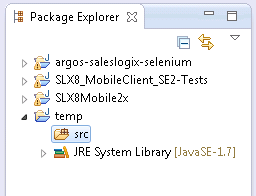
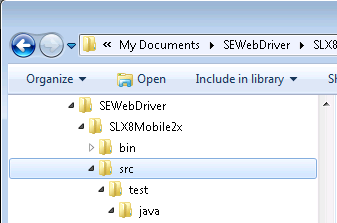
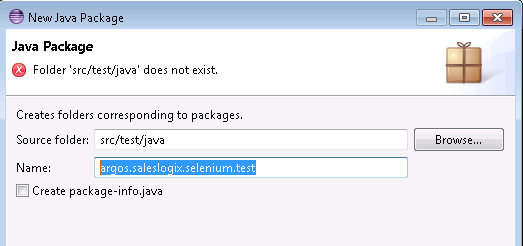
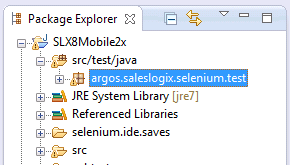
**How-To Setup a Selenium WebDriver Test Project for Jenkins CI Server Integration**

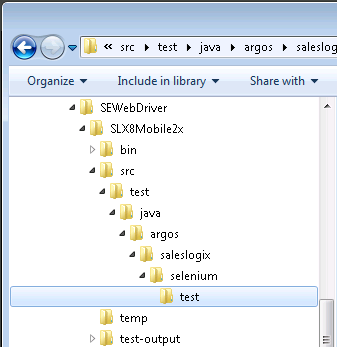
The purpose of this document is help guide you in setting up and configuring a Selenium Webdriver java test project to be compatible with a Jenkins CI automated build-test server.

**Step 1:** Have the Jenkins administrator setup a build project for Selenium WebDriver test automation on the Jenkins CI server (e.g. “argos-saleslogix-selenium”).

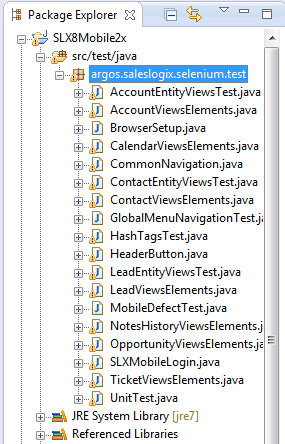
**Step 2:** Request user access to the Jenkins CI server and obtain permissions to execute new builds for the Selenium WebDriver build project setup from Step 1.  
  
**Step 3:** From your test host, launch the Eclipse IDE then create or open an existing Java project for your Selenium WebDriver tests.  
  
**Step 4:** By default, Eclipse will setup a “/src” folder under the main project folder.  
 <MAIN PROJECT FOLDER>/  
 src/  
  
  
  
Jenkins CI requires that the “/src” folder be setup a specific way. Setup/modify the “/src” folder as follows:  
 1. Using Windows Explorer or the command prompt, create the following sub-folder structure  
 under the “/src” folder: “/test/java”.  
 <MAIN PROJECT FOLDER>/  
 src/  
 test/  
 java/  
  
 

2. Create a new Java package under “/src/test/java” that has the **name** setup using the following  
 convention: **name** = <JENKINS BUILD PROJECT NAME>.test. Use the exact project name setup   
 from Step 1 above. (e.g. “argos.saleslogix.selenium.test”)  
  


As a result, in the Package Explorer you should see the following package item under the “/src/test/java” folder:  
  


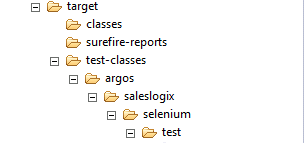
When viewed in File Explorer, the final source directory structure should have the following path:  
../src/test/java/<PROJECT NAME>/test such that each dot-segment of the project name is a separate folder.  
  


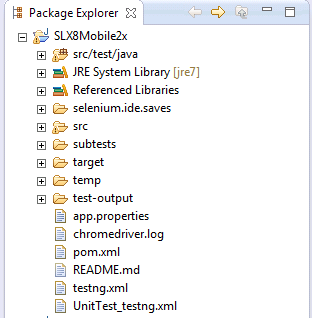
All your automation test .java class files should reside under the package.

**IMPORTANT NOTE:** By default, the Jenkins build-test server will automatically run all test class .java files under the test package that have the following naming convention: “\*Test.java” (e.g. “WidgetTest.java”).   
  


If you happen to have any other test .java files that you want to include in the test-run but they just happen to reside in another folder, then it’s recommended that you move those .java files under this source package test folder.

**Step 5:** Setup the following folder structure under the main project folder:  
 <MAIN PROJECT FOLDER>/  
 target/  
 classes/  
 surefire-reports/  
 test-classes/  
  
You do not have to place any files under these directories. These directories are populated by the Jenkins build-test process.

**Step 6:** Under the “/test-classes” directory setup from Step 5, setup a sub-directory structure that is identical to Step 4 above.  
  


**Step 7:** Create a new file: “pom.xml” that resides under the main project folder.  
  


A sample pom.xml is attached to this document. You can copy and paste to the Eclipse editor and save it as your pom.xml.

**Step 7:** Modify the following items in the pom.xml to customize it to your test project:  
 1. Setup the <groupId>GROUP\_ID</groupId> where the GROUPID is a unique value.  
 2. Setup the <artifactId>ARTIFACT\_ID</artifactId> where the ARTIFACT\_ID is a unique value.

**Sample pom.xml**:

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

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.saleslogix</groupId>

<artifactId>SLX8Mobile2x</artifactId>

<version>2.3</version>

<dependencies>

<dependency>

<groupId>org.seleniumhq.selenium</groupId>

<artifactId>selenium-java</artifactId>

<version>2.32.0</version>

</dependency>

<dependency>

<groupId>com.opera</groupId>

<artifactId>operadriver</artifactId>

</dependency>

<dependency>

<groupId>org.seleniumhq.selenium</groupId>

<artifactId>selenium-chrome-driver</artifactId>

<version>2.32.0</version>

</dependency>

<dependency>

<groupId>org.seleniumhq.selenium</groupId>

<artifactId>selenium-ie-driver</artifactId>

<version>2.32.0</version>

</dependency>

<dependency>

<groupId>org.testng</groupId>

<artifactId>testng</artifactId>

<version>6.1.1</version>

<scope>test</scope>

</dependency>

</dependencies>

<dependencyManagement>

<dependencies>

<dependency>

<groupId>com.opera</groupId>

<artifactId>operadriver</artifactId>

<version>1.2</version>

<exclusions>

<exclusion>

<groupId>org.seleniumhq.selenium</groupId>

<artifactId>selenium-remote-driver</artifactId>

</exclusion>

</exclusions>

</dependency>

</dependencies>

</dependencyManagement>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.1</version>

<configuration>

<source>1.7</source>

<target>1.7</target>

</configuration>

</plugin>

</plugins>

</build>

</project>