

# Python 인스타그램 크롤링

키워드: 웹 크롤러 ([https://ko.wikipedia.org/wiki/%EC%9B%B9\\_%ED%81%AC%EB%A1%A4%EB%9F%AC](https://ko.wikipedia.org/wiki/%EC%9B%B9_%ED%81%AC%EB%A1%A4%EB%9F%AC)), Selenium (<https://www.seleniumhq.org/>)

소셜미디어 채널의 크롤링을 진행하기 위해서 우리는 웹 사이트를 분석하고, 크롤링 코드를 작성하거나 기업에서 제공하는 "Open API"라는 것을 이용하여 데이터를 제공받을 수 있다고 알아보았습니다. 크롤링을 위해서 기본이 되는 파이썬 패키지는 크롤링 기초 참고 자료에서 살펴본 "requests"와 "BeautifulSoup" 두 개의 패키지가 사용될 수 있습니다. 다만, 소셜미디어 혹은, 포털 사이트의 게시판과 같은 사이트는 우리에게 정보를 보여주는 구조상 일반적으로 위 두 가지 패키지로 크롤링을 원활하게 진행하기 어려움이 있습니다. 이러한 경우 우리는 "Selenium"을 이용해서 동적으로 브라우저를 직접 조작하는 형태로 코드를 작성해야 합니다. 첫 번째 테스트 코드로 작성한 크롤링 코드인 인스타그램 테스트 코드를 살펴보도록 하겠습니다.

## 셀레니움 설치& 크롬 드라이버 설치

파이썬 코드를 작성하기 전에, 선행되어야 할 몇 가지 작업이 있는데요, 아래의 준비 과정을 진행해주세요!

아래의 명령어를 VS Code의 터미널에 입력하여 selenium 패키지를 다운로드합니다.

```
pip3 install selenium
```

다음은, [여기 \(http://chromedriver.chromium.org/downloads\)](http://chromedriver.chromium.org/downloads)에 접속하여 크롬 드라이버를 사용하는 운영체제에 맞게 프로젝트 폴더에 다운로드합니다.

## ChromeDriver - WebDriver for Chrome

CHROMEDRIVER  
CAPABILITIES & CHROME OPTIONS  
CHROME EXTENSIONS  
CHROMEDRIVER CANARY  
CONTRIBUTING  
DOWNLOADS  
GETTING STARTED  
ANDROID  
CHROME OS  
LOGGING  
PERFORMANCE LOG  
MOBILE EMULATION  
NEED HELP?  
CHROME DOESN'T START OR CRASHES IMMEDIATELY  
CHROMEDRIVER CRASHES  
CLICKING ISSUES  
DEVTOOLS WINDOW KEEPS CLOSING  
OPERATION NOT SUPPORTED WHEN USING REMOTE DEBUGGING

### Downloads

**Latest Release: ChromeDriver 2.43**

Supports Chrome v69-71

**Changes include:**






- Fixed Parsing of proxy configuration is not standard compliant
- Fixed Launch app command is flaky
- Fixed Screenshot of element inside iFrame is taken incorrectly
- Added ChromeDriver supports window resizing over a remote connection
- Fixed Error codes are not handled in Clear element
- Fixed Not waiting until element is visible
- Fixed Get element property is not implemented
- Fixed Switch to Frame is not spec compliant
- Fixed Execute Async Script does not return spec compliant error codes
- Fixed Execute Script does not return spec compliant error codes
- Fixed Error code in ExecuteGet is not conformant with spec
- Fixed Send Alert Text is not returning spec compliant error codes
- Fixed clear() on an input type="date" pretends element is not user-editable
- Fixed Chromedriver gets window handle for the tab which is opened manually
- Fixed Allow append or start a new log file for chromedriver
- Fixed New Session does not invoke w3c mode if flag is in firstMatch

**ChromeDriver 2.42**

Supports Chrome v68-70

zip 파일의 압축을 풀고, chromedriver 파일을 원하는 폴더 경로에 가져다 놓습니다.

## Index of /2.43/

	<u>Name</u>	Last modified	Size	ETag
	<a href="#">Parent Directory</a>		-	
	<a href="#">chromedriver_linux64.zip</a>	2018-10-17 02:46:13	3.89MB	1a67148288f4320e5125649f66e02962
	<a href="#">chromedriver_mac64.zip</a>	2018-10-17 04:09:49	5.71MB	249108ab937a3bf8ae8fd22366b1c208
	<a href="#">chromedriver_win32.zip</a>	2018-10-17 03:01:50	3.45MB	d238c157263ec7f668e0ea045f29f1b7
	<a href="#">notes.txt</a>	2018-10-17 05:00:45	0.02MB	a84902c9429641916b085a72ad5de724

## 인스타그램 크롤링

해당 코드에서 사용된 Selenium 패키지의 다양한 메서드 정보는 따로 PDF 파일로 구성되어 있으므로, 해당 내용에 대한 참고를 해주시면 되겠습니다. 우선, 테스트 코드를 작성하기 위해서 웹 사이트 분석 및 시나리오를 구성해야 한다고 말씀드렸습니다. 여기서 말하는 시나리오는, 해당 사이트가 어떤 식으로 우리에게 정보를 제공해주는지 파악하는 작업이라고 할 수 있습니다.

```

import time # 브라우저 조작을 통해 사이트 접근시 프로그램의 중간에 대기를 위한 모듈(웹 자원을
가져오는 네트워크 환경보다 코드가 너무 빠르게 동작하기 때문입니다.)
from selenium import webdriver # 브라우저 조작을 위한 모듈

URL = 'https://www.instagram.com/explore/tags/{}' # HashTag에 해당하는 URL 포맷팅
DRIVER_DIR = '/Users/temp/Project_FC/chromedriver' # 로컬 컴퓨터에 설치된 크롬 드라이버
경로

# 인스타그램 크롤링 함수 정의, 매개변수는 태그에 해당하는 키워드
def instagram_scrap(keyword):
    try:
        driver = webdriver.Chrome(DRIVER_DIR) # 크롬 브라우저 조작을 위한 객체 선언
        driver.implicitly_wait(10) # 웹 자원 가져오기까지 10초간 대기
        driver.get(URL.format(keyword)) # 미리 만들어진 URL 문자열 마지막 부분에 키워드
        키워드 넣기

        new_links = driver.find_elements_by_css_selector('div.v1Nh3 > a') # 웹
        사이트 분석을 통한 DOM 구조 접근
        links = [i.get_attribute('href') for i in new_links] # 각각의 링크의 연결
        주소만을 리스트로 반환
        print('content-length: ', len(links)) # 가져온 결과값 반환

        # 가져온 개별 게시물 링크만큼 반복
        for link in links:
            driver.get(link) # URL 이동
            time.sleep(1)
            # 게시물의 댓글 정보를 담고 있는 DOM 구조에 접근하여 찾은 크기만큼 반복
            for li in driver.find_elements_by_class_name('C4VMK'):
                user = li.find_element_by_tag_name('a').text # 작성자
                reply = li.find_element_by_tag_name('span').text # 댓글, 해시태그
                print("{} {}".format(user, reply)) # 결과 출력

    except Exception as e:
        print(e)
    finally:
        driver.quit() # 드라이버 닫기

if __name__ == "__main__":
    keyword = input('keyword(tag)') # 키워드 사용자 입력
    instagram_scrap(str(keyword)) # 인스타그램 크롤링 함수 호출

```

이미 느끼셨겠지만, 위 코드의 흐름은 우리가 인스타그램이라는 웹 사이트에서 게시물에 접근하여 댓글을 보기 위한 흐름과 동일합니다. 이처럼 단순히, 우리가 실제 브라우저에서 취하는 행동을 코드로 옮겨 놓는 논리적 흐름이라고 생각하시면 이해하시기 편하실 것 같습니다. 그렇기 때문에, 추출하고 싶은 다른 정보, 혹은 더 많은 정보를 가져오기 위해서 취해야 하는 코드 등은, 우리가 실제 인스타그램 사이트에서 취하는 행동을 코드로 옮겨 놓기만 하면 됩니다.

참고로, 인스타그램 사이트는 정보를 가져오고 보여주는 렌더링 방법이 타 사이트와 차이가 있습니다. 때문에 twitter 크롤링 파트에서 알아보는 자바스크립트 코드를 작성하여 데이터를 늘리는 방법에서 URL을 정상적으로 가져오기 어려울 수 있습니다. 이러한 경우에 취할 수 있는 방법 중 대표적인 것이 스크롤이 진행될 때마다 URL이 리스트에 있는지 확인하고, 값을 추가하는 방법인데요, 말이 조금 어렵게 느껴지실 수가 있지만, 전혀 어려운 내용이 아닙니다. 해당 내용이 추가된 코드를 살펴보도록 하겠습니다.

```

import time # 브라우저 조작을 통해 사이트 접근시 프로그램의 중간에 대기를 위한 모듈
from selenium import webdriver # 브라우저 조작을 위한 모듈

URL = 'https://www.instagram.com/explore/tags/{}' # HashTag에 해당하는 URL 포매팅
DRIVER_DIR = '/Users/temp/Project_FC/chromedriver' # 로컬 컴퓨터에 설치된 크롬 드라이버
경로

# 인스타그램 크롤링 함수 정의, 매개변수는 태그에 해당하는 키워드
def instagram_scrap(keyword):
    try:
        driver = webdriver.Chrome(DRIVER_DIR) # 크롬 브라우저 조작을 위한 객체 선언
        driver.implicitly_wait(10) # 웹 자원 가져오기까지 10초간 대기
        driver.get(URL.format(keyword)) # 미리 만들어둔 URL 문자열 마지막 부분에 키워드
        끼워 넣기

        new_links = [] # 최종 연결 주소들을 담을 리스트 선언
        no_page = 0 # 페이지 스크롤을 위한 변수 선언
        while no_page < 10: # 페이지 스크롤 10번 반복
            driver.execute_script('window.scrollTo(0,
document.body.scrollHeight)') # 페이지 스크롤
            for i in driver.find_elements_by_css_selector('div.v1Nh3 > a'): # 웹
            사이트 DOM 구조에 접근하여 리스트로 반환
                link = i.get_attribute('href') # 실제 연결 주소
                if not (link in new_links): # URL 주소가 결과 리스트에 있는지 확인
                    new_links.append(link) # URL 주소가 결과 리스트에 없다면 주소 할당
                no_page += 1 # 스크롤 변수 증가
                time.sleep(1.5) # 웹 자원을 가져오는 시간을 기다리기 위해 코드 일시 중지

            print('content-length: ', len(new_links)) # 가져온 결과값 반환

        # 가져온 개별 게시물 링크만큼 반복
        for link in links:
            driver.get(link) # URL 이동
            time.sleep(1)
            # 게시물의 댓글 정보를 담고 있는 DOM 구조에 접근하여 찾은 크기만큼 반복
            for li in driver.find_elements_by_class_name('C4VMK'):
                user = li.find_element_by_tag_name('a').text # 작성자
                reply = li.find_element_by_tag_name('span').text # 댓글, 해시태그
                print("{} {}".format(user, reply)) # 결과 출력

    except Exception as e:
        print(e)
    finally:
        driver.quit() # 드라이버 닫기

if __name__ == "__main__":
    keyword = input('keyword(tag)') # 키워드 사용자 입력
    instagram_scrap(str(keyword)) # 인스타그램 크롤링 함수 호출

```

위 코드를 간단히 살펴보면, 가져오는 정보를 증가시키기 위해 while 문을 통해 코드를 스크롤을 하는 과정에서 연결 주소가 결과 리스트에 있는지 확인하고, 없다면 값을 할당하는 방식입니다. 이렇게 코드를 작성하는 이유는, 인스타그램 사이트는 위, 아래로 사용자가 스크롤을 하는 과정에서 동적으로 DOM 구조를 바꾸는 형태로 정보가 보이기 때문에,

기존의 URL을 저장하고 중복을 제거하지 않을 경우에, 원하는 형태로 주소가 가져와지지 않기 때문에 그렇습니다.

DOM 구조를 분석하고, Selenium 메서드를 통해 웹 자원을 가져오는 코드를 작성하는 내용은 어렵지 않게 익숙해지실 수 있습니다. 다만, 이 세상엔 정말 다양한 사이트가 존재하고, 각각의 사이트마다 서로 다른 DOM 구조를 갖습니다. 각각의 사이트에서 원하는 데이터를 크롤링하기 위해서는 어쩔 수 없이 많은 연습이 필요하게 됩니다. 해당 코드를 참고하여 다양한 크롤링 코드를 작성해보세요 :)