**SELENIUM**

**SELENIUM IDE**

Selenium IDE (Integrated Development Environment) is the simplest tool in the Selenium Suite. It is a browser add-on (both Google Chrome and Firefox has this browser add-on) that creates tests very quickly through its record-and-playback functionality. It is effortless to install and easy to learn.

Because of its simplicity, Selenium IDE should only be used as a prototyping tool, not an overall solution for developing and maintaining complex test suites.

Selenium IDE supports autocomplete mode when creating tests. This feature serves two purposes:

* It helps the tester to enter commands more quickly.
* It restricts the user from entering invalid commands.
* Commands used to write the test case are known as ‘Selenese’ commands

Screens for Selenium IDE are as mentioned below:

Graphical user interface, table

Description automatically generated

Graphical user interface, application

Description automatically generated

Let’s go ahead and download Selenium IDE add on and start doing exercise to perform some basic operations and execution- This will include following points:

* Installing the Selenium IDE
* Walkthrough of the complete Selenium IDE tool
* Building test case with record and play
* Some add on features/modification which can be done

**Difference between Assert & Verify:**

* Both of the commands will check whether the expected condition has been met or not.
* In case Assert fails: It will stop the further execution of any command provided in the test case.
* In case Verify fails: It will not stop the further execution of any command provided in the test case.

**List of all the commands:**   
https://www.selenium.dev/selenium-ide/docs/en/api/commands

**SELENIUM WEBDRIVER ARCHITECTURE**

**Graphical user interface, diagram, application

Description automatically generated**

**WORKING DEMO**

## **Demo**

In real time, you write a code in your UI (say Eclipse IDE) using any one of the supported Selenium client libraries (say Java).

Example:

|  |  |
| --- | --- |
| 1  2 | WebDriver driver  = new ChromeDriver();    driver.get(<https://www.google.com>); |

Once you are ready with your script, you will click on Run to execute the program. Based on the above statements, the Firefox browser will be launched and it will navigate to mentioned website

Once you click on ‘Run’, every statement in your script will be converted as a URL, with the help of JSON Wire Protocol over HTTP. The URL’s will be passed to the Browser Drivers. (In the above code, I have used FirefoxDriver). Here, in this case, the client library (Java) will convert the statements of the script into JSON format and further communicate with the FirefoxDriver. Every Browser Driver uses an HTTP server to receive HTTP requests. Once the URL reaches the Browser Driver, then it will pass that request to the real browser over HTTP. Once done, the commands in your [Selenium](https://www.edureka.co/blog/selenium-using-python/) script will be executed on the browser.

**WEBDRIVER INTERFACE**

**Diagram

Description automatically generated**

Following are the few points based on the above image.

* SearchContext is the super most interface in selenium, which is extended by another interface called WebDriver.
* All the abstract methods of SearchContext and WebDriver interfaces are implemented in RemoteWebDriver class.
* All the browser related classes such as FirefoxDriver, ChromeDriver etc., extends the RemoteWebdriver class.

Preferred Way of writing:  
**WebDriver driver = new ChromeDriver(); OR WebDriver driver = new FirefoxDriver(); OR WebDriver driver = new InternetExplorerDriver();**

Let’s see why can’t we use the following statement.

**WebDriver driver = new WebDriver();**

We cannot write our code like this because we cannot create Object of an Interface. WebDriver is an interface.

But we can use any of the following statements in our script

*FirefoxDriver driver = new FirefoxDriver();*  
*ChromeDriver driver = new ChromeDriver();  
InternetExplorerDriver driver = new InternetExplorerDriver();*

**Advantage of writing like: WebDriver driver = new ChromeDriver();**

If you define driver as a WebDriver, switching will be very easy. If we use this statement in our script then the WebDriver driver can implement any browser. Every browser driver class implements WebDriver interface and we can get all the methods. It helps you when you do testing on multiple browsers.

**WEBELEMENT INTERFACE**

**Diagram

Description automatically generated**

Now, let’s explore WebElement Interface:

* As explained WebElement represents HTML element.
* The interface also extends SearchContext, TakesScreenshot interfaces.
* In general, whatever we see in the HTML page is a WebElement whether it’s a link, search-text, button, Drop-Down,Web-Table etc. So, we can say that every action on a Web-Page while automating your Web Application will have to go through WebElement interface.

**MAVEN**

Maven is a build tool. It is a software project management tool that provides a new concept of project object model (POM). Maven increases reusability and takes care of most of build-related tasks. It helps in bypassing my steps like adding jars to the project library, building reports, executing Junits test cases, creating Jar etc.

**DOWNLOAD MAVEN AS PLUGIN IN ECLIPSE**

1. Click Help from top menu in Eclipse & Select Install New Software

2. Click on the ***Add***button on the newly opened window.

3. In the Name box, type ‘**Maven**‘ and in the Location box, type ‘***http://download.eclipse.org/technology/m2e/releases/***‘

4. A check-box will appear in the pop window, ***Check*** the check-box and click on the ***Next*** button.

5. Please wait for some time and let the window complete its processing. It will not take long but 2 or 3 minutes.

6. Keep the default settings and click on the ***Next*** button.

7. Accept the ‘*Terms and Conditions*‘ and move forward by clicking on the *Finish* button.

8. Wait while it finishes the installation.

9. Once the installation is finished, it will ask you to restart the Eclipse. Please click on ***Yes***, so that changes can be reflected properly.

**DOWNLOAD AND CONFIGURE MAVEN**

PRE-CONDITION- JAVA IS INSTALL IN THE SYSTEM

1. Go To <https://maven.apache.org/download.cgi>

2.  Extract it to some location.You can choose your own location.

3. Set up the ***Maven Environment Variable*** the same way we set up the Java Environment Variable above.

4. Write ‘***MAVEN\_HOME***‘ in the Variable name box then enter ‘***Your downloaded path of Maven***‘ Maven path in the Variable value box and click ***OK***.

5. Update the PATH variable just as JAVA and provide the path till the bin folder of mvn.

6. Open command prompt & enter ‘mvn -version’

7. You should be able to see the version of the mvn installed

**LAUNCHING BROWSER**

public class ChromeBrowserLaunchDemo {s

public static void main(String[] args) {

//Creating a driver object referencing WebDriver interface

WebDriver driver;

//Setting the webdriver.chrome.driver property to its executable's location

System.setProperty("webdriver.chrome.driver", "/lib/chromeDriver/chromedriver.exe");

//Instantiating driver object

driver = new ChromeDriver();

//Using get() method to **OPEN A URL**

driver.get("https://www.madridsoftwaretrainings.com/");  
  
 //Using getTitle() method to get the title of the WebPage  
 driver.getTitle();  
  
 //  
 driver.refresh();

//Closing the browser

driver.quit();

}

}

***https://www.eclipse.org/downloads***