XPath 语法

XPath 使用路径表达式来选取 XML 文档中的节点或节点集。节点是通过沿着路径 (path) 或者步 (steps) 来选取的。

XML 实例文档

我们将在下面的例子中使用这个 XML 文档。

实例

```
<?xml version="1.0" encoding="UTF-8"?> <bookstore> <book>  <title lang="eng">Harry Potter</title>  <price>29.99</price></book> <book>  <title lang="eng">Learning XML</title>  <price>39.95</price></book> </bookstore>
```

选取节点

XPath 使用路径表达式在 XML 文档中选取节点。节点是通过沿着路径或者 step 来选取的。 下面列出了最有用的路径表达式：

表达式	描述
nodename	选取此节点的所有子节点。
/	从根节点选取。
//	从匹配选择的当前节点选择文档中的节点，而不考虑它们的位置。
.	选取当前节点。
..	选取当前节点的父节点。
@	选取属性。

在下面的表格中，我们已列出了一些路径表达式以及表达式的结果：

路径表达式	结果
bookstore	选取 bookstore 元素的所有子节点。
/bookstore	选取根元素 bookstore。 注释：假如路径起始于正斜杠(/)，则此路径始终代表到某元素的绝对路径！
bookstore/book	选取属于 bookstore 的子元素的所有 book 元素。
//book	选取所有 book 子元素，而不管它们在文档中的位置。
bookstore//book	选择属于 bookstore 元素的后代的所有 book 元素，而不管它们位于 bookstore 之下的什么位置。
//@lang	选取名为 lang 的所有属性。

谓语（Predicates）

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。

谓语被嵌在方括号中。

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 子元素的第一个 book 元素。
/bookstore/book[last()]	选取属于 bookstore 子元素的最后一个 book 元素。
/bookstore/book[last()-1]	选取属于 bookstore 子元素的倒数第二个 book 元素。
/bookstore/book[position()<3]	选取最前面的两个属于 bookstore 元素的子元素的 book 元素。
//title[@lang]	选取所有拥有名为 lang 的属性的 title 元素。
//title[@lang='eng']	选取所有 title 元素，且这些元素拥有值为 eng 的 lang 属性。
/bookstore/book[price>35.00]	选取 bookstore 元素的所有 book 元素，且其中的 price 元素的值须大于 35.00。
/bookstore/book[price>35.00]//title	选取 bookstore 元素中的 book 元素的所有 title 元素，且其中的 price 元素的值须大于 35.00。

选取未知节点

XPath 通配符可用来选取未知的 XML 元素。

通配符	描述

*	匹配任何元素节点。
@*	匹配任何属性节点。
node()	匹配任何类型的节点。

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 元素的所有子元素。
//*	选取文档中的所有元素。
//title[@*]	选取所有带有属性的 title 元素。

contains、starts-with、*ends-with、text()、and、or、not

//a[contains(text(),"下一页")]或//a[text()='百度搜索']
//input[@type='submit' and @name='calc']
//input[starts-with(@id,'calc')]
//input[not(@type="input")]
ends-with不匹配可以使用
//input[substring(@type, string-length(@type) - string-length('t') +1) = 't']

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。

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 和 price 元素。
//title | //price	选取文档中的所有 title 和 price 元素。
/bookstore/book/title | //price	选取属于 bookstore 元素的 book 元素的所有 title 元素，以及文档中所有的 price 元素。

XPath 轴（Axes）

XML 实例文档

我们将在下面的例子中使用此 XML 文档：

```
<?xml version="1.0" encoding="UTF-8"?> <bookstore> <book>  <title lang="en">Harry Potter</title>  <price>29.99</pr
ice></book> <book>  <title lang="en">Learning XML</title>  <price>39.95</price></book> </bookstore>
```

XPath 轴（Axes）

html/body/div/form/ancestor::*

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（父、祖父等）。
ancestor-or-self	选取当前节点的所有先辈（父、祖父等）以及当前节点本身。
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素（子、孙等）。
descendant-or-self	选取当前节点的所有后代元素（子、孙等）以及当前节点本身。
following	选取文档中当前节点的结束标签之后的所有节点。
following-sibling	选取当前节点之后的所有兄弟节点
namespace	选取当前节点的所有命名空间节点。
parent	选取当前节点的父节点。
preceding	选取文档中当前节点的开始标签之前的所有节点。

preceding-sibling	选取当前节点之前的所有同级节点。
self	选取当前节点。

XPath 运算符

XPath 表达式可返回节点集、字符串、逻辑值以及数字。

XPath 运算符

下面列出了可用在 XPath 表达式中的运算符：

运算符	描述	实例	返回值
|	计算两个节点集	//book | //cd	返回所有拥有 book 和 cd 元素的节点集
+	加法	6 + 4	10
-	减法	6 - 4	2
*	乘法	6 * 4	24
div	除法	8 div 4	2
=	等于	price=9.80	如果 price 是 9.80，则返回 true。 如果 price 是 9.90，则返回 false。
!=	不等于	price!=9.80	如果 price 是 9.90，则返回 true。 如果 price 是 9.80，则返回 false。
<	小于	price<9.80	如果 price 是 9.00，则返回 true。 如果 price 是 9.90，则返回 false。
<=	小于或等于	price<=9.80	如果 price 是 9.00，则返回 true。 如果 price 是 9.90，则返回 false。
>	大于	price>9.80	如果 price 是 9.90，则返回 true。 如果 price 是 9.80，则返回 false。
>=	大于或等于	price>=9.80	如果 price 是 9.90，则返回 true。 如果 price 是 9.70，则返回 false。
or	或	price=9.80 or price=9.70	如果 price 是 9.80，则返回 true。 如果 price 是 9.50，则返回 false。
and	与	price>9.00 and price<9.90	如果 price 是 9.80，则返回 true。 如果 price 是 8.50，则返回 false。
mod	计算除法的余数	5 mod 2	1

html（房屋结构）+css（软装）

CSS（房屋格局+软装）+HTML（柜子里面的东西）

CSS 选择器

CSS选择器用于选择你想要的元素的样式的模式。

"CSS"列表示在CSS版本的属性定义（CSS1，CSS2，或对CSS3）。

选择器	示例	示例说明	CSS
.class	.intro	选择所有class="intro"的元素	1
#id	#firstname	选择所有id="firstname"的元素	1
*	*	选择所有元素	2
element	p	选择所有<p>元素	1
element,element	div,p	选择所有 元素和<p>元素	1
element element	div p	选择 元素内的所有<p>元素	1
element>element	div>p	选择所有父级是 元素的 <p> 元素	2
element+element	div+p	选择所有紧接着 元素之后的<p>元素	2
[*attribute*]	[target]	选择所有带有target属性元素	2

[*attribute=value*]	[target=-blank]	选择所有使用target="-blank"的元素	2
[*attribute~=value*]	[title~=flower]	选择标题属性包含单词"flower"的所有元素	2
[*attribute|=language*]	[lang|=en]	选择 lang 属性以 en 为开头的所有元素	2
:link	a:link	选择所有未访问链接	1
:visited	a:visited	选择所有访问过的链接	1
:active	a:active	选择活动链接	1
:hover	a:hover	选择鼠标在链接上面时	1
:focus	input:focus	选择具有焦点的输入元素	2
:first-letter	p:first-letter	选择每一个<p>元素的第一个字母	1
:first-line	p:first-line	选择每一个<p>元素的第一行	1
:first-child	p:first-child	指定只有当<p>元素是其父级的第一个子级的样式。	2
:before	p:before	在每个<p>元素之前插入内容	2
:after	p:after	在每个<p>元素之后插入内容	2
:lang(*language*)	p:lang(it)	选择一个lang属性的起始值="it"的所有<p>元素	2
element1~element2	p~ul	选择p元素之后的每一个ul元素	3
[*attribute^=value*]	a[src^="https"]	选择每一个src属性的值以"https"开头的元素	3
[*attribute$=value*]	a[src$=".pdf"]	选择每一个src属性的值以".pdf"结尾的元素	3
[*attribute*=value*]	a[src*="runoob"]	选择每一个src属性的值包含子字符串"runoob"的元素	3
:first-of-type	p:first-of-type	选择每个p元素是其父级的第一个p元素	3
:last-of-type	p:last-of-type	选择每个p元素是其父级的最后一个p元素	3
:only-of-type	p:only-of-type	选择每个p元素是其父级的唯一p元素	3
:only-child	p:only-child	选择每个p元素是其父级的唯一子元素	3
:nth-child(*n*)	p:nth-child(2)	选择每个p元素是其父级的第二个子元素	3
:nth-last-child(*n*)	p:nth-last-child(2)	选择每个p元素的是其父级的第二个子元素，从最后一个子项计数	3
:nth-of-type(*n*)	p:nth-of-type(2)	选择每个p元素是其父级的第二个p元素	3
:nth-last-of-type(*n*)	p:nth-last-of-type(2)	选择每个p元素的是其父级的第二个p元素，从最后一个子项计数	3
:last-child	p:last-child	选择每个p元素是其父级的最后一个子级。	3
:root	:root	选择文档的根元素	3
:empty	p:empty	选择每个没有任何子级的p元素（包括文本节点）	3
:target	#news:target	选择当前活动的#news元素（包含该锚名称的点击的URL）	3
:enabled	input:enabled	选择每一个已启用的输入元素	3
:disabled	input:disabled	选择每一个禁用的输入元素	3
:checked	input:checked	选择每个选中的输入元素	3
:not(*selector*)	:not(p)	选择每个并非p元素的元素	3
::selection	::selection	匹配元素中被用户选中或处于高亮状态的部分	3
:out-of-range	:out-of-range	匹配值在指定区间之外的input元素	3
:in-range	:in-range	匹配值在指定区间之内的input元素	3
:read-write	:read-write	用于匹配可读及可写的元素	3
:read-only	:read-only	用于匹配设置 "readonly"（只读） 属性的元素	3
:optional	:optional	用于匹配可选的输入元素	3
:required	:required	用于匹配设置了 "required" 属性的元素	3

:valid	:valid	用于匹配输入值为合法的元素	3
:invalid	:invalid	用于匹配输入值为非法的元素	3

1.class属性唯一但是有空格，选择空格两边唯一的哪一个

<th id="" tid="grid-row-2" class="tabth field_security_dispay txt-left ">

2.class属性科普

class属性中间的空格并不是空字符串，那是间隔符号，表示的是一个元素有多个class的属性名称，class属性是比较特殊的一个，除了class这个元素类型有多个属性外，其他的像name，id是没多个属性的。

3.class的定位

既然知道class属性有空格就是有多个属性了，那么定位的时候取其中一个就行（并且要确定他的唯一性），也就是说 class="dtb-style-1 table-dragColumns"这两个属性选其中一个就行，取dtb-style-1 or table-dragColumn都是可以的，只要想办法保持唯一性就行。

4.判断元素唯一性

Chrome F12切换到HTML界面，Ctrl + f 出现搜索框然后输入关键字如：然后回车，如图所示：

5.class属性不唯一怎么办

如果这个class的多个属性都不唯一怎么办？元素不唯一不要慌，可以用复数定位，把所有相同元素定位出来，按下表取第几个就行。
如：

```
self.driver.find_elements_by_class_name('table-dragColumn')[0].click()
```

6.css定位

css定位class多个属性时要给元素前面加个点（.）就行，然后空格变成了点（.）就能定位了
当然css也可以取class属性的其中一个属性（保证唯一性）来定位，定位方法是灵活多变的。
如：
css定位：

```
    .dtb-style-1.table-dragColumns
    //#前面加（.）空格地方用点（.）来代替
```

class单个属性定位

```
    .table-dragColumns
    //#用单个属性来定位前面加个（.）
```

直接包含空格的css定位神器

```
    class="dtb-style-1 table-dragColumns"
    //#包含整个类
```
