爬虫实践---一次下完所有小说：排行榜小说批量下载

Josiah

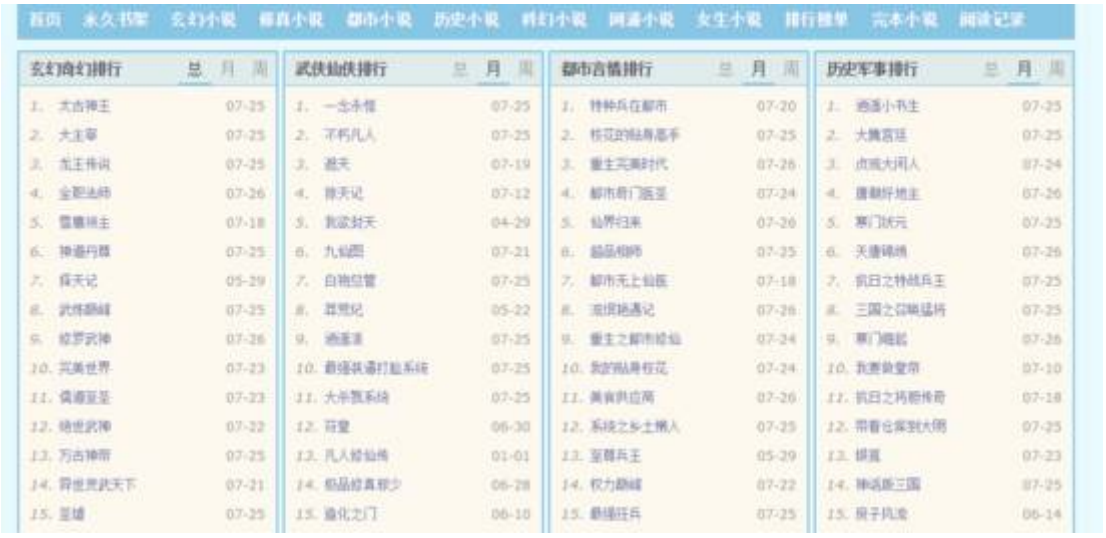
每日一个Linux、Python干货 ▲ 关注的人都加薪了

一、目标

排行榜的地址： http://www.qu.la/paihangbang/

找到各类排行旁的的每一部小说的名字，和在该网站的链接。

二、观察网页的结构



很容易就能发现，每一个分类都是包裹在：

```
<div class="index_toplist mright mbottom">
```

之中，

这种条理清晰的网站，大大方便了爬虫的编写。

在当前页面找到所有小说的连接，并保存在列表即可。

三、列表去重的小技巧：

就算是不同类别的小说，也是会重复出现在排行榜的。

这样无形之间就会浪费很多资源，尤其是在面对爬大量网页的时候。

这里只要一行代码就能解决：

```
url_list = list(set(url_list))
```

这里调用了一个list的构造函数set：这样就能保证列表里没有重复的元素了。

四、代码实现

模块化，函数式编程是一个非常好的习惯，坚持把每一个独立的功能都写成函数，这样会使代码简单又可复用。

1. **网页抓取头：**

```
import requests
from bs4 import BeautifulSoup

def get_html(url):
    try:
        r = requests.get(url,timeout=30)
        r.raise_for_status
        r.encoding='utf-8'
        return r.text
    except:
        return 'error!'
```

2. **获取排行榜小说及其链接：**

爬取每一类型小说排行榜，

按顺序写入文件，

文件内容为 小说名字+小说链接

将内容保存到列表

并且返回一个装满url链接的列表

```
def get_content(url):

    url_list = []
    html = get_html(url)
    soup = BeautifulSoup(html,'lxml')

    # 由于小说排版的原因，历史类和完本类小说不在一个div里
    category_list = soup.find_all('div',class_='index_toplist mright mbottom')
    history_list = soup.find_all('div',class_='index_toplist mbottom')

    for cate in category_list:
        name = cate.find('div',class_='toptab').span.text
        with open('novel_list.csv','a+') as f:
            f.write('\n小说种类：{} \n'.format(name))

        book_list = cate.find('div',class_='topbooks').find_all('li')

        # 循环遍历出每一个小说的的名字，以及链接
        for book in book_list:
            link = 'http://www.qu.la/' + book.a['href']
            title = book.a['title']
            url_list.append(link)

            # 这里使用a模式写入，防止清空文件
            with open('novel_list.csv','a') as f:
                f.write('小说名:{} \t 小说地址:{} \n'.format(title,link))

    for cate in history_list:
        name = cate.find('div',class_='toptab').span.text
        with open('novel_list.csv','a') as f:
            f.write('\n小说种类: {} \n'.format(name))

        book_list = cate.find('div',class_='topbooks').find_all('li')

        for book in book_list:
            link = 'http://www.qu.la/' + book.a['href']
            title = book.a['title']
            url_list.append(link)

            with open('novel_list.csv','a') as f:
                f.write('小说名:{} \t 小说地址:{} \n'.format(title,link))

    return url_list
```

3. 获取单本小说的所有章节链接：

获取该小说每个章节的url地址，并创建小说文件

```
# 获取单本小说的所有章节链接
def get_txt_url(url):

    url_list = []
    html = get_html(url)
    soup = BeautifulSoup(html,'lxml')
    list_a = soup.find_all('dd')
    txt_name = soup.find('dt').text

    with open('C:/Users/Administrator/Desktop/小说/{}.txt'.format(txt_name),'a+') as f:
        f.write('小说标题：{} \n'.format(txt_name))

    for url in list_a:
        url_list.append('http://www.qu.la/' + url.a['href'])

    return url_list,txt_name
```

4. 获取单页文章的内容并保存到本地

这里有个小技巧：

从网上爬下来的文件很多时候都是带着
之类的格式化标签，

可以通过一个简单的方法把它过滤掉：

```
html = get_html(url).replace('<br/>', '\n')
```

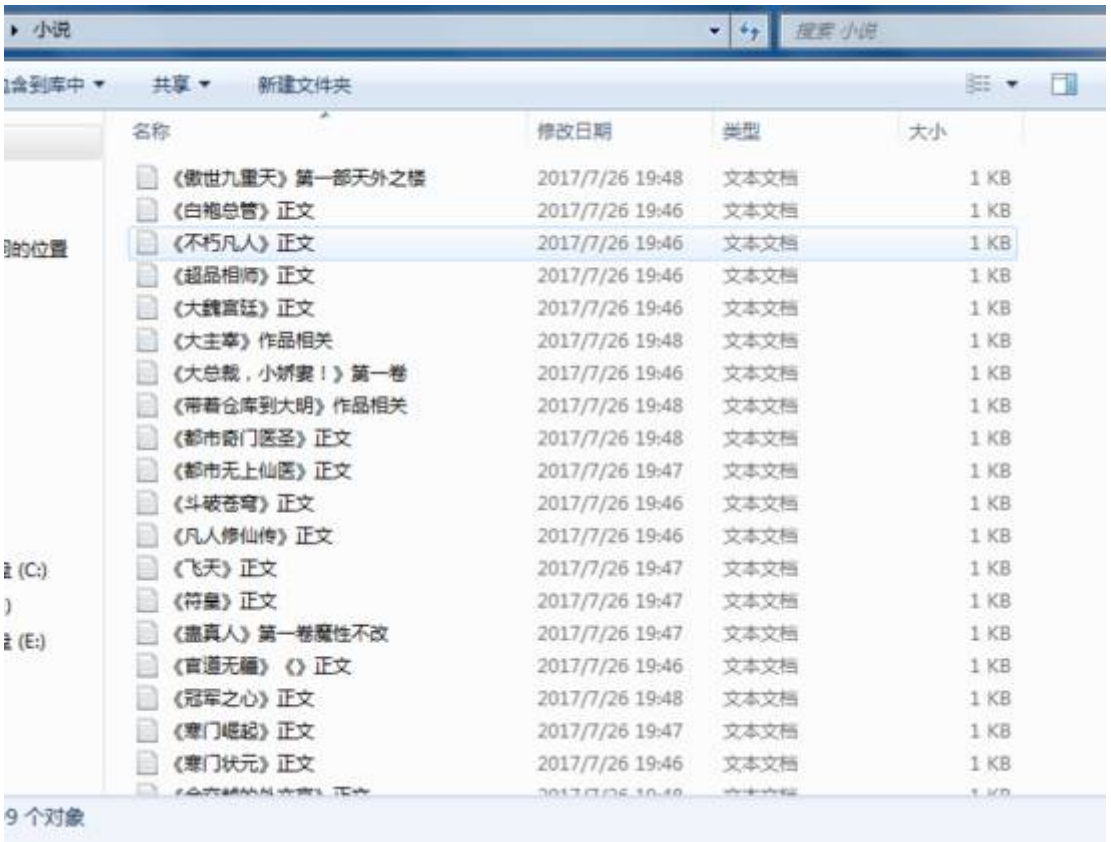
这里单单过滤了一种标签，并将其替换成‘\n’用于文章的换行，

```
def get_one_txt(url,txt_name):

    html = get_html(url).replace('<br/>','\n')
    soup = BeautifulSoup(html,'lxml')
    try:
        txt = soup.find('div',id='content').text
        title = soup.find('h1').text

        with open('C:/Users/Administrator/Desktop/小说/{}.txt'.format(txt.name),'a') as f:
            f.write(title + '\n\n')
            f.write(txt)
            print('当前小说：{}当前章节{}已经下载完毕'.format(txt_name,title))
    except:
        print('ERROR!')
```

6. 主函数

```
def get_all_txt(url_list):

    for url in url_list:
        # 遍历获取当前小说的所有章节的目录，并且生成小说头文件

        page_list,txt_name = get_txt_url(url)

def main():
    # 小说排行榜地址
    base_url = 'http://www.qu.la/paihangbang/'
    # 获取排行榜中所有小说的url链接
    url_list = get_content(base_url)
    # 除去重复的小说
    url_list = list(set(url_list))
    get_all_txt(url_list)

if __name__ == '__main__':
    main()
```

7. 输出结果



A1

	A	B	C	D	E	F	G	H
1								
2	小说种类：玄幻奇幻排行							
3	小说名:太古神王　小说地址:http://www.qu.la//book/4140/							
4	小说名:大主宰　小说地址:http://www.qu.la//book/176/							
5	小说名:龙王传说　小说地址:http://www.qu.la//book/13453/							
6	小说名:全职法师　小说地址:http://www.qu.la//book/4703/							
7	小说名:雪鹰领主　小说地址:http://www.qu.la//book/5094/							
8	小说名:神道丹尊　小说地址:http://www.qu.la//book/13781/							
9	小说名:择天记　小说地址:http://www.qu.la//book/168/							
10	小说名:武炼巅峰　小说地址:http://www.qu.la//book/68/							
11	小说名:修罗武神　小说地址:http://www.qu.la//book/175/							
12	小说名:完美世界　小说地址:http://www.qu.la//book/14/							
13	小说名:儒道至圣　小说地址:http://www.qu.la//book/903/							
14	小说名:绝世武神　小说地址:http://www.qu.la//book/322/							
15	小说名:万古神帝　小说地址:http://www.qu.la//book/14721/							
16	小说名:异世灵武天下　小说地址:http://www.qu.la//book/199/							
17	小说名:圣墟　小说地址:http://www.qu.la//book/24868/							
18								
19	小说种类：武侠仙侠排行							
20	小说名:一念永恒　小说地址:http://www.qu.la//book/16431/							
21	小说名:不朽凡人　小说地址:http://www.qu.la//book/18049/							
22	小说名:掠天记　小说地址:http://www.qu.la//book/4295/							
23	小说名:遮天　小说地址:http://www.qu.la//book/394/							
24	小说名:我欲封天　小说地址:http://www.qu.la//book/1/							
25	小说名:九仙图　小说地址:http://www.qu.la//book/11631/							
26	小说名:莽荒纪　小说地址:http://www.qu.la//book/76/							
27	小说名:最强装逼打脸系统　小说地址:http://www.qu.la//book/25052/							

5. 缺点：

本次爬虫写的这么顺利，更多的是因为爬的网站是没有反爬虫技术，以及文章分类清晰，结构优美。

但是，按照这篇文的思路去爬取小说，

大概计算了一下：

一篇文章需要：0.5s

一本小说（1000张左右）：8.5分钟

全部排行榜（60本）：　8.5小时！

那么，这种 单线程 的爬虫，速度如何能提高呢？

自己写个多线程模块？

其实还有更好的方式： Scrapy框架

后面可将这里的代码重构一边遍，

速度会几十倍甚至几百倍的提高了！

这其实也是多线程的威力！

作者：Josiah

来源：http://www.cnblogs.com/Josiah-Lin/p/7241678.html

*声明：推送内容及图片来源于网络，部分内容会有所改动，版权归原作者所有，如来源信息有误或侵犯权益，请联系我们删除或授权事宜。

- END -


长按识别二维码，即刻获取


每天精选技术干货，十万Linux人订阅
Linux人充电第一站
长按识别二维码 关注马哥Linux运维

[阅读原文]