웹스크래핑

데이터 수집

- **데이터 수집방법**
  - 파일 다운로드가 가능한가?
  - API로 제공되는가?

    : 구글 API 등

    * 업체 및 기관 제공
  - 웹스크래핑

    : 우리가 캐오는 것

웹스크래핑 절차

- **웹스크래핑 절차 [edureka]**

  1. Find the URL that you want to scrape.
  2. Inspecting the Page.
  3. Find the data you want to extract.
     : 표, 텍스트, 영상 등
  4. Write the code.
     : Python & Library
     : 웹 페이지 소스 검토
  5. Run the code and extract the data.
  6. Store the data in the required format.
     : csv / DB

웹스크래핑 도구

- 웹스크래핑 도구 - **Library**
  - BeautifulSoup

    : Beautiful Soup is a Python package for parsing HTML and XML documents. It creates parse trees that is helpful to extract the data easily.
  - requests
  - lxml
  - Selenium

    : Selenium is a web testing library. It is used to automate browser activities.
  - ChromeDriver

웹스크래핑은 합법인가?

- 웹사이트에 따라 정책이 다르다
  - 웹사이트에서 제공하는 “robots.txt” 파일 확인
  - 스크래핑하려는 웹사이트 URL에 "/robots.txt"를 붙이면 확인 가능
  - 예) Flipkart.com 웹사이트

```
User-agent: Mediapartners-Google
Disallow:

User-agent: Adsbot-Google
Disallow:

User-agent: Googlebot-Image
Disallow:

# cart
User-agent: *
Disallow: /viewcart

# Something related to carousel and recommendation carousel
User-agent: *
Disallow: /dynamic/

# Permanent Link For Individual Review
User-agent: *
Disallow: /reviews/

# Old Browse Page Experience
User-agent: *
Disallow: /store/
```

웹스크래핑

- **웹스크래핑** Web Scraping
  - 표(Table) 수집
  - 텍스트 수집
  - 영상 수집

- **웹 페이지 소스보기**
  - Chrome 브라우저로 웹 페이지를 열고, Ctrl+U

표 수집

- **HTML 웹 페이지에서 표 속성 가져오기**
  - Pandas read_html()함수는 HTML 웹페이지에 있는 <table>태그에서 표 형식의 데이터를 모두 찾아서 데이터프레임으로 변환
  - 표 데이터들은 각각 별도의 데이터프레임으로 변환되기 때문에 여러 개의 데이터프레임(표)을 원소로 가지는 리스트가 반환

- **lxml 라이브러리 설치**
  - lxml 라이브러리는 HTML, XML 문서를 문법적으로 분석하는 기능을 수행하기 때문에 이를 parser라고도 함
  - HTML을 해석하기 위해 필요

    cf. HTML버전에 따라 추가 라이브러리 설치 필요

lxml library 설치

- **Anaconda Prompt에서**
  - 가상환경 목록 확인

    : conda env list
  - 가상환경 활성화

    : conda activate data_eng
  - 설치된 library목록 확인

    : conda list

    (lxml library없는 것 확인)
  - lxml library 설치

    : conda install lxml

실습1.1 HTML 표 수집

- 표 수집
  - read_html()함수를 이용하여 웹 페이지의 표 정보를 파싱하면, HTML 웹 페이지의 주소(URL)를 따옴표 안에 입력
  - 예를 들어, pd.read_html('https://www.naver.com/')과 같이
  - 'sample.html' 웹 페이지에서 2개의 표 수집

```
import pandas as pd
```

```
url = './sample.html'
tables = pd.read_html(url)
```

```
print(len(tables))
```

```
2
```

실습1.2 HTML 표 출력

- 수집한 표 출력
  - 2개의 표를 출력

```
for i in range(len(tables)):
    print("tables[%d]"%i)
    print(tables[i])
    print('₩n')
```

```
tables[0]
   Unnamed: 0  c0  c1  c2  c3
0           0   0   1   4   7
1           1   1   2   5   8
2           2   2   3   6   9


tables[1]
         name  year         developer  opensource
0       NumPy  2006  Travis Oliphant        True
1  matplotlib  2003   John D. Hunter        True
2      pandas  2008    Wes Mckinneye        True
```

실습1.3 HTML 표 수집 사례

- 웹사이트실습
  - https://pandas.pydata.org/docs/user_guide/io.html#csv-text-files

```
import pandas as pd
```

```
url = 'https://pandas.pydata.org/docs/user_guide/io.html#csv-text-files'
tables = pd.read_html(url)
```

```
print(len(tables))
```

```
15
```

```
for i in range(len(tables)):
    print("tables[%s]" %i)
    print(tables[i])
    print('₩n')
```

```
tables[0]
   Format Type       Data Description          Reader         Writer
0         text                    CSV        read_csv        to_csv
1         text  Fixed-Width Text File        read_fwf            NaN
2         text                   JSON       read_json       to_json
3         text                   HTML       read_html       to_html
4         text        Local clipboard  read_clipboard  to_clipboard
5          NaN              MS Excel      read_excel      to_excel
6       binary           OpenDocument      read_excel            NaN
7       binary             HDF5 Format        read_hdf        to_hdf
8       binary          Feather Format    read_feather    to_feather
9       binary          Parquet Format    read_parquet    to_parquet
10      binary              ORC Format        read_orc            NaN
11      binary                 Msgpack    read_msgpack    to_msgpack
12      binary                   Stata      read_stata      to_stata
13      binary                     SAS        read_sas            NaN
14      binary                    SPSS       read_spss            NaN
15      binary   Python Pickle Format     read_pickle     to_pickle
16         SQL                     SQL        read_sql        to_sql
17         SQL        Google BigQuery        read_gbq        to_gbq
```

텍스트 수집

- **텍스트 수집 절차**

1. 수집할 텍스트가 있는 웹사이트 찾기
2. 웹 페이지 소스 검토
   : 수집할 텍스트 태그 등 확인
4. Library 등 수집 도구 준비
   : BeautifulSoup, requests 등
5. 코딩
   : Crawl, Parsing 등
6. 저장
   : csv / DB

텍스트 수집 도구

- **requests library**
  - 웹 페이지의 URL로 요청해서 소스코드를 얻어옴
  - 예를 들어, url = "http://www.yes24.com/24/Category/BestSeller" 에서
  - response = requests.request("GET", url) 또는

    response = requests.get(url) 로 요청

- **BeautifulSoup Library**
  - requests 라이브러리로 수집한 response객체를 파싱하기 위해 구문분석 트리를 만들어주는 라이브러리가 Beautiful Soup
  - find() 메소드 등으로 태그 및 속성을 통해 필요한 텍스트를 수집

beautifulsoup library 설치

- **Anaconda Prompt에서**
  - 가상환경 목록 확인

    : conda env list
  - 가상환경 활성화

    : conda activate data_eng
  - 설치된 library목록 확인

    : conda list

    (beautifulsoup4, requests library없는 것 확인)
  - beautifulsoup4, requests library 설치

    : conda install beautifulsoup4

    conda install requests

BeautifulSoup 메소드

- **find()**
  - find() : 가장 먼저 등장하는 해당 태그 값을 가져옴
  - find('a') : <a> 태그 객체 반환
  - find('div', attrs={'class' : 'hotel'} : <div'>태그 중 class속성이 'hotel'인 태그 객체 반환

- **find_all()**
  - 해당 태그를 가진 모든 값을 리스트(list) 형식으로 가져옴
  - find_all('a') : 모든 <a>태그 리스트 반환
  - 태그 옵션 사용법은 find()와 같음

BeautifulSoup로 값 추출

- 예제
  - 예를 들어, 달러구트 꿈 백화점 에서
  - tag = find('a') : <a> tag의 정보를 얻어와서
  - print(tag.text) 또는

    print(tag.get_text()) 태그내의 텍스트 수집

- **select()**
  - .select('상위태그 > 차상위 종속태그 > 하위 종속태그')
  - 상,하위 태그 종속관계 옵션 가능

실험 데이터세트

- **YES24 도서구매 사이트**
  - YES24 베스트셀러 부문
  - url : http://www.yes24.com/24/Category/BestSeller

- **실험 내용**
  - 베스트셀러 웹 페이지 분석
  - 베스트셀러 도서 정보를 수집
  - 베스트셀러의 도서 제목, 저자, 가격을 표로 작성

실습2.1 Crawl

- **Crawl**
  - requests, BeautifulSoup 라이브러리 import하기
  - requests 라이브러리로 url의 소스코드 수집

```
import requests
from bs4 import BeautifulSoup
import pandas as pd
```

```
# target URL to scrap
url = "http://www.yes24.com/24/Category/BestSeller"
# send request to download the data
response = requests.request("GET", url)
```

실습2.2 Parsing시작

- **HTML 파싱 시작**
  - BeautifulSoup로 HTML 구문분석 트리 생성

```
# parse the downloaded data
data = BeautifulSoup(response.text, 'html.parser')
print(data)
```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>
<head><base href="http://www.yes24.com/24/"/>
<meta content="IE=Edge" http-equiv="X-UA-Compatible"/>
<meta content="text/html;charset=utf-8" http-equiv="Content-Type"/>
<meta content="dpr, width, viewport-width, rtt, downlink, ect, UA, UA-Platform, UA-Arch, UA-Model, UA-Mobile, UA-Full-Version" ht
tp-equiv="Accept-CH"/>
<meta content="86400" http-equiv="Accept-CH-Lifetime"/>
<meta content="width=1170" name="viewport"/>
<title>YES24 | 대한민국 대표 인터넷서점 | 베스트셀러</title>
<meta content="YES24 - 대한민국 대표 인터넷서점" name="title"/>
<meta content="YES24는 대한민국 1위 인터넷 온라인 서점 입니다. 국내 최대의 도서정보를 보유하고 있으며, 음반, DVD, 공연, 영화까지
다양한 문화 콘텐츠 및 서비스를 제공합니다." name="description"/>
<meta content="인터넷 서점, 온라인 쇼핑, 상품 추천, 쇼핑몰, 상품 검색, 도서 정보, 국내도서, 외국도서, 전자책, eBook, 이북, 크레
마, 공연, 콘서트, 뮤지컬, 영화, 음반, 예매, DVD, 블루레이, 예스24, YES24, 교보문고, 알라딘" name="keywords"/>
<meta content="https://secimage.yes24.com/sysimage/renew/logo_meta.png" property="og:image"/>
<script src="https://secimage.yes24.com/sysimage/Contents/Scripts/p/jquery/jquery-1.2.6.min.js" type="text/javascript"></script>
<script src="https://secimage.yes24.com/sysimage/Contents/Scripts/p/jquery/jquery.menu-aim.js?v=20140801" type="text/javascript">
```

웹 페이지 소스 검토

- **웹 페이지 소스 검토**
  - 수집할 텍스트 태그 등 확인

- **실습**
  - 웹 페이지 소스 둘러보기
    : 페이지 소스 보기 (Ctrl + U)
  - 베스트셀러 도서명 '어떻게 말해줘야 할까'의 위치 확인
    : 검색 (Ctrl + F) 창에 '어떻게' 입력
  - 도서명, 저자, 가격을 수집하기 위한 태그 확인

실습2.3 select() 활용

- **select() 로 파싱**
  - select()로 ol > li 종속태그 모두 수집 : 40개 베스트셀러 도서

```
cards_data = data.select('ol > li')
# total number of cards
print('Total Number of Cards Found : ', len(cards_data))
```

```
Total Number of Cards Found :  40
```

```
ol p.image a .icon_19 { position:absolute;right:0;bottom:0; }
</style>
<ol class="">

    <li class="num1">

        <p class="copy"><a href="/Product/Goods/93522583">국민 육아멘토 오은영 박사의 현실밀착 육아회화</a></p>
        <p class="image" id="location_0">
            <a href="/Product/Goods/93522583">
                <img src="http://image.yes24.com/goods/93522583/S" alt="어떻게 말해줘야 할까"/>

            </a>
        </p>
        <p>[도서] <a href="/Product/Goods/93522583">어떻게 말해줘야 할까</a></p>

        <p class="aupu"><a href="http://www.yes24.com//SearchCorner/Result?domain=ALL&author_yn=Y&query=&auth_nc
```

실습2.4 select() 결과 출력

- **select()로 수집한 리스트 출력**
  - 40개 베스트셀러 도서 <li> 태그 리스트 출력

```
for card in cards_data:
    print(card)
```

```
<li class="num1">
<p class="copy"><a href="/Product/Goods/93522583">국민 육아멘토 오은영 박사의 현실밀착 육아회화</a></p>
<p class="image" id="location_0">
<a href="/Product/Goods/93522583">
<img alt="어떻게 말해줘야 할까" src="http://image.yes24.com/goods/93522583/S">
</img></a>
</p>
<p>[도서] <a href="/Product/Goods/93522583">어떻게 말해줘야 할까</a></p>
<p class="aupu"><a href="http://www.yes24.com//SearchCorner/Result?domain=ALL&amp;author_yn=Y&amp;query=
et="_blank">오은영</a> 저/<a href="http://www.yes24.com//SearchCorner/Result?domain=ALL&amp;author_yn=Y&
0309" target="_blank">차상미</a> 그림 | <a href="http://www.yes24.com//SearchCorner/Result?domain=ALL&am
김영사">김영사</a></p>
<p class="price"><strong>15,750원</strong>(10%<img src="http://image.yes24.com/sysimage/wel/i_dc.gif"/>+
yes24.com/sysimage/wel/i_p.gif"/>)</p>
<p>회원리뷰 (<a href="/Product/Goods/93522583#Review">85개</a>)</p>
```

태그로 정보 확인

- 책 제목, 저자, 가격 태그로 정보 확인
  - 책 제목은 3번째 <a>태그, 저자는 4번째, 가격은 첫번째 태그에

```
<ol class="">
    <li class="num1">
        <p class="copy"><a href="/Product/Goods/93522583">국민 육아멘토 오은영 박
        <p class="image" id="location_0">
            <a href="/Product/Goods/93522583">
                <img src="http://image.yes24.com/goods/93522583/S" alt="어떻게 말
            </a>
        </p>
        <p>[도서] <a href="/Product/Goods/93522583">어떻게 말해줘야 할까</a></p>
        <p class="aupu"><a href="http://www.yes24.com//SearchCorner/Result?domain
        <p class="price"><strong>15,750원</strong>(10%<img src="http://image.yes2
```

실습2.5 태그로 정보 수집

- 베스트셀러 정보 출력
  - 책 제목

    all_a = card.find_all('a')

    : 태그 내의 모든 <a>태그의 정보를 리스트로 반환

    all_a[2].text

    : 책 제목은 3번째 <a>태그에 텍스트로 있음
  - 저자
  - 책 가격

    : 첫번째 태그에 있음

```
# extract the book info
for card in cards_data:
    # get the book name
    all_a=card.find_all('a')
    print(all_a[2].text)
    # get author
    print(all_a[3].text)
    # get price
    price = card.find('strong')
    print(price.text)
```

```
어떻게 말해줘야 할까
오은영
15,750원
일인칭 단수
무라카미 하루키
13,050원
트렌드 코리아 2021
김난도
16,200원
달러구트 꿈 백화점
이미예
12,420원
```

실습2.6 베스트셀러 표 만들기

- 베스트셀러 정보 딕셔너리로 리스트 만들기
  - 책제목, 저장, 가격 Label과 값으로 card_details 딕셔너리를 생성하고, scraped_data 리스트에 저장

```
scraped_data = []
for card in cards_data:
    # initialize the dictionary
    card_details = {}
    all_a=card.find_all('a')
    book_name = all_a[2]
    author=all_a[3]
    price = card.find('strong')

    # add data to the dictionary
    card_details['book_name'] = book_name.text
    card_details['author'] = author.text
    card_details['price'] = price.text

    # append the scraped data to the list
    scraped_data.append(card_details)
```

실습2.7 베스트셀러 표 출력

- 데이터프레임으로 변환

```
# create a data frame from the list of dictionaries
df = pd.DataFrame.from_dict(scraped_data)
df
```

	book_name	author	price
0	어떻게 말해줘야 할까	오은영	15,750원
1	일인칭 단수	무라카미 하루키	13,050원
2	트렌드 코리아 2021	김난도	16,200원
3	달러구트 꿈 백화점	이미예	12,420원
4	공정하다는 착각	마이클 샌델	16,200원
5	마음챙김의 시	류시화	11,700원
6	미스터 마켓 2021	이한영	15,300원
7	나의 하루는 4시 30분에 시작된다	김유진	13,500원
8	Go Go 카카오프렌즈 17 러시아	김미영	10,800원
9	돈의 속성	김승호	15,120원
10	엑시트 EXIT	송희창	15,300원

실습2.8 csv파일로 저장

- **csv파일로 저장**
  - utf-16으로 encoding하여 한글 깨짐 해결

```
# save the scraped data as CSV file
df.to_csv('book_data.csv',encoding='utf-16', index=False)
```

영상 수집

- **HTML 웹 페이지에서 영상 수집**
  - 사례: YES24 베스트셀러 책 표지 사진 수집
  - HTML 웹 페이지에서 <img> 태그의 src속성 가져오기

```
<ol class="">
    <li class="num1">
        <p class="copy"><a href="/Product/Goods/93522583">국민 육아멘토 오은영 박사의 현실밀착 육
        <p class="image" id="location_0">
            <a href="/Product/Goods/93522583">
                <img src="http://image.yes24.com/goods/93522583/S" alt="어떻게 말해줘야 할까"/>
            </a>
        </p>
        <p>[도서] <a href="/Product/Goods/93522583">어떻게 말해줘야 할까</a></p>
        <p class="aupu"><a href="http://www.yes24.com//SearchCorner/Result?domain=ALL&author_yn=
        <p class="price"><strong>15,750원</strong>(10%<img src="http://image.yes24.com/sysimage/
```

실습3.1 <img>태그로 수집

- 실습2.3에 이어서
  - 첫번째 <img> 태그로 수집

```
# find all with the image tag
images=[]
for card in cards_data:
    images.append( card.find('img', src=True) )
print('Number of Images: ', len(images))
```

```
Number of Images:  40
```

```
for image in images:
    print(image)
```

```
<img alt="어떻게 말해줘야 할까" src="http://image.yes24.com/goods/93522583/S">
</img>
<img alt="일인칭 단수" src="http://image.yes24.com/goods/95538356/S">
</img>
<img alt="트렌드 코리아 2021" src="http://image.yes24.com/goods/93068681/S">
</img>
<img alt="달러구트 꿈 백화점" src="http://image.yes24.com/goods/91065309/S">
</img>
```

실습3.2 src 속성 수집

- 베스트셀러 책표지 src 속성으로 수집
  - 첫번째 <img> 태그의 src 속성 수집

```
# select src tag
image_src = [x['src'] for x in images]
for image in image_src:
    print(image)
```

```
http://image.yes24.com/goods/93522583/S
http://image.yes24.com/goods/95538356/S
http://image.yes24.com/goods/93068681/S
http://image.yes24.com/goods/91065309/S
http://image.yes24.com/goods/94489333/S
http://image.yes24.com/goods/92462696/S
http://image.yes24.com/goods/95563770/S
http://image.yes24.com/goods/93513663/S
http://image.yes24.com/goods/95716613/S
http://image.yes24.com/goods/90428162/S
```

실습3.3 영상 저장

- 베스트셀러 책표지 수집하여 저장
  - 영상 url을 요청하여 write()로 영상 저장

```
image_count = 1
for image in image_src:
    with open('image_'+str(image_count)+'.jpeg', 'wb') as f:
        res = requests.get(image)
        f.write(res.content)
    image_count = image_count+1
```

  - 수집하여 저장된 영상


image_1.jpeg


image_2.jpeg

image_3.jpeg


image_4.jpeg

웹 페이지 동적 추적

- **Selenium Library**
  - 사용자의 행동을 동적으로 추적하여 데이터 수집
  - web driver를 사용하기 위한 라이브러리

- **Web Driver**
  - FireFox, Chrome 과 같은 브라우저에서 제공하는 API로, 이를 사용하여 코드상에서 브라우저를 다룰 수 있음
  - Cromedriver : 크롬을 컨트롤 할 수 있게 만드는 프로그램. Selenium과 같이 사용한다. 현재 사용중인 크롬 버전과 동일한 버전의 chromedriver를 설치해야 함

selenium library 설치

- **Anaconda Prompt에서**
  - 가상환경 목록 확인

    : conda env list
  - 가상환경 활성화

    : conda activate data_eng
  - 설치된 library목록 확인

    : conda list

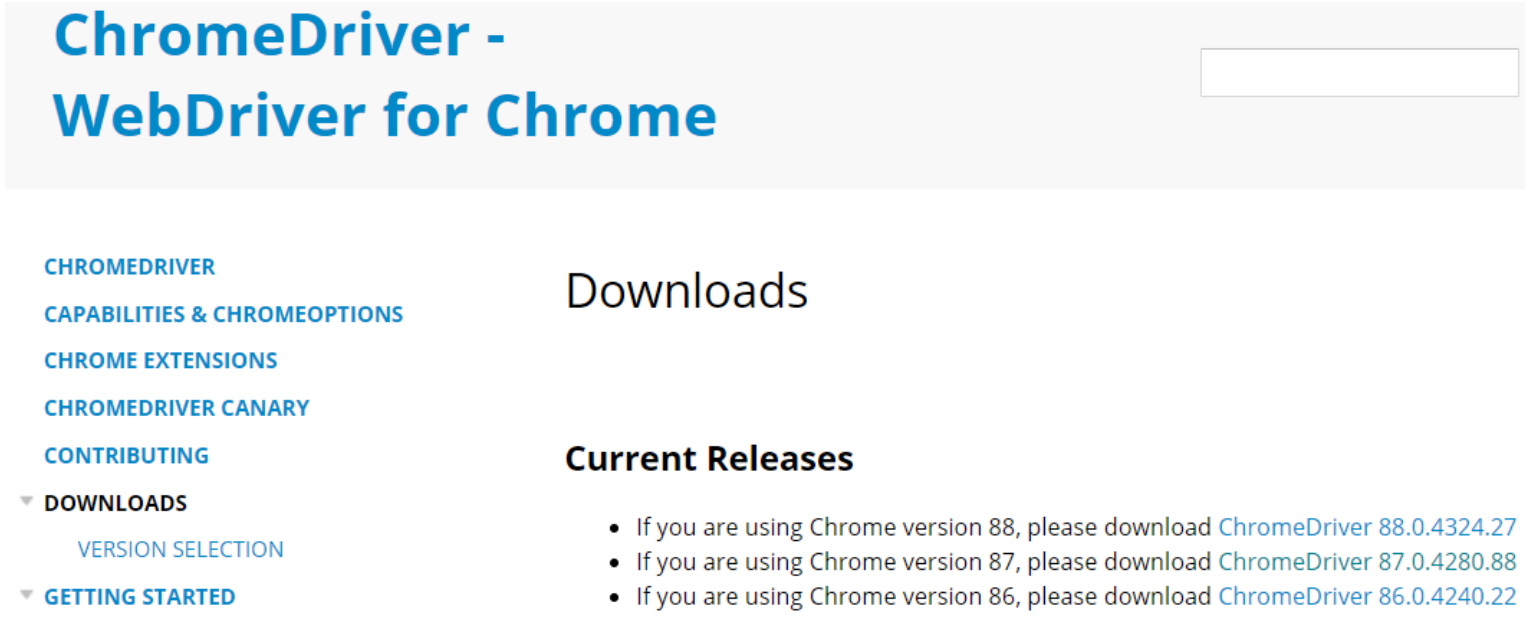
    (selenium library없는 것 확인)
  - selenium library 설치

    : conda install selenium

chromebrowser 설치

- 다운로드 및 설치
  - 다운로드 URL

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



실습4 브라우저 실행

- 코드로 브라우저 실행
  - 이쇼핑 사이트 열기

```
from selenium import webdriver
```

```
browser = webdriver.Chrome('./chromedriver.exe')
```

```
url = 'http://www.eshopping.co.kr'
browser.get(url)
```

실습5.1 코드로 검색

- 코드로 구글 검색
  - input()으로 검색어 입력

```
from urllib.parse import quote_plus

from selenium import webdriver

baseUrl = 'https://www.google.com/search?q=' # 기본 주소

plusUrl = input('무엇을 검색할까요? : ') # 검색명
```

무엇을 검색할까요? :

실습5.2 검색 실행

- 코드로 구글 검색 실행
  - ‘selenium’ 검색 요청
  - cf. urllib.parse : quote_plus()

```
plusUrl = input('무엇을 검색할까요? : ') # 검색명
```

```
무엇을 검색할까요? : selenium
```

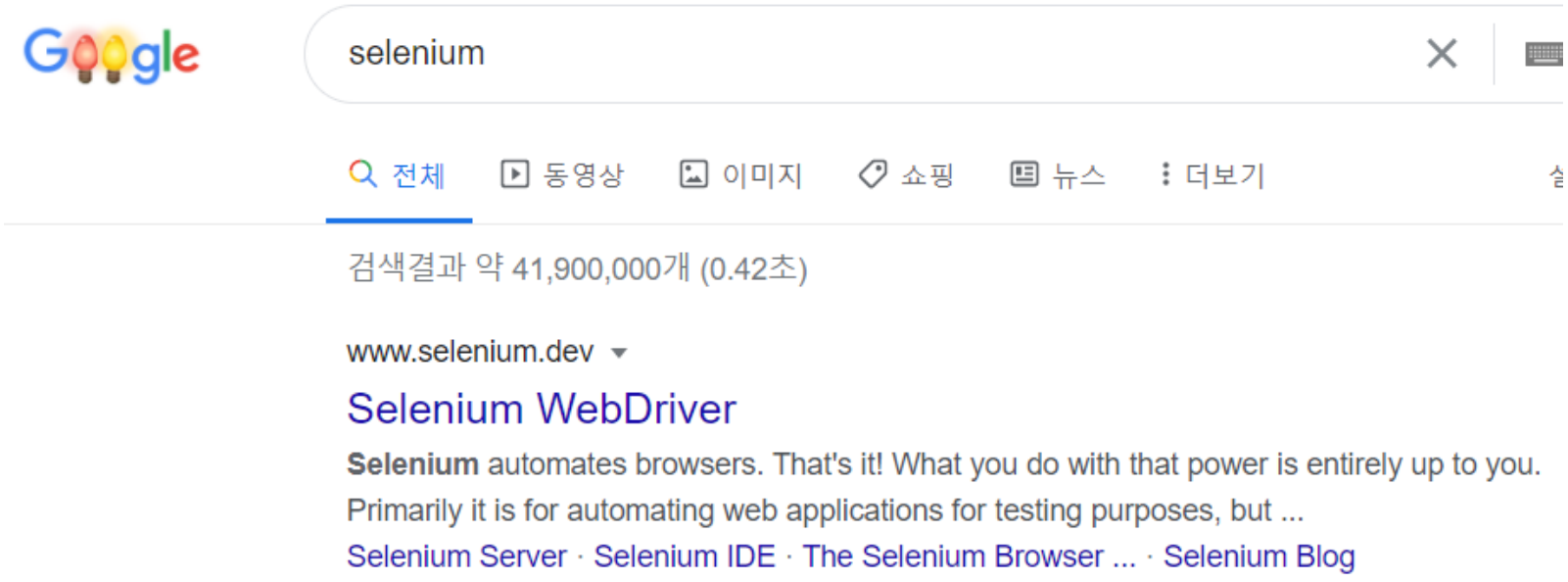
```
url = baseUrl + quote_plus(plusUrl) # 크롬에 검색할 주소
```

```
browser = webdriver.Chrome('./chromedriver.exe')
```

```
browser.get(url) # 검색명이 들어간 주소를 크롬주소창에 입력
```

검색 결과

- 코드로 구글 검색 결과
  - Chrome 브라우저에 검색 결과 출력



참고도서

➢ reference

[1] 파이썬을 활용한 데이터길들이기

- 프로그래밍인사이트

[2] 모두의 데이터분석

- 길벗

[3] 파이썬머신러닝 판다스데이터분석

- 정보문화사

End