**Selenium Tutorial**

This is a two part tutorial to get you familiar with features of Selenium. In this assignment you will first use Selenium IDE and then [Selenium RC](http://selenium-rc.openqa.org/) to automate web application testing.

**Part 1 Selenium IDE**

In this first part of the tutorial, you will install and play with Selenium IDE to get a big picture idea of web application testing.

**Install Selenium IDE**

Selenium IDE is an extension for Firefox, therefore Firefox is required to work through this tutorial.

With Firefox, visit <http://seleniumhq.org/projects/ide/>, click on “Download Selenium” and follow the instructions. Firefox needs to be restarted for complete the installation.

**Record a Test Case**

Open Selenium IDE in Firefox, “Tools” -> “Selenium IDE”.

At this point Selenium is already starting to record your actions within the browser.

Create a test case:

* Visit [www.google.com](http://www.google.com)
* In the search box type in “selenium”
* Click on “Google Search”
* Click on the first result
* Save the test case

**Examine the Test Case**

Open the saved test case with WordPad.

**Replay the Test Case**

* Restart Firefox and Selenium IDE
* Open the saved test case
* Set the speed to “Slow” and click on “Play entire test suite”

**Part 2 Selenium RC**

This part of the tutorial uses [Selenium RC](http://selenium-rc.openqa.org/) to write automated web application UI tests in the Java programming language against any HTTP website using Google Chrome browser.

**Installation**

Download the Selenium Server and the Java Selenium Client Driver release from <http://seleniumhq.org/>.

**Eclipse Project Setup**

We will be using Eclipse and JUnit to develop web application test cases. For each Eclipse Project, you will need to make sure the necessary Selenium libraries are in the “Java Build Path”.

Note: X.X.X means the newest stable version

There are two Selenium libraries we need 1) selenium-server-standalone-x.x.x.jar.jar and 2) selenium-java-x.x.x.jar

* Create a new “Java Project” in Eclipse
* Create a “lib” directory for this project
* Copy “selenium-server-standalone-x.x.x.jar” to the “lib” directory
* Copy “selenium-java-x.x.x.jar” from the zip file to the “lib” directory
* Add these two libraries to the “Java Build Path”

**Selenium Server**

The Selenium Server must be started before any Selenium tests can be run. The Selenium Server can be started by running the "selenium-server-standalone-x.x.x.jar " file.

Open a terminate and type  
  
java -jar …/selenium-server-standalone-x.x.x.jar

But we have a better way to do this!

**Selenium Server Control from Java**

An easier and automated way is to start/stop the Selenium Server from Java. You can then wire your JUnit tests to start/stop Selenium Server in setup()/teardown() methods.

**import** org.openqa.selenium.server.\*;

**public** **class** SeleniumServerControl {

**private** **static** **final** SeleniumServerControl *instance* =

**new** SeleniumServerControl();

**public** **static** SeleniumServerControl getInstance() {

**return** *instance*;

}

**private** SeleniumServer server = **null**;

**protected** SeleniumServerControl() {

}

**public** **void** startSeleniumServer() {

**if** (server == **null**) {

**try** {

server = **new** SeleniumServer();

server.start();

} **catch** (Exception e) {

e.printStackTrace();

}

}

}

**public** **void** stopSeleniumServer() {

**if** (server != **null**) {

**try** {

server.stop();

server = **null**;

} **catch** (Exception e) {

e.printStackTrace();

}

}

}

}

**Java Client Driver**

Now you are ready to write JUnit tests that use Selenium to web applications.

**import** **static** org.junit.Assert.\*;

**import** org.junit.\*;

**import** com.thoughtworks.selenium.\*;

**public** **class** SimpleTest {

**private** Selenium selenium;

@Before

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

SeleniumServerControl.*getInstance*().startSeleniumServer();

selenium = **new** DefaultSelenium("localhost", 4444, "\*googlechrome",

"http://localhost:8888/");

selenium.start();

}

@After

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

selenium.stop();

SeleniumServerControl.*getInstance*().stopSeleniumServer();

}

@Test

**public** **void** testSimple()**throws** Throwable {

selenium.open("http://localhost:8888/");

*assertEquals*("Assignment 3 Toy Web App", selenium.getTitle());

selenium.close();

}

}