目录

[一、 基础篇 1](#_Toc496291062)

[1. 背景和目的 1](#_Toc496291063)

[2. 文章结构安排 2](#_Toc496291064)

[3. 环境搭建 2](#_Toc496291065)

[4. 启动浏览器 18](#_Toc496291066)

[5. 自动化测试脚本 21](#_Toc496291067)

[二、 练习篇 22](#_Toc496291068)

[1. 如何在Eclipse上查看Selenium源码 22](#_Toc496291069)

[2. 从代码角度去解释启动firefox的过程 24](#_Toc496291070)

[3. 接口相关方法介绍 26](#_Toc496291071)

[4. 查找元素（findElement） 28](#_Toc496291072)

[5. findElement之By PartialLinkText 28](#_Toc496291073)

[6. findElement之By CSS 28](#_Toc496291074)

[三、 框架设计篇 28](#_Toc496291075)

1. 基础篇

# 背景和目的

       我大概是2015年开始接触Selenium这个web自动化测试框架。当时是selenium2.48好像，从Selenium2.0开始，就加入了webdriver，实际上，我们说的selenium自动化测试，大部分情况都是在使用webdriver的API。现在去Selenium官网，发现最新Selenium版本是3.4，我也不知道3.4和2.48之间有什么多大的区别和新的特性，唯一确定的是，在Selenium2.0的时候，启动后firefox的时候，还不需要额外的driver插件支持，现在到Selenium3.0，启动firefox，还需要geckdriver.exe这个在针对firefox浏览器的插件。如果了解Python+Selenium的朋友，应该对这个很清楚了。我看博客里的Python+Selenium自动化测试系列，很多人喜欢看，所以，在这里，也写一个Java系列的Selenium教程。同时，也是对Selenium3.0的在Java上的一个重新学习过程。

# 文章结构安排

       预计和Python+Selenium系列一样，主要分成：基础篇，练习篇，框架设计篇。主要的框架设计思想还是POM，即页面对象模型。前提条件，你要掌握Java基础语法和基本知识，如果没有学过Java，请看我博客里Java基础教程，这个Java基础教程也会一直更新，就是为了没有Java基础的朋友准备的。

# 环境搭建

## 3.1所需组建或工具

--JDK

--Eclipse

--Selenium for Java

--Selenium 在不同浏览器上的driver插件

## 3.2安装和配置JDK

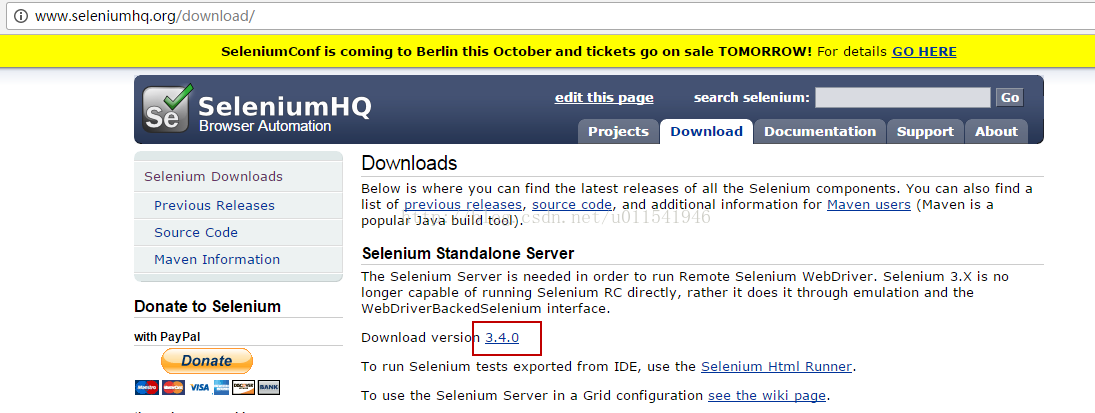
JDK安装我们这里选择最新的JDK 1.8.131,如果不会，请参照[这里](http://blog.csdn.net/u011541946/article/details/72848401" \t "_blank)。

## 3.3安装和设置Eclipse

关于Eclipse的下载和设置，请参考[这里](http://blog.csdn.net/u011541946/article/details/72852229" \t "_blank)。

## 3.4下载Selenium相关包和工具

Selenium下载的官网是<http://www.seleniumhq.org/download/> ，在中国，需要连接VPN才能下载，点击下图中的3.4.0链接触发下载selenium-server-standalone-3.4.0.jar



下载不同浏览器driver插件

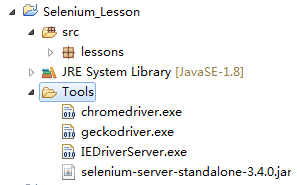


如果你没有VPN，或者无法访问Selenium官网，我已经给你们准备好了，[点这里](http://pan.baidu.com/s/1dFrdpeh" \t "_blank)。

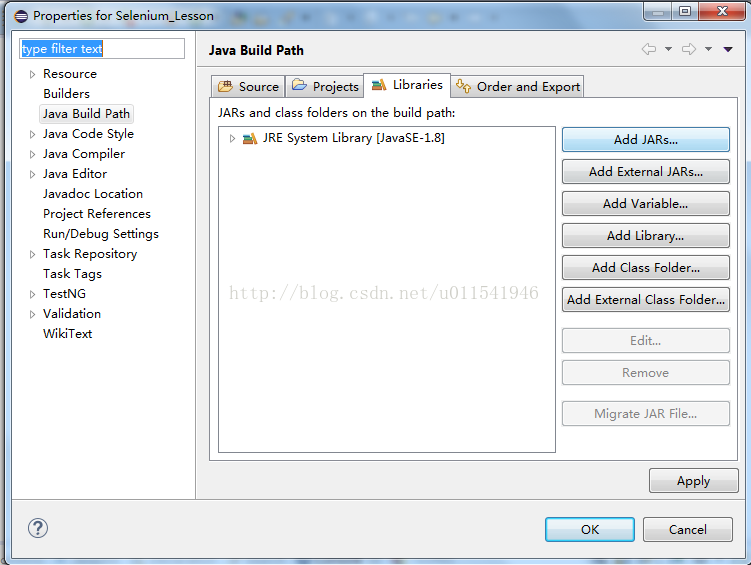
## 3.5创建一个Selenium的Java项目，用来练习

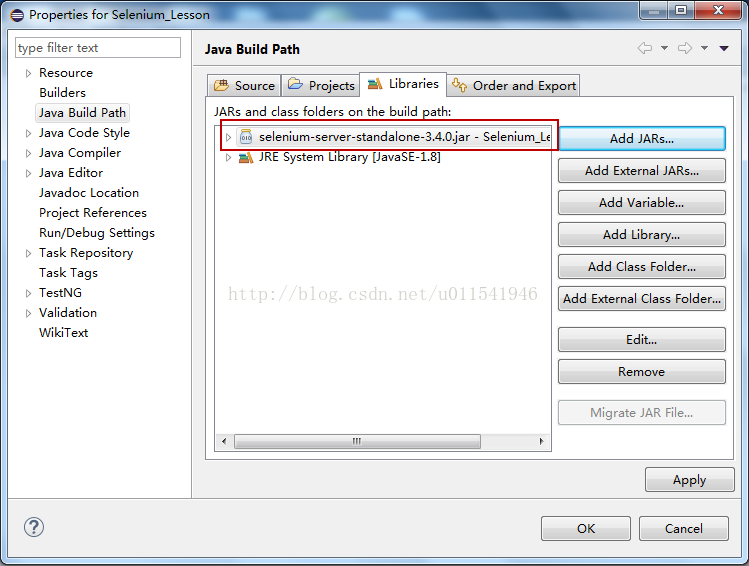
1) 打开Eclipse，创建一个名称为Selenium\_Lesson的Java 项目

2) 在Selenium\_Lesson项目下，新建一个文件夹，把Selenium相关包放到这个文件夹下



3) 右键项目，选择Build Path,然后选择Confige Build Path...，弹出框点击Libraries



这里点击Add JARs...，弹出框，选择当前项目下tools这个文件夹里面的selenium-server-standalone-3.4.0.jar，确定，效果如下。  


点击OK, Java+Selenium3.0的安装搭建就介绍到这里，下一篇介绍如何打开firefox浏览器。

## 3.6安装java

　　访问java官网下载自己需要的版本,我现在使用的是**jdk-8u144-windows-x64.exe**

　　java环境分JDK和JRE，JDK就是Java Development Kit。JDK是面向开发人员使用的SDK，它提供了Java的开发环境和运行环境。JRE是Java Runtime Eneiroment是指Java的运行环境，是面向Java程序的使用者。

　　JAVA安装直接下一步下一步就好了。安装目录你随意







安装完成后点击关闭。然后配置环境变量：“我的电脑/计算机”右键菜单--->属性--->高级系统设置--->环境变量--->系统变量（当时我设置环境变量的时候老是在用户变量里配置JAVA\_HOME跟CLASS\_PATH，所以一直在cmd里javac 总是提示不是内部命令也不是外部命令...）

　　1)新建

　　　　变量名：JAVA\_HOME

　　　　变量值：C:\Program Files\Java\jdk1.8.0\_101\

　　　　变量名：CLASS\_PATH

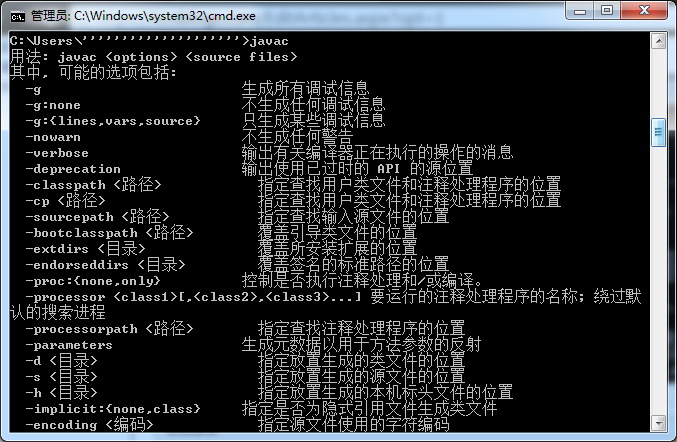
　　　　变量值：.;%JAVA\_HOME%\lib\dt.jar;%JAVA\_HOME%\lib\tools.jar;

　　2)找到path变量名--->“编辑”添加：

　　　　变量名：PATH

　　　　变量值：%JAVA\_HOME%\bin;%JAVA\_HOME%\jre\bin;

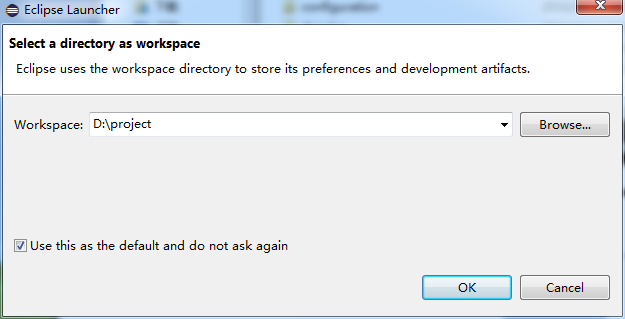
　　3)配置好环境变量后可在Windows命令提示符下验证Java是否成功：



## 3.7安装eclipse

　　Eclipse下载：<http://www.eclipse.org/downloads/>，将下载的文件**eclipse\_v4.7.0.exe**

双击eclipse启动程序。需要你指定java的项目目录，或者你希望java项目创建在哪个目录下。我写**D:\ eclipse-workspace**



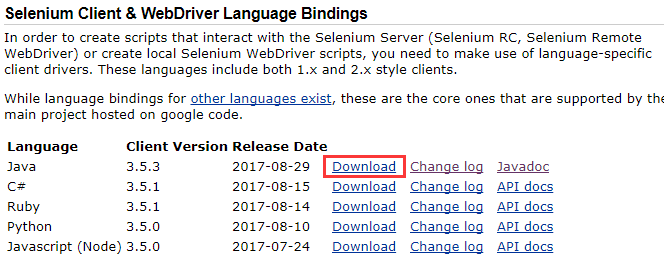
安装成功，启动是这样的一个welcome界面



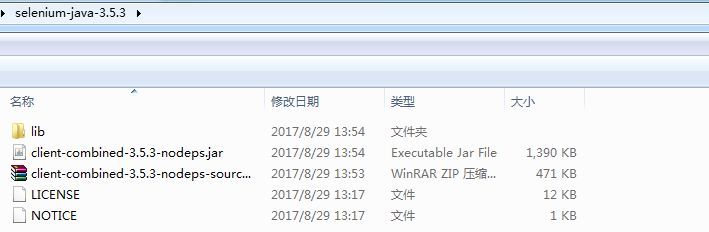
## 3.8安装selenium

我们需要安装Java版的selenium，**selenium-java-3.5.3.zip**

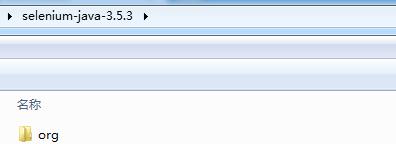
<http://docs.seleniumhq.org/download/>



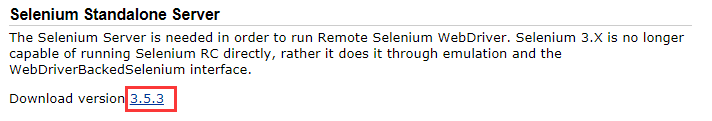
解压出来是这样的



再将**client-combined-3.5.3-nodeps-sources.zip**解压得到包文件

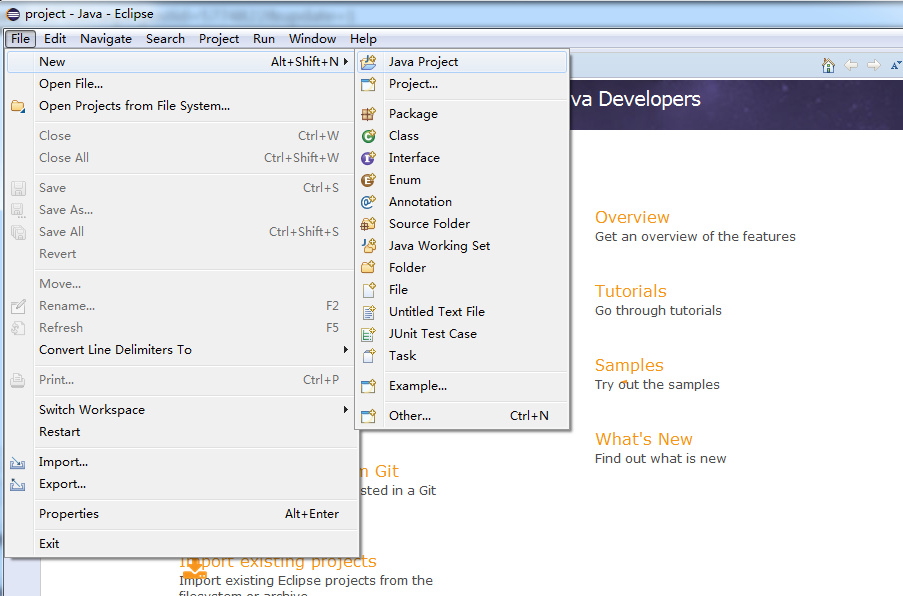


下载Selenium Standalone Server selenium-server-standalone-3.5.3.jar

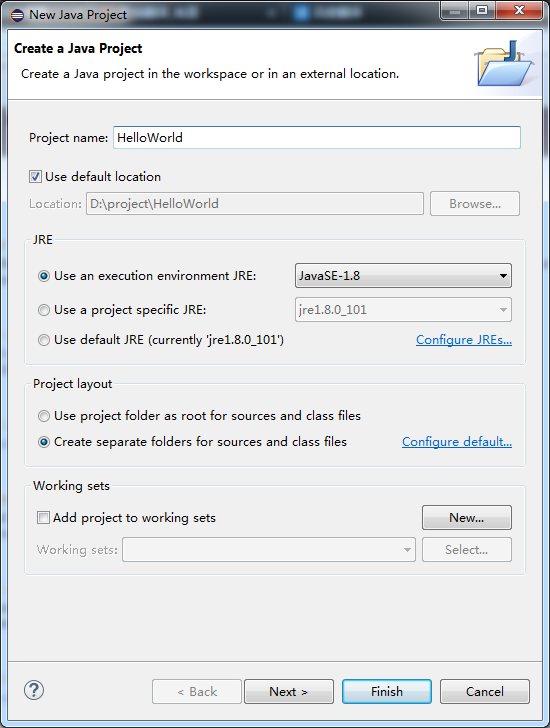


## 3.9将Selenium的包导入到Eclipse当中

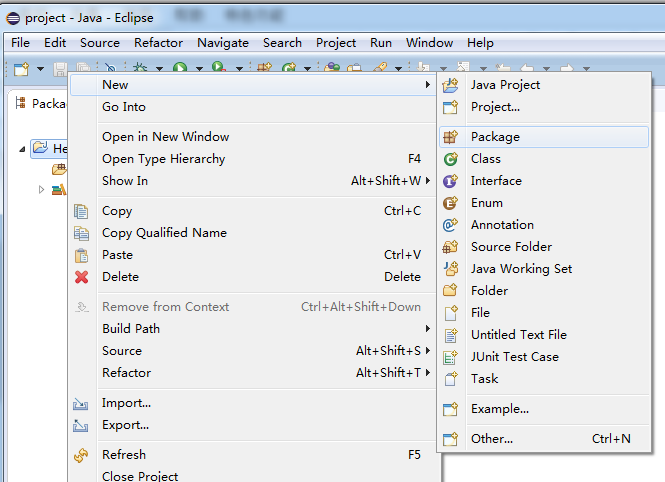
**3.9.1创建项目**

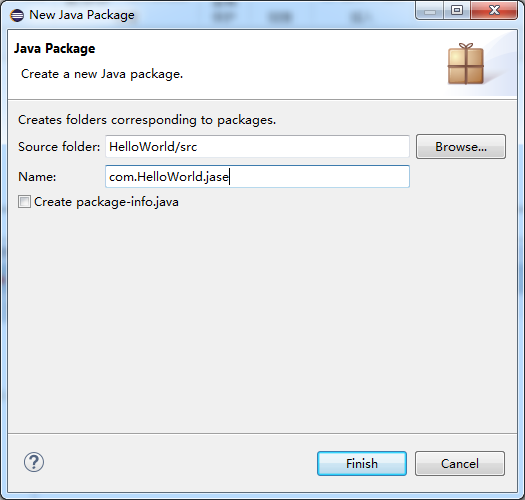


输入项目名称，直接finish。如果想看下一步创建项目有什么也可以next

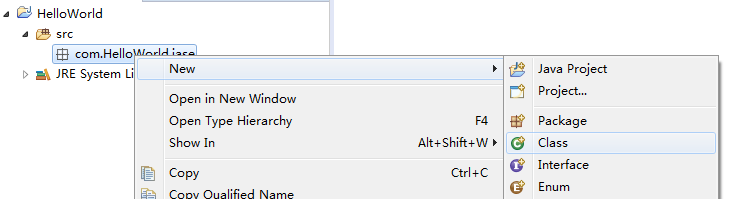


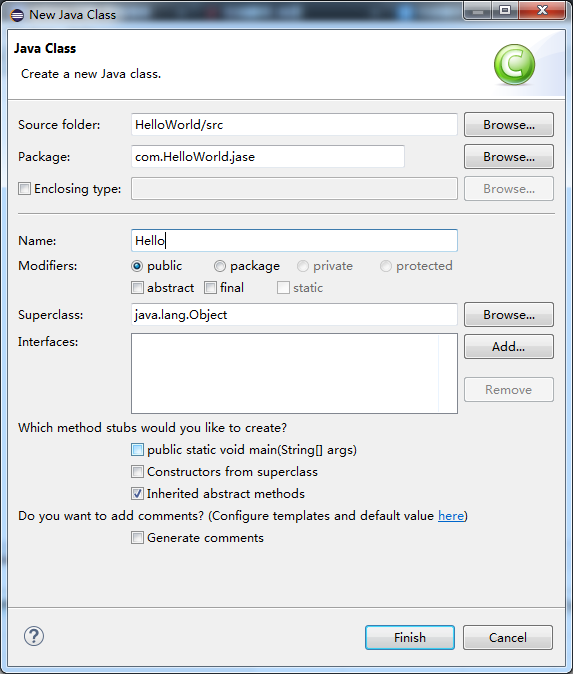
**3.9.2创建包：** 包（package）的概念可以理解为程序的集合

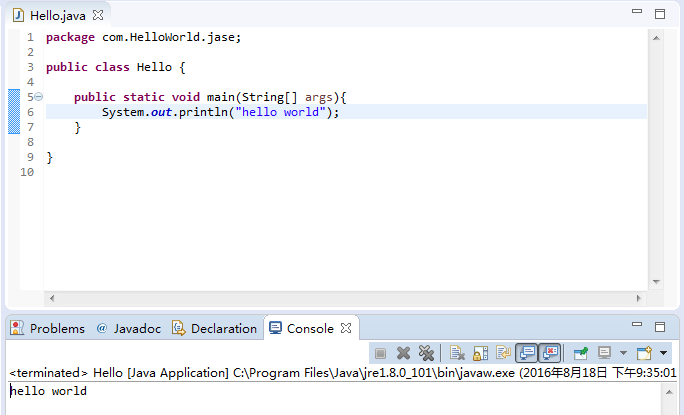




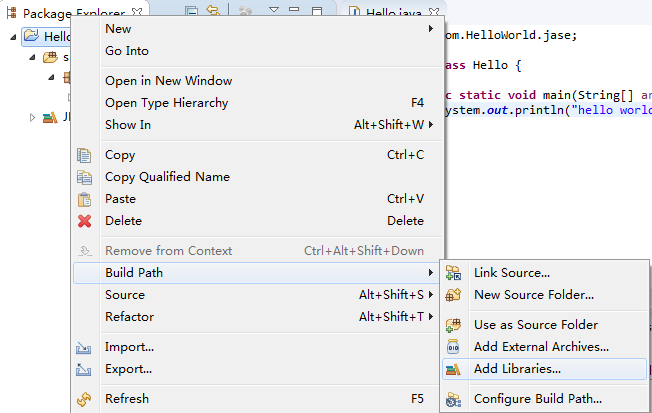
**3.9.3创建class**



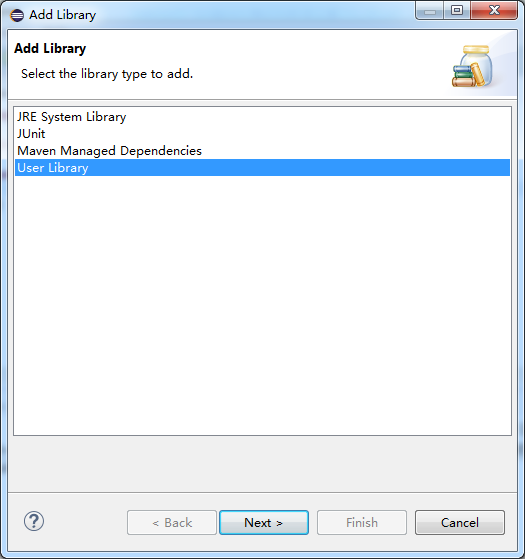


OK  咱们hello world出来了  


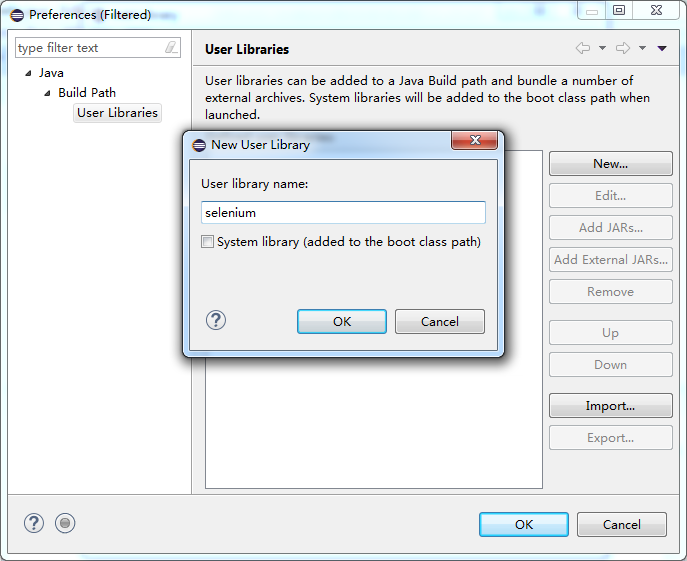
**3.9.4导入Jar包：**再创建的项目上右键Build Path--->Add Libraries

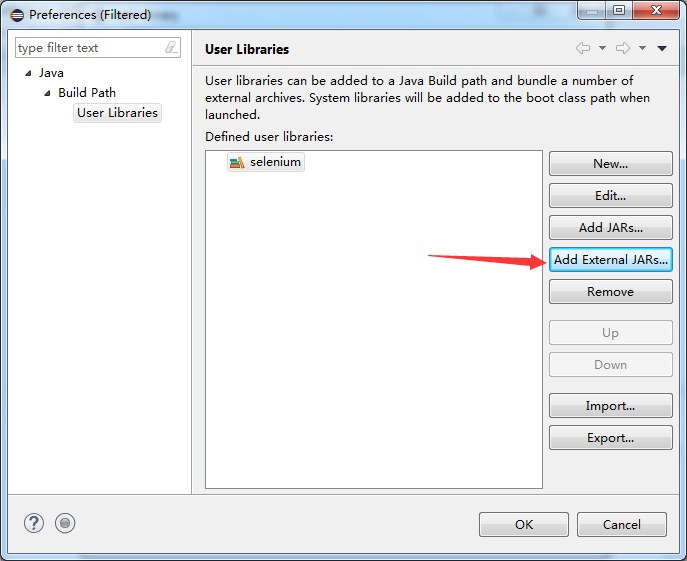


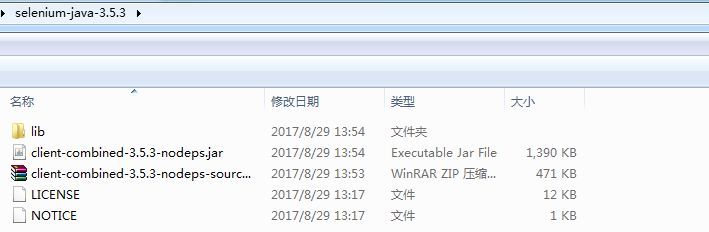
然后User Library然后next---》User Libraries...



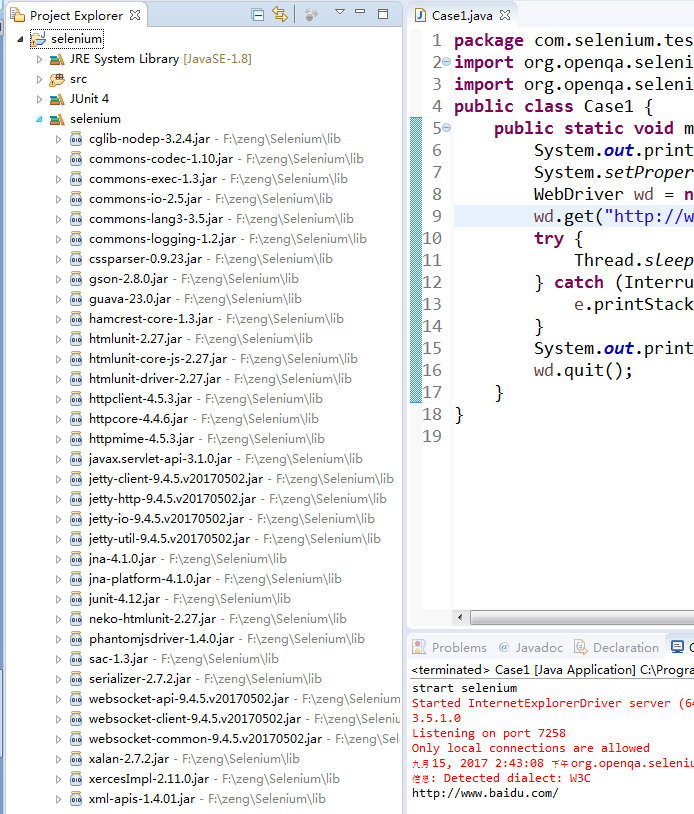
创建一个Library的目录，名字随便取





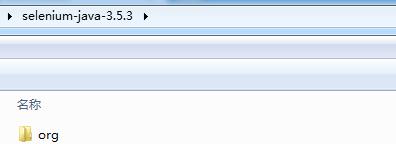


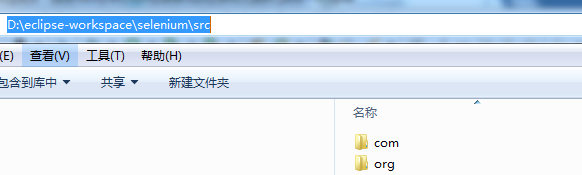
把这些jar包和下载好的selenium-server-standalone-3.5.3.jar导入进来就可以了



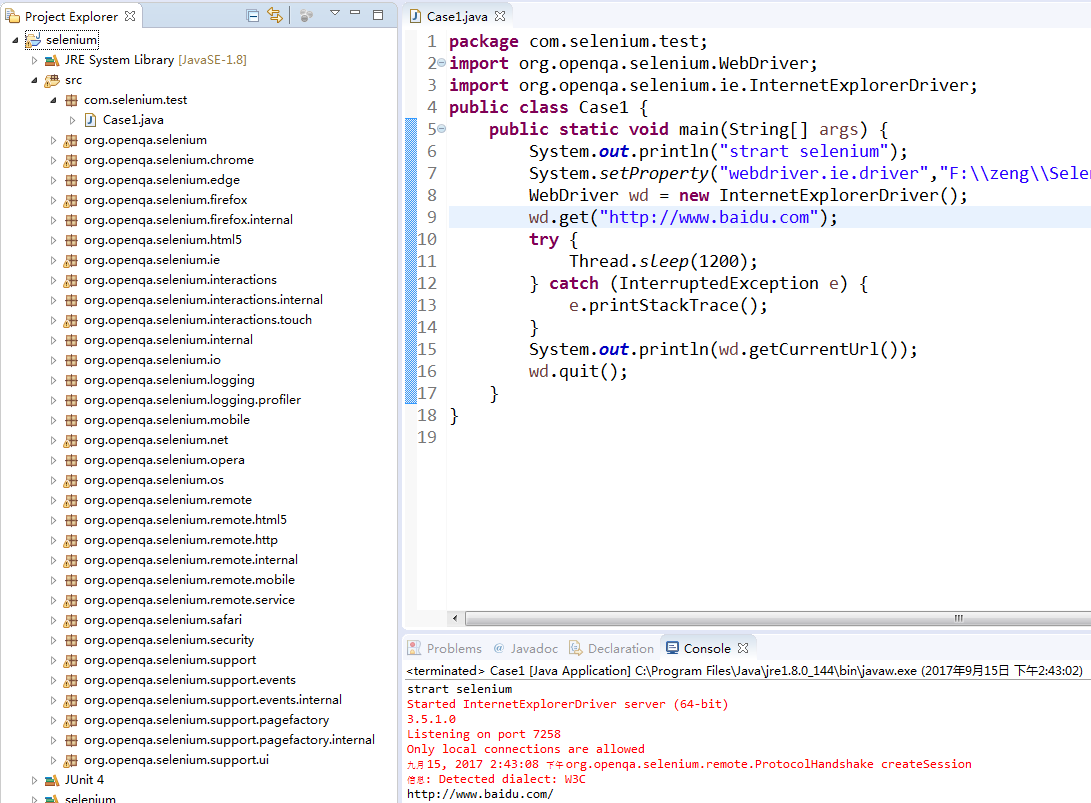
**3.9.5 导入Org包**

将**client-combined-3.5.3-nodeps-sources.zip**解压得到的包文件复制到eclipse项目的src文件夹





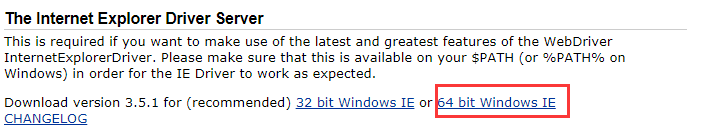
Src下面的包文件列表：



# 启动浏览器

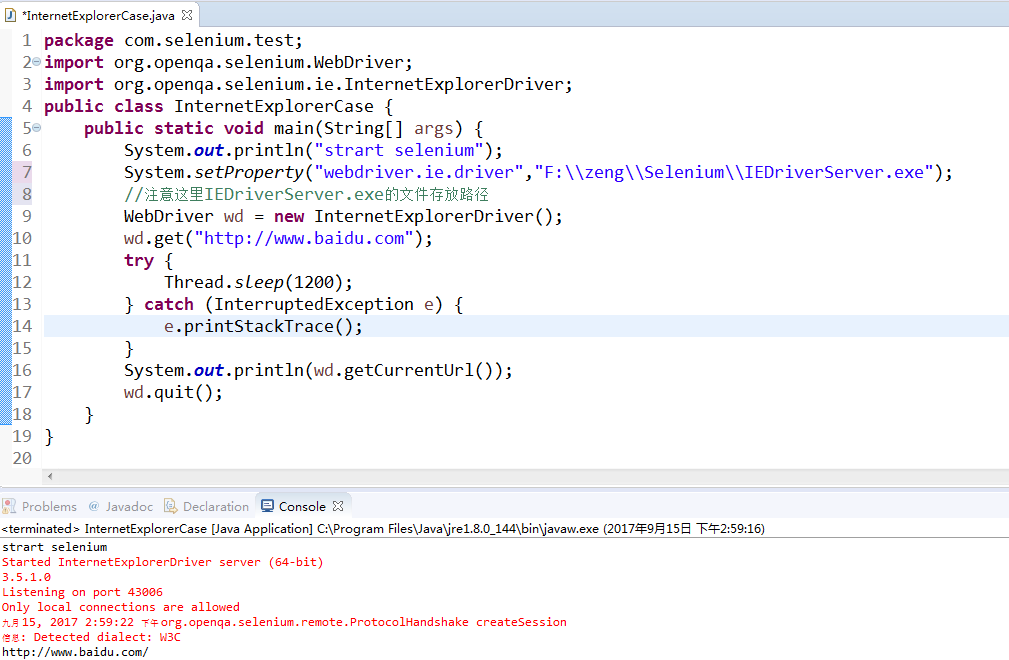
## 4.1创建IE驱动

本文介绍如何启动IE浏览器，由于IE浏览器的兼容性和不稳定性，我在实际测试过程中，基本上不用IE去测试自动化脚本。个人建议，你也不要去用IE测试Selenium自动化测试脚本，但是这里还是介绍下，如何启动IE浏览器。

1）先在<http://docs.seleniumhq.org/download/ 下载IE驱动>**IEDriverServer.exe**

2）设置IE驱动路径：放在java项目相对路径，或者绝对路径

System.setProperty("webdriver.ie.driver"," .\\Tools\\IEDriverServer.exe")



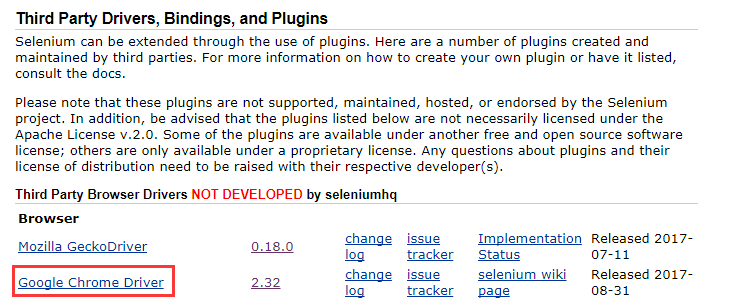
运行下，可能会出问题，我的IE是IE9，运行就死机。IE11应该没问题，IE11上还应该注意，需要把安全设置调整成相同的模式。具体可以参考：

Internet选项->安全; 把Internet站点，本地Intrant,受信任站点 三个地方的安全界面都设置相同等级，例如都设置中； 再次运行代码就可以用IE打开百度了。

如果你还需要测试Edge浏览器，你还需要去Selenium官网下载Edge的driver插件，这里我不使用Edge，就不介绍了，原理和代码和IE/FIrefpx/Chrome差不多。

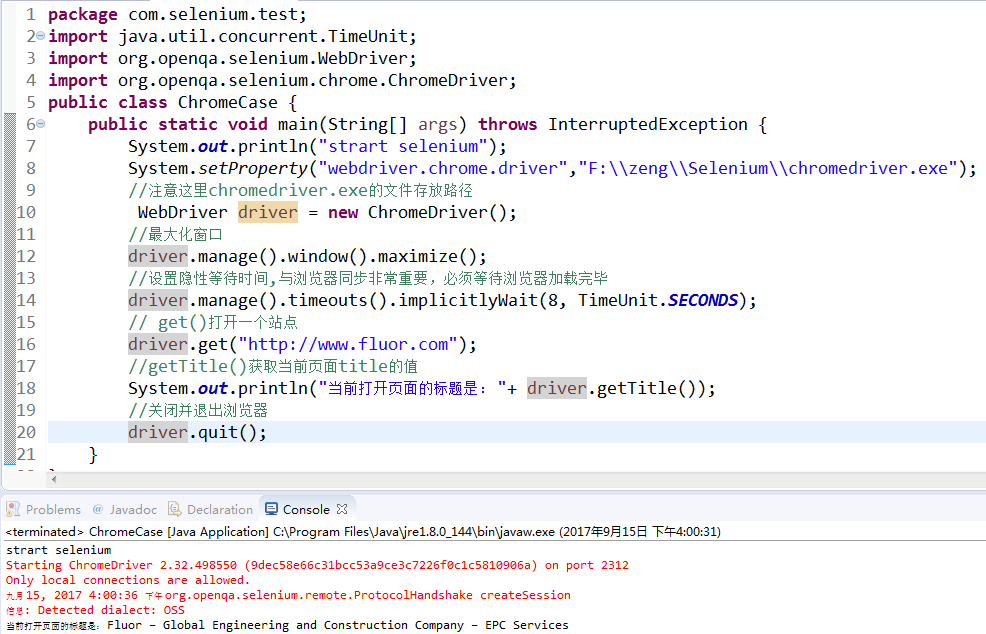
## 4.2创建chrome驱动

1）先在<http://docs.seleniumhq.org/download/ 下载chrome驱动>**chromedriver.exe**



2）设置chrome浏览器驱动路径：放在java项目相对路径，或者绝对路径

System.setProperty("webdriver.chrome.driver"," .\\Tools\\chromedriver.exe ")



## 4.3创建firefox驱动

本文介绍如何在Selenium3.0上，用Java脚本实现如何启动和关闭firefox浏览器。Selenium最显著的优点就是夸浏览器测试Web软件项目。

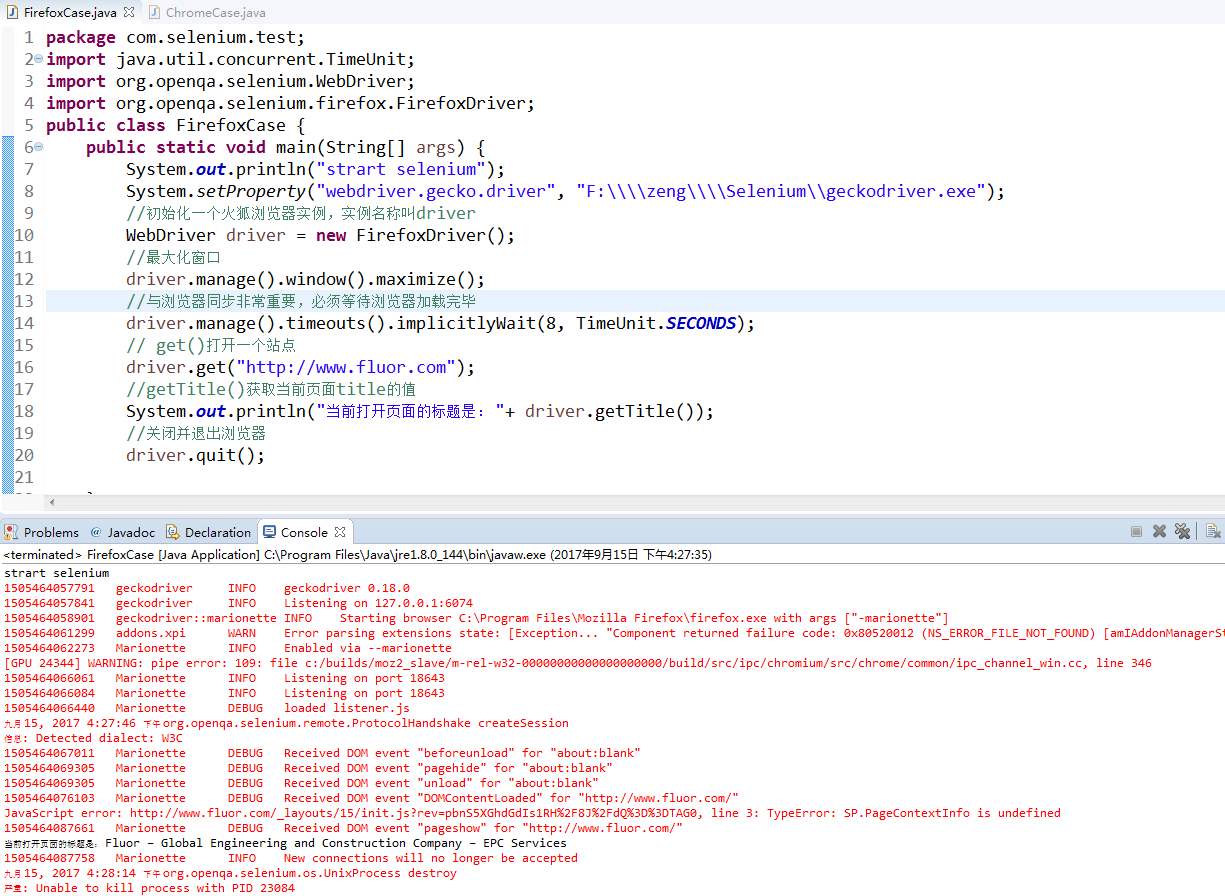
备注：关于在Mac上如何启动firefox浏览器，有需要的，请看这位朋友的文章：http://blog.csdn.net/u012911704/article/details/78215967，还是和windows版本有很大的区别。

1）先在[http://docs.seleniumhq.org/download/ 下载firefox驱动](http://docs.seleniumhq.org/download/%20下载firefox驱动)**geckodriver.exe**



2）设置firefox浏览器驱动路径：放在java项目相对路径，或者绝对路径

System.setProperty("webdriver.gecko.driver", "F:\\\\zeng\\\\Selenium\\geckodriver.exe")



# 自动化测试脚本

前面介绍了环境搭建和三大浏览器的启动方法，这里我们介绍自动化测试脚本。我的个人经验是，自动化脚本编写比较容易，最大的困难是如何去写测试断言。自动化测试，最重要的还是落在测试上面，而不是自动化，自动化只是手段。断言的写法，就考验出一个测试工程师是否考虑全面，体现出你的用例编写水平。

举例：验证打开链接是百度首页

分析问题：

我们输入了url，然后打开一个页面，如何判断这个页面是不是我们提前知道的页面呢。

1. 获取当前url是不是和输入的一致，假如链接没有发生重定向。

2. 获取当前页面title，是不是我们期待的结果。

3. 在页面元素body找一个特征元素，这个特征元素能够代表是该网站。

我想，也大概只有上面三点依据可以判断我们访问的网站对不对。第一点，有时候不靠谱，不知道你们是否听说过盗链，就是输入一个网站URL，显示的内容是别人的网站，而不是你测试的网站。甚至有链接的重定向出现。第二点，还是比较靠谱，title这个值是网站前端开发同学写入的，基本上可以代表是你要打开的网站。第三点，有点难找这样的特征元素，而且有时候找到了也不一定是正确的。

Java中有一个方法叫assert,中文意思叫断言，assert后面跟着的是一个表达式，表达式的值是布尔型。上面的例子就是，当判断表达式值是true，就打印Tess Pass。这里可以看到，我把断言部分代码放在try catch语句中。编写脚本或者编写代码要养成一个好习惯，习惯主动去捕获异常，并处理异常。这里我们自动化脚本，基本上没有需要去处理异常，一般选择直接打印出异常信息就可以。

到这里，Selenium WebDriver基础篇就结束了，下面主要是介绍WebDriver的一些常见接口或者方法的实际使用，尽量多介绍一些Selenium方法，为后面的方法封装和框架设计做铺垫。

1. 练习篇

# 如何在Eclipse上查看Selenium源码

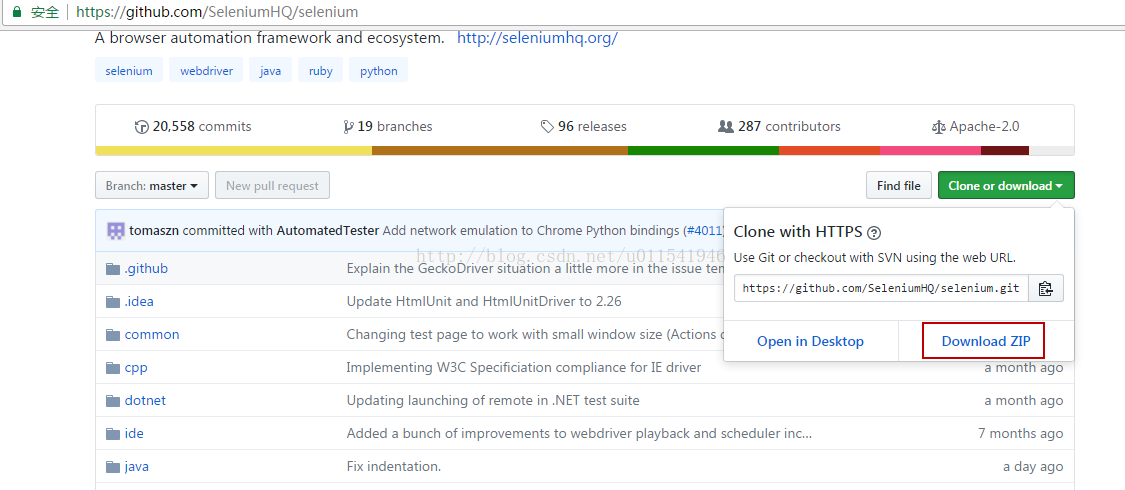
前面基础篇，足够你可以对Seleniun入门，至少知道写你第一个基于Java的Selenium自动化测试脚本。接下来我们介绍Selenium方法篇，主要是介绍一些常用的Selenium方法后者接口，通过这些接口或者方法的具体操作，达到能够熟练使用Selenium编写Java的自动化测试脚本，并且，为后续的Java+Selenium自动化测试框架设计打基础。只有你完全玩过或者使用了里面的一些方法，你在框架设计篇，才能有这个思想或者觉悟，为什么要封装，为什么要这么设计等等。

## 如何才能在Eclipse上查看源码？

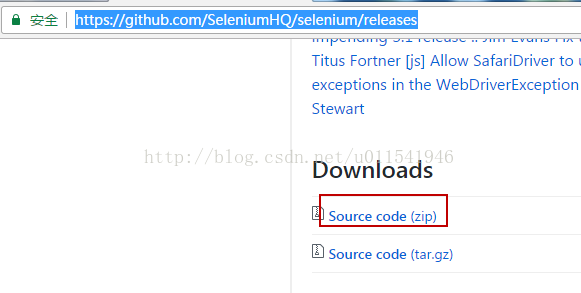
       在Java基础教程，我写了一篇文章，介绍如何查看JDK源码，在Eclipse上需要添加源码附件，具体可以看看[这里](http://blog.csdn.net/u011541946/article/details/72866315" \t "_blank)。查看Selenium源码也实现相同的步骤，只是我们首先需要下载Selenium的源码文件。

## 1.2如何下载Selenium源码文件

这里介绍两个下载页面，其实第一个下载页面可以点击跳转到第二个下载页面。



      不知道大家在github上下载过别人的项目代码没，就是点击红圈中的Download ZIP,就可以。第二种是点击上面那个96 releases出现如下界面。



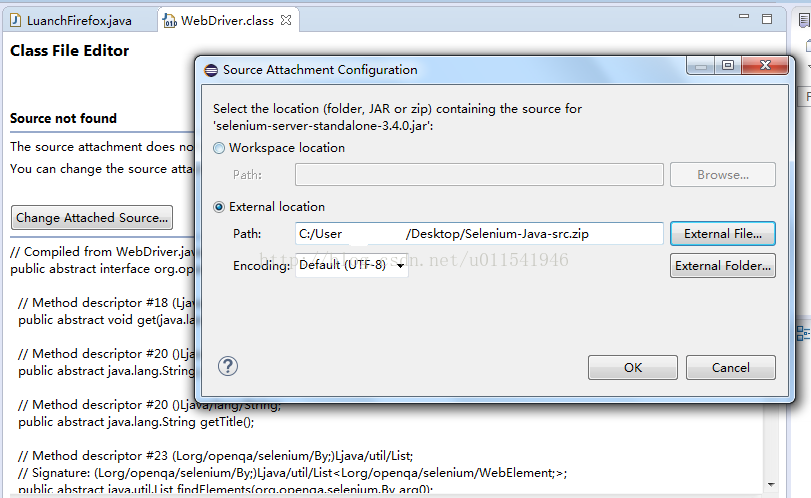
点击上图红圈的位置下载源码。

## 1.3制作Selenium-Java-src.zip源码文件

      下载完，解压之后，发现是一个很大的文件，几百M大小。我们只是看Java部分的Selenium源码，所以，我们找到解压之后的路径：C:\Users\Anthony\Desktop\selenium-selenium-3.4.0\java\client，在这个路径下，我们把src这个文件夹拷贝到桌面，然后右键-选择发送到ZIP文件夹，把桌面生成的ZIP命名为Selenium-Java-src.zip

## 1.4在Eclipse上添加Selenium-Java-src.zip

      打开之前你写的一个脚本，鼠标悬停到driver.get("https://www.baidu.com");鼠标悬停在get这个单词上方，然后按下ctrl键，点击鼠标左键，出现source not found页面，然后选择Selenium-Java-src.zip



之后里面可以看到get方法的源码，你可以看看src下面各种Selenium的方法，接下来的文章的方法都是从里面可以找到的。

# 从代码角度去解释启动firefox的过程

通过前面文章，我们知道如何去查看Selenium相关接口或者方法。一般，我们看到更多的是接口，在查看源码的时候，你可以看着这个接口的备注备份，它会告诉你，这个接口或者方法的作用，有哪些参数，参数类型是什么。为了更好去描述和理解这个过程，我们举例，通过查找源码的方式去理解Selnium启动firefox的过程。

**System.setProperty("webdriver.gecko.driver", ".\\Tools\\geckodriver.exe");**  
       按下Ctrl+鼠标悬停在setProperty上方，点击鼠标左键，可以看到java中setProperty的源码。自己去阅读下代码中关于setProperty的介绍。其实就是设置指定键对值的系统属性。上面webdriver.gecko.driver就是键，.\\Tools\\geckodriver.exe就是值。这样就把geckodriver设置成为系统的全局变量！这个时候driver就相当于一个静态变量，存放在内存里，直到driver关闭。  
       所谓的 system porperty，system 指的是 JRE (runtime)system，不是指 OS。   
设置指定键指示的系统属性,可以利用系统属性来加载多个驱动。所以，上面这行代码，就是通过键和值指定firefox 的驱动位置。  
**WebDriver driver = new FirefoxDriver();**  
       点击查看WebDriver发现是一个接口，它的备注这样写的：WebDriver是一个测试的主要接口，它展现了一个理想化的web浏览器，它主要包括三个目录。1）控制浏览器本身 2）查找和选择元素 3）调试程序，比如异常处理。  
       driver这里是一个实例对象，学习了Java中类和对象，就应该不难理解。  
       new 是一个关键字，Java中通过new这个关键字，可以在内存中开辟一块空间，用来加载变量。  
FirefoxDriver(),是WebDriver这个接口在firefox上的一个实现具体类。FirefoxDriver这个类里面，还包含一些firefox浏览器的一些选项设置。这行代码的意思用一句话来讲：初始化一个firefox类型的driver实例对象。这里除了firefox，还有IE，Safari,chrome等对应的driver启动方法，你可以查看\*\Selenium-Java-src\org\openqa\selenium,可以找到这些接口文件。  
**driver.manage().window().maximize();**  
      这里driver，就是指上面我们初始化的firefox的实例对象，就是类似一个真实浏览器。manage是Options这个接口的一个方法，window().maximize()，window也是一个接口，这个接口下，有maximize这个方法，也就是最大化浏览器，window下也有全屏，设置窗口大小的方法。  
**driver.manage().timeouts().implicitlyWait(8, TimeUnit.SECONDS);**        manage上面提到是一个方法，直接来看timeouts，timeouts是接口Timeouts的一个实例对象，它的左右是针对webdriver实例管理超时的一个接口。implicitlyWait是一个隐式等待，当在一定时间内，如果还没有找到页面元素，就报超时。参数有两个，第一个是8，第二个是时间单位，这里选择秒，所以这里是8秒后超时。  
**driver.get("https://www.baidu.com");**  
       这里的get方法的作用是，在当前浏览器窗口，加载一个新的web页面，是通过http get发生请求完成的。参数类型是String，一般是url。get方法就是打开一个网页的作用。  
**driver.quit();**  
      退出有quit和close两种，这里quit表示退出当前浏览器，关闭这个浏览器有关联的所有窗口。

# 接口相关方法介绍

## 3.1 Timeout接口相关方法介绍

这一篇介绍manage()接口下常用方法的使用。从Selenium的源码文件，可以知道,manage是得到Option这类接口用的。本文介绍的方法，都是在Option这个接口下面可以找到的。Option接口是这样描述的：An interfacefor managing stuff you would do in a browser menu，所以，这些操作都是来控制浏览器本身。

**Timeout接口**

跳转到Timeout接口描述文字部分，下面有三个常见的接口。

implicitlyWait(long time, TimeUnit unit);

setScriptTimeout(long time, TimeUnit unit);

pageLoadTimeout(long time, TimeUnit unit);

先看看implicitlyWait(),这个在前面介绍过了，叫隐式时间等待。我们来看看这个接口的描述部分，大致意思是：隐式时间等待是在查找元素的时候，设置一个最大时间值，如果driver第一次没有找到网页元素，那么就继续找，直到消耗了设置的最大时间值，还是没有找到，那么就抛出没有找到元素的异常。如果在设置时间范围内就找到了该元素，那么就执行下一行代码，不需要再等时间，这就是隐式等待的含义。所以，隐式时间等待一般结合查找网页元素一起。

setScriptTimeout()，作用是，在设置规定的时间内，等待异步脚本的执行结束，而不是在里面抛出错误。这个在执行javascript脚本的时候可能会使用，后面我们介绍了使用Javascript脚本执行Selenium相关操作后在介绍具体使用方法。

pageLoadTimeout(),字面意思就网页加载超时，作用就是在设置规定时间内，等待网页完整加载完成，而不是在里面抛出错误。我们知道如果网速慢，或者网页元素太多，例如网页做了大面积渲染，肯定会造成网页加载速度变慢。这个接口的作用就是，来处理这类问题的。从这个角度来看，我们前面的脚本中设置时间等待的语句应该换成这个语句更合适一些，毕竟我们的目的就是等待网页加载完成。

这里用pageLoadTimeout()举例一个打开百度和打开国外站点的例子，百度首页打开设置2秒，国外站点（icloud.com）打开设置1秒，由于中国大陆访问国外网站速度肯定慢，所以会报超时错误。

把上面的超时时间设置10秒，然后运行，就应该不会报这个超时的错误。我测试过5秒，也不会报超时。

以后我们把这些Selenium方法或者接口分成三类，一类是driver相关，也就是操作浏览器本身设置，第二类是元素相关，就是一些元素的点击，输入等操作，第三类是一些工具支持类，例如截图，日志输出，事件监听。

## 3.2 window接口相关方法介绍

本文继续介绍WebDriver.java这个文件里面的接口方法，上一篇介绍了timeout接口内相关方法，这篇文章介绍window接口。windows接口下有几个方法，主要是用来控制浏览器窗口设置，例如窗口大小，最大，最小，全屏，位置。我们先来看看window接口下有什么方法。

void setSize(Dimension targetSize);  
void setPosition(Point targetPosition);  
Dimension getSize();  
Point getPosition();  
void maximize();  
void fullscreen();

       我们前面使用过了maximize()，这里就不再介绍，第一个我们介绍setSize(Dimension targetSize)和getPosition()，这里要注意下参数 targetSize这个是一个类，不是一个普通的数据类型，来看看如何写关于浏览器位置设置的自动化脚本。

## 3.3 Navigation接口介绍

本文接着学习WebDriver,java这个类文件中的关于driver操作的接口，本篇介绍Navigation接口，主要包括平时浏览器的前进，后退，打开网址，刷新当前页操作。在Navigation接口下，我能找到下面四个方法：

void back()

void forward()

void to(String url)

void to(URL url)

void refresh()

先来解释下他们的作用，back()就是浏览器上地址栏前面向左的箭头，一般叫后退操作。forward()就是浏览器上地址栏前面向右边的箭头，一般叫前进操作，或者转到下一页。to(String url)就是在当前网页打开一个新的网页，这个和新的tab打开是有区别的，to(URL url)也是一样，就是url被作为一个URL对象传入，这个不怎么使用，这里不介绍。refresh()这个就是浏览器刷新按钮操作，或者等同于按下F5。

为了运行效果可见，上面多次使用了 Thread.sleep(2000)，这是一个线程强制暂停的代码，2000指的是2秒，单位是毫秒。这里的这个线程指的就是driver这个实例对象。

# 查找元素（findElement）

## 4.1 findElement之By id

## 4.2 findElement之By XPath

## findElement之By LinkText

## findElement之By ClassName

## findElement之By Name

# findElement之By PartialLinkText

# findElement之By CSS

## 4.8 findElement之By TagName

## 4.9 findElements举例

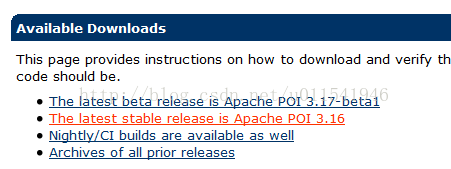
1. 框架设计篇

[利用POI读写excel文件](http://blog.csdn.net/u011541946/article/details/74726045)

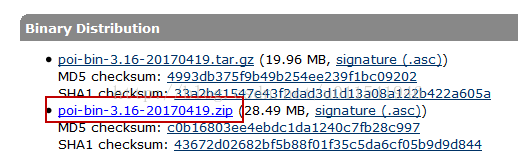
本篇介绍一个第三方工具POI，由于Java的跨平台特性，所以Java没有直接操作excel文件的库支持。在自动化测试，有些我们的测试数据是放到excel文件中，所以需要懂得如何操作获取excel内的内容。

1.下载POI

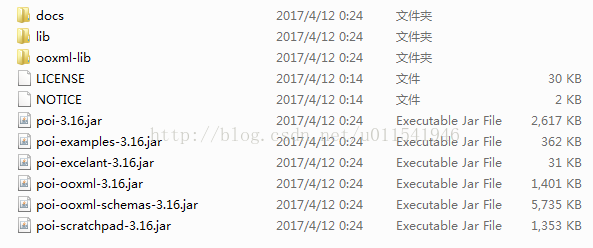
打开网站http://poi.apache.org/download.html，选择下面3.16版本下载。



选择ZIP包下载。



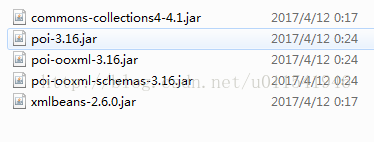
下载完后解压，文件结构如下



2. 把POI文件添加在你Selenium当前项目的library中

先新建一个文件夹，把解压出来的所有的xx.jar都放在新的文件夹，然后右键你的eclipse工程> Build path > Configure build path > 点击lib  > Add external jar，新建文件夹内所有的xxx.jar都选中。

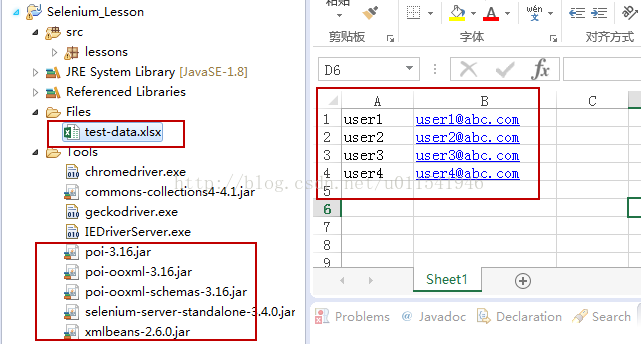
或者，你只需要以下几个jar文件就可以。



注意，这些jar文件不在同一个路径，你需要到不同路径去拷贝复制到一个新文件夹，然后把这个文件夹的所有jar文件拷贝到你eclipse项目下的Files文件夹，需要右键你工程，新建一个叫Files的文件夹。我的项目，采取这种方法，我不希望拷贝全部jar到项目lib中。

3. 准备一个test-data.xlsx文件

桌面新建一个test-data.xlsx，然后输入如下内容，并且把这个文件放到eclipse项目中。



4..新建一个ReadExcel.java的文件

[利用opencsv读取csv文件](http://blog.csdn.net/u011541946/article/details/74907045)

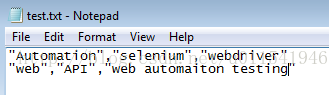
前面介绍了如何读取excel文件，本篇介绍如何读取vsc文件，什么是csv，它的定义是与逗号分隔的值（Comma-Separated Values），同样的Java中需要用到第三方lib去处理读取csv文件的数据。

1.下载opencsv.jar并添加到eclipse项目的lib中

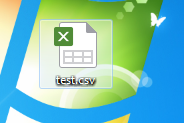
http://www.java2s.com/Code/Jar/o/Downloadopencsv23jar.htm，下载zip，解压出来就可以看到opencsv-2.3.jar

2.桌面制作一个test.csv文件

打开一个记事本，输入如下格式和内容



修改文件后缀，变成test.csv



把这个文件放到项目的Files文件夹下。

3.用opencsv方法去读取csv文件内容

[TestNG简单介绍](http://blog.csdn.net/u011541946/article/details/75070425)

前面文章我们运行测试用例都是在main方法下，有没有什么方法不需要用main方法去执行用例，这就需要单元测试框架的支持，这篇简单介绍TestNG单元测试框架的安装和基本使用。

**1.什么是TestNG**

你们也许很多人听说过TestNG ,Junit,Nunit 这些工具，也肯定想它们到底有什么区别。

1）三者都是独立于测试框架，在市场上都可以免费得到。

2）TestNG和Junit在JAVA上使用，Nunit在C#上使用

3）Junit是Eclipse自带工具，TestNG需要自己安装插件

4）相对TestNG而言，Junit的注释很少

5）Junit不能生成html格式报告，TestNG能生成丰富的html报告

其他对比就不讲了，有兴趣可以到各自官网去查看。

      TestNG是一个测试框架，它能覆盖不同层级的测试，例如，单元测试，集成测试，端对端测试等。我们可以把Selenium和TestNG结合一起，在Eclipse中用Java写TestNG脚本。

**2.TestNG一些特点**

1）注释，很多注释提供

2）支持数据驱动测试（@DataProvider）

3）支持变量

4）自动生成测试报告

5）通过配置testng.xml，我们可以只执行测试失败的用例

6）支持很多IDE工具（Eclipse等）

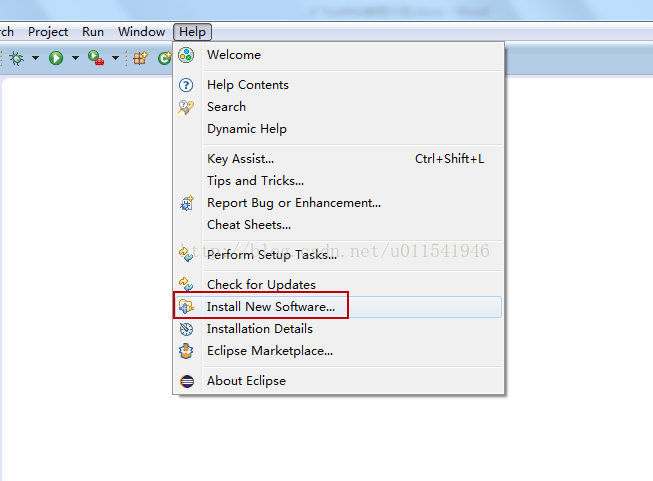
7）默认JDK支持监控运行和日志

8）独立的程序服务器测试方法（API）

**3.TestNG在Eclipse上的安装过程**

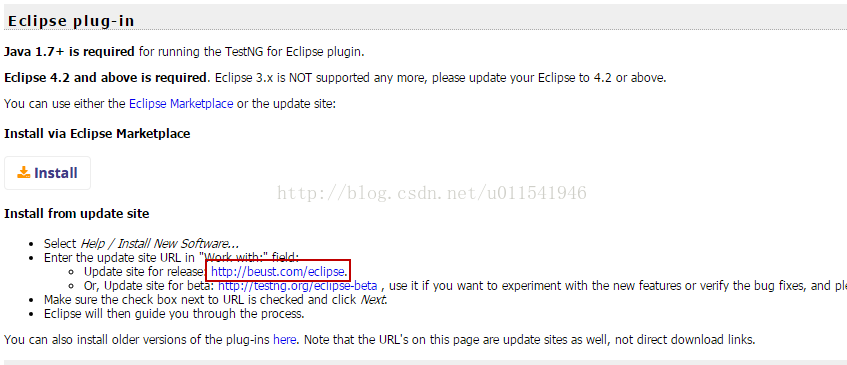
       由于下载站点被墙了，如果你有VPN，你可以安装下面这个正常的流程来安装。

步骤一，打开Eclipse，点击Help-Install New Software

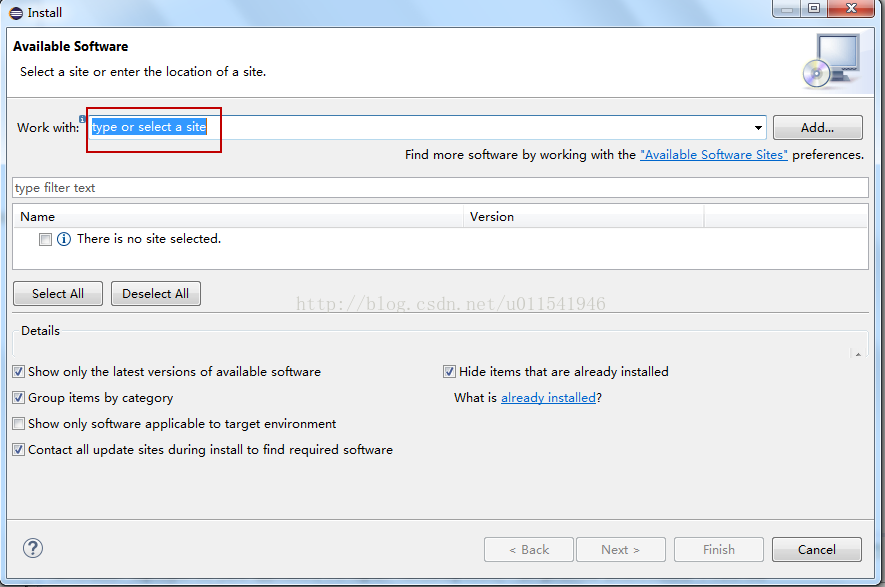


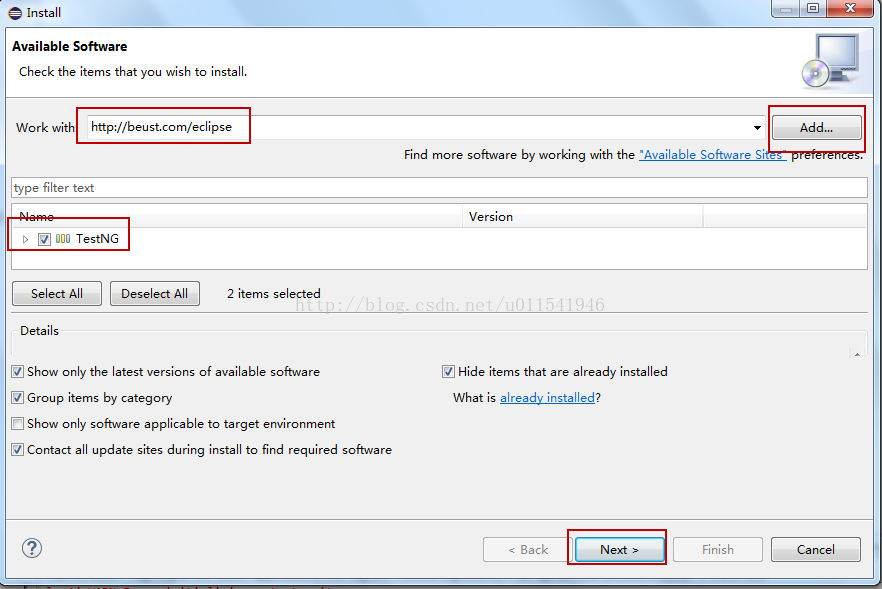
步骤二：打开浏览器，输入testng官网：[http://testng.org/doc/download.html](http://testng.org/doc/download.html" \t "_blank)

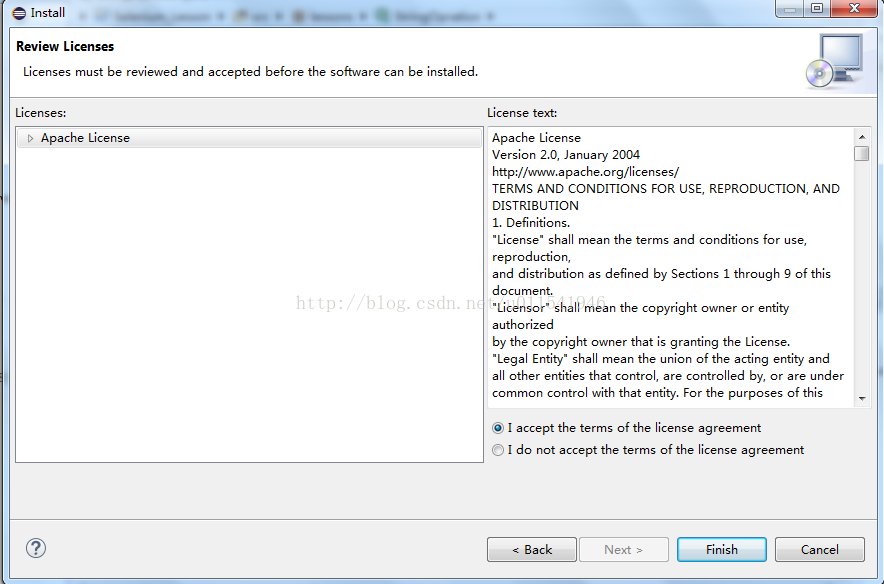
找到如下图界面，拷贝红框url到Eclipse中需要输入站点那个界面上。

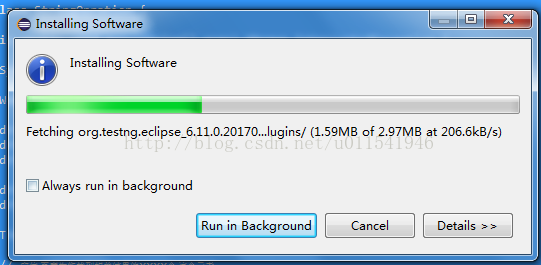


输入testNG的下载站点。





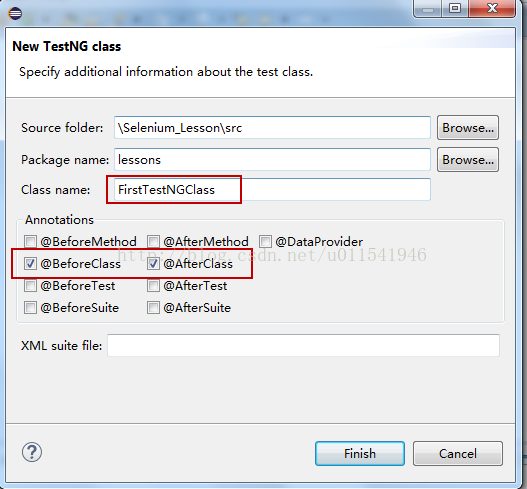




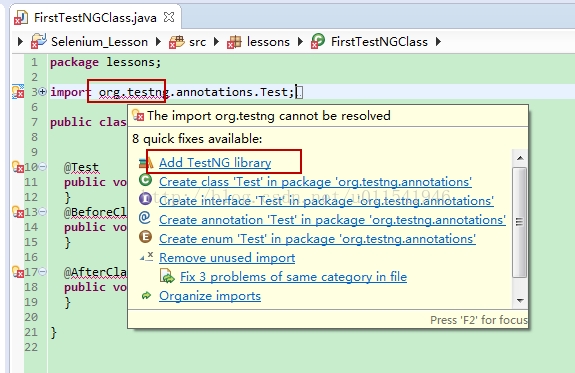
跟着向导完成安装，重启Eclipse，选择一个包鼠标右键，如果看到底部倒数第二项有TestNG这个菜单，说明安装成功。如果你没有VPN环境，你去我网盘[下载文件](http://pan.baidu.com/s/1kUVUlhl" \t "_blank)，解压出来，把这个org.testng.eclipse\_6.11.0.201703011520文件夹放到eclipse的路径\eclipse\plugins

**3.新建一个TestNG类**

在一个包下，鼠标右键，点击TestNG,然后选择Create TestNG class



默认TestNG没有添加到当前项目的lib中，需要安装如图添加。

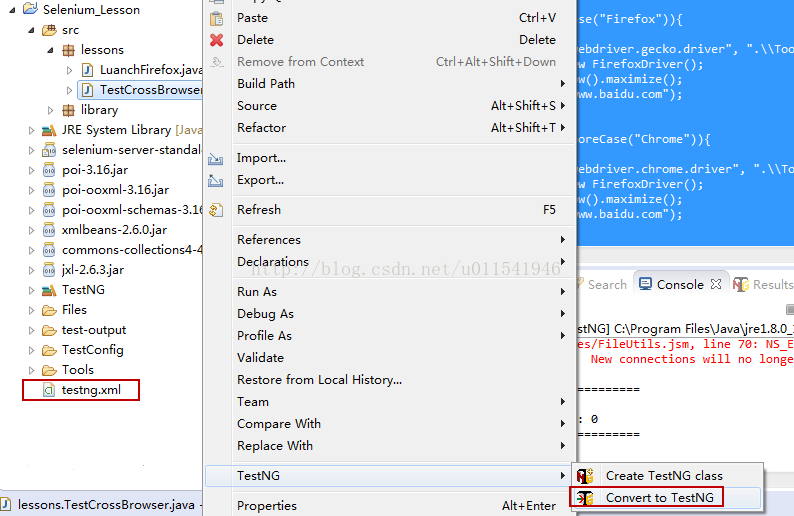


       在BeforeClass中写启动浏览器脚本。在AfterClass中写退出浏览器代码，在@Test中写我们测试脚本。记住，所有有关测试脚本，都需要写到@Test下的方法，也就是说，你写这个函数或者方法是关于测试脚本的，那么你就要在该方法前面加上@Test这个注释，可以添加多个@Test的测试函数。

举例一个完整的基于TestNG的Selenium脚本。注意下面我们把driver定义放到了类的成员变量里，所以在该类下任何地方都能调用driver和保持driver的唯一性。

2.如何创建testng.xml

       右键上面我们创建的TestCrossBrowser.java，选择TestNG-Convert to TestNG,根据提示，走完这个向导。完成后，会在当前项目的根目录生成一个testng.xml的文件。



我们把生成的testng.xml文件进行调整和修改，内容如下。

**[html]** [view plain](http://blog.csdn.net/u011541946/article/details/75109005) [copy](http://blog.csdn.net/u011541946/article/details/75109005)

**<?**xml version="1.0" encoding="UTF-8"**?>**

<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd"**>**

<!-- Suite相关设置，这里paraller是执行的模式，有tests classes methods三种取值，thread-count表示两个线程执行用例 -->

**<**suite name="Suite" parallel="tests" thread-count="2"**>**

**<**test name="Test"**>**

**<**parameter name="Browser" value="Firefox" **/>**

**<**classes**>**

**<**class name="lessons.TestCrossBrowser"**/>**

**</**classes**>**

**</**test**>**

**<**test name="Test1"**>**

**<**parameter name="Browser" value="chrome" **/>**

**<**classes**>**

**<**class name="lessons.TestCrossBrowser"**/>**

**</**classes**>**

**</**test**>**

**</**suite**>**

这里需要注意每个test标签下的name不能重名，下面解释下parallel这个属性，上面提到有三种值。

1）tests

TestNG 会在相同的线程中运行相同<test>标记下的所有方法，但是每个<test>下的方法会运行在不同的线程下。

应用：可以将非线程安全的类放在同一个<test>标签下，并且使其可以利用TestNG多线程的特性的同时，让这些类运行在相同的线程中。（不太理解）

2）methods

TestNG 会在不同的线程中运行测试方法，除非那些互相依赖的方法。那些相互依赖的方法会运行在同一个线程中，并且遵照其执行顺序。

3）classes

TestNG 会在相同线程中相同类中的运行所有的方法，但是每个类都会用不同的线程运行

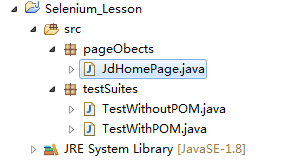
属性 thread-count 允许你为当前的执行指定可以运行的线程数量。

3.运行testng.xml

       右键testng.xml文件，选择Run AS->TestNG Suite，观察运行结果，在控制台输出地方可以看到运行了2个test。这里我们第一次用到testng.xml文件，以后在框架设计会介绍通过这个testng.xml来管理和运行我们成百上千的测试用例。

 页面对象模型（Page Object Model）

页面对象模型（Page Object Model）在Selenium Webdriver自动化测试中使用非常流行和受欢迎，作为自动化测试工程师应该至少听说过POM这个概念。本篇介绍POM的简介，接下来一步一步告诉你如何在你Java+Selenium3自动化测试框架中实现POM。



Selenium中使用POM的特点和优点？

1）  POM是一种设计模式，它能帮我们维护代码和减少代码的重复，这个在自动化测试领域是一个很重要的话题。

2）  在具体实施中，一个页面看做一个页面对象，每个页面写一个类。

3）  你可以把一个页面的元素的全部定位方式和该页面的业务操作方法写在一个类文件，然后在测试脚本类文件去调用页面类的元素和方法。这样就有一个好处，你的软件产某一个界面出现了变化，你只需要修改对应页面的元素定位和相关方法，而不需要去修改你的测试脚本，当然除非你这个页面由于需求问题，之后被砍了，你才要去修改你的测试脚本类。

4）  在POM中，我们把测试脚本和页面对象分层，例如页面对象统一放一个包下面，测试用例脚本统一放另外一个包下，到达测试脚本和页面对象分离。

前面介绍了POM的优点和非POM方式写脚本，这篇介绍利用页面工厂类（page factory）去实现POM，通过查看PageFactory类，我们可以知道它是一个初始化一个页面实例的功能，在实例化该页面对象时候，也会一起实例化该页面的元素定位。直接来看看京东登录的例子，如果用POM实现，在测试脚本中实际代码就2行。

1.在pageObjects包新建一个京东主页类，代码如下

**[java]** [view plain](http://blog.csdn.net/u011541946/article/details/75393386) [copy](http://blog.csdn.net/u011541946/article/details/75393386)

**package** pageObects;

**import** org.openqa.selenium.WebElement;

**import** org.openqa.selenium.support.FindBy;

**public** **class** JdHomePage {

    // 元素定位

    //登录链接

    @FindBy (id="ttbar-login")

    WebElement login\_link;

    //账户登录

    @FindBy (xpath="//\*/div[@class='login-form']/div[2]/a")

    WebElement login\_withAccount;

    //输入用户名框

    @FindBy (id="loginname")

    WebElement inputBox\_username;

    //输入密码

    @FindBy (id="nloginpwd")

    WebElement inputBox\_password;

    //登录按钮

    @FindBy (id="loginsubmit")

    WebElement login\_submitBtn;

    // 业务逻辑和操作方法

    //登录方法

**public** **void** login(String username, String password){

        login\_link.click();

        login\_withAccount.click();

        inputBox\_username.sendKeys(username);

        inputBox\_password.sendKeys(password);

        login\_submitBtn.click();

    }

}

2.在testSuites包下新建一个测试类

**[java]** [view plain](http://blog.csdn.net/u011541946/article/details/75393386) [copy](http://blog.csdn.net/u011541946/article/details/75393386)

**package** testSuites;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.chrome.ChromeDriver;

**import** org.openqa.selenium.support.PageFactory;

**import** org.testng.annotations.BeforeClass;

**import** org.testng.annotations.Test;

**import** pageObects.JdHomePage;

**public** **class** TestWithPOM {

WebDriver driver;

    @BeforeClass

**public** **void** setUp() **throws** Exception{

        System.setProperty("webdriver.chrome.driver", ".\\Tools\\chromedriver.exe");

        driver = **new** ChromeDriver();

        driver.manage().window().maximize();

        driver.get("https://www.jd.com/");

        Thread.sleep(2000);

    }

    @Test

**public** **void** testLogin(){

        JdHomePage hp = PageFactory.initElements(driver, JdHomePage.**class**);

        hp.login("user1", "123456");

    }

}<span style="font-family:Microsoft YaHei;font-size:18px;">

</span>

       处理打开浏览器和打开京东主页操作，实际测试登录的代码就2行对不对。这里我们借助了PageFactory类的方法，在初始化一个页面类的时候，也会一起把该页面定义的元素定位也会初始化，然后通过页面对象调用页面类的页面操作方法。这种方式，我们把元素定位和业务逻辑操作都写在了页面对象类中，测试脚本类，直接调用页面类的方法，这样的测试脚本看起来很简洁，方便阅读和维护，如果登录相关文案发生变更，我们只需要去改登录对应页面的元素定位和业务逻辑，不需要修改这个登录测试类。

[用Log4j实现日志输出](http://blog.csdn.net/u011541946/article/details/76038697)

 本篇来学习一下，Selenium自动化测试中如何输出日志文件。任何软件，都会涉及到日志输出。所以，在测试人员报bug，特别是崩溃的bug，一般都要提供软件产品的日志文件。开发通过看日志文件，知道这个崩溃产生的原因，至少知道触发崩溃的条件是什么。同样在自动化测试框架设计，日志文件输出是不可或缺的。我们习惯通过日志输出来记录我们用例执行的情况。当然测试报告更直观，但是如果出现问题，往往是日志文件更有效。这里我们介绍如何通过Log4j.jar来帮助我们输出一个日志。

       Log4j是开源的jar包，我们可以通过http://www.apache.org/dyn/closer.cgi/logging/log4j/1.2.17/log4j-1.2.17.zip下载最新版本是1.2.17，这个版本是2007年发布，目前没有更新的。下载好了之后，添加到我们的eclipse项目的lib中去。然后，在当前项目新建一个Log4j.properties的文件，内容如下。

**[plain]** [view plain](http://blog.csdn.net/u011541946/article/details/76038697) [copy](http://blog.csdn.net/u011541946/article/details/76038697)

log4j.rootLogger=INFO,CONSOLE,R,HTML,TTCC

// 定义一个appender

log4j.appender.CONSOLE=org.apache.log4j.ConsoleAppender

log4j.appender.R=org.apache.log4j.RollingFileAppender

log4j.appender.TTCC=org.apache.log4j.RollingFileAppender

log4j.appender.HTML=org.apache.log4j.FileAppender

// 定义log文件保存路径，三种日志文件格式

log4j.appender.R.File=./Log/testlog.log

log4j.appender.TTCC.File=./Log/testlog1.log

log4j.appender.HTML.File=./Log/application.html

// 定义layout和pattern

log4j.appender.CONSOLE.layout=org.apache.log4j.PatternLayout

log4j.appender.CONSOLE.layout.ConversionPattern= %5p [%t] (%F:%L)- %m%n

log4j.appender.R.layout=org.apache.log4j.PatternLayout

log4j.appender.R.layout.ConversionPattern=%d - %c -%p - %m%n

log4j.appender.TTCC.layout=org.apache.log4j.TTCCLayout

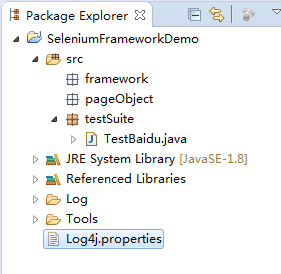
log4j.appender.TTCC.layout.DateFormat=ISO8601

log4j.appender.HTML.layout=org.apache.log4j.HTMLLayout

log4j.appender.HTML.layout.Title=Application log

log4j.appender.HTML.layout.LocationInfo=true

效果如下，并且在当前根路径新建一个Log的文件夹。



写一个测试类，测试日志输出。

**[java]** [view plain](http://blog.csdn.net/u011541946/article/details/76038697) [copy](http://blog.csdn.net/u011541946/article/details/76038697)

**package** testSuite;

**import** java.util.concurrent.TimeUnit;

**import** org.apache.log4j.Logger;

**import** org.apache.log4j.PropertyConfigurator;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.chrome.ChromeDriver;

**public** **class** TestBaidu {

**public** **static** **void** main(String[] args) {

         Logger logger=Logger.getLogger("baidu");

         PropertyConfigurator.configure(".\\Log4j.properties");

         // 打开浏览器

         System.setProperty("webdriver.chrome.driver", ".\\Tools\\chromedriver.exe");

         WebDriver driver = **new** ChromeDriver();

         logger.info("启动浏览器");

         driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

         logger.info("隐式等待10秒");

         driver.get("https://www.baidu.com");

         logger.info("打开百度首页");

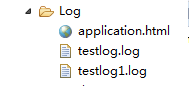
         driver.findElement(By.id("kw")).sendKeys("Selenium");

         logger.info("在搜索输入框输入selenium");

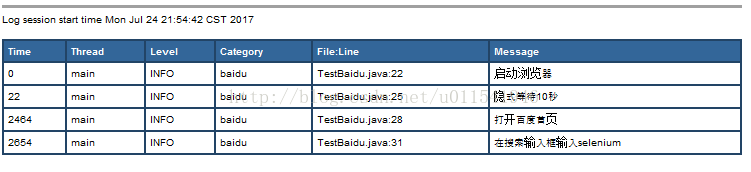
    }

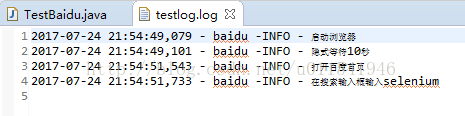
}

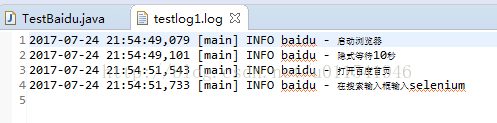
运行完后，右键刷新Log文件夹，可以看到三个日志文件。



分别打开三种文件，截图如下







[如何实现邮件发送测试报告](http://blog.csdn.net/u011541946/article/details/77278837)

  本篇继续回答网友的问题，这个主题是如何通过邮件发送测试报告。通过邮件发送测试报告，这个很重要，也很有必要。不管你使用什么自动化测试框架，都建议添加发送测试报告的功能。先来设计我们的发送邮件的场景：假如我们测试报告是固定的名称，放在固定的路径下。

1.下载java mail api包

       浏览器打开https://mvnrepository.com/artifact/com.sun.mail/javax.mail/1.6.0，点击download jar，把下载下来的jar文件，放在当前项目的tools文件夹下，并添加到当前项目的library里。具体这里不过多描述，前面我们添加了很多次jar包。

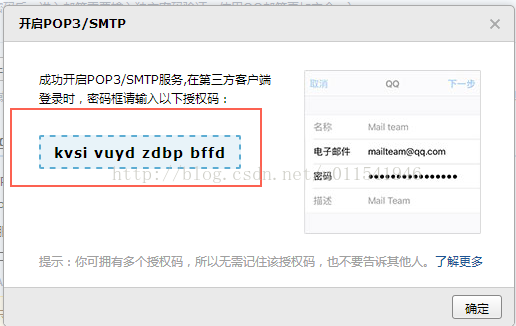
2.在testSuite包下新建一个测试发送邮件的类。

       其实这个发送邮件的类，我们可以单独写成一个静态类，设置一个变量，邮件的附件，也就是我们测试报告文件的路径就是这个变量。今天我们暂时不考虑这么多，只是简单来利用QQ邮箱发送一个测试报告。我的测试报告是TestNG自动化生成的一个html文件。

3.如何开启QQ上的smtp服务

我们在使用QQ的smtp服务器之前，我们需要登录web qq邮箱，开启smtp，具体看下面两张图。





上面红圈的授权码需要记录下来，放到我们接下来的代码中关于身份认证这块，替代password。如果你填写的是你QQ的密码，代码运行后应该报535错误，身份认证失败。

4. 测试发送邮件类的具体代码。