International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 06 Issue: 06 | June 2019 www.irjet.net p-ISSN: 2395-0072

# Automatic Device Functional Testing

**Miss. Pravisha D. Divsekar[1], Prof. Sonia Kuwelkar[2]**

*[1]Goa College of Engineering, Goa, India*
*[2]Goa College of Engineering, Goa, India*
*[1]Student, [2]Asst. Prof, Dept. of Electronics and Telecommunication Engineering, Goa College of Engineering, Goa, India*

---

**Abstract -** *Testing plays a key role in the software development process. Manual testing of applications is highly costly and more tedious when considered for iterative tests. Testing is also time consuming. Therefore, the need of the hour is to reduce testing time and effort by automating the testing process. Selenium is an online web-based application tool which is used for testing of web applications. In this paper, the design and functionality of Selenium tool and configuration of the device through Selenium tool is studied.*

***Key Words***: **Selenium, Device testing, Python scripts, Report generation, Excel.**

## 1. INTRODUCTION

Discovering errors in the software application is the main aim of software testing. More than 50% of the time is consumed for software testing, during the software development lifecycle. When testing is performed, the testing time is completely dependent on the type of algorithm being used, programming language used for writing the script, number of code lines and the interfaces enabled with the device [3].

Manual testing is the process in which the tester executes the test scripts manually to find errors in the testing process without any use of testing tool. The manual process of testing would be time consuming and would also require a lot of human efforts when considering large number of test cases to be run.

Automated testing is the process in which the tester develops test scripts using programming languages to test the software in order to find bugs. The advantages of using automated testing is 1) improving the accuracy of the test, 2) saving the testing time, 3) and saving the resources being used.

Test Automation is developed to automate the repetitive type of test in order to lay a comparison between the desired and the actual results that will be obtained after running the test. Advantages of using test automation are – a) can be used for repetitive testing of software application, b) reduces human efforts while performing the test, c) Finding bugs that are not identified during the manual testing [1].

So, numerous web-based application testing tools are available in the market to make the testing process simpler. Selenium is one of the testing tools that is widely been used to make the testing process simpler and easier. Selenium can work very easily on three different operating systems such as Windows, Linux and Mac. Selenium also provides a wide range of flexibility to work with different programming languages such as C#, Java, Ruby, Perl, HP, and Python.

## 2. SELENIUM ARCHITECTURE

Figure 1 shows the architecture of Selenium Web Automation tool. Selenium tool consists of two main components namely Selenium Client and Selenium Server. Selenium Client is further classified into two categories namely Web Driver API and Remote Web Driver. Selenium Server is also further classified into three categories namely Server Components, Web driver API, Selenium Grid.

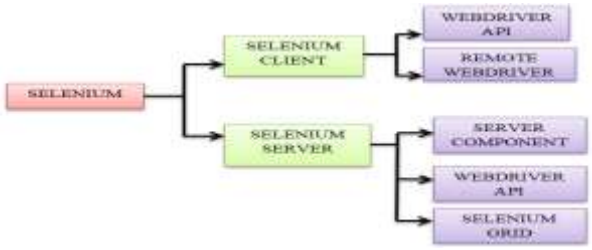


**Fig -1**: Architecture of Selenium Tool

Web Driver API of selenium client is used to generate test scripts for different applications and to interact with the web browser. Remote Web Driver of selenium client is used for communication with the Selenium server. The server component of selenium server is used to receive requests from selenium client's remote web driver. Web driver API of Server component is used to run the tests on web browser on server side. The selenium grid of selenium server' component is implemented by server. Selenium grid tool is used to run concurrent test scripts on different web browser. [2].

## 3. SELENIUM WEBDRIVER

Web driver performs the role of controlling the browser action, drives the browser page and uses the browser's own engine to control it. Web driver interacts with the web

elements on the web page by locating these elements on the page and then applying the actions that needs to be performed on the web elements. In order to find the web elements on the web page, locators are used to identify these elements. [1] There are different methods by which web elements can be identified, which are listed below.

- Name – name attribute is used to locate the element
- Id - id attribute is used to locate the element
- Class – class attribute is used to locate the element
- Xpath – xpath attribute is used to locate the element
- Tag name – tag names are used to locate the element
- CSS selector – CSS selector engine of web driver is used to locate the element
- Link text – using link of an element to locate
- Partial link text – using partial link of an element.

## 4. DESIGN FLOW

Following steps are executed to test the functionalities of the device using selenium tool.

STEP 1: First, enter the web URL of the device application and explore the xpath of all the components of device. Create an Excel file which will be namely called as 'Driver file' and enter the xpath of the components that user wants to navigate. In the driver file, we have described the type of operation that needs to be performed for the xpath's. Figure 2 shows the structure of the driver file that must be first updated before running the test. Figure 3 shows the different occ files that are kept in the driver file, which will start executing after the device parameters are set. The occ files contains the set of different test cases to be applied to check the device functionality. Occ files contains state sequencer which runs a set of test states in a defined manner for some time duration.

STEP 2: Once the driver file is ready, execute the script. The script is built using python programming language. Figure 4 below shows the execution of python script to execute the test.

| | | |
|---|---|---|
| open web browser | navigate_page | DUT home page |
| enter the password | set_val | |
| click | click_event | |
| navigate to configure window | click_event | |
| click option | click_event | |
| Expert mode | click_event | Yes |
| Power frequency | set_val | 50 Hz |
| Rated voltage (Vrated) | set_val | 11kV |
| Enabled | set_val | 3 |
| LED ENABLE | click_event | |
| LED indication for Temporary phase fault | check_event | set |
| LED indication for Permanent phase fault | check_event | set |
| LED indication for Temporary di/dt fault | check_event | set |
| LED indication for Permanent di/dt fault | check_event | set |
| LED indication for Battery Low | check_event | set |
| LED indication for Wireless | check_event | set |
| RESET ENABLE | click_event | |
| Enable Reset for Voltage restoration | check_event | set |
| Enable Reset for Fault indication time expired | check_event | set |
| Enable Reset for Wireless command | check_event | set |
| send | click_event | |
| send | click_event | |
| Binary Output -> P2/1 | set_val | Permanent Phase Fault |
| Binary Output -> P4/3 | set_val | Permanent Phase Fault |
| Binary Output -> P5/6/7 | set_val | Permanent Phase Fault |

**Fig -2**: Driver file in excel setting parameters

| Title | Data |
|---|---|
| Test_Name | |
| DUT_Name | |
| Test Repeat Count | NA |
| Test Type | NA |
| occ files | 10MM_50Hz_2.5_2.0Cycle.occ |
| occ files | NA |

**Fig -3**: Driver file containing occ scripts

STEP 3: For the test we have chosen the Chrome browser for device application. After executing the script by running from the command prompt, the application will first navigate to the device page, enter the password, click on login and then will further proceed to the next tab in the device application.


**Fig -4**: Python Script execution in Command Prompt

STEP 4: After performing all the operations of webpage using selenium tool such as clicking the checkbox event, radio button, button, choosing element from dropdown menu list, it will activate the settings of the device. After the device activation is complete (note that device activation takes 20 seconds).

STEP 5: After the activation is done and the settings are applied to the device, through python scripts new excel file is created and the settings that has been applied to the device are written in excel file and these results are saved as 'Result filename'.

## 5. EXPERIMENTAL RESULTS

The figures depict the results after successful execution of the python script to update the parameters of the device via selenium tool. Fig. 5 shows the results obtained using Selenium Testing Tool (a) displays the image of chrome browser being controlled by python script using selenium tool. (b) Image of the device parameters, where no settings are applied, before running the test. (c) Image of the device parameters, after updating the settings of the device through automation browser using python script. (d) Parameters being updated during the run of the test. (e) Functional testing of device after running the selenium test case to observe the device behaviour under different states. (f) After updating the parameters of the device through automation tool, storing results in Excel file.


**Fig -5** (a): Chrome browser controlled by python script using selenium tool.

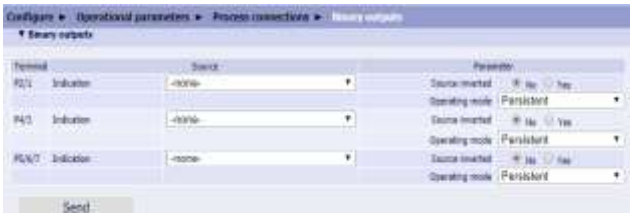


**Fig -5** (b): Device parameters before running the test.


**Fig-5** (c): Device parameters getting updates after running the test.


**Fig -5** (d) Device parameters being updated during run of the test.

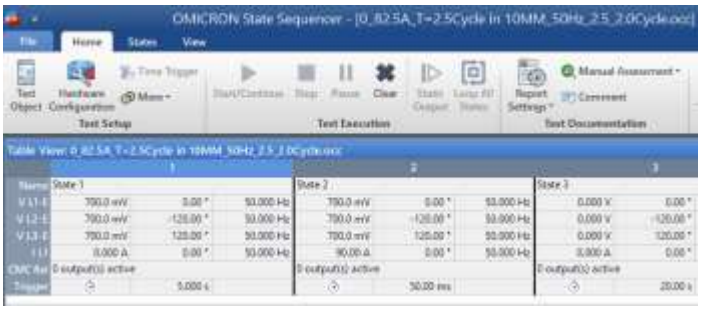

**Fig -5** (e) Device functional testing after running the selenium test case.

**Fig -5** (f) Result stored in excel file after successful run of the test.

## 6. CONCLUSION

Selenium is an open source online tool for web application. Selenium has gained a wide acceptance in short span of time because of its rich features in the testing domain. Online updating of parameters of the device was successfully implemented using Selenium tool. We also conclude that, when test was performed manually to update the device parameters it takes 5 minutes to update the settings, whereas when automated script was run through command prompt to control the browser, it takes only 30 seconds to update the device settings. The device functionality test was successfully tested using omicron device under different conditions at different time intervals.

## ACKNOWLEDGEMENT

I would like to sincerely thank Mr. Rupesh Porob, Siemens Ltd, Goa for his constant support and guidance in the technical field to complete this project. I would also like to thank Dr. H. G. Virani, Prof. and HOD of ETC Dept and Prof. Sonia Kuwelkar, GEC for their continuous support and encouragement for completing this project

## REFERENCES

[1] Renu Patil, Rohini Temkar, “Intelligent Testing Tool: Selenium Web Driver”, irjet Volume 4, Issue 6, June 2017.

[2] Jagannatha, Niranjanamurthy M, Manushree SP, Chiatra GS, “Comparative Study on Automation Testing using Selenium Testing Framework and QTP” ijcsmc Volume 3, Issue10, October 2014, pg. 258-267

[3] Jyoti Devi, Kirti Bhatia, Rohini Sharma, “A Study on Functioning of Selenium Automation Testing Structure”, ijarcsse Volume 7, Issue 5, May 2017.

[4] Neha Bhateja, “A Study on Various Software AutomationTesting Tools”, ijarcsse Volume 5, Issue 6, June 2015.

[5] Satish Gojare, Rahul Joshi, Dhanashree Gaigaware, “Analysis and Design of Selenium WebDriver Automation Testing Framework” 2nd International Symposium on Big Data and Cloud Computing (ISBCC’15)

[6] Purnima Bindal, Sonika Gupta, "Test Automation Selenium WebDriver using TestNG", JECAS Volume 3, No.9, September 2014.
[7] http://www.seleniumhq.org/
[8] http://www.tutorialspoint.com/selenium