

Crawling 강의자료

📅 날짜	@2021년 7월 8일
✍️ 작성자	한소율

크롤링이란?

인터넷에서 존재하는 데이터를 컴퓨터 프로그램을 통하여 자동화된 방법으로 웹에서 데이터를 수집하는 모든 작업을 말한다.

정적크롤링

한 페이지 안에 원하는 정보가 모두 드러난 경우 사용하는 것이 정적 크롤링이다.

정적인 웹 페이지는 어떠한 상황에서도 항상 일관된 페이지를 보여주는 경우를 말한다.

- 장점: 페이지의 변화가 필요 없기 때문에 속도가 빠르다.
- 단점: 페이지의 변화가 필요한 경우 수집 대상에 한계가 있다.
- 예시: 네이버 실시간 검색 순위, 음악 차트 순위 등

동적 크롤링

입력, 마우스 스크롤, 클릭 방식의 크롤링 의미한다.

동적인 웹 페이지는 사용자에게 따라 다른 데이터를 보여주는 경우를 말한다.

- 장점: 수집 대상에 한계가 거의 없다.
- 단점: 단계적인 접근이 필요하여 수집 속도가 느리다.
- 예시: 유튜브, 인스타그램, 네이버 주식 등

Selenium이란?

python으로 크롤링을 할 때, 크롤링 대상인 웹 페이지에 동적인 동작을 통해서 이벤트를 발생시키고, 그에 대한 응답 결과를 크롤링을 할 수 있게 도와주는 라이브러리이다.

Selenium은 web driver를 제어할 수 있다. 따라서 web driver라는 가상의 브라우저 프로그램(웹 테스트 도구)과 연동해서 위 기능을 구현할 수 있다.

Step-by-step installing selenium

▼ 1. selenium 설치

- Windows

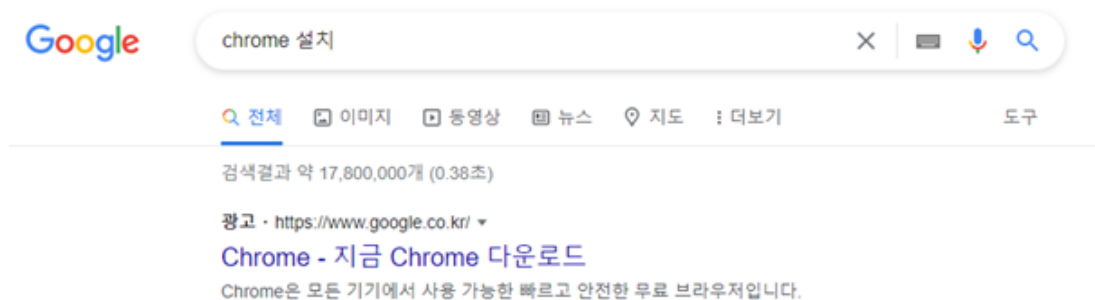
```
pip install selenium
```

- Mac

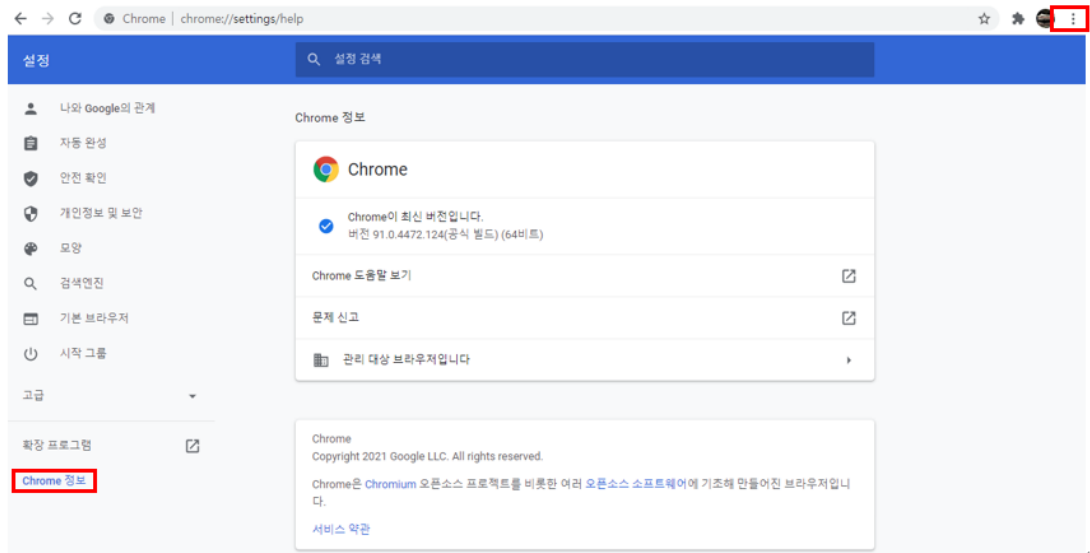
```
pip3 install selenium
```

▼ 2. Chrome browser

- 설치



- 설치되어 있을경우, 버전확인(Update 필요)

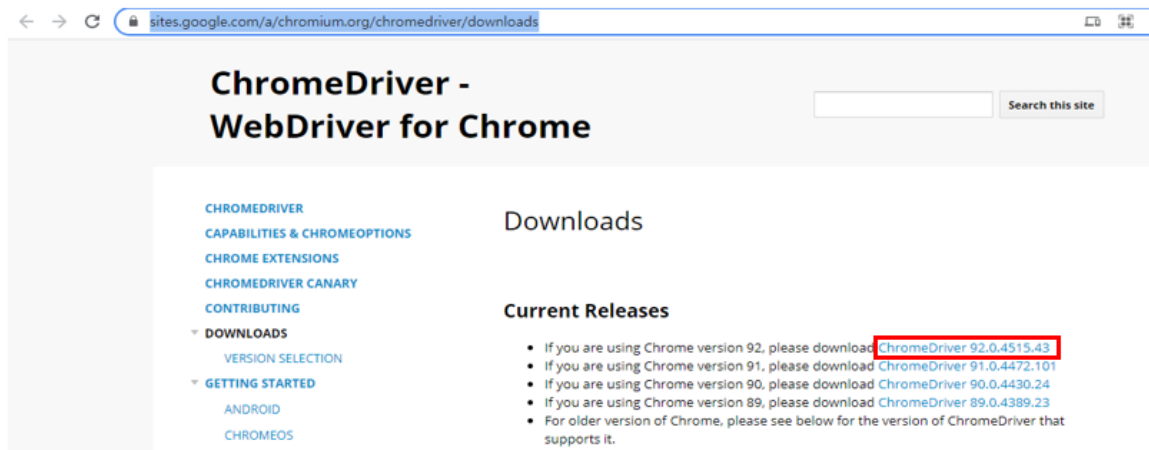


▼ 3. Chrome web driver 설치

- Chrome browser 버전에 맞는 driver 설치







<https://sites.google.com/a/chromium.org/chromedriver/downloads>

- 최신버전 설치



- 리눅스 vs MAC vs 윈도우
- 본인의 OS에 맞게 선택

Index of /92.0.4515.43/

	Name	Last modified	Size	ETag
	Parent Directory		-	
	chromedriver_linux64.zip	2021-06-11 09:57:52	5.74MB	b29501f84c0a3104958df2d950417450
	chromedriver_mac64.zip	2021-06-11 09:57:55	7.76MB	35c25e2c5a45310850c35e8288388dc6
	chromedriver_mac64_m1.zip	2021-06-11 09:57:58	7.10MB	0e20a81c41fb64b3d71791f764993ab7
	chromedriver_win32.zip	2021-06-11 09:58:00	5.66MB	400a977e0ed14e4340155ed330f72f5b
	notes.txt	2021-06-11 09:58:06	0.00MB	6d98c0a6a6f45a38376d5fa70213a08c

▼ 4. 드라이버 위치 변경

- 다운로드 받은 크롬 드라이버는 자신이 사용할 python이 있는 위치로 이동
- 다시말해서 자신이 코딩할 파이썬 파일이 있는 폴더 .py , .ipynb 가 있는 위치
- 경로확인

```
import os
os.getcwd()
```

실습 Code(Jupyter Notebook 사용)

- 크롬 브라우저 열고 닫기

```
from selenium import webdriver

#크롬브라우저 열기
driver = webdriver.Chrome('chromedriver.exe')
driver.get('https://www.naver.com/')

# 브라우저 닫기
driver.close()
```

- 검색

```

from selenium import webdriver

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

from bs4 import BeautifulSoup
import time
import re

# 크롬 브라우저 열기
driver =webdriver.Chrome('chromedriver.exe')
driver.get('http://www.instagram.com')

# -----
# 검색어 조건에 따른 url 생성
def insta_searching(word):
    url = "https://www.instagram.com/explore/tags/" + str(word)
    return url

word = '운동'
url = insta_searching(word)
driver.get(url)

```

• 인스타그램 자동 로그인

1. F12 클릭

- 자동꺼짐 발생 참고: <https://bustar.tistory.com/218>

2. 오른쪽 마우스 커서 아이콘 클릭

```

# 크롬 브라우저 열기
driver =webdriver.Chrome('chromedriver.exe')
driver.get('http://www.instagram.com')
p_tag = WebDriverWait(driver,timeout=5).until(EC.presence_of_element_located(
    (By.TAG_NAME, "p")))

time.sleep(3)

# 인스타그램 로그인을 위한 계정정보
email = 'user_id'
input_id = driver.find_elements_by_css_selector('input._2hvTZ.pexuQ.zyHYP')[0]
input_id.clear()
input_id.send_keys(email)

password = 'user_pw'
input_pw=driver.find_elements_by_css_selector('input._2hvTZ.pexuQ.zyHYP')[1]
input_pw.clear()

```

```

input_pw.send_keys(password)
input_pw.submit()

time.sleep(5)

# 정보 나중에 저장하기 클릭하고 넘어가기
driver.find_element_by_css_selector('button.sqd0P.ywX7d.y3zKF').click()
time.sleep(3)
# 설정 나중에하기 클릭하고 넘어가기
driver.find_element_by_css_selector('button.a00lW.HoLwm').click()
time.sleep(3)

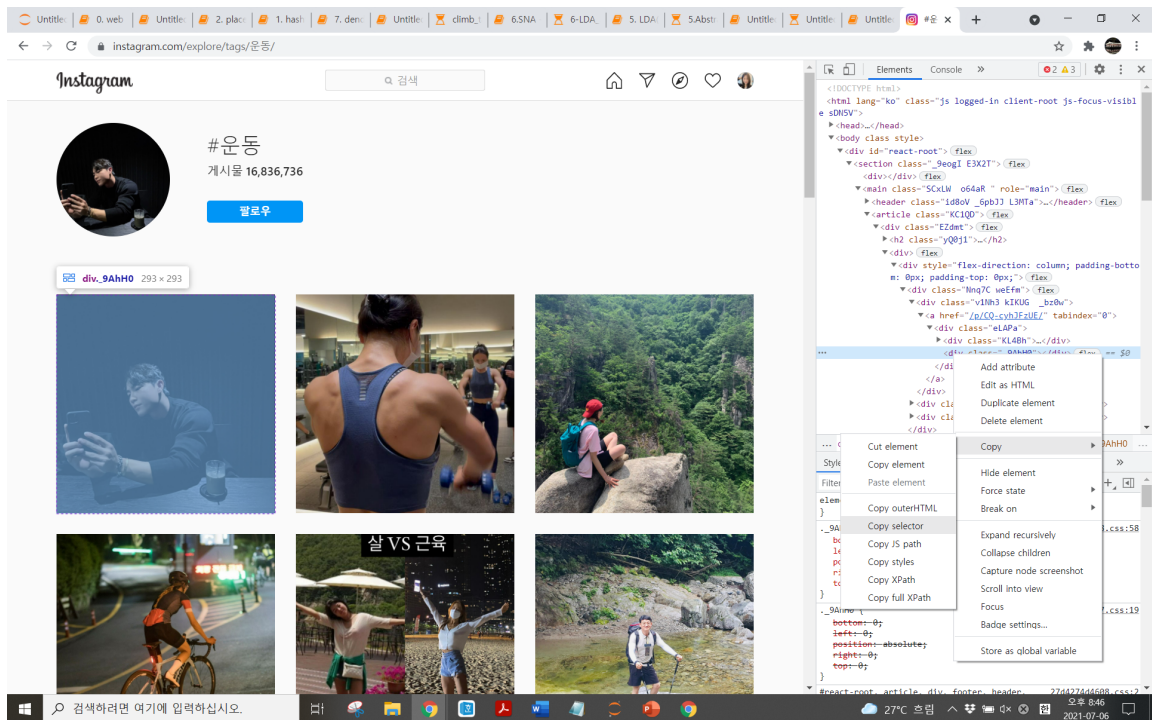
#-----
# 게시물 검색
def insta_searching(word):
    url = "https://www.instagram.com/explore/tags/" + str(word)
    return url

word = '운동'
url = insta_searching(word)
driver.get(url)

```

• 첫번째 게시물 열기

1. F12 클릭
- 자동꺼짐 발생 참고: <https://bustar.tistory.com/218>
2. 오른쪽 마우스 커서 아이콘 클릭
3. 첫 번째 인기 게시물 선택
4. 선택된 Elements 위치에서 오른쪽 마우스 클릭 > Copy > Copy Selector 클릭



```
# 첫번째 게시물 열기
def click_first(driver):
    first = driver.find_element_by_css_selector('#react-root > section > main > article > div:nth-child(3) > div > div:nth-child(1) > div:nth-child(1) > a > div.eLAPa > div._9AhH0')
    first.click()
    time.sleep(7)

click_first(driver)
```

```
# 첫번째 게시물 열기
def click_first(driver):
    first = driver.find_element_by_css_selector('div._9AhH0')
    first.click()
    time.sleep(7)

click_first(driver)
```

- 본문내용, 작성일자, 해시태그, 위치정보 추출

- 정규표현식

- findall()

- :정규식과 매치되는 모든 문자열(substring)을 리스트로 돌려준다.

- 문자 사이의 범위(하이픈, -)

[a-c] = [abc]

[0-5] = [012345]

```
a = ['a', '#1234', '#운동', 'sns', '135', '123']

import re
b = re.findall('#[A-Za-z0-9가-힣]+' , str(a))

c= ''.join(b).replace("#", " ")

c.split()
```

• 데이터 수집

```
import re
instagram_tags=[]

def get_content(driver):
    # 1. 현재 페이지의 HTML 정보 가져오기
    html = driver.page_source
    soup = BeautifulSoup(html, 'lxml')

    # 2. 본문내용
    try:
        content = soup.select('div.C4VMK > span')[0].text
    except:
        content = ''

    # 3. 본문 내용에서 해시태그 가져오기(정규표현식 활용)
    tags = re.findall('#[A-Za-z0-9가-힣]+' , content)
    tag = ''.join(tags).replace("#", " ") # "#" 제거
    tag_data = tag.split()
    for tag_one in tag_data:
        instagram_tags.append(tag_one)

    # content 변수의 본문 내용 중 #으로 시작하며, #뒤에 연속된 문자(공백이나 #, \ 기호가 아닌 경우)를 모두 찾아 tags 변수에 저장

    # 4. 작성 일자 가져오기
    try:
        date = soup.select('time._1o9PC.Nzb55')[0]['datetime'][:10] #앞에서부터 10자리 글자
    except:
        date = ''

    # 5. 위치 정보 가져오기
    try:
```



```

        place = soup.select('div.JF9hh')[0].text
    except:
        place = ''

```

```

# 6. 수집한 정보 저장하기
data = [content, date, place, tag_data]
return data

```

```

get_content(driver)

```

- 엑셀 파일로 데이터 저장

```

def next_page(driver):
    next_page = driver.find_element_by_css_selector('body > div._2dDPU.CkGkG > di
v.EfHg9 > div > div > a._65Bje.coreSpriteRightPaginationArrow')
    next_page.click()
    time.sleep(7)

```

```

next_page(driver)

```

- 데이터 수집

```

word = "클라이밍"
url = insta_searching(word)

# 검색페이지 접속
driver.implicitly_wait(9)

driver.get(url)
time.sleep(9)

# 첫 번째 게시물 열기
click_first(driver)

#크롤링 결과리스트를 리스트 생성
result1 = [ ]

# 여러 게시물 수집하기
target = 5      # 크롤링할 게시물 수
for i in range(target):

```

```
# 게시글 수집에 오류 발생시 7초 대기후, 다음 게시글로 넘어가도록 예외처리 구문 활용
try:
    data = get_content(driver)    # 게시글 정보 가져오기
    result1.append(data)
    next_page(driver)
except:
    time.sleep(7)
    next_page(driver)
```

- 다음페이지로 이동

```
import pandas as pd

# 크롤링 데이터 중복 처리
df = pd.DataFrame([])
df = df.append(result1)

df.columns = ['본문', '시간', '위치', '해시태그']

df.drop_duplicates(subset = ['본문'] , inplace = True)
df=df.reset_index(drop=True)

# 파일 저장하기
df.to_csv("데이터수집.csv", header = True, index = False, encoding='utf-8-sig')
```