CS570 – Software Testing

**HOP04B – Testing Web App using scripts & 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>

1. In VSCode, open Module 4 folder under the Hands-on Practice folder, the path should be similar to:

CS570-hop-Hands-on-practice/Module4

1. Open the terminal in VSCode, type the following command to clone a sample Selenium WebDriver project:

git clone <https://github.com/eviltester/startUsingSeleniumWebDriver.git>

1. Expand the project folder, open MyFristChromeTest.java file:

Text

Description automatically generated

1. Look for driver.close() and driver.quit() functions (line 73, 74), comment out these 2 lines and save changes afterwards:

Text

Description automatically generated

1. In VSCode terminal, change directory into the cloned project:



1. Type the following command to run the script:

mvn test -Dtest=MyFirstChromeTest

Text

Description automatically generated

You should see a Chrome browser open up:

Graphical user interface, application

Description automatically generated

1. Let’s say we want to add more tests, for example, clicking on the “Basic Web Page Example” url after accessing the home page:

Graphical user interface, application

Description automatically generated

1. To do so, open MyFristChromeTest.java file again, add the following lines of code:Text

   Description automatically generated

And

Text

Description automatically generated

1. Run the test again to see the result:

mvn test -Dtest=MyFirstChromeTest

End result:

Graphical user interface, text, application, email

Description automatically generated

**CHALLENGE:**

Edit MyFirstChromeTest.java to add following actions:

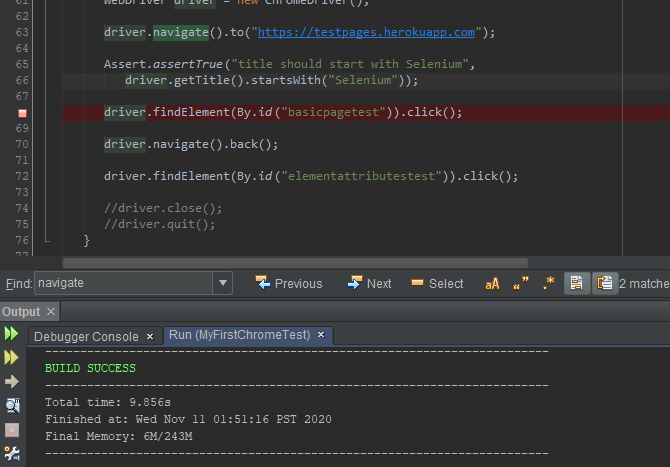
1. Navigate to the previous page (home page)
2. Click on “Element Attributes Example” page:

Graphical user interface, text

Description automatically generated

1. Close the browser.

**Run your code, take a screenshot showing that there’s no errors. Save all changes, submit your work to Github**



In VSCode terminal, Type the following command:

* git add . (to copy all changes you have made)
* git commit -m “Submission for Module 4 – Your Name” (To add a message to your submission)
* git push origin master (to upload your work to Github)