International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 02 | Feb 2020 www.irjet.net p-ISSN: 2395-0072

A Valuable and Speculative Approach to Manage the Item Testing by using Selenium Robotized Mechanical Assembly and Testing My Venture

P. Venkata Adi Narayana*, Dr.A.SwarupaRani**

**PG Scholar, Dept. of MCA, Sietk, Puttur*
***HOD, Dept. of MCA, Sietk, Puttur, A.P.*

Abstract:- Nowadays we can do automation testing by using different types of automated testing tools. We can use Selenium for the purpose of analysis. It is an open-source software testing tool for testing software applications. Software testing is the process of identifying the correctness of software by using Selenium. The software testing provides an independent view and objective of the software. Testing involves all components under the user requirements. It will be given a result of whether that component is work and satisfied with the user requirements. And it will be given information about the quality of the software. In software testing, test engineers will be involved to identify errors in software applications. Testing is done by either manual testing or automation testing. In manual testing we can test any software application it is time taken process. So, selenium tool will test the automated testing for web applications across different browsers and platforms. It is quite similar to HP Quick Test Pro (QTP now UFT) only that Selenium focuses on automating. Testing done by selenium automated tested tools. This paper represents a practical approach and theoretical knowledge of software testing. By using some testing constraints we can check practically to approach the automated testing. The selenium automation testing is given assure to the client and to give the best quality product.

Keywords:-Testing, Selenium setup, Web Drivers, Browsers, Manual Testing, Automation Testing.

Introduction

Selenium is open source programming. It will be utilized to test the product application by utilizing robotized testing apparatuses. Selenium suit for all sort of web applications crosswise over various internet browser (chrome, Firefox, Edge, and so on). It is centered on programmed online applications.

History

Selenium was made by "Jason Huggins" in 2004. A designer at thought works he was taking a shot at a web application that necessary visit testing. When a period he understood the reiterations manual testing of this application was turning out to be increasingly keen. He made a "Java Script" program that would naturally control the activities of the program. He named the projects like the "JavaScript Test Runner***".***

He made Java Script runner open-source with was later that name can be re-named as "Selenium core". Because of this idea help automated other web applications.

Software Testing

Software testing suggests that as associate degree activity to visualize the particular results match the expected results and guarantee which the computer code is error-free. Testing involves execution or system elements to gauge one or a lot of properties of interest. “In each development of life cycle testing section is there and it's the foremost necessary innovate each life cycle. (SDLC, AGILE, Waterfall Model, Spiral Model, etc...). It conjointly helps to spot errors, a missing demand or gaps in against the particular necessities. It may be done manually or victimization machine-driven tools.

Software Testing conjointly referred to as verification of AUT (Application under Test).

- *Finding Defects.*
- *To improve the quality of the products.*
- *The Process of finding defects in software/product.*
- *Execution of the program to the intention of finding defects.*
- *Verifying functionality of an application against user requirements specification.*

Why we do Software Testing

Now a day's testing is more important because we all make mistakes. Some of the mistakes is unimportant. But some of mistakes is very expensive or very dangerous. Mostly, it will be affected by software companies.

- *To check the quality of the software.*
- *To make sure the software works according to the client's requirements.*
- *Every Software is built to support a business. If there are any defects it is going to affect customers business. So, before we launch in production make sure all problems are recognized and fixed.*

Types of Software Testing

Typically testing is 3 types.They are

1. *Functional Testing*
2. *Non-functionalTesting or Performance Testing.*
3. *Maintenance (Regression and Maintenance).*

1. Functional Testing

Testing each and every component thoroughly against requirement specification is called functional testing. It is also called as "Field Level Testing" and "Component Testing".

- *Unit Testing*
- *Integration Testing*
- *Smoke*
- *UAT (User Acceptance Testing)*
- *Localization*
- *Globalization*
- *Interoperability and so on.*

2. Non Functional Testing

Non-functional testing is outlined as a sort of software package testing to examine non-functional aspects of a software package application. It's designed to check the readiness of a system as per non-functional parameters that are ne'er self-addressed by useful testing.
An excellent example of the non-functional take a look at would be to examine what number individuals will at the same time login during a software package.

Non-functional testing is equally vital as useful testing and affects client satisfaction.

- *Performance*
- *Endurance*
- *Load*
- *Volume*
- *Scalability*
- *Usability and so on.*

3. Maintenance Testing:

Maintenance Testing is completed on the already deployed software package. The deployed software package has to be increased, modified or migrated to alternative hardware. The Testing done throughout this improvement, amendment and migration cycle is understood as maintenance testing.
Once the software package is deployed in an operational setting it desires some maintenance from time to time so as to avoid system breakdown, most of the banking software package systems got to be operational 24*7*365. Thus it's terribly necessary to try to maintenance testing of software package applications.

- *Regression and Maintenance*

In Software Testing there are two types of "Box Testing" available. They are

1. *White Box Testing.*
2. *Black Box testing.*

1. *White Box Testing* means to check one line of code (smallest). It is testing of a software solution's internal structure, design, and coding. In this testing, the code is visible to the tester. It focuses on the verifying the flow of inputs and outputs through the application. It is also called as a "Clear Box testing, Open Box testing, Structural testing, Transparent Box testing, Code-Based testing, and Glass Box testing".

- *It is done by developers rarely done by testing.*
- *To do this we need knowledge of programming.*
- *To do this you need to know internal design of the code.*

Types of white box testing: path testing, loop testing, and condition testing.
Consider the sample code:

```
Sum ( int a, int b){
        result = a+b;
        if (result>0)
                printf ("Test case is passed!");
        else
                printf("Test case is failed!");
```

The above example we can use white box testing it will be checks the all decision branches, loops, statement in the code.

- a=1, b=2: Test case is passed!
- a=1, b= -2: Test case is failed!

2. *Black Box Testing*is outlined as a testing technique during which practicality of the Application under Testing (AUT) is tested while not viewing the inner code structure, implementation details, and data of internal ways of the code. This sort of testing relies entirely on code necessities and specifications.

Fig1. Black Box Testing Black Diagram.

The Black Box testing is focus on the inputs and outputs only no bothering about to the internal knowledge of the software program.

- *Functional Testing.*
- *Non Functional Testing is under comes to the Black Box Testing.*

There are more than 150 types of testing and still adding. Also note that not all testing types are used to all in project. But depend on the nature & scope of the project.
Now, we can do testing 2 types

1. *Manual Testing*
2. *Automated Testing.*

Manual Testing is a process of finding errors or bugs in software program. In this method, the tester plays important roles of end-user. it will verify all features of the application as working correctly. The test manually executes test cases without using a any automation tools.
Life Cycle steps:

1. *Requirement Analysis*
2. *Text plan creation*
3. *Test case creation*
4. *Test case execution*
5. *Defect logging*
6. *Defect fix & Re-verification.*

Automated Testing is a Software testing technique to test and compare the actual outcome with the expected outcome. It can be achieved by writing test scripts or using any automation testing tool. Test automation is used to automate repetitive tasks and other testing tasks which are difficult to perform manually.

Life cycle Methodology
1. *Determining The Scope Of Test Automation*
2. *Selecting The Right Tool For Automation*
3. *Test Plan + Test Design + Test Strategy*
4. *Setting Up The Test Environment*
5. *Automation Test Script Development + Execution*
6. *Analysis + Generation Of Test Reports*

Automation Testing Tools

There are several types of testing tools available in the market.
They are.
1. *Katalon Studio*
2. *Test sigma*
3. *Lambda Test*
4. *Qualibrate*
5. *Cross Browser Testing*
6. *Testimony*
7. *Web Test ware (Freeware)*
8. *Selenium*
9. *Appium*
10. *Micro Focus UFT*
11. *Test Studio*
12. *IBM Rational Functional Tester etc...*

The selenium as four components:
1. *Selenium Integrated Development Environment (IDE).*
2. *Selenium Remote Control (RC).*
3. *Web Driver.*
4. *Selenium Grid.*

1. **Selenium IDE**
- To learn about concepts on automated testing and Selenium, including:
- Selenece commands such as type, open, click and wait, assert, verify, etc.
- Locators such as id, name, XPath, CSS selector, etc.
- Executing customized JavaScript code using run Script
- Exporting test cases in various formats.
- To create tests with little or no prior knowledge in programming.
- To create simple test cases and test suites that you can export later to RC or Web Driver.

- To test a web application against Firefox and Chrome only.

2. **Selenium RC**
   - To design a test using a more expressive language than Selenese.
   - To run your test against different browsers (except Html Unit) on different operating systems.
   - To deploy your tests across multiple environments using Selenium Grid.
   - To test your application against a new browser that supports JavaScript.
   - To test web applications with complex AJAX-based scenarios.
3. **Web Driver**
   - To use a certain programming language in designing your test case.
   - To test applications those are rich in AJAX-based functionalities.
   - To execute tests on the Html Unit browser.
   - To create customized test results.
4. **Selenium Grid**
   - To run your Selenium RC scripts in multiple browsers and operating systems simultaneously.
   - To run a huge test suite, that needs to complete in the soonest time possible.

IMPLEMENTATION

Installation process of Selenium tool by using third party applications

Step 1:-

Download the Web – Driver for java by using the given Linkhttps://www.seleniumhq.org/download/

Fig2. Download java for selenium.

Step 2:-

Download the Eclipse by using the given link https://www.eclipse.org/downloads/

Fig3. Download java for selenium.

Step 3:-

Download the Third-party application driver for different Web- browsers.

- Chrome for “Google Chrome Driver”.
- Mozilla for “Mozilla Gecko Driver”.
- Edge for “Microsoft Edge Driver”.

By using this Linkhttps://www.seleniumhq.org/download/

Fig4. Download Web Drivers for selenium.

Step 4:-

Make it all web drivers and selenium for java in one place.(like D, E, F, etc…)

Fig5. To Store all in One Folder.

Step 5:-

Now install eclipse in your computer it will be shows a below image. To download "Eclipse IDE for Java-Developers".

Fig6. Download Eclipse IDE for java developers.

Step 6:-

After installation complete to launch eclipse IDE for " to create new java project".

Fig7. Create a new project.

Step 7:-

Give a project name is "Selenium test". It will be taken automatically which java software available in your system (In my experience better to use java version 8.0). And click on Finish button.

Fig8. Give a Project Name.

Step 8:-

Now you can create package inside the selenium text click on right click and give package name as testing it shows a below screen.

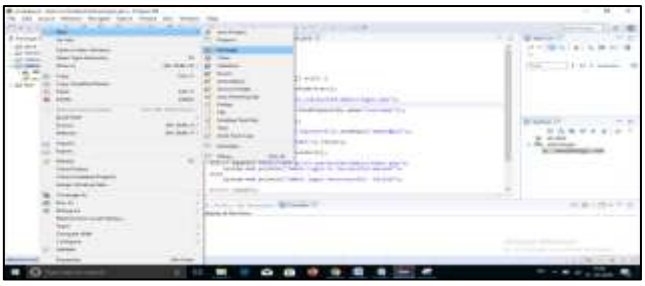

Fig9. To Create a new package.

Step 9:-

Now create a class file inside the package right click on the selenium test ▶ new ▶ class. Give a class name as "adminLogin".

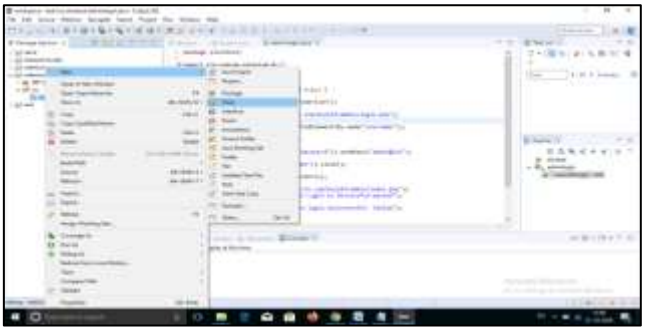

Fig10. To Create a new Class.

Step 10:-

Now we can add external jar files in Build Path.

Selenium test ⟶ add Build path ⟶ libraries ⟶ Add external jar files ⟶ Click on Apply button.

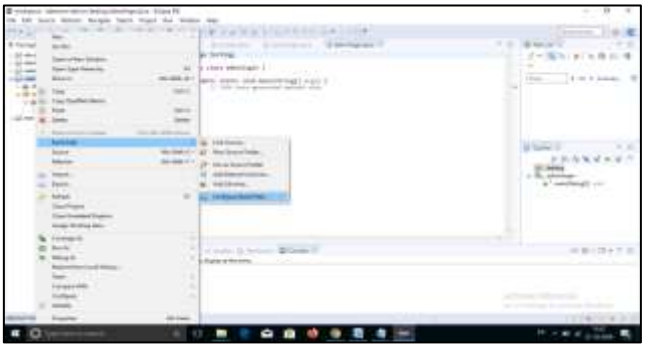

Fig11. To Add External JAR Files.

Step 11:-

Add external jars inside the selenium folder where you can save selenium software browse that one and add outside 2 jars and inside all jar files into your project(Make sure you can add every time that files into our project).

Internal jars

Fig12. To Add Inside External JAR Files.

Outside jars:-

Fig13. To Add Outside External JAR Files.

Step 12:-

Now we can create project.

- To access the web browser in our eclipse IDE. We can write one syntax
  Sy:- WebDriver variable = new browser driver name();
  Eg:-WebDriver driver = new ChromeDriver();

1. We can give property also
   E.g.:- System.getProperty("webdriver.chrome.driver","E://ChromeDriver.exe");
2. To inspect Elements
   a) *Mozilla Firefox:*
   Page Inspector: Built in Feature
   Firebug and Fire path (need to download and install)
   b) *Google Chrome*:
   Built-in Developer tools (F12)
   c) *Microsoft Edge:*
   Built-in Developer Tools (F12)
3. Selenium supports 8 element locators to recognize/locate elements in web pages.

1. id
2. name
3. className
4. tagName
5. linkText
6. partialLinkText
7. css Selector
8. xpath.

1. Id:-

Syntax:-

```
driverObject.WebDriverCommand/findelement(By.ElementLocator("Element locator
lvalue").WebdriverCommand([parameter]);
```

Ex:-

```
System.setProperty("webdriver.chrome.driver", "E://chromedriver.exe");
WebDriver driver = new ChromeDriver();
driver.get("https://www.gmail.com/");
driver.manage().window().maximize();
driver.findElement(By.id("identifierId")).sendKeys("India123");
```

2. Syntax:-

```
WebElement elementName = driverObject.WebDriverCommand(By.ElementLocator("value"));//recognize
element
//performing action on the element
elementName.WebDriverCommand(parameter Optional)
```

Ex:-

```
WebElement editbox=driver.findElement(By.id("identifierId"));
//editbox.sendKeys("India123");
Thread.sleep(3000);
String val=editbox.getAttribute("value");
```

```
System.out.println(val);
editbox.clear();
```

---

3. className

Syntax:-

```
driverObject.WebDriverCommand/findElement By.ElementLocator("value")).WebDriverCommand
([Parameter]);
```

Ex:-

```
driver.findElement (By.className("gb_e")).click();
```

---

4. tagName

Ex:- driver.findElement (By.tagName("input")).sendKeys("123456");

---

5. LinkText

Ex:-driver.findElemen t(By.linkText("Gmail")).click();

---

6. partialLinkText

Ex:-driver.findElement (By.partialLinkText("ma")).click();

---

7. CSS Selector

Ex: - driver.findElement (By.cssSelector("#identifierId")).sendKeys("123");

---

8. X-path

Ex:-driver.findElement (By.xpath ("//*[@id=\'identifierId\']")).sendKeys ("123");

Now we can write Manual Test Cases:-

Test Case Name: Verify admin Login into gcrshop Admin Interface
Test Steps:
1) Launch Browser
2) Navigate to GCR shop admin interface ("http://www.gcrit.com/build3/admin")
3) Enter valid user name
4) Enter valid Password
5) Click "Login" Button
Input/Test Data:
Username: admin
Password: admin@123

Verification Point:
Capture the current URL after login to the application and compare with expected Result.

Expected Result:
"http://www.gcrit.cm/build3/admin/login.php"

Actual Result:*After Execution (pass or file).

Now we can test by using selenium testing tool software:

Program:-

```
//positive test case
package testing;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
public class adminlogin {
```

```
public static void main(String[] args) throws InterruptedException {
                                        System.setProperty("webdriver.chrome.driver", "E:/chromedriver.exe");
                                        WebDriver driver = new ChromeDriver();
//Maximize the Browser Window
                                        driver.manage().window().maximize();
//navigate to gcr admin interface shop
                                        driver.get("http:\\www.gcrit.com/build3/admin/login.php");
//web driver test step
//idetify the user name and enter "admin" value
//whenever we perform single operations in test case
//driver.findElement(By.name("username")).sendKeys("admin");
//identify the user name edit box
//whenever we perform multiple operations
                                        WebElement username=driver.findElement(By.name("username"));
//enter "admin" data in to the edit box
                                        username.sendKeys("admin");
                                        driver.findElement(By.name("password")).sendKeys("admin@123");
                                        driver.findElement(By.id("tdb1")).click();
                                        Thread.sleep(4000);
                                        String url = driver.getCurrentUrl();
                                        if(url.equals("http://www.gcrit.com/build3/admin/index.php"))
                                        System.out.println("Admin Login is Successful");
                                        else
                                        System.out.println("Admin login Unsuccessful- Failed");
                                        driver.close();
                                        }
}
```

To run above code that will be shows a below screen:-

Fig14. Output Screen for GCR shop.

Fig15. Login Successful in positive test case.

Fig16. Test case for GCR shop.

Program

```
//negative test case
package testing;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
```

```
public class adminlogin {
public static void main(String[] args) throws InterruptedException {
                                        System.setProperty("webdriver.chrome.driver", "E:/chromedriver.exe");
                                        WebDriver driver = new ChromeDriver();
//Maximize the Browser Window
                                        driver.manage().window().maximize();
//navigate to gcr admin interface shop
                                        driver.get("http:\\www.gcrit.com/build3/admin/login.php");
//web driver test step
//idetify the user name and enter "admin" value
//whenever we perform single operations in test case
//driver.findElement(By.name("username")).sendKeys("admin");
//identify the user name edit box
//whenever we perform multiple operations
                                        WebElement username=driver.findElement(By.name("username"));
//enter "admin" data in to the edit box
                                        username.sendKeys("admina");
                                        driver.findElement(By.name("password")).sendKeys("admin@123");
                                        driver.findElement(By.id("tdb1")).click();
                                        Thread.sleep(4000);
                                        String url = driver.getCurrentUrl();
                                        if(url.equals("http://www.gcrit.com/build3/admin/index.php"))
                                        System.out.println("Admin Login is Successful");
                                        else
                                        System.out.println("Admin login Unsuccessful- Failed");
                                        driver.close();
                                        }
}
```

To run above code that will be shows a below screen:-

Fig17. Login Unsuccessful in Negative test case.

Login fail in negative test cases:

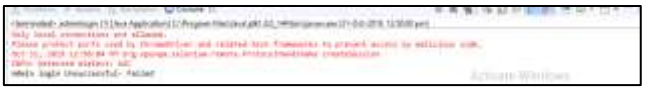

Fig18. Test case for GCR shop.

Final Result: The above test case for Manual and Automate testing is successfully passed.

Conclusion

The product computerizes testing devices will be utilized consummately in the product organizations. Present days Selenium is the most well-known testing apparatus and by utilizing the outsider application we can test any product application effectively (required fundamental java knowledge).Software testing is a crucial a piece of the product bundle advancement strategy. It's not one action that happens when code usage, in any case, is a component of each phase of the existence cycle. A definite fire take a gander at system can start circumspectly all through requirements particular. Testing subtleties got toss elevated level and low-level framework styles, and investigate are administrated by engineers and separate test groups when code usage. In the product lifecycle (SDLC), testing has its one kind of difficulties. As programming analyser to check every single segment by utilizing manual testing it is exceptionally perplexing. The significance of compelling, admirably arranged testing endeavours will just increment.In this paper, I can be done by manual testing and robotization testing. In the manual testing, we can compose situations of the product application. In

robotization testing, I can do by utilizing selenium and eclipse (IDE). What's more, there two experiments will be taken one is positive trying and second is negative trying. Both experiments will be effectively fulfilled the client's necessities.

References:

1. Sharma, Bharadwaj (April 2016). "Ardentia Technologies: Providing Cutting Edge Software Solutions and Comprehensive Testing Services". *CIOReview* (India ed.). Retrieved December 20, 2017.
2. Ammann, Paul; Offutt, Jeff (January 28, 2008). *Introduction to Software Testing*. Cambridge University Press. p. 215. ISBN 978-0-521-88038-1. Retrieved November 29, 2017.
3. Woods, Anthony J. (June 5, 2015). "Operational Acceptance – an application of the ISO 29119 Software Testing standard" (Whitepaper). Capgemini Australia. Retrieved January 9, 2018.
4. *Kaner, Cem; Falk, Jack; Nguyen, Hung Q. (April 26, 1999). Testing Computer Software. John Wiley & Sons. ISBN 9780471358466.*
5. *Kaner, Cem; Falk, Jack; Nguyen, Hung Quoc (1999). Testing Computer Software, 2nd Ed. New York, et al.: John Wiley and Sons, Inc. ISBN 978-0-471-35846-6.*

About authors:

[1] **Mr. P. Venkata Adi Narayana** is currently pursuing MCA in Siddhartha Institute of Engineering & Technology, Puttur, Andhra Pradesh, India .517583.

[2] **DR. A. Swarupa Rani,**
HOD in Dept. of MCA, Siddharth Institute of Engineering & Technology, Puttur, Andhra Pradesh, India.517583.