Selenium Introduction

Testing:-

Testing is the process of correctness and completeness of application or software with respect to customer requirements.

Types
1. Manual Testing
2. Automation Testing

Automation Testing :-

Testing an application or software with the help of an automation tool and executive test script called as automation testing.

We are comparing actual results and expected results so we can decide our test case is pass or fail.

Why we go for automation testing >> Disadvantages of Manual Testing

- Compatibility testing is difficult in manual testing because in which version browser and cross browser compatibility we are going to test so it is difficult in manual testing but it will be easy in automation testing
- As a tester we follow the testing life cycle so it will become lengthy and takes more time in manual testing
- More human resources are required in manual testing
- We need to perform regression testing so it will be time consuming in manual testing
- we will not able to achieve that much accuracy in manual testing

So that why we go for automation.

==

Advantages of Automation Testing

- Reusability of test scripts > We did not write the test cases multiple times so reusability is possible in this we are writing it once and used it many times,
- Project duration reduces > So everything depends on delivery we can reduce the project duration by using agile methodology with automation.
- Compatibility testing is easy > Compatibility testing is easy we can execute it parallelly in automation testing.
- Less human efforts
- Accuracy is more

History of Selenium

- Selenium Invented by Json Hugins and his colleagues in 2004
- Originally name was JSFT (JavaScript functional tester)
- Build as open-source browser-based integration test framework built by thought works at Chicago for time and expense keeping system
- Developed using JavaScript and HTML
- Designed to make test writing easy
- Ability to step through individual tests

==

Advantages of selenium >> Selenium:-more demandable

- It is Open source (free)
- It supports Multiple languages:- Java, python, c#, ruby etc...
- It supports Multiple OS:- Windows, MacOS, Linux etc...
- It supports Multiple browsers e.g. chrome, Firefox, safari, opera, edge etc...
- It helps is to do Parallel Testing
- It supports to integrate third party tools like TestNG, Cucumber, Git, Jenkins, GitHub, Maven etc.
- It has very large community

Disadvantages/Limitations of Selenium

- Selenium does not support automation testing for desktop applications.
- Captcha /barcode
- No reporting facility
- File upload

File upload >> We can achieve with AutoIT
No reporting facility >> We can achieve with TestNG

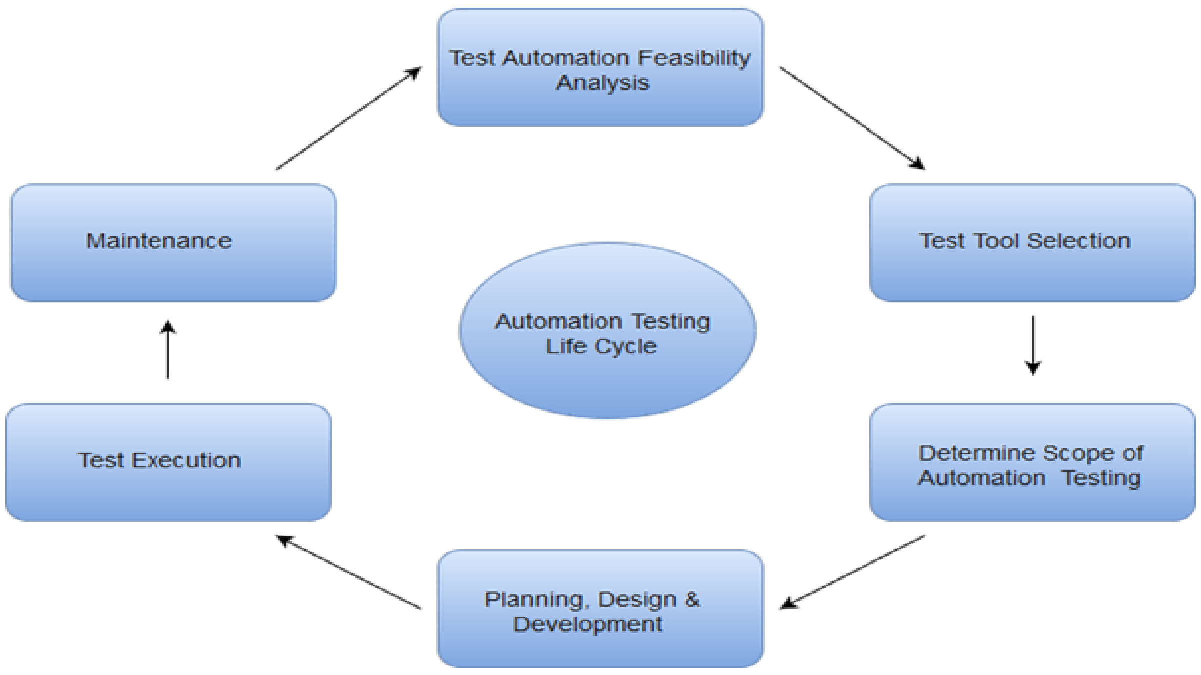
==

Tools :-

Test Management tool	JIRA
Databases Testing	Oracle, MySQL, MongoDB
Performance Testing	JMeter, LoadRunner
Security	OWASP

Website -- https://www.selenium.dev/

Automation Testing Life Cycle


Test Automation Feasibility Analysis
Test Tool Selection
Determine Scope of Automation Testing
Planning, Design & Development
Test Execution
Maintenance
Automation Testing Life Cycle

List of Automation tools

Selenium Components

1. Selenium IDE

- Selenium IDE (Integrated Development Environment) is a Firefox plugin.
- It is the simplest framework in the Selenium Suite.
- It allows us to record and playback the scripts.
- Even though we can create scripts using Selenium IDE, we need to use Selenium RC or Selenium WebDriver to write more advanced and robust test cases.

Operation System Support – Windows, Mac OS, Linux
Browser Support – Mozilla Firefox,

2. Selenium RC

- Selenium RC AKA Selenium 1. Selenium RC was the main Selenium project for a long time before the WebDriver merge brought up Selenium 2.
- Selenium 1 is still actively supported (in maintenance mode).
- It relies on JavaScript for automation. It supports Java, Javascript, Ruby, PHP, Python, Perl and C#.
- It supports almost every browser out there but doesn't support latest browsers

Operation System Support – Windows, Mac OS, Linux, Solaris
Browser Support – Mozilla Firefox, Internet Explorer, Google Chrome, Safari, Opera

3. Selenium WebDriver

- Selenium WebDriver AKA Selenium 2 is a browser automation framework that accepts commands and sends them to a browser.
- It is implemented through a browser-specific driver.
- It controls the browser by directly communicating with it.
- Selenium WebDriver supports Java, C#, PHP, Python, Perl, Ruby.

Operation System Support – Windows, Mac OS, Linux, Solaris
Browser Support – Mozilla Firefox, Internet Explorer, Google Chrome 12.0.712.0 and above, Safari, Opera 11.5 and above, Android, iOS etc.

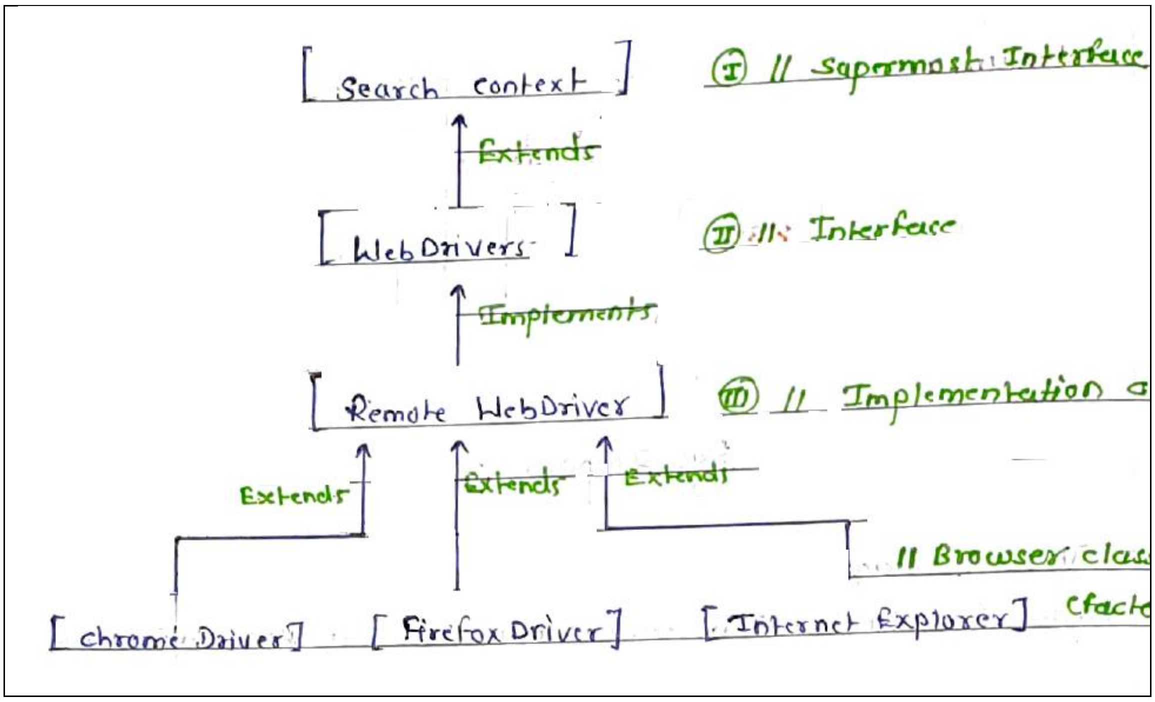
4. Selenium Grid

- Selenium Grid is a tool used together with Selenium RC to run tests on different machines against different browsers in parallel.

- That is, running multiple tests at the same time against different machines running different browsers and operating systems called as remote automations.

==

Selenium Architecture



1. Search context is the super most Interface which contains abstract method and inherited to WebDriver interface
2. WebDriver is an Interface which contain abstract method of search context and it's own abstract method
3. All the abstract methods are implemented in remote WebDriver class
4. Remote WebDriver is as class which implements all the abstract methods of both interfaces
5. Browser such as Firefox, chrome, safari, edge etc extends to remote WebDriver class
6. To run application in multiple browsers compatibility testing i.e. writhing test script in a single browser and run same script in different browser (WORA)
7. To achieve this we need to use upcasting in selenium

WebDriver driver =new ChromeDriver();

==

Installation of Selenium WebDriver

Selenium :- Its not a tool it's API which have combined logic of different classes

Selenium Installation

1. Visit https://www.selenium.dev/
2. click downloads
3. selenium server previous version
4. download 3.141.59 - Selenium - server - standalone
5. Copy from downloads and paste it in particular folder

How to add this Jra file to eclipse

1. New java project
2. Create package
3. Right click on project
4. Go to properties
5. Go to Java build path
6. Go to library
7. click on class path
8. add external jar files
9. choose our file – Selenium server standalone
10. Apply and close
11. you will see reference Library folder below JRE

How to set properties

1. Download driver file (chrome) - before download check version of browser

2. on eclipse set the path in System.setProperty method

```
System.setProperty("webdriver.chrome.driver","D://SeleniumFiles//ChromeDriver//chr
omedriver.exe");

//ChromeDriver driver =new ChromeDriver();

WebDriver driver =new ChromeDriver();  //upcasting
```

HTML

==

HTML stands for **Hyper Text Markup Language**, which is the most widely used language on Web to develop web pages.

Hypertext - link available on a webpage is called Hypertext.

Markup Language which means you use HTML to simply "mark-up" a text document with tags.

- ➢ Website - collection webpage
- ➢ Webpage - document - collection web element
- ➢ Web element - images, links text, video etc

HTML was created by Berners-Lee in late 1991.

HTML 2.0	1995
HTML 4.01	1999
HTML-5 version which is an extension to HTML 4.01	2012

Why to Learn HTML?

HTML was developed to define the structure of website like headings, paragraphs, lists etc with the help of different tags

1. To create Web site
2. To become a web designer
3. It helps to learn other languages

Applications of HTML

1. Web pages development
2. Internet Navigation
3. Responsive UI
4. Offline support
5. Game development

HTML Tags

- ➢ HTML is not case sensitive language.
- ➢ Tags are enclosed within angle braces **<Tag Name>**.

➢ Except few tags, most of the tags have their corresponding closing tags. For example, **<html>** has its closing tag **</html>** .

Hello World using HTML.

```
    <!DOCTYPE html>
    <html>
    <head>
           <title>This is document title</title>
    </head>
   <body>
           <h1>This is a heading</h1>
            <p>Hello World!</p>
   </body>
   </html>
```

Sr.No	Tag & Description
1. **<!DOCTYPE...>**	This tag defines the document type and HTML version.
2. <html>	This tag encloses the complete HTML document. It has header which is represented by <head>...</head> and document body which is represented by <body>...</body> tags.
3. <head>	This tag represents the document's header which can keep other HTML tags like <title>, <link> etc.
4. <title>	The <title> tag is used inside the <head> tag to mention the document title.
5. <body>	This tag represents the document's body which keeps other HTML tags like <h1>, <div>, <p> etc
6. <h1>	This tag represents the heading.
7. <p>	This tag represents a paragraph.

==

1. **HTML - Comments**
    a. Comment is a piece of code which is ignored by any web browser. It is a good practice to add comments into your HTML code , especially in complex documents
    b. HTML comments are placed in between <!-- ... --> tags.
2. **Heading Tags**
    a. Any document starts with a heading.
    b. You can use different sizes for your headings.

c. HTML also has six levels of headings, which use the elements <h1>, <h2>, <h3>, <h4>, <h5>, and <h6>

```
➢   <h1>This is heading 1</h1>
➢   <h2>This is heading 2</h2>
➢   <h3>This is heading 3</h3>
➢   <h4>This is heading 4</h4>
➢   <h5>This is heading 5</h5>
➢   <h6>This is heading 6</h6>
```

3. Paragraph Tag

a. The <p> tag offers a way to structure your text into different paragraphs.
b. opening <p> and a closing </p> tag

```
➢   <p>Here is a first paragraph of text.</p>
➢   <p>Here is a second paragraph of text.</p>
```

4. Line Break Tag

a. Whenever you use the <br /> element, anything following it starts from the next line

```
➢     <p>Hello<br />
➢     You delivered your assignment ontime.<br />
➢     Thanks<br />
➢     Mahnaz</p>
```

5. Centering Content Tag

a. You can use <center> tag to put any content in the center of the page or any table cell.

```
➢   <center>
➢     <p>This text is in the center.</p>
➢   </center>
```

6. Horizontal Lines

a. The <hr> tag creates a line from the current position

```
➢   <p>This is paragraph one and should be on top</p>
➢   <hr />
➢   <p>This is paragraph two and should be at bottom</p>
```

7. **Insert Image**
   a. You can insert any image in your web page by using <img> tag.

```
<img src = "Image URL" ... attributes-list/>
```

➢ Set Image Width/Height
   a. using width and height attributes

```
➢ <img src = "/html/images/test.png" alt = "Test Image" width = "150"
  height = "100"/>
```

➢ Set Image Border
   a. image will have a border around it,

```
➢ <img src = "/html/images/test.png" alt = "Test Image" border = "3"/>
```

8. **HTML - Text Links**
   a. A webpage can contain various links that take you directly to other pages and even specific parts of a given page. These links are known as hyperlinks.
   b. Hyperlinks allow visitors to navigate between Web sites by clicking on words, phrases, and images.
   c. A link is specified using HTML tag <a>. This tag is called anchor tag and anything between the opening <a> tag and the closing </a> tag becomes part of the link.

```
➢ <p>Click following link</p>
➢ <a href = "https://www.google.com/" >Navigate Point</a>
```

Notes :-

1. **void tag/elements** :- There are some HTML elements which don't need to be closed, such as <img.../>, <hr /> and <br /> elements. These are known as void elements.

==

HTML – Attributes

1. We used them so far in their simplest form, but most of the HTML tags can also have attributes, which are extra bits of information.
2. An attribute is used to define the characteristics of an HTML element and is placed inside the element's opening tag.
3. All attributes are made up of two parts − a **name** and a **value.**
    a. The **name** is the property you want to set.
        i. For example, **<p> align is attribute**, which use to indicate the alignment of paragraph on the page.
    b. The **value** is what you want the value of the property to be set and always put within quotations
        i. Three possible values of align attribute: **left, center** and **right**.

```
➢      <p align = "left">This is left aligned</p>
➢      <p align = "center">This is center aligned</p>
➢      <p align = "right">This is right aligned</p>
```

4. Core Attributes :- *Type, Id, Title, Class, Style, classname, maxlength*

HTML – Formatting

1. Bold Text
    a. Anything that appears within <b>...</b> element,

```
➢   <p>The following word uses a <b>bold</b> typeface.</p>
```

2. Italic Text
    a. Anything that appears within <i>...</i> element

```
➢  <p>The following word uses an <i>italicized</i> typeface.</p>
```

3. Underlined Text
    a. Anything that appears within <u>...</u> element,

```
➢  <p>The following word uses an <u>underlined</u> typeface.</p>
```

4. Grouping Content
    a. The <div> and <span> elements allow you to group together several elements to create sections or subsections of a page.

```
➢  <body>
➢       <div id = "menu" align = "middle" >
➢         <a href = "/index.htm">HOME</a> |
➢         <a href = "/about/contact_us.htm">CONTACT</a> |
➢         <a href = "/about/index.htm">ABOUT</a>
➢       </div>
➢
➢       <div id = "content" align = "left" >
➢         <h5>Content Articles</h5>
➢         <p>Actual content goes here.....</p>
➢       </div>
➢     </body>
```

HTML – Forms

- ➢ HTML Forms are required, when you want to collect some data from the site visitor.
  - For example, during user registration you would like to collect information such as name, email address, credit card, etc.
- ➢ A form will take input from the site visitor and then will post it to a back-end application.
- ➢ There are various form elements available like text fields, text area fields, drop-down menus, radio buttons, checkboxes, etc.

The HTML **<form>** tag is used to create an HTML form and it has following syntax –

```
➢         <form>
➢                    form elements like input, text area etc.
➢         </form>
```

There are different types of form controls that you can use to collect data using HTML form –

- Text Input Controls
- Checkboxes Controls
- Radio Box Controls
- Select Box Controls
- File Select boxes
- Clickable Buttons
- Submit and Reset Button

1. **Text Input Controls**
    a. Line of user input, such as search boxes or names. They are created using HTML <input> tag.

```
➢ First name: <input type = "text" name = "first_name" />
➢ <br>
➢  Last name: <input type = "text" name = "last_name" />
```

2. **Password input controls**
    a. They are also created using HTML <input>tag but type attribute is set to password.

```
➢ User ID : <input type = "text" name = "user_id" />
➢ <br>
➢ Password: <input type = "password" name = "password" />
```

3. **Checkbox Control**
    a. Checkboxes are used when more than one option is required to be selected.
    b. They are also created using HTML <input> tag but type attribute is set to checkbox.

```
➢ <input type = "checkbox" name = "maths" value = "on"> Maths
➢ <input type = "checkbox" name = "physics" value = "on"> Physics
```

4. **Radio Button Control**
    a. Radio buttons are used when out of many options, just one option is required to be selected.
    b. They are also created using HTML <input> tag but type attribute is set to radio.

```
➢ <input type = "radio" name = "subject" value = "maths"> Maths
➢ <input type = "radio" name = "subject" value = "physics"> Physics
```

5. **Select Box Control**
    a. A select box, also called drop down box which provides option to list down various options in the form of drop down list, from where a user can select one or more options.

```
➢ <select name = "dropdown">
➢ <option value = "Maths" selected>Maths</option>
➢ <option value = "Physics">Physics</option>
➢ </select>
```

6. **File Upload Box**
   a. If you want to allow a user to upload a file to your web site, you will need to use a file upload box, also known as a file select box.
   b. This is also created using the <input> element but type attribute is set to file.

```
➢ <input type = "file" name = "fileupload" accept = "image/*" />
```

7. **Button Controls**
   a. There are various ways in HTML to create clickable buttons.
   b. You can also create a clickable button using <input>tag by setting its type attribute to button.

```
➢ <input type = "submit" name = "submit" value = "Submit" />
➢ <input type = "reset" name = "reset"  value = "Reset" />
```

HTML - Tables

- The HTML tables allow web authors to arrange data like text, images, links, other tables, etc. into rows and columns of cells.
- The HTML tables are created using the <table> tag in which the <tr> tag is used to create table rows and <td> tag is used to create data cells. The elements under <td> are regular and left aligned by default.
- Table heading can be defined using <th> tag.

```
➢   <table border = "1">
➢       <tr>
➢         <th>Name</th>
➢         <th>Salary</th>
➢       </tr>
➢       <tr>
➢         <td>David Raman</td>
➢         <td>5000</td>
➢       </tr>
➢
➢       <tr>
➢         <td>Kumar Dawan</td>
➢         <td>7000</td>
➢       </tr>
```

WebDriver Methods/Commands

==

WebDriver Methods/Commands:-

WebDriver method or command is a interface used to perform actions on web element/Webpage.

Get Methods/Commands:-

1. Get Method- To launch URL in browser
   It is used to enter URL in a browser and it will wait for fully loaded the webpage.

```
Syntax :- driver.get("url");
Eg.  driver.get("https://www.amazon.com/");
```

2. Get Title Method - returns title of page
   It is used to get/return title of webpage

```
Syntax :- driver.getTitle();
Eg. driver.getTitle();
```

3. Get CurrentURL Method - Returns current URL of webpage
   It is used to get/return current URL of webpage

```
Syntax :- driver.getCurrentUrl();
Eg. driver.getCurrentUrl();
```

4. Get PageSource Method - Returns HTML code of webpage

```
Syntax :-driver.getPageSource();
Eg.driver.getPageSource();
```

Navigate Commands/Methods

Navigate is used to back,forward,refresh the browsers.

5. navigate().to()
   It is used to navigate from one webpage to other webpage and load new URL/webpage in the existing window of browser

```
Syntax :- driver.navigate().to("URL");
```

6. Forward command

This method is used to click on the forward button (arrow) of the browser window

```
Syntax :-driver.navigate().forward();
```

7. Back command

This method is used to click on back button(arrow) of the browser window

```
Syntax :- driver.navigate().back();
```

8. Refresh command

This method is used to refresh/reloads the current webpage in the browser

```
Syntax :- driver.navigate().refresh();
```

Browser Commands/Methods

9. Close - to close particular window

This method is used to close current browser window or selenium focused browser window

```
Syntax : - driver.close();
```

10. Quit - to cloe all windows

This method is used to quit/close all windows presnet in browser. It woll use to end selenium script

```
Syntax :- driver.quit();
```

11. Maximize - To maximize launched browser

This method is used to maximize the browser window

```
Syntax :-driver.manage().window().maximize();
```

12. Thread.sleep() -

It is used to stop loading of browser for some time by proving second

```
Syntax :- Thread.sleep();
```

Selenium Locators

===

Selenium Locators

- Locators are nothing but it is technique used to find web elements which are present on webpage.
- Each web element has it's certain position on webpage so selenium locators are used to find web element from webpage.
- Locators are methods of By class by which we can find element.
- Selenium Locators are one of the most powerful command
- It is the building block of selenium for automation script/testing.
- Hence it helps to locate GUI elements through which multiple user actions we can perform.
- The locators are one of the most important parameters for scripting base foundation and they may lead script failure
- Locating elements in Selenium WebDriver is performed with the help of findElement() and findElements() methods provided by WebDriver and WebElement class.

There are 8 locators which are used to find the web elements from the webpage

1. TagName
2. Id
3. Name
4. ClassName
5. CSS selector
6. Link Text
7. Partial LinkText
8. Xpath
   a. Absolute xpath
   b. Relative xpath

1. ID :-

- The most efficient way and preferred way to locate an element on a web page is By ID.
- ID will be the unique on web page which can be easily identified.
- IDs are the safest and fastest locator option
- It is like an Employee Number or Account which will be unique.
- Locates an element using the ID attribute

```
Syntax :- driver.findElement(By.id("id value"));
Eg. driver.findElement(By.id("search_query_top"));
```

2. Name

- Locating elements by name is similar to locating by ID except we use name as prefix
- When there is no Id to use, the next worth seeing if the desired element has a name attribute.
- But make sure there the name cannot be unique all the times.
- Locates an element using the Name attribute

```
Syntax :-driver.findElement(By.name("name_value"));
Eg . driver.findElement(By.name("submit_search"));
```

3. ClassName

- Locates an element using the Class attribute

```
Syatax :-driver.findElement(By.className("value"));
Eg . driver.findElement(By.className("submit_search"));
```

4. TagName

- Locates an element using the HTML tags
- Tag Name can be used with Group elements like , Select and check-boxes / dropdowns.

```
Syntax :-driver.findElement(By.tagName (<htmltagname>))
```

5. Link Text

- This type of locator applies only to hyperlink texts.
- Finding an element with link text is very simple. But make sure, there is only one unique link on the web page.

```
Syatax :-driver.findElement(By.linkText("Text"));
Eg.driver.findElement(By.linkText("Login"));
```

6. Partial Link Text

- Locates a link using the link's partial text if text is to long.

```
Syatax :-driver.findElement(By.partiallinkText("webpage_partial_value"));
Eg.driver.findElement(By.partiallinkText("password"));
```

XPath in Selenium

- XPath is a technique in Selenium that allows you to navigate the structure of a webpage's HTML and find web element.
- In automation, if the elements are not found by the general locators like id, class, name, etc. then XPath is used to find an element on the web page.
- XPath in Selenium is an XML path used for navigation through the HTML structure of the page.

Types of X-path

1. Absolute XPath
2. Relative XPath

Absolute XPath:

- It is technique of locators to find web elements which are present on web page
- It is the direct way to find the element
- To achieve absolute xpath we have to use single forward slash(/).
- Navigating from root of the parent to the immediate child it is nothing but absolute xpath.

  PARENT HTML >> CHILD

Absolute XPath:
/html/body/div[2]/div[1]/div/h4[1]/b/html[1]/body[1]/div[2]/div[1]/div[1]/h4[1]/b[1]

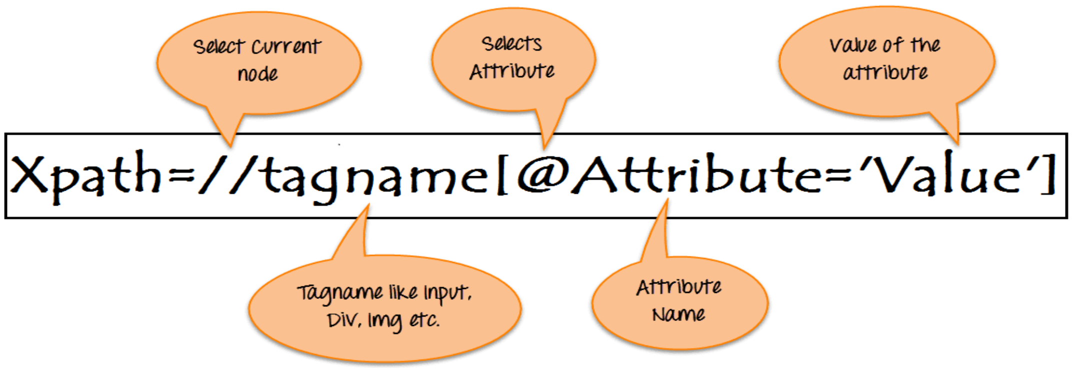
Drawbacks

- The disadvantage of the absolute XPath is that if there are any changes made in the path of the code then that XPath gets failed
- It is very difficult to locate web element as we need complete understanding of HTML code.
- Because of single slash / it provides less security.

Relative Xpath:

- Navigating root of the parent to any child is called relative xpath.
- To achieve relative xpath we have to use double forward slash(//).
- Relative Xpath starts from the middle of HTML DOM structure.
- It can search elements anywhere on the webpage, means no need to write a long xpath and you can start from the middle of HTML DOM structure.
- Relative Xpath is always preferred as it is not a complete path from the root element.

Syntax:-



//	: Select current node.
Tagname	: Tagname of the particular node
@	: Select attribute.
Attribute	: Attribute name of the node.
Value	: Value of the attribute.

Using XPath >> Handling complex & Dynamic elements in Selenium

1. Basic XPath:

- XPath expression select nodes or list of nodes on the basis of attributes like ID , Name, Classname, etc. from the HTML document

1. Xpath by tagname –
   Whenever we have only tagname then we will preferred this one

2. Xpath by attribute –
   Used different attributes and value to find xpath by using id, class, classname, name etc.

```
Syatax :-  //tagname[@attibute='value']
 Eg . //input[@id="search_query_top"]
      //input[@class="search_query form-control ac_input"]
      //input[@name='search_query']
```

3. Xpath by text function –
   If we want to find web element which is present along with link.

```
Syntax :- //tagname[text()='text_value']
Eg. //b[text()="Cart"]
```

2. **Xpath by using contains**

   If we want find web element by using partial text/value.

   1. Xpath by attribute –

```
Syntax :- //tagname[contains(@attribute,"value")] -
Eg. //a[contains(@id,"u_0_2")]
```

   2. Xpath by text function –

```
Syatax :-//tagname[contains(text(),"value")] - Syntax
Eg. //span[contains(text(),'Apple iPhone 12 Pro Max, 256GB, Pacific Blue - Unlocked (Renewed Premium)')]
```

==

Notes

Web Page –

Webpage is a collection of web elements like text box, buttons, radio buttons, checkbox, links, object, text etc

It contains live Element and dead Element

- Live Elements :- The Elements which have functionality
  - eg. click operation on button
- Dead Elements :- The Elements which don't have any functionality
  - eg.text

==

Questions :-

Dimension and Point Class

==

1. **Dimension Class :-**
   - While doing automation testing using selenium webdriver, there will be number of web elements and if we want to test stability and position of element is constant on webpage or not.
   - For that we need to change size of browser window and check it.
   - So we have dimension class which help us to resize browser window with specified co-ordinate

Dimension d=new Dimension(x,y);

Dimension	: class from webdriver along with x,y co-ordinate value
d	: reference variable

```
Syatax :-Dimension d=new Dimension(x,y);
         driver.manage().window().setSize(d);
   Eg.   Dimension d=new Dimension(200,300);
         driver.manage().window().setSize(d);
```

If we want to know size of browser need to use getSize() method so it will return size in the form co-ordinate,

driver.manage().window().getSize();

2. **Point Class :-**

   Whenever we want to change the location/position of browser window, we will use point class along with setposition() method.

Point p=new Point(x,y);

```
Syatax :- Point p=new Point(x,y);
         driver.manage().window().setposition(p);
   Eg.   Point p=new Point(50,100);
         driver.manage().window().setposition(p)
```

If we want to know position of browser need to use getPosition() method so it will return position in the form of co-ordinate.

driver.manage().window().getPosition();

Handling of ListBox

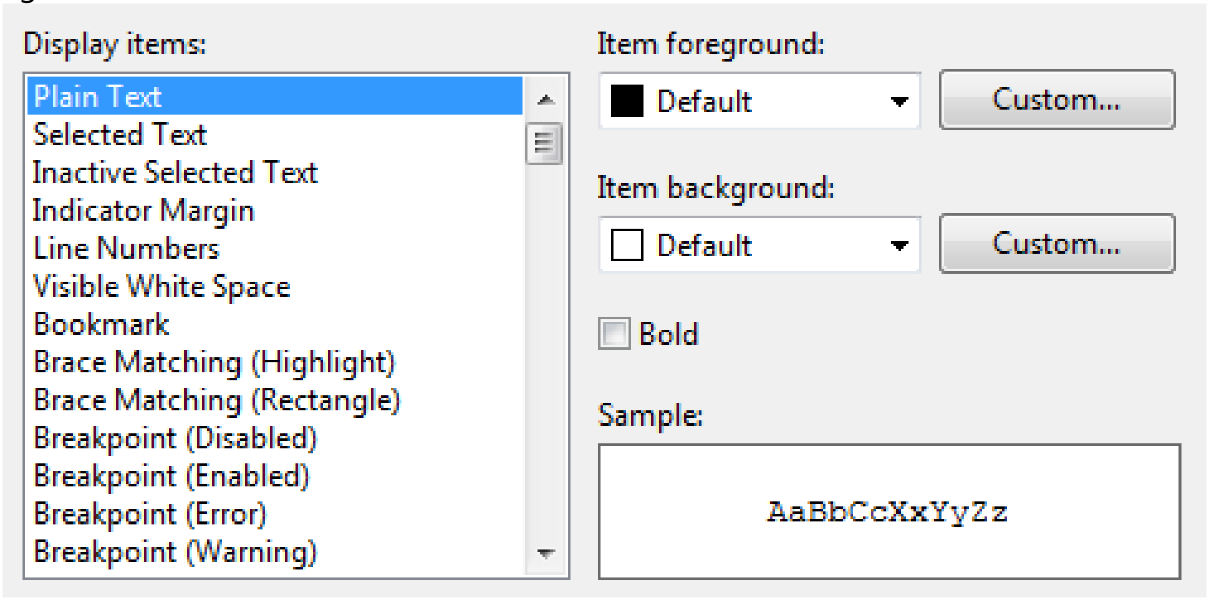
==

What is Listbox:

Listbox is web element which is available on webpage, has multiple options and user need select one of them.

Control element that allows the user to select one or more items from a list contained within a static, multiple line text box.

Eg.



Handling of ListBox:

- Whenever we want to handle listbox/dropdown we have to find that web element and store in the reference variable
- In list there will be multiple options are present and we need to select one of them for that create object of Select class along with ref variable

  Select s=new Select(web element);
- Now use select class methods to select option from listbox/dropdown for that we have 3 types of method

  - SelectByIndex(i);
  - SelectByValue("value");
  - SelectByVisibleText("text");

```
WebElement xyz= driver.findElement(By.xpath("Xpath"));

        Select s=new Select(xyz);
        s.selectByIndex(5);
        s.selectByValue("5");
        s.selectByVisibleText("1997");
```

Count of options:

We have methods getoptions() and size() which help us to find count options from particular list box

```
Select s=new Select();
List<WebElement> all_options = s.getOptions();
int count=all_options.size();
SOP(count);
```

Above statement gives us total count of options

Print all Options from listbox:

```
for(int i=0;i<count;i++)
{
String s=all_options.get(i).getText();
SOPln(s);
}
```

Screenshot :-

===

Why required screenshot?

- Whenever we are testing an application, if defects/bugs will find to us we required some kind of proof.
- Whenever script (test case) get pass or fail we need to show results and in those scenarios, screenshot behaves most important role as proof/evidence.
  - Every sprint minimum 100-150 screenshot required.

Tools

- Snipping tool
- Screenshot button from keyboard

How to capture screenshot in selenium?

1. To capture screenshot in selenium web-driver we need to type caste driver object along with takes screenshot interface.

   TakesScreenshot ts=((TakesScreenshot).driver);

2. Then we need to call function/method getScreenshot() and pass argument as OutputType.File

   File source=ts.getScreenshot(OutputType.FILE);

   which get screenshot of page but available in the internal memory let's stored in source file
3. So to move screenshot from source file to our destination, provide path of folder in File class from java.io

   File des=new File(path);

4. With the help of FileHandler.copy(source,des) we can easily get screenshot in specified location/folder

   FileHandler.copy(source,des)

Eg.

```
//Take screenshot
TakesScreenshot ts=((TakesScreenshot)driver);
File src = ts.getScreenshotAs(OutputType.FILE);
File des=new
File("C:\\Users\\232338\\OneDrive\\Desktop\\SeptEvening\\facebook.png");
FileHandler.copy(src, des);
```

Utility Class - util class >> Code re-usability

JavaScriptExecutor in Selenium WebDriver

==

What is JavaScriptExecutor?

- JavaScriptExecutor is an Interface that helps to execute JavaScript through Selenium Webdriver.
- JavaScriptExecutor provides two methods "executescript" & "executeAsyncScript" to run javascript on the selected window or current page.

Why do we need JavaScriptExecutor?

1. For Scrolling purpose
    - In Selenium Webdriver, whenever we are performing operation on web element which present on webpage, it may be available at any position so we need to scroll to that element, and JavaScript Executor helps us to scroll up and down
2. For locating web element
    - In Selenium Webdriver, locators like XPath, CSS, etc. are used to identify and perform operations on a web page.
    - In case, these locators do not work you can use JavaScriptExecutor. We can use JavaScriptExecutor to perform an desired operation on a web element.
    - Selenium supports javaScriptExecutor. There is no need for an extra plugin or add-on. We just need to import (org.openqa.selenium.JavascriptExecutor) to use JavaScriptExecutor.

How to scroll up or down>

1. To scroll up or down in Selenium WebDriver we use JavaScriptExecutor interface.

   JavascriptExecutor js = (JavascriptExecutor)driver;

2. It help us to use executeScript method and provide argument like

   js.executeScript("window.scrollBy(co-ordinate)");

Eg.

```
//Creating the JavascriptExecutor interface object by Type casting
JavascriptExecutor js = (JavascriptExecutor)driver;
 //Vertical scroll down by 600  pixels
 js.executeScript("window.scrollBy(0,600)");
```

1. What is getLocation()?

Iframe

==

Displaying webpage as part of another webpage called as iFrame.
There are some webpages which have another webpage so while automation sometimes we may face no such element found exception because may be there is possibility of having frame/iframe

iFrame in Selenium Webdriver

- iFrame in Selenium Webdriver is a web page which is embedded in another web page
- or an HTML document embedded inside another HTML document.
- The iframe is used to add content from other sources like an advertisement into a web page.
- The iframe is defined with the <iframe> tag.

How to identify the iframe:

- We cannot detect the frames by just seeing the page or by inspecting web element or with other plugin.
- Right click on the element, if you find the option like 'This Frame' then it is an iframe.
- Right click on the page and click 'View Page Source' and Search with the 'iframe', if you can find any tag name with the 'iframe' then we can say the page consist

How to switch over the elements in iframes

We can switch over the elements and handle frames in Selenium using 3 ways.

- By Index
- By Name or Id
- By Web Element

1. Switch to the frame by index:

Index is one of the attributes for frame handling in Selenium through which we can switch to it.

driver.switchTo().frame(0);
driver.switchTo().frame(1);

2. Switch to the frame by Name or ID:

Name and ID are attributes for frame handling in Selenium through which we can switch to the iframe.

driver.switchTo().frame("iframe1");
driver.switchTo().frame("id of the element");

3. **Switch to the frame by Web Element:**

   We can switch to the iframe using web element .

   driver.switchTo().frame(WebElement);

Identify total number of iframes:

int size = driver.findElements(By.tagName("iframe")).size();

How to switch back to the Main Page

- To move back to the parent frame, we can use

  driver.switchTo().parentFrame();

- To get back to the main (or most parent) page, we can use

  driver.switchTo().defaultContent();

Check for nested Iframe

==

Synchronisation/Selenium Waits:-

==

Matching selenium script speed with browser/application speed called as Synchronisation

So to do this we have waits in selenium.

Types of waits

1. Implicit wait
2. Explicit wait
3. Fluent wait

Why we need waits?

- Most of the webpage consist of ajax element when a page is loaded by browser, element which we want to interact may load at different time.
- If we are not able to find element selenium throws exception **ElementNotVisibleException**" exception, using waits we can avoid this kind problem/issue.

Implicit wait

- It is used when we know exactly how much time will take to load page.
- Webdriver will wait for certain amount of time and if not found that web element it will throws an exception like nosuchelementexcpetion.

Implicit wait = waiting time

Syntax -

```
driver.manage().timeout.implicitWait(Duration.ofSecond(Time));
Time – waiting time
```

Notes:

- The implicit wait will apply globally to each web element from webpage.
- We know exect time and which will be provided from deveplors
- Here we are using compile time polymorphism

Explicit wait

- It is used when we don't know exact time to require load web element from webpage

Explicit wait :Condition + specific time

- Explicit wait is used to tell webdriver to wait for certain condion or maximum time exceeds before throwing any exception like Element not found or element not visible exception
- It gives better options than implicit wait as it waits for dynamically loaded Ajax elements.
- It can be applied only on specific web element

Syntax:

```
WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(20));

wait.until(ExpectedConditions.elementToBeClickable(By.xpath("")));
```

If we provide 20 sec waiting time but web element find within 5 sec then it will not wait for 20 sec and moves to next step.

The following are the Expected Conditions that can be used in Selenium Explicit Wait

1. *alertIsPresent()*
2. *elementSelectionStateToBe()*
3. *elementToBeClickable()*
4. *elementToBeSelected()*
5. *frameToBeAvaliableAndSwitchToIt()*
6. *invisibilityOfTheElementLocated()*
7. *invisibilityOfElementWithText()*
8. *presenceOfAllElementsLocatedBy()*
9. *presenceOfElementLocated()*
10. *textToBePresentInElement()*
11. *textToBePresentInElementLocated()*
12. *textToBePresentInElementValue()*
13. *titleIs()*
14. *titleContains()*
15. *visibilityOf()*
16. *visibilityOfAllElements()*
17. *visibilityOfAllElementsLocatedBy()*
18. *visibilityOfElementLocated()*

Fluent Wait

- In Fluent wait we can give condtions and we can give specific waiting timing to occur that condion part third party parameter is frequency

  Fluent wait : Specific waiting time +confition +frequency.

- The Fluent wait is used to tell webdriver for condition as well as frequency with which we want to check condition before throughing **ElementNotVisibleException**

Frequency

- Setting up repeat cycle with time frame to verify condition

Let us consider a scenario where we want to find webelement which have different interval showing on webpage

In that case we declare explicit wait so it will wait till specified time begore throws expection and fluent wait is ideal to wait to use this try to find webelement at different frequency upto our time

```
Wait<WebDriver> fluentWait = new FluentWait<WebDriver>(driver)
                              .withTimeout(Duration.ofSeconds(10))
                              .pollingEvery(Duration.ofSeconds(1))
                              .ignoring(NoSuchElementException.class);
fluentWait.until(ExpectedConditions.elementToBeClickable(By.xpath("")));
```

Polling frequency - can change as per need

Ignore Exception - in case element is not found, can ignore any exception like ‘NoSuchElement’ exception etc.

===

Implicit Wait	**Explicit Wait**
Implicit Wait time is applied to all the elements in the script	Explicit Wait time is applied only to those elements which are intended by us
In Implicit Wait, we need **not** specify “ExpectedConditions” on the element to be located	In Explicit Wait, we need to specify “ExpectedConditions” on the element to be located
It is recommended to use when the elements are located with the time frame specified in Selenium implicit wait	It is recommended to use when the elements are taking long time to load and also for verifying the property of the element like(visibilityOfElementLocated, elementToBeClickable,elementToBeSelected)

==

Selenium Alert & Popup Window Handling

==

Popup :-

Small window which display on my webpage with some information when we perform operation on web element.

Types of popup -

Selenium support

1. Alert popup
2. window popup

Selenium don't support >> Need to use third party plugins like AutoIT, ATIOS

3. file upload
4. file download
5. authentication popup

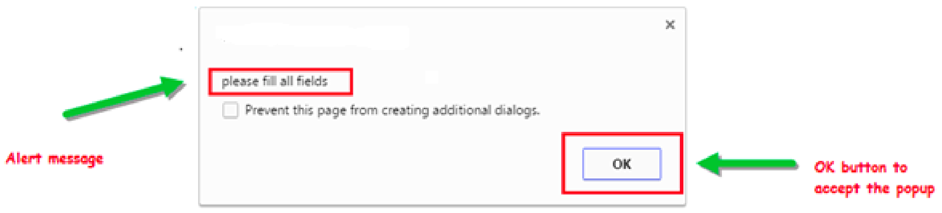
- What is Alert?
- How to handle Alert in Selenium WebDriver
- How to handle Selenium Popup window using Webdriver

What is Alert?

- An **Alert** is a small message box which appears on screen to give the user some information or notification.
- It notifies the user with some specific information or error, asks for permission to perform certain tasks and it also provides warning messages as well.
- We cannot drag and drop and contains symbol or ?
- To handle this popup we need to change focus of selenium from main page to alert window popup

1. Simple Alert

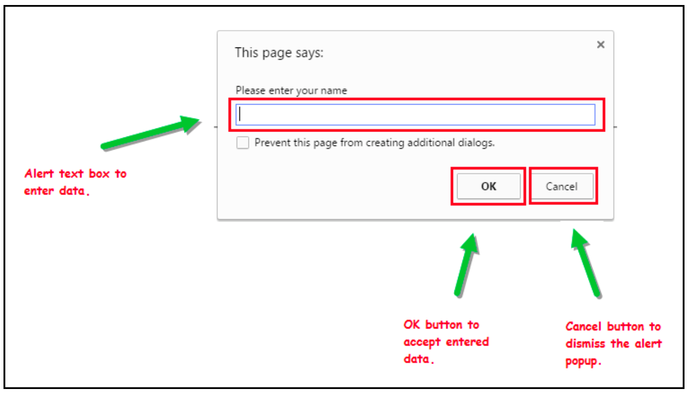
The simple alert class in Selenium displays some information or warning on the screen.


please fill all fields

Prevent this page from creating additional dialogs.

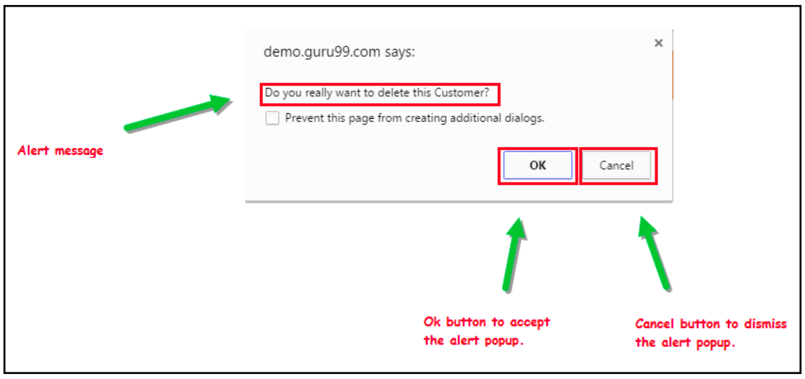
2. **Prompt Alert.**

This Prompt Alert asks some input from the user and Selenium webdriver can enter the text using sendkeys(" input…. ").


This page says:
Please enter your name
Prevent this page from creating additional dialogs.
OK
Cancel
Alert text box to enter data.
OK button to accept entered data.
Cancel button to dismiss the alert popup.

3. **Confirmation Alert.**

This confirmation alert asks permission to do some type of operation.


demo.guru99.com says:
Do you really want to delete this Customer?
Prevent this page from creating additional dialogs.
OK
Cancel
Alert message
Ok button to accept the alert popup.
Cancel button to dismiss the alert popup.

How to handle Alert in Selenium WebDriver

Alert is an interface which provides the few methods and help us to perform operation on popup.

1. **void dismiss()** // To click on the 'Cancel' button of the alert.
   driver.switchTo().alert().dismiss();

2. **void accept()** // To click on the 'OK' button of the alert.
   driver.switchTo().alert().accept();

3. **String getText()** // To capture the alert message.
   driver.switchTo().alert().getText();

4. **void sendKeys(String stringToSend)** // To send some data to alert box.
   driver.switchTo().alert().sendKeys("Text");

How to handle Selenium Pop-up window using Webdriver

What is a window in Selenium?

- A window in any browser is the main webpage on which the user is landed after hitting a link/URL./WebElement.
- Such a window in Selenium is referred to as the parent window also known as the main window

- In automation, when we have multiple windows in any web application, the activity may need to switch control among several windows from one to other in order to complete the operation.
- After completion of the operation, it has to return to the main window i.e. parent window in Selenium.

How do we identify parent window and child windows?

- When a user hits a URL, a webpage opens. This main page is the parent window i.e the main window on which the user has currently landed
- All the windows which will open inside your main window will be termed as child windows.

Methods to handle windows

- To handle window popup we need to change focus of selenium from main page to window popup for that we required ID of window.

  driver.switchTo().window(ID);

- we need to use following method

1. **Driver.getWindowHandle();**

   When the site opens, we need to handle the main window by *driver.getWindowHandle()*. This will handle the current window that uniquely identifies it within this driver instance. Its return type is String.

2. **Driver.getWindowHandles();**

   To handle all opened windows by web driver, we can use *driver.getWindowHandles()*, and then we can switch window from one window to another in a web application. Its return type is Iterator<String>.

===