파이썬 크롤링 - 2
(정적 웹 크롤링)

최도진

목차

학습목표

- API를 제공하지 않는 웹 페이지를 크롤링할 수 있다.
- BeautifulSoup 라이브러리로 정적 웹 페이지를 크롤링할 수 있다.
- Selenium 라이브러리로 동적 웹 페이지를 크롤링할 수 있다.

01. 정적 웹 페이지 크롤링

■ 정적 웹 페이지 크롤링 준비

- BeautifulSoup 연습하기 1

1. BeautifulSoup 라이브러리를 사용하기 위해 추가 설치작업을 실시:

   명령 프롬프트 창에서 pip 명령을 사용

```
C:₩> pip install beautifulsoup4
```

2. 설치가 끝나면 파이썬 셸 창에서 BeautifulSoup을 임포트하여 사용

```
>>> from bs4 import BeautifulSoup
```

3. 연습용 html을 작성

```
>>> html = '<h1 id="title">한빛출판네트워크</h1><div class="top"><ul
class="menu"><li><a href=http://www.hanbit.co.kr/member/login.html
class="login">로그인 </a></li></ul><ul class="brand"><li><a href="http://www.
hanbit.co.kr/media/>한빛미디어<li><a href="http://www.hanbit.co.kr/
academy/">한빛아카데미</a></li></ul></div>'
```

4. BeautifulSoup 객체를 생성

```
>>> soup = BeautifulSoup(html, 'html.parser')
```

01. 정적 웹 페이지 크롤링

■ 정적 웹 페이지 크롤링 준비

- BeautifulSoup 연습하기 1

5. 객체에 저장된 html 내용을 확인

```
>>> print(soup.prettify())
<h1 id="title">
 한빛출판네트워크
</h1>
<div class="top">
 <ul class="menu">
  <li>
   <a class="login" href="http://www.hanbit.co.kr/member/login.html">
    로그인
   </a>
  </li>
 </ul>
 <ul class="brand">
  <li>
   <a href="http://www.hanbit.co.kr/media/">
    한빛미디어
   </a>
  </li>
  <li>
   <a href="http://www.hanbit.co.kr/academy/">
    한빛아카데미
   </a>
  </li>
 </ul>
</div>
```

01. 정적 웹 페이지 크롤링

■ 정적 웹 페이지 크롤링 준비

- BeautifulSoup 연습하기 2

1. 태그 파싱하기 - 지정된 한 개의 태그만 파싱

```
>>> soup.h1
<h1 id="title">한빛출판네트워크</h1>
>>> tag_h1 = soup.h1
>>> tag_h1
<h1 id="title">한빛출판네트워크</h1>
>>> tag_div = soup.div
>>> tag_div
<div class="top"><ul class="menu"><li><a class="login" href="http://www.
hanbit.co.kr/member/login.html">로그인</a></li></ul><ul class="brand"><li><a
href="http://www.hanbit.co.kr/media/">한빛미디어</a></li><li><a href="http://www.
hanbit.co.kr/academy/">한빛아카데미</a></li></ul></div>
>>> tag_ul = soup.ul
>>> tag_ul
<ul class="menu"><li><a class="login" href="http://www.hanbit.co.kr/member/
login.html">로그인</a></li></ul>
>>> tag_li = soup.li
>>> tag_li
<li><a class="login" href="http://www.hanbit.co.kr/member/login.html">로그인</a></li>
>>> tag_a = soup.a
>>> tag_a
<a class="login" href="http://www.hanbit.co.kr/member/login.html">로그인</a>
```

01. 정적 웹 페이지 크롤링

■ 정적 웹 페이지 크롤링 준비

- BeautifulSoup 연습하기 2

  2. 태그 파싱하기 - 지정된 태그를 모두 파싱

```
>>> tag_ul_all = soup.find_all("ul")
>>> tag_ul_all
[<ul class="menu"><li><a class="login" href="http://www.hanbit.co.kr/member/login.html">로그인
</a></li></ul>,<ul class="brand"><li><a href="http://www.hanbit.co.kr/media/">한빛미디어</a></li><li><a
href="http://www.hanbit.co.kr/academy/">한빛아카데미</a></li></ul>]
>>> tag_li_all = soup.find_all("li")
>>> tag_li_all
[<li><a class="login" href="http://www.hanbit.co.kr/member/login.html">로그인</a></li>,<li><a
href="http://www.hanbit.co.kr/media/">한빛미디어</a></li>,<li><a href="http://www.hanbit.co.kr/academy/">
한빛아카데미</a></li>]
>>> tag_a_all = soup.find_all("a")
>>> tag_a_all
[<a class="login" href="http://www.hanbit.co.kr/member/login.html">로그인</a>,<a
href="http://www.hanbit.co.kr/media/">한빛미디어</a>,<a href="http://www.hanbit.co.kr/academy/">한빛아카데
미</a>]
```

01. 정적 웹 페이지 크롤링

■ 정적 웹 페이지 크롤링 준비

- BeautifulSoup 연습하기 2

3. 속성을 이용하여 파싱

❶ attrs: 속성 이름과 속성값으로 딕셔너리 구성

❷ find( ): 속성을 이용하여 특정 태그 파싱

❸ select( ): 지정한 태그를 모두 파싱하여 리스트 구성　　태그#id 속성값　/　태그.class 속성값

```
>>> tag_a.attrs
{'href': 'http://www.hanbit.co.kr/member/login.html', 'class': ['login']}
>>> tag_a['href']
'http://www.hanbit.co.kr/member/login.html'
>>> tag_a['class']
['login']
>>> tag_ul_2 = soup.find('ul', attrs={'class':'brand'})
>>> tag_ul_2
<ul class="brand"><li><a href="http://www.hanbit.co.kr/media/">한빛미디어</a></
li><li><a href="http://www.hanbit.co.kr/academy/">한빛아카데미</a></li></ul>
>>> title = soup.find(id="title")
>>> title
<h1 id="title">한빛출판네트워크</h1>
>>> title.string
'한빛출판네트워크'
>>> li_list = soup.select("div>ul.brand>li")
>>> li_list
[<li><a href="http://www.hanbit.co.kr/media/">한빛미디어</a></li>,<li><a
href="http://www.hanbit.co.kr/academy/">한빛아카데미</a></li>]
>>> for li in li_list: [Enter]
        print(li.string) [Enter]
        [Enter]
한빛미디어
한빛아카데미
```

01. 정적 웹 페이지 크롤링

■ 정적 웹 페이지 크롤링 실습

▪ 크롤링 허용 여부 확인하기

- 웹 페이지를 크롤링하기 전에 크롤링 허용 여부를 확인하기 위해 주소 창에 '크롤링할 주소/ robots.txt'를 입력
- 만약 robots.txt 파일이 없다면 수집에 대한 정책이 없으니 크롤링을 해도 된다는 의미

표시	허용 여부
User-agent: * Allow: / 또는 User-agent: * Disallow:	모든 접근 허용
User-agent: * Disallow: /	모든 접근 금지
User-agent: * Disallow: /user/	특정 디렉토리만 접근 금지

그림 6-1 특정 디렉토리만 접근 금지를 한 예

01. 정적 웹 페이지 크롤링

■ 정적 웹 페이지 크롤링 실습

- 웹 페이지 분석하기

1. 매장 정보 찾기

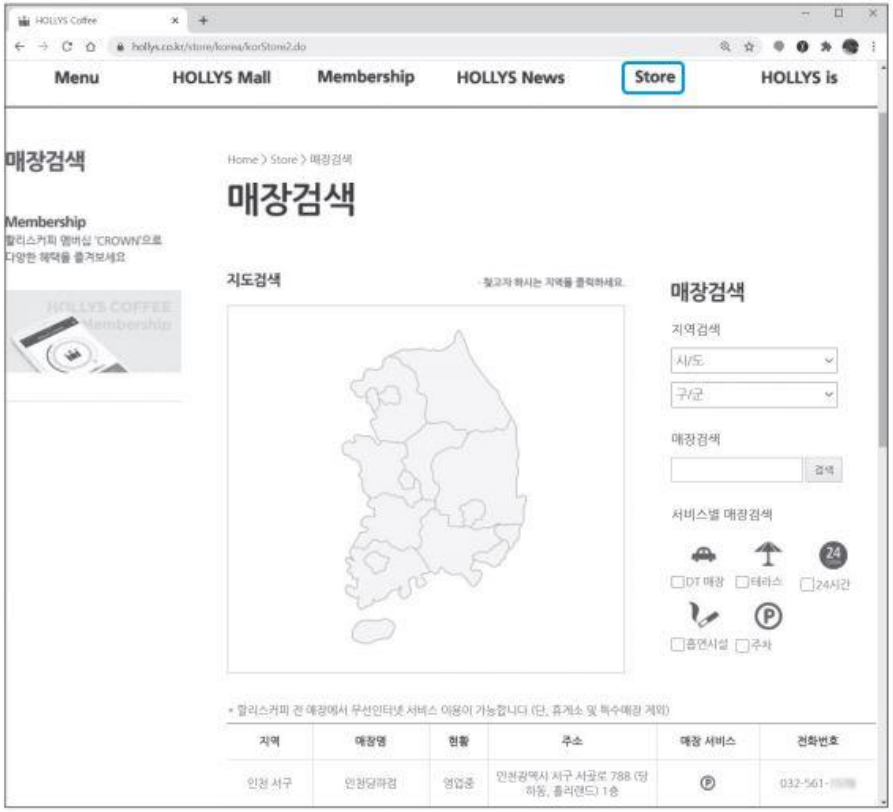

그림 6-2 매장 정보를 찾을 수 있는 매장검색 페이지

01. 정적 웹 페이지 크롤링

■ 정적 웹 페이지 크롤링 실습

- 웹 페이지 분석하기

2. HTML 코드 확인하기

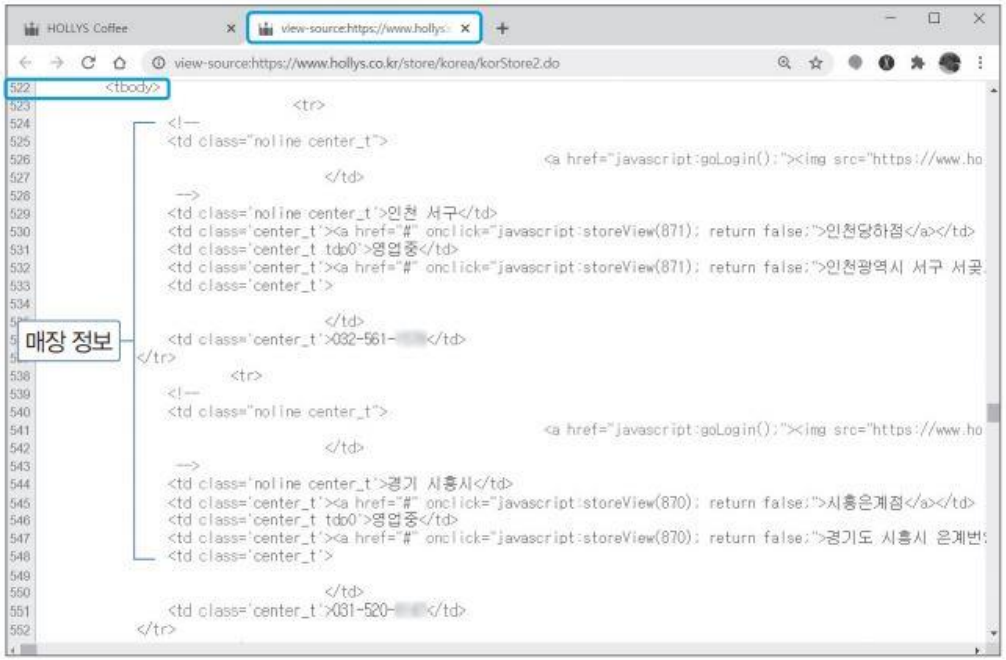

그림 6-3 매장 정보에 대한 HTML 코드

01. 정적 웹 페이지 크롤링

■ 정적 웹 페이지 크롤링 실습

- 웹 페이지 분석하기

3. 나머지 매장 정보 확인하기

(a) 매장 정보 2페이지

(b) 매장 정보 2페이지의 HTML 코드

그림 6-4 2페이지 이동 후 HTML 코드 확인

01. 정적 웹 페이지 크롤링

■ 정적 웹 페이지 크롤링 실습

- 웹 페이지 분석하기

4. 'pageNo=' 다음에 페이지 번호를 붙여 다음 페이지를 확인

그림 6-5 마지막 페이지 확인

01. 정적 웹 페이지 크롤링

■ 정적 웹 페이지 크롤링 실습

- 파이썬 셸 창에서 크롤링하기

1. BeautifulSoup과 urllib.request를 임포트

```
>>> from bs4 import BeautifulSoup
>>> import urllib.request
```

2. 작업 결과를 저장할 리스트를 준비

```
>>> result = []
```

01. 정적 웹 페이지 크롤링 – 테스트

■ lib 추가

```
from bs4 import BeautifulSoup
import requests
```

■ page 변수에 따른 url 변경

```
url = 'https://www.hollys.co.kr/store/korea/korStore2.do?pageNo=%d&sido=&gugun=&store=' % page

resp = requests.get(url)
```

01. 정적 웹 페이지 크롤링 – 테스트

■ 응답 확인 (html 확인)

```
soupHollys = BeautifulSoup(resp.content, 'html.parser')
print(soupHollys.prettify())
```

01. 정적 웹 페이지 크롤링 – 테스트

■ 파싱 타겟 검색

```
tag_tbody = soupHollys.find('tbody')
print(tag_tbody.prettify())
```

- tbody = 테이블 바디
- tr = 테이블 로우 (행)

01. 정적 웹 페이지 크롤링 – 테스트

■ table에 저장 된 모든 매장 검색 (tr)

```
for store in tag_tbody.find_all('tr') :
    print("==========================")
    print(store)
    if len(store) < 2 :
        break
```

■ 마지막 페이지 / 잘못 된 페이지 조건 추가 (if문, ex) page = 54)

01. 정적 웹 페이지 크롤링 – 테스트

■ 테이블 내부 정보 확인

```
store_td = store.find_all('td')
```

- td = 테이블 데이터 셀

```
<table>
    <tr>
        <th>밥류</th>
        <th>면류</th>
        <th>분식류</th>
    </tr>
    <tr>
        <td>김밥</td>
        <td>라면</td>
        <td>떡볶이</td>
    </tr>
    <tr>
        <td>주먹밥</td>
        <td>우동</td>
        <td>순대</td>
    </tr>
</table>
```

밥류	면류	분식류
김밥	라면	떡볶이
주먹밥	우동	순대

01. 정적 웹 페이지 크롤링 – 테스트

■ 개별 td 정보 파싱

```
store_sido = store_td[0].string
store_name = store_td[1].string
store_addr = store_td[3].string
store_phone = store_td[5].string
#print(store_sido, store_name , store_addr, store_phone)
```

- 시도, 매장명, 주소, 번호

01. 정적 웹 페이지 크롤링 – 테스트

■ 파싱 결과 리스트 추가

```
results.append([store_sido, store_name , store_addr, store_phone])
```

01. 정적 웹 페이지 크롤링 – 테스트

■ 모든 페이지(1~53) 검색 및 파싱

```
results = []

for page in range(1, 54) :
#    page = 53

    url = 'https://www.hollys.co.kr/store/korea/korStore2.do?pageNo=%d&sido=&gugun=&store=' % page

    resp = requests.get(url)
    #print(resp.encoding)
    # print(resp.content)

    soupHollys = BeautifulSoup(resp.content, 'html.parser')
    #print(soupHollys.prettify())

    tag_tbody = soupHollys.find('tbody')
    print(tag_tbody.prettify())

    for store in tag_tbody.find_all('tr') :
        print("==========================")
        print(store)
        if len(store) < 2 :
            break

        store_td = store.find_all('td')

        store_sido = store_td[0].string
        store_name = store_td[1].string
        store_addr = store_td[3].string
        store_phone = store_td[5].string
        #print(store_sido, store_name , store_addr, store_phone)

        results.append([store_sido, store_name , store_addr, store_phone])

        #print("==========================")
    print(f"----------------{page} page complete----------------")
print(results)
```

01. 정적 웹 페이지 크롤링 – 테스트

■ 파싱 결과 DataFrame으로 생성

- pandas lib 추가

```
hollys_df = pd.DataFrame(results, columns = ['store', 'sidu-gu', 'address', 'phone'])
hollys_df
```

	store	sidu-gu	address	phone
0	강원 홍천군	(상)홍천휴게소R점	강원도 홍천군 화촌면 서울양양고속도로 83 .	.
1	부산 부산진구	부산시민공원점	부산 부산진구 시민공원로 73, 푸드코트피크닉 범전동 200	.
2	충남 홍성군	충남도청점	충청남도 홍성군 홍북읍 신경리 553 .	041-631-4725
3	서울 강남구	강남역2점	서울특별시 강남구 강남대로 422 (역삼동) 역삼동816,816-7,816-8	02-568-9056
4	경남 김해시	김해복음병원점	경상남도 김해시 활천로 44 (삼정동) 할리스 김해복음병원점	055-723-2053
...	...	...	...	...
525	서울 관악구	신림점	서울특별시 관악구 신림로 353-1	02-877-0019
526	서울 중구	태평로점	서울특별시 중구 세종대로 64, 해남빌딩 1층 (태평로2가 70-5) 할리스	02-755-7795
527	서울 용산구	이태원점	서울특별시 용산구 이태원로 179	02-749-8752
528	부산 부산진구	부산서면본점	부산광역시 부산진구 동천로 73, DS타워 1~2층 (부전동 부전동 169-1) 할리스	051-819-9117
529	서울 서대문구	신촌점	서울특별시 서대문구 연세로 34 (창천동 31-12) 할리스	02-393-2004

01. 정적 웹 페이지 크롤링 – 테스트

■ 최종 결과 저장

```
hollys_df.to_csv("hollys.csv")
```

- encoding = 'cp949' 지정 테스트
  - 엑셀 파일로 확인
- File 위치 (C – 사용자 - 사용자명)
  - C:₩Users₩user

01. 정적 웹 페이지 크롤링 – 테스트 2

■ 다빈치 매장 파싱

- http://www.edavinci.co.kr/pg/bbs/board.php?bo_table=store&page=1

01. 정적 웹 페이지 크롤링 – 테스트 2

■ 달라지는 점

```
tag_tbody = soupHDavinci.find('table').find('table')
```

```
for store in tag_tbody.find_all('tr') :
    #print("=========================")

    store_td =  store.find_all('td')
    if len(store_td) > 0 :
```

```
import pprint
results = []

for page in range(1, 5) :
    #page = 1
    url = "http://www.edavinci.co.kr/pg/bbs/board.php?bo_table=store&page=%d" % page
    resp = requests.get(url)

    soupHDavinci = BeautifulSoup(resp.content, 'html.parser')
    #print(soupHDavinci.prettify())

    tag_tbody = soupHDavinci.find('table').find('table')
    #print(tag_tbody.prettify())

    #tag_tbody
    for store in tag_tbody.find_all('tr') :
        #print("=========================")

        store_td =  store.find_all('td')
        if len(store_td) > 0 :
            print(store_td)
            store_sido = store_td[1].string
            store_name = store_td[2].find('a').string
            store_phone = store_td[3].string
            store_addr= store_td[4].string.strip()
         #   print(store_name, store_phone, store_addr)
            results.append([store_sido, store_name, store_phone, store_addr])

pprint.pprint(results)

davinci_df = pd.DataFrame(results, columns = ['sido_gu', 'name', 'phone', 'addr'])
davinci_df
davinci_df.to_csv('divinci.csv', encoding='cp949')
```