Test Automation of Web Application Login Page by Using Selenium IDE in a Web Browser

V. Vamsi Krishna  
Research Scholar, School of Computer Science, Engineering & Applications, Bharathidasan University, Tiruchirappalli, Tamil Nadu, India.

G. Gopinath  
Professor, School of Computer Science, Engineering & Applications, Bharathidasan University, Tiruchirappalli, Tamil Nadu, India.

Received March 24, 2021; Accepted July 08, 2021  
ISSN: 1735-188X  
DOI: 10.14704 WEB/V18S04/WEB18160

Abstract

Selenium stands as open-source program apparatus aimed at computerization testing. Here, selenium-IDE (Integrated Developments Environment) Testing is investigated, which could be a remarkably vital movement in Program Improvement Handle. It is to look at and alter source code. A high-quality computer program can well be achieved by producing Compelling Testing. This Paper handles a critical in addition to the crucial issue of Program Testing. Testing can well be executed both manually and automatically. This paper sets to do Mechanization Testing through "Selenium". Test cases (TC) are consequently recorded with this web testing tool whilst the analyzer is entering the information in a web application screen.

Keywords

Manual Testing, Test Automation, Test Cases, Test Data, Selenium IDE.

Introduction

An augmenting field that draws the utmost benefits with least exertion computerization testing for extending the software quality is Automation testing [Xu D, Xu W, Bavikati BK, Wong WE, 2012]. A few program testing apparatuses are used in automation testing. Test mechanization involves employing a programming apparatus for running repeating tests on the application. For testing computerization, there are several preferences. Most are joined to the test's repeatability and the test's execution speed. For the growth of test mechanization, there is disparate commercial along with open-source instruments available [S. Doğan, A. Betin-Can, and V. Garousi, 2014]. The most widely-employed
open-source arrangement is Selenium. Successful methods are learned by selenium clients in creating test mechanization aimed at web applications. Open-source tool, i.e. selenium, is a computer program testing tool presented in this paper. In a web application, the selenium apparatus is utilized for testing its support in different browsers along with platforms. The components and their functionalities are characterized in selenium instruments. The software testing advantage is to identify all the surrenders existing in a computer program item [Y.F. Li, P.K. Das, and D.L. Dowe, 2014]. There exists no plausibility of software being blunder-free. Even with this, the significance of testing must not be thought little. Numerous abandons existing in a computer program item are uncovered by testing. A realistic way of reducing absconds in a framework and expanding the users’ certainty in a created framework is provided by testing. The solution to this problem is mechanization testing [You X, Wang Y, 2019]. It is the method of computerizing the manual TC using a computer program test instrument for Selenium. Furthermore, human mistakes are expelled [V. Garousi, A. Mesbah, A. Betin-Can, and S. Mirshokraie, 2013]. Producing high fine software is the primary objective of software development. Characteristics like low cost, reliable and consumer satisfaction are possessed by a good software program. The procedure of executing the software aimed at finding out errors is testing. It is a necessary activity for finding out all the mistakes in the software program improvement process. Emphatic and productive testing reduces the device cost. For executing a testing activity, software development homes employ excellent assurance personnel. A program’s response to ever-viable input is termed a test. Testing exercises could be executed in ‘2’ ways: (i) Manual trying out and (ii) automation testing. Any sort of software program testing can be completed manually as well as utilizing an automation tool [Selenium Tool in Software Testing, 2019].

**Existing Work**

Manual testing (MT) was defined as a testing activity that was executed through testing persons. A tester was needed by MT for functioning manual check operations on the test software excluding the assist of Test automation. MT was a method wherein a tester looks out for the flaws in the software by following a written test plan including sets of various TC [A.M. Memon and B.N. Nguyen, 2010]. A group of prerequisites written for particular purposes was a TC in software testing. All these conditions were run by the tester for ensuring the right operations of the software program applications. A laborious activity, i.e. MT, was needed by the tester to have a particular set of qualities like patience, observant, speculative, creativity, and innovation, open-mindedness, resourceful, opinionated, and skillful. For checking whether every requirement of an application is met, there ought to be at least '2' unit TC for each requirement: (i) one positive test and (ii)
one negative test. The defects related to the usability testing along with the GUI testing area were found by MT. Any new application must be manually examined before it could be automated with the aid of any tool. A greater effort was required by MT but is vital to make sure automation feasibility [Q. Xie and A.M. Memon, 2007]. The knowledge of any testing equipment was not needed by the MT concepts.

Proposing Work

Automation Testing tests the TCs; aimed at executing each one, the place guide intervention is not needed. It utilizes particular software for writing & performing check instances to analogize the real impact with the envisioned result. They can operate quickly and also repeatedly after the evaluations are automated. Implementing the automated software program is a suitable way to boost up the software testing effectiveness, efficacy, and coverage. Automation testing needs a huge extent of funding to purchase the software program & well-apt hardware resources [V. Garousi and M.V. Mäntylä, 2016]. It performs what MT doesn’t perform. It enhances the accuracy and preserves the tester’s time and the organization’s money. It stands as a big ideal in the environment wherein the requirements are changing regularly and a large quantity of regression testing is needed to be executed. It is acceptable in the surrounding places that contain vital check TCs, which are to be performed again and again. It increments the testing shape's satisfaction and decrements the future preservation cost. Automation testing's diverse advantage is the quick execution of TCs. Reusable TCs are produced and are dependable, Programmable, and also comprehensive [S. Choudhary, M.E. Dincturk, S.M. Mirtaheri, A. Moosavi, G. von Bochmann, G.V. Jourdan, and I.V. Onut, 2012]. The vital difference betwixt MT and Automation testing is that the Automation testing is of higher-quality and suitable aimed at the surroundings which comprise more recurring work (e.g., executing regression tests, re-entering the identical test data, and testing regarding coding standards) [L. Ran, C. Dyreson, A. Andrews, R. Bryce, C. Mallery, 2009]. MT is well apt for the environment wherein the necessity continuously changes [S. Elbaum, G. Rothermel, S. Karre, and M. F. II, 2005].

Advantages

Reusability: The code aimed at the same context can be utilized over different applications. Work duplication is decremented at every level.

Increased quality: The scripts are of identical quality as they utilize identical code.
Scripting capabilities are not needed by the end-user. Coding capabilities are not needed to be aimed at computerizing and also auditing the scripts. The scripts are user-friendly comprising great benefits.

Maintenance: Simple modifications to the application can be handled easily on the code [A. Mesbah and A. van Deursen, 2009].

Less effort: The summation of adjustments needed to move from one application to another on the same phase is decremented as the code stays the same aimed at all tools.

Repeatability: Whilst we computerize, the testing method becomes ready to execute the testing several times as we needed.

Reliability: As the TCs are saved and well held up if any fault happens, it can easily test against that fault. Therefore, unwavering quality prevails here [F. Groeneveld, A. Mesbah, and A. van Deursen, 2010].

**Test Automation**

1. It is a well-known testing protocol; it is utilized with an enormous quantity of browsers and allows recording the testing in nearly every language, say Java, .net, c#. It is stated as an open-source tool as well as is executed in JavaScript. It supports numerous browsers and platforms. The components utilized in selenium are IDE, RC, along Web Driver. It comprises strong tools set, which helps rapid development of automation testing aimed at web-centered applications. It offers an affluent set of testing functions that are designed certain to fulfill the necessity of testing a web application [S. Sprenkle, H. Esquivel, B. Hazelwood, and L. Pollock, 2008]. These are greatly flexible, permitting numerous choices to locate the UI elements. Selenium upholds implementing 'I' tests on manifold platforms, which are its key features [S. Sprenkle, E. Gibson, S. Sampath, and L. Pollock, 2005].

2. **Selenium IDE:** This function is a Firefox append, and it doesn't render any iteration or conditions aimed at testing scripts but encompasses a recording and playback feature [S. E. Sprenkle, 2007]. However, it furnishes an accessible interface for developing as well as functioning individual TC or complete Test Suites (TS). Besides, it gives complete editing of check instances aimed at better precision together with control [E. Soechting, K. Dobolyi, and W. Weimer, 2009].
Steps for Implementation of Test Cases using Selenium IDE

1. **Problem analysis:** It is the technique of gathering statistics for discovering the base of positives in addition to negatives of the proposed work. For instance, let's consider a problem for strengthening an internet site for individual login. For this, a website with the DEMO is designed [D. Roest, A. Mesbah, and A. van Deursen, 2009]. Here, the net web page of a registration structure of an internet site DEMO ought to be checked [Iyama M, Kirinuki H, Tanno H, KurabayashiT, 2018]. The Demo figure is

![Facebook Login Page](http://www.seleniumhq.org/projects/ide/)

**Fig. 1.0 Facebook Login Page**

2. **Writing test cases:** TC is a vital document that includes a manner for operating testing. A TC contains a group of test inputs, execution clauses, as well as predicted output created for a precise objective, e.g. for testing a particular application route or for corroborating that the specific input will meet with the preferred output. For writing a TC, there exists no prescribed layout. But, a TC must involve input, anticipated behavior, predicted output [P. Aho, M. Suarez, A. Memon, and T. Kanstrén, 2015]. A sample TC is possessed for the first identifying area of the registration page on the DEMO website [http://www.seleniumhq.org/projects/ide/].
Fig. 1.1 Agile Methodology TC Templates

Fig. 1.2 Enter the FaceBook URL and Launch Selenium IDE
Fig 1.3 Create a New Project in the Selenium IDE

Fig 1.4 Enter the Project Name in the Selenium IDE
Fig 1.5 Project File Created in the selenium IDE

Fig 1.6 Enter the Website Base URL in the Selenium IDE
Fig 1.7 Entered Base URL and Click Start Recording Functionality

Fig 1.8 Enter the Valid Username and Valid Password in the Login Page
Fig 1.9 Face book Application Login Successfully and it displays home page user profile

Fig 1.10 Facebook Application Logout Successfully
import org.junit.Test;
import org.junit.Before;
import org.junit.After;
import static org.junit.Assert.*;
import static org.hamcrest.CoreMatchers.
is;
import static org.hamcrest.core.IsNot.not;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.remote.RemoteWebDriver;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.interactions.Actions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqaqa.selenium.support.ui.WebDriverWait;

Fig 1.11 Click Stop Recording and User Must Save the Project

// Generated by Selenium IDE for Valid TC Coding
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.Alert;
import org.openqa.selenium.Keys;
import java.util.*;
import java.net.MalformedURLException;
import java.net.URL;

public class TESTTest {
    private WebDriver driver;
    private Map<String, Object> vars;
    JavascriptExecutor js;

    @Before
    public void setUp() throws MalformedURLException {
        driver = new RemoteWebDriver(new URL("http://localhost:4444/wd/hub"),
                                      DesiredCapabilities.firefox());
        js = (JavascriptExecutor) driver;
        vars = new HashMap<String, Object>();
    }

    @After
    public void tearDown() {
        driver.quit();
    }

    @Test
    public void tEST() {
        // Test name: TEST
        // Step # | name | target | value
        // 1 | open | / |
        driver.get("https://www.facebook.com/");
        // 2 | setWindowSize | 1550x838 |
        driver.manage().window().setSize(new Dimension(1550, 838));
        // 3 | click | id=email |
        driver.findElement(By.id("email")).click();
        // 4 | type | id=email | vamsikrishnavit@gmail.com
driver.findElement(By.id("email")).sendKeys("vamsikrishnavit@gmail.com");
// 5 | click | id=pass |
driver.findElement(By.id("pass")).click();
// 6 | type | id=pass | krishnadob13061990
driver.findElement(By.id("pass")).sendKeys("krisjjjYTR13051997");
// 7 | click | id=u_0_d_f/ |
driver.findElement(By.id("u_0_d_f/")).click();
// 8 | click | css=.sp_FbjfXmDnOCG_1_5x |
driver.findElement(By.cssSelector(".sp_FbjfXmDnOCG_1_5x")).click();
// 9 | click | css=div:nth-child(4) > .oajrlxb2 .d2edcug0 |
driver.findElement(By.cssSelector("div:nth-child(4) > .oajrlxb2 .d2edcug0")).click();
// 10 | click | css=.cxgpxx05 > div > div > .oajrlxb2 > .ow4ym5g4 |
driver.findElement(By.cssSelector(".cxgpxx05 > div > div > .oajrlxb2 > .ow4ym5g4")).click();
// 11 | click | css=_95k9 |
driver.findElement(By.cssSelector("_95k9")).click();
}
}

Fig 1.12 Project File Created in the selenium IDE
Fig 1.13 Enter the Valid Username and Invalid Password in the Login Page

Fig 1.14 Facebook Application was not Login Successfully and it does not display home page user profile
// Generated by Selenium IDE for Invalid TC Coding
import org.junit.Test;
import org.junit.Before;
import org.junit.After;
import static org.junit.Assert.*;
import static org.hamcrest.CoreMatchers.is;
import static org.hamcrest.core.IsNot.not;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.remote.RemoteWebDriver;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.interactions.Actions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.Alert;
import org.openqa.selenium.Keys;
import java.util.*;
import java.net.MalformedURLException;
import java.net.URL;
public class TESTTest {
    private WebDriver driver;
    private Map<String, Object> vars;
    JavascriptExecutor js;
    @Before
    public void setUp() throws MalformedURLException {
        driver = new RemoteWebDriver(new URL("http://localhost:4444/wd/hub"),
        DesiredCapabilities.firefox());
        js = (JavascriptExecutor) driver;
        vars = new HashMap<String, Object>();
    }
    @After
    public void tearDown() {
        driver.quit();
    }
    @Test
    public void tEST() { // Test name: TEST
        // Step # | name | target | value
        // 1 | open | / |
        driver.get("https://www.facebook.com/");
        // 2 | setWindowSize | 1550x838 |
        driver.manage().window().setSize(new Dimension(1550, 838));
        // 3 | click | id=email |
        driver.findElement(By.id("email")).click();
        // 4 | type | id=email | vamsikrishnavit@gmail.com
        driver.findElement(By.id("email")).sendKeys("vamsikrishnavit@gmail.com");
        // 5 | click | id=pass |
        driver.findElement(By.id("pass")).click();
        // 6 | type | id=pass | vamGH423
        driver.findElement(By.id("pass")).sendKeys("vamGH423");
        // 7 | click | id=u_0_h_D/ |
        driver.findElement(By.id("u_0_h_D/")).click();
    }
}
Test Execution

The code execution as well as contrasting the expected with the actual results is called the test execution [Test Command Auto-Wait Mechanisms for Record and Playback-Style Web Application Testing]. The aspects that should be considered are, i) choose a sub-set of TS that are to be executed aimed at this cycle centered upon a risk, ii) Assign the TC in every TS to testers aimed at execution, iii) incessantly carrying out tests, reporting bugs, along with taking test status, iv) the blocking issues should be resolved, v) Report status, regulate assignments, and also re-regard plans as well as priorities every day, vi) Report test cycle findings along with status [Altay I, Dar JA, ul Rashid F, Rafiq M, 2015].

Test Closure

Test Completion Report

This stands as a process of reporting the test metrics in a summarized format for updating the users. This aids in taking a well-versed decision [Bruns A, Kornstadt A, Wichmann D, 2009].

Test Completion Matrix

After Test completion reporting, disparate matrices are amassed for preparing the test reports [Ramya P, Sindhura V, Sagar PV, 2017]. The criterion aimed at setting up the reports is

- Total Tests Executed
- Total Tests Passed
- Total Tests Failed
- Total Test Failed centered on every module
- Total Test Defects Raised amid the execution cycle
- Total Test Defects Accepted
- Total Test Defects Rejected
- Total Test Defects Deferred
- Status of Active defects
- Gauging Quality Index of the Build

Test Results

Test results articulate (i.e. screenshots, database query results, recording, log files, etc) ought to be saved whilst executing a TC, re-testing defects along with performing
regression TC [Razak RA, Fahrurazi FR, 2011]. It can well be generated together with the test cycle closure documents for supporting the test execution completion [de Castro AM, Macedo GA, Collins EF, Dias-Neto AC, 2013].

Conclusion

Automation Software testing stands as the best approach to attain maximal effectiveness, efficacy, and coverage of the software program's testing. Selenium is the framework that comprises several equipment types to test internet applications. With the case study's help, we examine and find an internet utility's testing as well as the utility of automation testing tool “Selenium IDE”. The TC is regularly recorded in history utilizing this methodology when the tester is entering the facts prevalent within a web software display; these TCs are reusable for the Regression Testing environment. It will reduce the manpower, cost, and time for testing purposes and to deliver the quality of the product to customers.

References


Selenium IDE. http://www.seleniumhq.org/projects/ide/


