**lxml基本用法总结**

2017年02月24日 15:19:36

阅读数：979

lxml库结合libxml2快速强大的特性，使用xpath语法来进行文件格式解析，与Beautiful相比，效率更高。

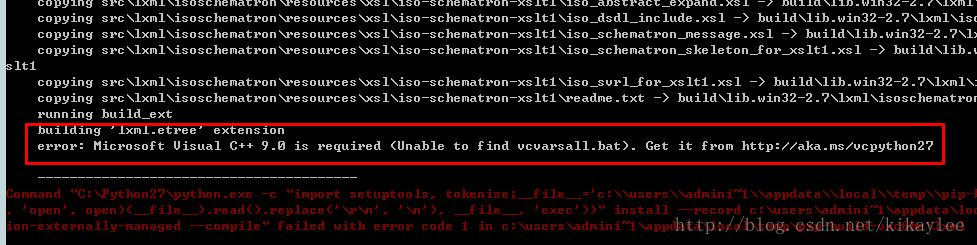
**0x01 安装**

可以利用pip安装lxml：

pip install lxml

* 1

在windows系统中安装时，可能会出现如下错误：



提示如下：

error: Microsoft Visual C++ 9.0 is required (Unable to find vcvarsall.bat). Get it from http://aka.ms/vcpython27

* 1

需要安装Microsoft Visual C++ 9.0，对应的是visual studio 2008版本。其实我们可以用一种更简单的方式来解决这个问题。

（1）首先升级下pip：

python -m pip install -U pip

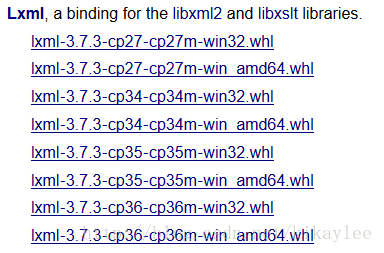
* 1

（2）安装wheel

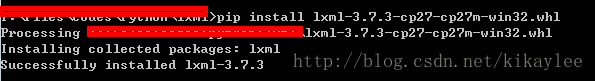
pip install wheel

* 1

（3）下载lxml对应python版本的wheel文件：[下载地址](http://www.lfd.uci.edu/~gohlke/pythonlibs/#lxml)



其中cp27表示版本就是Python2.7，最后执行安装：



lxml安装完毕。

**0x02 节点与属性**

Element类是lxml的一个基础类，大部分XML都是通过Element存储的。可以通过Element方法创建：

>>> from lxml import etree

>>> root=etree.Element('root');

>>> print root.tag

root

* 4

为root节点添加子节点：

>>> child1=etree.SubElement(root,'child1')

>>> print root

<Element root at 0x2246760>

>>> print etree.tostring(root)

<root><child1/></root>

* 5

XML Element的属性格式为Python的dict。可以通过get/set方法进行设置或获取操作：

>>> root.set('id','123')

>>> id=root.get('id')

>>> id

'123'

* 3
* 4

遍历全部属性：

>>> for value,name in root.items():

... print value,'\t',name

...

id 123

**0x03 文本操作**

Element的text属性可以访问标签的文本：

>>> print etree.tostring(root)

<root id="123">root<child1 name="kikay">ttt</child1></root>

>>> root.text

'root'

>>> child1.text

'ttt'

>>>

* 7

XML的标签是成对出现的，但是对于HTML而言，可能存在  
这样的单一标签，可以通过tail来读取文本：

>>> etree.tostring(root)

'<root id="123">root<child1 name="kikay">ttt</child1><br/>br\_test</root>'

>>> root.tail

>>> br.tail

'br\_test'

* 5

tail返回的是当前标签到下一次出现标签时的文本内容。

（2）xpath方式

>>> etree.tostring(root)

'<root><child1>child1 test</child1><child2>child2 test</child2></root>123'

#方法1：过滤标签，返回全部文本

>>> root.xpath('string()')

'child1 testchild2 test'

#方法2：以标签为间隔，返回list

>>> root.xpath('//text()')

['child1 test', 'child2 test', '123']

方法2中的list元素都携带了标签的信息，可以通过如下方式获取：

>>> lists=root.xpath('//text()')

>>> lists

['child1 test', 'child2 test', '123']

>>> lists[0].getparent()

<Element child1 at 0x2203c60>

>>> lists[0].getparent().tag

'child1'

>>> lists[1].getparent().tag

'child2'

>>> lists[2].getparent().tag

'root'

还可以通过is\_text和is\_tail判断标签类型：

>>> lists[2].is\_text

False

>>> lists[2].is\_tail

True

4

**0x04 文本输入与输出**

lxml提供如下方式输入文本：

fromstring():解析字符串

HTML():解析HTML对象

XML():解析XML对象

parse():解析文件类型对象4

输出就是前面讲的tostring()方法：

>>> root = etree.XML('<root><a><b/></a></root>')

>>> etree.tostring(root)

'<root><a><b/></a></root>'

>>> etree.tostring(root,xml\_declaration=True)

"<?xml version='1.0' encoding='ASCII'?>\n<root><a><b/></a></root>"

>>> etree.tostring(root,xml\_declaration=True,encoding='utf-8')

"<?xml version='1.0' encoding='utf-8'?>\n<root><a><b/></a></root>"

**0x05 标签搜索**

可以使用find、findall或者xpath来搜索Element包含的标签对象。区别如下：

find():返回第一个匹配对象，并且xpath语法只能使用相对路径（以’.//’开头）；

findall():返回一个标签对象的列表，并且xpath语法只能使用相对路径（以’.//’开头）；

xpath()：返回一个标签对象的列表，并且xpath语法的相对路径和绝对路径。

>>> root = etree.XML("<root><a x='123'>aText<b/><c/><b/></a></root>")

>>> x=root.find('.//a[@x]')

>>> x

<Element a at 0x2242c10>

>>> x.text

'aText'

>>> x.tag

'a'

>>> x2=root.findall('.//a[@x]')

>>> x2

[<Element a at 0x2242c10>]

>>> type(x2)

<type 'list'>

>>> x3=root.xpath('//a[@x]')

>>> type(x3)

<type 'list'>

>>> x3

[<Element a at 0x2242c10>]

此外，lxml还支持css语法的选择方式，对于熟悉JQuery选择器的开发者是一个很好的补充（需要安装pip install cssselect）：

>>> root = etree.XML("<root><a class='\_123'>aText<b id='b1'/><c/><b/></a></root>")

>>> a1=root.cssselect('.\_123')

>>> a1[0].tag

'a'

>>> root = etree.XML("<root><a class='c123'>aText<b id='b1'/><c/><b/></a></root>")

>>> a1=root.cssselect('a')

>>> a1[0].text

'aText'

>>> a2=root.cssselect('.c123')

>>> a2[0].text

'aText'

>>> b=root.cssselect('#b1')

>>> b[0].tag

'b'

4

**0x06 解析HTML**

lxml可以通过etree.HTML()来加载一个HTML页面:

#coding:utf-8

from lxml import etree

import requests

from chardet import detect

url='http://tool.chinaz.com/'

resp=requests.get(url,timeout=50)

html=resp.content

#识别编码

cder=detect(html)

html=html.decode(cder.get('encoding'))

tree=etree.HTML(html)

#打印全部a标签

hrefs=tree.xpath('//a')

for href in hrefs:

print href.get('href'),'\t',href.text

使用lxml解析HTML页面时，一定要注意编码的问题，参考（[Python学习笔记：Python字符编码问题的经验小结](http://blog.csdn.net/kikaylee/article/details/56686424)）

如果HTML页面中的script和style变迁之间的内容影响解析页面，可以将其清洗掉：

from lxml.html.clean import Cleaner

clear=Cleaner(style=True,scripts=True,page\_structure=False,safe\_attrs\_only=False)

print clear.clean\_html(html)

此外，可以借助浏览器来帮我们生成xpath语法:



下面是提取豆瓣读书主页图书信息的例子：

#coding:utf-8

from lxml import etree

import requests

from chardet import detect

url='https://book.douban.com/'

resp=requests.get(url,timeout=15)

ecoding=detect(resp.content).get('encoding')

html=resp.content.decode(ecoding)

tree=etree.HTML(html)

for book in tree.xpath('//div[@class="info"]'):

title=book.xpath('.//a[@title]')[0].text.strip()

author=book.xpath('.//div[@class="author"]')[0].text.strip()

print u'《',title,u'》','\t','--',author

结果：

《 这世界偷偷爱着你 》 -- 辉姑娘

《 我与世界只差一个你 》 -- 张皓宸

《 好好学习 》 -- 成甲

《 生育对话录 》 -- 宋涵

《 被误诊的艺术史 》 -- 董悠悠

《 多拉•布吕代 》 -- [法] 帕特里克•莫迪亚诺

《 我们的后人类未来 》 -- [美] 弗朗西斯•福山

版权声明：本文为博主原创文章，未经博主允许不得转载。 https://blog.csdn.net/kikaylee/article/details/56842194