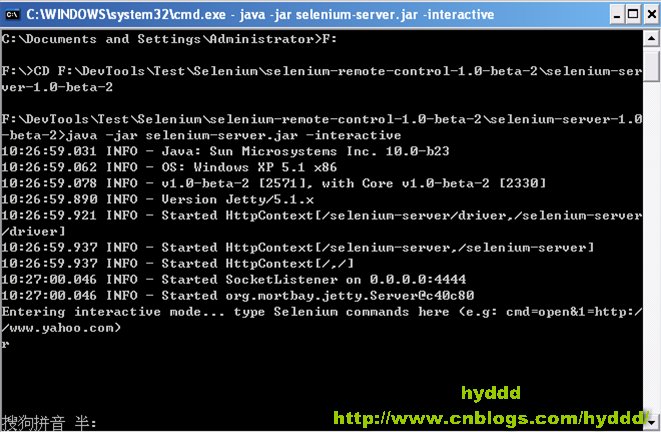
[Selenium私房菜系列8 -- 玩转Selenium Server](http://www.cnblogs.com/hyddd/archive/2009/05/30/1492492.html)

2009-05-30 22:26 by hyddd, 19058 阅读, 9 评论, [收藏](http://www.cnblogs.com/hyddd/archive/2009/05/30/1492492.html), [编辑](https://i.cnblogs.com/EditPosts.aspx?postid=1492492)

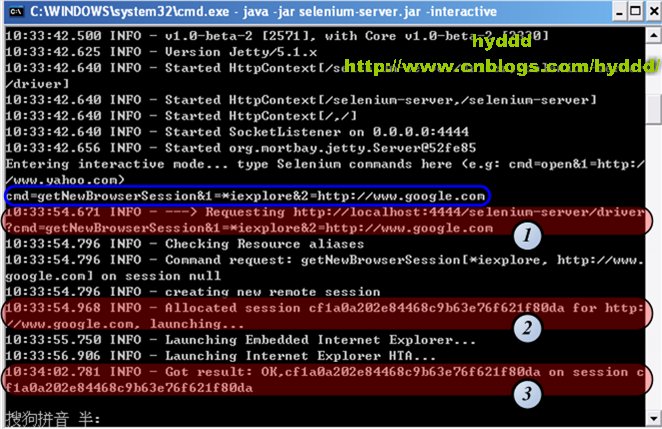
    本篇主要是想更进一步介绍Selenium Server的工作原理，这次我们从Selenium Server的**交互模式**开始。

    在《[第一个Selenium RC测试案例](http://www.cnblogs.com/hyddd/archive/2009/05/24/1488377.html)》中，我们以命令“java -jar selenium-server.jar”启动了Selenium Server，其实在启动Selenium Server时，我们还可以加上各种参数（具体的参数请参考《[Selenium RC服务器命令行参数列表](http://www.cnblogs.com/hyddd/archive/2009/05/30/1492511.html)》）， 而开启**Selenium Server交互模式**的命令为“**java -jar selenium-server.jar -interactive**”。交互模式，是Selenium Server提供的一种快速的测试方法，你可以对Selenium Server输入命令从而直接启动测试。

**1.启动Selenium Server交互模式**



**2.在命令行中输入：cmd=getNewBrowserSession&1=\*iexplore&2=http://www.google.com**。控制Selenium Server启动浏览器，以及创建Session。

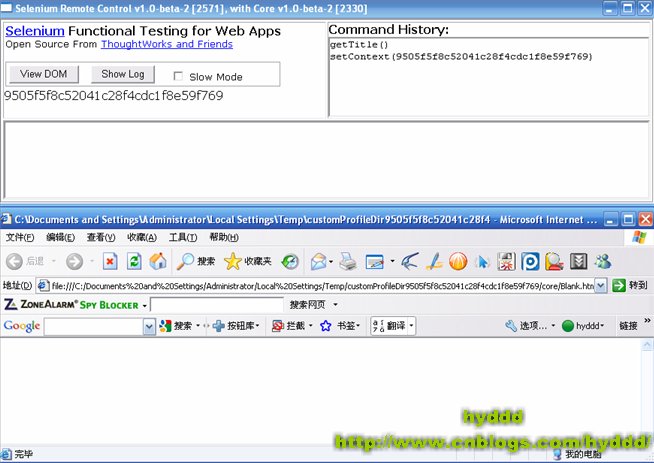


**（1）**.---> Requesting [http://localhost:4444/selenium-server/driver?cmd=getNewBrowserSession&1=\*iexplore&2=http://www.google.com](http://localhost:4444/selenium-server/driver?cmd=getNewBrowserSession&1=*iexplore&2=http://www.google.com" \o "http://localhost:4444/selenium-server/driver?cmd=getNewBrowserSession&1=*iexplore&2=http://www.google.com)

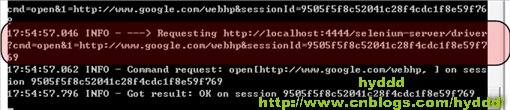
看过《[深入了解Selenium RC工作原理(1)](http://www.cnblogs.com/hyddd/archive/2009/05/30/1492213.html)》的应该了解：我们所编写的测试案例，其实是通过发送Http请求实现对Selenium Server的控制，而测试案例所发送的请求就正是：---> Requesting [http://localhost:4444/selenium-server/driver?cmd=getNewBrowserSession&1=\*iexplore&2=http://www.google.com](http://localhost:4444/selenium-server/driver?cmd=getNewBrowserSession&1=*iexplore&2=http://www.google.com)。我们可以再打开一个IE浏览器，在地址栏输入：[http://localhost:4444/selenium-server/driver?cmd=getNewBrowserSession&1=\*iexplore&2=http://www.google.com](http://localhost:4444/selenium-server/driver?cmd=getNewBrowserSession&1=*iexplore&2=http://www.google.com" \o "http://localhost:4444/selenium-server/driver?cmd=getNewBrowserSession&1=*iexplore&2=http://www.google.com)，回车！看，Selenium Server又为此产生了一个Session了！呵呵:>

**（2）**.这里，Selenium Server为上面的请求随机生成了一个Session ID：9505f5f8c52041c28f4cdc1f8e59f769（由于写这篇文章的时候中途重启了Selenium Server，所以这里和上图的Session ID不同，并且下文会继续使用Session ID：9505f5f8c52041c28f4cdc1f8e59f769）。

**（3）**.如果一切正常，Selenium Server最后会出现Get Result Ok的字样，并出现如下两个框框：



**3.控制浏览器访问[www.google.com](http://www.google.com/)/webhp，输入：cmd=open&1=http://www.google.com/webhp&sessionId=9505f5f8c52041c28f4cdc1f8e59f769**



噢，浏览器成功访问<http://www.google.com/webhp>了:>。

**总结一下：**

**（1）**.在Selenium Server中输入命令的格式为：**cmd=Command&1=Target&2=Value&SessionID=…**，这和Selenium IDE的案例语句很像。

**（2）**.在输入命令后，Selenium Server会发条Http请求给自己，请求的URL格式也是固定的：**[http://localhost:4444/selenium-server/driver?cmd=Command&1=Target&2=Value&SessionID=…](http://localhost:4444/selenium-server/driver?cmd=Command&1=Target&2=Value&SessionID" \o "http://localhost:4444/selenium-server/driver?cmd=getNewBrowserSession&1=*iexplore&2=http://www.google.com)**，我们完全可以用浏览器发送请求控制Selenium Server进行测试。

**（3）**.另外，sessionId是很重要的一个参数，当一个Selenium Server同时运行多个测试案例时，Selenium Server就是通过sessionId判断到底该操作哪个浏览器窗口。而在下面的C#代码中：

ISelenium selenium = new DefaultSelenium("127.0.0.1", 4444, "\*iexplore", "http://www.google.com");

selenium.Start();

selenium.Open("/webhp");

selenium就相当于上文中的sessionId。

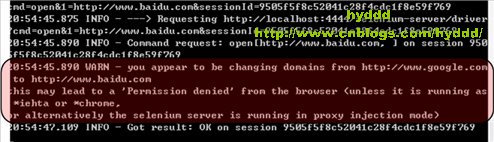
（4）.在Selenium Server启动一个Session时，必须先指定一个 “源”(原因见《[深入了解Selenium RC工作原理(2)》)，](http://www.cnblogs.com/hyddd/archive/2009/05/30/1492289.html)在上面的代码中[http://www.google.com](http://www.google.com/)就是“源”了，然而这是可能出现问题，请看下面代码：

ISelenium selenium = new DefaultSelenium("127.0.0.1", 4444, "\*iexplore", "http://www.google.com");

selenium.Start();

selenium.Open([http://www.baidu.com](http://www.baidu.com/));

我们在启动Session时，定义了源为[http://www.google.com](http://www.google.com/)，但在后来的操作中，我们打开的却是[http://www.baidu.com](http://www.baidu.com/)，由于二者非同源，所以接下来的操作就可能会出现各种问题，故此Selenium Server会给出以下警告：



Selenium Server提示说；如果测试案例是运行在\*iehta或者\*chrome上，或者改变Selenium Server的运行模式为proxy injection mode即可避免问题出现。

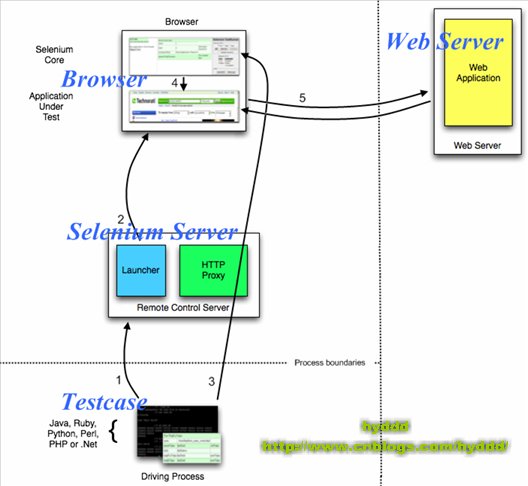
    恩，在这里，我不得不承认之前在《[深入了解Selenium RC工作原理(1)](http://www.cnblogs.com/hyddd/archive/2009/05/30/1492213.html)》中，为了简化问题，我故意少写了一些东西！

    其实，Selenium Server其实有2种运行模式：

(1).**Heightened Privileges Browsers**

(2).**Proxy Injection**

现在Selenium Server启动的默认模式为：Heightened Privileges Browsers。如果要启动Proxy Injection模式，可以加参数**“-proxyInjectionMode**”。而之前在《[深入了解Selenium RC工作原理(1)](http://www.cnblogs.com/hyddd/archive/2009/05/30/1492213.html)》中介绍Selenium RC与Testcase关系，其实就是在描述**Proxy Injection**的工作模式，因为我个人认为Proxy Injection设计模式更为合理，所以只对Proxy Injection模式作介绍。在这里我补充说明一下，为什么Heightened Privileges Browsers模式不能避免上面的问题。先看看Selenium Server在**Heightened Privileges Browsers**模式下的工作流程图：



和Proxy Injection模式不一样，在Heightened Privileges Browsers模式下，Browser对Web的请求没有经过Http Proxy，所以返回的Web页面就有可能和Selenium Core不同源了(一般的情况下，Open都应该只获取“源”下的某个子页面，用Open获取其他“源”的页面在测试中应该是极少数的，因为在这种情况下，我们还应该再new一个新的ISelenium selenium进行处理，虽然用Open获取其他“源”的页面出现的机会极少，但如果真的需要这样的话，也只能启动-proxyInjectionMode模式了，虽然这样的效率会低一些)。