关于BeautifulSoup的总结（二）

经常使用的两种方法：find(str)和find_all(str)。

其中的str，代表了tag的name。可以是纯字符串、正则表达式、列表（任一匹配就满足条件，是或运算）、True（返回所有Tag节点不返回字符串节点）。

另一种入参不是str，而是method。此方法是一个函数，只接受一个元素入参，若此函数返回True表示入参匹配要求。例如：

*def has_class_but_no_id(tag): *

return tag.has_attr('class') and not tag.has_attr('id')

综上，过滤器包括：纯字符串、正则表达式、列表、True、方法这几种。

1. find_all(name,attrs,recursive,text,kwargs)**

该方法搜索当前节点的所有tag子节点。

name参数：

指的是tag的name属性，字符串对象自动忽略。

过滤器可以使用全部种类。

keyword参数：

如果一个入参指定了名字，但是并不是上述提到的入参名字，搜索时会把该入参当做是tag的属性来搜索。例如：

soup.find_all(id='link2')

会返回tag中存在属性id，并且id对应的值是link2的tag。

以上方法可使用除方法之外的所有过滤器。

某些特殊属性不能这样直接使用，则使用如下方法：

soup.find_all(attrs={"key":"value"})

例如要使用class属性进行搜索，由于class是python中的保留字，不能直接写成入参，目前有两种方法：

soup.find_all('tag.name',class_='class_value')

soup.find_all('tag.name',attrs={'class':'class_value'})

class_方法可以使用全部过滤器。

另外，因为class是一个多值属性，所以只需要匹配一个值，就可以得到结果，所谓的不完全匹配。

使用完全匹配时，过滤器中的字符顺序需要和实际相符合才能得到对应结果。

text参数：

搜索的是Tag中的字符串内容，可使用全部过滤器。

limit参数：
限制返回数量。

recursive参数：
find_all()默认是搜索当前节点的所有子孙节点，若只需要搜索直接的子节点，则设置*recursive=False*。

find_all()是实际当中用的最广泛的。
因此有了等价的简化版：
soup.find_all('a')
soup('a')

2. find(name,attrs,recursive,text,kwargs)**

find()方法等价于find_all(limit=1)，返回符合条件的第一个对象。
区别在于，前者直接返回结果，后者返回只有一个元素的列表。若没有对象符合条件，前者返回None，后者返回空列表。

它也有简化版：
soup.find('head').find('title')
soup.head.title

除了find()和find_all()之外还有一些搜索的方法：
find_parent()
find_next_sibling()
find_previous_sibling()
上面三种可以在后面加's'表示所有。
find_next()
find_previous()
find_all_next()
find_all_previous()

3. CSS选择器
Tag或BeautifulSoup对象的.select()方法。

输出

*prettify()*将文档树格式化之后输出。
若不注重格式，则可使用python的*str()*或*unicode()*。

如果想得到tag中包含的文本内容，使用*get_text()*，可获取到当前节点的文本，以及子孙节点中的文本。返回的是Unicode。
可以指定参数设置分隔符如*get_text("|")*是以"|"作为分隔符。
*get_text(strip=True)*可去除文本前后的空白。
或者用*.stripped_strings*进行遍历。

文档解析器

BeautifulSoup的第一个入参是文档，第二个入参是文档解析器，默认情况下的优先顺序是：lxml， html5lib，python标准库。其中只有lxml支持xml文档的解析。

编码

soup使用Unicode编码。
BeautifulSoup进行了编码检测并自动转为Unicode。
BeautifulSoup对象的*.original_encoding*属性来获取自动识别编码的结果。
当然这样比较慢，有时候会出错。可以在创建BeautifulSoup对象时，指定入参from_encoding来告知文档的编码方式。

有时候转码时有些特殊字符替换成了特殊的Unicode，可通过BeautifulSoup对象的*.contains_repalcement_characters*属性来判断是否有此情况，为True即为有特殊替换。

输出编码统一为UTF8，若想要其他的编码，则和一般的python字符串相同，需要进行手动设置。

使用chardet库可提高编码检测效率。