虽然最新的[IE9仍没支持XPath](http://blog.brett-zamir.me/?p=169)，IE上运行XPath[慢](http://fafeng.blogbus.com/logs/44472600.html)，但[JavaScript-XPath](http://coderepos.org/share/wiki/JavaScript-XPath)已经[被纳入Selenium](http://code.google.com/p/selenium/source/browse/trunk/javascript/selenium-core/xpath/javascript-xpath-0.1.11.js)。在Java中使用：

Java语言:

selenium = new DefaultSelenium(location, port, browser, targetPath);   
selenium.start();   
if(browserString=="\*iexplore")   
selenium.useXpathLibrary("javascript-xpath");  
也可以[手工修改selenium-browserbot.js](http://blog.altom.ro/2010/10/selenium-xpath-and-internet-explorer.html" \t "_blank)中默认的XPath库ajaxslt。

[XPath](http://zh.wikipedia.org/wiki/XPath)原用作确定XML文档中树节点信息，详见参考[XPath 1.0](http://www.w3.org/TR/xpath/)和[XPath 2.0](http://www.w3.org/TR/xpath20/)。使用见文[How to parse web pages using XPath](http://www.script-tutorials.com/how-to-parse-web-pages-using-xpath/" \t "_blank)。

Firefox有两个插件可以有效定位元素的XPath，一个是鼎鼎大名的[Firebug](http://getfirebug.com/)，以下是定位方法：

1.右键选中该元素，点击“查看元素”（Inspect element）；

2.Firebug中HTML标签显示该元素前端代码被选，在此标签内右键选择“复制XPath”（Copy XPath）。

这个XPath来自于DOM树经过了修正，实际使用时需要对照页面实际HTML代码校正。此外Firebug命令行控制台提供了[$x](http://getfirebug.com/wiki/index.php/Command_Line_API" \l ".24x.28xpath.29)(xpath)功能，可用于验证该XPath的返回值。

**技术背景**  
      在Web应用中，用户通过键盘在输入框中输入值和鼠标点击按钮，链接等。比如在用户名输入框和密码输入框输入正确的用户名和密码，然后点击登录按钮进行登录。在[Selenium自动化](http://www.sigma-rt.com.cn/product/sigmationtf/homepage.php" \t "_blank)中，Selenium提供多种API来对HTML元素进行操作，对于每个HTML元素，需要一个可以标识它的标识符，在Selenium中称之为定位器，Selenium支持多种不同类型的定位器，有标识符，Id，Name, DOM Locator，XPath Locator, 以及CSS Locator等。本文主要讨论DOM, XPATH,以及CSS定位器在不同的浏览器中的使用的优缺点以及注意事项。

**问题与挑战**  
      针对一个使用Selenium RC的[Web自动化](http://www.testingorg.net/special/autotesting.html" \t "_blank)项 目，在项目初期在Firefox上采用了比较灵活，简洁的XPath定位器来对应用中的对象进行操作。但在项目开展到1/3时，增加了对IE浏览器支持的 需求。在使用现有基于XPath的脚本运行后，发现脚本运行时间呈现几十上百倍地增加，在Firefox上执行花费1分钟的脚本在IE上会花费10～20 分钟，甚至更长的时间。该问题就导致了在IE上直接运行之前已完成脚本变得不可行。要了解到Firefox支持原生的Xpath解析功能，而IE不支持原 生的Xpath解析，而是基于一个外部的javascript library（Google's library）来进行Xpath解析。由于需求是要求脚本能够同时支持Firefox和IE浏览器，所以需要寻求另外一种方式来使得在不同浏览器上调用 Selenium API所执行时间相当。

**解决方案**      在Selenium RC 0.9.2中，用以解析XPath的javascript库在IE上执行的时间增加了脚本执行时间，而且在越复杂的页面中，在IE上调用API所占用的时 间越长。于是决定换用其它类型的定位器，由于该被测Web应用中大多数HTML元素未指定有ID和Name属性，所以在这里首先采用比较灵活的Dom定位 器进行[**测试**](http://www.sigma-rt.com.cn/)。

使用Dom定位器来对HTML元素定位，该Locator表达式需要以“dom=” 或者是“document.”开头的形式，Selenium会执行这段javascript片断来最终地取得我们需要访问的HTML元素，由此在Dom定 位器中，可以使用Web页面中的DOM对象来获取文档内的所有HTML元素。 需要注意一点的是，Selenium执行这段javascript片断的时候，是采用整段javascript脚本执行的值，也即是在这段 javascript片断中最后一个表达式的值。

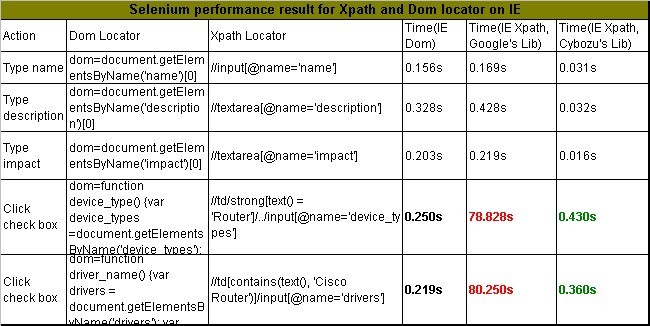
使用Dom定位器后，在IE上脚本执行的时间比使用Xpath的时候短很多，基本上跟在Firefox上面所花费的时候基本相当。

请参照以下的一个性能对照表：



    依照以上表格，不难发现，使用XPath定位器，能使得定位器比较简单，而对于比较复杂的对象（无IE和Name的动态对象）在使用Dom定位器则非常的复杂。由于XPath在IE下的效率极其低下，也只能使用Dom定位器来支持IE浏览器。

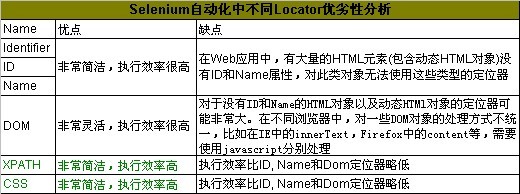
    在Selenium RC1.0.1版本发布后，Selenium增加了一个新的Xpath的javascript库：Cybozu Labs' faster library。



    由以上表格可见，在IE下使用了Cybozu Lab的XPath library后，执行效率有了很大提升，基本上可以与使用Dom定位器相当。通过比较，在新的项目中使用Selenium来进行Web自动化开发，使用 XPath定位器，可以使得定位器本生比较简洁，而且也得到较高执行效率。

    在Web开发中，有较多的人使用CSS来优化Web页面效果。而Selenium也支持CSS定位器，在IE和Firefox浏览器下，使用CSS定位器 时，执行的效率与XPath基本相当，而且CSS定位器同样与XPath比较简洁。所以对CSS比较熟悉的开发人员也可以使用CSS定位器来进行 Selenium [Web自动化](http://www.sigma-rt.com.cn/)开发。

   终上所述，在新的[Web自动化](http://www.testingorg.net/special/autotesting.html" \t "_blank)项目中推荐使用XPATH和CSS定位器，请参考下面的对照表格。



**环境设置描述**  
在需要运行Web浏览器的主机上运行Selenium RC服务，记主机1。  
Web[自动化](http://www.sigma-rt.com.cn/product/sigmationtf/homepage.php)程序可以运行在主机１上，也可以运行在任何一台可与主机１进行网络通信的其它主机上，

**使用工具简介**Selenium RC服务程序，需要使用JDK/JRE来运行Selenium RC服务。  
支持HTTP请求的程序语言，Perl, Ruby, Python, Java, C#等等。  
支持所有Javascript-based浏览器。

 在ie上运行selenium的测试程序时，如果使用的是xpath，会发现速度奇慢无比，   
例如这样一个测试

Java代码 [复制代码](http://www.iteye.com/topic/643709) [[收藏代码http://www.iteye.com/images/spinner.gif](http://blog.csdn.net/fhnp1983/article/details/6908749)](http://blog.csdn.net/fhnp1983/article/details/6908749)

1. assertTrue(selenium.isElementPresent("//div[@id='content']/div[1]/table/tbody/tr[2]/td[2]"));

**[java]** [view plaincopy](http://blog.csdn.net/fhnp1983/article/details/6908749)

1. assertTrue(selenium.isElementPresent("//div[@id='content']/div[1]/table/tbody/tr[2]/td[2]"));

在firefox上的时间是毫秒级的，但在ie上却要数十秒，经过查找资料，终于找到   
了解决办法：   
  
1. 更换默认的xpath库   
  除了ie，其他主要浏览器都是内置对xpath的支持的，但ie不行，所以selenium   
  使用了javascript库，默认使用的是ajaxslt，这个会比较慢，可以换成   
  javascript-xpath，  虽然比firefox还是慢，但也快多了，上面的例子只需要不   
  到1秒。换法很简单，如下：

Java代码 [复制代码](http://www.iteye.com/topic/643709) [[收藏代码http://www.iteye.com/images/spinner.gif](http://blog.csdn.net/fhnp1983/article/details/6908749)](http://blog.csdn.net/fhnp1983/article/details/6908749)

1. selenium = new DefaultSelenium(location, port, browser, targetPath);
2. selenium.start();
3. selenium.useXpathLibrary("javascript-xpath");

**[java]** [view plaincopy](http://blog.csdn.net/fhnp1983/article/details/6908749)

1. selenium = new DefaultSelenium(location, port, browser, targetPath);
2. selenium.start();
3. selenium.useXpathLibrary("javascript-xpath");

2. 写xpath时，尽量从一个具有id的元素开始，这样也可以大大提高执行速度，例如   
  如果上面的测试写成下面这样，运行时间就会变成几秒了。