International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 10
June 2016

A Study: Automation Technique Using Selenium Web driver

Anjaneya Bulla
Student, PG, Dept. of CSE, Sri Jayachamarajendra College of Engineering, Autonomous under VTU, Mysore, Karnataka, India
Brunda S
Assistant Professor, Dept. of CSE, Sri Jayachamarajendra College of Engineering, Autonomous under VTU, Mysore, Karnataka, India

***Abstract*—** *The automation in the present world of technology has seen tremendous progress. This has reduced manual efforts and enhanced the efficiency of the process in larger quantity. The effort needed to automate any process would take time but once it is achieved the time and resource can be utilized somewhere else without intervening into it unless there is a glitch. This will automatically bring in the change in technology and parallel resource utilization. In order to achieve high quality product the selenium can also be used to write test cases and validate on the same. This is termed as automation testing as the same set of test cases is reused and will not get affected with the change in actual software coding.*

Any website or html content which is needs human interaction to get some work done can be automated using a technique called as selenium web Driver. It could be the work of uploading files, downloading certain document, picture capturing, drag and drop of files, scroll up- scroll down, click, double click, passing some values through keyboard and many such events or tasks can be performed. In today's world of automation many of the tasks are being automated and human being is allowed to intervene only when there is a necessary due to the circumstance seen which is not proper as expected. Unless and until there is too much of tasks to be performed on the html based sites, selenium web Driver can bring in some automation to increase efficiency and reducing dependability. Selenium technique will bring about a tremendous change in the field of automation.

Keywords— group communication; access privilege; secure; join and leave protocol; group key

I. INTRODUCTION

The motivation for this paper is automation of the document uploading process. Any website which maintains the company's documents are of high importance tool. This website is used by many internet coordinators, in varying environment. Any small discrepancy found later will have tremendous effect on business. Manual approach to upload is always a tedious job because of various reasons. So, automation is required to reduce manual effort and fasten the upload process at the cost of efficient utilization of human resource.

Another motivation is time and efficiency in maintaining application. It is usual that requirements change from time to time as business process evolves. Because of this change in requirements, developer should ensure adding or removing a feature, will not have any bad impact on the application and the application is operating smoothly as intended. Testing improves usability and quality of automation application, reduces development time and number of bugs.

The paper aims to show how automation improves effective content publishing of the documents to any website.

Automation is something more than mechanization because automation is a self-regulated process in which the work is completed with minimum human efforts. The self-regulated process aims at continuous flow of information without minimum human intervention. So in brief the word automation denotes the art of recording, processing and controlling the information automatically by mechanical and electronic machine.

International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 10
June 2016

With an automated workflow, you bypass the expensive costs associated with errors and inefficiencies when a person is expected to own a business process. Your automated business process can show you the current state of any item and make sure these careless mistakes don't take place. Automation facilitates efficient and detailed information through the use of mechanical aids like computers. It ensures speedy recording. Processing and presenting of information. Increased volume of work, scarcity of time and the slow manual processes necessitate the introduction of automation. Automation increases the goodwill and reputation of the firm because it adds to the prestige and status symbol of the enterprise. It brings about a complete change in the organizational structure and involves a great deal of additional cost.

II. RELATED CONCEPTS

Selenium is a portable software testing framework for web applications. Selenium provides a record/playback tool for authoring tests without learning a test scripting language (Selenium IDE). It also provides a test domain-specific language (Selenese) to write tests in a number of popular programming languages, including Java, C#, Groovy, Perl, PHP, Python and Ruby. The tests can then be run against most modern web browsers. Selenium deploys on Windows, Linux, and Macintosh platforms. It is open-source software, released under the Apache 2.0 license, and can be downloaded and used without charge.

Selenium WebDriver accepts commands (sent in Selenese, or via a Client API) and sends them to a browser. This is implemented through a browser-specific browser driver, which sends commands to a browser, and retrieves results. Most browser drivers actually launch and access a browser application (such as Firefox or Internet Explorer); there is also an Html Unit browser driver, which simulates a browser using Html Unit.

Selenium WebDriver does not need a special server to execute tests. Instead, the WebDriver directly starts a browser instance and controls it. However, Selenium Grid can be used with WebDriver to execute tests on remote systems (see below). Where possible, WebDriver uses native operating system level functionality rather than browser-based JavaScript commands to drive the browser. This bypasses problems with subtle differences between native and JavaScript commands, including security restrictions.

The biggest change in Selenium recently has been the inclusion of the WebDriver API. Driving a browser natively as a user would either locally or on a remote machine using the Selenium Server it marks a leap forward in terms of browser automation. Selenium WebDriver fits in the same role as RC did, and has incorporated the original 1.x bindings. It refers to both the language bindings and the implementations of the individual browser controlling code. This is commonly referred to as just "WebDriver" or sometimes as Selenium 2.

Selenium 1.0 + WebDriver = Selenium 2.0

WebDriver is designed in a simpler and more concise programming interface along with addressing some limitations in the Selenium-RC API. WebDriver is a compact Object Oriented API when compared to Selenium1.0. It drives the browser much more effectively and overcomes the limitations of Selenium 1.x which affected our functional test coverage, like the file upload or download, pop-ups and dialogs barrier. WebDriver overcomes the limitation of Selenium RC's Single Host origin policy

WebDriver is the name of the key interface against which tests should be written in Java, the implementing classes one should use are listed – AndroidDriver, ChromeDriver, EventFiringWebDriver, SafariDriver, PhantomJSDriver, HtmlUnitInternetExplorerDriver, RemoteWebDriver, FirefoxDriver. Selenium-WebDriver makes direct calls to the browser using each browser's native support for automation. How these direct calls are made, and the features they support depends on the browser you are using

The main aspect which has to considered is that the webDriver only captures the webelements which are visible on the screen i.e., DOM elements.

III. SELENIUM WEBDRIVER API

WebDriver is a tool for automating web application testing, and in particular to verify that they work as expected. It aims to provide a friendly API that's easy to explore and understand, easier to use than the Selenium-RC (1.0) API, which will help to make your tests easier to read and maintain. It's not tied to any particular test framework, so it can be used equally well in a unit testing project or from a plain old "main" method.

```
using OpenQA.Selenium;
using OpenQA.Selenium.Firefox;
using OpenQA.Selenium.Support.UI;
```

Selenium-WebDriver API Commands and Operations

International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 10
June 2016

a) Fetching a Page - Dependent on several factors, including the OS/Browser combination, WebDriver may or may not wait for the page to load. In some circumstances, WebDriver may return control before the page has finished, or even started, loading. To ensure robustness, you need to wait for the element(s) to exist in the page using Explicit and Implicit Waits.

```
driver.Url = "http://www.google.com";
```

b) Locating UI Elements (WebElements) - Locating elements in WebDriver can be done on the WebDriver instance itself or on a WebElement. Each of the language bindings exposes a "Find Element" and "Find Elements" method. The former returns a WebElement object matching the query and throw an exception if such an element cannot be found. The latter returns a list of WebElements, possibly empty if no DOM elements match the query.

By ID, By ClassName, By CSS, By XPath, By TagName, By Name, By LinkText, By PartialLinkText

c) Getting text values - People often wish to retrieve the inner Text value contained within an element. This returns a single string value. Note that this will only return the visible text displayed on the page.

```
IWebElement element = driver.findElement(By.id("elementID"));
element.Text;
```

d) Moving Between Windows and Frames - Some web applications have many frames or multiple windows. WebDriver supports moving between named windows using the "switchTo" method:

```
driver.SwitchTo().Window("windowName");
```

e) Explicit Waits - An explicit wait is code you define to wait for a certain condition to occur before proceeding further in the code. The worst case of this is Thread.sleep(), which sets the condition to an exact time period to wait. There are some convenience methods provided that help you write code that will wait only as long as required. WebDriverWait in combination with ExpectedCondition is one way this can be accomplished.

```
using (IWebDriver driver = new FirefoxDriver())
{
   driver.Url = "http://somedomain/url_that_delays_loading";
   WebDriverWait wait = new WebDriverWait(driver, TimeSpan.FromSeconds(10));
   IWebElement myDynamicElement = wait.Until<IWebElement>(d => d.FindElement(By.Id("someDynamicElement")));
}
```

f) Implicit Waits - An implicit wait is to tell WebDriver to poll the DOM for a certain amount of time when trying to find an element or elements if they are not immediately available. The default setting is 0. Once set, the implicit wait is set for the life of the WebDriver object instance.

```
using (IWebDriver driver = new FirefoxDriver())
{

driver.Manage().Timeouts().ImplicitlyWait(TimeSpan.FromSeconds(10));
   driver.Url = "http://somedomain/url_that_delays_loading";
   IWebElement myDynamicElement= driver.FindElement(By.Id("someDynamicElement"));
}
```

IV. USE CASE

In general, if we consider the upload operation then we have to consider a lot of other operations like - invoking the website to which the documents are uploaded. - Crawling the control to a specific place in the website - Clicking on a specific location/button and if that position is not visible then scrolling down using the control and if the location is found out - Clicking on that, pops up some kind of user interface which allows the user to add the file name - Clicking on the continue or finish button to validating the right file has entered. Finally clicking on upload button to complete the upload operation.

Following structure shows how the above operations be achieved.

```
IWebDriver driver = new InternetExplorerDriver();
driver.Navigate().GoToUrl("https://www.google.co.in/");
driver.Manage().Window.Maximize();
```

Giving time to open the explorer and load all the web elements is necessary which can be achieved using

```
driver.Manage().Timeouts().ImplicitlyWait(TimeSpan.FromSeconds(60));
System.Threading.Thread.Sleep(20000);
```

Navigate to the required location using the

```
var expander = driver.FindElements(By.ClassName("x-tree3-el"))[1];
var expander = driver.FindElement(By.LinkText(PL));
```

Clicking on the buttons/expander can be achieved using

International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 10
June 2016

```
action.Click(expander).Perform();
```

Various options are provided like DoubleClick, ClickAndHold, Drag AndDrop etc

We can even make use of LINQ to locate the web element using

```
Var next_link = driver.FindElements(By.Id("menuFileImportDocument"))
.FirstOrDefault(o => o.Text.Contains("Document..."));
```

To achieve the keyboard operations say ctrl + V or to send some input value, we make use of

```
var copy_link = driver.FindElement(By.Id("sami_release_date-input"));
copy_link.SendKeys(OpenQA.Selenium.Keys.Control + "a");
copy_link.SendKeys(OpenQA.Selenium.Keys.Control + "c");
```

To make use of the mouse operations say scroll-down and scroll-up, we make use of

```
_currentFrame=driver.SwitchTo();
_currentFrame.ActiveElement().SendKeys(OpenQA.Selenium.Keys.Page
Down);
```

Last element in an array of items which are captured using class name is taken out using

```
var lastSelectedproj = driver.FindElements(By.ClassName("grid-fixed"))
.LastOrDefault();
```

Similar operations can be achieved using selenium and even we can capture the screenshot automatically and can be sent as a mail.

V. CONCLUSION

A In this world of automation if there is similar work done frequently then it becomes a need to automate the process. The selenium webDriver provides an option to achieve the automation using the programming language like C#, java. Making use of the selenium the resource utilization be achieved effectively reducing the manual effort.

ACKNOWLEDGMENT

We would like to thank the publication who has reviewed this technical paper. We would also like to thank Sri Jayachamarajendra College of Engineering for providing this opportunity for carrying out this work. Lastly we extend our warm thanks to Dr. H C Vijayalakshmi, Head of CS&E for her enlightening guidance.

REFERENCES

[1] http://docs.seleniumhq.org/
[2] http://en.wikipedia.org/
[3] http://onlineseleniumtraining.com/selenium-web-driver/
[4] http://www.mythoughts.co.in/2012/06/handling-drag-and-drop-actions-using.html#.V1j4SDV96zd
[5] http://stackoverflow.com/
[6] https://msdn.microsoft.com/
[7] Vishawjyoti and Sachin Sharma, "Study And Analysis Of Automation Testing Techniques", Deptt of Computer Applications, Manav Rachna International University, Faridabad, Volume 3, No. 12, December 2012, JGRCS.
[8] Rahul joshi, "Analysis and Design of Selenium WebDriver Automation Testing Framework", Big Data, Cloud and Computing Challenges, Volume 50, 2015, Pages 341-346, ELSVIER