



## 【零基础学爬虫】beautifulsoup基础学习

前言

其实爬虫有很多解析方式，他们各有千秋，今天我们就来说一下beautifulsoup，他的好处是继续兼容HTML格式非常的友好。



一 安装&简单使用入门。

01 安装

使用Pip可以很方便的安装：

```
pip install requests
pip install beautifulsoup
```

02 requests入门

```
import requests

## get请求
r = requests.get('https://github.com/timeline.json')
r.json()          ##如果是JSON 响应内容,使用r.json()会自动将json结果转换成dict
r.content()       ##二进制相应内容

headers = {'user-agent': 'my-app/0.0.1'}          #定制请求头
r = requests.get('https://github.com/timeline.json', headers=headers)

payload = {'key1': 'value1', 'key2': 'value2'}     #传递url参数
```

```
r = requests.get("http://httpbin.org/get", params=payload)

## post请求
payload = {'key1': 'value1', 'key2': 'value2'}          ## post数据
r = requests.post("http://httpbin.org/post", data=payload)

## 上传文件
url = 'http://httpbin.org/post'
files = {'file': open('report.xls', 'rb')}
r = requests.post(url, files=files)
```

更多详细的请查看官方的中文文档，详细易懂权威。

[http://docs.python-requests.org/zh\\_CN/latest/user/quickstart.html](http://docs.python-requests.org/zh_CN/latest/user/quickstart.html)

## 03 beautifulsoup入门

beautifulsoup可以快速的去定位HTML文档，HTML是用来描述网页的一种超文本标记语言，不是一种编程语言。如果你没有HTML基础，可以去花一天的时间了解下。 菜鸟教程--HTML

<http://www.runoob.com/html/html-tutorial.html>

### 注意的点

- 现在假设知道了HTML是个什么东西，你会发现HTML就是由一层又一层的tag组成，每个tag节点有自己的class属性或者其他属性、自己的父tag、子tag以及兄弟tag，而beautifulsoup的作用就是通过这种蛛丝马迹，轻易的把你想要的凶手。。哦不目标节点揪出来，免去了写正则表达式的繁琐噩梦。
- beautifulsoup对HTML的解释是依赖第三方解释库的，常用的有html.parser、lxml、支持xml解释的lxml-xml、html5Lib,各有优缺点，根据我的经验，有时候使用beautifulsoup返回的待处理文本，会缺少一些tag节点，但是你又确定这不是动态加载的话，其实这是因为解释器无法解释，直接跳过导致的，这时候，可以更换解释器尝试一下。

### 常用的方法

我这里只用下find和find\_all作为实例，详细的用法请去权威易懂的官方文档。毕竟做搬运工是件很累且无意义的事情。

[http://beautifulsoup.readthedocs.io/zh\\_CN/latest/](http://beautifulsoup.readthedocs.io/zh_CN/latest/)

```
from bs4 import BeautifulSoup
##配合requests
r = requests.get('http://www.douban.com')
##一锅待处理的soup汤
soup = BeautifulSoup(r.content, 'lxml')#使用lxml解释库
print(soup)
我们会得到如下的
```

我们会得到如下的soup体，然后定位到红色框的a块。

```

<ul>
<li> <a class="lnk-book" href="https://book.douban.com" target="_blank">豆瓣读书</a></li>
<li> <a class="lnk-movie" href="https://movie.douban.com" target="_blank">豆瓣电影</a></li>
<li> <a class="lnk-music" href="https://music.douban.com" target="_blank">豆瓣音乐</a></li>
<li> <a class="lnk-group" href="https://www.douban.com/group/" target="_blank">豆瓣小组</a></li>
<li> <a class="lnk-events" href="https://www.douban.com/location/" target="_blank">豆瓣同城</a></li>
<li> <a class="lnk-fm" href="https://douban.fm" target="_blank">豆瓣FM</a></li>
<li> <a class="lnk-shijian" href="https://www.douban.com/time/?dt time source=douban-web anonymous index top nav" target="_blank">豆瓣时间</a></li>
<li> <a class="lnk-market" href="https://market.douban.com?utm_campaign=anonymous_top_nav&utm_source=douban&utm_medium=referral" target="_blank">豆瓣市集</a></li>
</ul>

```

通过属性定位查找该节点 (find)

```

a = soup.find('a',attrs={'class':'lnk-book'})
print(a)
print('链接: '+a['href'])
print('文字: '+a.text)

```

```

E:\workspace\python_enve\py3_origin_enve\Scripts\python3.exe E:/workspace/python_project/01_爬虫入门/02_豆瓣图书/01_爬取豆瓣图书top250.py
<a class="lnk-book" href="https://book.douban.com" target="_blank">豆瓣读书</a>
链接: https://book.douban.com
文字: 豆瓣读书

```

返回包含所有该节点的列表 (find\_all)

```

a_s = soup.find_all('a') print (a_s)

```

```

[<a class="lnk-book" href="https://book.douban.com" target="_blank">豆瓣读书</a>, <a class="lnk-movie" href="https://movie.douban.com" target="_blank">豆瓣电影</a>, <a class="lnk-music" href="https://music.douban.com" target="_blank">豆瓣音乐</a>, <a class="lnk-group" href="https://www.douban.com/group/" target="_blank">豆瓣小组</a>, <a class="lnk-events" href="https://www.douban.com/location/" target="_blank">豆瓣同城</a>, <a class="lnk-fm" href="https://douban.fm" target="_blank">豆瓣FM</a>, <a class="lnk-shijian" href="https://www.douban.com/time/?dt time source=douban-web anonymous index top nav" target="_blank">豆瓣时间</a>, <a class="lnk-market" href="https://market.douban.com?utm_campaign=anonymous_top_nav&utm_source=douban&utm_medium=referral" target="_blank">豆瓣市集</a>]

```

提示：有时候需要先将目标范围逐层缩小，这样容易获取目标节点。

## 二 爬取豆瓣图书top250

分析页面。

1、我们点击底部的页码，会发现页数是25的倍数，从0开始，这样我们就可以构造对应页的url了。



2、我们定位到每页，查找书本的信息，获取对应的url，下图为每页爬取到的书本详情页url。

▼<div id="content">

<h1>豆瓣图书 Top 250</h1> == \$0

▼<div class="grid-16-8 clearfix">

▼<div class="article">

▼<div class="indent">

<p class="ulfirst"></p>

▼<table width="100%">

▼<tbody>

▼<tr class="item">

▼<td width="100" valign="top">

▼<a class="nbg" href="https://book.douban.com/subject/1770782/" onclick="moreurl(this,{i:'0'})">



</a>

</td>

▼<td valign="top">

▶<div class="pl2">...</div>

<p class="pl">[美] 卡勒德·胡赛尼 / 李继宏 / 上海人民出版社 / 2006-5 / 29.00元</p>

▶<div class="star clearfix">...</div>

<p class="quote" style="margin: 10px 0; color: #666">...</p>

</td>

</tr>

</tbody>

</table>

<p class="ul"></p>

▶<table width="100%">...</table>

<p class="ul"></p>

▶<table width="100%">...</table>

<p class="ul"></p>

定位到所有信息在该节点下

书本信息

公众号 · 猿榜编程

```

E:\workspace\python_enve\py3_origin_enve\Scripts\pytl
https://book.douban.com/subject/4242172/
https://book.douban.com/subject/1083428/
https://book.douban.com/subject/1090043/
https://book.douban.com/subject/1026425/
https://book.douban.com/subject/2256039/
https://book.douban.com/subject/1873231/
https://book.douban.com/subject/1071241/
https://book.douban.com/subject/3995526/
  
```

3、在图书的详情页，我们定位到如下元素。获取该书的书名，评分和评分人数。

</div>

▼<div id="interest\_sect1" class>

▼<div class="rating\_wrap clearfix">

▼<div class="rating\_logo">豆瓣评分</div>

▼<div class="rating\_self clearfix" typeof="v:Rating">

::before

<strong class="ll rating\_num" property="v:average"> 8.7 </strong>

<span property="v:best" content="10.0"></span>

▼<div class="rating\_right">

<div class="ll bigstar45"></div> == \$0

▼<div class="rating\_sum">

▼<span class>

▼<a href="collections" class="rating\_people">

<span property="v:votes">26586</span>

"人评价"

</a>

</span>

</div>

</div>

▼<div id="wrapper">

<div id="dale\_book\_subject\_top\_icon" ad-status="loaded"></div>

▼<h1>

<span property="v:itemreviewed">球状闪电</span> == \$0

<div class="clear"></div>

</h1>

▶<div id="content">...</div>

▶<div id="footer">...</div>

公众号 · 猿榜编程

## 代码编写

```
# -*- coding:utf-8 -*-

#author:waiwen
#email:iwaiwen@163.com
#time: 2017/12/3 12:27

from bs4 import BeautifulSoup
import requests
import random

#user_agent库, 随机选取, 防止被禁
USER_AGENT_LIST = [
    "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.1 (KHTML, like Gecko) Chrome/22.0.1207.1 Safari/537.1",
    "Mozilla/5.0 (X11; CrOS i686 2268.111.0) AppleWebKit/536.11 (KHTML, like Gecko) Chrome/20.0.1132.57 Safari/536.11",
    "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.6 (KHTML, like Gecko) Chrome/20.0.1092.0 Safari/536.6",
    "Mozilla/5.0 (Windows NT 6.2) AppleWebKit/536.6 (KHTML, like Gecko) Chrome/20.0.1090.0 Safari/536.6",
    "Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; 360SE)",
    "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1061.1 Safari/536.3",
    "Mozilla/5.0 (Windows NT 6.1) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1061.1 Safari/536.3",
    "Mozilla/5.0 (Windows NT 6.2) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1061.0 Safari/536.3",
    "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/535.24 (KHTML, like Gecko) Chrome/19.0.1055.1 Safari/535.24",
    "Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/535.24 (KHTML, like Gecko) Chrome/19.0.1055.1 Safari/535.24"
]

#请求网页的代码整合
def get_response(url):
    #random.choice从一个集合中随机选出请求头
    headers = {'user-agent':random.choice(USER_AGENT_LIST)}

    resp = requests.get(url,headers=headers)
    resp.raise_for_status()
    soup = BeautifulSoup(resp.content, 'lxml')
    return soup

#找到每本书的链接
def get_book_url(page):
    if page>10:
        return []
    num=(page-1)*25
    url = 'https://book.douban.com/top250?start=%s'%str(num)
    soup = get_response(url)
    book_div = soup.find('div', attrs={'class': 'indent'})
    books = book_div.find_all('tr', attrs={'class': 'item'})
    urls = [ book.td.a['href'] for book in books ]
    print('获取第%s页'%page,urls)
    return urls
```

#获得每本书的信息

```
def get_book_info(book_url):
    soup = get_response(book_url)

    div_info = soup.find('div',attrs={'id':'info'})

    book_author = div_info.a.text.split(' ')[-1]          #将空格去除
    book = soup.find('div',attrs={'class':'rating_wrap clearfix'})
    book_name= soup.find('span',attrs={'property':'v:itemreviewed'}).text

    book_grade = book.find('strong',attrs={'class':'ll rating_num '}).text
    book_man = book.find('a',attrs={'class':'rating_people'}).span.text
    book_info = {}

    book_info['name']=book_name
    book_info['author']=book_author
    book_info['rating_num'] = int(book_man)
    book_info['grade'] = float(book_grade)
    print(book_info)

    return book_info
```

```
if __name__ == '__main__':
    all_urls = []
    #从第1页到第10页爬取，链接拼接到一起。
    for page in range(1,11):
        urls = get_book_url(page)
        all_urls = all_urls+urls
    print('获取到的链接数:',len(all_urls))
    out=''
    for url in all_urls:
        try:
            info = get_book_info(url)
        except Exception as e:
            print(e)
            continue
        out=out+str(info)+'\n'
    with open('douban_book_top250.txt','w') as f:      #输出到TXT文件
        f.write(out)
```

```
{ 'name': '莲化', 'author': '安妮宝贝 (庆山)', 'rating_num': 77968, 'grade': 8.0}
{ 'name': '哈利·波特与火焰杯', 'author': 'J·K·罗琳', 'rating_num': 71663, 'grade': 9.0}
{ 'name': '边城', 'author': '沈从文', 'rating_num': 70873, 'grade': 8.6}
{ 'name': '月亮和六便士', 'author': '威廉·萨默塞特·毛姆', 'rating_num': 72474, 'grade': 9.0}
{ 'name': '向左走·向右走', 'author': '幾米', 'rating_num': 70966, 'grade': 8.4}
{ 'name': '活着', 'author': '余华', 'rating_num': 74124, 'grade': 9.3}
{ 'name': '穆斯林的葬礼', 'author': '霍达', 'rating_num': 71273, 'grade': 8.2}
{ 'name': '从你的全世界路过', 'author': '张嘉佳', 'rating_num': 79349, 'grade': 7.1}
{ 'name': '悲伤逆流成河', 'author': '郭敬明', 'rating_num': 85064, 'grade': 6.2}
{ 'name': '恶意', 'author': '东野圭吾', 'rating_num': 71198, 'grade': 8.4}
{ 'name': '天龙八部', 'author': '金庸', 'rating_num': 56809, 'grade': 9.1}
{ 'name': '放学后', 'author': '东野圭吾', 'rating_num': 76091, 'grade': 7.6}
```

公众号·猿榜编程



分析页面，找到目标元素所在的tag节点，记录其属性，通过beautifulsoup的find和find\_all找到该区域，然后进行相应的提取。  
这个过程中，要说明一下，使用单线程阻塞爬取250本书的信息，比较耗时。

