

# Python爬虫实战之爬取百度贴吧帖子

大家好，上次我们实验了爬取了[糗事百科的段子](#)，那么这次我们来尝试一下爬取百度贴吧的帖子。与上一篇不同的是，这次我们需要用到文件的相关操作。

## 本篇目标

1. 对百度贴吧的任意帖子进行抓取
2. 指定是否只抓取楼主发帖内容
3. 将抓取到的内容分析并保存到文件

## 1. URL格式的确定

首先，我们先观察一下百度贴吧的任意一个帖子。

比如：[http://tieba.baidu.com/p/3138733512?see\\_lz=1&pn=1](http://tieba.baidu.com/p/3138733512?see_lz=1&pn=1)，这是一个关于NBA50大的盘点，分析一下这个地址。

`http://` 代表资源传输使用http协议

`tieba.baidu.com` 是百度的二级域名，指向百度贴吧的服务器。

`/p/3138733512` 是服务器某个资源，即这个帖子的地址定位符

`see_lz` 和 `pn` 是该URL的两个参数，分别代表了只看楼主和帖子页码，等于1表示该条件为真

所以我们可以把URL分为两部分，一部分为基础部分，一部分为参数部分。

例如，上面的URL我们划分基础部分是 <http://tieba.baidu.com/p/3138733512>，参数部分是 `?see_lz=1&pn=1`

## 2. 页面的抓取

熟悉了URL的格式，那就让我们用urllib2库来试着抓取页面内容吧。上一篇糗事百科我们最后改成了面向对象的编码方式，这次我们直接尝试一下，定义一个类名叫BDTB(百度贴吧)，一个初始化方法，一个获取页面的方法。

其中，有些帖子我们想指定给程序是否要只看楼主，所以我们把只看楼主的参数初始化放在类的初始化上，即init方法。另外，获取页面的方法我们需要知道一个参数就是帖子页码，所以这个参数的指定我们放在该方法中。

综上，我们初步构建出基础代码如下：

```

1  #-*-coding:utf8-*-
2  #created by 10412
3
4  import urllib
5  import urllib2
6  import re
7
8  #百度贴吧爬虫类
9  class BDTB:
10
11     #初始化,传入基地址,是否只看楼主的参数
12     def __init__(self, baseUrl, seeLZ):
13         self.baseUrl = baseUrl
14         self.seeLZ = '?see_lz=' + str(seeLZ)
15
16     #传入页码,获取该页帖子的代码
17     def getPage(self, pageNum):
18         try:
19             url = self.baseUrl + self.seeLZ + '&pn=' + str(pageNum)
20             request = urllib2.Request(url)
21             response = urllib2.urlopen(request)
22             print response.read()
23             return response
24         except urllib2.URLError, e:
25             if hasattr(e, "reason"):
26                 print u"连接百度贴吧失败,错误原因",e.reason
27                 return None
28
29     baseUrl = 'http://tieba.baidu.com/p/3138733512'
30     bdtb = BDTB(baseUrl, 1)
31     bdtb.getPage(1)

```

运行代码,我们可以看到屏幕上打印出了这个帖子第一页楼主发言的所有内容,形式为HTML代码。

```

</p>
</div>
</li></ul></div><div class="p_thread thread_theme_5" id="thread_theme_5"><div class="l_thread_info"><ul class="l_posts_num">
  <li class="l_pager pager_theme_4 pb_list_pager"><span class="tP">1</span>
  <a href="/p/3138733512?see_lz=1&pn=2">2</a>
  <a href="/p/3138733512?see_lz=1&pn=3">3</a>
  <a href="/p/3138733512?see_lz=1&pn=4">4</a>
  <a href="/p/3138733512?see_lz=1&pn=5">5</a>
  <a href="/p/3138733512?see_lz=1&pn=2">下一页</a>
  <a href="/p/3138733512?see_lz=1&pn=5">尾页</a>
</li>
<li class="l_reply_num" style="margin-left:8px"><span class="red" style="margin-right:3px">138</span>回复贴,共<span class="red">5</span>页</li>
<li class="l_reply_num">, 跳到 <input theme="4" id="jumpPage4" max-page="5" type="text" class="jump_input_bright" /> 页<input type="button" value="确定" />
</ul>
<div id="tofrs_up" class="tofrs_up"><a href="/f?kw=nb&ie=utf-8" title="nba">返回nba吧</a></div></div><div class="loading-tip" style="display:none;"><span class="text">>0< 加载中.
  <div class="louzhubiaoshi j_louzhubiaoshi" author="明之翼">
    <a href="/p/3138733512?pid=53018668923&see_lz=1#53018668923"></a>
  </div>
</div>
<div>
  <ul class="p_author">
    <li class="icon">
      <div class="icon_relative j_user_card" data-field="{"&quot;un&quot;:&quot;,&quot;u660e&quot;:&quot;u4e4b&quot;,&quot;u7ffc&quot;:"}>

```

### 3. 提取相关信息

## 1)提取帖子标题

在浏览器中审查元素，或者按F12，查看页面源代码，我们找到标题所在的代码段如下：

```
1 <h3 class="core_title_txt pull-left text-overflow " title="纯原创我心中的NBA2014-2015赛季现役50大" style="width: 416px">纯原创我心中的NBA2014-2015赛季现役50大</h3>
```

所以我们要提取 `<h3>` 中的内容，因为一开始可以查看整个界面的原代码，查看里面含有 `<h3>` 标签的不止一个。所以需要写正则表达式来匹配，如下：

```
1 <h3 class="core_title_txt.*?>(.*?)</h3>
```

然后，我们可以写个获取标题的方法

```
1 # 获取帖子标题
2 def getTitle(self):
3     page = self.getPage(1)
4     pattern = re.compile('<h3 class="core_title_txt.*?>(.*?)</h3>', re.S)
5     result = re.search(pattern, page)
6     if result:
7         # print result.group(1) #测试输出
8         return result.group(1).strip()
9     else:
10        return None
```

## 2) 提取帖子页数

同样地，帖子总页数我们也可以通过分析页面中的共?页来获取。

```
1 <li class="l_reply_num" style="margin-left:8px"><span class="red" style="margin-right:3px">4784</span>回复贴，共<span class="red">36</span>页</li>
```

所以我们的获取总页数的方法如下

```
1 #获取帖子一共有多少页
2 def getPageNum(self):
3     page = self.getPage(1)
4     pattern = re.compile('<li class="l_reply_num.*?</span>.*?<span.*?>(.*?)</span>', re.S)
5     result = re.search(pattern,page)
6     if result:
7         #print result.group(1) #测试输出
8         return result.group(1).strip()
9     else:
10        return None
```

## 3) 提取正文内容

审查元素，可以看到百度贴吧每一层楼的主要内容都在

标签里面，所以我们可以写如下的正则表达式

```
1 <div id="post_content_.*?>(.*?)</div>
```

所以提取正文内容的方法：

```
1 #获取每一层楼的内容,传入页面内容
2 def getContent(self,page):
3     pattern = re.compile('<div id="post_content_.*?>(.*?)</div>',re.S)
4     items = re.findall(pattern,page)
5     for item in items:
6         print item
```

运行截图如下：

很多媒体都在每赛季之前给球员排个名，我也有这个癖好……，我会尽量理性的分析球队地位，<a href="http://www.baidu.com/s?wd=个人能力&ie=gbk&tn=SE\_hldp00990\_u6vqbz10" cl



今天再更新一个就回家啦



搞定今天更新完毕 准备下班了

晚上偷偷顶一个

开更今天的了！









顶一项，难道写着写着成单机了？

吃饭去了 下午来更吧

下午回来继续更新了



上个厕所再回来更新下一个



再顶顶



再顶一项，写得好辛苦，别又变成单机了

话说现在除了一个人都没有人来说高了，低了呢，好奇怪



可以看到有很多的换行符和图片符，既然出现这样的情况，那肯定不是我们想要的结果。那我们就必须要将文本进行处理，将各种复杂的标签给剔除，还原帖子的原来面貌。可以使用一个方法或者类将这个处理文本的实现，不过为了更好的代码重用和架构，还是建议使用一个类。

我们将这个类命名为 `Too`（工具类），里面定义一个 `replace` 方法，替换各种标签。然后在类中定义几个正则表达式，利用 `re.sub` 方法对文本进行匹配后然后替换。

```

1 import re
2
3 #处理页面标签类
4 class Tool:
5     #去除img标签,7位长空格
6     removeImg = re.compile('<img.*?>| {7}|')
7     #删除超链接标签
8     removeAddr = re.compile('<a.*?>|</a>')
9     #把换行的标签换为\n
10    replaceLine = re.compile('<tr>|<div>|</div>|</p>')
11    #将表格制表<td>替换为\t
12    replaceTD= re.compile('<td>')
13    #把段落开头换为\n加空两格
14    replacePara = re.compile('<p.*?>')
15    #将换行符或双换行符替换为\n
16    replaceBR = re.compile('<br><br>|<br>')
17    #将其余标签剔除
18    removeExtraTag = re.compile('<.*?>')
19    def replace(self,x):
20        x = re.sub(self.removeImg,"",x)
21        x = re.sub(self.removeAddr,"",x)
22        x = re.sub(self.replaceLine,"\n",x)
23        x = re.sub(self.replaceTD,"\t",x)
24        x = re.sub(self.replacePara,"\n    ",x)
25        x = re.sub(self.replaceBR,"\n",x)
26        x = re.sub(self.removeExtraTag,"",x)
27        #strip()将前后多余内容删除
28        return x.strip()

```

在使用时，我们只需要初始化一下这个类，然后调用replace方法即可。

现在整体代码是如下这样子的，现在我的代码是写到这样子的：

```

1  -*-coding:utf8 -*-
2  #created by 10412
3
4  import urllib
5  import urllib2
6  import re
7
8
9  # 处理页面标签类
10 class Tool:
11     # 去除img标签,7位长空格
12     removeImg = re.compile('<img.*?>| {7}|')
13     # 删除超链接标签
14     removeAddr = re.compile('<a.*?>|</a>')
15     # 把换行的标签换为\n
16     replaceLine = re.compile('<tr>|<div>|</div>|</p>')
17     # 将表格制表<td>替换为\t
18     replaceTD = re.compile('<td>')
19     # 把段落开头换为\n加空两格
20     replacePara = re.compile('<p.*?>')
21     # 将换行符或双换行符替换为\n
22     replaceBR = re.compile('<br><br>|<br>')
23     # 将其余标签剔除
24     removeExtraTag = re.compile('<.*?>')
25
26     def replace(self, x):
27         x = re.sub(self.removeImg, "", x)
28         x = re.sub(self.removeAddr, "", x)
29         x = re.sub(self.replaceLine, "\n", x)
30         x = re.sub(self.replaceTD, "\t", x)
31         x = re.sub(self.replacePara, "\n    ", x)
32         x = re.sub(self.replaceBR, "\n", x)
33         x = re.sub(self.removeExtraTag, "", x)
34         # strip()将前后多余内容删除
35         return x.strip()
36
37
38 # 百度贴吧爬虫类
39 class BDTB:
40     # 初始化,传入基地址,是否只看楼主的参数
41     def __init__(self, baseUrl, seeLZ):
42         self.baseUrl = baseUrl
43         self.seeLZ = '?see_lz=' + str(seeLZ)
44         self.tool = Tool()
45
46     # 传入页码,获取该页帖子的代码
47     def getPage(self, pageNum):
48         try:
49             url = self.baseUrl + self.seeLZ + '&pn=' + str(pageNum)
50             request = urllib2.Request(url)
51             response = urllib2.urlopen(request)
52             return response.read().decode('utf-8')
53
54         except urllib2.URLError, e:

```

```

54         if hasattr(e, "reason"):
55             print u"连接百度贴吧失败, 错误原因", e.reason
56             return None
57
58     # 获取帖子标题
59     def getTitle(self):
60         page = self.getPage(1)
61         pattern = re.compile('<h1 class="core_title_txt.*?>(.*?)</h1>', re.S)
62         result = re.search(pattern, page)
63         if result:
64             # print result.group(1) #测试输出
65             return result.group(1).strip()
66         else:
67             return None
68
69     # 获取帖子一共有多少页
70     def getPageNum(self):
71         page = self.getPage(1)
72         pattern = re.compile('<li class="l_reply_num.*?</span>.*?<span.*?>(.*?)</span>',
73 re.S)
74         result = re.search(pattern, page)
75         if result:
76             # print result.group(1) #测试输出
77             return result.group(1).strip()
78         else:
79             return None
80
81     # 获取每一层楼的内容,传入页面内容
82     def getContent(self, page):
83         pattern = re.compile('<div id="post_content_.*?>(.*?)</div>', re.S)
84         items = re.findall(pattern, page)
85         # for item in items:
86         #     print item
87         print self.tool.replace(items[1])
88
89     baseURL = 'http://tieba.baidu.com/p/3138733512'
90     bdtb = BDTB(baseURL, 1)
91     bdtb.getContent(bdtb.getPage(1))

```

运行截图如下:

50 惊喜新人王 迈卡威  
上赛季数据

篮板 6.2 助攻 6.3 抢断 1.9 盖帽 0.6 失误 3.5 犯规 3 得分 16.7

新赛季第50位, 我给上赛季的新人王迈卡威。上赛季迈卡威在彻底重建的76人中迅速掌握了球队, 一开始就三双搞定了热火赢得了万千眼球。后来也屡屡有经验的表现, 新秀赛季就拿过三双的球员不多, 作为上赛季弱队的老大, 迈卡威刷出了不错的数据, 但我们静下心来看一看他, 还是发现他有很多问题。首先, 投篮偏弱刚刚40%的命中率和惨淡的26%的三分命中率肯定是不合格的! 加之身体瘦弱, 个字薄说缺点, 来说说优点, 作为后卫篮板球非常突出, 高大的身形能较好的影响对方的出手, 也能发现己方的空位球员。突破虽然速度一般, 但节奏感不错, 大局观也在平均水准之上。提醒瘦而高大, 不会耽误球队地位而言, 迈卡威现在是绝对的老大, 球你想怎么玩就怎么玩, 数据你想怎么刷就怎么刷! 去年的潜力新人诺尔是蓝领, 其他人都可以清退, 恩比德还受伤不能打, 76人队的战绩怎么样, 就看你了!

Process finished with exit code 0

## 4) 替换楼层

至于这个问题，我感觉直接提取楼层没什么必要呀，因为只看楼主的话，有些楼层的编号是间隔的，所以我们得到的楼层序号是不连续的，这样我们保存下来也没什么用。

所以可以尝试下面的方法：

- 1.每打印输出一段楼层，写入一行横线来间隔，或者换行符也好。
- 2.试着重新编一个楼层，按照顺序，设置一个变量，每打印出一个结果变量加一，打印出这个变量当做楼层。

将 `getContent` 方法修改如下：

```
1 #获取每一层楼的内容,传入页面内容
2 def getContent(self,page):
3     pattern = re.compile('<div id="post_content_.*?>(.*?)</div>',re.S)
4     items = re.findall(pattern,page)
5     floor = 1
6     for item in items:
7         print floor,u"楼-----
8         -----\n"
9         print self.tool.replace(item)
10        floor += 1
```

运行结果截图如下：

```
21 楼-----

41 麦迪35秒13分? 我有28秒11分的  米尔萨普
上赛季数据
篮板8.5 助攻3.1 抢断1.7 盖帽1.1 失误2.5 犯规2.8 得分17.9

大米胖上赛季继续着自己低调而发光发热的表现，老鹰用他取代史密斯是太聪明的决定了，米胖除了快攻的时候跑得没史密斯快，几乎是完爆史密斯，入选全明星是对他最大的肯定！
米胖的进攻手段很丰富，虽然跑得不快，弹跳一般，臂展也不行，但咱们工人有力量！米胖力量很好，打球速率很快！（这里解释一下，熟练是变幻动作的速度，米胖的第一步转身的那一步非常迅速，可以防守端，米胖受制于身体条件，对高举高打的打法没有办法。但是米胖作为内线场均1.7的抢断显示着防守端下三路很优秀。防挡拆不如前任史密斯，但也不算太坏（起码比布鲁克好多了）。虽说是个攻击缺点，米胖天赋是硬伤，作为一个二轮末尾的球员，打成现在这个样子已经是他的上限了，对比易建联，他靠着自身的努力和锻炼打入了全明星，但是可惜，下赛季霍福德归来依然是老鹰老大，米尔萨普
22 楼-----

再顶一顶，写得辛苦，别又变成单机了
23 楼-----

话说现在除了一个人都没有人来说高了，低了呢，好奇怪
24 楼-----

篮下终结机  小乔丹
上赛季数据
篮板13.6 助攻0.9 抢断1 盖帽2.5 失误1.5 犯规3.2 得分10.4
在里弗斯到了快船以后，小乔丹获得了新生，从一个无脑弹簧人+暴扣男，变成了一个防守覆盖整个两分区域，进攻终结能力犀爆的真男人。他现在是名符其实的快船第三巨头
上赛季小乔丹两项数据冠绝全联盟，1是场均13.6的篮板球，2是恐怖的67.6%的命中率，你说投得少命中率就高就不说了，但这家伙本赛季得分创了生涯新高，第一次上双，命中率还越来越高（庄神哭晕在防守端的小乔丹就更恐怖了，光看数据就知道，篮板王+盖帽第三。还有些数据无法体现的，比如脚步灵活身高臂长，防守挡拆效果很好。弹速很快，协防潜力很大，经常飞人的冒，加之篮球智商的进步还是说说缺点，首先那可怜的罚球命中率我就不提了，进攻端还是完全没有自主进攻的能力，作为高大中锋抢下后场篮板后第一时间一传能力也不强（好吧我要求过高了）。防守端太喜欢协防，有时候会漏总的来说，小乔丹用自身的表现来回报快船给他的大合同，快船三叉戟有望在未来1-2年内打出自己最佳的表现。
25 楼-----
```

#### 4. 写入文件

代码：

```
1 file = open("tb.txt","w")
2 file.writelines(obj)
```

#### 5. 完善代码



```

1  #-*-coding:utf8-*-
2  #created by 10412
3
4
5  import urllib
6  import urllib2
7  import re
8
9  #处理页面标签类
10 class Tool:
11     #去除img标签,7位长空格
12     removeImg = re.compile('<img.*?>| {7}|')
13     #删除超链接标签
14     removeAddr = re.compile('<a.*?>|</a>')
15     #把换行的标签换为\n
16     replaceLine = re.compile('<tr>|<div>|</div>|</p>')
17     #将表格制表<td>替换为\t
18     replaceTD = re.compile('<td>')
19     #把段落开头换为\n加空两格
20     replacePara = re.compile('<p.*?>')
21     #将换行符或双换行符替换为\n
22     replaceBR = re.compile('<br><br>|<br>')
23     #将其余标签剔除
24     removeExtraTag = re.compile('<.*?>')
25     def replace(self,x):
26         x = re.sub(self.removeImg,"",x)
27         x = re.sub(self.removeAddr,"",x)
28         x = re.sub(self.replaceLine,"\n",x)
29         x = re.sub(self.replaceTD,"\t",x)
30         x = re.sub(self.replacePara,"\n    ",x)
31         x = re.sub(self.replaceBR,"\n",x)
32         x = re.sub(self.removeExtraTag,"",x)
33         #strip()将前后多余内容删除
34         return x.strip()
35
36
37 #百度贴吧爬虫类
38 class BDTB:
39
40     #初始化,传入基地址,是否只看楼主的参数
41     def __init__(self,baseUrl,seeLZ,floorTag):
42         #base链接地址
43         self.baseUrl = baseUrl
44         #是否只看楼主
45         self.seeLZ = '?see_lz='+str(seeLZ)
46         #HTML标签剔除工具类对象
47         self.tool = Tool()
48         #全局file变量,文件写入操作对象
49         self.file = None
50         #楼层标号,初始为1
51         self.floor = 1
52         #默认的标题,如果没有成功获取到标题的话则会用这个标题
53         self.defaultTitle = u"百度贴吧"

```

```

54         #是否写入楼分隔符的标记
55         self.floorTag = floorTag
56
57     #传入页码, 获取该页帖子的代码
58     def getPage(self, pageNum):
59         try:
60             #构建URL
61             url = self.baseURL + self.seelZ + '&pn=' + str(pageNum)
62             request = urllib2.Request(url)
63             response = urllib2.urlopen(request)
64             #返回UTF-8格式编码内容
65             return response.read().decode('utf-8')
66         #无法连接, 报错
67         except urllib2.URLError, e:
68             if hasattr(e, "reason"):
69                 print u"连接百度贴吧失败, 错误原因", e.reason
70                 return None
71
72     #获取帖子标题
73     def getTitle(self, page):
74         #得到标题的正则表达式
75         pattern = re.compile('<h1 class="core_title_txt.*?>(.*?)</h1>', re.S)
76         result = re.search(pattern, page)
77         if result:
78             #如果存在, 则返回标题
79             return result.group(1).strip()
80         else:
81             return None
82
83     #获取帖子一共有多少页
84     def getPageNum(self, page):
85         #获取帖子页数的正则表达式
86         pattern = re.compile('<li class="l_reply_num.*?</span>.*?<span.*?>(.*?)</span>', re.S)
87         result = re.search(pattern, page)
88         if result:
89             return result.group(1).strip()
90         else:
91             return None
92
93     #获取每一层楼的内容, 传入页面内容
94     def getContent(self, page):
95         #匹配所有楼层的内容
96         pattern = re.compile('<div id="post_content_.*?>(.*?)</div>', re.S)
97         items = re.findall(pattern, page)
98         contents = []
99         for item in items:
100             #将文本进行去除标签处理, 同时在前后加入换行符
101             content = "\n" + self.tool.replace(item) + "\n"
102             contents.append(content.encode('utf-8'))
103         return contents
104
105     def setFileTitle(self, title):

```

```

106         #如果标题不是为None, 即成功获取到标题
107         if title is not None:
108             self.file = open(title + ".txt", "w+")
109         else:
110             self.file = open(self.defaultTitle + ".txt", "w+")
111
112     def writeData(self, contents):
113         #向文件写入每一楼的信息
114         for item in contents:
115             if self.floorTag == '1':
116                 #楼之间的分隔符
117                 floorLine = "\n" + str(self.floor) + u"-----\n"
118
119                 self.file.write(floorLine)
120                 self.file.write(item)
121                 self.floor += 1
122
123     def start(self):
124         indexPage = self.getPage(1)
125         pageNum = self.getPageNum(indexPage)
126         title = self.getTitle(indexPage)
127         self.setFileTitle(title)
128         if pageNum == None:
129             print "URL已失效, 请重试"
130             return
131         try:
132             print "该帖子共有" + str(pageNum) + "页"
133             for i in range(1, int(pageNum)+1):
134                 print "正在写入第" + str(i) + "页数据"
135                 page = self.getPage(i)
136                 contents = self.getContent(page)
137                 self.writeData(contents)
138             #出现写入异常
139             except IOError, e:
140                 print "写入异常, 原因" + e.message
141             finally:
142                 print "写入任务完成"
143
144
145     print u"请输入帖子代号"
146     baseURL = 'http://tieba.baidu.com/p/' + str(raw_input(u'http://tieba.baidu.com/p/'))
147     seeLZ = raw_input("是否只获取楼主发言, 是输入1, 否输入0\n")
148     floorTag = raw_input("是否写入楼层信息, 是输入1, 否输入0\n")
149     bdtb = BDTB(baseURL, seeLZ, floorTag)
150     bdtb.start()

```

运行后截图如下:

请输入帖子代号

<http://tieba.baidu.com/p/3513281888>

是否只获取楼主发言，是输入1，否输入0

1

是否写入楼层信息，是输入1，否输入0

1

该帖子共有7页

正在写入第1页数据

正在写入第2页数据

正在写入第3页数据

正在写入第4页数据

正在写入第5页数据

正在写入第6页数据

正在写入第7页数据

写入任务完成