★ What is Automation

Testing an application or software with the help of any automation tool and executing the test cases is called the Automation.

★ Why we go for an Automation? (disadvantages of manual)

We are going for Automation because there some disadvantage of manual testing that's why go for automation.

1) Compatibility testing is very 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.
2) As we follow the Testing life cycle so it will become lengthy and takes more time in manual testing.
3) More Human resources are required in manual testing.
4) We need to perform Regression testing (means we need to execute the test cases multiple times) so it will time consuming in manual testing.
5) We will not able to achieve that much accuracy in manual testing.

so that why we go for an automation.

★ Advantages of Automation testing

1) We did not write the test cases multiple times so Reusability is possible in this we are writting it once and used it many times.

* Selenium Webdriver tool does not support "captcha".

PAGE No.
DATE / /

(2)

4) so Everything is depends on delivary we can reduce the project duration by using agile methodology with Automation.

5) Compatibillity testing is easy we can execute it parallely in automation testing.

6) Here less Human efforts are required.

8) We can adopt any new technology at any time during automation.

* Disadvantages of Automation :-

1) for automation we required skilled Labour.

2) for Automation more money investment is required.

3) Knowledgeble Resources are required for automation.

4) We need to do maintainance for Automation tool.

* Disadvantages of selenium :

1) In selenium standalone application like calculator, Net protector we can not able to browse it.

2) selenium does not support windows based application so for that we have third party application with in selenium.

3) Uploading and downloading is supposed to possible in selenium but nowdays in advance version it somehow possible.

PAGE No
DATE 30 August 2020

* History of selenium :-

1) selenium Invented by Jason R. Huggins and his colleagues.
2) original name was "Java script functional tester" (JSFT)
3) Built as open source browser based integration test framework built by thoughtworks for time and expense keeping system.
4) Developed using Javascript and HTML.
5) Designed to make test writing easy.
6) ability to step through individual tests.
7) cross browser - IE 6/7, Firefox 8+, opera, Safari 2.0+

* What is selenium :-

1) Is a portable Javascript framework that runs in your web browser.
2) It is used for testing web applications.
3) Works anywhere Javascript is supported.
4) The test can be written as HTML tables or coded in number of popular programming languages like Java, python, Ruby
5) Selenium is available on windows, Linux and macintosh.

* Selenium components :-

Selenium is composed of three major tools. Each one has a specific role in aiding the development of web application test automation.

1) selenium - IDE
2) selenium - RC (Remote control)
3) selenium Grid

DATE / /

1) Selenium - IDE :-

- selenium - IDE is the integrated development Enviornment for building selenium test cases. It operate as a firefox add-on and provides an easy-to-use interface for developing and running individual test cases or entire test suits.

2) Selenium - Rc :-

- selenium - Rc allows the test automation developer to use a programming language for maximum flexibility and extensibility in developing test logic.
- It provides support for language : HTML, Java, C#, perl, PHP, Python and Ruby.

3) Selenium Grid :-

- selenium grid allows the selenium - Rc solution to scale for large test suits or test suits that must be run in multiple enviornment.
- These test suits can be run on multiple operating systems and browsers in parallel.

(5)

* Selenium :-

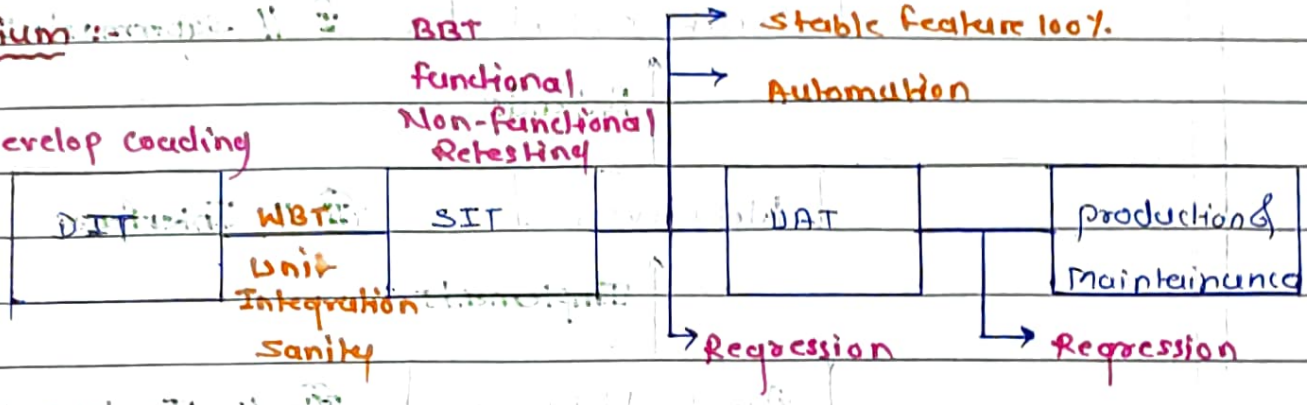


fig:- Diagram

- This is diagrame where we know that after SIT whenever our build is stable then will move towards Automation like we have 100% stable feature.

* In the selenium Internet Explorer not supported to icici Bank application so we are not using it.

- We are used to create an object with upcasting concept to access the webdriver in selenium.

Syntax:-

```
WebDriver driver = new ChromeDriver();
                          ↓
                       class()
```

* Selenium Architecture :-

In selenium we have selenium Architecture to follows to the access webdriver class, methods in our code directly. So they are as follows:

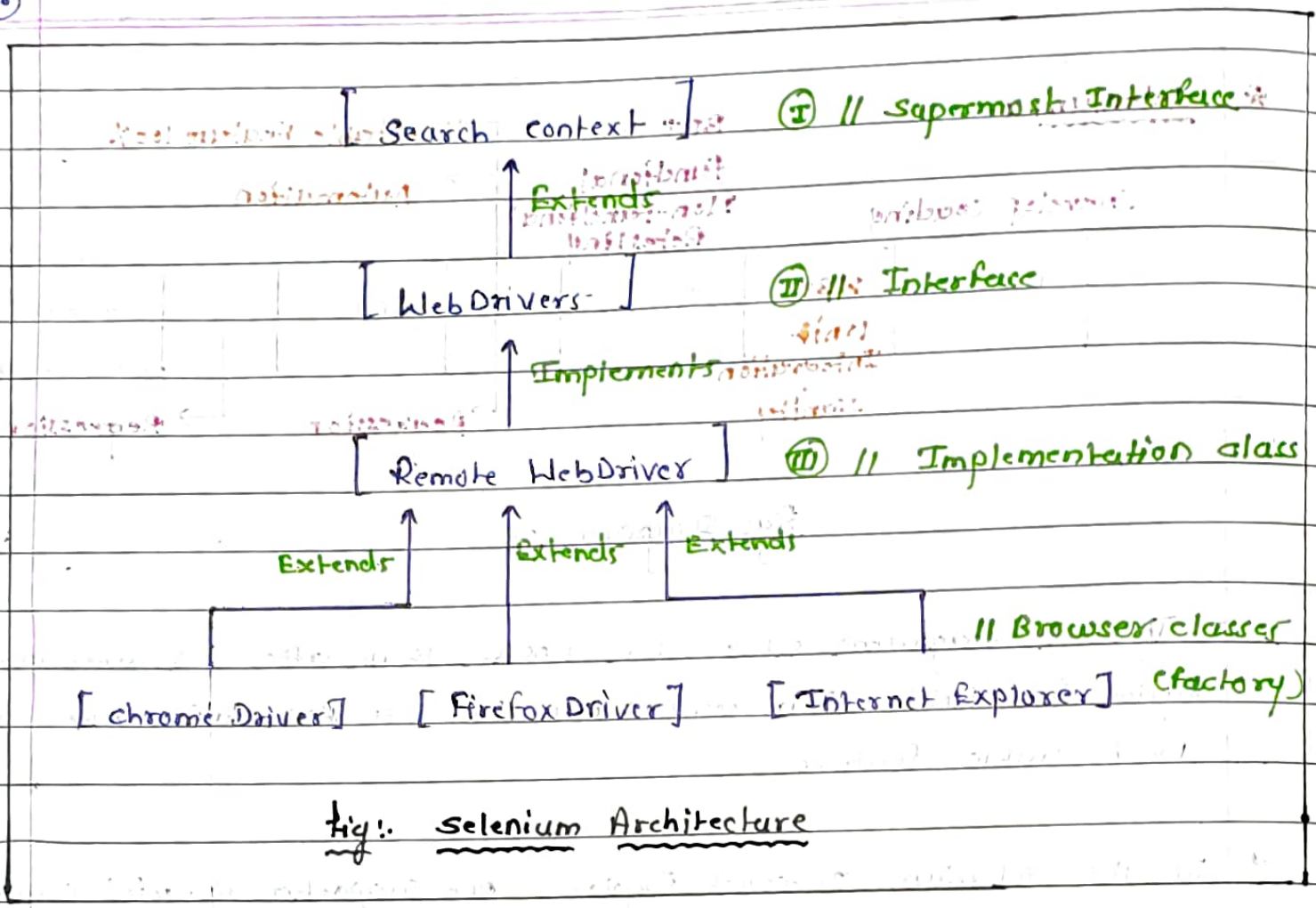


fig:- Selenium Architecture

Explanation :-

1) Search Context is the supermost 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 method are implemented in a remote webdriver class.
4) Remote webdriver is a class which implements all the abstract method of both the interfaces.
5) Browser such as firefox, Internet Explorer, Safari extends to remote webdriver class.
6) To Run application in multiple p browser [compatibility] testing i.e writing test script in a single browser and run the same script in different browser [WORA].

P) To achieve this we need to use "Upcasting" in selenium
" WebDriver driver = new chromeDriver ();"

* Automation Testing :-

Testing an application features with the help of Automation tool and executing test script is called " Automation Testing".

* Disadvantages of manual Testing :-

1) Compatibility testing is difficult.
2) Test cycle duration will be increase
3) more Human efforts are required
4) Regression testing is time consuming
5) Accuracy is less

* Advantages of Automation testing :-

1) Reusability of Test script
2) project duration will be reduce
3) Compatibility testing is easy
4) Less human efforts are required
5) we can adopt latest technology into a project.

* Advantages of selenium :-

1) It is open source.
2) multi language supportable i.e Java, C#, Ruby, Python, Perl, etc
3) cross browser testing is possible

* latest Version of selenium webDriver is - 3.141.59

(9) * latest website of selenium - http://selenium.dev

* Disadvantages of selenium :-

1) Does not support captcha.
2) we cannot automate standalone - application like windows based Application. ex:- wps office, calculator etc.
3) selenium will not support file upload.

* Installation of selenium :-

Search → official website of selenium

↓

Selenium we have to click on first link →

Selenium webdriver click on → Download link

↓

Latest version 3.141.59 click on this link

↓

Download

↓

Copy this file and paste on other drive

* Browser supported in selenium :-

1) Firefox
2) Internet Explorer
3) Safari
4) opera
5) chrome
6) Edge

* tool - consist of multiple jars.
* we can use class - by creating an object

PAGE No.:
DATE: / /

(3)

* operating systems supported in selenium:-

1) Microsoft windows
2) mac os
3) Linux
4) source code

* - "System" is a class.
- "setproperty" is a method
- "webDriver.chrome.driver" is a set by selenium people we need to import the webDriver and chromeDriver.

* How to get webdriver files in eclipse (Installation of Eclipse)

open the eclipse
↓
Create a new Java project
↓
Create a package (selenium Browser)
↓
Creak a class (selenium.chromeDriver)
↓
Right-click on Package
↓
go to "Build path"
↓
the "Configure build path"
↓
click on it " we get pop up window"

(10)

- whenever we add external jars of Selenium in eclipse then on editor will see three folder, ① SRC, ② JRE libraries ★ ③ Reference libraries

properties: for (project name) → comes on screen

↓

go into the libraries → just click on "classpath"

↓

click on "add external jars"

↓

New window of "Jarselection" comes

↓

select the "selenium-webdriver" downloaded file → select → ok.

Q. where we used oop's concept in selenium?

Ans: In Selenium webdriver, in which interfaces and some abstract methods we have in it so there we use oop's concept in selenium.

Q. How will use the class in Java?

Ans: By creating an object of that class we can able to use the class in Java.

Q. which browsers support in selenium?

Ans: firefox, IE (Internet Explorer), Safari, opera, Edge are the browsers support in selenium.

PAGE No. :
DATE : 1 August 2020

* **Installation of chrome :-**

Before Installation of chrome driver → please check our machine chrome version

↓

go to : → Help → about → setting

↓

then go on Google → selenium → download → scroll down → Browsers → chrome → documentation

↓

click on latest version of chrome driver

↓

click for 32 Windows (according specification)

↓

show in folder → extract that zip file to folder

* **How to copy path :-**

- If you want to copy the path so go on that particular folder simple one click press shift and do right click → click on → copy as path → past whereever we want.

* If you found '\' single slash in url then provide '\\' double slash to complete url. because sometime because of this '\' we got error in programe. so resolve it we need to provide '\\'.

* Before starting to write code or after creating object of webdriver you need to import webdriver and chromie property.

PAGE No.
DATE : 2 September 2020

Q. Write a first programe of selenium to start the "Url".

```
package aStartProgram;
    public class FirstSeleniumProgram {
    public static void main (String [] args) {

System. setProperty ("Webdriver.chrome.driver", "path");
    WebDriver driver = new chromeDriver ();
                    // path :- folder path where we save chrome.
                            driver downloaded file.
    driver. get ("https://www.facebook.com");
}
}
```

* Webdriver Methods :-

1) driver.get ("Url"); :- To get the web page of that particular Url.

2) driver.navigate().to ("particular page Url"); :- To navigate the particular page of web application.

3) driver.close (); :- It is used for closing Browser.

4) driver.navigate().back(); :- It is used for going backword in the browser.

5) driver.navigate().forward(); :- It is used for going forword in history.

6) driver.Navigate().refresh(); :- It is used to Reload a web page.

7) driver.quite(); :- It is used to closing all the tabs and browsers and windows associated with the drivers.

8) String title = driver.getTitle(); :- It is used to retrive the title of the current page.

9) driver.getCurrentUrl(); :- it is used to get the current url of the current web page.

10) Thread.sleep(); :- we are providing some time in seconds to hold the screen / web page for some time

Eg: int x = 2000;

Q. If we want output at console? then what will do?

Ans: If we want output at console then will pass the printing statement for getting output at console.

Q. which will be the another way to fetch url instead of get() method?

Ans: for getting the Url instead of get() method we have navigate().to(""); method to fetch the Url.

Q. Difference between quite() and close() methods?

Ans: The quite() and close() methods are used to close browser windows.

1) close() is used to close a single window

2) quite() is used to close all windows associated with the browser

PAGE No.
DATE 3 sept 2020

* How to get a web page for write Html code?

By Right- clicking → open your Notepade in our machine

↓

Give the Name → openit.

↓

go to file → file (save as) → Name :- fileName.html → save

↓

Now your Notepade convert into web page.

↓

By double clicking → yuget web page at chrome.

↓

Start writing Html code.

Q. What is difference between get() and Navigate.to() method?

Ans. get() :- It is used to get the particular url and we have to wait till page load.

Navigate.to() :- It is used to Navigate the particular url and does not wait for page load.

Q. Write a code for Html common web elements

<html>

<head>

<title> Velocity Web page </title>

</head>

<body>

<h1> Velocity Corporate Training center </h1>

PAGE No. :
DATE : 4 Sept 2020

Username < input type = 'text' id = 'Aqw12a' name = 'UN' class = 'username class' > < /br>

password < input type = 'password' > < /br>

Email Id < input type = 'text' id = 'Aq1234' nam = 'Email Id' > < /br>

male < input type = 'radio' id = '136AqB' name = 'radiobutton' >

female < input type = 'radio' id = 'APq12' > < /br>

I agree terms and conditions < input type = 'checkbox' >

< input type = 'button' value = 'signup' id = 'a1234' >

select country

< select >

< option > India < /option >

< option > US < /option >

< option > UK < /option >

< option > Canada < /option >

< / select >

< table >

< table border = '1' >

< tr >

< th > Sr.No < /th >

< th > Language < /th >

< th > Release year < /th >

< /tr >

< tr >

< td > 1 < /td >

<td> Java </td>
<td> 1995 </td>
</tr>
</table>
</body>
</html>

* Selenium is an Interfeice.

* Methods for selenium programe :-

1) System.setProperty ("Webdriver.chrome.driver", "*");
   - System: class
   - setProperty: Method Name
   - "*": give path for chromedriver file.

2) For Maximizing Window :-

driver.manage().maximize().window();

3) Thread.sleep (3000);
   - Thread: explicited time
   - 3000: time in msec (3x1000sec)

4) quite ();
   - for quitting the current windows.

5) close ();
   - for closing the all windows.

Q. Can we minimize the window?

Ans:- No we connot able to minimize the window by selenium code.

★ **program :-**

```
public static void main (String [] args) {

system.setproperty ("Webdriver.chromedriver", 'path');
                                                of folder

Webdriver d = new chromedriver();
d.manage().window().maximize();
Thread.sleep (2000);
d.get ("http://www.google.com");
Thread.sleep (3000);
d.get ("http://www.youtube.com");
d.quite();
d.close();
```

Q. when you run the script which message will shows on ?

Ans:- chrome is Automated by testing software.

★ **Webdriver methods :-**

It is an Interface used to perform actions on webdriver/webpage.

1) **get method();-**

It is used to enter URL in a browser and it will wait for to fully loaded to webpage.

Syntax:- Driver.get (" ");

2) **Maximize ():-**

This method is used to maximize the browser and window

Syntax: d.manage().window().maximize();

3) **Thread.sleep():-**

It is used to stop loading stop the application for sometime by providing seconds, minutes and hours.

Syntax: Thread.sleep(3000);

4) **close ():-**

The method is used to close a current browser or selenium focuses browser tabs.

Syntax: d.close();

most of the times it is used to close a current tab.

5) **quit():-**

This method is used to quite / close all the tabs present in the browsers. It will be used is the end of the script.

syntax: d.quit();

6) **navigate().to:-**

Navigate used back, forward multiple used for URL.

It is used to Navigate URL on the browser without loading the page.

syntax: navigate().back();

and if that will not properly loaded that time we provide,

Thread.sleep (ms);

Also navigate().forward();
navigate().Backward();
Navigate().Refresh();

so for Navigate().to command, In webdriver this method loads a new web page in the existing browser window.

Q. Can we minimize the browser?

Ans: No we cannot minimize the browser.

* for testing enviornment URL so for that developer directly gives a testing enviornment.

Q. How we can check the title for browser?

Ans: driver.getTitle()

At that time they given and self placed as own stored as memory.

How will get title on console :: System.out.println(T);

for TITle and URL :-

String h = d.getCurrentURL();

Q. What is a Webpage?

Ans: Webpage is a collection of webelements like textbox, button, radio button, checkbox, select table elements, object, link, title, text.

Live functionality :- The element which shows functionality : The functionality will get refresh this is live.

DATE / /

Dead functionality :-

The element which are not refresh or reload that page then dead.

eg:- facebook application

① Create an Account

② facebook help you connect and shaur with the people.

The element showing purpose inside the Application is called dead element.

a) UI developer

b) front-end developer

c) Back-end developer

Right click → View page source.

1) forward command :-

In webdriver, this methode enables the web browser to click on the forward button in the existing browser window. It neither accept anything nor returns anything.

driver.navigate().forward();

2) Back Command :-

In webdriver, this method enable the web browser to click on the back button in the existing browser window. It neither accept anything nor return anything.

driver.navigate().back();

3) Refresh Command :-

In webdriver, this method refresh / reload the current webpage in existing browser.

driver.navigate().refresh();

DATE / /

Q. How will create webpage?

1) To create web page we need to use keyword : html.
2) Every keyword should be close brace with angular brace using forward slash.
3) To create a component and element we need to use keyword input.
4) To create a list box or select table element we need to used select keyword.
5) To create a link we use keyword a with href hypes reference
6) To create a web table we need to use keyword Table.

* Locators :-

1) TagName :-

Any keyword which will be presented immediately after angular braces less than symbol.

< - less than

For example: < html → less than
< title → less than
< head → less than
< hr → less than
< tr → less than
< input → less than

In webpage multiple variable are present with same tagname of we use the tag name locators type to identify of element then selenium will perform action on first element present in webpage by default.

2) **Attribute** :-

Any keyword which will be presented after tagname with equal to symbol until greater than angular brace

eg:- UN < input type = 'text' >

- input type → Property Name
- 'text' → Property value
- Attribute Name
- Attribute value

3) **Text** :-

Any keyword which will be presented after angular brace greater than symbol and until end of that keyword

eg:- forgot password

- forgot password → text

* data type → webelement dp → tagname "input"
return type

we have find on the webpage

Sendr → input to the username.

* **Locators** :-

Selenium user to find webelement is nothing locator it method of "by" class by which we can find element.

Locators in selenium are one of the most powerful command It ideally the building block of the selenium automation script.

- Hence it helps to locate GUI elements through which multiple user actions can we perform.
- The Locators are one of the important parameters for scripting base foundation and they may lead to script failure.
- A Good scripting base foundation requires element to be located appropriately.
- For this and to achieved we have multiple locators in selenium webdriver using the below locators in selenium webdriver we can locate elements through "find element" & "find elements" method.
- Their are locators which are used to find the webelements from the web page.

1) TagName
2) Name
3) className
4) Id
5) Css selector
6) Link text
7) partial linktext
8) xpath → Normal
   → Index [relative x path by index]

1) ID:-

This is the most common way of locating elements since it's are supposed to be unique for each elements.

Syntax: d.findelement(By.TagName(" ")).sendkeys(" ");

Target formate: id = id of element

2) Locating by Name :-

Locating elements by name are very similar to locating by id, except that we use the "name" = Prefix instead

Target formate name = name of the element

d. findelement (By.name ("Username")). click();

3) Locating by link Text :-

This type of locaters applies only to Hyperlink texts. we access the link by prefixing our target with "link" = and then followed by the hyperlink text.

Target formate : link = link Text

syntax :- d. findelement (By.linkText ("")) click();

4) Locating By Id :-

To find the element by ID of the element

syntax :- d. findelement (By.Id ("")).sendkeys("");

5) Locating By partial link text :-

To find the element by text partially

syntax: d. findelement (By.partiallinkText ("got.pass")).click();

6) class :-

To find the element by classname of the element.

syntax :

d. findelement (By.classname ("mode")). click();

PAGE No.
DATE / /

7) Tagname :-

Tagname like Input, Div, Img etc.

Tagname of the particulars node.

Syntax :-

```
Webelement y = d.findelement(By.tagName("input")).sendkeys("abcde");
```

Link Text → search for exact text which present inside it.

partial Link Text → Search for similar text which is present inside it.

findelement method which is used to access a single web element on a page. Return the first matching element. It throws a NosuchElementException when it fails to find if the element. Findelement method return the list of all matching element.

* Questions :-

1) Why we use Id?

Ans:- - The ID stored in ID attribute of an HTML Dom element, is unique for every element in the page by design.
- Thus an ID can uniquely identify an element.
- To use this feature, one need to call the findelement(By.Id("")) method of the webdriver class.

2) Why HTML code is required in ~~HTML~~ (Selenium)?

Ans:- The reason is selenium webdriver identifies the elements which are there on the application using HTML tags of that Application. So, as a selenium resource we should know the basic HTML knowledge to identify the elements uniquely.

Q. Write down the procedure to identify html code for an webelement in webpage?

Ans: go on webpage → Right click → Inspect (so will get the html code for webelement in webpage)

Q. difference between partial linktext and Link text?

Ans: - Using partial link text in selenium to locate an Element
- partial link text in selenium is another way of locating an element via a link.
- The only difference from the link text in selenium to partial link text is that it does not look into the exact match of the string value but works on a partial match.

* Questions:-

Q1) write a test script to enter name in webpage using Tagname locator?

Q2) Write a html code for open webpage which include username and password element.

Q3) what are the exceptions you handle in selenium?

Q4) why do you use selenium webdriver ??

Q5) How you clear browser cookies by using selenium webdriver?

d.manage().deleteAllCookies();

DATE : / /

* Absolute path :- Root of the parent to child is called absolute xpath.

eg:- Html / body / div / Link

* Relative path :- Root of the parent of it's immediate child.

eg:- // div / Link /

PAGE NO.
DATE 5 september 2020

* Notes By Me *

* **Locators :-** - Locators are the methods of By class.
- Locators are used to identify elements present in a webpage.
- also we can say that Locators are defined as an address that identifies a web element uniquely within a webpage.
- so there are 8 types of Locator we have they are as follows:
  1) ID
  2) Name
  3) className
  4) LinkText
  5) partial LinkText
  6) TagName
  7) css selector
  8) xpath
- Locator provides a way to access the HTML elements from a web page.
- Locator Helps to perform actions on the text-boxes, links, checkboxes and other web element.

1) **ID :-**
- ID locator has a unique value for the element which is present on webpage.
- It is unique reference for web object.
- for example :- (syntax)

If there is html code :- `<input id="user" class="required" type="text"`

```
Webelement a = driver.findElement(By.id("user"));
```

Datatype — reference Variable — reference variable of webdriver — method of webdriver — class — Locator — Value

DATE : / /

2) **Name** :-

- Every form has a Input fields with unique Names.
- Names are unique most of the times.
- Name locator is best choice for testing a login form.
- for example (with syntax) :-

If there is html code :- <input id = "user" name = "admin" class = "required type = "text" >

```
Webelement b = driver. findElement (By. name ("admin"));
```

locator

3) **Link Text** :-

- It is a perfect way to find links on a page.
- It is used for checking Navigation flow.
- for example (with syntax) :-

If there is Link we take from HTML code (by inspect) the element-

 How to use locator

```
WebElement c = driver. findElement (By. LinkText ("How to use locators"));
```

locator

4) **Partial Link Text** :-

- partial link text is similar as link text.
- It only differs in the way of using it.
- for example (with syntax) :-

link inspect in HTML code

 How to used partial link text locator?

```
WebElement d = driver.findElement(By.partialLinkText("How to use partial link Text?"));
```
(partialLinkText → Locator)

5) **Tag Name :-**

- TagName locator is also used to find the element.
- for example (with syntax) :-

```
List<WebElement> e = driver.findElements(By.tagName("result"));
```
(List<WebElement> → Datatype we used for "findelements"; findElements → method of webdriver; tagName → Locator)

6) **class Name :-**

- The class Name locator is used a specific class attribute to get a first element on a web page.
- for example (with syntax) :-

```
WebElement f = driver.findElement(By.className("sample"));
```
(className → locator)

7) **Css. selector :-**

- Css selector works faster with Internet explorer.
- flexibility of using multiple attributes for locating an element.
- special symbol also can be used.
- syntax :-

```
("TagName[Attribute='Atribute value']");
```

- for example :-

Attribute

```
WebElement g = driver.findElement(By.cssSelector("input[id='email']"));
```

Locator — Tagname — Attribute value

8) Xpath :-

- It is perfect technique for walking through the Dom structure (Document object Module) of the web page.
- xpath locator Rubust and Reliable.
- It is one of the method which is used to locate any element on the page using the xpath.
- There are two types of x-path :-
  1) Absolute path
  2) Relative path

1) Absolute xpath :-

- It start from the root element within the web page or part of the page and goes to identify the target element.
- It will start from starting Node and always start with HTML
- for writing xpath for absolute use '/' single forward slash.
- For example :- Html / Head / Body / Table / tr / td

2) Relative xpath :-

- The relative xpath are easy to manage as they are short and consise.

- Navigate root of the parent to any child that is Relative xpath.
- relative xpath start from middle Node.
- It will always start with '//' double forward slash.
- for example:-

```
//table/tr/td
```

* **How to write xpath :-** [Q. How to approach for writing xpath?]

1) First give main "HTML"
2) secondly start with '/'
3) In which body so write "body"
4) After that give the Name of "div"
5) Inside div which tagname is present "<input ....>"
6) If you didn't get then simply look in the (control + F) search box → Find :- | No of No. - (ex: 1 of 3) so that you can easily configure out.
7) for giving index No you should write within "[ ]".
8) example:-

```
//Html/Body/Div2/Input[2] ← xpath
```

* **Where to write xpath :-** [Q. where we write xpath ?]

- After Inspect the HTML code for Webelement.
- press [ctrl+F] on your Dom (Document object model) page.
- So you will get text box for "Find" so in that we have to write x-path.

Find :-	"Write xpath Here"	0 of 0

* Disadvantages of Absolute xpath :-

1) '/' forward slash gives minimal security to path.
2) We need to understand the HTML tree code whole before write xpath.
3) Absolute xpath is lengthy and it takes too much time.
4) Identifying HTML tree code is time consuming.

* Relative xpath :-

- For writting Relative xpath we should start with '//' double forward slash.
- '//' provides more security.
- Relative xpath divided into 4 types :-
  1) xpath by Index
  2) xpath by Contains
  3) xpath by Text
  4) xpath by Attribute Name
  5) xpath by TagName

1) Relative xpath by Attribute Name :-

- Syntax :-

```
//TagName [@ Attribute Name = 'Attribute Value']
```

- for example :- (Actual command)

```
driver.findElement(By.xpath("//input[@type='password']")).
                                                sendKeys("1234");
```

class Locator — Attribute Name — Attribute Value — Tagname — Input

2) Relative xpath by Text function

- syntax :- // TagName [Text() = 'text value']

eg:- // div [text() = 'Login']

- for example :-

Byclass Locator

driver. findElement (By. xpath ("// a [text()='Forgot password']")). click();

(don't take this eg:- in consideration)

By text function click action

above example correctly :- (This is correct)

driver. findElement (By. xpath ("// a [text() = 'Forgot password']")). click();

3) Relative xpath by contains :-

- In this there are two ways to write xpath by contains

1) syntax :- (1 type) :-

// tagName [contains (@ Attribute Name, 'Attribute value')]

for example :-

driver. findElement (By. xpath ("// input [contains (@ id, 'abcd')]"));

- syntax(syntax of a type) :-

```
// tagname [contains (text(), 'text value')]
```

- for example :-

```
d.findElement(By.xpath("//a[contains(text(),'abc')]"));
```

4) Relative xpath by TagName :-

- (syntax) with example :-

```
d.findElement(By.xpath("//div/input")).click();
```

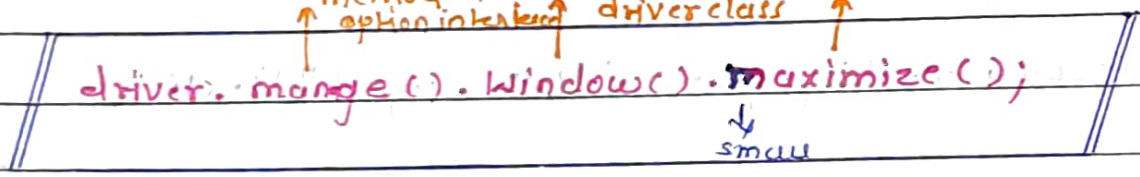
5) Relative xpath by Index :-

- syntax :- (for example) :-

```
d.findElement(By.xpath("//div/input[2]")).click();
```

★ Maximize Browser method() :- [Q. Why maximize browser in Selenium Automation?]

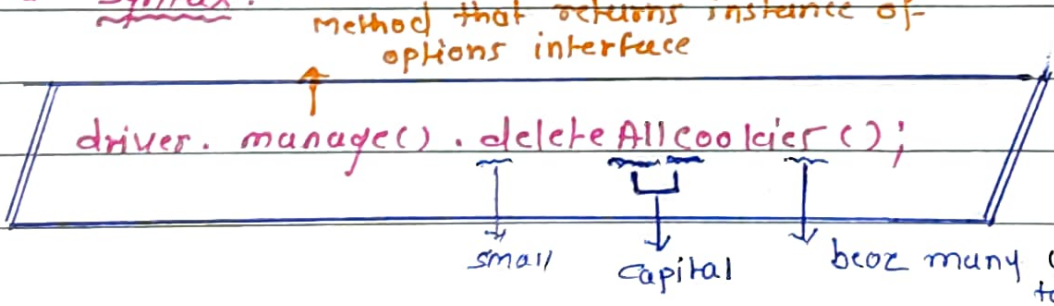
- Sometimes it happens that Elements on the web Application may not be recognized by the selenium if the browser is not maximized.
- So for that purpose we need to maximize the Browser.
- It is good practice to maximize the browser while automating any web application.
- So that script gets executed successfully without any error.
- Method for maximize :- `maximize();`
- Syntax :-



- Maximize() - is a method used to maximize the current browser.

★ How to Delete All cookies :-

- Each cookie is associated with a name, username, password, value, expiry, path and status of whether it is secure or Not.
- In order to validate a client, a server parser all of these values in a cookie.
- After deleting all cookies it will smoothly run.
- Method for DeleteAllcookies :- `DeleteAllcookies();`
- Syntax :-



* **How to set size of Windows :-** Q. It is possible to set the size of window ? J
  Ans: yes

- Selenium webdriver does not provide any method for minimizing browser / window, there is not such direct method.
- so in that case we are using resize method to minimize the browser / window.
- **method for resize :-** setSize() ;
  ↳ capital.
- There is setsize() method used to set the size of the current browser.
- **syntax :-**

```
Dimension d = new Diamension (Width, Height);
// Resize current window to set diamension
driver.manage().window().setsize(d);
```

(class; parameterized constructor; ↳ method)

- firstly we have to Import "Diamension" class.
- we need to create an object for Diamension class.
- In diamension we need to pass the width and Height i.e (parameterized constructor.)

where,

setsize() :- is a method of Diamension class.
Diamension :- is a class
munge() :- method of option Interface.
window() :- is an interface of driver class

- setsize() method is Implement into Diamension class.
- In this way we can achieve resize the window / browser.

DATE

set position [Q. can we set the position of window ?.]

* How to set the position of Windows :-
  - Method for setting the position :- setPosition();
  - For setting the position of Window / browser we use using selenium code / syntax to setposition.
  - For set the position we have point class.
  - syntax :-

```
Point p = new Point (x, y);
// for set the position
driver.manage().window().setPosition(p);
```

(Point: ↑ class; setPosition: capital)

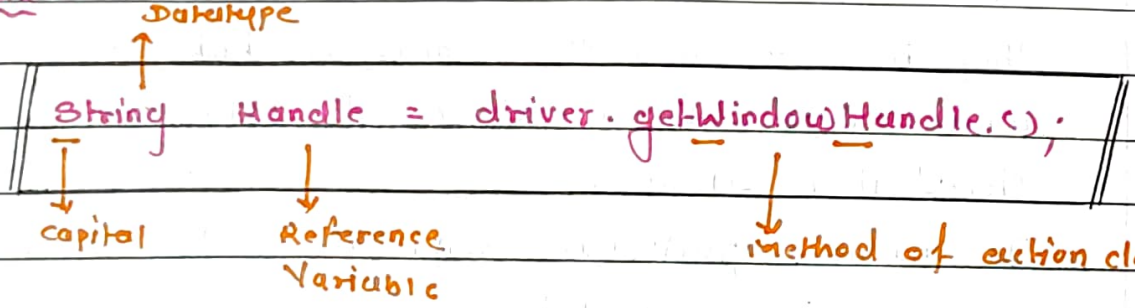
  - setposition() :- is a method of point class.
  - point :- is a class
  - We need to create object for the point class and in that pass the Argument like values of (x, y) through the point class constructor (i.e parameterized constructor).
  - In this way, by using selenium webdriver we can set the position of Window.

* How to Handle multiple Windows :-
  - Q. What is Window Handle :-
  - A Window Handle is a Unique Identifier that holds the address of all the windows.
  - This is basically a pointer to a window, which returns the "string value".
  - This Window Handle function help in getting the handles of all the Windows.
  - Each browser has a unique Window Handle.

- There are two methods to handle windows in web driver
  1) getWindowHandles ()
  2) getWindowHandle ()

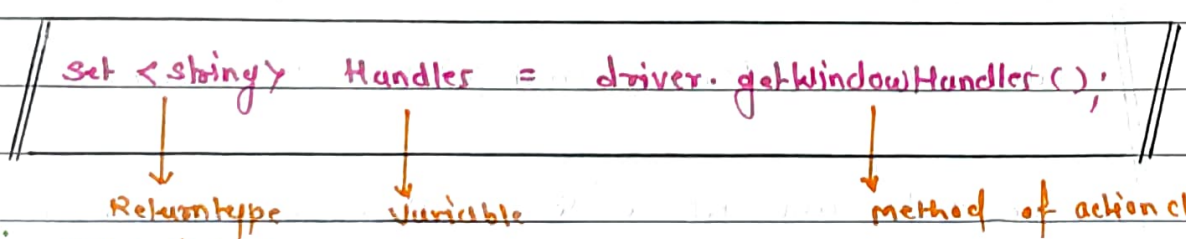
1) getWindowHandle () :-
- It help in getting the Window Handle of the current Window.
- for this getWindowHandle () method we use action class.
- It will Return only "1 Window" and it's return type is "String" (i.e ids means Alphanumerics).

- syntax :-



2) getWindowHandles () :-
- It help in getting the handles of the all the Window opened.
- Here also we used the action class.
- It returns a "set of Handles" of all the pages available.
- It is used to get the address of all the open browser and it's return type is iterator<String>." (i.e (String set))
- It Return set of WindowHandle (string) which can be used to iterate over all open windows of this webdriver instance by passing them to "switchTo(), webdriver.options.window()"

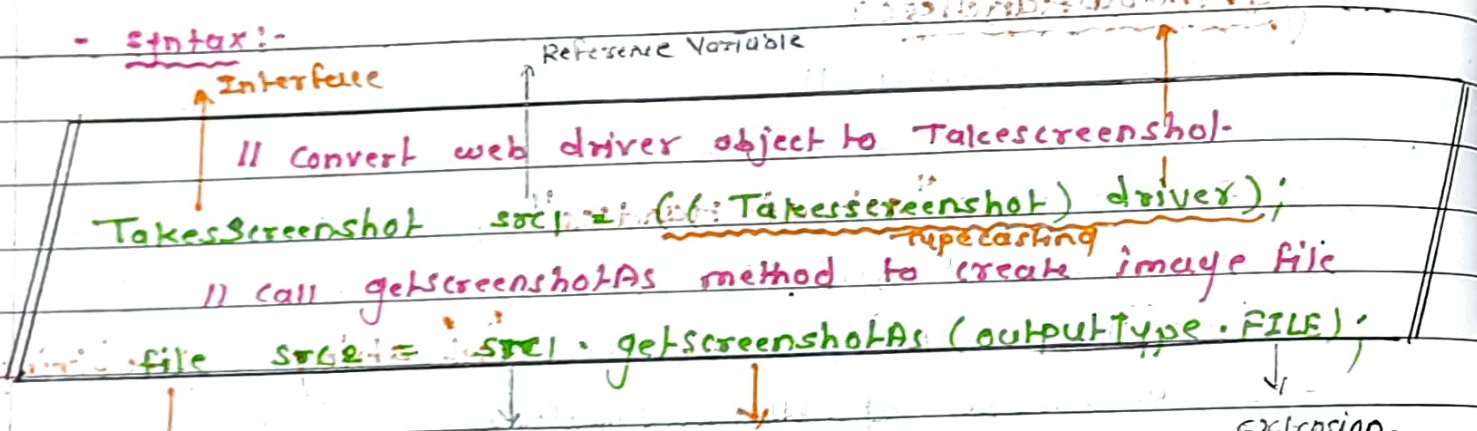
- syntax :-



* How to take screenshot by selenium code :-

- Generally as tester every time we should take screenshots for proof.
- so it is hectique to go everytime on snippin tool and take screenshot and it will consume more time.
- So A screenshot in selenium webdriver is used for bug / defect Analysis.
- selenium webdriver can automatically take screenshots during the execution.
- But if users need to capture a screenshot on their own, they need to used the Takesscreenshot method (Interface) which notifies the webdriver to take the screenshot and stored it in selenium.
- To maintain particular script record like it may pass or fail we take screenshot.

- syntax :-



- example :-

```
File src = ((TakesScreenshot) driver).getScreenshotAs(OutputType.FILE);

File dest = new File("C:\\Job\\screenshotmethod\\signup.png");
```

DATE / /

FileHandler.copy(src, dest);

where,

File :- class

dest, src :- reference variable

TakesScreenshot - is an Interface

getScreenShotAs() - method of TakesScreenshot

("C:\Job\Screenshot-method\signup.png") :- path where we need to store screenshot

(It is folder path where I create to save screenshot) (output.FILE) ← Name given by us

FileHandler :- is a class to copy the contents of the source directory to the destination directory.

output.FILE :- (outputType.FILE) - .file is a extension

Copy :- method of filehandler class which is static method, it will asic two argument i.e source & destination.

Note:-

1) file path save with (.png) extension)

[Q. where we store screenshot? / Q. Is selenium store screenshot itself or we need to take action?]

- By using selenium Webdriver we can take a screenshot but that screenshot selenium webdriver keep it with Selenium itself. (store in selenium)
- so for that screenshot share in local drive we have to copy that file with the help of filehandler class in which we have copy method.
- So first we need to create a folder in local drive and pass the path ("path\\Name of Screenshot.png").
- Then it will be stored at this location/destination.

Q. How screenshot will take by selenium webdriver?

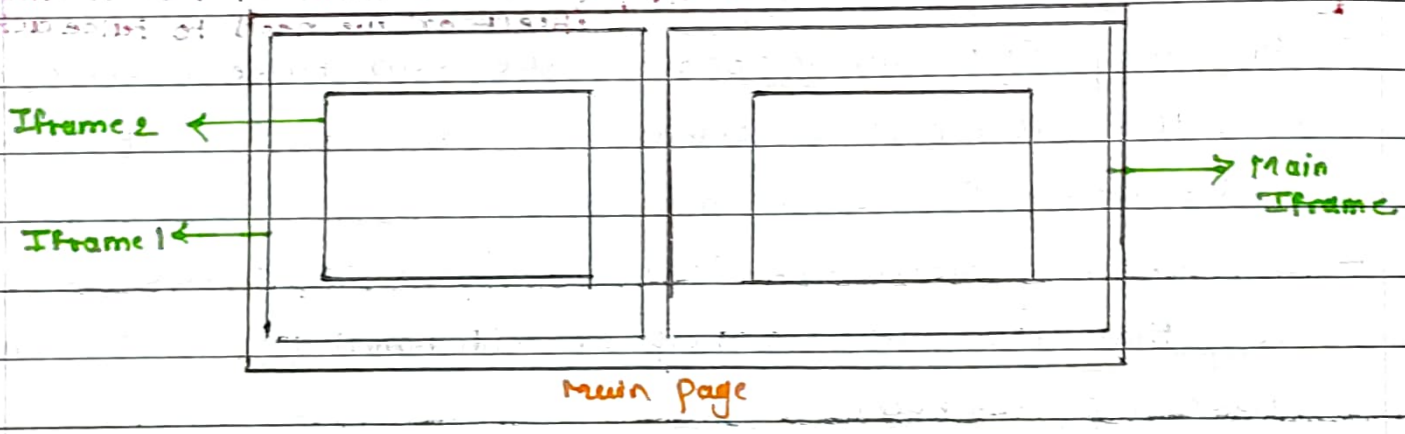
Ans:- Selenium will take screenshot of body of the page.
- It will take screenshot from below the URL.
- It means the desktop / Laptop visible screen, only that screenshot will taken by selenium webdriver code.

Q. which file we need to import that means with which extension?

Ans: Import org.openqa.selenium.webdriver;

* How to Handle Iframe in selenium Webdriver :-

- In our Webpage there are many iframes present, so if we want to access the webelement which is present on webpage but in different iframes.
- We have method to access the iframe webelements with "switchTo()" method.



- generally, if on webpage there are many webelements are present and you have to provide security to webelement- so in that case also Iframes concepts are used.
- In HTML code iframe defined with tag "<iframe>"

1) How to switch over the elements in iframe using Webdriver Commands :-

- Basically, we can switch over the elements in frames using 3 ways they are as follows:-
  - i) By Index
  - ii) By Name or Id
  - iii) By Webelement
- For Navigating Mainframe to Iframe :- driver.switchTo().frame("zd, Name, Webelemnt");

i) switch to the frame by Index :-

- Index is one of the attributes for the iframe through which we can switch to it.
- Index of the iframe starts with '0'.

Syntax :-

Method of Interface

```
driver.switchTo().Frame();
```

- example :-

suppose if there are 50 frames in page, we can switch to the iframe by using index

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

ii) switch to the frame by Name or ID :-

- Name and ID are attribute of iframe through which we can switch to it.
- syntax :-

by Name = "Name" iframe

```
driver.switchTo().frame("Name of iframe");
driver.switchTo().frame("Id of the element");
```

by Id = "value"

- example:

1) driver.switchTo().frame("frame20");
2) driver.switchTo().frame("id of the element");
3) driver.switchTo().frame("a077aa56");

iii) switch to the frame by web Element :-

- We can even switch to the iframe using web element:

- syntax:

we store element in webelement id
we have to this here
WebElement element = new

```
driver.switchTo().frame(WebElement);
```

2) If we want to set the focus / switch the focus from sub iframe to mainframe / parent frame :-

- syntax:-

```
driver.switchTo().defaultContaint();
```

3) If we want to Navigate from sub-sub-iframe to iframe:-

- for example iframe 3 to iframe 2 (it's immediate frame or parent frame)

- syntax:-

```
driver.switchTo().parentFrame();
```

- So In this way to handle iframes which is in the webpage so, to set focus from main to iframe, iframe to main and sub-sub iframe to iframe, we have those methods:-

Q. How will set the focus of iframes?

Ans: By using method like 1) driver.switchTo().frame();
iframe to mainframe ← 2) driver.switchTo().defaultContent();
sub-sub-iframe to iframe ← 3) driver.switchTo().parentFrame();

Note: 1) In html code :- code for Iframe

```
<iframe width='560' height='315' wmode='transparent'>
```

* Scroll Up-Down Method :-

- scroll up-down is a action of mouse perform by browser.
- A scrollbar is a lets you move around screen in horizontal or vertical direction on current page.
- If the current page scroll does not fit the visible area of the screen.
- It is used to move the window up and down.
- for scroll using selenium we can use JavascriptExecutor Interface that helps to execute javascript method through Selenium webdriver.
- Syntax :-

Interface class ↑

```
JavascriptExecutor js = ((javascriptExecutor) driver);
js.executeScript(script, Argument);
```

Typecasting

↓ Method of JavascriptExecutor

- example :- 1) for "scroll down" :- (Vertically down) (0,Y) Y :- positive Y value.

```
JavascriptExecutor j = ((JavascriptExecutor) driver);
j.executeScript("scroll(0, 3000)");
```

2) for "scroll up" :- Vertically up (0,-y) :- (-y) pass value

```
JavascriptExecutor j = ((JavascriptExecutor)driver);
j.executeScript("scroll (0,-2000)");
```

3) for "scroll Right" :- (Right →) (x,0) -x :- Negative value

```
JavascriptExecutor j = ((JavascriptExecutor) driver);
j.executeScript("scroll (2000,0)");
```

4) for "scroll Left" :- (Left ←) (x,0) x :- positive value

```
JavascriptExecutor j = ((JavascriptExecutor) driver);
j.executeScript("scroll (3000,0)");
```

Note:- 1) x-pixel is the number at x-axis it moves to the left if No is positive and it moves to the Right if No is Negative.

2) y-pixel is the number at y-axis it moves to the down if number is positive and it move to the up when no is Negative.

* Drag and Drop Action :- [Mouse Action]

- This is an action performed with a mouse when a user moves (drags) a web element and then places (drops) it into an alternate area.
- This is very common action used in windows Explorer while moving any file from one folder to another. Here, the user selects any file in the folder, drags it to the desired folder and just drops it.
- To perform the drag-drop action through ei selenium script- we have the "actions class" which provides various methods for Drag-Drop action so they are as follows:-

i) dragAndDrop (WebElement source, WebElement target)
ii) dragAndDropBy (WebElement source, int xOffset, int yOffset)
iii) dragAndDrop (Source, target)

i) dragAndDrop (WebElement source, WebElement target) :-

- This method performs left click, hold the click to hold the source element, moves to the location of the target-element and then releases the mouse click.
- Syntax:-

```
Actions a = new Actions(driver);
WebElement source = driver.FindElement(By.Locator(" "));
WebElement target = driver.FindElement(By.Locator(" "));
a.clickAndHold(source).moveToElement(target).release().build().
perform();
a.dragAndDrop(source, target).perform();
```

(Actions: class; clickAndHold, moveToElement, release, build: action class methods; perform(): action class method; dragAndDrop: actions class method)

- example :-

```
Actions a = new Actions(driver);
WebElement ele1 = driver.findElement(By.xpath("//div[@id='draggable']"));
WebElement ele2 = driver.findElement(By.xpath("//div[@id='droppable']"));
a.clickAndHold(ele1).MoveToElement(ele2).release().build().perform();
```

where :-

clickAndHold() :- dragAndDrop() method first perform click-and-Hold at the location of the source element.

MoveToElement() :- The source element gets moved to the location of the target element.

release() :- finally it release the mouse.

build() :- build() method is used to perform multiple action on one click.
The build is executed in the perform method internally.

perform() :- method is used to perform an single action on single click.

ii) dragAndDrop(WebElement source, int xOffset, int yOffset) :-

This method click and hold the source element and moves by a given offset, then release the mouse.

- offset are defined by x and y
  xoffset is horizontal movement
  yoffset is a vertical movement
- syntax:

```
Actions action = new Actions(driver);
WebElement source = driver.findElement(By.Locator(" "));
WebElement target = driver.findElement(By.Locator(" "));
action.dragDropBy(from, 100, 100).perform();
```

xoffset - Horizontal, y offset - Vertical

- example :-

```
Actions a = new Actions(driver);
WebElement b = driver.findElement(By.id("draggable"));
a.dragAndDropBy(from, 100, 100).perform();
system.out.println("Dropped");
```

iii) dragAndDrop (source, target) :-

- By using this method we can direct drag and drop.
- syntax:

```
Actions a = new Actions(driver);
WebElement s = driver.findElement(By.locator(" "));
WebElement d = driver.findElement(By.locator(" "));
a.dragAndDrop(s, d).perform();
```

Note:- 1) for "~~Right~~ Left click" by mouse :- doubleclick();
- a. doubleClick (reference Variable) . perform();

2) for "~~Left~~ Right click" by mouse :- contextclick();
- a. contextClick (reference Variable) . perform();

3) for "click" by mouse :- click();
- a. click() . perform(); **or**
- a. click();

★ How to select option from Drop-Down :-

- from Drop-Down we can select only 1 option.
- A Drop-Down Allow a user to select a value from a series of option.
- The "select" class in selenium webdriver is used for selecting and deselecting option in a dropdown.
- The object of select Type can be initialized by passing the dropdown webelement as parameter to it's constructor

- Syntax:-

Datatype ↑ WebElement; Reference Variable ↑ element

```
WebElement element = driver.findElement(By.xpath(" "));
Select s = new Select(element);       ← class
// there are 3 method to selection of option from drop-down
    1) s.selectByIndex(3);
    2) s.selectByVisibleText(" June");
    3) s.selectByValue(" ");
```

↓ methods to select option from drop-down

1) How to select an option from drop down menu?

- Webdriver provides three ways to select an option from the drop-down menu.
  1) selectByIndex()
  2) selectByValue()
  3) selectByVisibleText()

i) selectByIndex():-

- It is used to select an option based on it's index, beginning with '0'.
- syntax:-

```
s.selectByIndex(5);
```

ii) selectByValue():-

- It is used to select an option based on it's "value" Attribute.
- syntax:-

```
s.selectByValue("Database");
```

iii) selectByVisibleText():-

- It is used to select an option based on the text over the option.
- syntax:-

```
s.selectByVisibleText("Functional Testing");
```

* Handling pop-ups in webDriver :-

- popup are small window or seperate window which will be display only when we perform action on any compone present on webpage which have specific requirement of popup.
- A pop-up is a graphical user interface (GUI) display area usually a small window, that suddenly appears ("pops up") in the forground of the visual interface.
- Some pop-ups are handled by selenium directly or there are some pop-ups which is not handled by selenium, so we have to use third party tool i.e. "ATIos" and "AUTOIT".
- In Web application there will be different types of pop-ups they are as follows:-

1) Hidden Division pop-up
2) child Browser pop-up
3) Alert Pop-up
4) Authentication pop-up
5) File upload popup
6) File Download popup
7) Window popup

} These popup are mostly used and able to handle by selenium

} These are mostly handled by third party tool.

1) Hidden Division pop-up :-

- If the pop-up displayed in the browser opened by the "webDriver" is colorful, we can't move it and also we can't inspect it then it is a hidden pop-up.

- It is created by the developer by using driver. by default it will be hidden.
- It will be made visible whenever it is required by changing value of display property from none to block.
- since hidden-division pop-up is a part of web page only so we don't use switchTo() to identify the buttons present on the hidden division using findElement() only.

2) **child Browser pop-up** :-

- If the child Browser is displayed in order to perform any operation on the child browser we need to make the web driver to switch from parent browser to child browser.
- so for this we use "driver.switchTo.window()" which takes window handle of the child browser.
- To get Window Handle of the child browser first we get window handle of the all the browser using "getWindowHandles()" of WebDriver.
- We can drag and drop it.
- We can inspect element present in popup.
- This popup will contain address field, maximize, minimize and closed option.
- If multiple childbrowser are displayed then we need to identify webpage are child browser Id or Address which is unique.

- **Syntax**:-

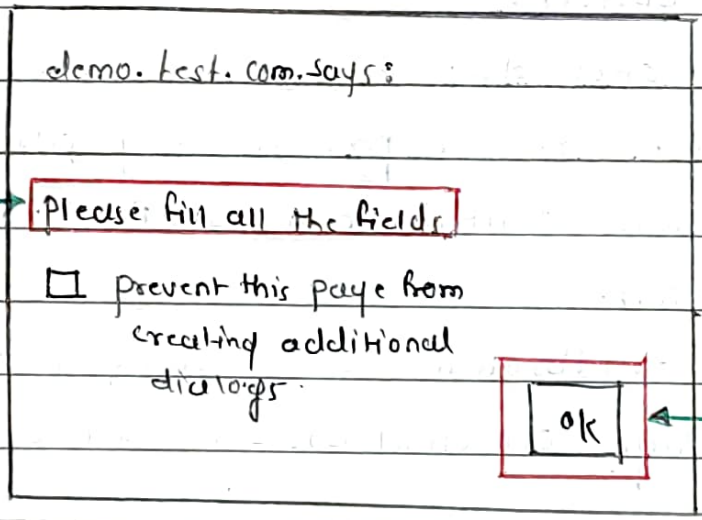
```
driver.switchTo().window();
```

3) Alert pop-up :-

- An Alert in selenium 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, ask for permission to perform certain tasks and it also provides warning message as well.
- There are few type of Alert :-
  1) Simple Alert
  2) prompt Alert
  3) Confirmation Alert

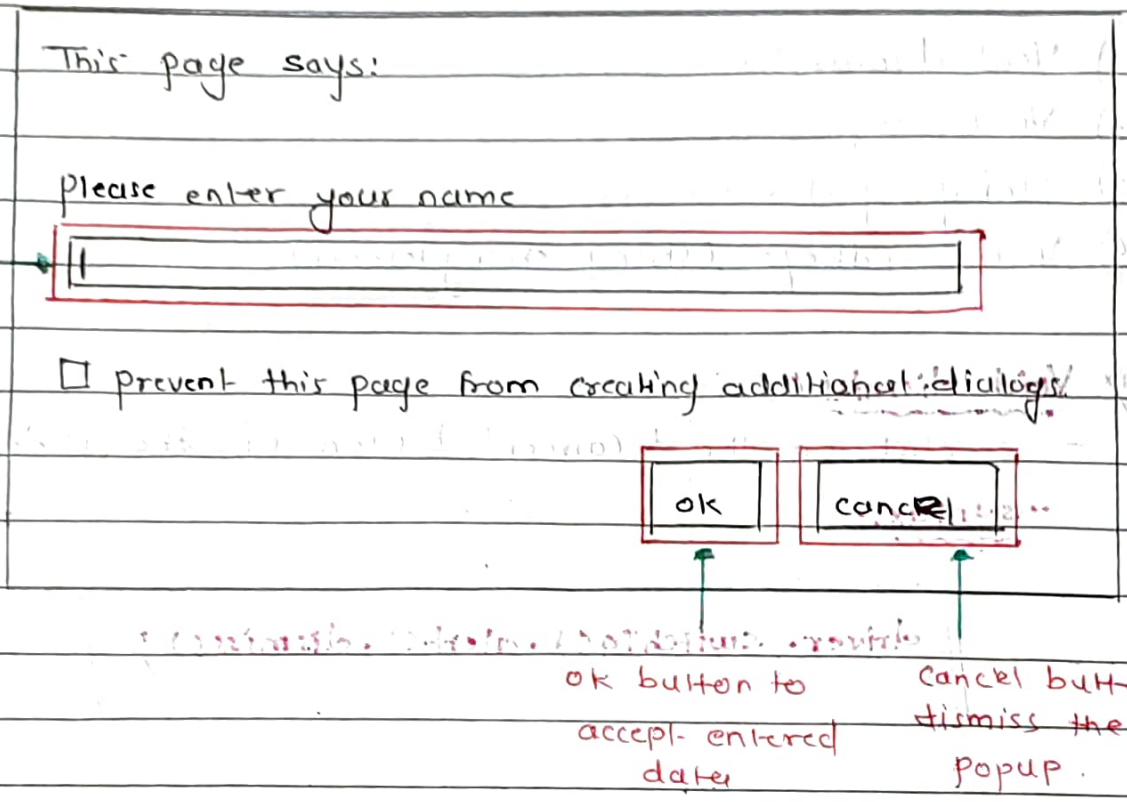
1) Simple Alert :-

- This simple Alert displays some information or warning on the screen.



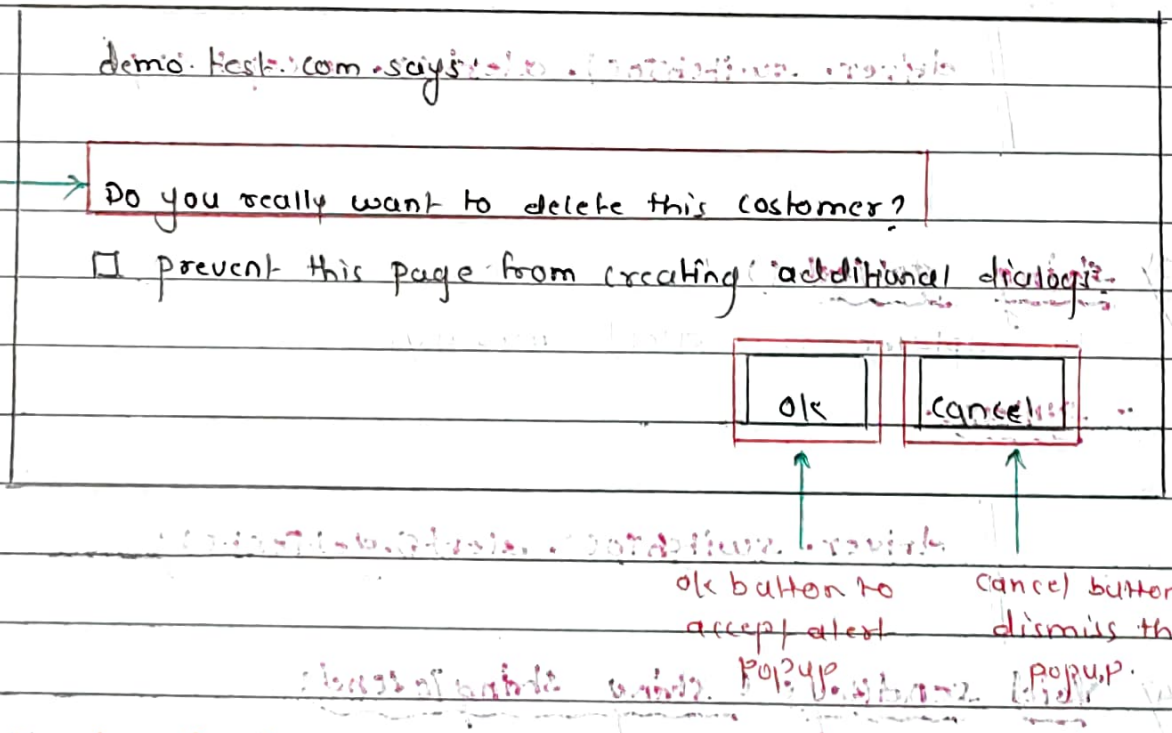
2) prompt Alert :-

- This prompt Alert asks some input from the user and selenium webdriver can enter the text using sendkeys ("input



3) **Confirmation Alert** :-

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



How to Handle Alert :-

Alert interface provides the below few methods which are widely used in selenium webdriver.

1) Void dismiss ()
2) Void accept ()
3) string gettext()
4) Void sendkeys (string stringTosend)

1) Void dismiss () :-
- To click on the 'Cancel' button of the alert.
- syntax:-

```
driver.switchTo().alert().dismiss();
```

2) Void accept() :-
- To click on the 'Ok' button of the alert
- syntax:-

```
driver.switchTo().alert().accept();
```

3) string getText() :-
- To capture the alert message.
- syntax:-

```
driver.switchTo().alert().getText();
```

4) Void sendkeys (string stringTosend) :-
- To send some data to alert box
- syntax:-

```
driver.switchTo().alert().sendkeys("Text");
```

* Synchronization (Waits in selenium) :-

- It is wait or hold till the object or it's properties get fullfilled.
- Why synchronization ? - 1) To match the speed of tool with the application speed (because tool speed is high).
  2) To match the speed of our tool with the application which is under test.
- When two or more components involved to perform any action. we expect these components to work together with the same pace. The cordination between these components to run parallely is called "synchronization".
- Synchronization can be classified into two categories:
  1) Unconditional synchronization
  2) Conditional synchronization

  Also these synchronization having their sub types they are as follows:

1) Unconditional synchronization :-

- In this we just specify timeout value only.
- We will make the tool to wait until certain amount of time and then proceed further.
- To this there is another subtype is
  - i) Thread.sleep() :- In this we have to make the application to wait for certain amount of time by specifying the timeout value.

2) Conditional synchronization :-

In conditional synchronization we specify a condition along with timeout value, so that tool waits to check for the condition for a particular condition to satisfy.

- There are three types of conditional Synchronization.
  1) Implicite wait
  2) Explicite wait
  3) Fluent wait

1) **Implicite Wait :-** (compile time polymorphism)

- The Implicite wait in selenium is used to tell the web driver to wait for a certain amount of time before it throws a "No such Element Exception".
- The default setting is '0'.
- Once we set the time, the web driver will wait for the element for that time before throwing an exception.
- Compile time polymorphism is used in Implicite wait.
- We will pass Argument as Timeout(hour, day, month, sec, year) and TimeUnit.seconds through ImplicitlyWait constructor so it will behave as compile time polymorphism
- Implicite wait is globally declared.
- It is used when we don't exactly know how much time it will take to load web page.
- **syntax :-**

method

```
driver.manage().timeouts().implicitlyWait(TimeOut, TimeUnit.SECONDS);
```

where,

TimeOut - waiting Time

TimeUnit.SECONDS - Unit for Time period

2) **Explicit Wait** :- (Run Time polymorphism)

- The Explicit wait in selenium is used to tell the Web Driver to wait for certain conditions (Expected conditions) or maximum time exceeded before throwing "ElementNotVisibleException" exception.
- It can be applied for only specified elements. (It is applicable to specific element).
- Explicit wait is nothing but,

  Explicit wait = Waiting Time + condition
- Explicite wait gives us better option than implicite wait as will wait for dynamically loaded Ajax element.

  (Ajax - Asynchronous Javascript and xml)
- syntax:-

```
WebDriverWait wait = new WebDriverWait ( driver , Timeout);
WebElement expwait;
expwait = wait.Until (ExpectedConditions.VisibilityOfElementLocated (By.xpath("...")));
expwait.click();
```

  (WebDriver reference, 20 sec)

  where, Until - Repeatedly applies the instances input value to the given predicate until the timeout expires or the predicate evaluate to be true.
- example:-

```
WebDriver wait = new WebDriverWait ( driver, 20);
WebElement wait1;
wait1 = wait.Until (ExpectedConditions.VisibilityOfElementLocated (By.xpath ("\Html\Body\Div\")));.click();
```

- Expected conditions that can be used in Explicit wait:-

1) AlertIsPresent ();
2) ElementIsDisplayed ();

3) isSelected();
4) enabled();
5) elementToBeClickable();
6) TitleIs();
7) VisibilityOf();

3) **Fluent Wait :-**

- The fluent wait in selenium is used to defined maximum time for the web driver to wait for a condition as well as the frequency with which we want to check the condition before throwing an "ElementNotVisibleException" exception.
- It checks for the web element at regular interval until the object is found or timeout happens.
- Fluent wait is,

Fluent wait := (Condition + WaitingTime + frequency)

Explicite wait ... Fluent wait

- Syntax:-

```
Wait wb = new FluentWait ("Webdriver reference")
W. withTimeout (timeout, SECONDS)
W. pollingEvery (timeout, SECONDS)
W. ignoring (Exception.class);
```

Where,

withTimeout - set how long to wait for the evaluate condition to be true.

pollingEvery - set how often the condition should be evaluated.

Ignoring - configure the instance to ignore specific type of exception while waiting for a condition.

- example :-

Fluent wait + Explicit wait

```
Wait <WebDriver> w = new FluentWait <WebDriver> (driver);
w.withTimeout (10, TimeUnit.SECONDS);
w.pollingEvery (5, TimeUnit.SECONDS);
w.ignoring (NoSuchElementException.class);
w.until (ExpectedConditions.visibilityOfElementLocated (By.linkText
    ("logout"))).click();
```

- In this instead of passing TimeUnit.SECONDS we can able to pass "TimeUnit.NANOSECONDS".

where,

Frequency - setting up a repeat cycle with the time frame to verify the condition at regular interval of time.

4) **PageLoad Timeout Wait** :-

- Set the amount of time to wait for a page load to complete before throwing an error.

- syntax :-

```
driver.manage().timeouts().pageLoadTimeOut (TimeOut,
                                            TimeUnit.SECONDS);
```

- example :-

```
driver.manage().timeouts().pageLoadTimeout (10, TimeUnit.SECONDS);
```