KUCIS 대학정보보호동아리연합회 OneScore

기술문서	'14. 03. 06. 작성

파이썬을 활용한 웹크롤러 제작

소속 : 인천대학교 OneScore

작성자 : 최창원

메일 : qwefgh90@naver.com

1. 소개

가. 웹크롤러란?

1) 웹크롤러란 월드 와이드 웹을 탐색하여 원하는 정보를 얻어 내거나 기억시킨 동작을 하는 컴퓨터 프로그램이다. 주로 웹 페이지 크롤링 통해 직접 접근해서 정보를 빠르게 수집하거나 자동 이메일 수집, 웹 유지관리를 위해 사용되기도 한다. 검색엔진에서는 크롤링을 통해 정보를 찾은 후 색인과정을 통해 데이터를 수집한다.

나. 웹크롤러 제작을 위한 준비물

1) 웹크롤러를 제작하기 위해선 HTTP 프로토콜에 대한 지식이 조금 필요하다. 제작 언어로는 어떤 언어든 상관없지만 본 문서에서는 파이썬 [1]언어를 활용하여 제작할 것이다.

2) 파이썬 라이브러리인 BeautifulSoup, requests, selenium등을 사용하면 웹 크롤링을 쉽게 할 수 있다.

3) 웹 표준을 따르는 headless Webkit[2] 이다. 자바스크립트를 사용할 수 있으며 DOM, CSS, JSON, SVG등을 컨트롤 할 수 있다.

2. 라이브러리 소개

가. requests[3]

1) HTTP 프로토콜 조작을 유연하게 할 수 있고 사용이 쉬운 파이썬 라이브러리이다. urllib2, urllib등 여러개로 나눠진 모듈보다 프로토콜 조작을 쉽고 간단하게 할 수 있다.

나. BeautifulSoup[4]

1) HTML이나 XML같은 Markup language에서 Node의 속성이나 값들을 쉽게 가져오거나 수정할 수 있는 파이썬 라이브러리이다.

다. selenium[5]

1) 브라우저와 드라이버로 연결되어 브라우저에서 하는 작업들을 자동화 시킬 수 있다. selenium은 웹 인터페이스로 동작하며 많은 브라우저가 selenium과 호환이 되고 PhantomJS와도 호환이 된다.

3. 샘플 예제

가. 로그인 예제 (POST 요청)

1) 사전조사

http://hi-bogo.net 로그인 후

WireShark에서 로그인에 요청할 URL과 필요한 필드를 확인.

요청 방식 : POST

요청할 페이지 : http://hi-bogo.net/cdsb/login_process.php

로그인에 필요한 정보 : mode=login&kinds=outlogin&user_id=아이디&passwd=비밀번호

2) 샘플소스 (afterlogin.html에서 로그인 성공 여부를 확인할 수 있다)

```
# coding: utf-8
import requests as rq
hibogo_url="http://hi-bogo.net/"
#LOGIN URL
h_login_url="http://www.hi-bogo.net/cdsb/login_process.php"
h_id = "yourID"
h_passwd = "yourPwd"
#hibogo_login function (POST)
(mode=login&kinds=outlogin&user_id=qwefgh90&passwd=qwefgh90)
def hibogo_login():
        post_bd = {'mode':'login','kinds':'outlogin','user_id':h_id,'passwd':h_passwd}
        rs = rq.post(h_login_url,data = post_bd)
        h =rs.headers
        c = rs.content
        a=open('afterlogin.html','wb');a.write('<!DOCTYPE html><head><meta
charset=\"utf-8\"></head>');a.write(c);a.close()
        return True

if __name__=='__main__':
        hibogo_login()
        pass
```

나. 쿠키 활용 예제

1) 사전조사

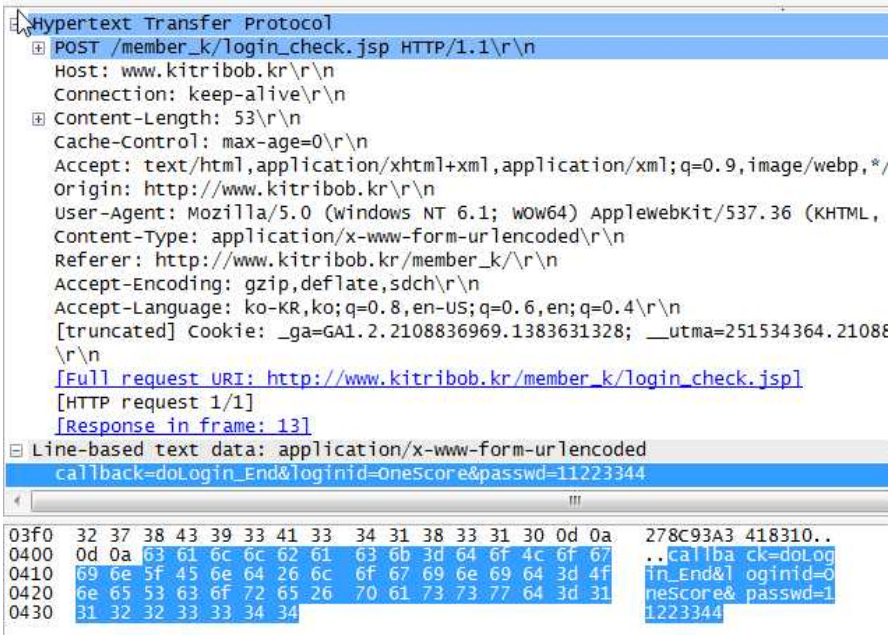
Wireshark를 통해 HTTP Request, Response 필드에 쿠키값 존재 여부 확인

대상 도메인 : http://www.kitribob.kr

요청 방식 : POST

요청할 페이지 : http://www.kitribob.kr/member_k/login_check.jsp

로그인에 필요한 정보 : callback=doLogin_End&loginid=yourID&passwd=yourPassword



2) 샘플소스

```
# coding: utf-8
# by chang
import os,sys,re
import requests as rq
import string
url = 'http://www.kitribob.kr/member_k/login_check.jsp'
value = {'callback':'doLogin_End','loginid':'OneScore','passwd':'11223344'}
response = rq.post(url,data=value)
h= response.headers
bd = response.content
cookie = h['set-cookie']          #cookie 값을 확인할 수 있다.
print cookie
headers=  {'cookie':cookie}      #cookie를 사용하여 로그인 세션유지 가능
#headers 방법 (이게 더 편리함)
```

※Chrome, IE에서 톰캣 웹서버에 접속 할 경우 클라이언트 측에서 로그인 하기 전 미리 세션(쿠키)[6]을 할당 받기 때문에 로그인 후 서버로 부터오는 Response header에는 set-cookie 필드[7]에서 세션값이 따로 포함되어 오진 않는다. 파이썬에서 로그인을 바로 요청할 경우 서버측 에서 세션값을 생성하여 준다. 즉 JSESSIONID를 보내지 않을 경우 서버 “set-cookie“ 필드를 통해 쿠키값을 전해주게 된다.

다. headless browser(PhantomJS)를 활용한 Daum 로그인 & 쿠키값 핸들링

1) 샘플 코드

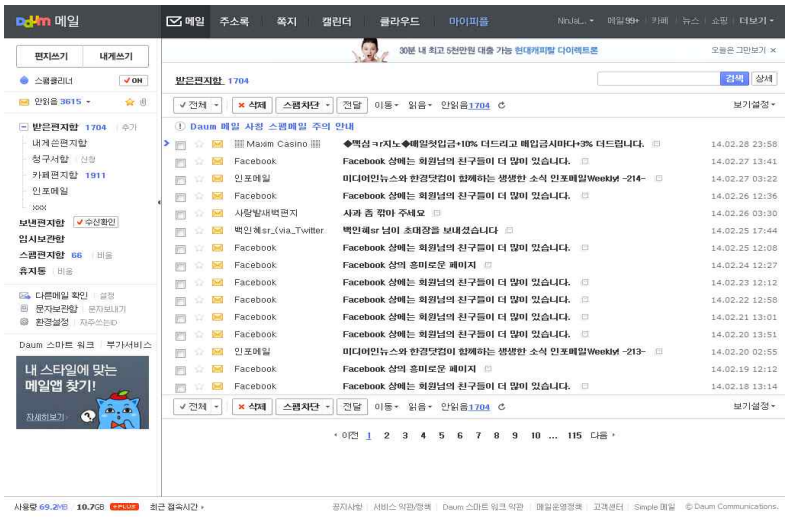
```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait # available since 2.4.0
from selenium.webdriver.support import expected_conditions as EC # available since
2.26.0
daum_url="http://www.daum.net"
d_id = ""
d_passwd = ""
def daum_login():
        driver = webdriver.PhantomJS()
        driver.get(daum_url)
        driver.set_window_size(1024,768)
        idinput = driver.find_element_by_id("id")
        idinput.send_keys(d_id)                                          #아이디 입력
        driver.find_element_by_id("inputPwd").send_keys(d_passwd)  #비밀번호 입력
        driver.save_screenshot('daum_before_login.png')
        login_button = driver.find_element_by_id("loginSubmit")
        login_button.click()          #버튼 클릭
        idinput.submit()              #요소에서 Submit 호출

        driver.get(daum_url)
        driver.save_screenshot('daum_login.png')
        driver.get("http://mail2.daum.net/hanmailex/Top.daum?")
        driver.save_screenshot('daum_mail.png')
```

※ click(), submit() 중 하나의 함수만 시도해보았지만 로그인에 계속 실패하였다. click() 함수는 버튼 노드를 클릭하는 행동을 하는 함수이고 submit() 함수는 form태그에 둘러쌓인 노드에서 호출할 수 있다. 그 후 form태그를 찾고 submit을 호출하는 행동을 한다. click() ,submit() 의 기능상 차이는 없는 것 같지만 둘중 하나가 동작하지 않을 경우 두 개의 함수를 같이 호출해서 해결 할 수 있는 것 같다.

<Daum 로그인 전 폼 데이터 스크린샷>

<Daum 로그인 후 스크린샷>



<Daum 메일 접속 스크린샷>

2) 샘플 코드2 (쿠키값 핸들링)

```
def getKitriPhantom():
        driver = webdriver.PhantomJS()
        driver.get(testurl3)
        cookies = {'name':'JSESSIONID','value':'70EC4B2594509C4703257525A7DE6DFA'}
        driver.add_cookie(cookies)    # 쿠키 추가
        driver.get(testurl2)
        .....

        for cookie in driver.get_cookies():  # 쿠키 정보 확인
                print "%s -> %s" % (cookie['name'], cookie['value'])
        .....
        driver.delete_cookie('__utma')      #__utma 키 값 삭제
```

※ 위와 같은 형태로 팬텀JS에 쿠키값을 변경할 수 있다.

4. 활용 방안

많은 포탈에선 웹 API를 제공해준다. 오히려 크롤링을 위해 파이썬을 배우는 것 보다 웹 API로 다양한 동작들을 구현하는 것이 시간이 절약될 수 도 있다. 필자는 푸쉬알림 기능을 같이 사용하여 웹툰 업데이트 푸시 알림, 영화티켓예매, 공지사항 크롤링이나 해볼 생각이다.

5. 참조

- https://docs.python-requests.org/en/latest/# (Requests)
- http://docs.seleniumhq.org/docs/03_webdriver.jsp#selenium-webdriver
- http://www.crummy.com/software/BeautifulSoup/bs4/doc/#beautiful-soup-documentation
- Web Crawling Lecture with PhantomJS of BoB
- https://developers.google.com/webmasters/control-crawl-index/docs/getting_started?hl=ko

---

1) 파이썬은 스크립트 기반 언어로 적은 량의 코드로 많은 작업을 할 수 있어서 생산성이 매우 높은 언어이다.
2) headless Webkit이란 기본 브라우저와 같은 기능을 하면서 더 빠르게 작업을 처리할 수 있으며 GUI를 포함하지 않은 브라우저이다.
3) http://docs.python-requests.org/en/latest, http://01073144993.tistory.com/153에 자세한 정보
4) http://www.crummy.com/software/BeautifulSoup/bs4/doc/#beautiful-soup-documentation, http://01073144993.tistory.com/154에 자세한 정보
5) http://docs.seleniumhq.org/docs/03_webdriver.jsp#selenium-webdriver ,http://01073144993.tistory.com/155에 자세한 정보
6) http://lng1982.tistory.com/143
7) http://en.wikipedia.org/wiki/HTTP_cookie