**[selenium2.0 WebDriver原理分析](http://wiki.primeton.com/pages/viewpage.action?pageId=10847668)**

**【方案背景】**

Selenium是一个用于Web应用程序测试的工具。Selenium测试直接运行在浏览器中，就像真正的用户在操作一样。然 而，selenium的使用上，还存在着一些使用比较繁琐的步骤和高低版本API使用不兼容的情况。如果要对selenium现有的API进行封装和兼容 selenium的高低版本的话，就需要理解selenium的运行机制。

**【需求来源】**

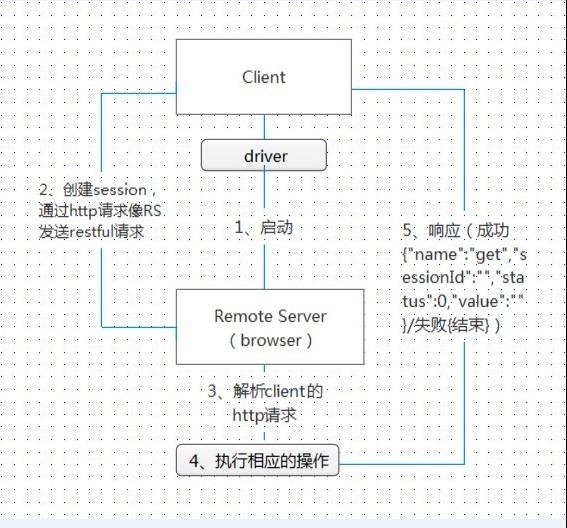
深登自动化测试项目中，整个测试框架为了达到测试人员编写测试用例的简易性、易维护性的目的，需要对selenium现有的一些使用繁琐、 selenium高低版本API使用不兼容的技术进行封装，以及根据业务的需求封装成业务组件。深登自动化测试项目时基于selenium2.0的版本进 行封装的，所以需要理解selenium2.0中WebDriver的运行机制。

**【需求分析】**

Selenium2.0 = Selenium1.0 + WebDriver，Selenium1.0可以使用任何编程语言，但是有个先决条件就是必须支持HTTP库。Selenium1.0起初就是一个 Javascript库，到后面引入了SeleniumRC。SeleniumRC作为一个代理服务器并且发送操作命令给Selenium Core（javascript代码，且为SeleniumRC的一部分）。SeleniumRC从测试程序接收指令并翻译，返回测试结果给测试程序。 Selenium Core在client API打开浏览器后就注入到浏览器中，然后Selenium Core接收测试程序的指令，解释成selenese命令，在浏览器执行。

WebDrive是一个轻便简洁的自动化测试框架。WebDriver通过尝试不同的方法去解决Selenium1.0所面临的问题。不单单是使用 javascript，WebDriver会使用任何一种更合适的机制来操作浏览器。这样就能很好的绕过浏览器javascript的安全限制。当这些技 术还不够用时，还可以调用系统设备操作，尤其是需要一些键盘和鼠标操作时，通过这些技术，可以更好的模拟用户的真实浏览器操作。

**【方案设计】**

**1、** **WebDriver原理图**   
****  
**2、** **firefox-webdriver实现为例**

1.当测试脚本启动firefox的时候，selenium-webdriver 会首先在新线程中启动firefox浏览器。如果测试脚本指定了firefox的profile，那么就以该profile启动，否则的话就新启1个profile，并启动firefox；

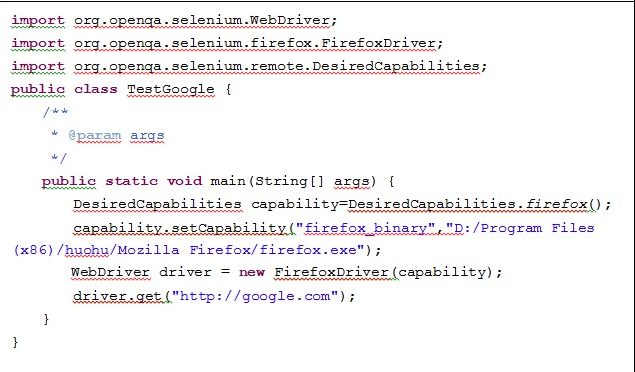
2.firefox一般是以no-remote的方法启动，启动后selenium-webdriver会将firefox绑定到特定的端口，绑定完成后该firefox实例便作为webdriver的remote server存在；

3.客户端(也就是测试脚本)创建1个session，在该session中通过http请求向remote server发送restful的请求，remote server解析请求，完成相应操作并返回response；

4.客户端接受response，并分析其返回值以决定是转到第3步还是结束脚本；

**3、** **调试例子**

下面代码的作用是”命令”firefox转跳到google主页：



在执行 driver.get("http://google.com");  这句代码时，client也就是我们的测试代码向remote server发送了如下的请求：



通过post的方式请求localhost:port/hub/session/session\_id/url地址，请求浏览器完成跳转url的操作。

如果上述请求是可接受的，或者说remote server是实现了这个接口，那么remote server会跳转到该post data包含的url,并返回如下的response：



该response中包含如下信息

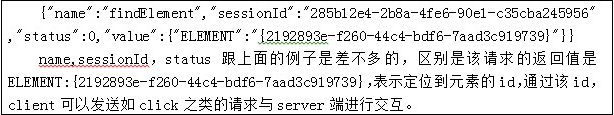
name：remote server端的实现的方法的名称，这里是get，表示跳转到指定url；

sessionId：当前session的id；

status：请求执行的状态码，非0表示未正确执行，这里是0，表示一切ok不必担心；

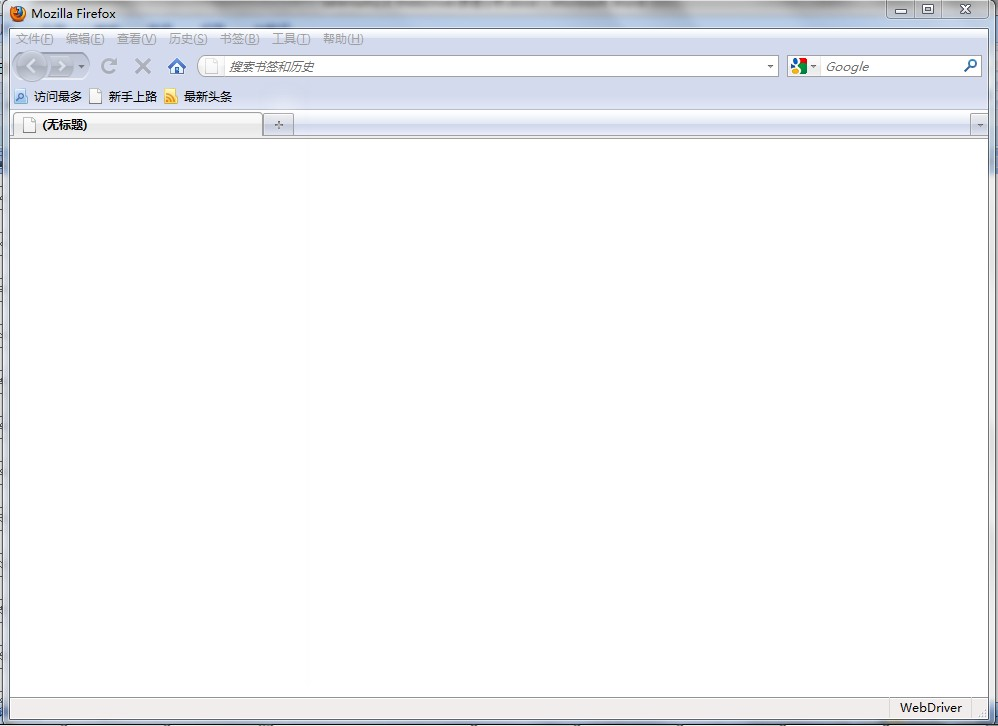
value：请求的返回值，这里返回值为空，如果client调用title接口，则该值应该是当前页面的title；

如果client发送的请求是定位某个特定的页面元素，则response的返回值可能是这样的：



     那么remote server端的这些功能是如何实现的呢？答案是浏览器实现了webdriver的统一接口，这样client就可以通过统一的restful的接口去进 行浏览器的自动化操作。目前webdriver支持ie, chrome, firefox, opera等主流浏览器，其主要原因是这些浏览器实现了webdriver约定的各种接口。

**【实施效果】**

**1、** 运行代码，selenium-webdriver 会首先在新线程中启动firefox浏览器:   
  
**2、** 执行 driver.get("http://google.com")；浏览器完成跳转url的操作:

  
**【相关下载】**

         对Selenium WebDriver感兴趣的同学，可以下载以下可运行实例进行调试，也可以下载帮助文档进行更深入的了解！

         附件 selenium.zip 是eos项目(eos版本为：Platform 6.2，项目文件夹libs下包含了selenium需要的jar包)；

         项目构建包com.primeton.eos 下的SeleniumTestCase类的main方法模拟了governor的登录操作；  
         运行时，在eos中导入项目，启动项目，然后执行SeleniumTestCase的main方法，整个模拟governor登录的操作就开始了。

         相关实例下载：[seleniumTest.zip](http://wiki.primeton.com/pages/viewpageattachments.action?pageId=10846542&highlight=seleniumTest.zip#Home-attachment-seleniumTest.zip)

         相关帮助文档下载：[selenium2.0\_中文帮助文档.doc](http://wiki.primeton.com/pages/viewpageattachments.action?pageId=10846542&highlight=selenium2.0_%E4%B8%AD%E6%96%87%E5%B8%AE%E5%8A%A9%E6%96%87%E6%A1%A3.doc#Home-attachment-selenium2.0_%E4%B8%AD%E6%96%87%E5%B8%AE%E5%8A%A9%E6%96%87%E6%A1%A3.doc)