Crawling

#라이브러리 설치

```
from bs4 import BeautifulSoup
import urllib.request
import requests
```

url html 읽기

```
url = 'https://www.naver.com'
html=urllib.request.urlopen(url)
html.read()
```

사진 저장하기 (영상, 텍스트 모두 가능)

```
crawlier='https://d1sr4ybm5bj1wl.cloudfront.net/img/2017-01-19-HowToMakeWebCrawler/crawler.jpg'
```

```
save_name='크롤러.jpg'
```

```
urllib.request.urlretrieve(crawlier, save_name)
```

```
('크롤러.jpg', <http.client.HTTPMessage at 0x27aa5d41130>)
```

```
soup = beatifulsoup(html, html.parser)
*parser - 구문 분석
└import lxml을 하면 변수.parser할 필요 없음
└soup=BeautifulSoup(html, 'lxml')
find() - 원하는 구문만 출력하는 함수
soup.find('ul') - ul태그만 출력
findAll() 두 개 이상의 구문 출력시 사용
find('tag',{'class':'class명'}) - 해당 클래스만 출력
```

영화 평점 크롤링 절차

```
#영화 평점 분석
movie='https://movie.naver.com/movie/sdb/rank/rmovie.nhn?sel=cur&date=20210427' #url 변수
movie_html=req.urlopen(movie) #url 읽기
movie_soup=BeautifulSoup(movie_html,'lxml') #구문 분석
movie_title=movie_soup.findAll('td',{'class':'title'}) #class=title 모두 불러오기
movie_title=movie_soup('div',{'class':'tit5'}) #div의 tit5만 불러오기
for i in movie_title:
    print(i.find('a').string) #영화 제목 출력

#영화 평점 분석 - 2 (평점 불러오기)
points = movie_soup.findAll('td',{'class':'point'})
title=[]
for i in movie_title:
```

```
    title.append(i.find('a').string)
point=[]
for i in points:
    point.append(i.string)

#영화, 평점 불러오기
for i,j in zip(title, point):
    print(i,j)
```

requests VS urllib.request 차이

HTTP method를 따르는 지에 대한 유무의 차이이다. (보통 requests를 많이 사용한다.)

*request가 HTTP method를 따르기 때문에 더욱 직관적이다.

#기상청 날씨 데이터 크롤링

```
import requests
from bs4 import BeautifulSoup
import lxml

#기상청 날씨 데이터 분석
weather=requests.get('http://www.weather.go.kr/weather/observation/currentweather.jsp')
weather=weather.text #object error 발생 시 .content or .text를 넣어주어야 함
#content = html 코드에서 한글이 숫자로 보임(encoding)
#text = html 코드에서 한글이 글자 그대로 출력(decoding)
soup=BeautifulSoup(weather,'lxml')

table = soup.find('table',{'class':'table_develop3'}) #기상청 지역별 날씨 정보 테이블 전체 가져오기
data=[]                                               #데이터를 넣어줄 빈 리스트 생성
for i in table.findAll('tr'):                         #tr(하나의 행)에서 td를 리스트로 묶어 tds 생성
    tds = list(i.findAll('td'))

    for j in tds:                                     #tds가 리스트이기 때문에 인덱스 값 하나씩 순회해야함
        if j.find('a'):
            point=j.find('a').text                    #a(지역정보)를 찾은 후 point 변수에 저장
            temperature=tds[5].text                   #temperature(기온)의 인덱스 값[5]를 지정
            humidity=tds[10].text                     #humidity(습도)의 인덱스 값[10]를 지정
            data.append([point,temperature,humidity]) #빈 리스트에 append

df_data=pd.DataFrame(data, columns=['지역','기온(℃)','습도(%)'])
df_data

#데이터프레임 저장하기
df_data.to_csv('weather.csv', index=False)

#데이터프레임 불러오기
weather_df=pd.read_csv('weather.csv')
weather_df

#서울의 기온 습도만 보기
seoul=weather_df['지역']=='서울'
weather_df[seoul]
```

Selenium - 동적페이지 크롤링 라이브러리

```
#동적페이지 - 서버는 추가적인 처리 과정 이후 클라이언트에게 응답을 보낸다.
#방문자와 상호작용을 한다. (사용자의 행동에 의해서 페이지가 재생성된다.)
#동적페이지 크롤링 방법1 - OpenAPI (서버 접속 폭주를 방지하고 독립적으로 크롤링하기 위함)
#동적페이지 크롤링 방법2 - Web Driver + Selenium (OpenAPI를 제공하지 않는 경우)
```

```
#Web Driver - 웹브라우저 UI 테스팅 툴 (동적 로딩 지원 및 브라우징 자동화)
#동적페이지 구성 파악하는 방법 (개발자 도구(F12)의 dev tools로 파악 가능)
#(개발자 도구(F12) -> settings -> Debugger -> Disable Java script)
#설정 후 동작하지 않은 것들이 동적기능을 하는 페이지들이다.

#라이브러리 설치
from selenium import webdriver as wd
import requests
import lxml
import time

#크롬 드라이버 설치 후 vscode를 사용할 폴더에 넣어주기

#드라이버 및 웹페이지 로드
url='http://tour.interpark.com'
driver=wd.Chrome(executable_path='chromedriver.exe') #웹드라이버 load
driver.get(url) #웹페이지 로드(기존의 웹페이지와 독립적 형태)

#웹페이지 로드 대기시간 설정 - 로드 대기시간 설정 필요할 때마다 설정해주기
driver.implicitly_wait(10)

#인터파크 검색기능 이용하기

#키워드 및 검색버튼 변수 설정
searchbox='SearchGNBText'  #검색창 id
keyword='스위스'            #검색할 키워드
searchbtn='search-btn'     #검색버튼 id
infobtn='li_R'             #해외여행 카테고리 버튼 id

#키워드 입력 및 검색버튼 클릭
driver.find_element_by_id(searchbox).send_keys(keyword) #키워드 입력하기
driver.find_element_by_class_name(searchbtn).click()    #검색버튼 클릭
※class가 여러개인 경우 element->elements_by_class를 이용해야 한다.
driver.find_element_by_id(infobtn).click() #해외여행 카테고리 클릭

#페이지 이동
for i in range(1,8): #마지막 페이지까지 range설정 (총 7페이지)
    driver.execute_script("searchModule.SetCategoryList({},'')".format(i)) #driver.execute_script - 자바스크립트 명령어 실행 함수
    time.sleep(3)                                                           #페이지 이동 스크립트 실행 대기 시간
    print("{}번 째 페이지로 이동".format(i))
    boxitems=driver.find_elements_by_css_selector('.panelZone>.oTravelBox>.boxList>li') #여행 패널 -> 해외여행 패널 -> 해외여행 목록
    for li in boxitems:
        print('상품명',li.find_element_by_css_selector('h5.proTit').text)               #여행 제목 css의 문자만 출력
        print('가격',li.find_element_by_css_selector('.proPrice').text.split('원')[0])  #가격의 숫자만 출력
```