Python 페이스북 크롤링

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

이번 장에서는 크롤링 테스트 코드의 마지막 내용인 페이스북 크롤링과 관련된 내용에 대해서 알아보도록 하겠습니다. 페이스북의 경우에는 상대적으로 많은 양의 비정형 데이터를 사용하기 때문에, 렌더링(브라우저에 웹 자원을 표현하는 과정)하는 과정이 오래 걸리는 단점이 있습니다. 하지만, 그만큼 사진, 동영상 등과 같은 비정형 데이터가 굉장히 방대합니다. 분석을 원하는 데이터가 비정형 데이터라면 페이스북의 다양한 데이터를 크롤링하는 것도 좋을 거라 생각되네요. ^^ 앞선 내용의 크롤링 코드를 모두 이해하실 수 있다면, 어렵지 않게 크롤링 코드를 이해하실 수 있을 거라 생각합니다.

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
import time

SEARCH_URL = "https://www.facebook.com/search/videos/?q={}"
LOGIN_URL = "https://www.facebook.com/"
DRIVER_DIR = '/Users/temp/Project_FC/chromedriver'
ID = "아이디"
PW = "비밀번호"

def facebook_scrap(keyword):
    try:
        driver = webdriver.Chrome(DRIVER_DIR)
        driver.implicitly_wait(10)
        driver.get(LOGIN_URL)
        time.sleep(1)
        e = driver.find_element_by_id('email')
        e.clear()
        e.send_keys(ID)
        e = driver.find_element_by_id('pass')
        e.clear()
        e.send_keys(PW)
        e.send_keys(Keys.ENTER)
        print('로그인')

        driver.get(SEARCH_URL.format(keyword))

        links = []
        for link in driver.find_elements_by_css_selector('div._14ax > a'): # 
게시물 링크 a 태그 전체 찾기
            links.append(link.get_attribute('href'))
            break

        print('content-len:', len(links))

        for link in links:
            driver.implicitly_wait(10)
            driver.get(link) # 개별 링크 접속
```

```
            author = driver.find_element_by_class_name('_371y').text # 작성자
            likes = driver.find_element_by_class_name('_1g5v').text # 좋아요 수
            com_share = driver.find_elements_by_class_name('_36_q')
            com = com_share[0].text # 코멘트
            share = com_share[1].text # 공유수
            print('(글 정보) ->', author, likes, com, share)

            time.sleep(1.5) # 네트워크 자원 불러오기까지 대기
            comment_btn = driver.find_element_by_class_name('_2xui') # 댓글 보기
            comment_btn.click() # 클릭
            time.sleep(1.5) # 네트워크 자원 불러오기까지 대기

            try:
                view_more = driver.find_element_by_class_name('UFIPagerLink')
                view_more.click() # 댓글 더보기 클릭
            except:
                pass

            classes =
driver.find_elements_by_class_name('UFICommentActorAndBodySpacing')

            for clas in classes:
                user =
clas.find_element_by_css_selector('div.UFICommentActorAndBodySpacing > span >
a').text # 글쓴이 가져오기
                reply = clas.find_element_by_class_name('UFICommentBody').text
# 댓글 가져오기
                print("({}): {}".format(user, reply))
    except Exception as e:
        print(e)
    finally:
        driver.quit()

if __name__ == "__main__":
    keyword = input('keyword?')
    facebook_scrap(str(keyword))
```

위의 페이스북 크롤링 테스트 코드에서 중점적으로 봐야 할 부분이 하나 있습니다. 바로 댓글 더보기 버튼을 클릭하는 부분인데요, 일반적으로 댓글과 같은 정보는 소셜미디어 채널에서 수 천, 수 만개의 정보를 포함하게 됩니다. 그렇기 때문에, 한 번에 모든 정보를 보여주는 것이 아닌, 특정 버튼 혹은, 스크롤 등을 취할 경우 정보를 데이터베이스에서 더 가지고 와서 보여주는 형식으로 되어 있습니다.

Selenium을 통해서 댓글 더보기 버튼을 클릭할 경우에, 버튼을 찾지 못할 경우 "NoSuchElementException"이 발생하게 됩니다. 예외가 발생한 경우 원치 않은 프로그램의 종료가 생길 수 있습니다. 그렇기 때문에, 위 테스트 코드뿐만 아니라, 기타 사이트에서 크롤링을 Selenium 패키지를 통해 진행할 때, 적절한 예외처리는 필수라고 할 수 있습니다.

웹 크롤링 기술은 좋은 방향으로 사용되면(예를 들어, 연구, 학업, 분석 등), 굉장히 활용 가치가 높은 기술입니다. 요새 화두가 되고 있는 데이터 사이언스, 데이터 분석 등의 기술들은 그 근간에 "데이터" 그 자체가 있습니다. 하지만, 이러한 크롤링 기술도 나쁜 방향으로 사용되면(공격적인 크롤링, 불법 배포, 불법 게시 등) 걷잡을 수 없게 나쁜 용도로 사용될 수 있습니다. 이러한 점을 항상 염두하시면서, 내가 크롤링하는 목적에 대한 투명성에 대해서 생각하고, 2차 저작물, 3차 저작물 등의 라이선스를 꼭 확인하시길 바라겠습니다. 감사합니다. :)