Chapter. 02

데이터 크롤링

I requests, BeautifulSoup

FASTCAMPUS

ONLINE

금융공학/퀀트I

강사. 서찬웅

I 이번 시간에 공부할 내용을 알아보겠습니다.

1. http의 방식 get, post
2. requests 패키지 사용법
3. html 문서를 파싱하는 방법
    1. BeautifulSoup 사용법

I 웹 데이터를 수집하는 requests에 대해서 알아보겠습니다.

- 파이썬의 표준 라이브러리에는 웹 클라이언트를 작성하는데 사용되는 urllib 패키지가 있습니다. 이를 사용해서 웹 데이터를 수집할 수 있습니다. 하지만 수업에는 'HTTP for Humans'라는 표어를 사용하는 requests 패키지를 사용해서 데이터를 수집하는 것을 연습해보겠습니다.
- requests 패키지는 지난 시간에 이야기 했듯이 자바스크립트로 둘려쌓인 데이터는 가져올수 없습니다. 하지만 그렇지 않은 경우에서는 빠른 속도로 데이터를 가져올 수 있기 때문에 사용합니다. 또 HTTP for humans이기 때문에 매우 직관적입니다.
- requests 패키지는 Thrid Party Library 입니다. 그렇기 때문에 추가로 설치해야 할 수도 있습니다.
  - pip install requests
- 이번 시간에는 GET 방식과 POST 방식에 대응하는 requests 메소드를 사용하는 방법을 알아보겠습니다.

HTTP 메소드	대응 requests 메소드
GET	requests.get()
POST	requests.post()

FAST CAMPUS
ONLINE
서찬웅 강사.

Ihttp의 응답 상태 코드에 대해서 알아보겠습니다.

- 서버에서 처리 결과를 응답 메시지의 상태 코드(Status Code)를 통해 확인할 수 있습니다.
- 아래는 대표적인 상태 코드를 정리한 표입니다.

상태코드	의미
200 – OK	서버가 요청을 성공적으로 처리
301 – Moved Permanenlty	요청한 리소스가 새로운 URI로 이동
400 – Bad Request	클라이언트의 요청이 잘못 되었음
401 – Unauthorized	인증되지 않아 지정한 리소스에 대한 권한 없음
403 – Forbidden	요청에 대한 허가가 금지되었음
404 – Not Found	지정한 리소스가 존재하지 않음
500 – Internal Server Error	서버 내부 문제 발생

I requests 패키지 설치 및 사용 방법에 대해서 알아보겠습니다.

- requests 패키지의 공식 홈페이지에 있는 문서를 기반으로 설명하겠습니다.
  - https://2.python-requests.org/en/master/user/quickstart/#make-a-request

- requests 패키지에서 GET 요청하는 방법에 대해서 알아보겠습니다.
  - GET 방식은 URL의 정보를 가져오는 방법으로 가장 많이 사용됩니다. 웹 브라우저를 사용하여 서버에서 데이터(웹 페이지, 이미지, 동영상등)을 가져올 때 GET 방식의 많은 요청을 하는 것을 크롬(파이어폭스)의 네트워크 탭에서 확인할 수 있었습니다.
- 예제

```
import requests

r = requests.get('https://api.github.com/events')

payload = {'key1' : 'value1', 'key2' : 'value2'}
r = requests.get('https://httpbin.org/get', params=payload)
r.status_code
r.url
```

'https://httpbin.org/get?key1=value1&key2=value2 ' URL로 접근하는 거와 동일

I 검색 엔진을 통한 GET 방식 이해 및 예제

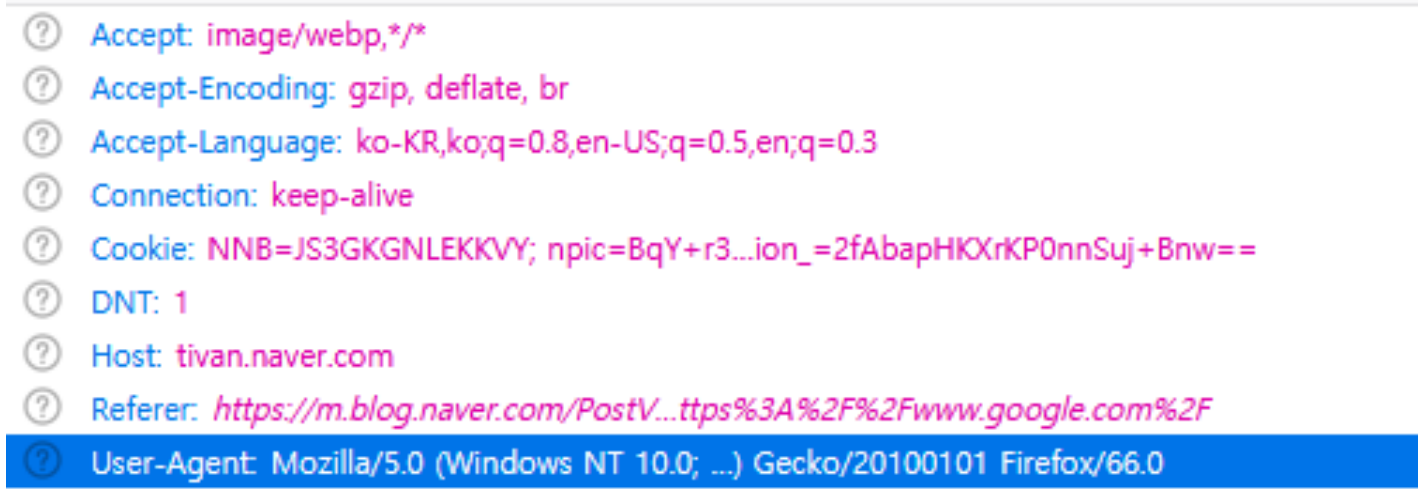
- 네이버 검색 엔진에서 검색을 할 경우 그 방식이 GET 방식으로 전달 된다고 이야기 했습니다. 아래는 네이버에서 퀀트라는 단어를 검색했을 때 URL 주소입니다.
- https://search.naver.com/search.naver?sm=top_hty&fbm=1&ie=utf8&query=퀀트
  호스트명 경로 쿼리스트링
  - GET 주소는 https://search.naver.com/search.naver 이며,
  - 뒤에 있는 값들은 파라미터 값입니다.
- requests 패키지를 이용하여 GET 방식의 URL을 생성해 보겠습니다.

```
naver_data = {'sm' : 'top_hty',
              'fbm' : 1,
              'ie' : 'utf8',
              'query' : '퀀트' }

r = requests.get('https://search.naver.com/search.naver', params=naver_data)
r.url
```

Iheader 값을 수정하여 python으로 접근하는 것을 알지 못하게 하는 방법

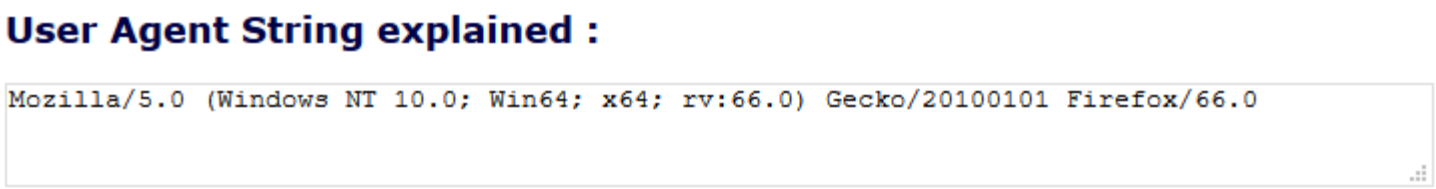
- 클라이언트(지금 사용중인 컴퓨터)에서 서버에 접속할 때 요청 헤더라는 정보도 같이 전송합니다. 아래 정보는 제 컴퓨터에서 네이버 블로그에 접속했을 때 전송했던 요청 헤더의 값입니다.



- 위의 그림은 개발자 모드 – 네트워크에서 확인할 수 있고, 예전 수업 시간에 언급했습니다. 여기서 User-Agent를 보면 제가 Firefox로 접속 했다는 사실을 알 수가 있습니다. 하지만 python으로 접속을 하면 아래처럼 표시됩니다.
  - python-requests/2.21.0
- 그렇기 때문에 특정 사이트에선 Python으로 접속을 하면 제대로 된 정보를 제공하지 않습니다.

I header 값을 수정하여 python으로 접근하는 것을 알지 못하게 하는 방법

- requests의 get 메소드에는 headers 매개변수가 존재합니다. 이 값은 필수 값은 아니며, 만약 변경을 하고 싶다면 아래 예제처럼 변경할 수 있습니다.
- 지금 접속하는 브라우저의 헤더 정보를 그대로 파이썬에서 사용하고 싶다면 아래 사이트에 접속하여 정보를 복사합니다.
  - http://www.useragentstring.com/



- 아래는 위의 정보를 바탕으로 header값을 수정하여 GET 메소드 호출할 때 함께 전달합니다.

```
header = {'User-Agent' : 'Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:66.0) Gecko/20100101 Firefox/66.0'}
url = 'http://www.useragentstring.com/'

r = requests.get(url, headers = header)
```

I 결과 값에 대한 대표적인 메소드를 알아보겠습니다.

- 앞에서 requests의 GET 메소드를 사용해서 전달 받은 객체는 Response 객체입니다. 이 객체의 메소드를 사용해서 해당 페이지의 텍스트를 가져올 수도 있으며, 상태 코드값을 출력하고, 본문을 JSON 포맷으로 해석하고 변환하여 반환할수도 있습니다.
- 아래는 대표적인 속성(메소드)를 정리했습니다.

속성	설명
Response.request	요청 정보를 가지고 있는 객체
Response.url	요청한 URL 문자열
Response.cookie	응답에 포함되어 있는 cookie 정보를 가지고 있는 객체
Response.headers	dict 형식의 응답 헤더
Response.status_code	응답의 http 상태 코드
Response.ok	상태 코드가 정상이면 True, 아니면 False
Response.text	응답 본문의 값을 문자열로 출력
Response.json()	응답 본문을 JSON 방식으로 출력하여 dict 형식으로 반환

I post 방식에 대해서 알아보겠습니다.

- 웹 서버로 데이터를 전송할 때 요청 하는 메시지를 get 방식처럼 URL에 담아서 전송하는 것이 아니라 BODY에 데이터를 key-value 형식으로 넣어서 전송합니다. 그래서 get 방식처럼 주소창만 보고 전송하는 데이터를 추측할 수 없습니다.
- 이러한 이유 때문에 웹 페이지의 입력 폼이나 로그인 정보들은 Post방식으로 전달 됩니다.
- 현재 내가 전송한 양식이 get 방식인지 post 방식인지 알고 싶다면?
  - 크롬 혹은 파이어폭스의 네트워크 탭에서 확인할 수 있습니다.(이전 수업 참조)
- requests 패키지에서 post 방식으로 요청하기 위해서는 아래의 메소드를 사용합니다.
  - requests.post() 메소드를 사용하면 post 방식으로 전달할 수 있습니다.

```
import requests

payload = {'key1' : 'value1', 'key2' : 'value2'}

r = requests.post("https://httpbin.org/post", data = payload)
print (r.text)
```

  - data 인수에 dict 형식의 데이터를 전달하면 application/x-www-form-urlencoded 형식의 인수로 변환

I Session에 대해서 알아보겠습니다.

- Session이란 쉽게 이야기하면 웹 브라우저안에서 일정 시간동안에 서버가 해당 사용자를 식별하고 그에 따라 상태를 일정하게 유지시켜주는데 이때 서버-클라이언트 간의 정보 교환이 일어나고, 이 정보를 세션이라고 합니다.
- requests에서는 Session 객체를 사용하면 서버-클라이언트의 연결을 유지할 수 있습니다.
  - s = requests.Session()
- 여러 페이지를 연속으로 크롤링할 때 효과적으로 사용할 수 있습니다. Session 객체를 사용하면 http 헤더 또는 인증 등의 설정을 한 번만 수행하고 그 상태 값 그대로 여러 번 사용할 수 있습니다.

```
s = requests.Session()
s.headers.update({'user-agent' :
    "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.157 Safari/537.36"})

r = s.get('http://naver.com')
r = s.get('http://daum.net')
```

  - Session 객체의 header의 값을 한번 변경하면 계속 사용할 수 있습니다.

I Session() 메소드를 with와 함께 사용하기

- 앞장의 Session() 메소드를 with와 함께 사용하겠습니다. 예전에 설명했듯이 with 구문을 사용하면 context manager를 사용하여 효과적으로 처리할 수 있습니다.

```
with requests.Session() as s:
    s.headers.update({'user-agent' :
    "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.157 Safari/537.36"})

    r = s.get('http://naver.com')
    r = s.get('http://daum.net')
```

- with 구문은 context manager가 __enter__ 메소드와 __exit__ 메서드를 호출한다고 했습니다. 실제 존재하는지 확인해 보겠습니다.
- 처음 진입할 때 __enter__ 메소드가 실행되며, 종료될 때 자동으로 __exit__ 메소드가 실행될 것입니다.

```
In [22]: dir(s)
Out[22]:
['__attrs__',
 '__class__',
 '__delattr__',
 '__dict__',
 '__dir__',
 '__doc__',
 '__enter__',
 '__eq__',
 '__exit__',
```

I html 파일을 쉽게 파싱할 수 있는 BeautifulSoup에 대해서 알아보겠습니다.

- Beautiful Soup는 웹 크롤러에서 가장 중요한 요소 중 하나인 라이브러리이며, 번거로운 작업들을 아주 간단하게 해결하게 만들어 줍니다.
- https://www.crummy.com/software/BeautifulSoup/bs4/doc/
- BeautifulSoup는 Standard Library가 아니기 때문에 설치가 필요합니다.
  - pip install bs4
- Anaconda는 기본적으로 설치되어 있습니다.
- Beautiful Soup는 아래에서 따왔다고 합니다.

"Beautiful Soup, so rich and green,
Waiting in a hot tureen!
Who for such dainties would not stoop?
Soup of the evening, beautiful Soup!"

- 이상한 나라에서 나와서, 이상한 것들을 잘 이해 할 수 있다고 합니다.
- 사실 BeautifulSoup라는 패키지는 2012년에 개발이 종료, 현재는 BeautifulSoup4입니다.

Ibs의 예제를 통해서 알아보겠습니다.

- bs 공식 홈페이지에 있는 내용으로 간단히 설명하겠습니다. 지금 간단히 하는 이유는 이 과정 끝날때까지 사용할 것이며, 미리 힘 빼지 마세요.

```
html_doc = """
<html><head><title>The Dormouse's story</title></head>
<body>
<p class="title"><b>The Dormouse's story</b></p>

<p class="story">Once upon a time there were three little sisters; and their names were
<a href="http://example.com/elsie" class="sister" id="link1">Elsie</a>,
<a href="http://example.com/lacie" class="sister" id="link2">Lacie</a> and
<a href="http://example.com/tillie" class="sister" id="link3">Tillie</a>;
and they lived at the bottom of a well.</p>

<p class="story">...</p>
"""
```

```
from bs4 import BeautifulSoup
soup = BeautifulSoup(html_doc, 'html.parser')

print(soup.prettify())
# <html>
#  <head>
#   <title>
#    The Dormouse's story
#   </title>
#  </head>
#  <body>
#   <p class="title">
#    <b>
#     The Dormouse's story
#    </b>
#   </p>
#   <p class="story">
#    Once upon a time there were three little sisters; and their names were
#    <a class="sister" href="http://example.com/elsie" id="link1">
#     Elsie
#    </a>
#    ,
#    <a class="sister" href="http://example.com/lacie" id="link2">
#     Lacie
#    </a>
#    and
#    <a class="sister" href="http://example.com/tillie" id="link2">
#     Tillie
#    </a>
#    ; and they lived at the bottom of a well.
#   </p>
#   <p class="story">
#    ...
#   </p>
#  </body>
# </html>
```

prettify() 메소드를 사용하여
html를 가독성이 좋게 출력했습니다.

Ibs가 사용하는 파서의 종류에 대해서 알아보겠습니다.

- BeautifulSoup(html_doc, “html.parser”)는 html_doc 웹 문서를 html.parser 파서를 사용하여 파싱하겠다는 의미입니다. html.parser 이외에 몇 가지 파서들이 존재합니다.

파서	지정 매개변수	설명
표준 라이브러리의 html.parser	html.parser	기본 설치되어 있는 파서
lxml의 html 파서	lxml	빠른 처리 가능
html5lib	html5lib	html5 문서에 맞게 파싱

Ibs의 설명을 계속...

- 소스에서 사용된 BeautifulSoup라는 것에 대해서 알아보겠습니다.
- BeautifulSoup 클래스

형식	self, markup='', features=None, builder=None, parse_only=None, from_encoding=None, exclude_encodings=None, **kwargs)
설명	html을 해석한다.
인수	markup - 해석 대상 html features - parser를 지정한다.
반환값	BeautifulSoup 객체

- BeautifulSoup 객체는 태그 정보를 바탕으로 정보를 탐색할 수 있습니다.

Ibs의 메소드 사용법

- 우선은 아래 코드들을 타이핑 해보겠습니다.

```
soup.title
```
```
<title>The Dormouse's story</title>
```
문서의 title 태그의 정보를 출력합니다.

```
soup.title.name
```
```
'title'
```
title 태그의 이름을 출력합니다.

```
soup.title.string
```
```
"The Dormouse's story"
```
title 태그가 가지고 있는 내용을 출력합니다.

```
soup.p
```
```
<p class="title"><b>The Dormouse's story</b></p>
```
문서의 p 태그의 정보를 출력합니다.

```
soup.p['class']
```
```
['title']
```
p 태그가 가지고 있는 class를 출력합니다.

```
soup.a
```
```
<a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>
```
a 태그의 정보를 출력합니다.

```
soup.find_all('a')
```
```
[<a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>,
 <a class="sister" href="http://example.com/lacie" id="link2">Lacie</a>,
 <a class="sister" href="http://example.com/tillie" id="link3">Tillie</a>]
```
문서를 보면 a태그가 하나가 아니기 때문에 전체를 탐색하기 위해서 find_all()를 사용합니다.

```
soup.find(id="link3")
```
```
<a class="sister" href="http://example.com/tillie" id="link3">Tillie</a>
```
id가 link3라는 것만 출력합니다.

Ibs의 메소드 사용법

- BeautifulSoup.find()

형식	name=None, attrs={}, recursive=True, text=None, **kwargs
설명	요소를 탐색하여 맨 처음 발견된 하나를 얻는다.
인수	name - 요소 이름을 검색 조건으로 한다. attrs - 요소의 속성 이름과 값을 검색 조건으로 한다. 사전형으로 작성한다. recursive - False를 지정하면 바로 아래 자식 요소만 대상이 된다. text - 요소의 내용(시작 태그와 종료 태그 사이의 텍스트)를 검색 조건으로 한다.
반환값	Tag 객체

- id 속성값으로 요소를 찾는 경우 find(id='ID 이름')과 같이 지정합니다.
- class 속성은 find(class_='class 이름')과 같이 하면 해당 클래스를 탐색합니다.
- class_에서 _는 Python 예약어의 class와 충돌을 피하기 위해서 붙어 있습니다.
- find('요소이름', attr={'속성이름':'값}과 같이 지정하면 조건을 세밀히 설정할 수 있습니다.

Ibs의 메소드 사용법

- BeautifulSoup.find_all()

형식	name=None, attrs={}, recursive=True, text=None, limit=None **kwargs
설명	요소를 탐색하여 조건에 일치하는 모두를 구한다.
인수	name - 요소 이름을 검색 조건으로 한다. attrs - 요소의 속성 이름과 값을 검색 조건으로 한다. 사전형으로 작성한다. recursive - False를 지정하면 바로 아래 자식 요소만 대상이 된다. text - 요소의 내용(시작 태그와 종료 태그 사이의 텍스트)를 검색 조건으로 한다. limit - 지정한 수만큼 요소가 발견되면 검색을 종료한다.
반환값	Tag 객체의 리스트

  - 조건에 일치하는 Tag 객체의 리스트를 반환합니다.

- Tag 객체의 리스트를 순회하면서 get메서드를 사용하여 해당 키에 맞는 값을 출력

```
for link in soup.find_all('a'):
    print (link.get('href'))
```

```
http://example.com/elsie
http://example.com/lacie
http://example.com/tillie
```

I 정리

- requests 모듈 설치
- http의 get 방식과 post 방식의 차이 및 사용법
- Session 사용법
- BeautifulSoup 설치 및 개요
- html 파싱
- 데이터 수집의 기초 문법들....

FAST CAMPUS
ONLINE
서찬웅 강사.

감사합니다

FAST CAMPUS
ONLINE
서찬웅 강사.