Selenium 2 (WebDriver) and JUnit 4

What is Selenium?

- ✓ It is a suite of tools for web applications. In short, Selenium automates browsers.
- ✓ It has three different components: Selenium IDE, Selenium WebDriver and Selenium Grid.
    - IDE: record-and-playback of interactions with the browser
    - WebDriver: ability to replicate all the user actions on the browser from the script. (For reference, WebDriver is the second generation of the original Selenium Remote Control for better performance.)
    - Grid: ability to run the WebDriver on different virtual machines or server. Purpose is to spread out the testing load and shorten the testing time.
- ✓ WebDriver can be used with user-defined language such as Java, C, Python and so forth.
- ✓ Since Selenium is a suite of tools, it can be controlled by many testing frameworks as well. For my project, I used Java as my main programming language and JUnit 4 as my testing framework.

Set Up

- ✓ Eclipse IDE is my main development environment.
- ✓ Along with eclipse, you need Maven to get the required Selenium pieces. (Maven is the project Management Tool for building projects, getting the dependencies and plug-in for the project).
- ✓ Maven has a direct integration plug-in for Eclipse. In order to install Maven in Eclipse, go to Help → Install New Software → next to "Work with:" , type in http://download.eclipse.org/technology/m2e/releases → follow the steps to complete the installation.
- ✓ After that, restart the eclipse.
- ✓ File → New → Other → Choose Maven, under which there will be Maven Project → Click Next, Next (no changes) → when you get to group id and artifact id, group id is the name of your package folder where all your required files will be stored. Artifact id is the name of your project which will be appeared on Eclipse. → Click Finish
- ✓ You should see your project name in the project explorer of Eclipse. Expand your project, and near the end of it, you can see pom.xml (Project object model – POM).
- ✓ Open that, you can copy the following.

```xml
<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.Trapeze.Selenium</groupId>
  <artifactId>Simulation</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>jar</packaging>

  <name>Simulation</name>
  <url>http://maven.apache.org</url>
  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  </properties>
  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.9</version>
      <scope>test</scope>
    </dependency>
    <dependency>
      <groupId>org.seleniumhq.selenium</groupId>
      <artifactId>selenium-java</artifactId>
      <version>2.23.1</version>
    </dependency>
    <dependency>
      <groupId>com.opera</groupId>
      <artifactId>operadriver</artifactId>
    </dependency>
    <dependency>
      <groupId>org.seleniumhq.selenium</groupId>
      <artifactId>selenium-server</artifactId>
      <version>2.23.1</version>
    </dependency>
  </dependencies>
  <dependencyManagement>
            <dependencies>
                <dependency>
                    <groupId>com.opera</groupId>
                    <artifactId>operadriver</artifactId>
                    <version>0.14</version>
                    <exclusions>
                        <exclusion>
                            <groupId>org.seleniumhq.selenium</groupId>
                            <artifactId>selenium-remote-driver</artifactId>
                        </exclusion>
                    </exclusions>
                </dependency>
            </dependencies>
  </dependencyManagement>
</project>
```

- ✓ Basically, this process is, we are using maven to download the required pieces of Selenium. In the xml file, you can see that we downloaded junit, selenium-java and so on. If those versions are not the latest versions by the time you used Selenium, just change the number in version tag to the latest version and save it. Eclipse will automatically build and update it to the latest version.

Dive into Selenium WebDriver

- ✓ Now that setting up with the WebDriver is done, it is time to dive into some of the most useful commands of WebDriver.

Opening Browsers & Getting the website

- ✓ You can start the java class as in the usual way. In order to use the WebDriver, you just have to create the WebDriver instance.

```
WebDriver driver = new WebDriver(); This will open the firefox driver;
```

- ✓ WebDriver opens the Firefox as its original browser. So, if your testing requires a specific browser such as Internet Explorer, or Chrome, then you need to download that specific driver from Selenium website. In the following, I will show the opening of Internet Explorer as an example.

```
System.setProperty("webdriver.ie.driver", "path to your downloaded ie driver");
DesiredCapabilities iecapability = DesiredCapabilities.internetExplorer();
WebDriver driver = new InternetExplorerDriver(iecapability);
```

- ✓ These three lines will do the trick in opening any types of browser.

- ✓ To go to the website you want to go is pretty simple.

```
driver.get("Name of the website");
```

- ✓ However, this is for opening only one browser. If you want to open two browsers, you can use the following method to start the second one. Be aware that WebDriver can instantiate only one InternetExplorerDriver.

```
((JavascriptExecutor)driver).executeScript("window.open()");
Set<String> handles = driver.getWindowHandles();
Iterator<String> ite = handles.iterator();
While (ite.hasNext())
{String last_window = ite.next().toString(); }
driver.switcTo().window(last_window);
```

Locating the Elements on the website

- ✓ This can be the most complicated learning curve for the Selenium Users. There are a couple of ways to locate and select the elements on the website.

- ✓ If the element has a **unique** class name or **unique** id in the whole table, the story is pretty simple.
  - driver.findElement(By.classname("…."));{OR}driver.findElement(By.id("…."));

- ✓ Else, we have to use the "cssSelector" to select the element in the HTML table. Some of the most used cssSelectors are:

  - Selecting the button tag with the class name "A" and text "Add" → driver.findElement(By.cssSelector("button.A:contains('Add')) → "." refers to button class and "contains" pick outs the button class with "Add" text in it.

  - Selecting the button tag with value name "B" → driver.findElement(By.cssSelector("button[value = \"B\"]));

  - Selecting the button tag with id "123" but it is the child of div class → driver.findElement(By.cssSelector("div button#123)) → the "space" between the div and button means that parent-child relationship in cssSelector. Likewise, for the sibling relationship, you can use " + " sign between the elements.

- ✓ Those are the most commonly used cssSelectors in the Selenium. However, there are a bunch more and for more info, please refer to http://www.w3.org/TR/css3-selectors/#selectors .

Mouse and Keyboard Events

- ✓ After locating the elements on the HTML table, the next thing you need to do either click, or select and type or get the content, right?
- ✓ For the simple click, this will do the job.

```
driver.findElement(By.cssSelector("…")).click();
```

- ✓ However, in some cases, when the Element is not visible and needed to be scrolled down, (Selenium is like user eyes, if you can't see it, you can't click it.), this will do the trick.

```
WebElement temp = driver.findElement(By.cssSelector("…"));
((JavascriptExecutor)driver).executeScript("arguments[0].focus()",temp);
temp.click();
```

- ✓ If you want to use right click, use the below method:

```
Actions action = new Actions (driver);
WebElement source = driver.findElement(By.cssSelector("…"));
WebElement target = driver.findElement (By.cssSelector("…"));
action.dragAndDrop(source , target);
Thread.sleep(500);                    // giving some breathing time.
action.contextClick(target).perform();
```

- ✓ After the mouse events, if you want to type in a blank field, it is also a simple story.

```
driver.findElement(By.cssSelector("…")).sendKeys("your message here!");
That's it.
```

- ✓ However, if the text area is not blank then, you will need to get some help from javascript again.

```
WebElement temp = driver.findElement(By.cssSelector("…"));
((JavascriptExecutor)driver).executeScript("arguments[0].select()",temp);
temp.sendKeys("yourmessagehere!");
```

- ✓ So far, you might have noticed the use of JavascriptExecutor every so often every time the WebDriver alone can't finish the job. For more info about the commands for using javascript, use this link, http://api.jquery.com/category/events/ .

JUnit

- ✓ The above topics pretty much covered what you can do with Selenium WebDriver. As I said before that Selenium just being a suite of tools, I used the JUnit test framework as my main framework to test the Selenium scripts.
- ✓ There are a bunch of JUnit tutorials available for free online. Below, I will explain how I set up my project with the JUnit test structure.
- ✓ JUnit is annotations -type framework. We have to put all those JUnit annotations either before the class or before the method. Typical JUnit annotations are:
  - @BeforeClass → call the "method" only once before the class starts
  - @AfterClass → execute the "method" only once at the end of the class
  - @Test → "method" where you places the actual test code
  - @Before → call every time before the @Test is executed
  - @After → execute every time after the @Test is executed
  - @RunWith(Suite.class) →combine all the sub-test-classes into one main test class

- ✓ Example Code:

```
@RunWith(Suite.class)
@SuiteClasses({
	ScheduleAdherenceTest.class,
	RouteAdherenceTest.class,
	RemoteLogOffTest.class
})
public class AllTests{
	…….

	@BeforeClass
	public static void setUpBeforeClass() throws Exception {

		----------------------------------------(Code)

	}

	@AfterClass
	public static void setUpBeforeClass() throws Exception {

		----------------------------------------(Code)

	}

	@Test
	public void ExampleTest(){

		----------------------------------------(Code)

	}

	}
```

How to run the existing project

To run it from Eclipse:

- ✓ Open the eclipse → Click File → Import → Select "Under General, Existing Projects into Workspace" →Click Next → Find the directory of the project folder → Click Next and Finish
- ✓ After importing the project, it might be flagging as having the errors. Most likely, it is because of the errors in the build path.
- ✓ For this, right click on the current project → Choose Build Path → Configure build path → Click "Java Build Path" on the left side → select "Libraries" tag on the right → Check if there is any error and fix by either removing or editing the location of those libraries → OK.
- ✓ Then, in order to configure run time arguments, right click on the current project again → Choose Run As → Run Configurations → On the left panel, select the name of your project if it is not already selected → Then on the right, select "Arguments" tag → In the VM(virtual machine) arguments, type in the following:

-Dlog = (the place where you want to record your output) for eg: "C:\\Users\\zaw.win\\Documents\\NOVUSTests\\Novus Regression Tests - Run\\log_test.log" –Droutesim = (the location of your MDTRouteSime.exe)

- ✓ Close and Save.

To run it from the batch file:

- ✓ Under the right folder, you can see "jar.cmd" file. That command file creates the jar files for the classes of the project. So, every time you fix the code from Eclipse, you have to run "jar.cmd" file to update the newly built classes in the jar file.
- ✓ For those who don't have access to the code, you don't need to worry about "jar.cmd".
- ✓ You can go into "Novus Regression Tests – Run" folder, there you will find "RunSeleniumTests.cmd".
- ✓ For the first time running, you will need to right click on that command file and click "Edit".
- ✓ And find –Dlog and –Droutesim and change the following:

-Dlog = (the place where you want to record your output) for eg: "C:\\Users\\zaw.win\\Documents\\NOVUSTests\\Novus Regression Tests - Run\\log_test.log" –Droutesim = (the location of your MDTRouteSime.exe)

- ✓ Save and close.
- ✓ PS: for those who don't have administration right to the computer, you have to run those command files as an administrator by right click.

If you have any questions or are not clear with the code that I wrote, please Contact ME: Zaw Win
626-703-8543
zawnaingwynn@gmail.com (General Email)
zwin13@cornellcollege.edu (School Email – valid until May 2013)

Last but not least, I hope you have a great year ahead and most of fun with your coworkers! ☺ ☺ ☺

Peace,
Zaw