

# [python] 常用正则表达式爬取网页信息及分析HTML标签总结

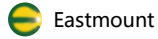
原创 Eastmount 最后发布于2016-04-07 06:13:37 阅读数 77205 ☆ 收藏

编辑 展开



## Python+TensorFlow人工智能

该专栏为人工智能入门专栏，采用Python3和TensorFlow实现人工智能相关算法。前期介绍安装流程、基础语法...



¥9.90

去订阅

这篇文章主要是介绍Python爬取网页信息时，经常使用的正则表达式及方法。它是一篇总结性文章，实用性比较大，主要解决自己遇到的爬虫问题，也希望对你有有所帮助~

当然如果会Selenium基于自动化测试爬虫、BeautifulSoup分析网页DOM节点，这就更方便了，但本文更多的是介绍基于正则的底层爬取分析。

涉及内容如下：

- 常用正则表达式爬取网页信息及HTML分析总结
  - 1.获取<tr></tr>标签之间内容
  - 2.获取<a href..></a>超链接之间内容
  - 3.获取URL最后一个参数命名图片或传递参数
  - 4.爬取网页中所有URL链接
  - 5.爬取网页标题title两种方法
  - 6.定位table位置并爬取属性-属性值
  - 7.过滤<span></span>等标签
  - 8.获取<script></script>等标签内容
  - 9.通过replace函数过滤<br />标签
  - 10.获取<img ../>中超链接及过滤<img>标签

推荐基础文章：[Python正则表达式指南 - AstralWind](#)

---

## 1.获取<tr></tr>标签之间内容

该部分主要是通过正则表达式获取两个标签之间的内容，通常这种标签都是成对出现的。

开始标签如：<tr>、<th>、<td>、<a>、<table>、<div>...

后缀标签如：</tr>、</th>、</td>、</a>、</table>、</div>...

核心代码：

```
res_tr = r'<tr>(.*?)</tr>'
```

```
m_tr = re.findall(res_tr,language,re.S|re.M)
```

例子：

```
# coding=utf-8
import re

language = '''<tr><th>性别: </th><td>男</td></tr><tr>'''

# 正则表达式获取<tr></tr>之间内容
res_tr = r'<tr>(.*?)</tr>'
m_tr = re.findall(res_tr, language, re.S|re.M)
for line in m_tr:
    print line
    # 获取表格第一列th 属性
    res_th = r'<th>(.*?)</th>'
    m_th = re.findall(res_th, line, re.S|re.M)
    for mm in m_th:
        print unicode(mm, 'utf-8'), #unicode防止乱
    # 获取表格第二列td 属性值
    res_td = r'<td>(.*?)</td>'
    m_td = re.findall(res_td, line, re.S|re.M)
    for nn in m_td:
        print unicode(nn, 'utf-8')
```

输出如下所示:

```
>>>
<th>性别: </th><td>男</td>
性别: 男
>>>
```

python通过re模块提供对正则表达式的支持。使用re的一般步骤是先将正则表达式的字符串形式编译为Pattern实例，然后使用Pattern实例处理文本并获得匹配结果（一个Match实例），最后使用Match实例获得信息，进行其他的操作。

findall(string[, pos[, endpos]]) | re.findall(pattern, string[, flags]): 搜索string，以列表形式返回全部能匹配的子串。其中RE的常见参数包括:

- re.I(re.IGNORECASE): 忽略大小写（括号内是完整写法）
- re.M(re.MULTILINE): 多行模式，改变'^'和'\$'的行为
- re.S(re.DOTALL): 点任意匹配模式，改变'.'的行为

## 2. 获取超链接<a href=...></a>之间内容

通常在使用正则表达式时，需要分析网页链接，获取URL或网页内容。核心代码如下:

```
res = r'<a .*?>(.*?)</a>'
mm = re.findall(res, content, re.S|re.M)
urls=re.findall(r"<a.*?href=.*?<\a>", content, re.I|re.S|re.M)
```

例子:

```
# coding=utf-8
```

```

import re

content = '''
<td>
<a href="https://www.baidu.com/articles/zj.html" title="浙江省">浙江省主题介绍</a>
<a href="https://www.baidu.com//articles/gz.html" title="贵州省">贵州省主题介绍</a>
</td>
'''

#获取<a href></a>之间的内容
print u'获取链接文本内容:'
res = r'<a .*?>(.*?)</a>'
mm = re.findall(
    res, content, re.S|re.M)
for value in mm:
    print value

#获取所有<a href></a>链接所有内容
print u'\n获取完整链接内容:'
urls=re.findall(r"<a.*?href=.*?</a>", content, re.I|re.S|re.M)
for i in urls:
    print i

#获取<a href></a>中的URL
print u'\n获取链接中URL:'
res_url = r"(?<=href=\\").+?(?=\\")|(?<=href=\\').+?(?=\\')"
link = re.findall(res_url , content, re.I|re.S|re.M)
for url in link:
    print url

```

输出如下图所示：

```

>>>
获取链接文本内容：
浙江省主题介绍
贵州省主题介绍

获取完整链接内容：
<a href="https://www.baidu.com/articles/zj.html" title="浙江省">浙江省主题介绍</a>
<a href="https://www.baidu.com//articles/gz.html" title="贵州省">贵州省主题介绍</a>

获取链接中URL：
https://www.baidu.com/articles/zj.html
https://www.baidu.com//articles/gz.html
>>>

```

当然如果是通过Selenium分析DOM树结构获取href对应的url或title中的值，其核心代码如下所示，这里主要是给大家做个对比，理解不同方法的优势：

```

driver.get(link)
elem = driver.find_elements_by_xpath("//div[@class='piclist']/tr/dd[1]")
for url in elem:
    pic_url = url.get_attribute("href")
    print pic_url

```

参考文章：[\[python爬虫\] Selenium定向爬取虎扑篮球海量精美图片](#)

---

### 3. 获取URL最后一个参数命名图片或传递参数

通常在使用Python爬取图片过程中，会遇到图片对应的URL最后一个字段通常用于命名图片，如虎扑孙悦妻子图片：

[http://i1.hoopchina.com.cn/blogfile/201411/11/BbsImg141568417848931\\_640\\*640.jpg](http://i1.hoopchina.com.cn/blogfile/201411/11/BbsImg141568417848931_640*640.jpg)

此时需要通过该URL的"/"后面的参数命名图片，则方法如下：

```
urls = "http://i1.hoopchina.com.cn/blogfile/201411/11/BbsImg141568417848931_640*640.jpg"
values = urls.split('/')[-1]
print values
```

输出如下所示：

```
>>>
BbsImg141568417848931_640*640.jpg
>>>
```

在使用Python获取GET方法的URL链接中，还可能存在传递参数的值。

此时获取参数方法如下：

```
url = 'http://localhost/test.py?a=hello&b=world'
values = url.split('?')[1]
print values
for key_value in values.split('&'):
    print key_value.split('=')
```

输出如下所示：

```
>>>
a=hello&b=world
['a', 'hello']
['b', 'world']
>>>
```

---

### 4. 爬取网页中所有URL链接

在学习爬虫过程中，你肯定需要从固有网页中爬取URL链接，再进行下一步的循环爬取或URL抓取。如下，爬取CSDN首页的所有URL链接。

```
# coding=utf-8
import re
import urllib

url = "http://www.csdn.net/"
```

```

content = urllib.urlopen(url).read()
urls = re.findall(r"<a.*?href=.*?</a>", content, re.I)
for url in urls:
    print unicode(url, 'utf-8')

link_list = re.findall(r"(<=href=\\\".+?(?=\\\")|(<=href=\\\".+?(?=\\')\"), content)
for url in link_list:
    print url

```

输出如下所示:

```

>>>
<a href="https://passport.csdn.net/account/login?from=http://my.csdn.net/my/mycsdn">登录</a>
<a href="http://passport.csdn.net/account/mobileregister?action=mobileRegister">注册</a>
<a href="https://passport.csdn.net/help/faq">帮助</a>
...
https://passport.csdn.net/account/login?from=http://my.csdn.net/my/mycsdn
http://passport.csdn.net/account/mobileregister?action=mobileRegister
https://passport.csdn.net/help/faq
...
>>>

```

## 5.爬取网页标题title两种方法

获取网页标题也是一种常见的爬虫，如我在爬取维基百科国家信息时，就需要爬取网页title。通常位于<html><head><title>标题</title></head></html>中。

下面是爬取CSDN标题的两种方法介绍：

```

# coding=utf-8
import re
import urllib

url = "http://www.csdn.net/"
content = urllib.urlopen(url).read()

print u'方法一:'
title_pat = r'(<=<title>).*?(?=</title>)'
title_ex = re.compile(title_pat, re.M|re.S)
title_obj = re.search(title_ex, content)
title = title_obj.group()
print title

print u'方法二:'
title = re.findall(r'<title>(.*?)</title>', content)
print title[0]

```

输出如下所示:

```

>>>
方法一:
CSDN.NET - 全球最大中文IT社区，为IT专业技术人员提供最全面的信息传播和服务平台
方法二:
CSDN.NET - 全球最大中文IT社区，为IT专业技术人员提供最全面的信息传播和服务平台

```

>>>

## 6.定位table位置并爬取属性-属性值

如果使用Python库的一些爬取，通常可以通过DOM树结构进行定位，如代码：

```
login = driver.find_element_by_xpath("//form[@id='loginForm']")
```

参考文章：[\[Python爬虫\] Selenium实现自动登录163邮箱和Locating Elements介绍](#)

但如果是正则表达式这种相对传统傻瓜式的方法，通过通过find函数寻找指定table方法进行定位。如：获取Infobox的table信息。通过分析源代码发现“程序设计语言列表”的消息盒如下：

```
<table class="infobox vevent" ..><tr><th></th><td></td></tr></table>
```

```
1 | start = content.find(r'<table class="infobox vevent"') #起点记录查询位置
2 | end = content.find(r'</table>')
3 | infobox = language[start:end]
4 | print infobox
```

print infobox 输出其中一门语言ActionScript的InfoBox消息盒部分源代码如下：

```
1 |
  <table class="infobox vevent" cellspacing="3" style="border-spacing:3px;width:22em;text-align:left;font-
  size:small;line-height:1.5em;">
2 | <caption class="summary"><b>ActionScript</b></caption>      3 | <tr>      4 |
  <th scope="row" style="text-align:left;white-space:nowrap;;">发行时间</th>      5 | <td style=";;">1998年</td>
6 | </tr>
7 | <tr>
8 | <th scope="row" style="text-align:left;white-space:nowrap;;">实现者</th>
9 |
  <td class="organiser" style=";;"><a href="/wiki/Adobe_Systems" title="Adobe Systems">Adobe Systems</a></td>
10 | </tr>      11 | <tr>
12 | <tr>
13 | <th scope="row" style="text-align:left;white-space:nowrap;;">启发语言</th>
14 |
  <td style=";;"><a href="/wiki/JavaScript" title="JavaScript">JavaScript</a>、<a href="/wiki/Java"
  title="Java">Java</a></td>
15 | </tr>      16 | </table>
```

参考文章：[\[python学习\] 简单爬取维基百科程序语言消息盒](#)

然后再在这个infobox内容中通过正则表达式进行分析爬取。下面讲述爬取属性-属性值：

爬取格式如下：

```
<table>
  <tr>
    <th>属性</th>
    <td>属性值</td>
  </tr>
</table>
```

其中th表示加粗处理，td和th中可能存在属性如title、id、type等值；同时<td></td>之间的内容可能存在<a href=..></a>或<span></span>或<br />等值，都需要处理。下面先讲解正则表达式获取td值的例子：

参考：<http://bbs.csdn.net/topics/390353859?page=1>

```

1 <table>
2 <tr>
3 <td>序列号</td><td>DEIN3-39CD3-2093J3</td>
4 <td>日期</td><td>2013年1月22日</td>
5 <td>售价</td><td>392.70 元</td>
6 <td>说明</td><td>仅限5用户使用</td>
7 </tr>
8 </table>

```

Python代码如下：

```

1 # coding=utf-8
2 import re
3
4 s = '''<table>
5 <tr>
6 <td>序列号</td><td>DEIN3-39CD3-2093J3</td>
7 <td>日期</td><td>2013年1月22日</td>
8 <td>售价</td><td>392.70 元</td>
9 <td>说明</td><td>仅限5用户使用</td>
10 </tr>
11 </table>
12 '''
13
14 res = r'<td>(.*?)</td><td>(.*?)</td>'
15 m = re.findall(res,s,re.S|re.M)
16 for line in m:
17     print unicode(line[0],'utf-8'),unicode(line[1],'utf-8') #unicode防止乱码
18
19 #输出结果如下:
20 #序列号 DEIN3-39CD3-2093J3
21 #日期 2013年1月22日
22 #售价 392.70 元
23 #说明 仅限5用户使用

```

如果<td id="">包含该属性则正则表达式为r'<td id=.\*?>(.\*?)</td>'; 同样如果不一定是id属性开头，则可以使用正则表达式r'<td .\*?>(.\*?)</td>'。

## 7.过滤<span></span>等标签

在获取值过程中，通常会存在<span>、<br>、<a href>等标签，下面举个例子过滤。

<td><span class="nickname">(字) 翔宇</span></td>过滤标签<span>核心代码：

```

elif "span" in nn: #处理标签<span>
    res_value = r'<span .*?>(.*?)</span>'
    m_value = re.findall(res_value,nn,re.S|re.M)
    for value in m_value:
        print unicode(value,'utf-8'),

```

代码如下，注意print中逗号连接字符串：

```

1 # coding=utf-8
2 import re
3

```

```

4 | language = ''
5 |
<table class="infobox bordered vcard" style="width: 21em; font-size: 89%; text-align: left;" cellpadding="3">
6 | <caption style="text-align: center; font-size: larger;" class="fn"><b>周恩来</b></caption> 7 | <tr>
8 | <th>性别: </th>
9 | <td>男</td>d
10 | </tr>
11 | <tr>
12 | <th>異名: </th>
13 | <td><span class="nickname">(字) 翔宇</span></td>
14 | </tr>
15 | <tr>
16 | <th>籍貫: </th>
17 |
<td><a href="../articles/%E6%B5%9981.html" title="浙江省">浙江省</a><a href="../articles/%E7%BB%8D82.html"
title="绍兴市">绍兴市</a></td>
18 | </tr>19 | </table>20 | '''
21 |
22 | #获取table中tr值
23 | res_tr = r'<tr>(.*?)</tr>'
24 | m_tr = re.findall(res_tr,language,re.S|re.M)
25 | for line in m_tr:
26 |     #获取表格第一列th 属性
27 |     res_th = r'<th>(.*?)</th>'
28 |     m_th = re.findall(res_th,line,re.S|re.M)
29 |     for mm in m_th:
30 |         if "href" in mm: #如果获取加粗的th中含超链接则处理
31 |             restr = r'<a href=.*?>(.*?)</a>'
32 |             h = re.findall(restr,mm,re.S|re.M)
33 |             print unicode(h[0],'utf-8'), #逗号连接属性值 防止换行
34 |         else:
35 |             print unicode(mm,'utf-8'), #unicode防止乱
36 |
37 |     #获取表格第二列td 属性值
38 |     res_td = r'<td>(.*?)</td>' #r'<td .*?>(.*?)</td>'
39 |     m_td = re.findall(res_td,line,re.S|re.M)
40 |     for nn in m_td:
41 |         if "href" in nn: #处理超链接<a href=../rel=..></a>
42 |             res_value = r'<a .*?>(.*?)</a>'
43 |             m_value = re.findall(res_value,nn,re.S|re.M)
44 |             for value in m_value:
45 |                 print unicode(value,'utf-8'),
46 |         elif "span" in nn: #处理标签<span>
47 |             res_value = r'<span .*?>(.*?)</span>'
48 |             m_value = re.findall(res_value,nn,re.S|re.M) #<td><span class="nickname">(字) 翔宇</span></td>
49 |             for value in m_value:
50 |                 print unicode(value,'utf-8'),
51 |         else:
52 |             print unicode(nn,'utf-8'),
53 |     print ' ' #换行

```

输出如下所示:

```

1 | >>>
2 | 性别: 男
3 | 異名: (字) 翔宇
4 | 籍貫: 浙江省 绍兴市
5 | >>>

```



---

## 8. 获取<script></script>等标签内容

比如在获取游讯网图库中，图集对应的原图它是存储在script中，其中获取原图-original即可，缩略图-thumb，大图-big，通过正则表达式下载URL：

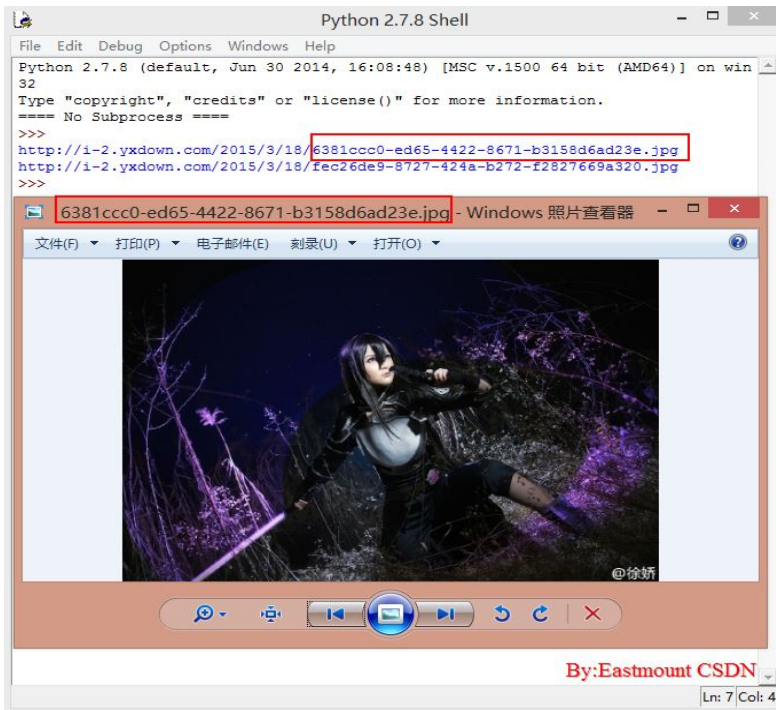
```
res_original = r'"original": "(.*?)"' #原图
m_original = re.findall(res_original, script)
```

代码如下：

```
1 # coding=utf-8
2 import re
3 import os
4
5 content = '''
6 <script>var images = [
7 { "big": "http://i-2.yxdown.com/2015/3/18/KDkwMHgp/6381ccc0-ed65-4422-8671-b3158d6ad23e.jpg",
8   "thumb": "http://i-2.yxdown.com/2015/3/18/KHgxMjAp/6381ccc0-ed65-4422-8671-b3158d6ad23e.jpg",
9   "original": "http://i-2.yxdown.com/2015/3/18/6381ccc0-ed65-4422-8671-b3158d6ad23e.jpg",
10  "title": "", "descript": "", "id": 75109},
11 { "big": "http://i-2.yxdown.com/2015/3/18/KDkwMHgp/fec26de9-8727-424a-b272-f2827669a320.jpg",
12   "thumb": "http://i-2.yxdown.com/2015/3/18/KHgxMjAp/fec26de9-8727-424a-b272-f2827669a320.jpg",
13   "original": "http://i-2.yxdown.com/2015/3/18/fec26de9-8727-424a-b272-f2827669a320.jpg",
14   "title": "", "descript": "", "id": 75110},
15 </script>
16 '''
17
18 html_script = r'<script>(.*?)</script>'
19 m_script = re.findall(html_script, content, re.S|re.M)
20 for script in m_script:
21     res_original = r'"original": "(.*?)"' #原图
22     m_original = re.findall(res_original, script)
23     for pic_url in m_original:
24         print pic_url
25         filename = os.path.basename(pic_url) #去掉目录路径, 返回文件名
26         urllib.urlretrieve(pic_url, 'E:\\'+filename) #下载图片
```

运行结果如下图所示，同时下载图片至E盘。

参考文章：[\[python学习\] 简单爬取图片网站图库中图片](#)



## 9.通过replace过滤<br />标签

在获取值过程中，通常会存<br />标签，它表示HTML换行的意思。常用的方法可以通过标签'<'和'>'进行过滤，但是这里我想讲述的是一种Python常用的过滤方法，在处理中文乱码或一些特殊字符时，可以使用函数replace过滤掉这些字符。核心代码如下：

```
if '<br />' in value:
    value = value.replace('<br />', '') #过滤该标签
    value = value.replace('\n', '') #换行空格替代 否则总换行
```

例如过滤前后的例子：

達洪阿 異名：(字) 厚菴<br />(諡) 武壯<br />(勇號) 阿克達春巴圖魯

達洪阿 異名：(字) 厚菴 (諡) 武壯 (勇號) 阿克達春巴圖魯

## 10.获取<img ../>中超链接及过滤<img>标签

在获取值属性值过程中，可能在分析table/tr/th/td标签后，仍然存在<img />图片链接，此时在获取文字内容时，你可能需要过滤掉这些<img>标签。这里采用的方法如下：

```
value = re.sub('<[^>]+>', '', value)
```

例如：

```
1 #encoding:utf-8
2 import os
3 import re
4
5 value = ''
6 <table class="infobox" style="width: 21em; text-align: left;" cellpadding="3">
7 <tr bgcolor="#CDDBE8">
8 <th colspan="2">
```

```

9 | <center class="role"><b>中華民國政治人士</b><br /></center>
10 | </th>
11 | </tr>
12 | <tr>
13 | <th>性別: </th>
14 | <td>男</td>
15 | </tr>
16 | <tr>
17 | <th>政黨: </th>
18 | <td><span class="org">
19 | 
20 | <a href="../../articles/%8B%E6%B0%91%E9%BB%A8.html" title="中國國民黨">中國國民黨</a></span></td>
21 | </tr>
22 | </table>
23 | '''
24 |
25 | value = re.sub('<[^>]+>', '', value) #过滤HTML 标签
26 | print value

```

输出如下:

```

1 | >>>
2 |
3 | 中華民國政治人士
4 |
5 | 性別:
6 | 男
7 |
8 | 政黨:
9 |
10 | 中國國民黨
11 |
12 | >>>

```

虽然仅仅包括汉字,但是中间会存在换行,需要过滤<br />即可:

```

1 | if '<br />' in value:
2 |     value = value.replace('<br />', '')
3 |     value = value.replace('\n', ' ')
4 | value = re.sub('<[^>]+>', '', value) #过滤HTML 标签
5 | print value
6 |
7 | #输出仅仅一行如下:
8 | #中華民國政治人士    性別: 男    政黨:    中國國民黨

```

下面讲述第二部分,通过正则表达式获取<img>中的src超链接,代码如下:

```

1 | test = ''''''
2 | print re.findall('src="(.*?)"', test)

```

输出如下所示:

```

1 | >>>
2 | ['../../images/Kuomintang.png']
3 | >>>

```


findall函数返回的总是正则表达式在字符串中所有匹配结果的列表，即findall中返回列表中每个元素包含的信息。

最后希望文章对你有所帮助，后面如果遇到新的相关知识，我也会即使的更新添加的。

(By: Eastmount 2016-04-07 清晨6点 <http://blog.csdn.net/eastmount/>)

👍 点赞 39    ☆ 收藏    🔄 分享



Eastmount  博客专家

发布了444 篇原创文章 · 获赞 5918 · 访问量 484万+