**

# A Complete Selenium Guide

**Softech** **Solutions Inc.**

[www.softechsolutionsgroup.com](http://www.softechsolutionsgroup.com)

[saney.alam@softechsolutionsgroup.com](mailto:saney.alam@softechsolutionsgroup.com%20)

Selenium –Headless Browser testing

***Lecture Notes***

Selenium –Headless Browser testing

In this section we will try to learn about headless browsers and their real usage. This section is divided into following sections

1. What is a headless browser?

2. What is the use of headless browsers?

3. Points to consider before you run your tests on Headless browser

4. HtmlUnitDriver – Selenium’s headless browser class

5. Java script support in HtmlUnitWebDriver

6. Testing on different browser versions

**What is a headless browser?**

Headless browser is a term used to define browser simulation programs which do not have a GUI. These programs behave just like a browser but don’t show any GUI. Famous ones are HtmlUnit and the NodeJs headless browsers. There are a good number of more browsers too.

**What is the use of Headless browsers?**

Headless browsers are typically used in following situations

1. You have a central build tool which does not have any browser installed on it. So to do the basic level of sanity tests after every build you may use the headless browser to run your tests.

2. You want to write a crawler program that goes through different pages and collects data; headless browser will be your choice. Because you really don’t care about opening a browser. All you need is to access the webpages.

3. You would like to simulate multiple browser versions on the same machine. In that case you would want to use a headless browser, because most of them support simulation of different versions of browsers. We will come to this point soon.

**Things to pay attention to before using headless browser**

Headless browsers are simulation programs; they are not your real browsers. Most of these headless browsers have evolved enough to simulate, to a pretty close approximation, like a real browser. Still you would not want to run all your tests in a headless browser. JavaScript is one area where you would want to be really careful before using a Headless browser. JavaScript are implemented differently by different browsers. Although JavaScript is a standard but each browser has its own little differences in the way that they have implemented JavaScript. This is also true in case of headless browsers also. For example HtmlUnit headless browser uses the Rihno JavaScript engine which not being used by any other browser.

**Selenium support for headless browser**

Selenium supports headless testing using its class called HtmlUnitDriver. This class internally uses HtmlUnit headless browser. HtmlUnit is a pure Java implementation so you will not find this tutorial to be focused on java binding’s Quick start! You can create an HtmlUnitWebDriver like this

|  |
| --- |
| **HtmlUnitDriver unitDriver = new HtmlUnitDriver();** |

Once you have the Html unit driver you can just use it as a regular WebDriver. Let’s see it with an example where we would perform following task using HtmlUnitDriver.

1. Open webpage http://google.com

2. Get the title of the page.

3. Search for ‘Selenium’

4. Check the title of the page again.

public static void main(String[] args) {

// Declaring and initialising the HtmlUnitWebDriver

HtmlUnitDriver unitDriver = new HtmlUnitDriver();

// open google.com webpage

unitDriver.get("http://google.com");

System.out.println("Title of the page is -> " + unitDriver.getTitle());

// find the search edit box on the google page

WebElement searchBox = unitDriver.findElement(By.name("q"));

// type in Selenium

searchBox.sendKeys("Selenium");

// find the search button

WebElement button = unitDriver.findElement(By.name("btnK"));

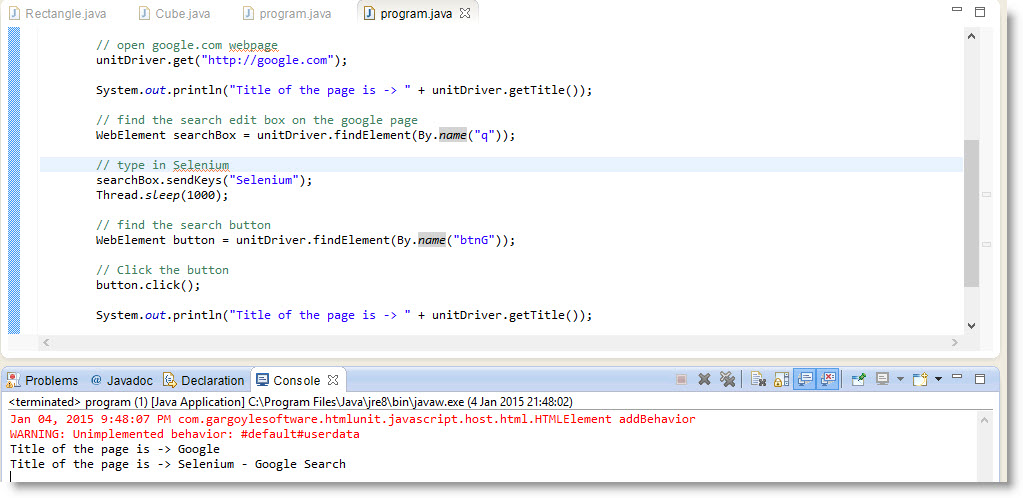
// Click the button

button.click();

System.out.println("Title of the page is -> " + unitDriver.getTitle());

}

You would notice that the commands on the HtmlUnitDriver are exactly the same as any other browser object in Selenium WebDriver. That’s actually the beauty of it, Selenium WebDriver team has made it quite easy to use HtmlUnit headless browser. One you run this program the output you would see will be:

[](http://toolsqa.wpengine.com/wp-content/uploads/2015/01/HtmlUnitRun1.jpg)

**Java script support in HtmlUnitWebDriver**

HtmlUnitWebDriver by default doesn’t does not enable JavaScript. This means in the above code if you try to execute java script and error will be thrown stating that JavaScript is not enabled. To enable Java script in two ways

1. By passing the java script enable flag in constructor

|  |
| --- |
| **HtmlUnitDriver unitDriver = new HtmlUnitDriver(true);** |

2. By specifying it using the **setJavascriptEnabled** method

|  |
| --- |
| **HtmlUnitDriver unitDriver = new HtmlUnitDriver();**  **unitDriver.setJavascriptEnabled(true);** |

Once you have html unit Webdriver set for Java script you can execute any script exactly like you would do in any other browser object. Here is a sample code.

// Declaring and initialising the HtmlUnitWebDriver

HtmlUnitDriver unitDriver = new HtmlUnitDriver(); unitDriver.setJavascriptEnabled(true);

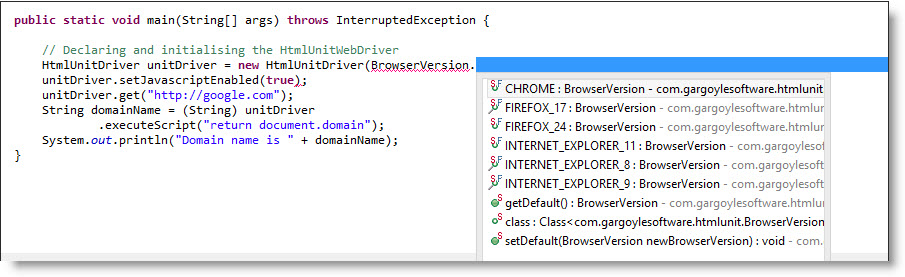
unitDriver.get("http://google.com");

String domainName = (String) unitDriver.executeScript("return document.domain");

System.out.println("Domain name is " + domainName);

**Testing on different browser versions**

HtmlUnitDriver allows you to select the version of browser that you would like to run your tests on. You can select the browser version like this

[](http://toolsqa.wpengine.com/wp-content/uploads/2015/01/headless2.jpg)BrowserVersion object is present in the com.gargoylesoftware.htmlunit.BrowserVersion package. At this point you can see that we support for Chrome, Firefox version 17 to 24 and IE version 9 to 11. Use whatever version you want before creating the HtmlUnitDriver class and use it. Sample code is shown below

import org.openqa.selenium.htmlunit.HtmlUnitDriver;

import com.gargoylesoftware.htmlunit.BrowserVersion;

public class program {

public static void main(String[] args) throws InterruptedException {

// Declaring and initialising the HtmlUnitWebDriver

HtmlUnitDriver unitDriver = new HtmlUnitDriver(BrowserVersion.FIREFOX\_24);

unitDriver.setJavascriptEnabled(true);

unitDriver.get("http://google.com");

String domainName = (String) unitDriver.executeScript("return document.domain");

System.out.println("Domain name is " + domainName);

}

}