PyQuery库也是一个非常强大又灵活的网页解析库，如果你有前端开发经验的，都应该接触过jQuery,那么PyQuery就是你非常绝佳的选择，PyQuery 是 Python 仿照 jQuery 的严格实现。语法与 jQuery 几乎完全相同，所以不用再去费心去记一些奇怪的方法了。

*官网地址：*[*http://pyquery.readthedocs.io/en/latest/*](http://pyquery.readthedocs.io/en/latest/) *jQuery参考文档：*[*http://jquery.cuishifeng.cn/*](http://jquery.cuishifeng.cn/)

**初始化**

初始化的时候一般有三种传入方式：传入字符串，传入url,传入文件

**字符串初始化**

[复制代码](javascript:void(0);)

html = '''

<div>

<ul>

<li class="item-0">first item</li>

<li class="item-1"><a href="link2.html">second item</a></li>

<li class="item-0 active"><a href="link3.html"><span class="bold">third item</span></a></li>

<li class="item-1 active"><a href="link4.html">fourth item</a></li>

<li class="item-0"><a href="link5.html">fifth item</a></li>

</ul>

</div>

'''

from pyquery import PyQuery as pq

doc = pq(html)

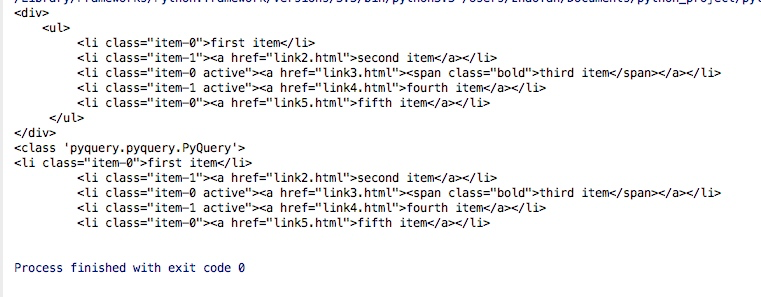
print(doc)

print(type(doc))

print(doc('li'))

[复制代码](javascript:void(0);)

结果如下：



由于PyQuery写起来比较麻烦，所以我们导入的时候都会添加别名：  
from pyquery import PyQuery as pq

这里我们可以知道上述代码中的doc其实就是一个pyquery对象，我们可以通过doc可以进行元素的选择，其实这里就是一个css选择器，所以CSS选择器的规则都可以用，直接doc(标签名)就可以获取所有的该标签的内容，如果想要获取class 则doc('.class\_name'),如果是id则doc('#id\_name')....

**URL初始化**

from pyquery import PyQuery as pq

doc = pq(url="http://www.baidu.com",encoding='utf-8')

print(doc('head'))

**文件初始化**

我们在pq()这里可以传入url参数也可以传入文件参数，当然这里的文件通常是一个html文件，例如：pq(filename='index.html')

**基本的CSS选择器**

[复制代码](javascript:void(0);)

html = '''

<div id="container">

<ul class="list">

<li class="item-0">first item</li>

<li class="item-1"><a href="link2.html">second item</a></li>

<li class="item-0 active"><a href="link3.html"><span class="bold">third item</span></a></li>

<li class="item-1 active"><a href="link4.html">fourth item</a></li>

<li class="item-0"><a href="link5.html">fifth item</a></li>

</ul>

</div>

'''

from pyquery import PyQuery as pq

doc = pq(html)

print(doc('#container .list li'))

[复制代码](javascript:void(0);)

这里我们需要注意的一个地方是doc('#container .list li')，这里的三者之间的并不是必须要挨着，只要是层级关系就可以,下面是常用的CSS选择器方法：



**查找元素**

**子元素**  
children,find  
代码例子：

[复制代码](javascript:void(0);)

html = '''

<div id="container">

<ul class="list">

<li class="item-0">first item</li>

<li class="item-1"><a href="link2.html">second item</a></li>

<li class="item-0 active"><a href="link3.html"><span class="bold">third item</span></a></li>

<li class="item-1 active"><a href="link4.html">fourth item</a></li>

<li class="item-0"><a href="link5.html">fifth item</a></li>

</ul>

</div>

'''

from pyquery import PyQuery as pq

doc = pq(html)

items = doc('.list')

print(type(items))

print(items)

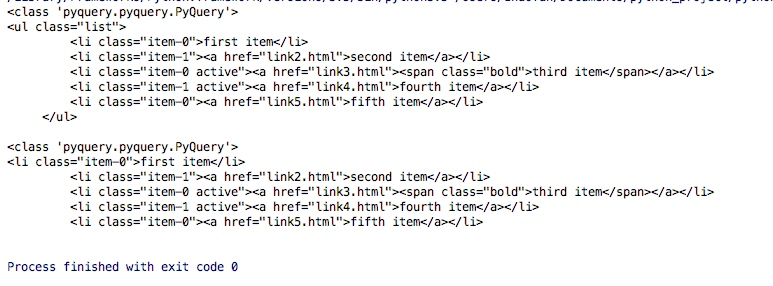
lis = items.find('li')

print(type(lis))

print(lis)

[复制代码](javascript:void(0);)

运行结果如下



从结果里我们也可以看出通过pyquery找到结果其实还是一个pyquery对象，可以继续查找，上述中的代码中的items.find('li') 则表示查找ul里的所有的li标签  
当然这里通过children可以实现同样的效果,并且通过.children方法得到的结果也是一个pyquery对象

li = items.children()

print(type(li))

print(li)

同时在children里也可以用CSS选择器

li2 = items.children('.active') print(li2)

**父元素**  
parent,parents方法

通过.parent就可以找到父元素的内容，例子如下：

[复制代码](javascript:void(0);)

html = '''

<div id="container">

<ul class="list">

<li class="item-0">first item</li>

<li class="item-1"><a href="link2.html">second item</a></li>

<li class="item-0 active"><a href="link3.html"><span class="bold">third item</span></a></li>

<li class="item-1 active"><a href="link4.html">fourth item</a></li>

<li class="item-0"><a href="link5.html">fifth item</a></li>

</ul>

</div>

'''

from pyquery import PyQuery as pq

doc = pq(html)

items = doc('.list')

container = items.parent()

print(type(container))

print(container)

[复制代码](javascript:void(0);)

通过.parents就可以找到祖先节点的内容，例子如下：

[复制代码](javascript:void(0);)

html = '''

<div class="wrap">

<div id="container">

<ul class="list">

<li class="item-0">first item</li>

<li class="item-1"><a href="link2.html">second item</a></li>

<li class="item-0 active"><a href="link3.html"><span class="bold">third item</span></a></li>

<li class="item-1 active"><a href="link4.html">fourth item</a></li>

<li class="item-0"><a href="link5.html">fifth item</a></li>

</ul>

</div>

</div>

'''

from pyquery import PyQuery as pq

doc = pq(html)

items = doc('.list')

parents = items.parents()

print(type(parents))

print(parents)

[复制代码](javascript:void(0);)

结果如下：从结果我们可以看出返回了两部分内容，一个是的父节点的信息，一个是父节点的父节点的信息即祖先节点的信息



同样我们通过.parents查找的时候也可以添加css选择器来进行内容的筛选

**兄弟元素**  
siblings

[复制代码](javascript:void(0);)

html = '''

<div class="wrap">

<div id="container">

<ul class="list">

<li class="item-0">first item</li>

<li class="item-1"><a href="link2.html">second item</a></li>

<li class="item-0 active"><a href="link3.html"><span class="bold">third item</span></a></li>

<li class="item-1 active"><a href="link4.html">fourth item</a></li>

<li class="item-0"><a href="link5.html">fifth item</a></li>

</ul>

</div>

</div>

'''

from pyquery import PyQuery as pq

doc = pq(html)

li = doc('.list .item-0.active')

print(li.siblings())

[复制代码](javascript:void(0);)

代码中doc('.list .item-0.active') 中的.tem-0和.active是紧挨着的，所以表示是并的关系，这样满足条件的就剩下一个了：thired item的那个标签了  
这样在通过.siblings就可以获取所有的兄弟标签，当然这里是不包括自己的  
同样的在.siblings()里也是可以通过CSS选择器进行筛选

**遍历**

**单个元素**

[复制代码](javascript:void(0);)

html = '''

<div class="wrap">

<div id="container">

<ul class="list">

<li class="item-0">first item</li>

<li class="item-1"><a href="link2.html">second item</a></li>

<li class="item-0 active"><a href="link3.html"><span class="bold">third item</span></a></li>

<li class="item-1 active"><a href="link4.html">fourth item</a></li>

<li class="item-0"><a href="link5.html">fifth item</a></li>

</ul>

</div>

</div>

'''

from pyquery import PyQuery as pq

doc = pq(html)

li = doc('.item-0.active')

print(li)

lis = doc('li').items()

print(type(lis))

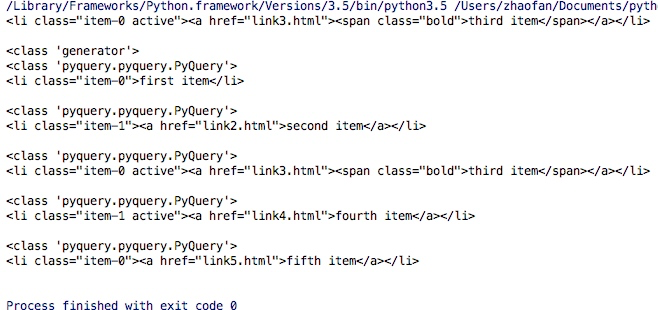
for li in lis:

print(type(li))

print(li)

[复制代码](javascript:void(0);)

运行结果如下：从结果中我们可以看出通过items()可以得到一个生成器，并且我们通过for循环得到的每个元素依然是一个pyquery对象。



**获取信息**

**获取属性**  
pyquery对象.attr(属性名)  
pyquery对象.attr.属性名

[复制代码](javascript:void(0);)

html = '''

<div class="wrap">

<div id="container">

<ul class="list">

<li class="item-0">first item</li>

<li class="item-1"><a href="link2.html">second item</a></li>

<li class="item-0 active"><a href="link3.html"><span class="bold">third item</span></a></li>

<li class="item-1 active"><a href="link4.html">fourth item</a></li>

<li class="item-0"><a href="link5.html">fifth item</a></li>

</ul>

</div>

</div>

'''

from pyquery import PyQuery as pq

doc = pq(html)

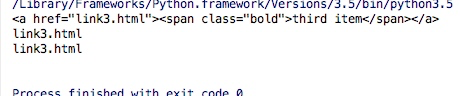
a = doc('.item-0.active a')

print(a)

print(a.attr('href'))

print(a.attr.href)

[复制代码](javascript:void(0);)



所以这里我们也可以知道获得属性值的时候可以直接a.attr(属性名)或者a.attr.属性名

**获取文本**  
在很多时候我们是需要获取被html标签包含的文本信息,通过.text()就可以获取文本信息

[复制代码](javascript:void(0);)

html = '''

<div class="wrap">

<div id="container">

<ul class="list">

<li class="item-0">first item</li>

<li class="item-1"><a href="link2.html">second item</a></li>

<li class="item-0 active"><a href="link3.html"><span class="bold">third item</span></a></li>

<li class="item-1 active"><a href="link4.html">fourth item</a></li>

<li class="item-0"><a href="link5.html">fifth item</a></li>

</ul>

</div>

</div>

'''

from pyquery import PyQuery as pq

doc = pq(html)

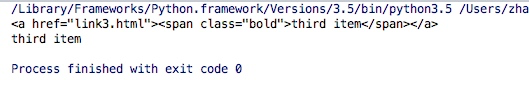
a = doc('.item-0.active a')

print(a)

print(a.text())

[复制代码](javascript:void(0);)

结果如下：



**获取html**  
我们通过.html()的方式可以获取当前标签所包含的html信息，例子如下：

[复制代码](javascript:void(0);)

html = '''

<div class="wrap">

<div id="container">

<ul class="list">

<li class="item-0">first item</li>

<li class="item-1"><a href="link2.html">second item</a></li>

<li class="item-0 active"><a href="link3.html"><span class="bold">third item</span></a></li>

<li class="item-1 active"><a href="link4.html">fourth item</a></li>

<li class="item-0"><a href="link5.html">fifth item</a></li>

</ul>

</div>

</div>

'''

from pyquery import PyQuery as pq

doc = pq(html)

li = doc('.item-0.active')

print(li)

print(li.html())

[复制代码](javascript:void(0);)

结果如下：



**DOM操作**

**addClass、removeClass**  
熟悉前端操作的话，通过这两个操作可以添加和删除属性

[复制代码](javascript:void(0);)

html = '''

<div class="wrap">

<div id="container">

<ul class="list">

<li class="item-0">first item</li>

<li class="item-1"><a href="link2.html">second item</a></li>

<li class="item-0 active"><a href="link3.html"><span class="bold">third item</span></a></li>

<li class="item-1 active"><a href="link4.html">fourth item</a></li>

<li class="item-0"><a href="link5.html">fifth item</a></li>

</ul>

</div>

</div>

'''

from pyquery import PyQuery as pq

doc = pq(html)

li = doc('.item-0.active')

print(li)

li.removeClass('active')

print(li)

li.addClass('active')

print(li)

[复制代码](javascript:void(0);)

**attr,css**  
同样的我们可以通过attr给标签添加和修改属性，  
如果之前没有该属性则是添加，如果有则是修改  
我们也可以通过css添加一些css属性，这个时候，标签的属性里会多一个style属性

[复制代码](javascript:void(0);)

html = '''

<div class="wrap">

<div id="container">

<ul class="list">

<li class="item-0">first item</li>

<li class="item-1"><a href="link2.html">second item</a></li>

<li class="item-0 active"><a href="link3.html"><span class="bold">third item</span></a></li>

<li class="item-1 active"><a href="link4.html">fourth item</a></li>

<li class="item-0"><a href="link5.html">fifth item</a></li>

</ul>

</div>

</div>

'''

from pyquery import PyQuery as pq

doc = pq(html)

li = doc('.item-0.active')

print(li)

li.attr('name', 'link')

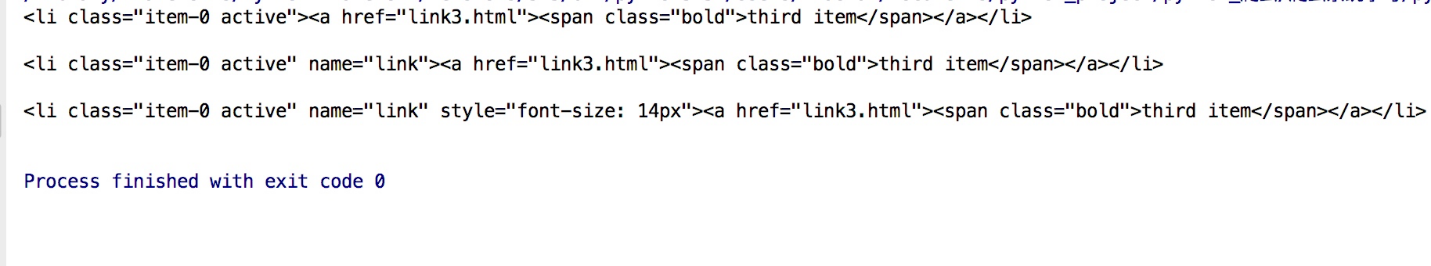
print(li)

li.css('font-size', '14px')

print(li)

[复制代码](javascript:void(0);)

结果如下：



**remove**  
有时候我们获取文本信息的时候可能并列的会有一些其他标签干扰，这个时候通过remove就可以将无用的或者干扰的标签直接删除，从而方便操作

[复制代码](javascript:void(0);)

html = '''

<div class="wrap">

Hello, World

<p>This is a paragraph.</p>

</div>

'''

from pyquery import PyQuery as pq

doc = pq(html)

wrap = doc('.wrap')

print(wrap.text())

wrap.find('p').remove()

print(wrap.text())

[复制代码](javascript:void(0);)

结果如下：

