Automation Testing

Automation testing is an application using automation tool with a programming language

Without any human intervention.

Need for Automation

- Time saving
- Automation supports regression testing
- Automation supports smoke testing
- Cost Saving(less resources)
- Testing improves accuracy (human errors can be minimized)

When to use Automation

- When the project has enough budget
- Using Long term Projects
- Test cases needs to executed frequently
- Number of test cases is more
- Feasibility of the application

Selenium

- ➢ Selenium is a free (open-source) automated testing tool used to validate web applications across different browsers and platforms.
- ➢ Can use multiple programming languages like Java, C#, Python, ruby etc to create Selenium Test Scripts
- ➢ Cross Browser Application- Can run tests across different browsers
- ➢ Supports Various Operating Systems and multiple web browsers (Safari, Edge, Internet Explorer)
- ➢ To generate reports, integrate with frameworks (TestNG, JUNIT, Cucumber)
- ➢ Integrate with CI/CD tools (Jenkins, azure devops)

Advantages

- Open Source
- Easy to setup
- Re-Usability
- Support cross browser testing
- Support multiple OS, multiple programming languages, and multiple browsers.

Disadvantages

- It is open source, so in case of issues there is no prompt vendor assistance.
- It supports web based applications only. Can't test desktop applications.
- Need programming skills to write the scripts.
- Need to depend external tools like Testing, Testng for the reporting.

Variants or Components of Selenium

Selenium Software is not just a single tool but a suite of software, each piece catering to different Selenium QA testing needs of an organization. Here is the list of tools.

Selenium IDE	**Selenium RC**	**Selenium Web driver**	**Selenium Grid**
*No need of Programming Language *Record and Playback tool *Only support in firefox	*Outdated version *This is the first automated web testing Tool that users to use a programming language they prefer RC can support the following programming Languages 1.Java 2.C# 3.PHP 4.Python 5.Perl 6.Ruby	*Updated version of Selenium RC *Multiple browser Support *Direct interaction with elements *Multiple languages Support	*Hub-node architecture (it allows us to run our test scripts across multiple machines or systems.) *Used for parallel exceution

Differences b/w Automation testing & Manual Testing

Automation Testing	Manual Testing
More accurate compared to manual testing and reliable.	Not accurate all the time.
Fast	Time consuming
Testcases are executed using automation tools	Testers execute test cases manually
Requires human intervention for each step	Minimal human intervention once scripts are setup

Architecture of Selenium

1. **Selenium 3:** uses JSON Wire Protocol for communication. JSON means JavaScript Object Notation. The activity of JSON Wire Protocol is to facilitate information between Client and Server through HTTP (Hypertext Transfer Protocol).
2. **Selenium 4:** uses WebDriver W3C protocol for direct communication between client and server. Selenium 4 does not have JSON Wire Protocol because here there is a direct transfer of information between the client and server.

How to add selenium Project in eclipse –Steps

1. create Project from file menu
2. Select maven project and click on create a new project
3. Create artifact ID ids then finish

How to add selenium dependencies in project

1. click on htttp://mvnrepository.com
2. Search selenium
3. Click on the first link then copy the dependencies code
4. Paste the dependency code to the pom.xml in our maven project

How to launch a browser?

```
WebDriver driver;

driver =new ChromeDriver();
```

WebDriver is an interface implements ChromeDriver() class by using object driver. We cannot create an object of the interface. The object that has been created for chrome driver cannot be used to run the scripts of firefox driver. Ie, we are creating the instance of webdriver interface and casting it to chrome driver.

Adv: easy to switch to another browsers

Browser Commands

Sl No	Commands	Definition	Example
1	driver.get(“url”)	url is the website address to load	driver.get("https://selenium.qabible.in/") ;
2	driver. close();	This method close the current window that the web driver is currently controlling	driver.close();
3	driver.quit()	Closes the whole browser set along with the browser windows & tabs &popups	driver.quit();
3	driver.manage().window ().maximize();	This method maximizes the current window that the web driver is currently controlling	driver.manage().window().maximize();

4	driver.getTitle();	To get page title in the Selenium webbrowser	String title=driver.getTitle();

5	driver.getCurrentUrl();	Command is used to retrieve the URL of the webpage the user is currently accessing	String currenturl=driver.getCurrentUrl();
			
			System.out.println(currenturl);

6	driver.getPageSource();	Command is used to retrieve the page source of the webpage the user is currently accessing	String pagesource=driver.getPageSource();
			
			System.*out*.println(pagesource);

7	driver.getWindowHandle();	This method helps to get the window handle of the current window	String handleid=driver.getWindowHandle();
			
			System.*out*.println(handleid);

Navigation Commands

1. driver.navigate().to("https://www.amazon.in/");

   // This method loads a new page in the existing browser window, takes a String as a parameter, and returns void.
2. driver.navigate().back();

   // This method enables the web browser to click on the back button in the existing browser window. It accepts no parameter and returns void.
3. driver.navigate().forward();

   // This method enables the web browser to click on the forward button in the existing browser window. It accepts no parameter and returns void.
4. driver.navigate().refresh();

   // This method refreshes the current web page in the existing web browser window. It accepts no parameter and returns void.

WebElements

Anything that is present on the web page is a WebElement such as text box, button, etc. WebElement represents an HTML element. It basically represents a DOM (Document Object Model) element and all the HTML documents are made up by these HTML elements.

Locators Of Selenium

Locators are the way to identify an HTML element on a webpage.

	Locators	Definition	Example
	id	Locates an element using the ID attribute.ID Selenium locators are unique for each element.it is the fastest and safest locator	driver.findElement(By.id("element ID"));
	name	Locates an element using the Name attribute	driver.findElement(By.name("element ID"));
	className	Locates an element using the class attribute	driver.findElement(By.className("element ID"));
	XPath	Locates an element using XPath query.	driver.findElement(By.xpath(<xpath>));
	Link Text	Locates a link using Link text	driver.findElement(By.linkText(<linktext>));

	partialLinkText	Locates a link using the link's partial text	driver.findElement(By.partialLinkText(<linktext));
	cssSelector	Locates an element using CSS selector	SYNTAX://css selctor= tagname[attribute='value'] driver.findElement(By.cssSelector("button[id='button-one']"));
	tagName	Locates an element using the HTML tag	driver.findElement(By.tagName(<htmlTagName));

BY XPath

XPath is the xml path of an element used to find the location of any element on a webpage using

Xml path expression.

There are Two types of XPath.

1) **Absolute XPath :**

- It is one of the easiest and direct ways to locate a web element.
- It selects an element path from the root node.
- It starts with a single forward slash '/'.
- The only disadvantage of using the Absolute path in selenium is that if any changes are made in the website or a document, then the whole XPath of that element changes ,and hence the previous path will not work,and the whole program will get failed.

Steps to take Absolute XPath.

a. Inspect an Element.

b. Right click on the code while getting inspect.

c. Click on copy

d. Click on Copy full Xpath.

2) Relative XPath

- Used to define an element path from the middle of the HTML DOM structure.
- It starts with double forward slash ’//’
- It is not necessary to define the path from the root level.

How to write XPath in different ways(Relative XPath)

1. **Using tagname and attribute**

   Syntax://tagName[@attribute=’value’]

   eg: //button[@id='button-one']

2. **Using Text**

   Syntax://tagname[text()='textvalue']

   Eg: //label[text()='Enter Message']

3. **Using starts-with**

   Syntax://tagname[starts-with(@attribute,’value’)]

   Eg: //button[starts-with(text(),'Show ')]

4. **Using OR/AND**

   Syntax: //tagname[@attribute1='value' and @attribute2='value2']

   Eg: //button[@id='button-one' and @type='button']

   Syntax: //tagname[@attribute1='value' or @attribute2='value2']

   Eg: //button[@id='button-one' or @id='button-one-electronics']

   Xpath axis methods :

   SYNTAX of axis method : //tagname[@attribute=’value’]//axesMethodName::tagname

5. **Using following**

   Syntax://tagname[@attribute='value']/following::tagname

   eg: //button[@id='button-one']//following::div[@class='card']

6. **Using Parent**

   Eg: //div[contains(text(),'Single Input Field')]//parent::div[@class='card']

7. **Using Child**

   Eg: //div[@class='card']//child::button[@id='button-one']

8. **Using following**

   Syntax://tagname[@attribute='value']/following::tagname

   eg: //button[@id='button-one']//following::div[@class='card']

9. **Using preceding**

   Eg: //button[@id='button-one']//preceding::div[@class='card']

10. **Using Ancestor**

    Syntax://tagname[@attribute='value']/ancestor::div

    eg: //button[@id='button-one']//ancestor::div"

11. **Using Descendant**

    Eg: //div[@class='card']//descendant::div"

Difference between XPath and cssSelector

XPath	**CSS Selector**
XPath stands for XML Path	CSS stands for cascading style sheet
XPath is used to find the element in the HTML DOM	CSS selector is used to find the element in the HTML DOM using style sheet language.
The success rate of finding an element using XPath is too high.	The success rate of finding an element using CSS Selector is less compare to XPath.

XPath is used where element has no other way of locating	Older browsers donot support all CSS features.
Slower in IE	Faster in all browsers
Can locate element by text	Cannot locate by element by text
Can locate parent elements	Locates elements only in forward direction.
Strats with "/" or "//"	Locator looks neat and clean
More flexible	Some CSS selectors will not work with all browsers.

<u>Web Element Commands in Selenium</u>

WebElement is a predefined interface.

How to locate a Web element : To inspect a web element in chrome is to simply

right- click on that particular element and select the inspect option.

<u>Web element commands</u>

Sl. No	Commands	Definition	Syntax & Example
1	sendKeys()	Use to enter text on a field	element.sendKeys("text"); Eg: msgbox.sendKeys("hello");
2	click()	Used for clicking an element	element.click(); Eg: getTotalButton.click();
3	getText()	It helps to retrieve the text for a specific web element. getText() method returns string as a result	String mymessage =message.getText(); System.out.println(mymessage);

4	clear()	If this element is a text entry element, this will clear the value	Element.clear();
			
			Eg: msgbox.clear();

5	getTagName();	This method gets the tagname of this element. This returns a string value	element.getTagName("tagname")
			
			Eg: entervaluea.getTagName();

6	getCssValue()	This is used to get the Css properties like color, font size etc. This returns a string value.(colour code in rgba format)	element.getCssValue();
			
			Eg:showmsg.getCssValue("background-color")

Difference between findElement() and findElements()

findElement()	**findElements()**
Command used to uniquely identify a single web element within the web page. Eg:WebElement elementName =driver.findElement(By.LocatorStrategy("LocatorValue"));	Command used to identify a list of web elements within the web page. Eg:List<WebElement> elementName = driver.findElements(By.LocatorStrategy("LocatorValue"));

Dropdown in Selenium

In *HTML,* the *dropdowns* are generally implemented either using the <select> tag or the <input> tag. To perform certain operations on the *dropdowns,* which are declared using the <select> HTML tag, *Selenium WebDrivers* provides a class called *"Select "* class.

Select in Selenium WebDriver

"Select " class is provided by the **"org.openqa.selenium.support.ui "** package of Selenium WebDriver. You can create an object of the Select class, by-passing the object of the **"WebElement"** class, which shows the object returned by the corresponding **locator** of the WebElement.

Syntax:

Select select = new Select(WebElement webelement);

Eg: *WebElement javadropdown=driver.findElement(By.xpath("//select[@id='dropdowm-menu-1']"));*

javadropdown.click();

How to select an option from drop-down menu?

WebDriver provides three ways to select an option from the drop-down menu.

1. selectByIndex - It is used to select an option based on its index, beginning with 0.

1. dropdown.selectByIndex(5);

2. selectByValue - It is used to select an option based on its 'value' attribute.

1. dropdown.selectByValue("sql");

3. selectByVisibleText - It is used to select an option based on the text over the option.

1. dropdown.selectByVisibleText("python");

Handling Checkbox

The checkbox is a GUI element that allows the user to make certain choices for the given options. Users may get a list of choices, and the checkbox records the choices made by the user. The checkbox allows users to select either single or multiple choices out of the given list.

Selecting a single value :

Eg: WebElement value1=driver.findElement(By.*xpath*(" //input[@value='option-1']"));

value1.click(); // This method selects the locator

Handling Radio buttons

A Radio Button is an *HTML* element, which allows the user to select only one of the given options.

Eg: WebElement radio=driver.findElement(By.*xpath*("//input[@value='blue']"));

radio.click();

Visibility Of WebElement

To check visibility of the WebElements. It is widely used in radio button, dropdown, checkbox etc. the output of the these methods are either true or false.

1. ***isSelected():*** *Checks whether the webelement is selected or not.*
2. ***isDisplayed()****: Checks whether the webelement is displayed on the web page or not.*
3. ***isEnabled()****: Checks whether the webelement is enabled or not*

Handling Dynamic WebTables

- To get a entire table data

```
WebElement fulltable=driver.findElement(By.xpath("//table[@id='dataTable']"));

System.out.println(fulltable.getText());
```

- To select particular row

```
WebElement tablerow=driver.findElement(By.xpath("//table[@id='dataTable']/tbody/tr[2]"));

System.out.println(tablerow.getText());
```

Tbody: tabledata includes

Actions Class

Actions class is a collection of individual Action that you want to perform. For e.g. we may want to perform a mouse click on an element.

There are a lot of methods in this class which can be categorized into two main categories:

- *Keyboard Events*
- *Mouse Events*

Actions class syntax:

```
Actions action = new Actions(driver);

action.moveToElement(element).click().perform();
```

Different Methods for performing Mouse Events:

1. click(): Clicks on the element.

2. doubleClick (): Double clicks on the element.

3. contextClick() : Performs a context-click (right-click) on the element.

   Eg: Actions action=**new** Actions(driver);

   action.contextClick(homeicon).build().perform();

4. dragAndDrop(source, target): Click-and-hold at the location of the source element, moves to the location of the target element

   eg: WebElement dragme=driver.findElement(By.*xpath*("//div[@id='draggable']"));

   WebElement drophere=driver.findElement(By.*id*("droppable"));

   Actions action=**new** Actions(driver);

   action.dragAndDrop(dragme, drophere).build().perform();

5. moveToElement(toElement): Moves the mouse to the middle of the element

   eg:
   WebElementprogressbarmenu=driver.findElement(By.*xpath*("//a[@id='progressbars']"));

   Actions action=**new** Actions(driver);

   action.moveToElement(progressbarmenu).build().perform();

Robot class

Robot class is a class which is used to perform the keyboard action in java.

Declare Robot

Robot robot = new Robot()

Methods:

r.keyPress()	To press any key
r.keyRelease()	To release key (whenever keypress() takes place keyRelease() should also mentioned.
KeyEvent : class (takes place inside robot method where it consists of all keyboard keys)	

Eg*: To open new tab (ctrl+T)*

```
Robot robot=new Robot(); //robot class for handling keyboard actions

robot.keyPress(KeyEvent.VK_CONTROL); // pressing ctrl key . VK:virtual key , keyevent:class

robot.keyPress(KeyEvent.VK_T);

robot.keyRelease(KeyEvent.VK_CONTROL); // to release a key

robot.keyRelease(KeyEvent.VK_T);
```