</학생>
            <학생 value="3번" 학번='1-1-3">강감찬</학생>
            <학생 value="4번" 학번='1-1-4">이몽룡</학생>
            <학생 value="5번" 학번='1-1-5">홍길동</학생>
        </반>
        <반 value="2반"/>
        <반 value="3반"/>
        <반 value="4반"/>
    </학년>
    <학년 value="1학년">... 유재석. ..</학년>
    <학년 value="2학년">... 유재석. ..</학년>
    <학년 value="3학년"></학년>
    <학년 value="4학년"></학년>
</학교>
```

- requests 라이브러리 설치

```
pip install requests
```

[c:]-[data]-[python]-[scraping] 01_requests.py

```
import requests

res = requests.get("http://google.com")
#res = requests.get("http://nadocoding.tistory.com")
res.raise_for_status()  #오류가 발생한 경우 프로그램을 종료한다.

# print("응답코드: " , res.status_code)  #200이면 정상

# if res.status_code == requests.codes.ok:
#     print("정상입니다.")
# else:
#     print("문제가 생겼습니다. [에러코드 ", res.status_code, "]")

print(len(res.text))
print(res.text)

with open("mygoogle.html", "w", encoding="utf-8") as file:
    file.write(res.text)
```

02. 정규식 표현

정규표현식(Regular expressions) 은 특정한 규칙을 가진 문자열의 집합을 표현하는 데 사용하는 형식 언어이다. 복잡한 문자열의 검색과 치환을 위해 사용되며, Python 뿐만 아니라 문자열을 처리하는 모든 곳에서 사용된다.

[c:]-[data]-[python]-[scraping] 02_re01.py

```
import re

p = re.compile('ca.e')  #'ca?e'인 단어들 'care', 'cafe', 'case', 'cave'
# . (ca.e): 하나의 문자를 의미  예) care, cafe, case (O) | caffe (X)
# ^ (^de) : 문자열의 시작  예) desk, destination (O) | fade (X)
# $ (se$) : 문자열의 끝  예) case, base (O) | face (X)

m = p.match("care")  #match 함수는 주어진 문자열이 처음부터 일치하는지 확인

if m:
    print(m.group())
else:
    print("매칭 되지 않음")

m = p.match("cafe)
if m:
    print(m.group())
else:
    print("매칭 되지 않음")

m = p.match("good care")
if m:
    print(m.group())
else:
    print("매칭 되지 않음 “)
```

[c:]-[data]-[python]-[scraping] 02_re01.py 실행결과

```
care
cafe
매칭 되지 않음
```

[c:]-[data]-[python]-[scraping] 02_re02.py

```
import re

p = re.compile("ca.e")

def print_match(m):
    if m:
        print(m.group())
    else:
        print("매칭 되지 않음")

m = p.match("case")
print_match(m)

m = p.match("cafe")
print_match(m)

m = p.match("good care")
print_match(m)

m = p.match("careless")
print_match(m)
```

[c:]-[data]-[python]-[scraping] 02_re02.py 실행결과

```
case
cafe
매칭 되지 않음
care
```

[c:]-[data]-[python]-[scraping] 02_re03.py

```
import re

p = re.compile("ca.e")

def print_match(m):
    if m:
        print("m.group():", m.group())  #일치하는 문자열 반환
        print("m.string:", m.string)  #입력받은 문자열
        print("m.start():", m.start())  #일치하는 문자열의 시작 index
        print("m.end():", m.end())  #일치하는 문자열의 끝 index
        print("m.span()", m.span())  #일치하는 문자열의 시작 / 끝 index
        print()
    else:
        print("매칭 되지 않음")

#search 함수는 주어진 문자열 중에 일치하는 것이 있는지 확인
m = p.search("care")
print_match(m)

m = p.search("good care")
print_match(m)

m = p.search("careless")
print_match(m)
```

[c:]-[data]-[python]-[scraping] 02_re03.py 실행결과

```
m.group(): care
m.string: care
m.start(): 0
m.end(): 4
m.span() (0, 4)

m.group(): care
m.string: good care
m.start(): 5
m.end(): 9
m.span() (5, 9)

m.group(): care
m.string: careless
m.start(): 0
m.end(): 4
m.span() (0, 4)
```

[c:]-[data]-[python]-[scraping] 02_re04.py

```
import re

p = re.compile('ca.e')

#일치하는 모든 것을 리스트 형태로 출력
lst = p.findall("good care")
print(lst)

lst = p.findall("good care cafe")
print(lst)
```

[c:]-[data]-[python]-[scraping] 02_re04.py 실행결과

```
['care']
['care', 'cafe']
```

03. User Agent

일반 사용자가 아닌 크롤러 등이 사이트에 접근하려고 할 때 접근이 차단되도록 설정되어 있다. 따라서 사이트에게 사람인 척 해주어야한다. 사람인 척 하는 방법은 페이지에 접속할 때 User Agent 값을 넘겨주는 것이다. User agent는 접속하는 PC, 브라우저에 따라 달라진다.

[c:]-[data]-[python]-[scraping] 03_user_agent.py

```
import requests

url = "http://nadocoding.tistory.com"
headers = {"User-Agent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/101.0.4951.54 Safari/537.36"}  #구글에서 'User Agent String' 검색

res = requests.get(url, headers=headers)
res.raise_for_status()

with open("nadocoding.html", "w", encoding="utf-8") as f:
    f.write(res.text)
```

04. BeautifulSoup4

- **스크랩핑에 필요한 라이브러리들을 설치한다.**

```
pip install beautifulSoup4  스크랩핑을 위한 패키지
pip install lxml  구문 파서용 패키지
```

1) 네이버 웹툰

[c:]-[data]-[python]-[scraping] 04_bs4.py

```
import requests
from bs4 import BeautifulSoup

url = "https://comic.naver.com/webtoon/weekday"
res = requests.get(url)
res.raise_for_status()

soup = BeautifulSoup(res.text, "lxml")
print("1).....................")
print(soup.title)
print("2).....................")
print(soup.title.get_text())
print("3).....................")
print(soup.a)  #soup 객체에서 처음 발견되는 a element 출력
print("4).....................")
print(soup.a.attrs)  #soup 객체에서 a element의 속성들 정보를 dictionary로 출력
print("5).....................")
print(soup.a["href"])  #soup 객체에서 a element의 속성 값 정보를 출력
```

[c:]-[data]-[python]-[scraping] 04_bs4.py 실행결과

```
1).....................
<title>네이버 웹툰 &gt; 요일별  웹툰 &gt; 전체웹툰</title>
2).....................
네이버 웹툰 > 요일별  웹툰 > 전체웹툰
3).....................
<a href="#menu" onclick="document.getElementById('menu').tabIndex=-1;document.getElementById('menu').focus();return false;"><span>메인 메뉴로
바로가
기</span></a>
4).....................
{'href': '#menu', 'onclick': "document.getElementById('menu').tabIndex=-1;document.getElementById('menu').focus();return false;"}
5).....................
#menu
```

[c:]-[data]-[python]-[scraping] 04_bs4.py

```
import requests
from bs4 import BeautifulSoup

url = "https://comic.naver.com/webtoon/weekday"
res = requests.get(url)
res.raise_for_status()

soup = BeautifulSoup(res.text, "lxml")
print("1)웹툰 올리기 버튼.....")
print(soup.find("a", attrs={"class": "Nbtn_upload"}))  #class="Nbtn_upload"인 a element를 검색
print(soup.find(attrs={"class": "Nbtn_upload"}))  #class="Nbtn_upload"인 어떤 element를 검색
```

[c:]-[data]-[python]-[scraping] 04_bs4.py 실행결과

```
1)웹툰 올리기 버튼.....
<a class="Nbtn_upload" href="/mypage/myActivity" onclick="nclk_v2(event,'olk.upload');">웹툰 올리기</a>
<a class="Nbtn_upload" href="/mypage/myActivity" onclick="nclk_v2(event,'olk.upload');">웹툰 올리기</a>
```

[c:]-[data]-[python]-[scraping] 04_bs4.py

```
import requests
from bs4 import BeautifulSoup

url = "https://comic.naver.com/webtoon/weekday"
res = requests.get(url)
res.raise_for_status()

soup = BeautifulSoup(res.text, "lxml")

print("1)인기급상승 1등.....")
rank1 = soup.find("li", attrs={"class":"rank01"})
print(rank1.a.getText())
#print(rank1.parent) rank1의 부모 element 출력

print("2)인기급상승 2등.....")
rank2 = rank1.find_next_sibling("li")
print(rank2.a.get_text())

print("3)인기급상승 3등.....")
rank3 = rank1.find_next_sibling("li")
print(rank2.a.get_text())

print("4)인기급상승 2등.....")
rank2 = rank3.find_previous_sibling("li")
print(rank2.a.get_text())

#print(rank1.find_next_siblings("li")) rank1의 li element 모든 형제들 출력

print("5)인기급상승 a element text값이 김부장.....")
webtoon = soup.find("a", text="김부장")
print(webtoon)
```

[c:]-[data]-[python]-[scraping] 04_bs4.py 실행결과

```
1)인기급상승 1등.....
김부장-29화 너무 위험한 면담 [1/2]
2)인기급상승 2등.....
사신소년-1부 에필로그
3)인기급상승 3등.....
사신소년-1부 에필로그
4)인기급상승 2등.....
김부장-29화 너무 위험한 면담 [1/2]
5)인기급상승 a element text값이 김부장.....
<a class="title" href="/webtoon/list?titleId=783053&amp;weekday=tue" onclick="nclk_v2(event,'thm*t.tit','','1')" title="김부장">김부장</a>
```

[c:]-[data]-[python]-[scraping] 04_bs4_webtoons.py

```
import requests
from bs4 import BeautifulSoup

url = "https://comic.naver.com/webtoon/weekday"
res = requests.get(url)
res.raise_for_status()

soup = BeautifulSoup(res.text, "lxml")

print("1)요일별 전체 웹툰.....................")
cartoons = soup.find_all("a", attrs={"class":"title"}) #a element의 class 속성이 title인 모든 element를 반환
for cartoon in cartoons:
    print(cartoon.get_text())
```

[c:]-[data]-[python]-[scraping] 04_bs4_webtoons.py 실행결과

```
1)요일별 전체 웹툰.....................
참교육
신의 탑
뷰티풀 군바리
윈드브레이커
퀘스트지상주의
...
```

[c:]-[data]-[python]-[scraping] 04_gauss.py

```
import requests
from bs4 import BeautifulSoup

url = "https://comic.naver.com/webtoon/list?titleId=335885"
res = requests.get(url)
res.raise_for_status()

soup = BeautifulSoup(res.text, "lxml")

print("1)가우스전자 첫번째 제목과 링크주소.....................")
cartoons = soup.find_all("td", attrs={"class":"title"})
title = cartoons[0].a.get_text()
link = cartoons[0].a["href"]
print(title)
print("https://comic.naver.com" + link)


print("1)가우스전자 무든 제목과 링크주소.....................")
for cartoon in cartoons:
    title = cartoon.a.get_text()
    link = "https://comic.naver.com" + cartoon.a["href"]
    print(title, link)
```

[c:]-[data]-[python]-[scraping] 04_gauss.py 실행결과

```
1)가우스전자 첫번째 제목과 링크주소.....................
250화 상식씨!
https://comic.naver.com/webtoon/detail?titleId=335885&no=252&weekday=mon
2)가우스전자 무든 제목과 링크주소.....................
250화 상식씨! https://comic.naver.com/webtoon/detail?titleId=335885&no=252&weekday=mon
249화 이해 https://comic.naver.com/webtoon/detail?titleId=335885&no=251&weekday=mon
248화 결재타이밍 https://comic.naver.com/webtoon/detail?titleId=335885&no=250&weekday=mon
247화 남동생 https://comic.naver.com/webtoon/detail?titleId=335885&no=249&weekday=mon
246화 도시농업 https://comic.naver.com/webtoon/detail?titleId=335885&no=248&weekday=mon
245화 소개팅 https://comic.naver.com/webtoon/detail?titleId=335885&no=247&weekday=mon
244화 다이어트식단 https://comic.naver.com/webtoon/detail?titleId=335885&no=246&weekday=mon
243화 설계 https://comic.naver.com/webtoon/detail?titleId=335885&no=245&weekday=mon
242화 보릿고개 https://comic.naver.com/webtoon/detail?titleId=335885&no=244&weekday=mon
241화 용기 https://comic.naver.com/webtoon/detail?titleId=335885&no=243&weekday=mon
```

[c:]-[data]-[python]-[scraping] 04_gauss.py

```
import requests
from bs4 import BeautifulSoup

url = "https://comic.naver.com/webtoon/list?titleId=335885"
res = requests.get(url)
res.raise_for_status()

soup = BeautifulSoup(res.text, "lxml")

cartoons = soup.find_all("div", attrs={"class":"rating_type"})

print("1)가우스전자 평점구하기.....................")
total_rates = 0
for cartoon in cartoons:
    rate = cartoon.find("strong").get_text()
    print(rate)
    total_rates += float(rate)

print("전체 점수: " , total_rates)
print("평균 점수:" , total_rates/len(cartoons))
```

[c:]-[data]-[python]-[scraping] 04_gauss.py 실행결과

```
1)가우스전자 평점구하기.....................
9.96
9.96
9.96
9.96
9.96
9.96
9.96
9.94
9.96
9.95
전체 점수:  99.57000000000001
평균 점수: 9.957
```

- **터미널에서 Shell을 이용한 명령어 실행**

Python 3.x Shell

```
PS C:\data\python\scraping> python  python sell 창을 실행한다.
Python 3.9.6 (tags/v3.9.6:db3ff76, Jun 28 2021, 15:26:21) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import requests
>>> from bs4 import BeautifulSoup
>>>
>>> url = "https://comic.naver.com/webtoon/list?titleId=335885"
>>> res = requests.get(url)
>>> res.raise_for_status()
>>>
>>> soup = BeautifulSoup(res.text, "lxml")
>>> cartoons = soup.find_all("div", attrs={"class":"rating_type"})
>>> cartoons
[<div class="rating_type">
<span class="star"><em style="width:99.56%">평점</em></span>
<strong>9.96</strong>
...
</div>]
>>> cartoons[0].find("strong")
<strong>9.96</strong>
>>> cartoons[0].find("strong").get_text()
'9.96'
>>> exit()  명령창을 종료한다.
```

2) 지마켓

- 지마켓에서 '노트북' 검색 후 1페이지에 '상품명', '가격', '평점', '피드백수' 출력

[c:]-[data]-[python]-[scraping] 04_bs4_gmarket.py

```
import requests
import re
from bs4 import BeautifulSoup

url="https://browse.gmarket.co.kr/search?keyword=%eb%85%b8%ed%8a%b8%eb%b6%81&k=32&p=1"
headers = {"User-Agent": "Mozilla/5.0 ... AppleWebKit/537.36 (KHTL, like Gecko) Chrome/101.0.4951.54 Safari/537.36"}
res = requests.get(url, headers=headers)
res.raise_for_status()

soup = BeautifulSoup(res.text, "lxml")
items = soup.find_all("div", attrs={"class":re.compile("^box__item-container")})

for idx, item in enumerate(items, start=1):
    title = item.find("span", attrs={"class":"text__item"})  #제품명
    price = item.find("strong", attrs={"class":"text text__value"}).get_text()  #가격
    rate = item.find("span", attrs={"class":"image__awards-points"})  #평점
    feedback = item.find("li", attrs={"class":re.compile("list-item__feedback-count$")})  #피드백 수

    if rate:
        rate = rate["style"]
        index = rate.find(":") + 1
        rate = rate[index:]
    else:
        rate = "평점 없음"
    if feedback:
        feedback_cnt = feedback.find("span", attrs={"class":"text"}).get_text()
    else:
        feedback = "피드백 없음"

    print(idx, title["title"], price, rate, feedback_cnt)
```

[c:]-[data]-[python]-[scraping] 04_bs4_gmarket.py 실행결과

```
1 LG 그램 2022 신제품 16ZD95P-GX76K RAM 16GB/당일발송 1,729,000 98% (44)
2 LG 그램 12세대 노트북 16ZD90Q-EX76K 외장그래픽 탑재 2,249,000 84% (16)
...
105 UX3402ZA-KM054W 최종126만/EVO12세대i7/600NITS/Win11 1,599,000 100% (4)
```

- 지마켓에서 '노트북' 검색 후 1페이지에서 평점이 90점이상아고 피드백 수가 300이상인 '상품명', '가격', '평점', '피드백수' 출력

[c:]-[data]-[python]-[scraping] 04_bs4_gmarket.py

```
...
for idx, item in enumerate(items, start=1):
    title = item.find("span", attrs={"class":"text__item"})  #제품명
    price = item.find("strong", attrs={"class":"text text__value"}).get_text()  #가격
    rate = item.find("span", attrs={"class":"image__awards-points"})  #평점
    feedback = item.find("li", attrs={"class":re.compile("list-item__feedback-count$")})

    if rate:
        rate = rate["style"]
        index = rate.find(":") + 1  #예) 95%
        rate = rate[index:-1]  #예) 95% -> 95
    else:
         continue
    if feedback:
        feedback_cnt = feedback.find("span", attrs={"class":"text"}).get_text()  #예) (26)
        feedback_cnt = feedback_cnt[1:-1]  #예) (26)-> 26
        feedback_cnt = feedback_cnt.replace(",","")  #예) 1,345 -> 1345
    else:
        continue

    if int(rate) >= 90 and int(feedback_cnt) >= 300:  #평점 90%이상이면서 피드백수가 300이상
        print(idx, title["title"], price, rate, feedback_cnt)
```

- **지마켓에서 '노트북' 검색 후 1~5 페이지의 위 조건을 만족하는 상품정보 출력**

[c:]-[data]-[python]-[scraping] 04_bs4_gmarket_pages.py

```
import requests
import re
from bs4 import BeautifulSoup

headers = {"User-Agent": "Mozilla/5.0 ... /537.36 (KHTL, like Gecko) Chrome/101.0.4951.54 Safari/537.36"}

for i in range(1, 6):
    #print("페이지: " , i)

    url="https://browse.gmarket.co.kr/search?keyword=%eb%85%b8%ed%8a%b8%eb%b6%81&k=32&p={}".format(i)
    res = requests.get(url, headers=headers)
    res.raise_for_status()
    soup = BeautifulSoup(res.text, "lxml")

    items = soup.find_all("div", attrs={"class":re.compile("^box__item-container")})

    for idx, item in enumerate(items, start=1):
        title = item.find("span", attrs={"class":"text__item"}).get_text()
        price = item.find("strong", attrs={"class":"text text__value"}).get_text()
        rate = item.find("span", attrs={"class":"image__awards-points"})
        feedback = item.find("li", attrs={"class":re.compile("list-item__feedback-count$")})
        link = item.find("a", attrs={"class": "link__item"})["href"]

        if rate:
            rate = rate["style"]
            index = rate.find(":") + 1
            rate = rate[index:-1]
        else:
            continue

        if feedback:
            feedback_cnt = feedback.find("span", attrs={"class":"text"}).get_text()
            feedback_cnt = feedback_cnt[1:-1]
            feedback_cnt = feedback_cnt.replace(",","")
        else:
            continue

        if int(rate) >= 90 and int(feedback_cnt) >= 300:
            print(f"제품명 : {title}")
            print(f"가격: {price}")
            print(f"평점: {rate}% ({feedback_cnt}개)")
            print(f"바로가기: {link}")
            print("-" * 100)
```

[c:]-[data]-[python]-[scraping] 04_bs4_gmarket_pages.py 실행결과

```
제품명 :  15.6인치 17인치 LG그램 맥북 노트북 파우치 가방 P60
가격: 19,900
평점: 92% (351개)
바로가기: http://item.gmarket.co.kr/Item?goodscode=2102111886
----------------------------------------------------------------------------------------------------
제품명 :  .잘만 ZM-NS2000 노트북거치대 받침대 쿨링패드 쿨러
가격: 39,000
평점: 92% (323개)
바로가기: http://item.gmarket.co.kr/Item?goodscode=695224886
----------------------------------------------------------------------------------------------------
제품명 :  SMO-3550B 노트북 컴퓨터 PC USB 피시 무선 광 마우스
가격: 20,000
평점: 92% (755개)
바로가기: http://item.gmarket.co.kr/Item?goodscode=1100902613
----------------------------------------------------------------------------------------------------
제품명 :  잘만 ZM-NS1000 노트북거치대 받침대 쿨링패드 쿨러
가격: 23,000
평점: 96% (404개)
바로가기: http://item.gmarket.co.kr/Item?goodscode=694896609
----------------------------------------------------------------------------------------------------
...
```

3) 다음 영화 이미지 다운로드

- [다음]-[영화]-[역대 관객 순위]-[2019] 상위 5개이미지 다운로드 검색

[c:]-[data]-[python]-[scraping] 04_daum_movies.py

```
import requests
from bs4 import BeautifulSoup

url = "https://search.daum.net/search?w=tot&q=2019%EB%85%84%EC%98%81%ED%99%94%EC%88%9C%EC%9C%84&DA=MOR&rtmaxcoll=MOR"
res = requests.get(url)
res.raise_for_status()

soup = BeautifulSoup(res.text, "lxml")
images = soup.find_all("img", attrs={"class":"thumb_img"})

for idx, image in enumerate(images):  #인덱스와 원소로 이루어진 Tuple을 생성한다.
    image_url = image["src"]
    if image_url.startswith("//"):
        image_url  = "https:" + image_url

    print(image_url)
    image_res = requests.get(image_url)
    res.raise_for_status()

    with open("move{}.jpg".format(idx+1), "wb") as f:
        f.write(image_res.content)

    if idx >= 4:  #상위 5개 데이터 이미지만 다운로드
        break
```

[c:]-[data]-[python]-[scraping] 04_bs4.py

```
https://search1.kakaocdn.net/thumb/R232x328.q85/?fname=http%3A%2F%2Ft1.daumcdn.net%2Fmovie%2F4e00e81f2b6f4d2eb65b3387240cc3
https://search1.kakaocdn.net/thumb/R232x328.q85/?fname=http%3A%2F%2Ft1.daumcdn.net%2Fmovie%2F5574fb2c20c844629aa9ad1d6043ee
https://search1.kakaocdn.net/thumb/R232x328.q85/?fname=http%3A%2F%2Ft1.daumcdn.net%2Fmovie%2F5afd212b68e34e61a964d969dd898e
https://search1.kakaocdn.net/thumb/R232x328.q85/?fname=http%3A%2F%2Ft1.daumcdn.net%2Fmovie%2F3673a8a0c5ff4f5c8c25cc959fd6985b
https://search1.kakaocdn.net/thumb/R232x328.q85/?fname=http%3A%2F%2Ft1.daumcdn.net%2Fmovie%2Fcab3b02a7b274bd6838b80a5e481fe
```

- 2015~2019년 관객순위 5위 이미지 다운로드

[c:]-[data]-[python]-[scraping] 04_daum_movies.py

```
import requests
from bs4 import BeautifulSoup

for year in range(2015, 2020):
    url = "https://search.daum.net/search?w=tot&q={0}%EB%85%84%EC%98%81%ED%99%94%EC%88%9C%E&DA=MOR&rtmaxcoll=MOR".format(year)
    res = requests.get(url)
    res.raise_for_status()

    soup = BeautifulSoup(res.text, "lxml")
    images = soup.find_all("img", attrs={"class":"thumb_img"})

    for idx, image in enumerate(images):

        image_url = image["src"]
        if image_url.startswith("//"):
            image_url  = "https:" + image_url

        print(image_url)
        image_res = requests.get(image_url)
        res.raise_for_status()

        with open("move_{}_{}.jpg".format(year, idx+1), "wb") as f:
            f.write(image_res.content)

        if idx >= 4:
            break
```

---

4) 웹 스크래핑 데이터 CSV 파일 저장

- [네이버]-[코스피 시가총액 순위]-[시가총액 상위종목 더보기]

[c:]-[data]-[python]-[scraping] 04_bs4.stock.py

```
import csv
import requests
from bs4 import BeautifulSoup

url ="https://finance.naver.com/sise/sise_market_sum.nhn?sosok=0&page="

for page in range(1, 5):
    res  = requests.get(url + str(page))
    res.raise_for_status()
    soup = BeautifulSoup(res.text, "lxml")

    data_rows = soup.find("table", attrs={"class":"type_2"}).find("tbody").find_all("tr")

    for row in data_rows:
        columns = row.find_all("td")

        #의미 없는 데이터는 skip
        if len(columns) <=1 :
            continue

        data = [column.get_text().strip() for column in columns]  #한 줄 for문
        print(data)
```

[c:]-[data]-[python]-[scraping] 04_bs4.stock.py

```
import csv
import requests
from bs4 import BeautifulSoup

url ="https://finance.naver.com/sise/sise_market_sum.nhn?sosok=0&page="

fileName = "시가총액1-200.csv"  #1위 ~ 200위 종목 출력
file = open(fileName, "w", encoding="utf8", newline="")
writer = csv.writer(file)

title ="N    종목명    현재가    전일비    등락률    ...    거래량    PER    ROE".split("\t")
writer.writerow(title)

for page in range(1, 5):
    res  = requests.get(url + str(page))
    res.raise_for_status()
    soup = BeautifulSoup(res.text, "lxml")

    data_rows = soup.find("table", attrs={"class":"type_2"}).find("tbody").find_all("tr")

    for row in data_rows:
        columns = row.find_all("td")
        if len(columns) <=1 :
            continue

        data = [column.get_text().strip() for column in columns]
        #print(data)
        writer.writerow(data)
```

[c:]-[data]-[python]-[scraping] 시가총액1-20.csv

```
N,종목명,현재가,전일비,등락률,액면가,시가총액,상장주식수,외국인비율,거래량,PER,ROE
1,삼성전자,"68,100","500,+0.74%,100,"4,065,422","5,969,783",50.74,"15,079,552",11.79,13.92,
2,LG에너지솔루션,"411,000","2,500",+0.61%,500,"961,740","234,000",2.95,"213,490",103.71,10.68,
3,SK하이닉스,"114,500","2,000",+1.78%,"5,000","833,563","728,002",49.65,"3,314,733",8.68,16.84,
...
197,두산,"78,700","3,900",-4.72%,"5,000","13,004","16,524",11.30,"213,371",7.96,11.08,
198,삼성 레버리지 WTI원유 선물 ETN,"2,560",65,-2.48%,0,"12,902","504,000",0.00,"704,281",N/A,N/A,
199,TIGER 미국테크TOP10 INDXX,"10,425",90,+0.87%,0,"12,896","123,700",0.44,"581,945",N/A,N/A,
200,일동제약,"47,600","1,000",+2.15%,"1,000","12,758","26,803",1.82,"3,514,922",-11.36,-48.94,
```

05. Selenium

selenium은 웹사이트 테스트를 위한 도구로 브라우저 동작을 자동화할 수 있다. 셀레니움을 이용하는 웹크롤링 방식은 바로 이점을 적극적으로 활용하는 것이다. 프로그래밍으로 브라우저 동작을 제어해서 마치 사람이 이용하는 것 같이 웹페이지를 요청하고 응답을 받아올 수 있다.

1) Selenium 기본

- Selenium 라이브러리를 설치한다.

```
pip install selenium
```

- 아래 사이트로 이동하여 현재 크롬버전과 같은 프로그램을 다운로드 받은 후 압축을 해제하여 프로젝트 폴더에 저장한다.

```
https://chromedriver.chromium.org/downloads
```

- 네이버 사이트로 이동한 후 로그인 페이지로 이동한다.

```
[c:]-[data]-[python]-[scraping] 05_selenium.py
```

```
from selenium import webdriver

browser = webdriver.Chrome()  #현재 폴더에 저장된 경우 "./chromedriver.exe" 생략가능

browser.get("http://naver.com")

elem = browser.find_element_by_class_name("link_login")
elem.click()  #로그인 버튼을 클릭한다.

browser.back()  #뒤로 가기 버튼을 클릭한다.
browser.forward()  #앞으로 가기 버튼을 클릭한다.
browser.refresh()  #새로고침 버튼을 클릭한다.
```

- 네이버 검색창에서 '나도코딩' 검색후 'a' 태그의 링크값들을 출력한다.

```
[c:]-[data]-[python]-[scraping] 05_selenium.py
```

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys

browser = webdriver.Chrome()

browser.get("http://naver.com")

elem = browser.find_element_by_id("query")
elem.send_keys("나도코딩")
elem.send_keys(Keys.ENTER)

elem = browser.find_elements_by_tag_name("a")
for e in elem:
    print(e.get_attribute("href"))
```

- 다음 검색창에서 '나도코딩' 검색한 후 엔터키를 친다.

```
[c:]-[data]-[python]-[scraping] 05_selenium.py
```

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys

browser = webdriver.Chrome()

browser.get("http://daum.net)

elem = browser.find_element_by_name("q")
elem.send_keys("나도코딩")
elem.send_keys(Keys.ENTER)
```

- **다음 검색창에서 '나도코딩' 검색한 후 검색버튼을 클릭한다..**

[c:]-[data]-[python]-[scraping] 05_selenium.py

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys

browser = webdriver.Chrome()

browser.get("http://daum.net")

elem = browser.find_element_by_name("q")
elem.send_keys("나도코딩")
#elem.send_keys(Keys.ENTER)

#마우스오른쪽 -> copy -> Copy XPath
elem = browser.find_element_by_xpath('//*[@id="daumSearch"]/fieldset/div/div/button[2]')
elem.click()

#browser.close()  #현재 탭 브라우저 종료
browser.quit()  #모든 탭 브라우저 종료
```

2) Selenium 심화

- **네이버 로그인**

[c:]-[data]-[python]-[scraping] 05_naver_login.py

```
import time
from selenium import webdriver

browser = webdriver.Chrome()

#1. 네이버 이동
browser.get("http://naver.com")

#2. 로그인 버튼 클릭
elem = browser.find_element_by_class_name("link_login")
elem.click()

#3. id, pw 입력
browser.find_element_by_id("id").send_keys("my_id")
browser.find_element_by_id("pw").send_keys("my_password")

#4. 로그인 버튼 클릭
browser.find_element_by_id("log.login").click()
time.sleep(3)

#5. id를 새로입력
browser.find_element_by_id("id").clear()
browser.find_element_by_id("id").send_keys("new_id")

#6. html 정보 출력
print(browser.page_source)

#7. 브라우저 종료
browser.quit()  #전체 브라우저 종료
```

[c:]-[data]-[python]-[scraping] 05_naver_login.py 실행결과

```
...
<input type="hidden" id="session_keys" name="session_keys" value="">
<input type="hidden" id="ncaptchaSplit" name="ncaptchaSplit" value="none">
<input type="hidden" id="failUrl" name="failUrl" value="">
<input type="hidden" id="nclicks_nsc" name="nclicks_nsc" value="nid.login_kr">
<input type="hidden" id="id_error_msg" name="id_error_msg" value="<strong>아이디</strong>를 입력해주세요.">
<input type="hidden" id="pw_error_msg" name="pw_error_msg" value="<strong>비밀번호</strong>를 입력해주세요.">
<input type="hidden" id="changeToSound" name="pw_error_msg" value="음성으로 듣기">
<input type="hidden" id="changeToPicture" name="changeToPicture" value="이미지로 보기">
...
```

3) Selenium 활용

- [네이버]-[네이버 항공권]에서 이번달 25일, 26일 제주도 항공권 검색

[c:]-[data]-[python]-[scraping] 05_naver_fights.py

```
import time
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC

browser = webdriver.Chrome()
browser.maximize_window()

url = "https://flight.naver.com/flights"
browser.get(url)

#가는 날 선택 클릭
#find_element(by=By.XPATH, value="") by=By.CLASS_NAME, CSS_SELECTOR, LINK_TEXT, ID, NAME, TAG_NAME
browser.find_element(By.XPATH,"//*[@id='__next']/div/div[1]/div[4]/div/div/div[2]/div[2]/button[1]").click()
time.sleep(2)

#이번달 25일, 26일
#browser.find_elements(By.XPATH, "//td/button/b[text()=28]")[0].click()
#browser.find_elements(By.XPATH, "//td/button/b[text()=29]")[0].click()

#다음달 25일, 26일
browser.find_elements(By.XPATH, "//td/button/b[text()=28]")[1].click()
browser.find_elements(By.XPATH, "//td/button/b[text()=29]")[1].click()

#도착지 선택
browser.find_element(By.XPATH, "//*[@id='__next']/div/div[1]/div[4]/div/div/div[2]/div[1]/button[2]").click()
time.sleep(2)

#국내 선택
browser.find_element(By.XPATH, "//*[@id='__next']/div/div[1]/div[9]/div[2]/section/section/button[1]").click()

#제주도 선택
browser.find_element(By.XPATH, "//*[@id='__next']/div/div[1]/div[9]/div[2]/section/section/div/button[2]").click()

#항공권 검색 선택
browser.find_element(By.XPATH, "//*[@id='__next']/div/div[1]/div[4]/div/div/button").click()

#첫 번째 결과 출력
try:
    first ="//*[@id='__next']/div/div[1]/div[5]/div/div[2]/div[2]/div"
    #최대 20초까지 기다리는데 XPATH의 first 내용 값이 존재할 때까지 기다린다.
    elem = WebDriverWait(browser, 20).until(EC.presence_of_all_elements_located((By.XPATH, first)))[0]
    print(elem.text)
finally:
    browser.quit()
```

[c:]-[data]-[python]-[scraping] 05_naver_fights.py 실행결과

```
에어서울
이벤트혜택
06:00GMP
07:00CJU
01시간 00분
할인석편도 50,200원~
1,000원 적립
편도 49,200원~
```

- [구글]-[송중기]-[이미지] 검색후 제목과 link를 출력

[c:]-[data]-[python]-[scraping] 05_google_images.py

```
import requests
from bs4 import BeautifulSoup

url = "https://www.google.com/search?q=%EC%86%A1%EC%A4%91%EA%B8%B0&hl=ko&sxsrf=A..."
headers = {
    "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 ... Chrome/102.0.5005.62 Safari/537.36",
    "Accept-Language" : "ko-KR, ko"
}

res = requests.get(url, headers=headers)
res.raise_for_status()
soup = BeautifulSoup(res.text, "lxml")

images = soup.find_all("div", attrs={"class": "isv-r PNCib MSM1fd BUooTd"})
print(len(images))

with open("images.html", "w", encoding="utf8") as f:  #images.html 파일생성 후 브라우저에 출력해본다.
    #f.write(res.text)
    f.write(soup.prettify())  #prettify() html 문서를 예쁘게 출력
```

[c:]-[data]-[python]-[scraping] images.html

```
...
            <div class="isv-r PNCib MSM1fd BUooTd" data-cb="0" data-cl="21" data-cr="12" data-ct="0" data-hveid="CAEQiQI" data-ictx="1"
data-id="ySnpdAzcKAIFgM" data-oh="667" data-ow="1000" data-ri="25" data-sc="1" data-tbnid="ySnpdAzcKAIFgM" data-tw="275"
data-ved="2ahUKEwj9tdq_tP_3AhUChMYKHSaeA10QMygZegUIARCJAg" jsaction="dtRDof:s370ud;" jscontroller="H9Mlue"
jsdata="j0Opre;ySnpdAzcKAIFgM;40" jsmodel="lbVNPd Whqy4b" jsname="N9Xkfe" style="width:145px;">
              <h3 class="bytUYc">
               야나두, 송중기와 '야핏 사이클' 광고모델 계약 - 전자신문
              </h3>
              <a class="wXeWr islib nfEiy" data-nav="1" jsaction="J9iaEb;mousedown:npT2md; touchstart:npT2md;" jsname="sTFXNd"
style="height:144px;" tabindex="0">
               <div class="bRMDJf islir" jsaction="mousedown:npT2md; touchstart:npT2md;" jsname="DeysSe" role="button"
style="background:rgb(139,146,152);margin-left:-45px; margin-right:-26px;width: 216px;" tabindex="0">
                <img alt="야나두, 송중기와 '야핏 사이클' 광고모델 계약 - 전자신문" class="rg_i Q4LuWd" data-ils="4"
data-src="https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcSIHhMPXKRkhzoZzHFY9i7G3_NeuL-REAOzIg&amp;usqp=CAU" height="144"
jsaction="rcuQ6b:trigger.M8vzZb;" jsname="Q4LuWd" width="216"/>
              </div>
...
```

[c:]-[data]-[python]-[scraping] 05_google_images.py

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

res = requests.get(url, headers=headers)
res.raise_for_status()
soup = BeautifulSoup(res.text, "lxml")

images = soup.find_all("div", attrs={"class": "isv-r PNCib MSM1fd BUooTd"})
print(len(images))

for image in images:
    title = image.find("span", attrs={"class": "OztcRd"}).get_text()
    print("title:", title)
```

[c:]-[data]-[python]-[scraping] 05_google_images.py 실행결과

```
48
title: 멜로부터 느와르까지 해낸 송중기 "현실에 나쁜 놈 많아 공감 끌어낸 듯" | 중앙일보
title: 비즈한국
...
title: 결혼 앞둔 배우 송중기가 물었다 "제가 상남자인가요?"
```

[c:]-[data]-[python]-[scraping] 05_google_images_selenium.py

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
import time

browser = webdriver.Chrome()
browser.maximize_window()

url = "https://www.google.com"
browser.get(url)

#검색어 송중기 입력후 엔터
elem=browser.find_element_by_name("q")
elem.send_keys("송중기")
elem.send_keys(Keys.ENTER)

#이미지버튼 클릭
browser.find_element_by_link_text("이미지").click()
time.sleep(2)

#지정한 위치로 스크롤 내리기
#browser.execute_script("window.scrollTo(0, 1080)") #1920 x 1080
#browser.execute_script("window.scrollTo(0, 2080)")

#화면 가장 아래로 스크롤 내리기
#browser.execute_script("window.scrollTo(0, document.body.scrollHeight)")

interval = 2
prev_height = browser.execute_script("return document.body.scrollHeight")

#반복수행
while True:
    browser.execute_script("window.scrollTo(0, document.body.scrollHeight)")
    time.sleep(interval)

    curr_height = browser.execute_script("return document.body.scrollHeight")
    if curr_height == prev_height:
        break

    prev_height = curr_height

print("스크롤 완료")

from bs4 import BeautifulSoup
soup = BeautifulSoup(browser.page_source, "lxml")

images = soup.find_all("div", attrs={"class": "isv-r PNCib MSM1fd BUooTd"})
print(len(images))

for image in images:
     title = image.find("span", attrs={"class": "OztcRd"}).get_text()
     link = image.find("a", attrs={"class": "VFACy kGQAp sMi44c lNHeqe WGvvNb"})["href"]
     print(f"title: {title}")
     print(f"link: {link}")
     print("-" * 120)

browser.quit()
```

[c:]-[data]-[python]-[scraping] 05_google_images_selenium.py 실행결과

```
400
...
------------------------------------------------------------------------------------------------------------------------
title: 문화]한 달 전까지만 해도...송중기·송혜교 이혼이 더 충격적인 이유 | YTN
link: https://ytn.co.kr/_ln/0106_201906271156331025
------------------------------------------------------------------------------------------------------------------------
title: 송중기, 이혼과 함께 했던 일…심정은? "어려울 때도 있었지만 내 욕심이 너무 컸다"
link: https://m.fntimes.com/html/view.php?ud=20191122022123757659eb9bc7c2_18
------------------------------------------------------------------------------------------------------------------------
...
```

4) Headless 크롬

크롬 브라우저를 띄우지 않고 빠르게 스크랩핑 작업이 기능하다.

[c:]-[data]-[python]-[scraping] 05_google_images_selenium.py

```
...

options = webdriver.ChromeOptions()
options.headless = True
options.add_argument("window-size=1920x1080")
browser = webdriver.Chrome(options=options)
...

print("스크롤 완료")
browser.get_screenshot_as_file("google_images.png")
...
```

HeadlessChrome이고 user agent값이 설정되지 않은 경우 User agent값이 날아가서 브라우저의 접속을 막을 수 있다.

[c:]-[data]-[python]-[scraping] 05_headless_usergent.py

```
from selenium import webdriver

options = webdriver.ChromeOptions()
options.headless = True
options.add_argument("window-size=1920x1080")

browser = webdriver.Chrome(options=options)
browser.maximize_window()

#구글에서 user agent string 검색후 what is my user agent 클릭
url="https://www.whatismybrowser.com/detect/what-is-my-user-agent/"
browser.get(url)

detected_value = browser.find_element_by_id("detected_value")
print(detected_value.text)

browser.quit()
```

[c:]-[data]-[python]-[scraping] 05_headless_usergent.py 실행결과

```
Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) HeadlessChrome/102.0.5005.62
Safari/537.36
```

user agent값이 설정된 경우에는 user agent값이 정상적 적용되어 출력된다.

[c:]-[data]-[python]-[scraping] 05_headless_usergent.py

```
from selenium import webdriver

options = webdriver.ChromeOptions()
options.headless = True
options.add_argument("window-size=1920x1080")
options.add_argument("user-agent=Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/102.0.5005.62 Safari/537.36")

browser = webdriver.Chrome(options=options)
browser.maximize_window()
...
```

[c:]-[data]-[python]-[scraping] 05_headless_usergent.py 실행결과

```
Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/102.0.5005.62 Safari/537.36
```

06. 프로젝트

무분별한 웹 크롤링이나 웹 스크래핑은 대상 서버에 부하가 걸려 계정/IP가 차단된다. 또한 이미지나 텍스트 등 데이터 무단 활용 시 저작권 등 침해요소가 있어 법적 제재를 받을 수 있다. robots.txt 파일은 법적 효력은 없지만 대상 사이트에서 크롤링을 금지하는 권고 사항이다.

- **네이버 부동산 사이트에서 '청라자이' 검색 결과를 출력**

[c:]-[data]-[python]-[scraping] 06_naver.land.py

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
import time

browser = webdriver.Chrome()
browser.maximize_window()

url = "https://land.naver.com/"
browser.get(url)

elem = browser.find_element_by_id("queryInputHeader")
elem.send_keys("청라자이")
time.sleep(2)

elem.send_keys(Keys.ENTER)
time.sleep(2)

from bs4 import BeautifulSoup
soup = BeautifulSoup(browser.page_source, "lxml")

with open("ralty.html", "w", encoding="utf8") as f:
    f.write(soup.prettify())

elems = soup.find_all("div", attrs={"class":"item"})
print(len(elems))

for index, e in enumerate(elems):
    title = e.find("div", attrs={"class": "title"}).get_text()
    address = e.find("div", attrs={"class": "address"}).get_text()
    info = e.find("div", attrs={"class": "info_area"}).get_text()
    print("========== 정보 {} ==========".format(index+1))
    print(f"{index}.{title}")
    print(address)
    print(info)

browser.quit()
```

[c:]-[data]-[python]-[scraping] 06_naver.land.py 실행결과

```
4
========== 정보 1 ==========
0.청라자이
인천시 서구 청라동
아파트884세대총19동2010.06.123.5m² ~278.12m²
========== 정보 2 ==========
1.청라힐스자이
대구시 중구 남산동
아파트분양권947세대총13동2023.01.77.33m² ~129.68m²
========== 정보 3 ==========
2.청라파크자이더테라스(2블럭)
인천시 서구 청라동
아파트376세대총18동2016.03.102.15m² ~112.64m²
========== 정보 4 ==========
3.청라파크자이더테라스(1블럭)
인천시 서구 청라동
아파트270세대총17동2016.03.102.32m² ~112.83m²
```

- [네이버 검색]-[서울 날씨] 검색 후 날씨정보 아래와 같이 출력

```
[오늘의 날씨]
---------------------------------------------------
어제보다 7° 낮아요 흐림
현재: 1, 체감: 23.1° , 습도: 75%
오전 강수확률: 70%
오후 강수확률: 60%
미세먼지: 좋음
초미세먼지: 좋음
---------------------------------------------------
```

[c:]-[data]-[python]-[scraping] 06_naver_weather.py

```
import requests
from bs4 import BeautifulSoup

def create_suop(url):
    res = requests.get(url)
    res.raise_for_status()
    soup = BeautifulSoup(res.text, "lxml")
    return soup

def scrape_weather():
    print("[오늘의 날씨]")
    url = "https://search.naver.com/search.naver?where=nexearch&sm=top_hty&fbm=1&ie=utf8&query=%EC%84%9C%EC..."
    soup = create_suop(url)

    summary=soup.find("p", attrs={"class":"summary"}).get_text()

    #현재 온도
    temp_text=soup.find("div", attrs={"class":"temperature_text"})
    temp_contents=temp_text.strong.contents[1]

    #체감온도, 습도
    sum_list = soup.find("dl", attrs={"class":"summary_list"}).find_all("dd",attrs={"class":"desc"})

    #오전, 오후 강수량
    cell_weather = soup.find("div", attrs={"class":"cell_weather"}).find_all("span",attrs={"class":"rainfall"})

    #클래스명이 className1이거나 className2 검색 attrs={"class":["className1", "className2"]}
    #클래스명이 className이고 아이디가 idName 검색 attrs={"class":"className", "id":"idName"}
    #클래스명이 className이고 텍스트내용이 미세먼지이거나 초미세먼지 검색 attrs={"class":"className"}, text=["미세먼지, "초미세먼지]

    #미세먼지
    item_today= soup.find("li", attrs={"class":"item_today level2"}).find_all("span", attrs={"class":"txt"})
    print("-"*50)
    print(summary)
    print("현재: {}, 체감: {}, 습도: {}".format(temp_contents[1], sum_list[0].get_text(), sum_list[1].get_text() ))
    print("오전 강수확률: {}".format(cell_weather[0].get_text()))
    print("오후 강수확률: {}".format(cell_weather[1].get_text()))
    print("미세먼지: {}".format(item_today[0].get_text()))
    print("초미세먼지: {}".format(item_today[0].get_text()))
    print("-"*50)

if __name__ == "__main__":
    scrape_weather()
```

[c:]-[data]-[python]-[scraping] 06_naver_weather.py 실행결과

```
[오늘의 날씨]
---------------------------------------------------
어제보다 7° 낮아요 흐림
현재: 1, 체감: 23.1° , 습도: 75%
오전 강수확률: 70%
오후 강수확률: 60%
미세먼지: 좋음
초미세먼지: 좋음
---------------------------------------------------
```

- **[네이버 뉴스]-[IT/과학] 검색 후 뉴스 정보 아래와 같이 출력**

[IT/과학 헤드라인 뉴스]
---
1. "반도체 인재" 양성 급한데 4대 과기원 반도체학과 확대
  (https://...)
2. 7월부터 "메타버스 근무제" 시행 카카오
  (https://...)
3. AI로 만성질환관리 돕는다 SKT 국민건강보험공단과 맞손
  (https://...)

[c:]-[data]-[python]-[scraping] 06_naver_news.py

```
from selenium import webdriver
from selenium.webdriver.common.by import By
import time

options = webdriver.ChromeOptions()
options.headless = True

browser = webdriver.Chrome(options=options)
browser.maximize_window()

url = "https://news.naver.com"
browser.get(url)

#IT/과학 메뉴 클릭
browser.find_element_by_link_text("IT/과학").click()
time.sleep(2)

#스크랩핑
from bs4 import BeautifulSoup
soup = BeautifulSoup(browser.page_source, "lxml")

#스크랩핑 결과를 파일에 임시저장
with open("news.html", "w", encoding="utf8") as f:
    f.write(soup.prettify())

print("[IT/과학 헤드라인 뉴스]")
print("-"*150)

#헤드라인 뉴스 3개만 출력
news_list = soup.find("div",attrs={"id":"main_content"}).find_all("h2",attrs={"class":"cluster_head_topic"}, limit=3)

for index, news in enumerate(news_list):
    title = news.find("a").get_text().strip()
    link = url + news.find("a")["href"]
    print("{}. {}".format(index+1, title))
    print("  ({})".format(link))

print("-"*150)
browser.quit()
```

[c:]-[data]-[python]-[scraping] 06_naver_news.py 실행결과

```
[IT/과학 헤드라인 뉴스]
------------------------------------------------------------------------------------------------------------------------------------------------------
1. "반도체 인재" 양성 급한데 4대 과기원 반도체학과 확대
  (https://news.naver.com/main/clusterArticles.naver?id=c_202205301410_00000007&mode=LSD&mid=shm&sid1=105&oid=008&aid=0004752872)
2. 7월부터 "메타버스 근무제" 시행 카카오
  (https://news.naver.com/main/clusterArticles.naver?id=c_202205301720_00000001&mode=LSD&mid=shm&sid1=105&oid=031&aid=0000675754)
3. AI로 만성질환관리 돕는다 SKT 국민건강보험공단과 맞손
  (https://news.naver.com/main/clusterArticles.naver?id=c_202205310900_00000017&mode=LSD&mid=shm&sid1=105&oid=011&aid=0004060048)
------------------------------------------------------------------------------------------------------------------------------------------------------
```

- **네이버의 오늘의 영단어, 해커스의 오늘의 영어회화 정보 출력**

[c:]-[data]-[python]-[scraping] 06_naver_english.py

```
import requests
from bs4 import BeautifulSoup
import re

def create_soup(url):
    res = requests.get(url)
    res.raise_for_status()
    soup = BeautifulSoup(res.text, "lxml")
    return soup

#[네이버 검색]-[해커스]-[해커스 토익]-[기초영화/회화]-[매일영어회화 학습]
def scrape_hackers():
    print("[오늘의 영어 회화]")
    url="https://www.hackers.co.kr/?c=s_eng/eng_contents/...&logger_kw=haceng_submain_lnb_eng_l_others_english"
    soup = create_soup(url)
    sentences = soup.find_all("div", attrs={"id":re.compile("^conv_kor_t")})

    print("(영어지문)")
    for sentence in sentences[len(sentences)//2:]:  #8문장이 있다고 가정할 때, index 기준 4~7까지
        print(sentence.get_text().strip())

    print("(한글지문)")
    for sentence in sentences[:len(sentences)//2]:  #8문장이 있다고 가정할 때, index 기준 0~3까지
        print(sentence.get_text().strip())

#[네이버 검색]-[네이버영어]
def scrape_naver():
    print("[오늘의 영어 단어]")
    url = "https://search.naver.com/search.naver?where=nexearch&sm=top_hty&fbm=1&ie=utf8&query=..."
    soup = create_soup(url)
    words = soup.find("ul", attrs={"class":"word_lst _sap_list"}).find_all("li")

    for word in words:
        eng = word.find("a").get_text()
        kor = word.find("span").get_text()
        print("{} : {}".format(eng, kor))

if __name__ == "__main__":
    scrape_naver()
    print("-" * 50)
    scrape_hackers()
```

[c:]-[data]-[python]-[scraping] 06_naver_english.py 실행결과

```
[오늘의 영어 단어]
idle : 게으른, 나태한
hasty : 서두른 (좋지 못한 결과를 초래함을 나타냄)
baleful : 악의적인, 해로운
complain : 불평[항의]하다
insulation : 절연[단열/방음] 처리[처리용 자재]
--------------------------------------------------
[오늘의 영어 회화]
(영어지문)
Mrs. Flores : And just what is your explanation?
Rob : Actually, I've completed the research but I still need time to write up the report.
Mrs. Flores : And how long do you expect that to take?
Rob : Just another few days.
(한글지문)
Mrs. Flores : 이유가 무엇인가요?
Rob : 사실 연구를 끝냈습니다만, 보고서를 작성하기 위한 시간이 더 필요해요.
Mrs. Flores : 얼마나 오랫동안 걸릴 거 같아요?
Rob : 단 며칠이요.
```