CS570 – Software Testing

**HOP04A - Introducing & Installing Selenium WebDriver**

10/22/2020 Developed by Kim Nguyen

Center for Information Assurance (CIAE) @City University of Seattle (CityU)



**Caution**

* If you already finished this module through any CityU School of Technology & Computing (STC) courses, just skim this module and skip it.
* Some version numbers may not match with the newly released ones. If so, stay with the most recent ones.
* This tutorial targets Windows OS and Mac users.
* We cannot explain every step. This cookbook always needs your own creative judgement. Try to solve the problem on your own, after a few tries, if you cannot solve the issue, contact TA for help.

**Learning Outcomes**

* Deeper understanding of Selenium and Web App testing
* Understand Selenium WebDriver.
* Install and use Selenium WebDriver to continue perform deeper web app testing.

**Resources**

* Javapoint.come | Selenium WebDriver - <https://www.javatpoint.com/selenium-webdriver>

**Selenium Webdriver**

Last week, we have learned Selenium IDE – a quick and easy browser extension to test web apps. This week, let’s continue to explore Selenium using Selenium WebDriver.

Graphical user interface, text

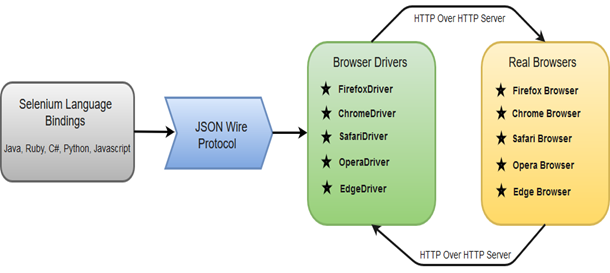
Description automatically generated

Selenium WebDriver is the most important component of Selenium Tool's Suite. In WebDriver, test scripts can be developed using any of the supported programming languages and can be run directly in most modern web browsers. Languages supported by WebDriver include C#, Java, Perl, PHP, Python and Ruby. More about Selenium WebDriver: <https://www.selenium.dev/documentation/en/webdriver/>

**Selenium WebDriver – Architecture**

Selenium WebDriver API provides communication facility between languages and browsers.

The following image shows the architectural representation of Selenium WebDriver. *(consists of 4 basic components: Selenium Language Bindings, JSON Wire Protocol, Browser Driver, Real Browser):*



Selenium uses drivers, specific to each browser in order to establish a secure connection with the browser without revealing the internal logic of browser's functionality. The browser driver is also specific to the language used for automation such as Java, C#, etc.

When we execute a test **script** using **WebDriver**, the following operations are performed internally.

* HTTP request is generated and sent to the browser driver for each Selenium command.
* The driver receives the HTTP request through HTTP server.
* HTTP Server decides all the steps to perform instructions which are executed on browser.
* Execution status is sent back to HTTP Server which is subsequently sent back to automation script.

Most of the commands used in Selenium WebDriver are easy to implement. For instance, to launch Chrome browser in WebDriver following command can be used:

WebDriver driver = new ChromeDriver();

**Let’s install Selenium WebDriver and practice!**

**Install Selenium WebDriver**

Throughout this course, we will focus on testing web app using Chrome driver. Thus, we will be installing web driver for Chrome browser, you can install webdrivers for other browsers if you wish to experiment beyond this HOP.

**For Mac Users:**

1. Open Terminal and type the following command to install Chrome driver:

brew install chromedriver

You should see the similar message if the installation was successful:



**NOTE**: If you do not have brew installed, use the following command to install brew BEFORE installing chromedriver:

/bin/bash -c "$(curl -fsSL <https://raw.githubusercontent.com/Homebrew/install/master/install.sh>)"

1. Check chromedriver version to confirm the installation:

chromedriver --version



**For Windows Users:**

1. Go to <https://sites.google.com/a/chromium.org/chromedriver/downloads> , download WebDriver for the appropriate Chrome version you are using:

Graphical user interface, text, application

Description automatically generated

1. Choose the chromedriver\_win32.zip file to download:

Table

Description automatically generated

1. The file will be downloaded in a zip format, unzip the file into a location of your choice, you should then see a file called chromedriver.exe

A picture containing graphical user interface

Description automatically generated

1. Add the path of chromedriver.exe into the system Environment Variable:

Graphical user interface, text, application

Description automatically generated

We can now create a script in VSCode with Selenium WebDriver to test a sample web app. If you have completed all the steps in this document, proceeded to HOP04B.