**TEST AUTOMATION What tools are we going to use**

**07/01/2014**

**Authors:**

**Deepa Javvaji**

**Srinivas pasupulati**

**Table of Contents**

[1 Java 3](#_Toc376945731)

[2 Eclipse 3](#_Toc376945732)

[3 TestNG 3](#_Toc376945733)

[4 Apache Ant 5](#_Toc376945734)

[5 XSLT 8](#_Toc376945735)

[6 API 9](#_Toc376945736)

[7 Selenium Web driver 10](#_Toc376945737)

# 1 Java

Available by default

# 2 Eclipse

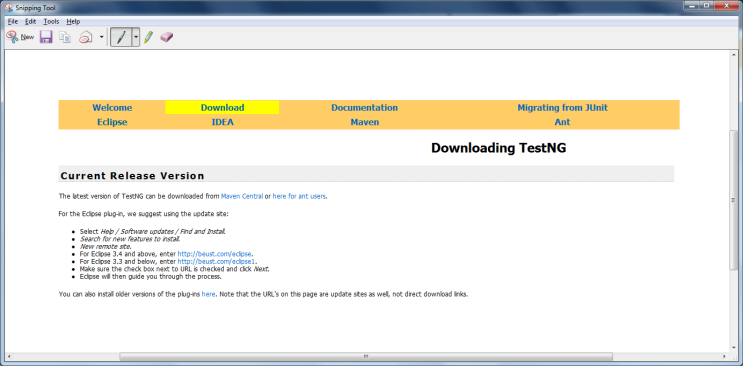
Install Eclipse

[\\sluktsf1\data\restrict\appstore\eclipse\eclipse-jee-juno-SR1-win32.zip](file:///\\sluktsf1\data\restrict\appstore\eclipse\eclipse-jee-juno-SR1-win32.zip)

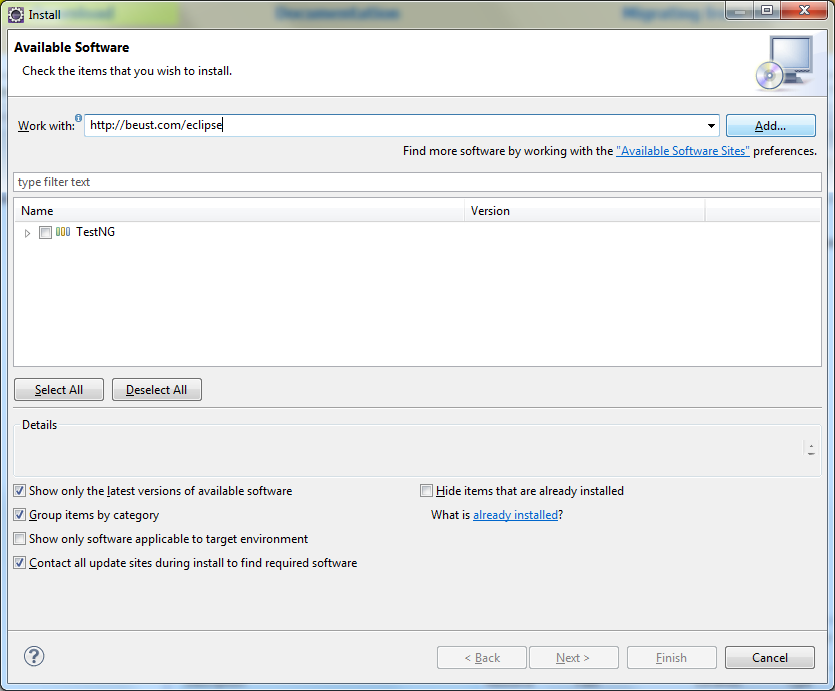
# 3 TestNG

Go to <http://testng.org/doc/download.html>

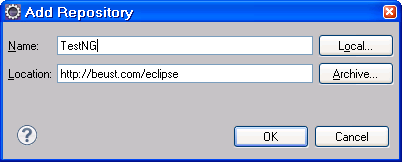
Copy the link <http://beust.com/eclipse>



Go to Eclipse 🡪 Help 🡪 Install New Software 🡪 Paste the link 🡪 Click Add



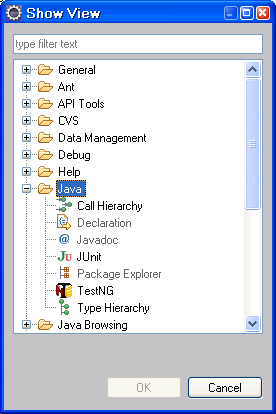
Click OK



Once you have [installed the plug-in](http://testng.org/doc/download.html), restart Eclipse

To check if it is installed correctly

Go to Window 🡪 Show View 🡪 Other 🡪 Java 🡪 TestNG



# 4 Apache Ant

[\\sluktsf1\data\restrict\appstore\ant\apache-ant-1.9.1-bin.zip](file:///\\sluktsf1\data\restrict\appstore\ant\apache-ant-1.9.1-bin.zip)

**Selenium – Use Ant to Generate HTML and XSLT Reports**

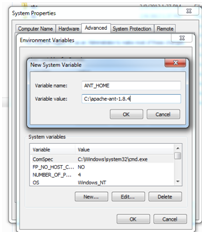
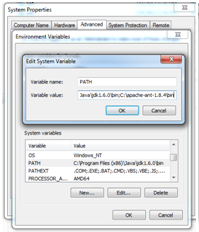
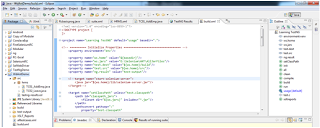
Apache Ant is an open source build tool. A build tool can be used to compile the source code, creating the build artefacts such as JAR, WAR and EAR files. Some of the other usage of ANT is to run unit tests, do the application deployment on containers such as JBoss, Tomcat, WebSphere, WebLogic, GlassFish, etc, and to run Automated Selenium Tests.  
Ant is a powerful build tool and also is very much extensible. There are several open source Ant libraries available which need to be just downloaded, unzipped, and copied (JAR file) into the Ant’s “lib” folder. Once the library file is copied, then we can utilize the “Tasks” in the Ant’s build.xml file. Ant’s “contrib library, jsch library, etc” are the best examples for such libraries.

**Install and configure ANT**  
1. Download ANT from Apache web site – ANT download link  
2. Unzip the ant and copy to C:\Ant folder  
3. Now we need to configure environment variables: ANT\_HOME   
4. Navigate to environment variables and click on new button.

Go to My computer 🡪 Properties

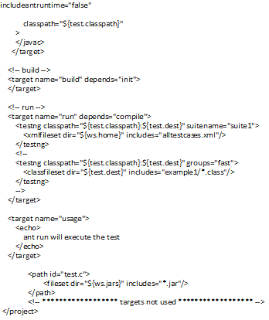
Click on Advanced 🡪 Environment Variables

Create a System Variable for ANT

5. Enter Variable name as ANT\_HOME and Variable Value as C:\apache-ant-1.9.1.Then click on OK button  
[](http://2.bp.blogspot.com/-KJdbNcJIi08/URoNKlytgwI/AAAAAAAAAFk/58Fk5fcbwxU/s1600/ANT_HOME.png)  
6. Now select the PATH and click on EDIT button  
7. Now give the semicolon (;) at the end and add ANT path up to bin folder  
[](http://3.bp.blogspot.com/-DhVg13zTN2I/URoNbwAVOhI/AAAAAAAAAFs/-CPGtSjggVE/s1600/PATH.png)  
8. Now click on OK button and go to command prompt and type ANT.  
“It will give message as build.xml not fount”  
9. Now go to eclipse and add below build.xml file to your selenium project.  
[](http://1.bp.blogspot.com/-cn6fLEeYhwk/URoNvXRbHoI/AAAAAAAAAF0/dW83SE6YbMA/s1600/build.XML+File.png)

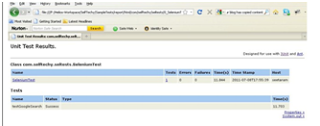
Build.xml file code

[](http://3.bp.blogspot.com/-yv-FjhSa82I/URoRTS9hlyI/AAAAAAAAAGU/gCo_OjwueS0/s1600/build.XML+Code.png)

[](http://2.bp.blogspot.com/-H0K0u6qHZic/URoRb4yiqeI/AAAAAAAAAGc/IYhTgoVR9kU/s1600/build.XML+Code1.png)

10. In this Build.xml file give the path where you copied all the required jar file.(Copy all the required jar files in any of the folder and mention that path)

11. Now go to command prompt and type > ant clean  
It will clean all the required files related to ant.  
12. Now type > ant compile  
It will compile your code  
13. Now type >ant run

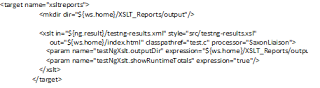
[](http://2.bp.blogspot.com/-YBhDw1xJZ-g/URoN9x1kOjI/AAAAAAAAAF8/ZlHnrKqUbkc/s1600/ANT-TestNG-HTML+Report.png)

It will run your program and will generate HTML reports.

# 5 XSLT

**Generating Selenium reports using TestNG-xslt through Ant**   
TestNG-xslt generates user friendly reports using the TestNG results output (testng-results.xml). Its uses the pure XSL for report generation and Saxon as an XSL2.0 implementation.

**For generating testng-xslt report for your project do the following:**  
1. Download the testng-xslt from  
http://code.google.com/p/testng-xslt/  
2. Unzip and copy the testng-results.xsl from the testng-xslt folder(testng-xslt-1.1\src\main\resources) to your own project folder.  
3. Now copy the saxon library from (testng-xslt-1.1\lib\saxon-8.7.jar)to where you created a folder for which you copied all the JAR files to run ANT.  
4. Modify your build.xml of ant and add the following target to it.

[](http://4.bp.blogspot.com/-xD0coAziy6A/URoRq4uyp-I/AAAAAAAAAGk/eExJBUaOKo0/s1600/XSLT+code.png)

5. Now go to command prompt and follow below steps to generate XSLT reports.  
6. Now type > ant clean  
It will clean all the required files related to ant.  
7. Now type > ant compile  
It will compile your code  
8. Now type >ant run   
It will run your program.  
9. Now type >ant xsltreports  
It will generate xslt reports as below

[](http://1.bp.blogspot.com/-aTX0qUA5IgA/URoOQocwmqI/AAAAAAAAAGE/hAateaXS-Fw/s1600/ANT-TestNG-XSLT+Report.png)

# 6 API

2 Application Programming Interfaces (API)

1. Apache POI – the Java API for Microsoft Documents

[\\sluktsf1\data\restrict\appstore\poi\poi-bin-3.9-20121203.zip](file:///\\sluktsf1\data\restrict\appstore\poi\poi-bin-3.9-20121203.zip)

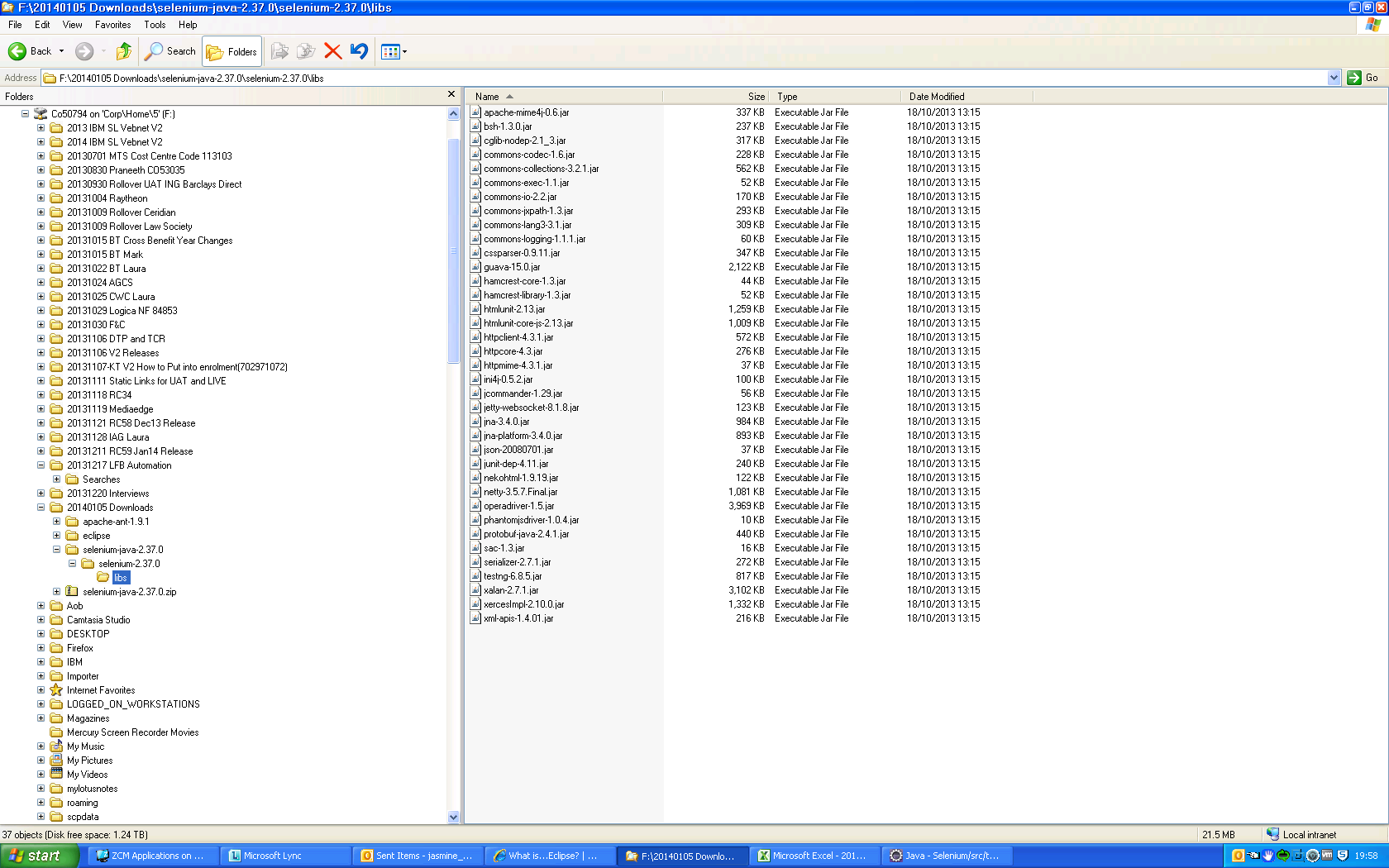
1. LifelensWebElements – Designed to identify lifelens web elements

# 7 Selenium Web driver

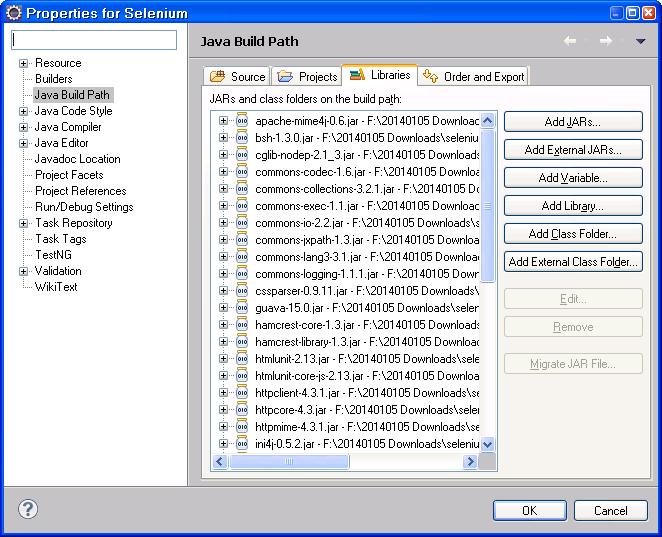
Portable Software Testing Framework for web applications

[\\sluktsf1\data\restrict\appstore\selenium\selenium-java-2.37.0.zip](file:///\\sluktsf1\data\restrict\appstore\selenium\selenium-java-2.37.0.zip)

Copy the Jar files to Eclipse.

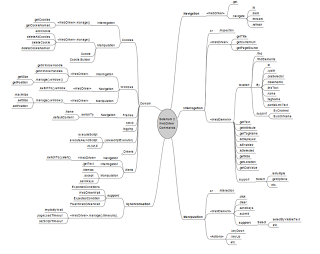


Go to Eclipse 🡪 File 🡪 Properties 🡪 Java Build Path



Selenium Web Driver Command List

**Command                    Description**  
**driver.get("http://www.google.com");** To open an application  
**driver.findElement(By.id("passwd-id"));** Finding Element using Id  
**driver.findElement(By.name("passwd"));** Finding Element using Name  
**driver.findElement(By.xpath("//input[@id=’passwd-id’]"));** Finding Element using Xpath  
**element.sendKeys("some text");** To type some data  
**element.clear();** clear thecontents of a text ﬁeld or textarea  
**driver.findElement(By.xpath("//select"));** Selecting the value  
**select.findElements(By.tagName("option"));** Selecting the value  
**select.deselectAll();** This will deselect all OPTIONs from the ﬁrst SELECT on the page  
**select.selectByVisibleText("Edam");** select the OPTION withthe displayed text of “Edam”  
**findElement(By.id("submit")).click();** To click on Any button/Link  
**driver.switchTo().window("windowName");** Moving from one window to another window  
**driver.switchTo().frame("frameName");** swing from frame to frame (or into iframes)  
**driver.switchTo().frame("frameName.0.child");** to access subframes by separating the path with a dot, and you can specify the frame by itsindex too.  
**driver.switchTo().alert();** Handling Alerts  
**driver.navigate().to("http://www.example.com");** To Navigate Paeticular URL  
**driver.navigate().forward();** To Navigate Forward  
**driver.navigate().back();** To Navigate Backword  
**driver.close()** Closes the current window  
**driver.quit()** Quits the driver and closes every associated window.  
**driver.switch\_to\_alert()** Switches focus to an alert on the page.  
**driver.refresh()** Refreshes the current page.  
**driver.implicitly\_wait(30)** Amount of time to wait  
**driver.set\_script\_timeout(30)** The amount of time to wait  
**driver.get\_screenshot\_as\_file('/Screenshots/foo.png')** The full path you wish to save your screenshot to  
**driver.get\_screenshot\_as\_base64()** Gets the screenshot of the current window as a base64 encoded string which is useful in embedded images in HTML

[](http://3.bp.blogspot.com/-RonvzEf7184/ULXoenIkIeI/AAAAAAAAAEo/2TsgeZidCvY/s1600/WebDriverCommands.png)