## Python爬虫高级匹配

**本章所讲内容：**

1. **爬虫的概念**
2. **Beautifulsoup 概念**
3. **beautifulsoup操作**

**实战：爬取百度贴吧图片**

**实战：爬取不得姐视频**

1. **爬虫的概念**

**爬虫（spider：网络蜘蛛）:是一个用脚本代替浏览器请求服务器获取服务器资源的程序。**

**爬虫的用处**

**数据收集**

**数据的范围，数据存储**

**数据比对**

**倒买倒卖**

**模拟操作**

**模拟登陆**

**暴力登陆**

**代码登陆**

**浏览器驱动登陆**

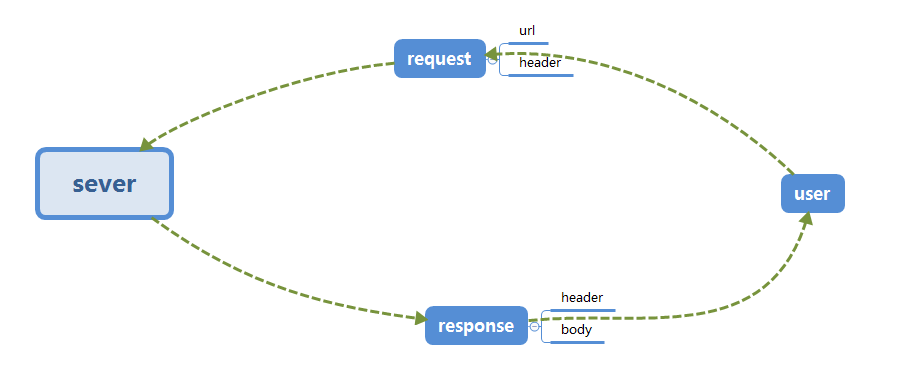
**评论、购票**

**自动化测试，压力测试**

**接口操作**

**爬虫的原理：**

**说到底，我们的爬虫是模拟web请求，不论学习什么框架我们都需要对http协议的请求和响应有所了解：**



**简单的了解一下这幅图。**

**2、beautifulsoup**

**如果一个正则匹配稍有差池，那可能程序就处在永久的循环之中，而且有的小伙伴们也对写正则表达式的写法用得不熟练，没关系，我们还有一个更强大的工具，叫Beautiful Soup，有了它我们可以很方便地提取出HTML或XML标签中的内容，实在是方便，这一节就让我们一起来感受一下Beautiful Soup的强大吧。**

**什么是Beautiful Soup**

**简单来说，Beautiful Soup是python的一个库，最主要的功能是从网页抓取数据。**

**官方解释如下:**

**Beautiful Soup提供一些简单的、python式的函数用来处理导航、搜索、修改分析树等功能。它是一个工具箱，通过解析文档为用户提供需要抓取的数据，因为简单，所以不需要多少代码就可以写出一个完整的应用程序。**

**Beautiful Soup自动将输入文档转换为Unicode编码，输出文档转换为utf-8编码。你不需要考虑编码方式，除非文档没有指定一个编码方式，这时，Beautiful Soup就不能自动识别编码方式了。然后，你仅仅需要说明一下原始编码方式就可以了。**

**Beautiful Soup已成为和lxml一样出色的python解释器，为用户灵活地提供不同的解析策略或强劲的速度。Beautiful Soup支持Python标准库中的HTML解析器,还支持一些第三方的解析器，如果我们不安装它，Python 会使用 Python默认的解析器，lxml 解析器更加强大，速度更快，推荐安装。**

### 2.2.1 安装与使用

**1、安装pip install bs4**

**2、使用**

**首先必须要导入 bs4 库， 创建BeautifulSoup对象**

**from** bs4 **import** BeautifulSoup **as** BS  
  
text = **'''  
<html>  
<head>  
 <meta = charset='UTF-8' >  
 <title id =1 href = 'http://example.com/elsie' class = 'title'>Test</title>  
</head>  
<body>  
 <div class = 'ok'>  
 <div class = 'nice'>  
 <p class = 'p'>  
 Hello World  
 </p>  
 <p class = 'e'>  
 风一般的男人  
 </p>  
 </div>  
 </div>  
</body>   
</html>  
'''  
soup = BS(text,"lxml")*#前面是要解析的内容，后面是指定的解析器*print(soup.prettify())*#转换字符串*print(type(soup.prettify()))  
print(type(soup))**

1. **解析器的分类**

**什么是网页解析器，简单的说就是用来解析html网页的工具，准确的说：它是一个HTML网页信息提取工具，就是从html网页中解析提取出“我们需要的有价值的数据”或者“新的URL链接”的工具。**

|  |  |  |  |
| --- | --- | --- | --- |
| **解析器** | **使用方法** | **优势** | **劣势** |
| **Python标准库** | **BeautifulSoup(markup, “html.parser”)** | **Python的内置标准库**  **执行速度适中**  **文档容错能力强** | **(Python 2.7.3 or 3.2.2)前 的版本中文档容错能力差** |
| **lxml HTML 解析器** | **BeautifulSoup(markup, “lxml”)** | **速度快**  **文档容错能力强** | **需要安装C语言库** |
| **lxml XML 解析器** | **BeautifulSoup(markup, [“lxml”, “xml”])**  **BeautifulSoup(markup, “xml”)** | **速度快**  **唯一支持XML的解析器** | **需要安装C语言库** |
| **html5lib** | **BeautifulSoup(markup, “html5lib”)** | **最好的容错性**  **以浏览器的方式解析文档**  **生成HTML5格式的文档** | **速度慢**  **不依** |

**Beautiful Soup将复杂HTML文档转换成一个复杂的树形结构,每个节点都是Python对象,所有对象可以归纳为4种：**

**1、Tag 就是Html中的标签，包含name和attrs**

**我们可以利用 soup加标签名轻松地获取这些标签的内容，是不是感觉比正则表达式方便多了？不过有一点是，它查找的是在所有内容中的第一个符合要求的标签，如果要查询所有的标签，我们在后面进行介绍。**

**#获取title标签**

**print(soup.title)**

**#获取标签的名字  
print(soup.title.name)**

**#获取标签的内容  
print(soup.title.text)**

**#获取标签的属性  
print(soup.title.attrs)**

**#指定标签的某个属性  
print(soup.title.get('class'))**

**2、NavigableString 获取标签的内容**

**#获取title标签中的内容**

**print(soup.title.string)**

**#获取p标签中的内容  
print(soup.p.string)**

**3、Beautifulsoup**

**BeautifulSoup 对象表示的是一个文档的全部内容.大部分时候,可以把它当作 Tag 对象，是一个特殊的 Tag，我们可以分别获取它的类型，名称，以及属性。**

**4、Comment 注释**

**Comment 对象是一个特殊类型的 NavigableString 对象，其实输出的内容仍然不包括注释符号，但是如果不好好处理它，可能会对我们的文本处理造成意想不到的麻烦。所以要先进行判断是否是注释，再输出。**

### 2.2.2 搜索文档树

**find()和find\_all()**

**find\_all()方法搜索当前tag的所有tag子节点,并判断是否符合过滤器的条件。**

**find（）和find\_all()的区别就是，find直接返回元素的一个结果，find\_all返回元素列表**

**find\_all( name , attrs , recursive , text , \*\*kwargs )简介一下参数**

**name 参数可以查找所有名字为name的tag,字符串对象会被自动忽略掉；name参数可以传入字符串、正则表达式、列表、True、自定义的方法等但是各自代表的含义不一样。**

**字符串，在搜索方法中传入一个字符串参数,Beautiful Soup会查找与字符串完整匹配的内容。**

**print(soup.find('body'))  
print(soup.find\_all('body')**

**正则表达式，Beautiful Soup会通过正则表达式的match()来匹配内容列表，Beautiful Soup会将与列表中任一元素匹配的内容返回。**

**#获取所有p标签的内容**

**import re**

**print(soup.find\_all(re.compile("^p")))**

**如果匹配成功将会匹配所有的tag**

**如果一个指定名字的参数不是搜索内置的一些参数名,搜索时会把该参数当作指定名字tag的属性来**

**搜索;例如id=1**

**如果包含一个名字为 id 的参数,Beautiful Soup会搜索每个tag的”id”属性；**

**如果传入 href 参数,Beautiful Soup会搜索每个tag的”href”属性；**

**使用多个指定名字的参数可以同时过滤tag的多个属性；**

**对于class ，可以使用class\_来搜索**

**import re  
print(soup.find\_all(href = re.compile('elsie'),id=1))**

**#返回这个class=‘p’的标签内容。**

**print(soup.find\_all('p',class\_='p'))**

**对于某些tag属性不能通过搜索得到值，可以使用attrs参数得到**

**#返回class为e的标签**

**print(soup.find\_all(attrs={'class':'e'}))**

### 2.2.3 CSS选择器

**我们在写 CSS 时，标签名不加任何修饰，类名前加点，id名前加 #，在这里我们也可以利用类似的方法来筛选元素，用到的方法是 soup.select()，返回类型是list。**

**#通过标签查找**

**print(soup.select('title'))**

**#通过类名查找**

**print(soup.select('.nice'))**

**#通过id查找**

**print(soup.select('#ok'))**

**练习一爬取百度贴吧**

**from bs4 import BeautifulSoup  
import requests  
*#爬取目标网页*html = requests.get("https://tieba.baidu.com/p/5950745302").text  
*#解析网页*result = BeautifulSoup(html,"html.parser")  
*#获取所有的图片img*result\_imgs = result.find\_all("img",class\_="BDE\_Image")  
i=1  
for result\_img in result\_imgs:  
 *#获取链接* img\_url = result\_img['src']  
 *#获取文件* result\_img\_content=requests.get(img\_url).content  
 *#声明文件名* file\_name = './images/'+str(i)+".jpg"  
 *#保存图片* with open(file\_name,"wb") as wf:  
 wf.write(result\_img\_content)  
 i+=1  
*#逐个将图片保存到本地***

**练习二爬取不得姐**

**from bs4 import BeautifulSoup  
import requests  
import re,time  
url="http://www.budejie.com/video/"  
def get\_page(url,data=None):  
 header={'User-Agent': 'Mozilla/5.0 (Windows NT 6.1; WOW64; rv:51.0) Gecko/20100101 Firefox/51.0'}  
 html = requests.get(url,headers=header)  
 soup=BeautifulSoup(html.text,"html.parser")  
 lists = soup.findAll('a',href=re.compile('http://svideo.spriteapp.com/video/2019/(.\*?).mp4'))  
 print(lists)  
 a = 0  
 for i in lists:  
 a+=1  
 url\_href = i.get("href")  
 print(url\_href)  
 req= requests.get(url\_href)  
 print("num"+str(a)+"video")  
 with open(str(time.time())+".mp4","wb") as file:  
 file.write(req.content)  
def get\_more\_pages(start,end):  
 for one in range(start,end):  
 get\_page(url+str(one))  
 time.sleep(2)  
if \_\_name\_\_ == '\_\_main\_\_':  
 get\_more\_pages(1,2)**