UNIVERSITY OF ANTWERP
DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE

Software Testing Lab

Assignment 7

Submission Deadline: **May 2th, 20:00**

Refer to Assignment 1 for introductory information.

1 GUI TESTING

1.1 SELENIUM IDE

For these exercises we will use Selenium to test the GUI of a web-application. Selenium[1] is a powerful open-source framework for automated web testing. It has two different versions, 1) the WebDriver, and 2) the IDE. Using the Selenium IDE we can easily record and playback scripts. However, we cannot (yet) export them. We will thus use a different tool called Katalon[2]. You can install this tool by searching the Add-on's in Firefox[3].

These exercises will be verified using Ubuntu and Firefox. Feel free to use your favourite browser, but make sure that your scripts work with Firefox on Ubuntu.

Keep in mind that the goal of these exercises is to test the interface, not the implementation!

[1] `https://www.seleniumhq.org`
[2] `https://www.katalon.com/web-testing/`
[3] `https://addons.mozilla.org/en-GB/firefox/addon/katalon-automation-record/`

- **Exercise 1.** Create a "hello world" test script by recording a test script (using the Selenium IDE) that navigates from the ansymore home page (`http://ansymore.uantwerpen.be`) to the "Software Testing" course using the menu. Make sure you can replay your script. Save the script (as .html). Document the procedure. **(Required, 5 points)**

- **Exercise 2.** Clone the script from 1, and add the following steps to the script: A) Wait until the page is loaded, B) Verify that "Software Testing" is in the header, C) verify that the correct name of the professor and assistant are located after "Professor" and "Assistant". Save the script (as .html). **(Required, 5 points)**

- **Exercise 3.** Create a test script that navigates to all the different courses from the menu. The script should loop over the different menu items and verify that each page is loaded. Verify that each course has a professor (field not empty). The script should only fail when a page cannot be loaded or when the course does not have a professor. Save the script (as .html). **(Required, 10 points)**

1.2 Selenium WebDriver

For the next exercises, you are **free** to use either a framework for Behaviour-Driven Development (BDD) and a scripting language, e.g. JBehave and Groovy together with JUnitStories, or use the selenium webdriver and stick to plain java files and JUnit. Either way, you should familiarise yourself with BDD to answer exercise 9.

1.2.1 Behaviour-Driven Development (BDD)

BDD is an evolution of test-driven development (TDD) and acceptance-test driven design, and is intended to make these practices more accessible and intuitive to newcomers and experts alike. It shifts the vocabulary from being test-based to behaviour-based, and positions itself as a design philosophy."[4]. Groovy[5] is a powerful dynamic scripting language that compiles directly to the JVM. It can be used to write the steps of BDD test scripts.

Before you start the next part of the exercise session, read the "introduction to BDD"[6] and the "getting started" pages. Next, read the Groovy documentation[7]. Finally, read the "using Selenium" page[8]. If you choose to use JBehave, use the Selenium WebDriver API.

1.2.2 Selenium WebDriver Example

A working example of BDD can be found at: `https://github.com/jbehave/jbehave-tutorial`. The example is set up to use Firefox as the browser. Import the "groovy-pico" project in Eclipse, wait for maven to install all its dependancies and possibly restart eclipse. Select the Maven Profile "Stable" by right clicking on the "etsy-stories-groovy-pico" project. Make

[4] `jbehave.org`
[5] `http://groovy-lang.org`
[6] `https://jbehave.org/introduction.html`
[7] `https://jbehave.org/reference/stable/using-groovy.html`
[8] `https://jbehave.org/reference/web/stable/using-selenium.html`

sure the "maven-compiler-plugin" uses at least (Java) 1.6 as source and target in the pom, and set the Selenium version to 2.53. Finally, execute the "Maven install" command.

1.2.3 Dependencies

- **Exercise 4.** Use and document a recent version of Selenium with a recent version of Firefox. Do note that specific versions of Selenium are only compatible with certain versions of Firefox. This is a bonus question. You cannot lose any points with it, only gain points on top of your total grade. You can skip this exercise if you want and use the following compatible versions of Selenium and Firefox: **(Bonus, 10 points)**

For Selenium version 2.53, you need to downgrade Firefox to version 45. For ubuntu:

```
$ sudo apt-get purge firefox
$ sudo apt-cache show firefox | grep Version
$ sudo apt-get install firefox=45.0.2+build1-0ubuntu1
```

For the following exercises, you can use the Selenium IDE to export code, but these scripts should be runnable java programs (use maven).

- **Exercise 5.** Create the same script from exercise 3, you can export its code, but make sure that the script can run using javac. Document your steps. **(Required, 10 points)**
- **Exercise 6.** Create a script that navigates to the "Software Testing" course. For each assignment, verify that the link to the assignment is: "/system/files/uploads/courses/Testing/assignment<NR>.pdf". The script should fail when the link layout differs. Make sure that the script is dynamic! Adding or removing assignments should not result in a failure. Verify that the document (link) exists on the server. Give a warning when it does not exist. **(Required, 20 points)**
- **Exercise 7.** Create a function with student and group as input that verifies whether the student is in the provided group. Give a warning when the student is not in the provided group. Fail if the student cannot be found in any of the groups. Create a script that verifies that you are in the correct group. **(Required, 20 points)**
- **Exercise 8.** Using the function from exercise 7:

  A) Create a function that returns the date a student needs to present a lecture.

  B) Create a function that returns the date a student is an opponent.

  Create a script that verifies when you needed to present/were the opponent using the above functions. **(Required, 20 points)**
- **Exercise 9.** Repeat the selenium webdriver exercises using a BDD framework. **(Only when late, -35 to 0 points)**
- **Exercise 10.** Given your experience with creating scripts using the Selenium IDE and writing them in code, reflect on their strengths. If you used BDD, reflect on your experience using it, what are its advantages/disadvantages? If you did not use BDD, what do you think are the advantages/disadvantages of using it? **(Required, 10 points)**