Selenium Automation set up with TestNG and Eclipse- A Beginners Guide

Authors: Eevuri Sri Harsha, Ranjani Sivagnanam

Sri Harsha is working as an Associate Software Engineer (QA) for IBM Policy Atlas team under ECM. Reach out to her at eeharsha@in.ibm.com

Ranjani, is working as a Staff Software Engineer (QA) for IBM Policy Atlas team under ECM. Reach out to her at ranjani.sivagnanam@in.ibm.com

Abstract: This article is a guide for the beginners when they use Selenium, Testing and Eclipse IDE for automation testing

1. Introduction

Selenium IDE/Web driver has becoming a very popular testing tool as it is an open source. This is a GUI Automation tool. This is a beginner's guide to install and use Selenium with Testing and Eclipse IDE. This gives a step by step installation guide to Kick-start an Automation journey.

This document will describe one of the ways of setting up the environment automating Web Applications. Described here will be a sample test script, object repository build up using XPath or CSS, reporting configuration for the test run.

1.1 Install Firefox in Windows:

1. Visit the Firefox download page in any browser (e.g. Microsoft Internet Explorer). The page will automatically recommend the best version(s) of Firefox for you.
2. Firefox Download Page - Windows-> Download 32-bit-Firefox Setup 3.6.0.exe
3. Click on the green download link to download the Firefox installer. Start the process by clicking Run.
4. Install Firefox 3.6 by a double click on Firefox icon -> click on next buttons, install button, Launch Firefox by click on finish button.
5. See the below screens to follow up.

Figure 1

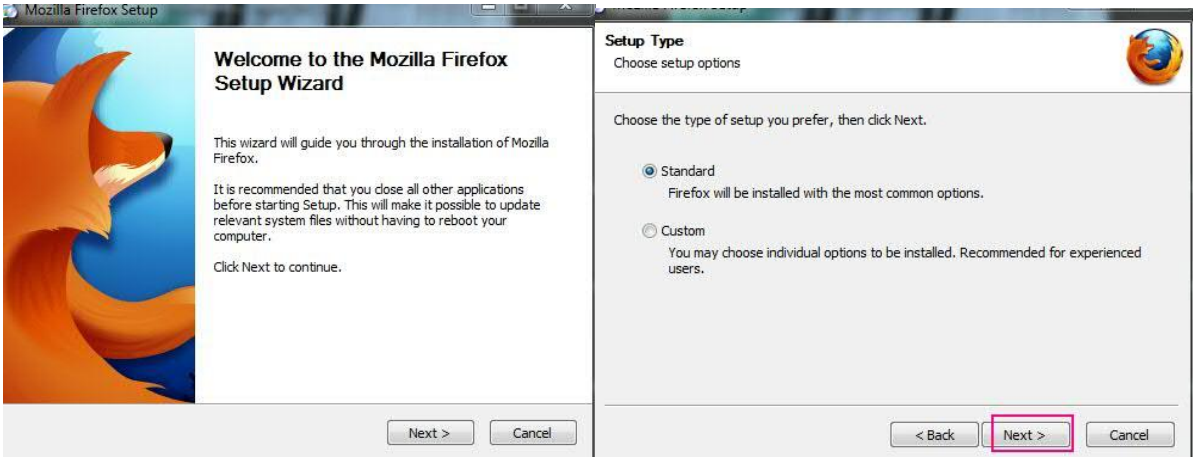
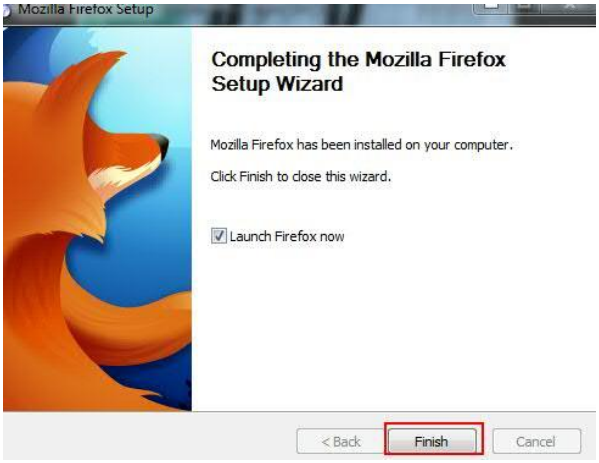


Figure 2



1.2 Install DOM Inspector:

1. Open Firefox 3.6 and go to Tools -> Add ons.
2. In Search tab give DOM Inspector: click on install the latest version.
3. Go to Tools-> Add ons -> Search -> give Inspect this as search term ad click on install the latest version.
4. See the below screens to follow up.

Figure 3

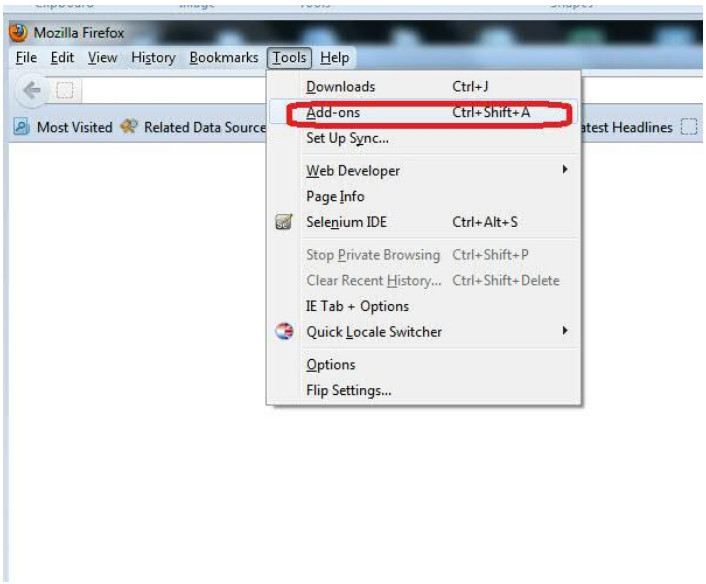
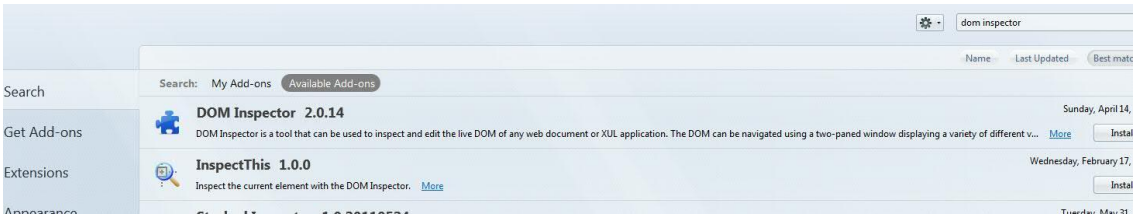


Figure 4



1.3 Install JDK 1.7:

1. Download Windows JDK 1.7 from http://www.oracle.com/technetwork/java/javase/downloads/index.html
2. Install JDK 1.7

2. Selenium Introduction :

The Selenium-IDE (Integrated Development Environment) is the tool you use to develop your Selenium test cases. It's an easy-to-use Firefox plug-in and is generally the most efficient way to develop test cases. It also contains a context menu that allows you to first select a UI element from the browser's currently displayed page and then select from a list of Selenium commands with parameters pre-defined according to the context of the selected UI element. This is not only a time-saver, but also an excellent way of learning Selenium script syntax.

2.1 Install Selenium IDE:

1. Open Firefox 3.6 and goto Tools -> ADD ONS.
2. In Search tab give Selenium IDE : click on install the latest version.
3. After install Selenium IDE -> Go to Tools-> Add Ons-> Extensions sub tab -> Selenium IDE .
4. Click on options in Selenium IDE.
5. Identify below tabs to work on Different platform Environments and to see default settings

Figure 5

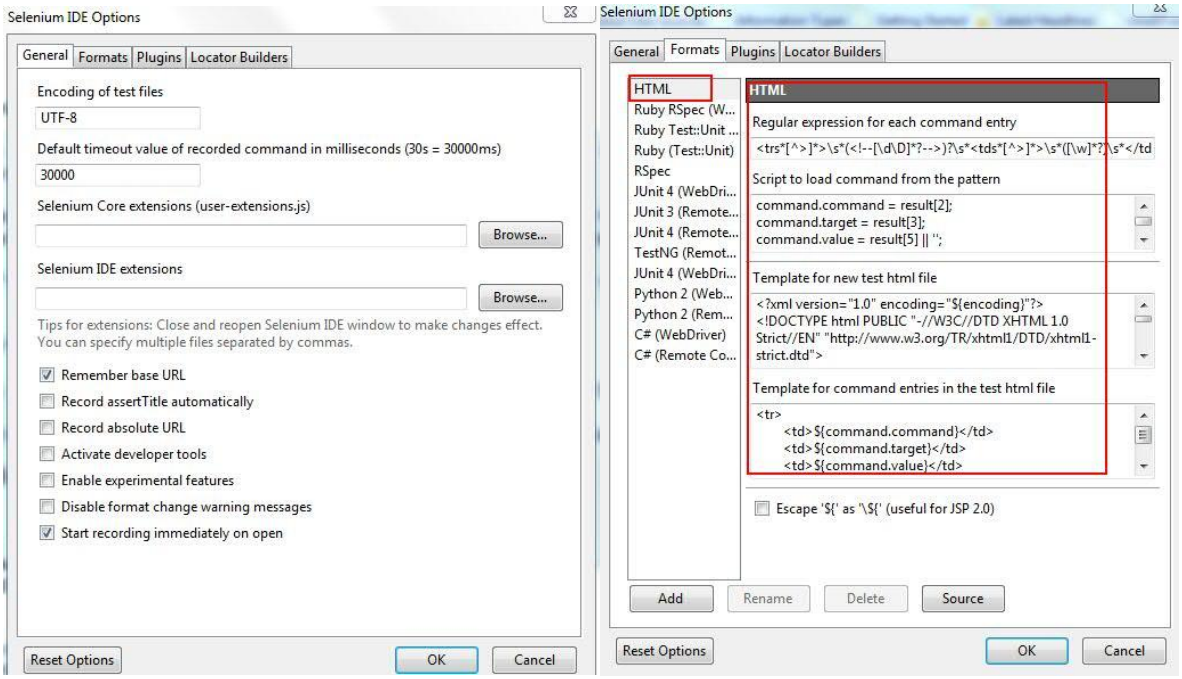
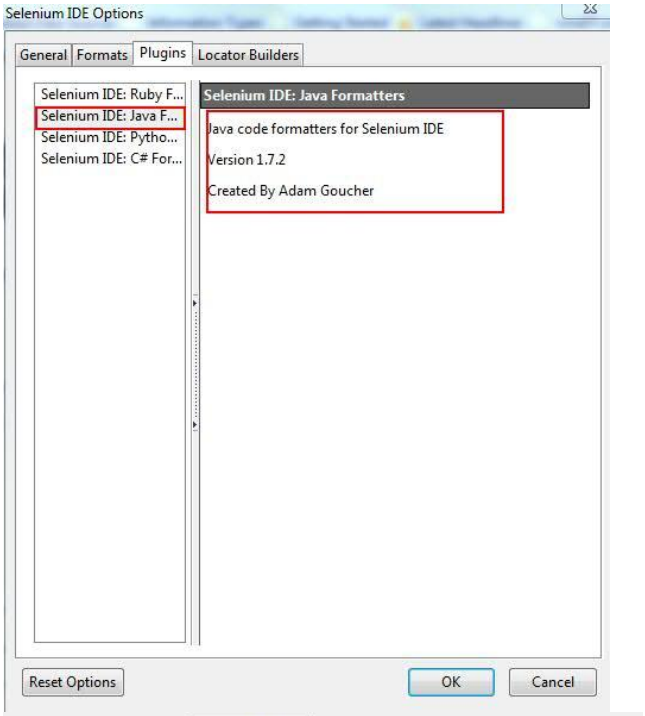
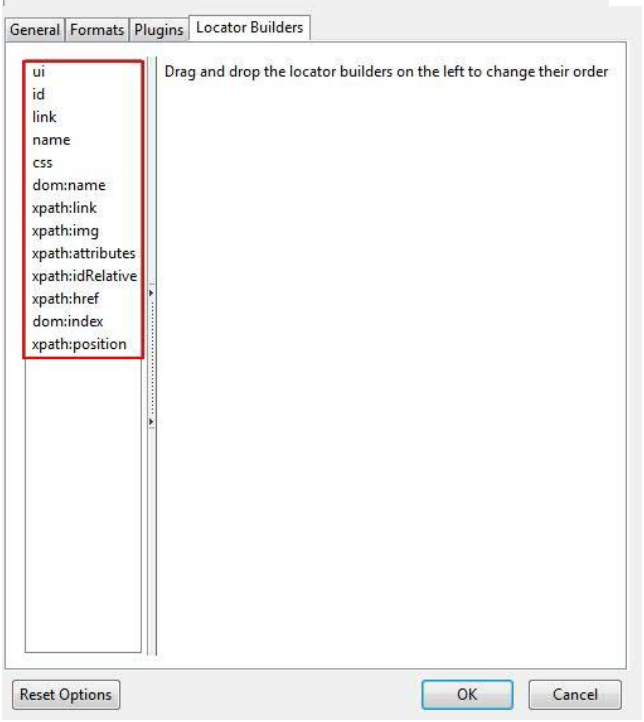

Selenium IDE Options
General
Formats
Plugins
Locator Builders
Encoding of test files
UTF-8
Default timeout value of recorded command in milliseconds (30s = 30000ms)
30000
Selenium Core extensions (user-extensions.js)
Browse...
Selenium IDE extensions
Browse...
Tips for extensions: Close and reopen Selenium IDE window to make changes effect.
You can specify multiple files separated by commas.
Remember base URL
Record assertTitle automatically
Record absolute URL
Activate developer tools
Enable experimental features
Disable format change warning messages
Start recording immediately on open
Reset Options
OK
Cancel
Selenium IDE Options
General
Formats
Plugins
Locator Builders
HTML
Ruby RSpec (W...
Ruby Test::Unit ...
Ruby (Test::Unit)
RSpec
JUnit 4 (WebDri...
JUnit 3 (Remote...
JUnit 4 (Remote...
TestNG (Remot...
JUnit 4 (WebDri...
Python 2 (Web...
Python 2 (Rem...
C# (WebDriver)
C# (Remote Co...
HTML
Regular expression for each command entry
<trs*[^>]*>\s*(<!--[\d\D]*?-->)?\s*<tds*[^>]*>\s*([\w]*?)\s*</td
Script to load command from the pattern
command.command = result[2];
command.target = result[3];
command.value = result[5] || '';
Template for new test html file
<?xml version="1.0" encoding="${encoding}"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
Template for command entries in the test html file
<tr>
<td>${command.command}</td>
<td>${command.target}</td>
<td>${command.value}</td>
Escape '${' as '\${' (useful for JSP 2.0)
Add
Rename
Delete
Source
Reset Options
OK
Cancel

Figure 6


Selenium IDE Options
General
Formats
Plugins
Locator Builders
Selenium IDE: Ruby F...
Selenium IDE: Java F...
Selenium IDE: Pytho...
Selenium IDE: C# For...
Selenium IDE: Java Formatters
Java code formatters for Selenium IDE
Version 1.7.2
Created By Adam Goucher
Reset Options
OK
Cancel


General
Formats
Plugins
Locator Builders
ui
id
link
name
css
dom:name
xpath:link
xpath:img
xpath:attributes
xpath:idRelative
xpath:href
dom:index
xpath:position
Drag and drop the locator builders on the left to change their order
Reset Options
OK
Cancel

2.2 Selenium Web Driver:

The biggest change in Selenium recently has been the inclusion of the Web Driver API. Driving a browser natively as a user would either locally or on a remote machine using the Selenium Server it marks a leap forward in terms of browser automation

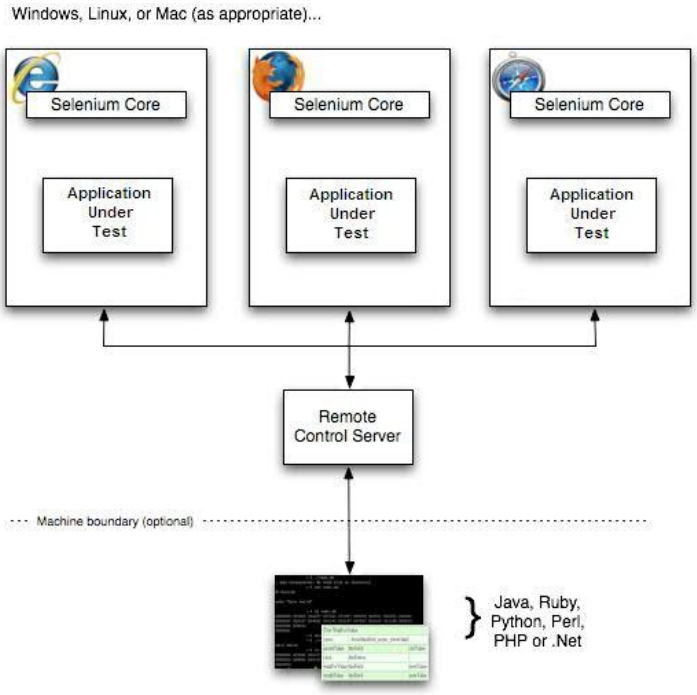
Selenium 1.0 + Web Driver = Selenium 2.0

1. Web Driver is designed in a simpler and more concise programming interface along with addressing some limitations in the Selenium-RC API.
2. It drives the browser much more effectively and over comes the limitations of Selenium 1.x which affected our functional test coverage, like the file upload or download, pop-ups and dialogs barrier.
3. To Support Web Driver we need to keep selenium standard 2.33 jar file in Project class path.

Install selenium-server-standalone-2.33.0.jar file to support Selenium Remote Web driver:

1. Download selenium-server-standalone-2.33.0.jar from web.
2. We need to keep selenium-server-standalone-2.33.0.jar in created Selenium Project Class path.
3. This jar file will support handling Remote browsers and web drivers, different browsers.
4. See the below screen for RC components.

Figure 7



2.3 Install DOM Inspector to inspect web elements using Xpath:

The web contains static and dynamic objects/elements. For static elements Ids are same at any time.

Ex: Button , Text box , Text Area ids .Using Inspect this element we can identify element id xpath.

1. While using Selenium IDE, sometimes the dynamic elements cannot be identified. These have to be identified at run time.
2. Static elements are like button, text box, text area. For static elements Ids are same at any time.
3. But for Dynamic elements Ids will change at execution time.
4. DOM Inspector is useful to identify dynamic elements with common id at any time.
5. Here Xpath is creating id to identify dynamic element.
6. Example : Identify one checkbox id among multiple check boxes ids.
7. Check box id will change dynamically. That time Xpath using Dom inspector is useful to identify the Id.

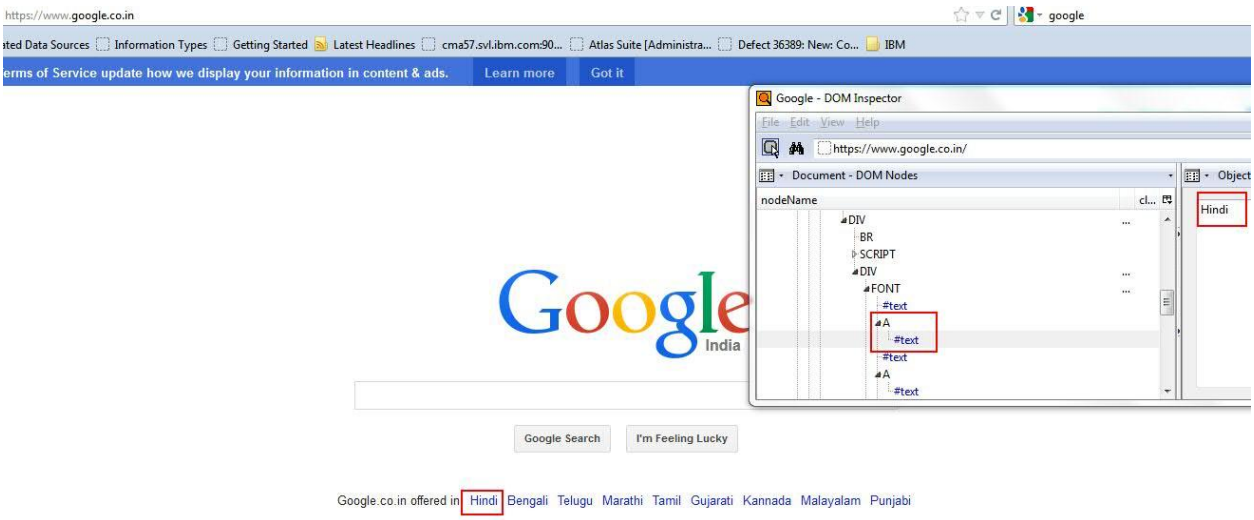
To explain how we use XPATH or CSS, here is an example:

On the home page of Google India search page, we see list of languages which can be used to convert Google page into particular language.

The example is covering Google English to Google Hindi.

1. Open Firefox.
2. Open www.google.co.in page.
3. Open DOM Inspector from Tools-> Web developer -> DOM inspector -> click on cursor icon of Dom inspector and click on any web element in Firefox to get id path.
4. See below screen for XPATH. Here its clicking on Hindi sub link and getting xpath.

Figure 8

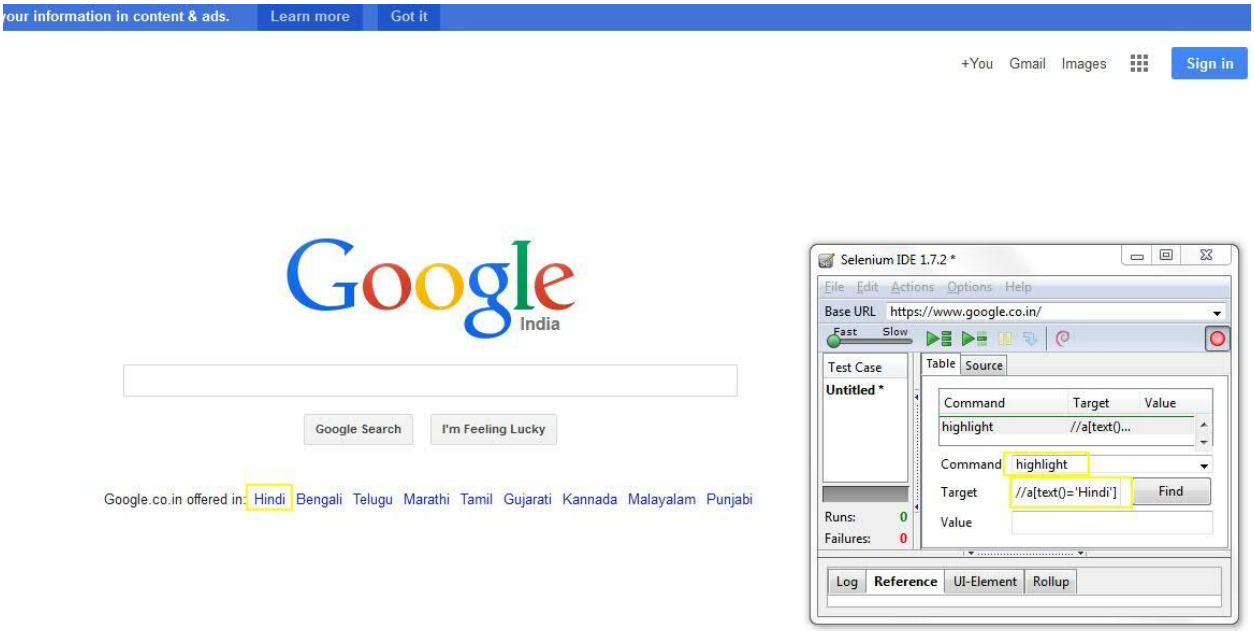


1. XPATH generated is : //a[text()='Hindi']
2. Open Selenium IDE and give command as “highlight” or “Click”
3. Target is generated XPATH: //a[text()='Hindi']
4. Click on Find button in selenium IDE.

Results its highlights and click the "Hindi" Language.

See below screen.

Figure 9



2.3.1 Few points to note about XPATH:

An XPATH axis is a path through the node tree making use of particular relationship between nodes. We use the "child::*" axis and the "attribute::*" axis all the time but mostly their short form: "*" and "@*". The other axes are used far less often.

Follow below website to get good command on XPATH creation.

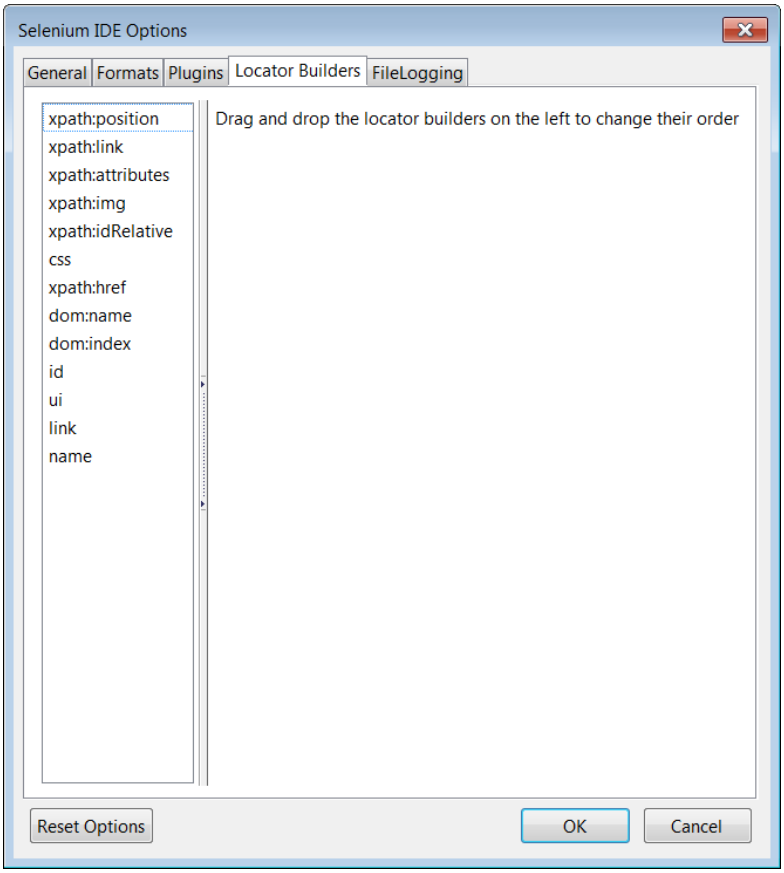
http://www.xmlplease.com/axis

CSS

Another way of locating the objects is by using CSS Selectors instead of XPATH:

Open the Selenium IDE > Options > Options> Locator Builders > Make sure CSS is present in the list

Figure 10

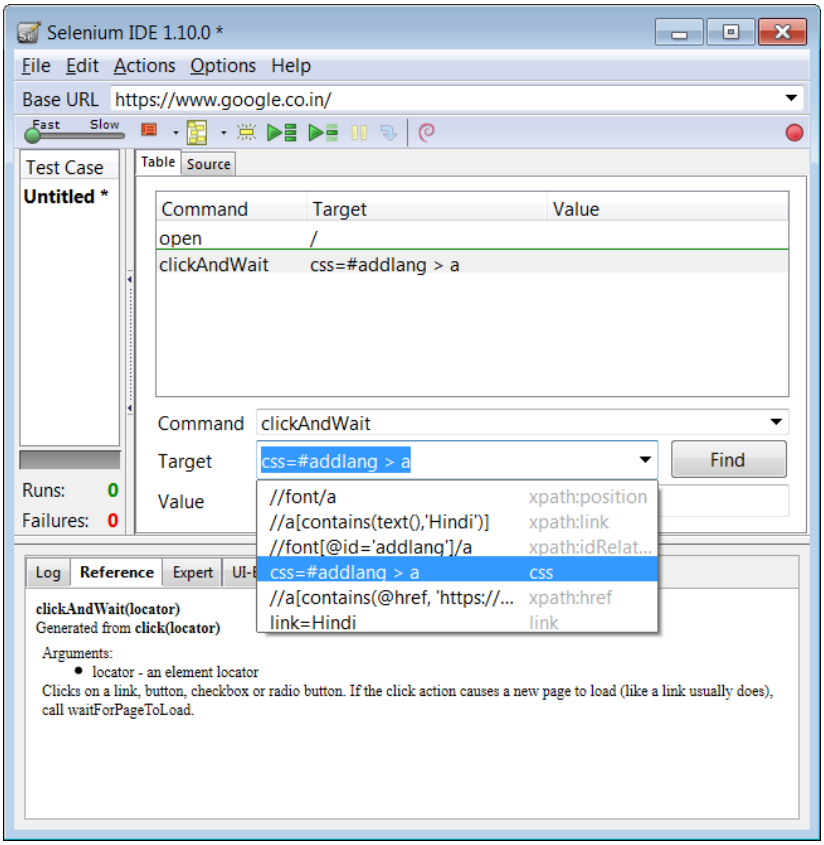


For this we can use DOM Inspector to get the elements, its attributes and its hierarchy

To locate a text field >

1. Open the www.google.co.in
2. Open the Selenium IDE
3. Record the Clicking on the Hindi link
4. Stop recording
5. Select the Target as css

Figure 11

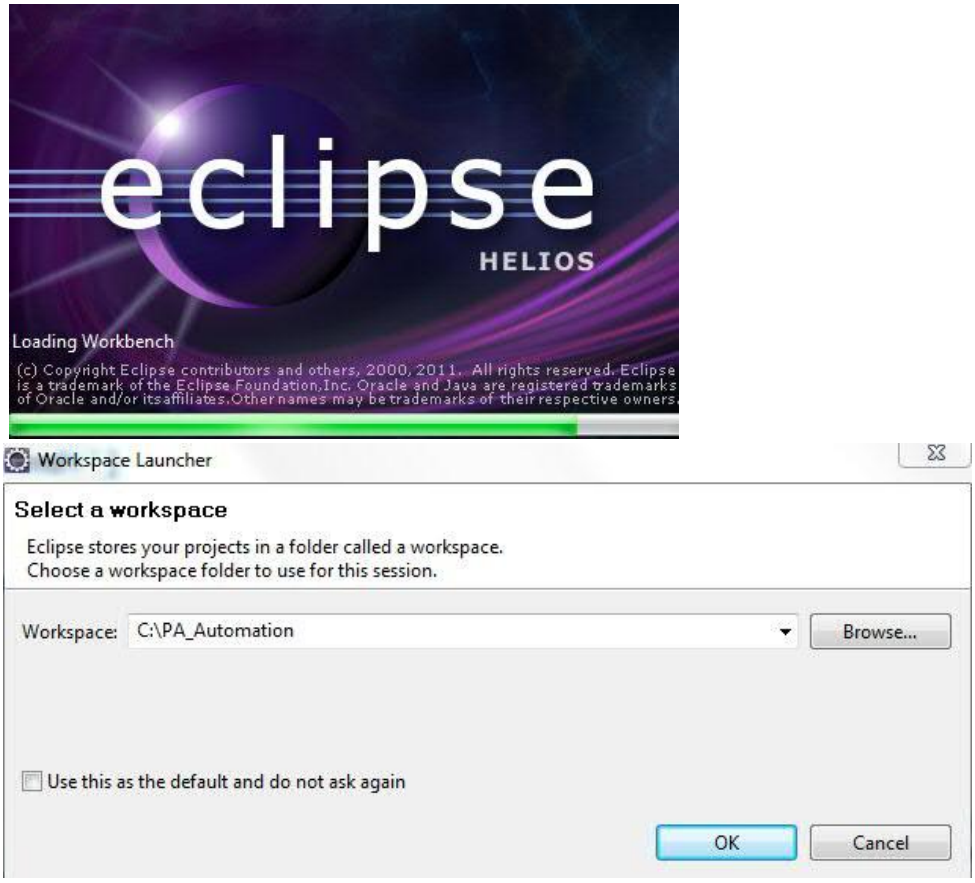


Please refer the site http://www.w3schools.com/cssref/css_selectors.asp to learn about CSS selectors

3.Install eclipse.exe:

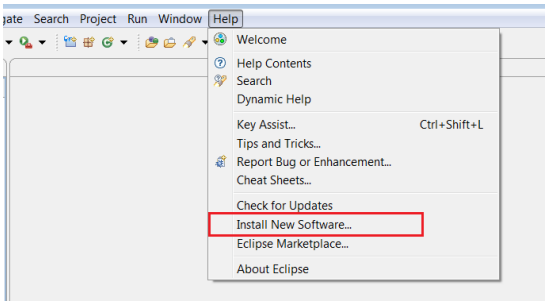
1. Download Eclipse Standard 4.3.1 from web.
2. Install Eclipse by double click on eclipse.exe.
3. Give project location as Workspace.
4. See the screens to install eclipse

Figure 12



3.1 Install TestNG

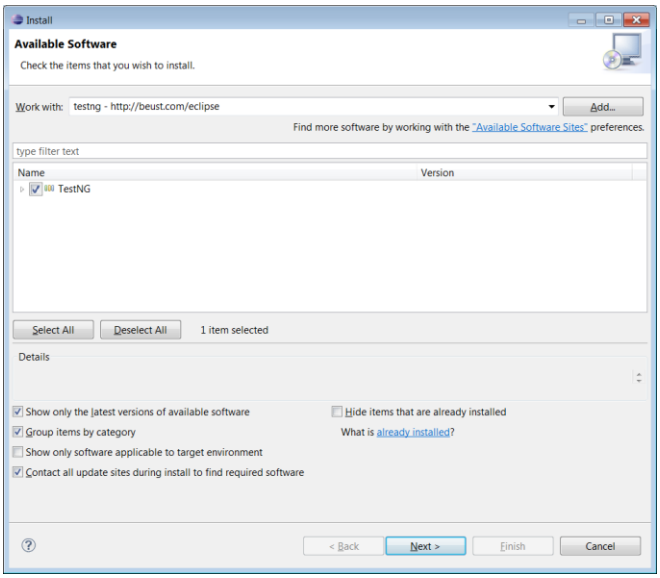
1. Open the eclipse.exe
2. Goto Help > Install New Software



3. Give the url http://beust.com/eclipse in the Work with field
4. Click on Add

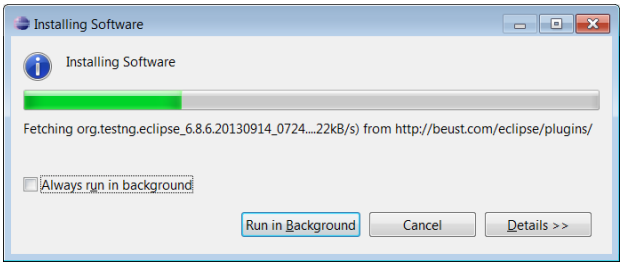
5. TestNG will be listed
6. Check the checkbox against TestNG
7. Click Next

Figure 13



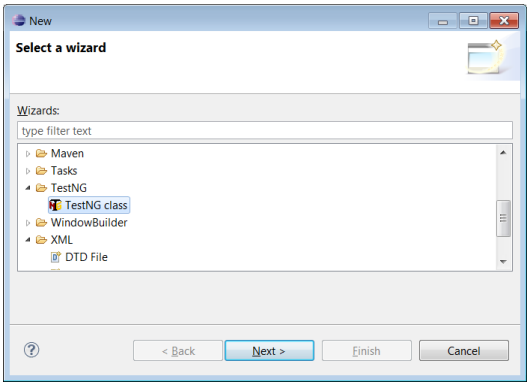
8. Click Next until the License Agreement screen appears
9. Click “I agree” option
10. Installing Software pop up comes up

Figure 14



11. Once the installation is complete, open File> New> Other(or Ctrl + N) > Testing Class must be listed in the window

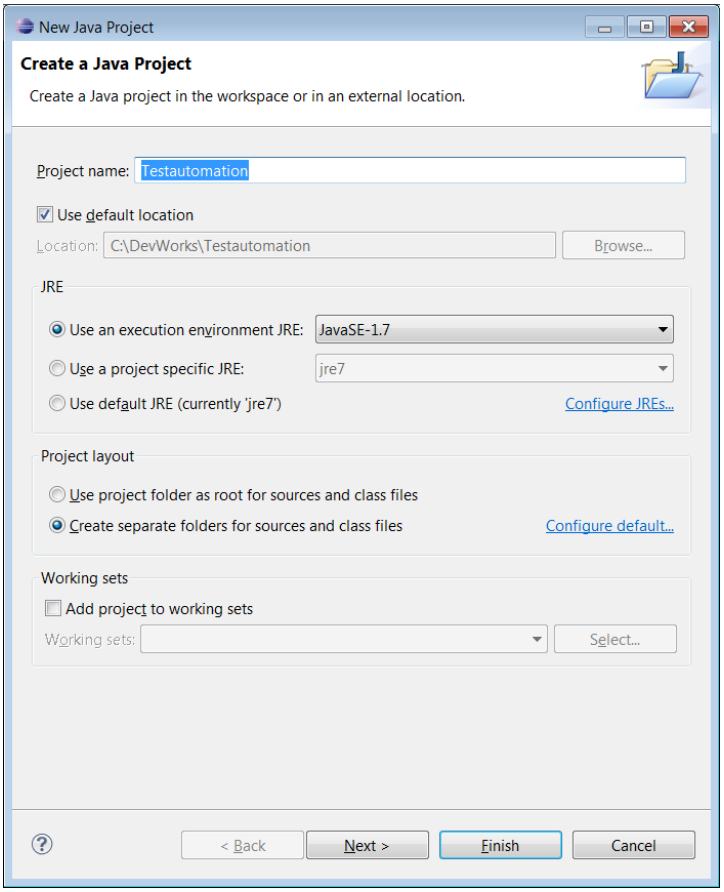
Figure 16



3.2 Creating a Java project:

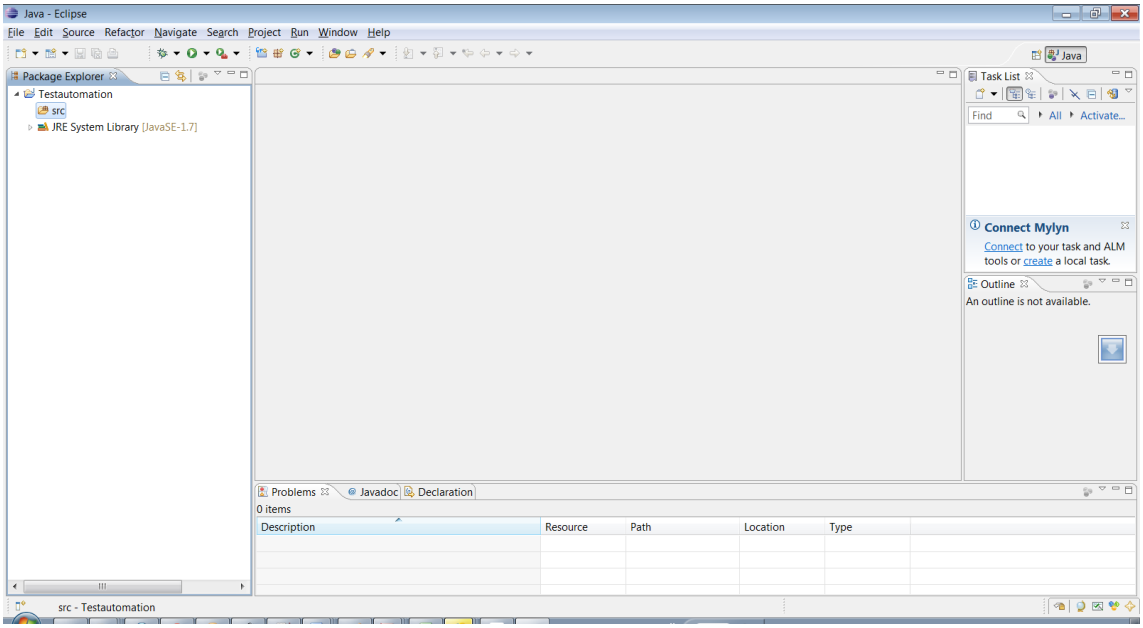
1. Goto to Workbench
2. Click on File > New > Java Project

Figure 17



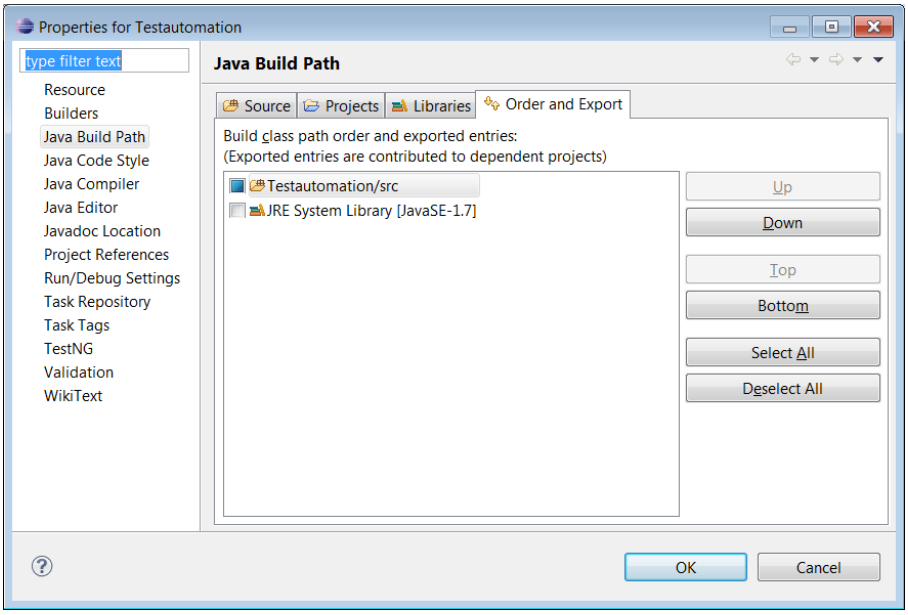
3. Choose your JRE version, latest being JavaSE – 1.7
4. Click on Finish

Figure 18



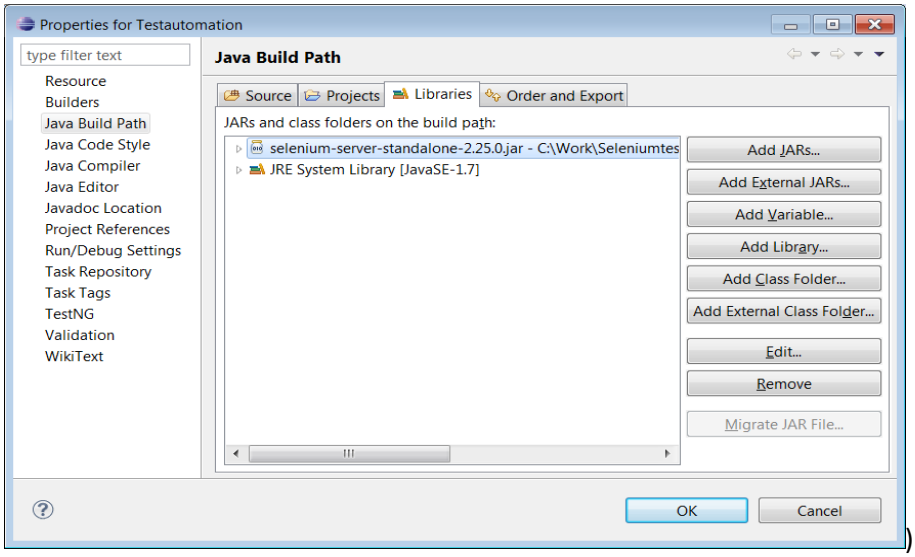
5. Add the Selenium jar to the library (File> Properties > Goto Libraries > Add External JARs .. > Select the selenium-server-standalone-2.25.0.jar or the latest selenium jar

Figure 19



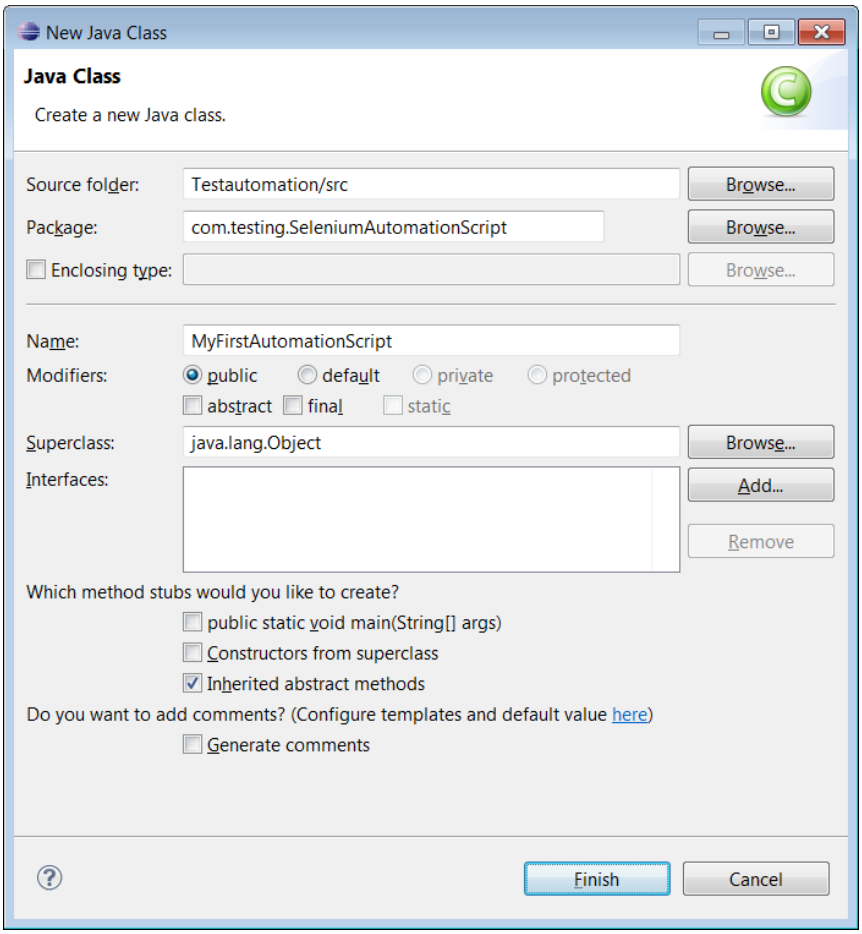
It should be added

Figure 20



6. Create a new Class MyFirstAutomationScript.java

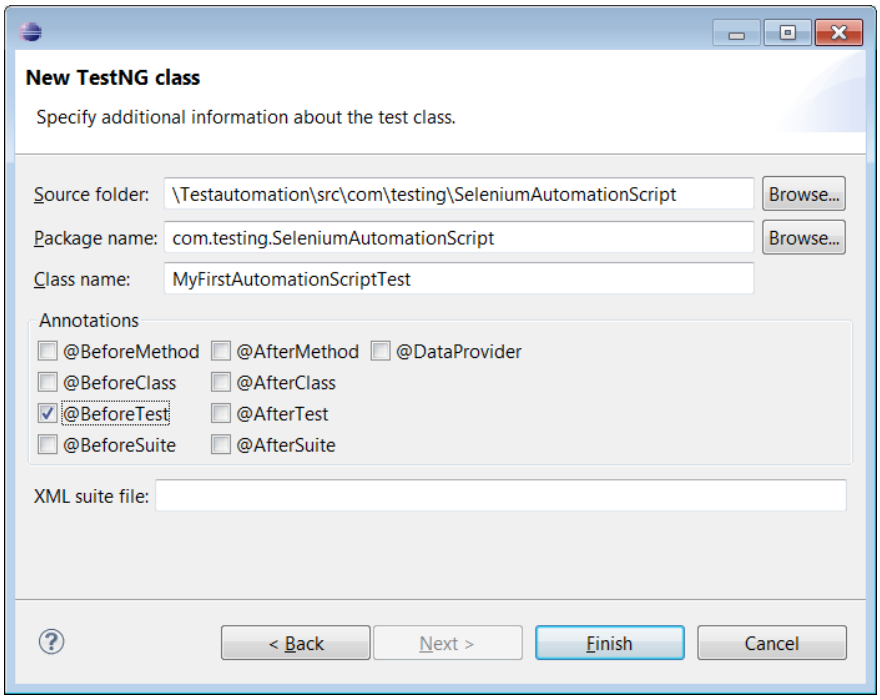
Figure 21



7. Import Testing packages

a. Click on MyFirstAutomationScript.java
b. Click New
c. Click TestNG class
d. Click @BeforeTest
e. Finish

Figure 22



8. Add the @BeforeTest method
9. Add the @AfterTest method
10. Add the test you have to do in the @Test annotation

```
import org.testng.annotations.Test;
import org.testng.annotations.BeforeTest;
import org.testng.annotations.AfterTest;
import org.openqa.selenium.By;
import org.seleniumhq.*;
import com.thoughtworks.selenium.Selenium;
import com.beust.testng.*;
import static org.testng.Assert.*;


public class MyFirstAutomationScript{
	@Test
	public void firstTest() {
		//Write the selenium code to test a particular functionality
	}

	@BeforeTest
	public void beforeTest() {
```

```
            //Write the setup steps like starting the Selenium server, bringing
    the application url up
      }

      @AfterTest
      public void afterTest() {
            //Write the cleanup steps like logging out of the application,
    stopping the Selenium server
      }

    }
```

11. Click on the java class and Select Run as > TestNG Test
12. As a beginner in Automation, the first steps are to give proper logging setup. In TESTNG, we have Reporter class which has logging functions which can be used.

```
Reporter.log("PASS : <font color=\"green\"><b>"+ "Test case has
passed"+"</b></font><br/>");
```

The below given code is an example of how java classes and functions can be linked to xml files

Java classes and Functions:

We should write java functions to connect xml file and run the web application.

Write java class for each web component of application .Each java class is one component.

Template consists of set of functions as well as abstract methods.

Write java functions for each and every element (buttons, textbox) action items functions of particular component.

Sample java code:

```
//import packages
//import Templates;


public class Template1 extends BaseTemplate
{

        public static Template t1;
        public static int i=0;


        public static void function1 (ArrayList<StepData> stepDatas) throws Exception
        {

                        Click ("elementid");
                        System.out.println ("Google");
     }
}
```

Each Java function should be called inside xml file, values passed inside xml file will be passed to java function and execution will be done according to xml file steps.

XML file:

1. XML stands for EXtensible Markup Language
2. XML is a markup language much like HTML.
3. Here every test script is written in xml file format with steps having element value and ID of the element.
4. Each XML file generation is used to run each test script.
   Example:

```
<TestStep action="function name" type="template path">

        <StepDatas>

                <StepData value="elementid"></StepData>

                <StepData value="elementvalue"></StepData>

        </StepDatas>

</TestStep>
```

5. Each Test step is one single action in web application. Web page element id, values will be pass as single individual steps .XML test script generation is main part in automation web application.

Conclusion:

This article provides the outline for setting up all the tools required to start automation using Selenium with Java Eclipse IDE. Additional to this, knowledge of Core JAVA is required to test the functionalities.

This document in brief overview these points:

1. Setup all the required software
2. Get to know how to write basic classes and methods in Core JAVA
3. Using DOM Inspector, recognize the objects in web page
4. Run the tests as Testing Test
5. Reporting can be done by using the TestNG Reporter class.