웹크롤링vs웹스크랩핑

웹크롤링과 웹스크랩핑의 차이를 이해하고, 관련 모듈의 사용법을 알아보도록 하겠습니다.

1. 정의

▼ 웹 크롤링

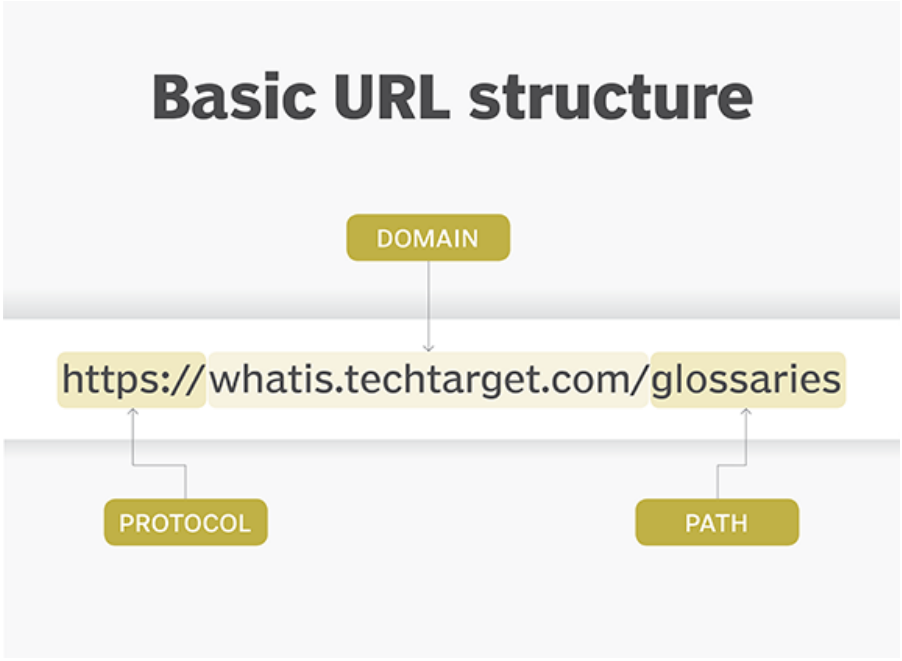
웹 페이지의 하이퍼링크를 순회하면서 웹 페이지를 다운로드하는 작업을 의미한다.

▼ 웹스크랩핑

1. 다운로드한 웹 페이지에서 필요한 콘텐츠를 추출하는 작업
2. 웹 페이지를 구성하고 있는 HTML 태그의 콘텐츠나 속성의 값을 읽는 작업

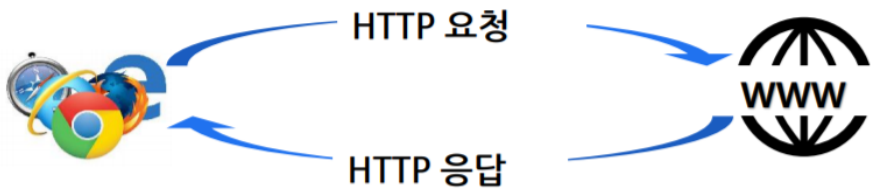
2. 주요용어

- URL(Uniform Resource Locator) : 네트워크 상에서 자원이 어디 있는지를 알려주기 위한 규약



출처 > https://images.app.goo.gl/pDybuwdiYMNru9kq9

- URI(Uniform Resource Identity) : 웹 사이트에 요청하고자 하는 대상의 패스정보와 파일명으로 구성되어 있고, 파일명이 생략되면 디폴트로 index.html 사용함.
- HTTP(HyperText Transfer Protocol) : 웹상에서 클라이언트와 서버 간에 정보를 주고받을 수 있는 통신 규약(프로토콜)
  - URL 문자열을 직접 입력하거나, 하이퍼링크 텍스트 또는 이미지를 클릭하여 HTML 문서를 주고받는 데 사용
  - 디폴트로 80번 포트 사용
  - 다른 포트 번호를 사용하는 웹 서버에 요청 시 도메인명 뒤에 : 기호와 함께 포트 번호 지정
  - 웹 클라이언트에서 웹 서버에 HTTP 요청을 전달할 때 요청 방식 명시
  - 일반적으로 GET,POST 2가지 방식 사용



GET 방식 → 브라우저에서 직접 요청하려는 페이지의 URL 문자열을 입력하여 요청
POST 방식 → Query 문자열이 요청 바디에 따로 담겨서 전달되므로, 요청 URL 문자열에서는 볼 수 없음

3. 사용 모듈

▼ urllib, urllib2, urllib3

- URL 문자열을 가지고 HTTP 요청을 수행
- urlopen() 함수를 사용하여 웹 서버에 페이지를 요청하고, 서버로부터 받은 응답을 저장하여 응답 객체(http.client.HTTPResponse)를 반환
- 버전에 따른 차이, 제공하는 메서드 차이, 표준 or 3rd Party 모듈

```
# 예제
from urllib import request
import os
# 1. 텍스트 데이터
# res = request.urlopen('https://www.google.co.kr')

# 2. 바이너리 데이터 (아래 주석을 해제!)
res = request.urlopen('https://www.google.co.kr/images/branding/googlelogo/2x/googlelogo_color_272x92dp.png')


# 3. 텍스트 데이터 읽어서 변수에 기록
# res_text = res.read()
res_img = res.read()

# 4. 텍스트 데이터 출력
# print(res_text)
path = os.getcwd()

if '/static' not in path:
    os.mkdir('static')
    os.chdir('static')
else:
    os.chdir('static')

with open('google_logo.png', 'wb') as f:
    f.write(res_img)
print('completed')

'''
예외처리
from urllib import request
import os
# 1. 텍스트 데이터
# res = request.urlopen('https://www.google.co.kr')

# 2. 바이너리 데이터 (아래 주석을 해제!)
res = request.urlopen('https://www.google.co.kr/images/branding/googlelogo/2x/googlelogo_color_272x92dp.png')


# 3. 텍스트 데이터 읽어서 변수에 기록
# res_text = res.read()
res_img = res.read()

# 4. 텍스트 데이터 출력
# print(res_text)
path = os.getcwd()

try:
    if '/static' not in path:
        os.mkdir('static')
        os.chdir('static')
except:
    os.chdir('static')
else:
    with open('google_logo.png', 'wb') as f:
        f.write(res_img)
```

```
        print('completed')
'''
```

▼ requests

- Kenneth Reitz에 의해 개발된 파이썬 라이브러리
- HTTP 프로토콜과 관련된 기능 지원
- https://2.python-requests.org/en/master/
- urllib vs requests

```
'''
import requests
url = 'https://api.github.com'
res = requests.get(url, auth=('userId','userPw'))
print(res.status_code)
print(res.headers['Content-Type'])
print(res.encoding)
print(res.json())
'''

import requests
# 2020.1 이후 변경(네이버)

# 아래 주소가 메인페이지 내부에서 호출되는 실시간 검색어 데이터를 넘겨주는 주소
# requests.get("주소").json() 을 하면 데이터를 json 형태로 받아올 수 있습니다.

# 아래 주소를 직접 브라우저에서 접속해보시기 바랍니다.
json = requests.get('https://www.naver.com/srchrank?frm=main').json()

# json 데이터에서 "data" 항목의 값을 추출
ranks = json.get("data")

# 해당 값은 리스트 형태로 제공되기에 리스트만큼 반복
print('--------실시간 급상승 검색어--------')
for r in ranks:
    # 각 데이터는 rank, keyword, keyword_synomyms
    rank = r.get("rank")
    keyword = r.get("keyword")
    print(rank, keyword)
```

urllib vs requests

Aa urllib	requests
인코딩하여 바이너리 형태로 데이터 전송	딕셔너리 형태로 데이터 전송
데이터 전달 방식에 따라 GET 요청, POST 요청을 구분	요청 메서드(GET, POST)를 명시하여 요청

▼ BeautifulSoup4

Beautiful Soup은 HTML과 XML 파일에서 데이터를 추출하기 위한 파이썬 라이브러리이다. 이 기능은 즐겨 찾는 구문 분석기와 함께 작동하여 구문 분석 트리를 탐색, 검색 및 수정할 수 있는 방법을 제공합니다.

```
# 네이버 로또 번호 가져오기

import requests
from bs4 import BeautifulSoup
start = 900
end = 947
for count in range(start, end):
    url = f'https://search.naver.com/search.naver?sm=tab_drt&where=nexearch&query={count}회로또'
    equal_ind = url.rfind('=')
    title = url[equal_ind + 1:]
    res = requests.get(url)
    if res.status_code != 200:
        print('connect error!')
        exit()

    soup = BeautifulSoup(res.text, 'html.parser')
    # print(soup.prettify())
    numbers = soup.select('.num_box > span')
    output = {}
    output['title'] = title
    num = [number.text for number in numbers]
    # for number in numbers:
    #     num.append(number.text)
```

```
    output['number'] = str(num)
    with open('lotto2.txt', 'a', encoding='utf-8') as f:
        f.write()
        f.write('\n')

# 다음(daum) 로또 번호도 가져오기 해보세요!

# 버스노선 조회, 기상청 날씨 조회등 공공데이터를 활용

'''
```

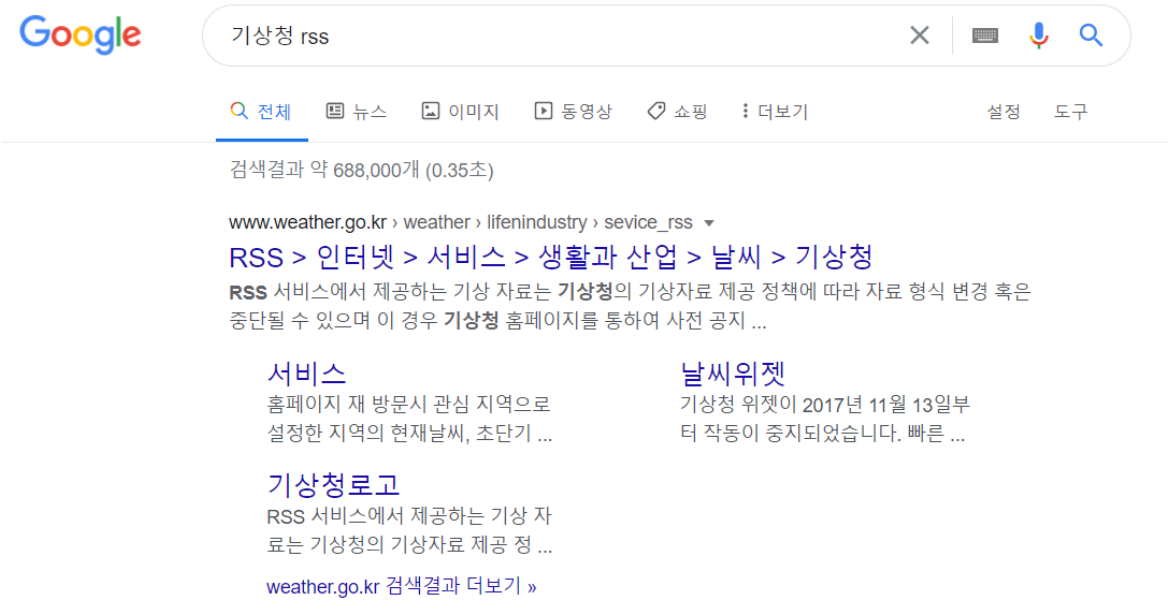
※ 파이썬 v3.9.1 표준 라이브러리https://docs.python.org/ko/3/library/index.html)

https://docs.python.org/ko/3/library/index.html

4. 응용예제

▼ Beautiful Soup 모듈로 날씨정보 가져오기

4-1. 구글 검색 > 기상청 rss



4-2. RSS 주소 얻기 (지역등 선택)

4-3. 중기예보 > 전라남도(광주/전남) - RSS ▶ (클릭), 주소복사 [CTRL]+[C]

4-3. 웹브라우저 > 주소창 > (복사한 RSS 주소) 붙여넣기, 확인

This XML file does not appear to have any style information associated with it. The document tree is shown below.


```
▼<rss version="2.0">
  ▼<channel>
     <title>기상청 육상 중기예보</title>
     <link>http://www.kma.go.kr/weather/forecast/mid-term_05.jsp</link>
     <description>기상청 날씨 웹서비스</description>
     <language>ko</language>
     <generator>기상청</generator>
     <pubDate>2021년 01월 24일 (일)요일 18:00</pubDate>
    ▼<item>
       <author>기상청</author>
       <category>육상중기예보</category>
       <title>전라남,북도 육상 중기예보 - 2021년 01월 24일 (일)요일 18:00 발표</title>
       <link>http://www.kma.go.kr/weather/forecast/mid-term_05.jsp</link>
       <guid>http://www.kma.go.kr/weather/forecast/mid-term_05.jsp</guid>
      ▼<description>
        ▼<header>
           <title>전라남,북도 육상중기예보</title>
           <tm>202101241800</tm>
          ▼<wf>
             <![CDATA[ ○ (날씨) 28일(목)은 비 또는 눈, 29일(금)은 눈이 오겠습니다. 그 밖의 날은 구름이 많겠습니다.<br />○ (기온) 이번
             겠습니다.<br /> 특히, 29일(금)~30일(토)는 오늘보다 기온이 매우 낮아 춥겠습니다.<br />○ (해상) 바다의 물결은 28일(목)~29일(
          </wf>
        </header>
        ▼<body>
          ▼<location wl_ver="3">
             <province>광주·전라남도</province>
             <city>광주</city>
            ▼<data>
               <mode>A02</mode>
               <tmEf>2021-01-27 00:00</tmEf>
               <wf>구름많음</wf>
               <tmn>2</tmn>
               <tmx>9</tmx>
               <reliability/>
               <rnSt>20</rnSt>
             </data>
```


4-4. 파이참 > 새 파일 > weather.py 생성, 아래의 코드 입력해보기!

```
import requests
from bs4 import BeautifulSoup

url = 'http://www.kma.go.kr/weather/forecast/mid-term-rss3.jsp?stnId=156'

res = requests.get(url)
if res.status_code != 200:
    print('error!')
    exit()

soup = BeautifulSoup(res.text, 'html.parser')

# with open('weather.html', 'w', encoding='utf-8') as f:
#     f.write(str(soup))

# soup.prettify()
# print(soup)
for location in soup.select('location'):
    print('도시 :', location.select_one('city').string)
    print('날씨 :', location.select_one('wf').string)
    print('최저기온 :', location.select_one('tmn').string)
    print('최고기온 :', location.select_one('tmx').string)
    print()
```

▼ Beautiful Soup + Flask 사용해보기

- Flask는 파이썬 웹 개발시 사용하는 마이크로 프레임워크로 소형화된 프레임 워크 입니다. 이에 반해 Django(장고) 라는 프레임워크는 서버사이트 웹프레임워크 입니다. 거대한 프레임 워크는 구동이 느립니다(일반적으로..아닐수도 있고..성능은 계속 향상되니까요)
- 이미 기존에 설치한 Flask 모듈이 있어야 합니다. 만약, 설치 전 이거나, 새로운 가상환경이라면 pip install flask 명령을 터미널 [ALT]+[F12]에서 실행해야 합니다.
- (터미널에서) pip list 라는 명령으로 현재 가상환경에 설치된 모듈 목록을 확인할 수 있습니다.

```
(python_basic) D:\python_venv\python_basic\Scripts>pip list
Package                Version
---------------------- ---------
beautifulsoup4         4.9.3
certifi                2020.12.5
chardet                4.0.0
google-images-download 2.8.0
idna                   2.10
pip                    20.3.3
requests               2.25.1
selenium               3.141.0
setuptools             49.2.1
soupsieve              2.1
urllib3                1.26.2
```

- 설치되어 있지 않다면, 설치해볼까요?!

```
(python_basic) D:\python_venv\python_basic\Scripts>pip install flask
Collecting flask
  Using cached Flask-1.1.2-py2.py3-none-any.whl (94 kB)
Collecting click>=5.1
  Using cached click-7.1.2-py2.py3-none-any.whl (82 kB)
Collecting itsdangerous>=0.24
  Using cached itsdangerous-1.1.0-py2.py3-none-any.whl (16 kB)
Collecting Jinja2>=2.10.1
  Using cached Jinja2-2.11.2-py2.py3-none-any.whl (125 kB)
Collecting MarkupSafe>=0.23
  Using cached MarkupSafe-1.1.1-cp38-cp38-win_amd64.whl (16 kB)
Collecting Werkzeug>=0.15
  Using cached Werkzeug-1.0.1-py2.py3-none-any.whl (298 kB)
Installing collected packages: MarkupSafe, Werkzeug, Jinja2, itsdangerous, click, flask
Successfully installed Jinja2-2.11.2 MarkupSafe-1.1.1 Werkzeug-1.0.1 click-7.1.2 flask-1.1.2 itsdangerous-1.1.0
```

- 준비되었다면, flask와 BeautifulSoup, requests를 함께 적용해보겠습니다. 자..잠깐!, 아래 기본 Flask 모듈 사용하기를 다시 한번 살펴보세요(안까먹으셨기를..)

```python
# Flask 모듈 사용하기
# 1.
from flask import Flask
app = Flask(__name__)

@app.route('/')
def hello():
    return '<h1>hello world</h1>'

if __name__ == '__main__':
    app.run()

'''
from flask import Flask, render_template
# micro framework - Flask vs Django
# render_template는 templates 기능을 사용하기 위한 함수
app = Flask(__name__)

# decorator : @
# 라우팅 - 복잡한 URL를 쉽게 처리하는 기능
# <name> : 일종의 변수

@app.route('/')
def index():
    return render_template('index.html')

# str 타입은 파라미터의 기본값(생략가능)
```

```
@app.route('/page/<name>')
def show_page(name):
    return '지금 보고있는 페이지는 %s 입니다' % name

# int 타입은 파라미터
@app.route('/post/<int:postNum>')
def show_post(postNum):
    return '지금 보여지는 포스트 넘버는 %d 입니다' % postNum

# float 타입의 파라미터
@app.route('/calc/<float:num>')
def show_calc(num):
    return '요청하신 값은 %f 입니다' % num

if __name__ == '__main__':
    app.run()
'''
```

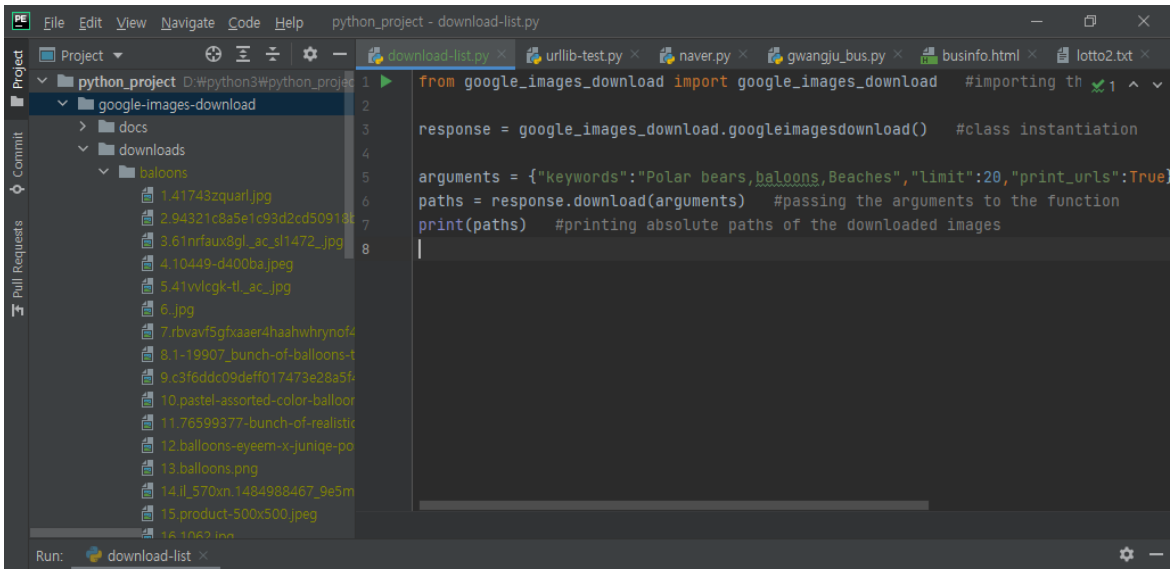
- requests + BeautifulSoup + Flask 완성하기

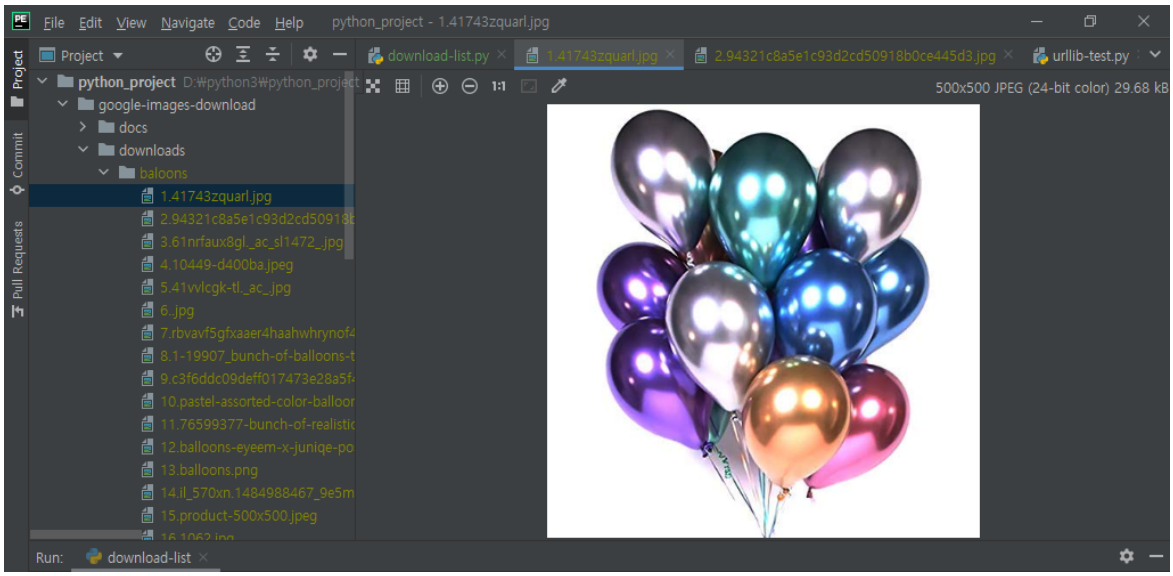
```
from flask import Flask
import requests
from bs4 import BeautifulSoup

app = Flask(__name__)
@app.route('/')

def index():
    url = 'http://www.kma.go.kr/weather/forecast/mid-term-rss3.jsp?stnId=156'
    res = requests.get(url)
    if res.status_code != 200:
        print('error!')
        exit()

    soup = BeautifulSoup(res.text, 'html.parser')
    locations = soup.select('location')
    output = ''
    output += '<h1>광주/전남 중기예보</h1>'
    for location in locations:
        output += '<h2>{}</h2>'.format(location.select_one('city').string)
        output += '<p>날씨 : {}</p>'.format(location.select_one('wf').string)
        output += '<p>최저기온 : {}</p>'.format(location.select_one('tmn').string)
        output += '<p>최고기온 : {}</p>'.format(location.select_one('tmx').string)
        output += '<p>강수확률 : {}</p>'.format(location.select_one('rnSt').string)
    return output

if __name__ == '__main__':
    app.run()
```

▼ google images download

```
# 참고 URL
# https://github.com/hardikvasa/google-images-download
# https://google-images-download.readthedocs.io/en/latest/index.html
# google chromium 업데이트 해야지 가능!
# 저작권 있는 자료까지 다운로드 됨

# google update로 현재 막힘 (2020.08)
# Unfortunately all 10 could not be downloaded because some images were not downloadable. 0 is all we got for this search f

# 2021.01.26 update
# 1. 기존 google-images-download 모듈 제거
# 2. 업데이트된 모듈 설치 or 아래 URL을 git clone
# git clone https://github.com/Joeclinton1/google-images-download.git
# 3. 파일 생성 (단, google-images-download 폴더에 저장)
'''
from google_images_download import google_images_download   #importing the library

response = google_images_download.googleimagesdownload()   #class instantiation

arguments = {"keywords":"Polar bears,baloons,Beaches","limit":20,"print_urls":True}   #creating list of arguments
paths = response.download(arguments)   #passing the arguments to the function
print(paths)   #printing absolute paths of the downloaded images
'''
# 4. 실행 / 확인 (MAX : 400)
```

<실행결과>





5. 그밖에

- scrpy (framework)
- selenium