4- BeautifulSoup实践

1、项目：爬取豆瓣新片榜中电影的基本信息

1-1、确定目标

（1）目标网站：https://movie.douban.com/chart

（2）网站协议：https://movie.douban.com/robots.txt（目标网站 + robots.txt 可查看目标网站的页面爬取许可）；

（3）项目目标：爬取电影名、URL、电影基本信息和电影评分信息。

1-2、过程分析

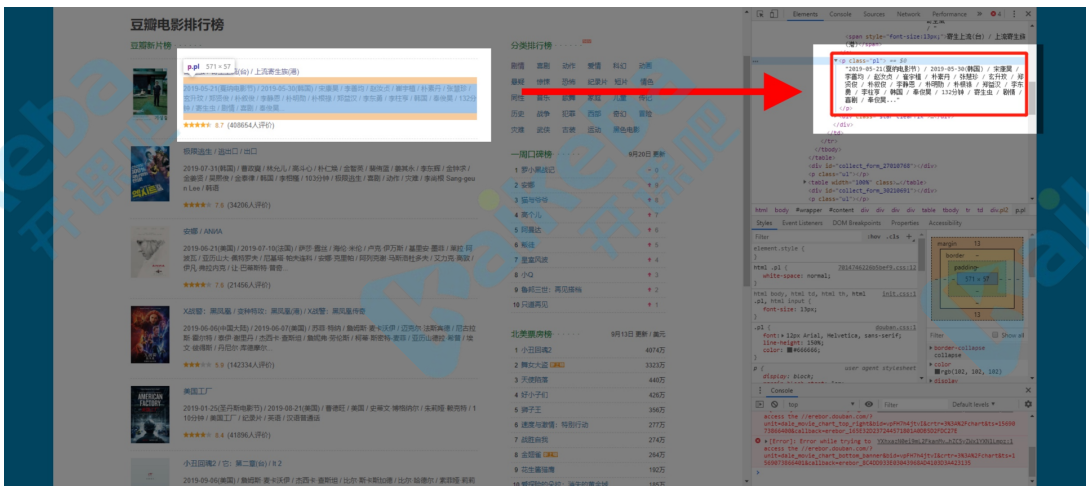
（1）确定数据位置

- 电影名、电影基本信息和电影评分信息详情页、URL均在 html 页面上；
- 获取数据用 requests.get() ；
- 解析数据用 BeautifulSoup。

（2）提取数据

- 【windows】：在网页的空白处点击右键，然后选择“检查”（快捷方式是ctrl+shift+i），再在 Elements 页面按 ctrl+f；【mac】：在网页的空白处点击右键，然后选择“检查”（快捷键 command + option + I(大写i)）；

- 点击【检查】页面左上角的“鼠标”按钮，再点击后右侧想要获取的内容可以定位到该内容对应的标签；
- 如此，我们就定位到了电影名的所在位置，a标签内的文本，甚至还顺带找到了详情页URL的所在位置。如上图，a标签里有属性href，其值是https://movie.douban.com/subject/27010768/。点击它，你会跳转到这部电影的详情页：
- 所以到时候，我们可以去提取a标签。接着，先用text拿到它的文本，再使用[href]获取到URL。



- 当我们的光标放在这个p标签上时，这个p标签里包含了寄生虫这部电影所有的基本信息，包含了上映时间、演员、导演等信息，即：2019-05-21(戛纳电影节) / 2019-05-30(韩国) / 宋康昊 / 李善均 / 赵汝贞 / 崔宇植 / 朴素丹 / 张慧珍 / 玄升玟 / 郑贤俊 / 朴叙俊 / 李静恩 / 朴明勋 / 朴根禄 / 郑益汉 / 李东勇 / 李柱亨 / 韩国 / 奉俊昊 / 132分钟 / 寄生虫 / 剧情 / 喜剧 / 奉俊昊...
- 这些都是p标签里的纯文本。这个p标签的属性是class=”pl”
- 根据电影名、URL、电影基本信息和电影评分信息的路径，我们可以知道这四者的最小共同父级标签是：div class=”pl2”。

1-3、代码实现（一）

1-3-1、数据获取

requests.get() 获取数据，BeautifulSoup 解析数据。

```
import requests
# 引用requests库
from bs4 import BeautifulSoup
# 引用BeautifulSoup库
headers = {
    'User-Agent': 'Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.79 Safari/537.36'
}
res_movies = requests.get('https://movie.douban.com/chart', headers=headers)
# 获取数据
bs_movies = BeautifulSoup(res_movies.text, 'html.parser')
# 解析数据
print(bs_movies)
# 打印解析结果
```

1-3-2、提取最小父级标签

- 电影名是a标签内的文本，URL是a标签里属性href的值，电影基本信息藏身于p class="pl"，电影评分信息藏身于div class="star clearfix"。最后，它们三者的最小共同父级标签，是div class="pl2"。
- 根据我们【过程分析】中所有菜谱的共同标签 class_='pl2'，我们用 find_all 获取所有菜谱（find_all 获取后返回的是一个列表），下面我们提取出第0个父级标签中的第0个<a>标签，并输出菜名和URL：
- 具体思路：

1、先爬取所有的最小父级标签div class="pl2"，然后针对每一个父级标签，想办法提取里面的电影名、URL、电影基本信息和电影的评价信息。

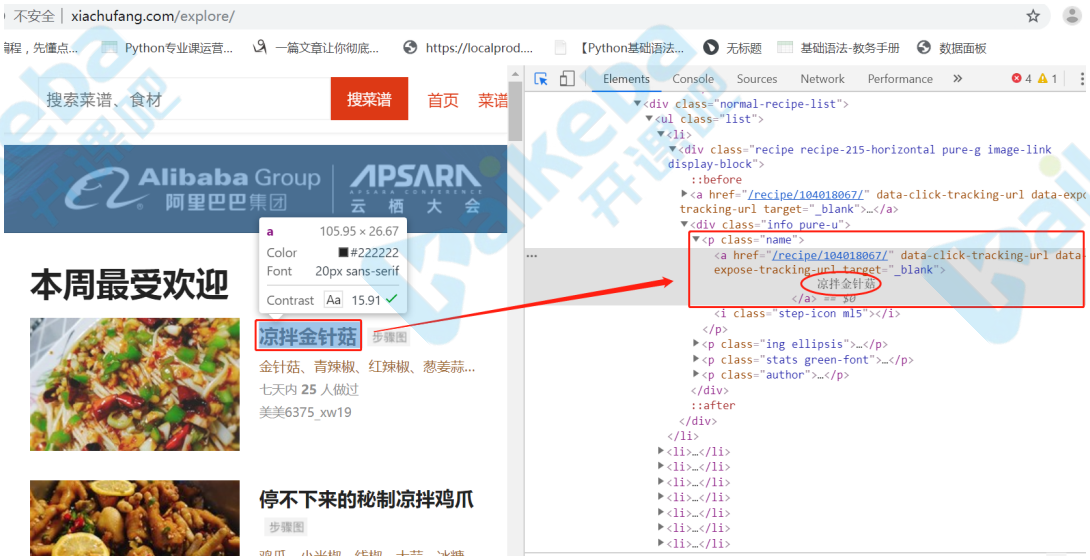
2、分别提取所有的电影名、URL、电影基本信息和电影的评价信息。然后让电影名、URL、电影基本信息和电影评分信息给一一对应起来（这并不复杂，第0个电影名，对应第0个URL，对应第0组电影基本信息，对应第0组电影评分信息，按顺序走即可）。

```
import requests
# 引用requests库
from bs4 import BeautifulSoup
# 引用BeautifulSoup库
headers = {
    'User-Agent': 'Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.79 Safari/537.36'
}
res_movies = requests.get('https://movie.douban.com/chart', headers=headers)
# 获取数据
bs_movies = BeautifulSoup(res_movies.text, 'html.parser')
# 解析数据
list_movies = bs_movies.find_all('div', class_='pl2')
# 查找最小父级标签
```

```
print(list_movies)
# 打印最小父级标签
```

1. 提取菜名

依旧是根据我们的内容定位我们的标签，可以找到菜名是在我们的标签 a 中，再用 text 取到该标签对应的菜名。




```
import requests
# 引用requests库
from bs4 import BeautifulSoup
# 引用BeautifulSoup库
res_foods = requests.get('http://www.xiachufang.com/explore/')
# 获取数据
bs_foods = BeautifulSoup(res_foods.text,'html.parser')
# 解析数据
list_foods = bs_foods.find_all('div',class_='info pure-u')
# 查找最小父级标签
tag_a = list_foods[0].find('a')
# 提取第0个父级标签中的<a>标签
print(tag_a.text[17:-13])
# 输出菜名，使用[17:-13]切掉了多余的信息
```

1. 提取 URL

我们发现在标签 a 后面的 href 有我们需要的链接，但是不完整，所以需要拼接后才能得到我们要的菜谱 URL。

```
import requests
# 引用requests库
from bs4 import BeautifulSoup
# 引用BeautifulSoup库
res_foods = requests.get('http://www.xiachufang.com/explore/')
# 获取数据
bs_foods = BeautifulSoup(res_foods.text,'html.parser')
# 解析数据
list_foods = bs_foods.find_all('div',class_='info pure-u')
```

```
# 查找最小父级标签
tag_a = list_foods[0].find('a')
# 提取第0个父级标签中的<a>标签
print('http://www.xiachufang.com'+tag_a['href'])
# 拼接后输出URL
```

1. 提取食材

我们可以看到我们的食材是在 p 中，但是只靠这个是不够的的，所以我们要精确取值，可以看到食材对应的 class 属性为 ing ellipsis。


```
import requests
# 引用requests库
from bs4 import BeautifulSoup
# 引用BeautifulSoup库
res_foods = requests.get('http://www.xiachufang.com/explore/')
# 获取数据
bs_foods = BeautifulSoup(res_foods.text,'html.parser')
# 解析数据
list_foods = bs_foods.find_all('div',class_='info pure-u')
# 查找最小父级标签
tag_p = list_foods[0].find('p',class_='ing ellipsis')
# 提取第0个父级标签中的<p>标签
ingredients = tag_p.text[1:-1]
# 食材，使用[1:-1]切掉了多余的信息
print(ingredients)
# 打印食材
```

1-3-3、写循环，存列表

```
import requests
# 引用requests库
from bs4 import BeautifulSoup
# 引用BeautifulSoup库
res_foods = requests.get('http://www.xiachufang.com/explore/')
# 获取数据
bs_foods = BeautifulSoup(res_foods.text,'html.parser')
# 解析数据
list_foods = bs_foods.find_all('div',class_='info pure-u')
```

```
# 查找最小父级标签
list_all = []
# 创建一个空列表，用于存储信息
for food in list_foods:
    tag_a = food.find('a')
    # 提取第0个父级标签中的<a>标签
    name = tag_a.text[17:-13]
    # 菜名，使用[17:-13]切掉了多余的信息
    URL = 'http://www.xiachufang.com'+tag_a['href']
    # 获取URL
    tag_p = food.find('p',class_='ing ellipsis')
    # 提取第0个父级标签中的<p>标签
    ingredients = tag_p.text[1:-1]
    # 食材，使用[1:-1]切掉了多余的信息
    list_all.append([name,URL,ingredients])
    # 将菜名、URL、食材，封装为列表，添加进list_all
print(list_all)
# 打印
```

1-4、代码实现（二）

创建一个空列表，启动循环，循环长度等于 <p> 标签的总数——我们可以借助 range(len()) 语法。

```
import requests
# 引用requests库
from bs4 import BeautifulSoup
# 引用BeautifulSoup库
res_foods = requests.get('http://www.xiachufang.com/explore/')
# 获取数据
bs_foods = BeautifulSoup(res_foods.text,'html.parser')
# 解析数据
tag_name = bs_foods.find_all('p',class_='name')
# 查找包含菜名和URL的<p>标签
tag_ingredients = bs_foods.find_all('p',class_='ing ellipsis')
# 查找包含食材的<p>标签
list_all = []
# 创建一个空列表，用于存储信息
for x in range(len(tag_name)):
# 启动一个循环，次数等于菜名的数量
    list_food = [tag_name[x].text[18:-14],tag_name[x].find('a')
['href'],tag_ingredients[x].text[1:-1]]
    # 提取信息，封装为列表。注意此处[18:-14]切片和之前不同，是因为此处使用的是<p>
标签，而之前是<a>
    list_all.append(list_food)
    # 将信息添加进list_all
print(list_all)
# 打印
```