

Evaluating Selenium for Automated Testing

Reynaldo Teodoro

Org 9351 – Engineering Solutions



Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Internship goal

- Evaluate an alternative automated testing tool for a large, nuclear weapons related web application
 - Currently using SmartBear TestComplete
 - Only runs on Windows
 - Requires user to be logged into the console
 - Difficult to integrated into automated testing tools in Sandia's environment due to cyber restrictions
 - Evaluating Selenium – a popular, open source web testing tool
 - Can run without requiring a console at all (headless)
 - Can run on other operating systems such as Linux
- Chose a website template with representative UI widgets and created automated tests to demonstrate Selenium's viability for this project

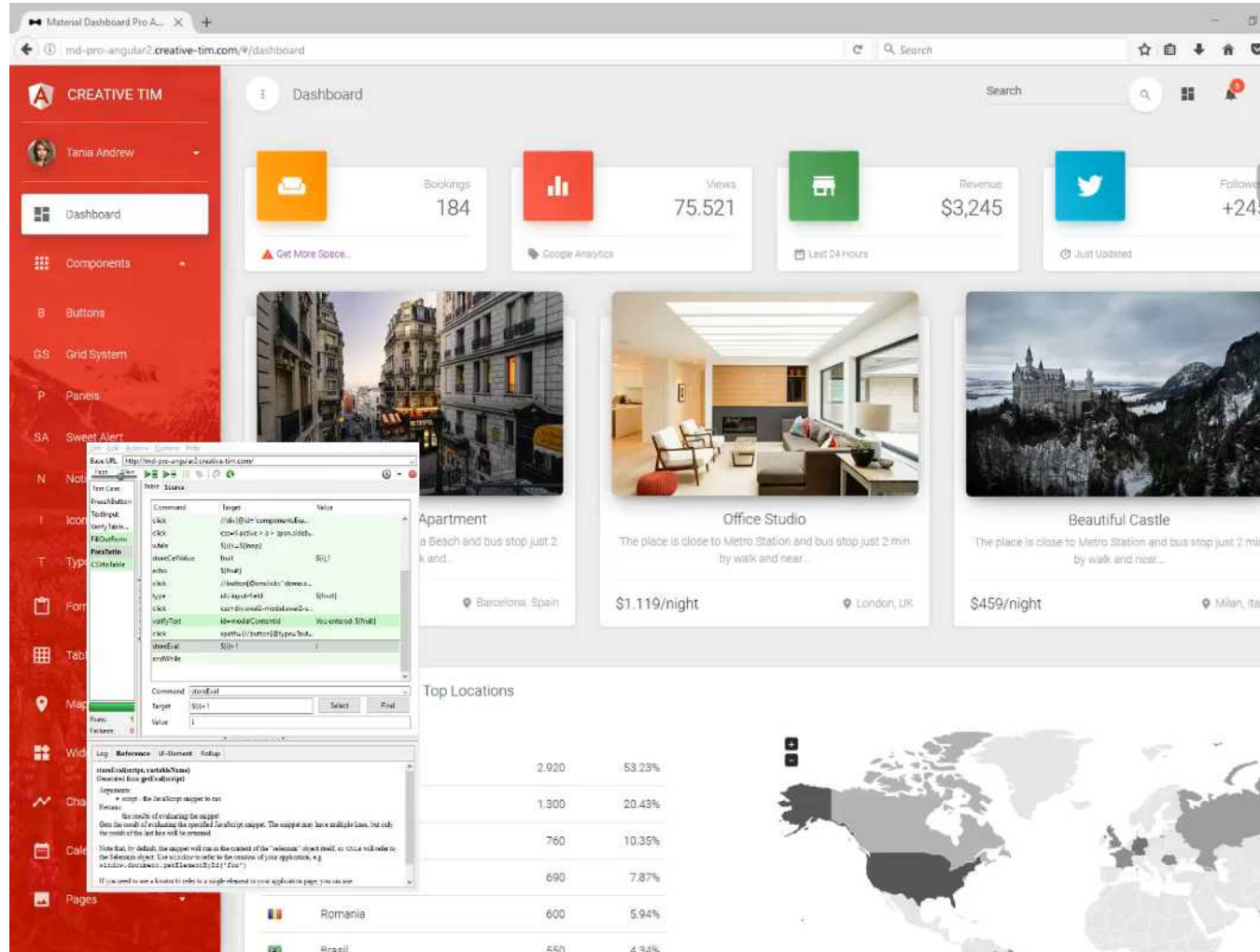
Why Automated Testing?

- Application requires high quality, so testing is essential
- Manually testing is difficult, time-consuming, and costly
 - Often done quickly when projects are under schedule pressure, leading to missed issues
 - Only run periodically, so issues can go undetected for a long period of time
 - Leads to increased costs as testing must be repeated when anything changes
- Automated tests can be run nightly or even whenever a developer checks in code
 - Instant feedback to the developers
 - Easy to run repeatedly whenever any change is made

Selenium Overview

- Testers use the Selenium IDE to generate tests
 - Plug-in into Firefox
 - Navigate just like a user, identifying aspects to test and verify
- Can generate Java code using a standard test framework (JUnit)
 - Developers can tweak the generated code for minor changes to the system
- Java code can be added into an automated process that builds and tests the application
 - Generates a report so developers can know if anything broke

Selenium IDE



The screenshot shows a Selenium IDE session on a web browser. The browser window displays a dashboard for 'CREATIVE TIM' with metrics for Bookings (184), Views (75,521), Revenue (\$3,245), and Followers (+245). Below these are three property cards: 'Apartment' in Barcelona, Spain; 'Office Studio' in London, UK; and 'Beautiful Castle' in Milan, Italy. A 'Top Locations' table is visible at the bottom right of the browser view.

Location	Count	Percentage
Romania	2,920	53.23%
Russia	1,300	20.43%
USA	760	10.35%
France	690	7.87%
Germany	600	5.94%
Spain	550	4.94%

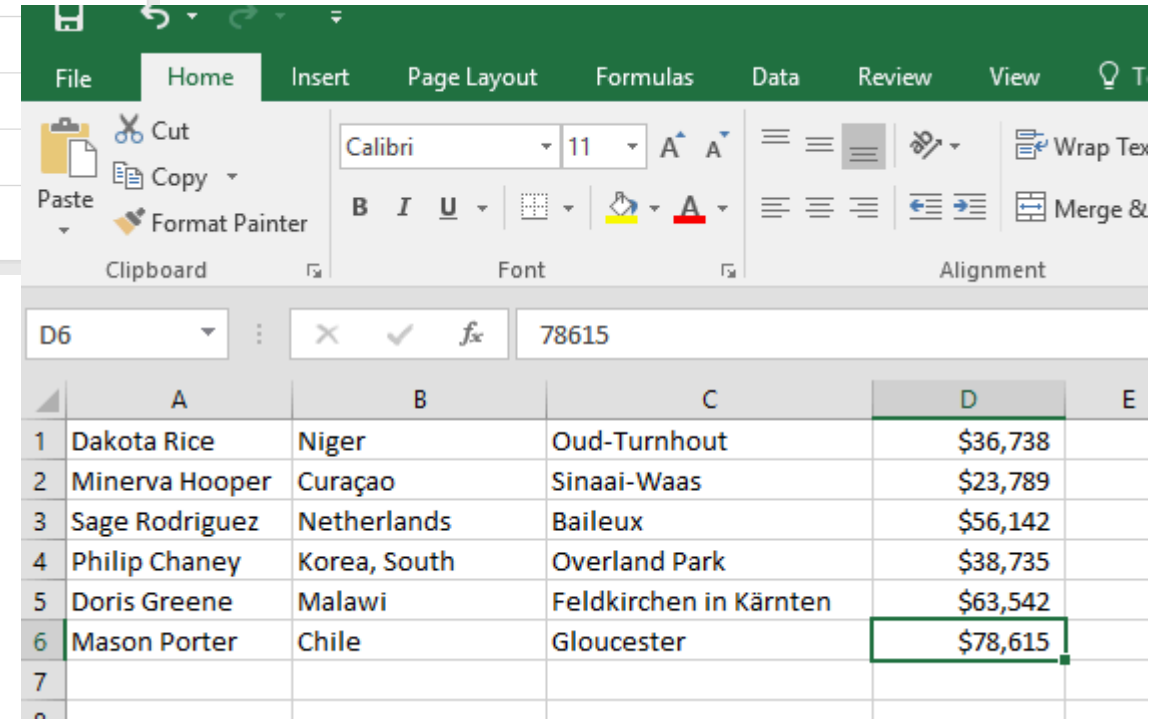
The Selenium IDE interface shows a list of test steps with columns for Comment, Target, and Value. The log window at the bottom displays the execution results of the test script.

<http://www.apache.org/licenses/LICENSE-2.0>

Example Application

Simple Table
⚙️

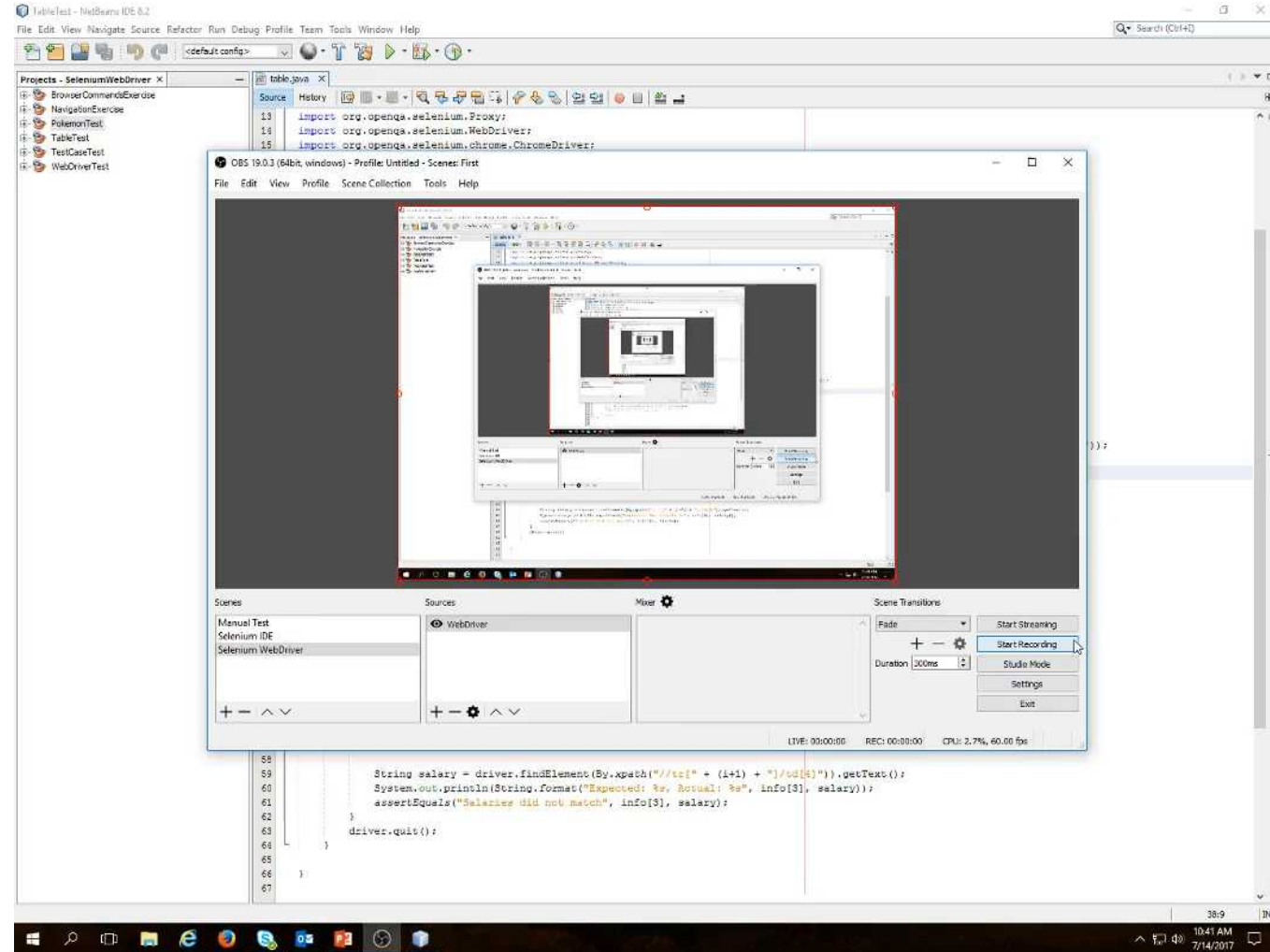
Name	Country	City	Salary
Dakota Rice	Niger	Oud-Turnhout	\$36,738
Minerva Hooper	Curaçao	Sinaai-Waas	\$23,789
Sage Rodriguez	Netherlands	Baileux	\$56,142
Philip Chaney	Korea, South	Overland Park	\$38,735
Doris Greene	Malawi	Feldkirchen in Kärnten	\$63,542
Mason Porter	Chile	Gloucester	\$78,615



	A	B	C	D	E
1	Dakota Rice	Niger	Oud-Turnhout	\$36,738	
2	Minerva Hooper	Curaçao	Sinaai-Waas	\$23,789	
3	Sage Rodriguez	Netherlands	Baileux	\$56,142	
4	Philip Chaney	Korea, South	Overland Park	\$38,735	
5	Doris Greene	Malawi	Feldkirchen in Kärnten	\$63,542	
6	Mason Porter	Chile	Gloucester	\$78,615	
7					

- You can also verify values within a website by comparing them against expected values on an outside source like an excel file

Generated Java Code



The screenshot shows the NetBeans IDE environment. The main editor displays the following Java code for Selenium WebDriver:

```
13 import org.openqa.selenium.Proxy;
14 import org.openqa.selenium.WebDriver;
15 import org.openqa.selenium.chrome.ChromeDriver;
```

Below the code, the Selenium IDE interface is visible, which is being recorded by OBS Studio. The OBS Studio window shows the Selenium IDE interface with a red border, indicating it is the active scene. The OBS Studio interface includes a 'Start Recording' button, a 'Duration' of 00:00, and a 'CPU' usage of 2.7%.

```
58
59     String salary = driver.findElement(By.xpath("//tr[*] + (L+1) + ")//td[4]")).getText();
60     System.out.println(String.format("Expected: %s, Actual: %s", info[3], salary));
61     assertEquals("Salaries did not match", info[3], salary);
62
63     driver.quit();
64
65
66
67
```

Conclusions

- Selenium appears to be a viable automated testing solution
- Easy for Testers to use the Selenium IDE to generate new tests through the record feature
- Can generate Java code that can then be automated

Thank you!