免费代理IP网站：http://www.goubanjia.com
彼岸图片网：https://pic.netbian.com
中文教程：http://www.w3school.com.cn/xpath/index.asp

xpath解析

什么是xpath

XPath 是一门在 XML 文档中查找信息的语言。XPath 用于在 XML 文档中通过元素和属性进行导航，XPath 使用路径表达式来选取 XML 文档中的节点或者节点集
XPath 包含一个标准函数库
XPath 是 XSLT 中的主要元素
XPath 是一个 W3C 标准

XPath语法

在 XPath 中，有七种类型的节点：元素、属性、文本、命名空间、处理指令、注释以及文档节点（或称为根节点）。

节点选择

操作符或表达式	含义
/	从根节点开始找
//	从当前节点开始的任意层找
.	当前节点
..	当前结点的父节点

@	选择属性
节点名	选取所有这个节点名的节点
*	匹配任意元素节点
@*	匹配任意属性节点
node()	匹配任意类型的节点
text()	匹配text类型节点

实例

在下面的表格中，我们列出了一些路径表达式，以及这些表达式的结果：

路径表达式	结果
/bookstore/*	选取 bookstore 元素的所有子
//*	选取文档中的所有元素。
//title[@*]	选取所有带有属性的 title 元素

选取若干路径

通过在路径表达式中使用"|"运算符，您可以选取若干个路径。

实例

在下面的表格中，我们列出了一些路径表达式，以及这些表达式的结果：

路径表达式	结果
//book/title | //book/price	选取 book 元素的所有
//title | //price	选取文档中的所有 title
/bookstore/book/title | //price	选取属于 bookstore 元 title 元素，以及文档中

谓语(Predicates)

1. 谓语用来查找某个特定的节点或者包含某个指定的值的节点。
2. **谓语被嵌在方括号中。**
3. 谓语就是查询的条件。
4. 即在路径选择时，在中括号内指定查询条件。

实例

在下面的表格中，我们列出了带有谓语的一些路径表达式，以及表达式的结果：

路径表达式	结果
/bookstore/book[1]	选取属于 bookstore 子 素。
/bookstore/book[last()]	选取属于 bookstore 子 素。
/bookstore/book[last()-1]	选取属于 bookstore 子 元素。
/bookstore/book[position()<3]	选取最前面的两个属于 的 book 元素。
//title[@lang]	选取所有拥有名为 lang
//title[@lang='eng']	选取所有 title 元素，且 lang 属性。
/bookstore/book[price>35.00]	选取 bookstore 元素的 的 price 元素的值须大
/bookstore/book[price>35.00]/title	选取 bookstore 元素中 元素，且其中的 price

XPath轴(Axes)

轴可定义相对于当前节点的节点集。

轴名称	结果
ancestor	选取当前节点的所有先辈
ancestor-or-self	选取当前节点的所有先辈 点本身。
attribute	选取当前节点的所有属性。
child	选取当前节点的所有子元素
descendant	选取当前节点的所有后代元
descendant-or-self	选取当前节点的所有后代元 节点本身。
following	选取文档中当前节点的结束
namespace	选取当前节点的所有命名空
parent	选取当前节点的父节点。
preceding	选取文档中当前节点的开始
preceding-sibling	选取当前节点之前的所有同
self	选取当前节点。

位置路径表达式

位置路径可以是绝对的，也可以是相对的。

绝对路径起始于正斜杠(/)，而相对路径不会这样。在两种情况中，位置路径均包括一个或多个步，每个步均被斜杠分割：

绝对位置路径：

/step/step/...

相对位置路径：

step/step/...

每个步均根据当前节点集之中的节点来进行计算。

步（step）包括：

轴（axis）

定义所选节点与当前节点之间的树关系

节点测试（node-test）

识别某个轴内部的节点

零个或者更多谓语（predicate）

更深入地提炼所选的节点集

步的语法：

轴名称::节点测试[谓语]

实例

例子	结果
child::book	选取所有属于当前节点的子
attribute::lang	选取当前节点的 lang 属性
child::*	选取当前节点的所有子元素
attribute::*	选取当前节点的所有属性值
child::text()	选取当前节点的所有文本子
child::node()	选取当前节点的所有子节点
descendant::book	选取当前节点的所有 book
ancestor::book	选择当前节点的所有 book
ancestor-or-self::book	选取当前节点的所有 book 此节点是 book 节点)
child::*/child::price	选取当前节点的所有 price

常用函数总结

函数	含义
local-name()	获取不带限定名的名称。相当于指定标签元素
text()	获取标签之间的文本内容
node()	所有节点。
contains(@class,str)	包含
starts-with(local-name(),"book")	以book开头
last()	最后一个元素索引
position()	元素索引

xpath解析基本方法

- 安装:
    - pip install lxml
- 导包
    - from lxml import etree
- xpath解析原理:
    - 1. 实例化一个etree对象，将页面数据加载到该对象中
    - 2. 使用该对象的方法结合着xpath表达式进行数据解析
- xpath数据解析的方式
    - 本地加载
        - tree = etree.parse('./test_page.html')
    - 网络加载
        - tree = etree.HTML(page_text)

```
/: 从标签开始实现层级定位
//: 从任意位置实现标签的定位

属性定位:
 语法: tag[@attrName="attrValue"]
 //div[@class="song"] # 找到class属性值为song的div标签
层级&索引定位:
 # 注意索引值是从1开始
 //div[@class="tang"]/ul/li[2]/a # 找到class属性值为tang的div的直系子标签ul下的第二个子标签li下的直系子标签a
```

```
逻辑运算：
 //a[@href="" and @class="du"] # 找到href属性值为空且class属性值为du的a标签
模糊匹配：
 //div[contains(@class, "ng")]
 //div[starts-with(@class, "ta")]
取文本：
 //div[@class="song"]/p[1]/text() # 取直系文本内容
 //div[@class="tang"]//text() # 取所有文本内容
取属性：
 //div[@class="tang"]//li[2]/a/@href
```

用法举例，所用到的HTML文件是正则、bs4解析章节的test_page.html测试页面，jupyter-notebook中写的，所以无print即可输出

```
from lxml import etree

tree = etree.parse('./test_page.html')
tree.xpath('//div[@class="tang"]')
tree.xpath('//div[@class="tang"]/ul')
tree.xpath('//div[@class="tang"]/ul/li')
tree.xpath('//div[@class="tang"]/ul/li[3]')
tree.xpath('//div[@class="tang"]/ul/li[3]/a')
# 取属性
tree.xpath('//div[@class="tang"]/ul/li[3]/a/@href')[0]
# 取文本
tree.xpath('//div[@class="tang"]/ul/li[3]/a/text()')[0]
# 取度密月
tree.xpath('//div[@class="tang"]/ul/li[7]/i/text()')[0]
tree.xpath('//div[@class="song"]/img/@src')[0]

tree.xpath('//div[@class="tang"]/ul/li[position()<4]/preceding-sibling::li/a/@href')
tree.xpath('//div[@class="tang"]/ul/li[3]/preceding-sibling::li/a/@href')
tree.xpath('//div[@class="tang"]/ul/li[3]/child::a/@href')
```

爬取58二手房上面的房屋信息标题和价格， 还有详情页的概况

```
# 爬取58二手房上面的房屋信息标题和价格， 还有详情页的概况
import requests
from lxml import etree

```

```
5
6 home_url = "https://sz.58.com/ershoufang/?PGTID=0d30000c-0000-42b6-8f83-4e6db7977c08&ClickID=1"
7 headers = {
8  "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/76.0.3809.132 Safari/537.36"
9 }
10
11 # 向首页发送请求，获取房屋列表数据
12 page_text = requests.get(url=home_url, headers=headers).text
13
14 # 实例化一个etree对象，进行数据解析
15 tree = etree.HTML(page_text)
16 li_list = tree.xpath('//ul[@class="house-list-wrap"]/li')
17
18 all_data_list = list()
19 for li in li_list:
20  title = li.xpath('./div[2]/h2/a/text()')[0]
21  detail_url = li.xpath('./div[2]/h2/a/@href')[0]
22  price = "".join(li.xpath('./div[3]//text()')).strip()
23
24  # 向详情页的URL发送请求，获取房屋概况
25  detail_page_text = requests.get(url=detail_url, headers=headers).text
26  detail_tree = etree.HTML(detail_page_text)
27  content_list = detail_tree.xpath('//div[@id="generalSituation"]//text()')
28
29  content = "".join(content_list).strip()
30
31  dic = {
32  "title": title,
33  "detail_url": detail_url,
34  "price": price,
35  "content": content
36  }
37
38  all_data_list.append(dic)
39
40 print(all_data_list)
```

解析出所有城市名称https://www.aqistudy.cn/historydata/

```python
# 解析出所有城市名称https://www.aqistudy.cn/historydata/

import requests
from lxml import etree

url = "https://www.aqistudy.cn/historydata/"
headers = {
    "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/76.0.3809.132 Safari/537.36"
}

page_text = requests.get(url=url,headers=headers)
page_text.encoding = 'utf-8'
page_text = page_text.text

tree = etree.HTML(page_text)
citys = tree.xpath('//div[@class="all"]/div[2]/ul/div[2]/li/a/text()')
print(citys)
```