

Becoming an UI Testing Rock Star

Introduction

Selenium is a cross-platform/browser software testing framework for web applications that provides tools for writing the test cases without learning the unit test frameworks.

We haven been using JUnit/HtmlUnit to simulate the user behaviors for the past few projects. Although it works, it takes a lot of time to write UI tests, since you cannot easily see what you test against without printing out the results. Besides, there is no guarantee that you will get the same results if running the tests on a different browser. The idea of using Selenium is to solve these problems and more.

Tools

- Java SDK
- Selenium IDE
- Selenium Server (previously Selenium RC)
- selenium-java-client-driver.jar
- Firefox 3.x (Selenium IDE doesn't support Firefox 4 when this guide was created)

Technical Details

Installing Selenium IDE

Selenium IDE is a Firefox plugin that allow users to record and automate the UI test. It provides the functionalities to create single tests, test suites, and export the test scripts to different formats based on the programming languages picked by the user , e.g. html, java (JUnit / TestNG), ruby, php, perl, c#, python, etc.

Download information could be found at:

<http://seleniumhq.org/projects/ide/>

Downloading Selenium Server

Selenium server is a Jetty integrated web server that acts as a HTTP proxy for web requests. It has the ability to launch and kill browsers. By default, it listens to the port 4444 for incoming requests when it is started.

Download information could be found at:

<http://seleniumhq.org/projects/remote-control/>

Downloading the Selenium Java Client Driver

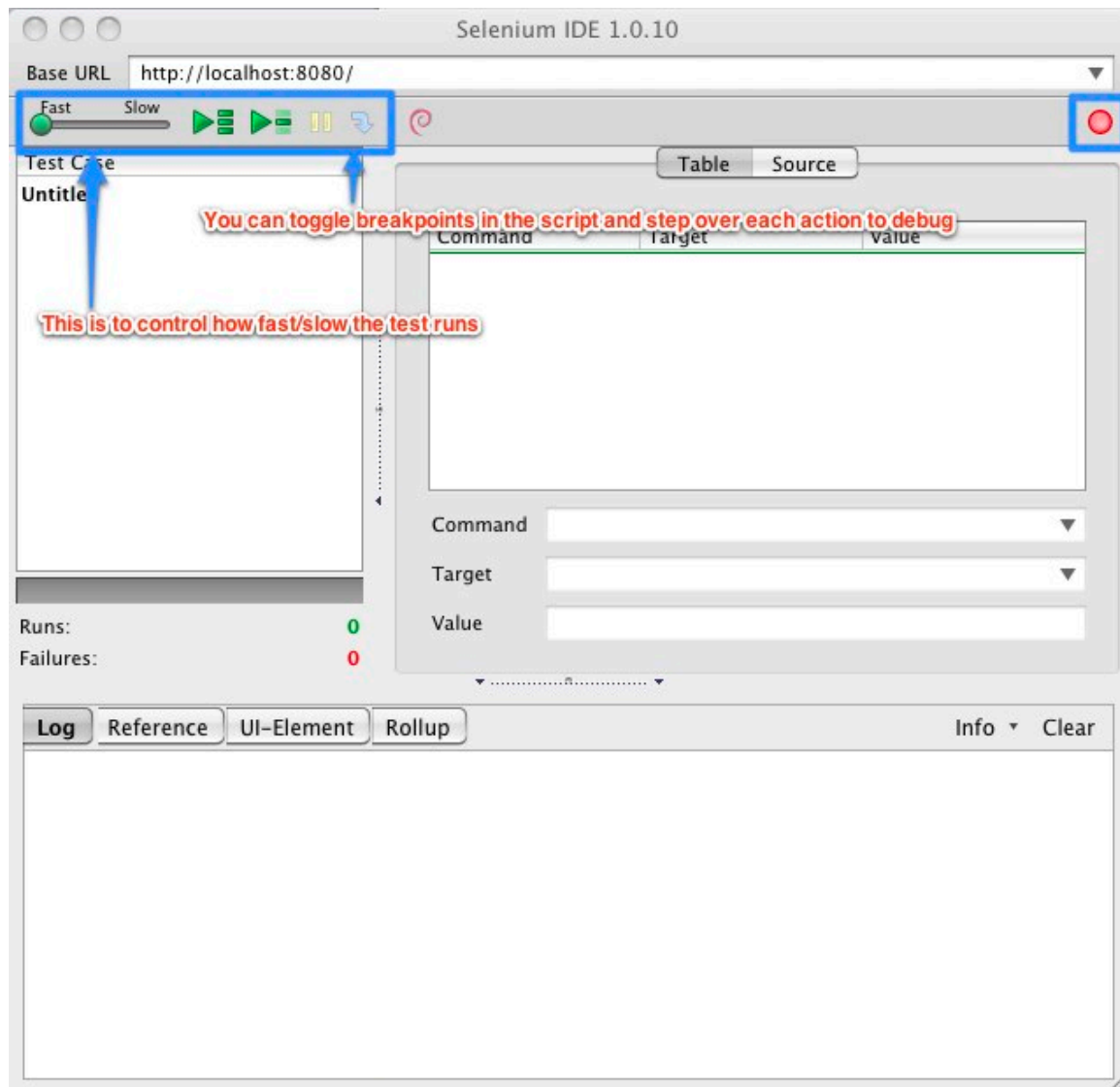
This is used when you want to run Selenium tests as JUnit tests.


Download information could be found at:

<http://selenium.googlecode.com/files/selenium-java-2.0b3.zip>

How to create a Selenium test script using Selenium IDE

After installing the Selenium IDE, launch it in Firefox under Tools and you will see the screen like this:



Click the Record button  to begin recording the actions. After you finish recording the actions, click the record button again to stop recording. See the [Selenium commands](#) section for more information about the available command and the options to locate the elements.

How to run a Selenium test as a JUnit test

1. Create a test using Selenium IDE by the instruction above.
2. Export the test case as JUnit 4 and it will create Java code like this:

```
Java #
1  package org.kuali.hr.time.web;
2
3  import org.junit.After;
4  import org.junit.Before;
5  import org.junit.Test;
6
7  import com.thoughtworks.selenium.DefaultSelenium;
8  import com.thoughtworks.selenium.SeleneseTestCase;
9
10 public class UICalendarTest extends SeleneseTestCase {
11     @Before
12     public void setUp() throws Exception {
13         selenium = new DefaultSelenium("localhost", 4444, "*firefox", "http://localhost:8080/");
14         try {
15             selenium.start();
16         } catch (Exception e) {
17
18         }
19     }
20
21     @Test
22     public void testAddTimeBlock() throws Exception {
23         selenium.open("/tk-dev/TimeDetail.do");
24         selenium.type("__login_user", "fran");
25         selenium.click("//input[@name='login']");
26         selenium.waitForPageToLoad("30000");
27         selenium.mouseDownAt("css=td.fc-day4", "10,10");
28         selenium.mouseUpAt("css=td.fc-day4", "10,10");
29         selenium.type("beginTimeField", "08:15 AM");
30         selenium.type("endTimeField", "05:15 PM");
31         selenium.click("//button[@type='button']");
32         selenium.waitForPageToLoad("30000");
33         verifyTrue(selenium.isTextPresent("08:15 AM"));
34         verifyTrue(selenium.isTextPresent("05:15 PM"));
35     }
36
37     @After
38     public void tearDown() throws Exception {
39         selenium.stop();
40     }
41 }
```

3. Include the selenium-server-standalone-xxx.jar and selenium-java-client-driver.jar to the java classpath or in maven (pom.xml):

```

<dependency>
  <groupId>org.seleniumhq.selenium.client-drivers</groupId>
  <artifactId>selenium-java-client-driver</artifactId>
  <version>1.0.2</version>
</dependency>
<dependency>
  <groupId>org.seleniumhq.selenium</groupId>
  <artifactId>selenium-server</artifactId>
  <version>2.0a5</version>
</dependency>
<dependency>
  <groupId>org.codehaus.mojo</groupId>
  <artifactId>selenium-maven-plugin</artifactId>
  <version>1.0.1</version>
  <type>maven-plugin</type>
</dependency>

```

4. Start the Selenium Server from the command line:

```
java -jar selenium-server-standalone-2.0b3.jar -interactive
```

The interactive mode allows you to test the Selenium command directly in the console.

5. Try the command below in the console to make sure you can successfully launch the browser from the selenium server:

```
cmd=getNewBrowserSession&1=*firefox&2=http://www.google.com
```

If you want to use a different browser than Firefox to run the test scripts. See the [Troubleshooting](#) section for more information.

6. If you use Eclipse, run the Java file as a JUnit test and it will launch the browser and execute the script. You can also set the break points in Java and step through each action.

Troubleshooting

- How to run Selenium server on different platforms
 - o Windows
 - You will have to add the browser folder to the PATH variable under environment variables, so when you specify *firefox in the command, the system knows where to find the exe file. If you want to run the tests by using a different browser, you will have to do the same thing. This link shows you how to do that: <http://goo.gl/pjoqA>
 - Here is a list of the browsers that Selenium supports: <http://seleniumhq.org/about/platforms.html>
 - Note that the command *chrome doesn't mean to use the Google Chrome as the browser. *googlechrome is the one you are looking for.
 - o Ubuntu

- Since you can directly open the browsers from the command line, there isn't any extra setting needed for the linux environments.
- Mac OS
 - I haven't figured out how to run Selenium Server on Mac OS. I got this error message no matter what command I used:
Unexpected end of file from server. I suspected that this is something to do with the browser path, but adding the firefox folder to the user / system path didn't help. This section needs to be updated in the future once this is figured out.

Resources

- Selenium Documentation
<http://seleniumhq.org/docs/>
- Selenium commands
 - http://seleniumhq.org/docs/02_selenium_ide.html#commonly-used-selenium-commands
 - http://seleniumhq.org/docs/02_selenium_ide.html#locating-elements