# 专栏：006：实战爬取博客

E - 爬虫技术　爬虫专栏

> 系列爬虫专栏
>
> 崇尚的学习思维是：输入，输出平衡，且平衡点不断攀升。
>
> 曾经有大神告诫说：没事别瞎写文章；所以，很认真的写的是能力范围内的，看客要是看不懂，不是你的问题，问题在我，得持续输入，再输出。
>
> 今天的主题是：实战爬取. (涉及python文件操作，requests，BeautifulSoup，结构化数据)

# 1：框架

| 序号 | 内容 | 解释 |
| --- | --- | --- |
| 01 | 内容介绍 | – |
| 02 | 分解如何操作 | – |
| 03 | 参考及介绍 | – |

# 2：内容介绍

- 目标
  抓取目标网站的全部博文：
  01： 博文的链接
  02： 博文的标题
  03： 博文的摘要
- 由来

`url = http://xlzd.me/`
昨天在学习基于github搭建博客的时候，无意间查看到这个人的博客，里面也有好些python和爬虫知识。xlzd杂谈
进一步发现是个罗粉。
你懂的。

- 时间花销
  一般写一篇专栏，需要花费80min，今天又是放假时间，所以效率不高，光从零开始写代码就花了60min。水平还是不够。
  原则上，基本的爬虫知识已经快属于我的舒适区。最近持续的专栏写作，一系列的总结，意味着，写出的多，输入的少，甚至写完这个系列，会写不出像样的其他专栏…

但我的理念是：持续精进。
所以，会尽快写完这个系列，进行输入数据科学知识。

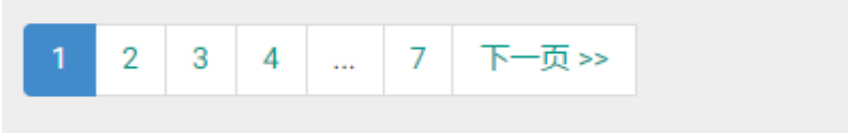
# 3：步骤分解

先总结下爬虫的大概步骤是什么：

1. 获取url : 不管是自己构造的符合要求的url还是抓取的url.
2. 下载网页源代码：requests 或者 urllib模块
3. 对网页源代码进行解析：re， BeautifulSoup系列，xpath系列等
4. 结构化数据，存储：本地，数据等

    - 原始：url
      `http://xlzd.me/`

查看网页翻页：




网页源代码：


得出网页的组成特征：给出两种自己构造url的方法

```
# 方法1:
    def search_url_one(self):
        self.all_url = ["http://xlzd.me/page/{}/".format(i) for i in range
(1, 8)]
        return self.all_url
        pass
# 方法2:
    def search_url_two(self):
        for one in range(1, 8):
            url = "http://xlzd.me/page/" + str(one) + '/'
            self.all_url.append(url)
        return self.all_url
```

当然也可以自己在网页中匹配，每抓取一页，把下一页的url抓取出来，总共7页，抓最后一页进行判断，说明这是最后一页。

- 对第一页分析抓取的目标：

文章的url
文章的标题
文章的摘要
网页源代码显示：

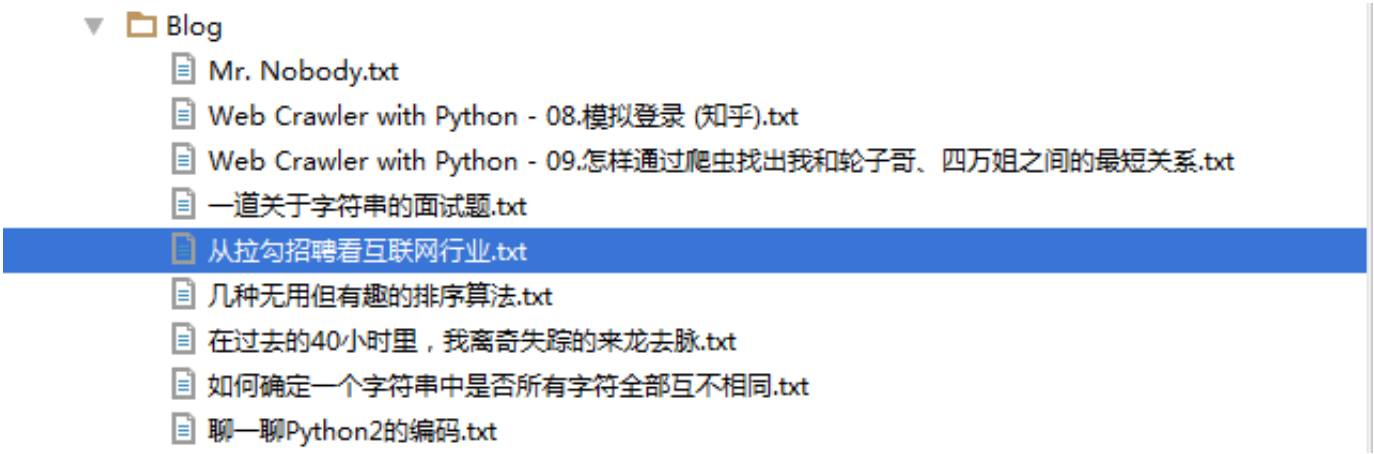
```
<h3 class="post-title"><a href="http://xlzd.me/2016/03/30/string-compress">一道关于字符串的面试题</a></h3>
<div class="post-meta">
        <span>作者：<a href="http://xlzd.me" target="_blank">xlzd</a> | </span>
        <!-- <span>作者：<a href="http://xlzd.me/author/1/">xlzd</a> | </span> -->
        <span>时间：2016-03-30 21:16:00 | </span>
        <span>分类：<a href="http://xlzd.me/category/tech/">技术</a> | </span>
        <span>评论：<a href="http://xlzd.me/2016/03/30/string-compress">0 评论</a> </span>
</div>
<div class="post-content"><p>今天的题目是：<strong>编写一个算法，实现基本的字符串“压缩”算法，比如对于字符串<code>abbbbbccccdddccccc</code>，经过算法处理之后得到的输出为<code>a1b5c4d3c6</code>，如果处理后的字符串长度不小于原串长度，则返回原串。</strong></p>
<p><img src="http://7xkpi6.com1.z0.glb.clouddn.com/blog/2016/03/30/1446758678.png" alt="屏幕快照 2016-03-30 21.58.25.png" /></p>
<p class="more"><a href="http://xlzd.me/2016/03/30/string-compress" title="一道关于字符串的面试题">阅读全文...</a></p></div>
</div>
```

```
all_title = Soup.find_all(class_="post-title")
all_abstract = Soup.find_all(class_="post-content")

# 上面已经把所需要的目标都容纳下来了。
# 具体的url 和标题 的匹配规则是：
for one_title in all_title:
    one_title.a.get('href'))   # 获取 一个url
    one_title.get_text()       # 获取 一个title

# 具体的摘要匹配规则是：
for one_abstract in all_abstract:
    one_abstract.get_text()    #获取一个abstract
```

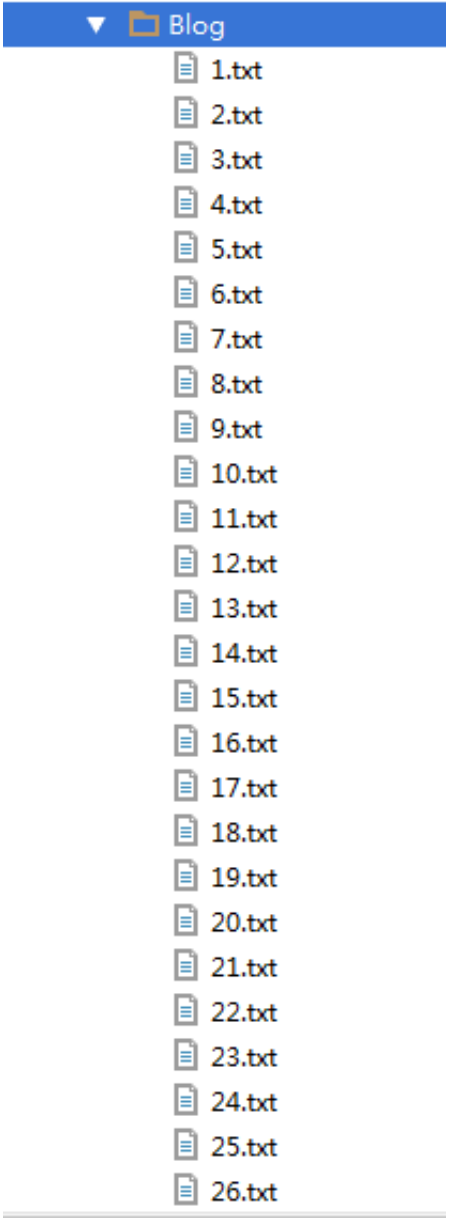
具体方法参考BeautifulSoup文档

大概的任务已经完成了。
一页中有8篇文章，一共有7页。
循环操作就可以实现抓取任务了。

- 写入文本操作
  具体要求是每篇文章的url，title，abstract 写入一个文本中。
  刚开始我的想法是把title当做 文本的名称：
  如下显示：




全部抓取的时候发现有些标题不规则会出错。所以进行了简化操作。
第一篇：1.txt
依次类推到最后一篇。



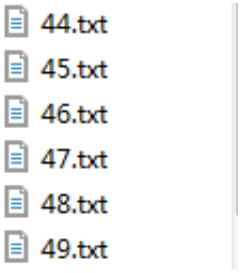

```
# 文本写入总会出现编码问题，注意下。
    def save(self, passage):
        global name
        for one in range(len(passage)):
            with codecs.open("Blog\\" + str(name) + ".txt", 'wb', encoding
='utf-8') as f:
                f.write(passage[one]["url"])
                f.write("\n")
                f.write(passage[one]["title"])
                f.write("\n")
                f.write(passage[one]["abstract"])
                f.write("\n")
                name +=1
```

具体一篇的显示如下：

010.py × 1.txt × 010.md ×

```
1  http://xlzd.me/2016/03/30/string-compress
2  一道关于字符串的面试题
3  今天的题目是：编写一个算法，实现基本的字符串“压缩”算法，比如对于字符串abbbbbccccdddccccccc，经过算法
4
5  阅读全文...
```

结果：7页网页，一页8篇文章，最后一页只有篇文章。




全部抓取完成。

获取到的全部文章的url还可以进行分析，比如如何把每篇文章的内容都抓取下来。

代码还可以进行重构。
你懂的。

# 4：参考及总结

大致流程走下来。爬虫的整体思路应该明朗了。

完整代码：完整代码

> 关于本人：
>
> > 国内小硕，半路出家的IT学习者。
> >
> > > 兴趣领域：爬虫 ， 数据科学
> > >
> > > > 本人正在构建一个共同成长爬虫小型社群。有兴趣私信
> > > > 答疑及分享。