**Selenium Webdriver元素定位的八种常用方式**

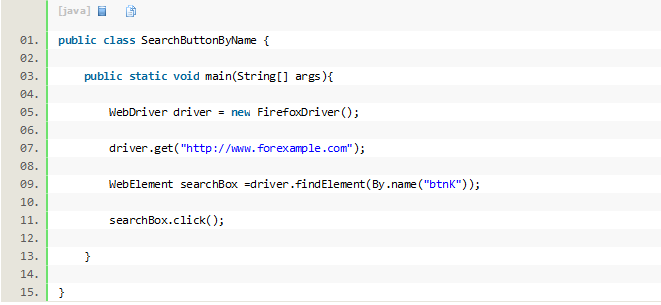
在使用selenium webdriver进行元素定位时，通常使用findElement或findElements方法结合By类返回的元素句柄来定位元素。其中By类的常用定位方式共八种，现分别介绍如下。

1. By.name()

页面源码如下：



当我们要用name属性来引用这个button并点击它时，代码如下：



2. By.id()

页面源码如下：



要引用该button并点击它时，代码如下：

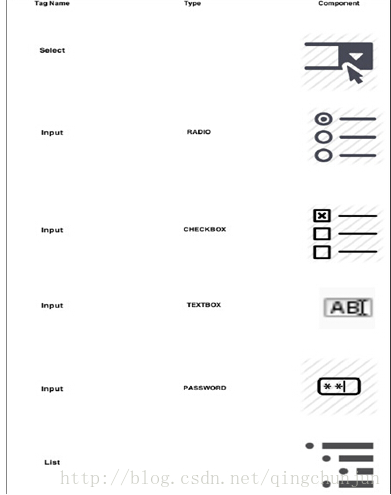


3. By.tagName()

该方法可以通过元素的标签名称来查找元素。该方法跟之前两个方法的区别是，这个方法搜索到的元素通常不止一个，所以一般建议结合使用findElements方法来使用。比如我们现在要查找页面上有多少个button，就可以用button这个tagName来进行查找，代码如下：



另外，在使用tagName方法进行定位时，还有一个地方需要注意的是，通常有些HTML元素的tagName是相同的，如下图(1)所示。



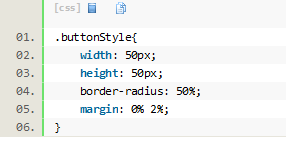
图（1）

从图中我们可以看到，单选框、复选框、文本框和密码框的元素标签都是input，此时单靠tagName无法准确地得到我们想要的元素，需要结合type属性才能过滤出我们要的元素。示例代码如下：



4. By.className()

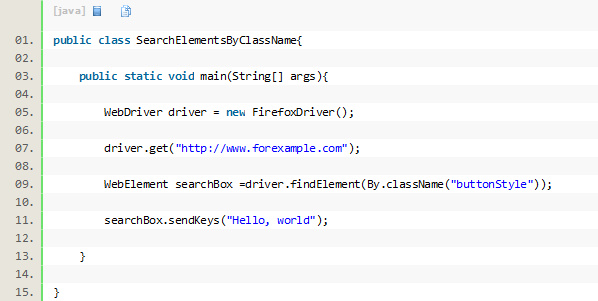
className属性是利用元素的css样式表所引用的伪类名称来进行元素查找的方法。对于任何HTML页面的元素来说，一般程序员或页面设计师会给元素直接赋予一个样式属性或者利用css文件里的伪类来定义元素样式，使元素在页面上显示时能够更加美观。一般css样式表可能会长成下面这个样子：



定义好后，就可以在页面元素中引用上述定义好的样式，如下：



如果此时我们要通过className属性来查找该button并操作它的话，就可以使用className属性了，代码如下：



注意：使用className来进行元素定位时，有时会碰到一个元素指定了若干个class属性值的“复合样式”的情况，如下面这个button：

这个button元素指定了三个不同的css伪类名作为它的样式属性值，此时就必须结合后面要介绍的cssSelector方法来定位了，稍后会有详细例子。

5. By.linkText()

这个方法比较直接，即通过超文本链接上的文字信息来定位元素，这种方式一般专门用于定位页面上的超文本链接。通常一个超文本链接会长成这个样子：<a href="/intl/en/about.html">About Google</a>。我们定位这个元素时，可以使用下面的代码进行操作：

6. By.partialLinkText()

这个方法是上一个方法的扩展。当你不能准确知道超链接上的文本信息或者只想通过一些关键字进行匹配时，可以使用这个方法来通过部分链接文字进行匹配。代码如下：



注意：使用这种方法进行定位时，可能会引起的问题是，当你的页面中不止一个超链接包含About时，findElement方法只会返回第一个查找到的元素，而不会返回所有符合条件的元素。如果你要想获得所有符合条件的元素，还是只能使用findElements方法。

7. By.xpath()

这个方法是非常强大的元素查找方式，使用这种方法几乎可以定位到页面上的任意元素。在正式开始使用XPath进行定位前，我们先了解下什么是XPath。XPath是XML Path的简称，由于HTML文档本身就是一个标准的XML页面，所以我们可以使用XPath的语法来定位页面元素。

假设我们现在以图(2)所示HTML代码为例，要引用对应的对象，XPath语法如下：



图（2）

绝对路径写法(只有一种)，写法如下：

引用页面上的form元素(即源码中的第3行)：/html/body/form[1]

这里要注意以下几点：

1. 元素的xpath绝对路径可通过firebug直接查询，一般不需要你手工去写。

2. 一般不推荐使用绝对路径的写法，因为一旦页面结构发生变化，该路径也随之失效，必须重新写。

3. 绝对路径以单/号表示，而下面要讲的相对路径则以//表示，这个区别非常重要。另外需要多说一句的是，当xpath的路径以/开头时，表示让Xpath解析引擎从文档的根节点开始解析。当xpath路径以//开头时，则表示让xpath引擎从文档的任意符合的元素节点开始进行解析。而当/出现在xpath路径中时，则表示寻找父节点的直接子节点，当//出现在xpath路径中时，表示寻找父节点下任意符合条件的子节点，不管嵌套了多少层级（这些下面都有例子，大家可以参照来试验）。弄清这个原则，就可以理解其实xpath的路径可以绝对路径和相对路径混合在一起来进行表示，想怎么玩就怎么玩。

下面是相对路径的引用写法：

查找页面根元素：//

查找页面上所有的input元素：//input

查找页面上第一个form元素内的直接子input元素(即只包括form元素的下一级input元素，使用绝对路径表示，单/号)：//form[1]/input

查找页面上第一个form元素内的所有子input元素(只要在form元素内的input都算，不管还嵌套了多少个其他标签，使用相对路径表示，双//号)：//form[1]//input

查找页面上第一个form元素内的最后一个input元素：//form[1]//input[last()]（注意：在xpath中没有first()这样的方法，用//form[1]//input[1]这种表示法就可以取第一个）

查找页面上第一个form元素：//form[1]

查找页面上id为loginForm的form元素：//form[@id='loginForm']

查找页面上具有name属性为username的input元素：//input[@name='username']

查找页面上id为loginForm的form元素下的第一个input元素：//form[@id='loginForm']/input[1]（注意：在xpath中没有第0元素这样的表示方法，都是从1开始）

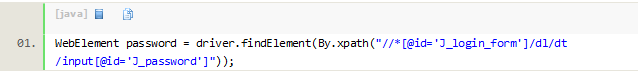
查找页面具有name属性为contiune并且type属性为button的input元素：//input[@name='continue'][@type='button']

查找页面上id为loginForm的form元素下第4个input元素：//form[@id='loginForm']/input[4]

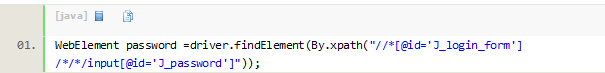
Xpath功能很强大，所以也可以写得更加复杂一些，如下面图（3）的HTML源码。

图（3）

如果我们现在要引用id为“J\_password”的input元素，该怎么写呢？我们可以像下面这样写：



也可以写成：



这里解释一下，其中//\*[@id=’J\_login\_form’]这一段是指在根元素下查找任意id为J\_login\_form的元素，此时相当于引用到了form元素。后面的路径必须按照源码的层级依次往下写。因为我们要找的input元素包含在一个dt标签内，而dt又包含在dl标签内，所以中间必须写上dl和dt两层，才到input这层。当然我们也可以用\*号省略具体的标签名称，但元素的层级关系必须体现出来，比如我们不能写成//\*[@id='J\_login\_form']/input[@id='J\_password']，这样肯定会报错的。

前面讲的都是xpath中基于准确元素属性的定位，其实xpath作为定位神器也可以用于模糊匹配。比如下面图（4）所示代码：



图（4）

这段代码中的“退出”这个超链接，没有标准id元素，只有一个rel和href，不是很好定位。不妨我们就用xpath的几种模糊匹配模式来定位它吧，主要有三种方式，举例如下。

a. 用contains关键字，定位代码如下：



这句话的意思是寻找页面中href属性值包含有logout这个单词的所有a元素，由于这个退出按钮的href属性里肯定会包含logout，所以这种方式是可行的，也会经常用到。其中@后面可以跟该元素任意的属性名。

b. 用start-with，定位代码如下：



这句的意思是寻找rel属性以nofo开头的a元素。其中@后面的rel可以替换成元素的任意其他属性。

c. 用Text关键字，定位代码如下：



这个方法可谓相当霸气啊。直接查找页面当中所有的退出二字，根本就不用知道它是个a元素了。这种方法也经常用于纯文字的查找。

另外，如果知道超链接元素的文本内容，也可以用



这种方式一般用于知道超链接上显示的部分或全部文本信息时，可以使用。

最后，关于xpath这种定位方式，webdriver会将整个页面的所有元素进行扫描以定位我们所需要的元素，所以这是一个非常费时的操作，如果你的脚本中大量使用xpath做元素定位的话，将导致你的脚本执行速度大大降低，所以请慎用。

8. By.cssSelector()

cssSelector这种元素定位方式跟xpath比较类似，但执行速度较快，而且各种浏览器对它的支持都相当到位，所以功能也是蛮强大的。

下面是一些常见的cssSelector的定位方式：

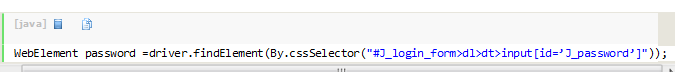
定位id为flrs的div元素，可以写成：#flrs 注：相当于xpath语法的//div[@id=’flrs’]

定位id为flrs下的a元素，可以写成 #flrs > a 注：相当于xpath语法的//div[@id=’flrs’]/a

定位id为flrs下的href属性值为/forexample/about.html的元素，可以写成： #flrs > a[href=”/forexample/about.html”]

如果需要指定多个属性值时，可以逐一加在后面，如#flrs> input[name=”username”][type=”text”]。

明白基本语法后，我们来尝试用cssSelector方式来引用图（3）中选中的那个input对象，代码如下：



同样必须注意层级关系，这个不能省略。

cssSelector还有一个用处是定位使用了复合样式表的元素，之前在第4种方式className里面提到过。现在我们就来看看如何通过cssSelector来引用到第4种方式中提到的那个button。button代码如下：



cssSelector引用元素代码如下：



这样就可以顺利引用到使用了复合样式的元素了。

此外，cssSelector还有一些高级用法，如果熟练后可以更加方便地帮助我们定位元素，如我们可以利用^用于匹配一个前缀，$用于匹配一个后缀，\*用于匹配任意字符。例如：

匹配一个有id属性，并且id属性是以”id\_prefix\_”开头的超链接元素：a[id^='id\_prefix\_']

匹配一个有id属性，并且id属性是以”\_id\_sufix”结尾的超链接元素：a[id$='\_id\_sufix']

匹配一个有id属性，并且id属性中包含”id\_pattern”字符的超链接元素：a[id\*='id\_pattern']

最后再总结一下，各种方式在选择的时候应该怎么选择：

1. 当页面元素有id属性时，最好尽量用id来定位。但由于现实项目中很多程序员其实写的代码并不规范，会缺少很多标准属性，这时就只有选择其他定位方法。

2. xpath很强悍，但定位性能不是很好，所以还是尽量少用。如果确实少数元素不好定位，可以选择xpath或cssSelector。

3. 当要定位一组元素相同元素时，可以考虑用tagName或name。

4. 当有链接需要定位时，可以考虑linkText或partialLinkText方式。

参考资料：

《Selenium Webdriver PracticalGuide》

https://saucelabs.com/resources/selenium/css-selectors