python SGMLParser

python的**SGMLParser**模块处理html解析非常的方便，它是处理网页解析的好帮手，能够将HTML 处理分成三步：将 HTML 分解成它的组成片段，对片段进行加工，接着将片段再重新合成 HTML。第一步是通过 sgmllib.py 来完成的，它是标准 Python 库的一部分。  
  
理解本章的关键是要知道 HTML 不只是文本，更是结构化文本。这种结构来源于开始与结束标记的或多或少分级序列。通常您并不以这种方式处理 HTML ，而是以文本方式 在一个文本编辑中对其进行处理，或以可视的方式 在一个浏览器中进行浏览或页面编辑工具中进行编辑。sgmllib.py 表现出了 HTML 的结构。  
  
sgmllib.py 包含一个重要的类：SGMLParser。SGMLParser 将 HTML 分解成有用的片段，比如开始标记和结束标记。在它成功地分解出某个数据为一个有用的片段后，它会根据所发现的数据，调用一个自身内部的方法。为了使用这个分析器，您需要子类化 SGMLParser 类，并且覆盖这些方法。这就是当我说它表示了 HTML 结构 的意思：HTML 的结构决定了方法调用的次序和传给每个方法的参数。  
  
SGMLParser 将 HTML 分析成 8 类数据，然后对每一类调用单独的方法：  
  
开始标记 (Start tag)  
是开始一个块的 HTML 标记，像 、、 或

等，或是一个独一的标记，

像

或 等。当它找到一个开始标记 tagname，

SGMLParser 将查找名为 start\_tagname 或 do\_tagname 的方法。

例如，当它找到一个

标记，它将查找一个 start\_pre 或 do\_pre 的方法。

如果找到了，SGMLParser 会使用这个标记的属性列表来调用这个方法；

否则，它用这个标记的名字和属性列表来调用 unknown\_starttag 方法。

结束标记 (End tag)

是结束一个块的 HTML 标记，像 、、 或

等。  
当找到一个结束标记时，SGMLParser 将查找名为 end\_tagname 的方法。  
如果找到，SGMLParser 调用这个方法，否则它使用标记的名字来调用 unknown\_endtag 。   
字符引用 (Character reference)  
用字符的十进制或等同的十六进制来表示的转义字符，  
像  。当找到，SGMLParser 使用十进制或等同的十六进制字符文本来调用 handle\_charref 。   
实体引用 (Entity reference)  
HTML 实体，像 ©。当找到，SGMLParser 使用 HTML 实体的名字来调用 handle\_entityref 。   
注释 (Comment)  
HTML 注释，包括在 之间。当找到，SGMLParser 用注释内容来调用 handle\_comment。   
处理指令 (Processing instruction)  
HTML 处理指令，包括在 之间。当找到，SGMLParser 用处理指令内容来调用 handle\_pi。   
声明 (Declaration)  
HTML 声明，如 DOCTYPE，包括在 之间。当找到，SGMLParser 用声明内容来调用 handle\_decl。   
文本数据 (Text data)  
文本块。不满足其它 7 种类别的任何东西。当找到，SGMLParser 用文本来调用 handle\_data。   
  
重要  
Python 2.0 存在一个 bug，即 SGMLParser 完全不能识别声明 (handle\_decl 永远不会调用)，  
这就意味着 DOCTYPE 被静静地忽略掉了。这个错误在 Python 2.1 中改正了。  
sgmllib.py 所附带的一个测试套件举例说明了这一点。您可以运行 sgmllib.py，  
在命令行下传入一个 HTML 文件的名字，然后它会在分析标记和其它元素的同时将它们打印出来。  
它的实现是通过子类化 SGMLParser 类，然后定义 unknown\_starttag，unknown\_endtag，handle\_data   
和其它方法来实现的。这些方法简单地打印出它们的参数。  
  
如果要详细的了解python SGMLParser的用法的话，可以看看[python SGMLParser](http://www.tsnc.edu.cn/default/tsnc_wgrj/doc/pythonhtml/html/html_processing/introducing_sgmllib.html)的文档，  
下面放上个我用python SGMLParsr写的例子，希望对大家有帮助：  
  
#encoding=utf-8  
#@description:baidutiba content  
  
import sys  
import re  
  
import httplib  
import urllib  
from sgmllib import SGMLParser  
  
class BaidutiebaParser(SGMLParser):  
'''在百度贴吧里采集相应的关键字的标题'''  
  
def reset(self):  
SGMLParser.reset(self)  
self.info = []#  
self.q\_check = 0  
self.num = 0  
self.strcontent = ''  
  
  
def start\_td(self, tag):  
'''匹配 标签'''  
if len(tag)!=0 and tag[0][1] == 's':  
self.num = self.num + 1  
self.q\_check = 1  
  
def handle\_data(self, text):  
'''处理文本'''  
txt = text.strip()  
if txt and self.q\_check:  
for i in checklist:  
pipei = r'%s' % str(i)#在要匹配的信息里找到和关键字匹配  
check\_pan = re.compile(pipei)  
if check\_pan.search(txt) is not None:  
self.info.append(txt)  
else:  
continue  
self.strcontent = '$|$'.join(self.info)  
  
def end\_td(self):  
'''匹配'''  
self.q\_check = 0  
  
  
############################################配置信息#############################   
keylist = ['旅游']#贴吧名称  
checklist = ['张家界','韩国']#要查询的关键字  
content = {}#采集内容  
  
for m in keylist:  
page = 0  
keyword = urllib.quote(m.decode('utf-8').encode('gbk'))  
for i in xrange(10):  
url = '''http://tieba.baidu.com/f?z=0&ct=318767104&lm=11&sc=0&rn=50&tn=baiduKeywordSearch&rs3=0&rs4=0&word=%s&pn=%s''' % (str(keyword),str(page))  
data = urllib.urlopen(url).read()  
data = unicode(data, 'gbk').encode('utf-8')  
parser = BaidutiebaParser()  
parser.feed(data)  
content[i+1] = parser.strcontent  
page = page + 50  
  
for k in content.keys():  
print k  
print content[k]   
  
  
如果你对[python queue模块](http://www.cnpythoner.com/post/46.html)也感兴趣的话，可以看看！  
上面的例子也用到了[python 字符串替换](http://www.cnpythoner.com/post/65.html)的一些方法,还有一些和python网页采集有关的： [使用python BeautifulSoup 解析网页](http://www.cnpythoner.com/post/116.html) ， [python beautifulsoup多线程分析抓取网页](http://www.cnpythoner.com/post/pythonduoxianchen.html)  
  
作者:老王@[python](http://www.cnpythoner.com/) [python教程](http://www.cnpythoner.com/)  
老王python,提供pythn相关的python教程和[python下载](http://www.cnpythoner.com/)，希望大家能够喜欢