Selenium

Robust

Selenium is a free and open source web application automation tool. It supports multiple browsers, programming lang and platforms.

Q// what are the types of selenium ?

i) Selenium core (Notepad)
ii) Selenium RC (Remote control)
iii) Selenium IDE (Record and play) Firefox Plugin.
iv) Selenium web Driver
v) Selendroid (Android only)
vi) Appium (ios & Android & windows)
vii) Winium (only windows)

IDE → Integrated Development environment

Q// what are the os supported by selenium ?

Ans: Windows, Linux, mac, ubuntu, Android, ios.

Q// what are the browser supported by selenium ?

→ Google chrome, Mozilla firefox, Internet Explorer Microsoft edge, safari, Opera, html, phantom JS.

Q// what are the languages supported by selenium ?

→ java, python, Ruby, C#, java script, Node js objective C, php, pearl, R, dart, TCL, Elixit haskell,.

Q// In selenium what types of Applications we Automate

→ Only Web Application.

Q// what type of test cases we Automate in selenium ?

→ Regression Test cases (FT, IT, ST)

Q// Do we Automate negative Testcases ?

→ Yes.

Q// which is is not supported in selenium ?

→ venix.

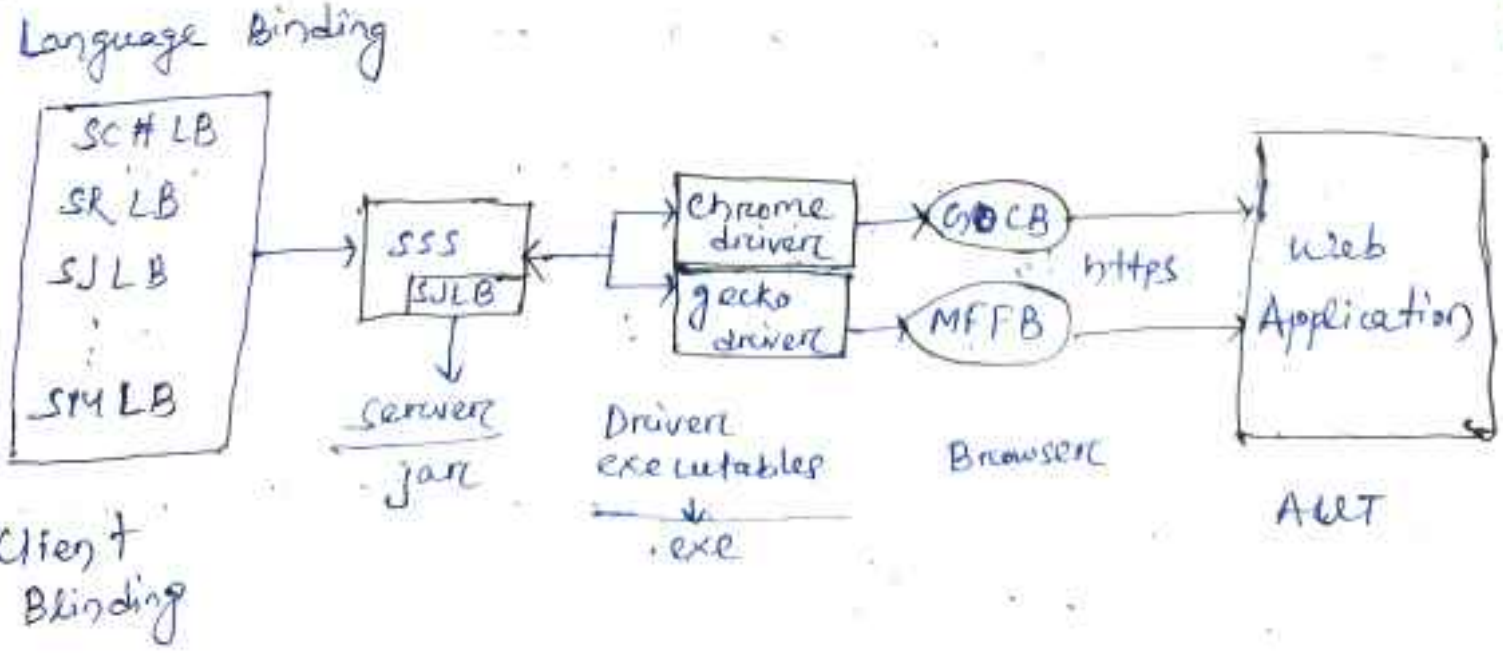
Q// which testcases are automated first ?

→ Smoke testcases.

Q// Is 100% Automation is possible ? why ?

No, Because we don't have technology to automate the features or it may be very expensive, It may requires manual intervention (Ex: otp, captcha cardswipping ; Biometric scanning , Barcode scanning , verification of audio & video files)

Architecture of Selenium :



i) Selenium supports 14 different coding languages like - C#, Ruby, Java etc these are called Language bindings or client bindings.

ii) These client bindings communicate with selenium server, then server performs the action on the browser with the help of driver executables.

iii) In order to perform action it uses JSON wire protocols (Java script Object Notation)

iv) Since selenium server internally contains selenium java language binding also, hence while installing selenium we use only server.jar file and driver executable file.

Steps to Install Selenium:-

1) Download the selenium server.jar file and Browser specific driver executable file from the download page of selenium (https://www.selenium.dev)

2) Extract the driver executable file (unzip)

3) In Eclipse create a new java project and create two new folders for driver & jar inside the java project.

4) Copy & paste driver executeable file into the driver folder. similarly copy & paste selenium jar file into the jar folder.

5) Add jar file to the build path, so that we can import statement to import the required class of selenium.

6) Dont add driver exe file into the build path. eclipse will not allow us to add exe file to the build path. To specify the path either we add it manually to the environment variable or we should add it programatically using System.setProperty statement

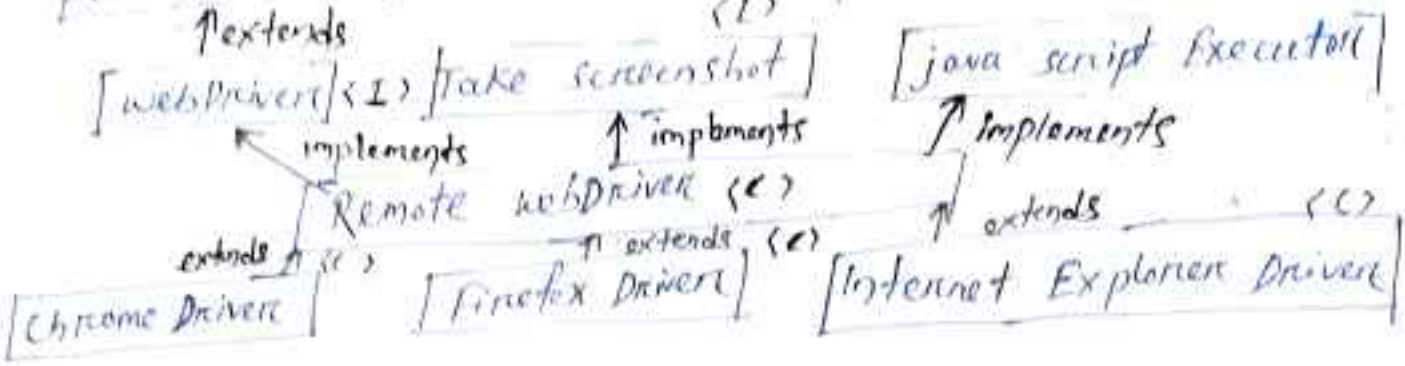
7) While specifying the path we use relative path by specifying dot (.) operator at the begening which represents current java project.

8) We should set the path before opening the browser or else we will get Illegal exception

9) For easy maintenance we set the path of all driver executable in the static block as shown below.

Ex →
```
class Demo {
    static {
        System.setProperty("webdriver.chrome.driver",
                           "./driver/chromedriver.exe");
    }
    p.s.v.m(s[] args) {
ChromeDriver driver = new ChromeDriver(); // To open chrome browser.
    driver.get("https://www.google.com"); // script to enter url
String title = driver.getTitle();
    s.o.println(title);
    driver.close();
}
}
```

selenium WebDriver



Method presenting in search context of interface ?

public
1. find elements (By) : web element
2. find elements (By) : list < webElement

Description of selenium webDriver Architecture:

i) The supermost Interface in selenium webDrive architecture is search context and it is extends by by webDriver interface and implemented in Remote webDriver class

ii) Remote webDriver class also implements other Interface such as java script executor, takes screenshot, java script webDriver etc.

iii) All the browser specific classes extends Remote webDriver such as Chrome Driver, firefox Driver internet Explorer Driver etc.

~~why we direct~~

Method of WebDriver Interface:-

1) close() datatype ⇒ void
2) get (string argument) : void
3) get currenturl() : string
4) get page source () ; string
5) get Title () ; string
6) get window Handle () ; string - webDriver
7) get window Handles ; set (string)
8) Manage () ; options - web driver
9) navigate () : Navigation - webDriver
10) quit () ; void - webdriver
11) switch To () : TargetLocator - webDriver

Method of Webelement Interface:-

1) clear () ; void - webElement
2) click () ; void -
3) get Attribute (string arg) : string - web element
4) get css Value (string arg) : string :
5) get Location () : point - webdriver
6) get Rect () : Rectangle
7) get screenshot As (output Type <x> arg)
8) get size () : Dimension
9) get Text () : string
10) get Tag Name () : string
11) is Enabled () : boolean
12) is Displayed () : boolean
13) is Selected () : boolean
14) send Keys (char sequence - args ()) : void
15) Submit () : void.

Method of takes screenshot interface:-

⇒ getScreenshotAs (outputType<x> arg) : x - TakeScreenshot

Methods of java script Executor Interface:-

1) ExecuteAsignScript (String arg(), object... args) : Object - JRI

2) ExecuteScript (string args(), object... args) : object

How do you ~~write a script to~~ enter the url without using get Method.

By using navigate().to

What is the difference bet^n get and navigate ?

→ By using get() we can enter only url

+ By using navigate, we can enter the url & we can perform backword, forward and refresh option.

Ex→

```
import org.openqa.selenium.chrome.ChromeDriver;
public class EnteredUrlWithoutget {
    static {
System.setProperty("webdriver.chrome.driver", "./driver/chromedriver.exe");
}
p s v m (s[] args) throws InterruptedException {
    ChromeDriver driver = new ChromeDriver();
    driver.navigate().to("https://www.google.com/");
    Thread.sleep(2000);
driver.navigate().to("https://www.gmail.com");
    Thread.sleep(2000);
```

```
driver.navigate().back();
Thread.sleep(2000);
driver.navigate().forward();
Thread.sleep(2000);
driver.navigate().refresh();
Thread.sleep(2000);
driver.close();
}
}
```

What is the difference between get() and to method?

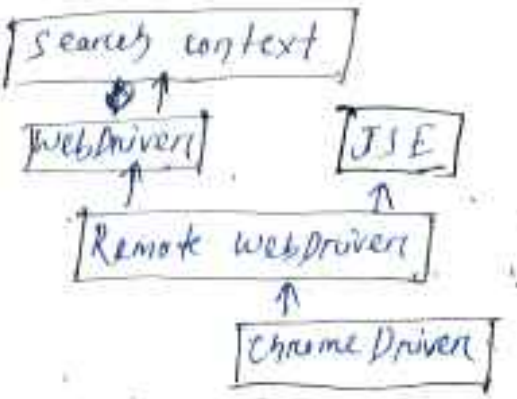
i) There is no difference bet^n get() & to() method because to() method internally call get() method both are used to enter the url. But get() will wait till page load and to() is does'nt wait for page load

Q// which protocol is used by selenium to communicate or interact with the browser.

JSON wire protocol
(java script object Notation)

Q// How selenium performing action on the browser?

By using the native methods of the browser



What is upcasting & it used in selenium ? why ?

→ converting sub class object into super class type is called upcasting

Example →

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

In selenium we use upcasting so that we can execute the same script in any browser

Ex →

```
public class DemoA {
    static void testA(WebDriver driver)
    {
        driver.get("https://www.google.com/");
        String title = driver.getTitle();
        S.o.p.ln(title);
        driver.close();
    }
}
class DemoB {
    static {
System.setProperty("webdriver.chrome.driver", "./driver
                                    /chromedriver.exe")

System.setProperty("webdriver.gecko.driver", "./driver
                                    /geckodriver.exe")

}
```

Why we direct upcast chromedrive to webdriver instead of Remote webdriver ?

If we upcasting to Remote Webdrive then it is also implementing others interfaces so it access unwanted method so accessing that's why we upcast direct to webdriver.

```
psvm(String[] args)
{
    WebDriver driver = new ChromeDriver();
        DemoA.testA(driver);
    WebDriver driver1 = new FirefoxDriver();
        DemoA.testA(driver1);
}
}
```

*Explain WebDriver driver = new ChromeDriver(); ?

i) WebDriver is an interface
ii) driver is a object reference variable
iii) new is a assignment operator
iv) new is a keyword to create a object
v) ChromeDriver() is a constructor.
vi) ; is used to end the statement (statement delimeter)

How do you close the browser without using close()?

→ By using quit()

What is the difference bet" close() & quit()?

→ close() method closes current browser
→ whereas quit() method closes all the browser (current / as well as child browser)
parent

Q/ How do you maximize the browser ?

→ By using driver.manage().window().Maximize();

Q/ How do you delete the cookies present in the browser ?

→ By using driver.manage().deleteAllCookies();

1) Write a script to check wheather seleniumhq.org url is successfully navigating to selenium.Dev or not

```
public class VerifyUrl {
    static {
    System.setProperty("wd.chrome.bdriver","./driver
                        /chromedriver.exe");
    }
  p s v m (s [] args) {
WebDriver driver = new ChromeDriver();
 driver.get("https://www.seleniumhq.org/");
 String eURL = "https://www.selenium.dev/";
String aURL = driver.getCurrentUrl();
 if (aURL.equals(eURL)) {
s.o.pln("URL is successfully navigating and pass");
 }
else {
 s.o.pln("URL is not sucessfully navigating & fail");
}
driver.close();
}
```

2) Write a script to print the html code of the web page

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class PrintHtmlSourceCode {
    p s v m (s [] args) {
        static {
System.setProperty("webdriver.chrome.driver", "/driver/
                                    chromedriver.exe");
    }
    p s v m (s [] args) {
WebDriver driver = new ChromeDriver();
    driver.get("https://www.google.com/");
String htmlCode = driver.getPageSource();
    s.o.pln (htmlCode);
    driver.close();
    }
}
```

Web Element :-

i) Anything present in the Web page is called as Web element.

Like ex→ Button, image, text field, logo etc

ii) Web elements are created using HTML

iii) Each HTML elements contains 3 things

1) Tag (tagname)
2) Attribute → Attribute name
   Attribute value
3) text

Ex→ HTML code of a login button is

```
<div id = "d1" > login </div>
```

In the above HTML code div is the tag.
id is the attribute-name, d1 is the attribute value
login is the text

*In order to see the html code of required element
→ right click on that element & select the option Inspect

if right click is disable on the page then press

→ ctrl + shift + i or F12

In order to inspect another element click on inspect button & then click on required element

Note:-

In selenium before performing any action such as clicking, typing, selecting etc we should find the element first using Locators.

Locators:-

i) Locators are used to find the elements.

ii) In selenium there are 8 type of locators & all of them are static methods present in By class

1) tagName()
2) id()
3) ~~class~~ Name()
4) className()
5) LinkText()
6) partialLinkText()
7) cssSelector()
8) xpath()

Q// What is the return type of findelement() method?

A → WebElement

Q// if the specified locator is not maching with any of the elements then what find element does?

A → It throws no-such element Exception

Q// if the specified locator is matching with multiple elements then what find element does.

A → It takes the first matching element & returns the address of first matching element.

Q What is the argument accepted by findElement method ?

A → By - type

Example for tagname locator:

```
import org.openqa.selenium.By;
public class DemoLocators {
  static {
System.setProperty("webdriver.chrome.driver", "./drivers/chromedriver.exe");
  }
    p s v m (S [] args) {
  WebDriver driver = new ChromeDriver ();
    driver.get("file:///C:/Users/qspte/oneDrive/Desktop/Demo.html");
    WebElement e = driver.findElement(By.tagname("a"));
    e.click();
  }
}
```

Example for tagname, id, name, className locator:

```
  public class DemoLocators {
  static {
System.setProperty("webdriver.chrome.driver", "./driver/chromedriver.exe");
}
  p s v m (S [] args) {
    WebDriver driver = new ChromDriver();
    driver.get("file:///C:/Users/qspte/oneDrive/Desktop/Demo.html");
  driver.findElement(By.tagname("a")).click();
    driver.navigate().back();
```

```
driver.findElement(By.id("id")).click();
driver.navigate().back();
driver.findElement(By.className("c1")).click();
driver.navigate().back();
driver.findelement(By.name("n1")).click();
driver.close();
}
}
```

LinkText & PartialLinkText:

i) Both LinkText & PartialLinkText can be used to find the link, if we try it on any other type of element we get no such element Exception.

Ex→ driver.findElement(By.linkText("Google")).click();

ii) if the text of the link is changing partially then we can use partial link text.

Ex: html code of a link is

<a>Inbox(7)

selenium code is:

driver.findElement(By.partialLinkText("Inbox")).click();

Limitation of PartialLinkText :

i) Element should be a link

Ex→ Inbox(7) It is not link

ii) Text should be partially changing

Ex→ <a> 7 → It is not partially changing - completly changing

Css selector:

i) css stands for cashcading style sheet and it is one of the feature present in html.

ii) css selector is one of the locator present in selenium.

The syntax is as ~~testo~~ followed →

"tag [Attribute name = 'Attribute value']"

Ex for css selector → "a [id = 'd1']" a [name = 'n1']

a [class = 'c1'], a [href = "https://www.jepidex.com"]

These are called as css expression

Note:

We can check the css expression in the browser by using following steps

1) inspect ~~inspect~~ any element
2) press ctrl + f
3) type the above expression

rule

1 of 1 → one maching element

1 of 3 → Multiple maching element

0 of 0 → no maching elements

```
DemoCssSelector {
    static {
System.setProperty("webdriver.chrome.driver", "./driver/chrome_driver.exe");
    }

    P.S.V.m(S[] args) {
    WebDriver driver = new ChromeDriver();
driver.get("file:///C:/Users/gspte/OneDrive/Desktop/Demo.html");
driver.findElement(By.cssSelector("a[id='d1']")).click();
    driver.navigate().back();
driver.findElement(By.cssSelector("a[name='n1']")).click();
    driver.navigate().back();
driver.findElement(By.cssSelector("a[class='c1']")).click();
    driver.navigate().back();
driver.findElement(By.cssSelector("a[href='https://www.jspiders.com/']"))
                                                    .click();
    driver.close();

    }
}
```

Note :-

If we do any syntax mistake while writing css selector or expression / xpath then we will get InvalidSelectorException.

e limitation of CSS selector is :-
It doesn't support text & we can't use text

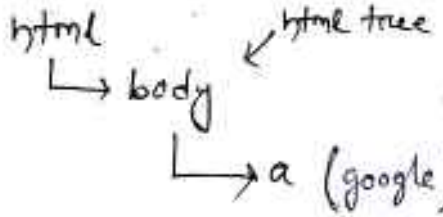
tee

P
Path

It
~~xpath~~ is the path of the element in a html tree is called x-path

While writing x-path it starts with dot (.)
which represents current webpage or html document

) using . dot is not Mandatory



xpath Expression :- ./html/body/a
(or)
/html/body/a

xpath in selenium :-

driver.findElement(By.xpath("/html/body/a")).click();

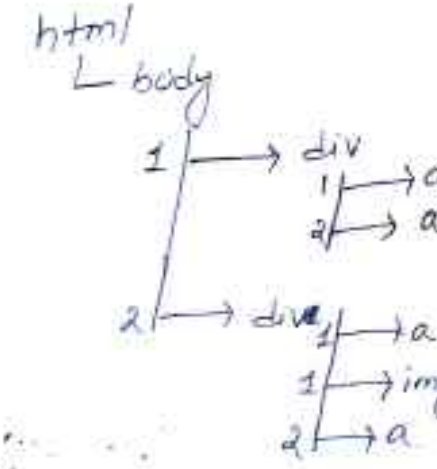
It is a selenium Technique to navigate through the html structure of the web page.

Types of xpath:

1) Absolute xpath
2) **Relative xpath**
   - i) xpath by attribute
   - ii) xpath by textFunction
   - iii) xpath by containsfunction
   - iv) xpath by traversing
   - v) Independent - dependant xpath
   - vi) xpath by group index

Absolute xpath:

Starting from html to the required or desired element is called as Absolute xpath



Link 1 → /html/body/div[1]/a[1]
Link 2 → /html/body/div[1]/a[2]
Link 3 → /html/body/div[2]/a[1]
Link 4 → /html/body/div[2]/a[2]
image → /html/body/div[2]/img
Link 1 & 2 → /html/body/div[1]/a
Link 3 & 4 → /html/body/div[2]/a
Link 1 & 3 → /html/body/div/a[1]

imp → Link 2 & 4 → /html/body/div/a[2]

Link 1, 2, 3, 4 → /html/body/div/a

All the above expression is called as absolute x-path

Note:-

In x-path we can use index which starts from 1, if there is another element under the same parent with the same tag name then index become 2 and so on

~~Att~~ ~~the~~

Relative x-path

Relative xpath starts with double forward slash (//) which represents descendent (child, grand child etc, great grand child)

Ex→ html
↳ body
↳ a Google

Ex→ //a

Link 1 → //div[1]/a[1]
Link 2 → //div[1]/a[2]
Link 3 → //div[2]/a[1]
Link 4 → //div[2]/a[2]
image → //div//img ← Relative xpath
link 1, 2, 3, 4 → //div/a or //a
Link 1&2 → //div[1]/a
Link 3&4 → //div[2]/a
Link 1&3 → //div/a[1]
Link 2&4 → //div/a[2]

Q// Write an xpath to identify all the links & all the images:

A → //a | //img

link 1 & 4 → //div[1]/a[1] | //div[2]/a[2]

link 2 & 3 → //div[2]/a[2] | //div[3]/a[1]

imp

Q// what is the difference bet'n //a & //table//a ?

→ //a matches with all the links which are present anywhere on the web page

→ //table//a matches with all the links which are inside the table

xpath By Attribute

We can also use attribute of the element while writing xpath

SYNTAX: 1) //tag [@AttributeName = 'Attribute value']

2) //tag [@AN1='AV1' and @AN2='AV2']

3) //tag [@AN1='AV1' or @AN2='AV2']

4) //tag [not(@AN='AV')]

Sample HTML code:

A1 : <input type="text" value="A" />

B1 : <input type="text" value="B" />

A2 : <input type="button" value="A" />

B2 : <input type="button" value="B" />

C1 : <input type="checkbox" checked />

C2 : <input type="checkbox" ~~checked~~ />

Q// Write an xpath to identify the following elements.

1) All the textbox
2) All the buttons
3) All the checkbox
4) Only 1st textbox
5) only 2nd textbox
6) All the textbox and all the buttons
7) 1st textbox & 1st button
8) only selected checkbox
9) only unselected checkbox

All the textbox → //input[@type='text']

All the buttons → //input[@type='button']

All the checkbox → //input[@type='checkbox']

Only 1st textbox → //input[@type='text' and @value='A']

Only 2nd textbox → //input[@type='text' and @value='B']

All the textbox and all the buttons →

//input[@type='text' or @type='button']

1st textbox and 1st button →

// input [@value = 'A']

Only selected check box →

// input [@checked]

// input [@type = 'checkbox' and @checked]

SYNTAX for checkbox ⇒ tag [@Attribute Name]

Only unselected check box →

// input [not (@checked) and @type = "checkbox"]

// input [not(@checked)]

xpath by Text function

i) In xpath we can also use text

ii) The SYNTAX is :- //.tag [text() = 'value']

iii) html code of a login button is :-

 Login

xpath is → // div [text () = 'Login']

or

// div [. = 'Login']

Ex → 2 → html code: <td id = "header container" class = "header">
Please identify yourself </td>

xpath is : //td [text() = "Please identify yourself"]

xpath by contains function:

i) If the text value is partially changing then we can use contains function on the xpath.

ii) The SYNTAX is :- //tag [contains (text(), 'value')]

Ex→ html code: <nobr> actiTime 2020 Online </nobr>

xpath: //nobr [contains ((text(), 'actiTime')]

Q) How do you identify the element without using partial link text.

→ By using xpath by contains function.

Ex→ html code: <a..> Inbox (7)

xpath :- // a [contains (text(), 'Inbox')]

Note:-

We can use contains function for Attribute also

The SYNTAX is : // tag [contains (@AN, 'AV')]

html code: <img class = "fb_logo_sith img"
src = "https://static.xx.fbcdn.net/rsrcs.php/yS/r/
3F55bl3UtHW2.svg" >

xpath: //img [contains (class, 'logo')]

xpath by Traversing :

i) navigating from one element to another element is called as traversing

ii) There are 2 types of traversing
   1) forward traversing
   2) backward traversing

iii) navigating from parent element to child element is called as Forward traversing

iv) navigating from child element to parent element is called as backward traversing

Ex for Forward traversing
(from html to selenium)

xpath : //tr[2]/td[1]

or

//td[text()='selenium']

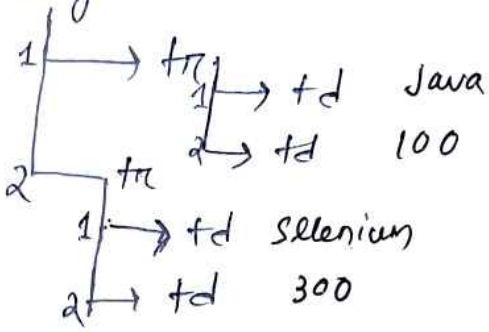
Sample HTML code :

```
<table border="1">
<tr>
<td> java </td>
<td> 100 </td>
</tr>
<tr>
<td> Selenium </td>
<td> 300 </td>
</tr>
</table>
```

l Tree:

html
↳ body
↳ table
↳ tbody

java	100
selenium	300



x for backward traversing:

~~for~~ (from selenium to html)

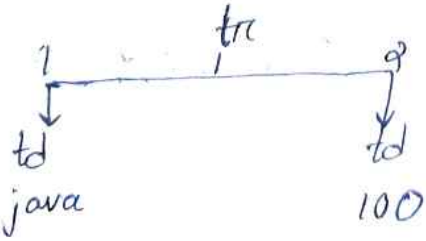
//td [text() = 'Selenium'] /../../../../..

Independent - Dependent xpath:

i) if the value is changing completely then we handle it using independent - dependent xpath.

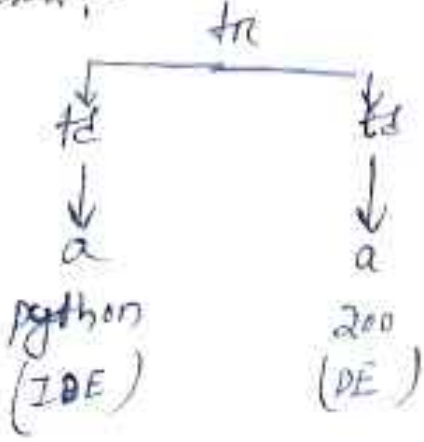
ii) We always starts from independent element and ends with dependent element.

Ex>

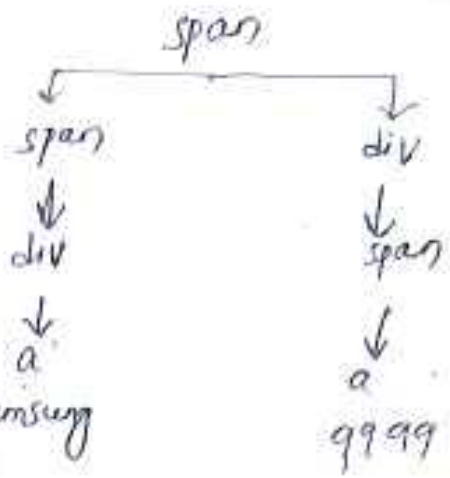


// td [text() = 'java'] /../ td[2]

Ex-2 :-



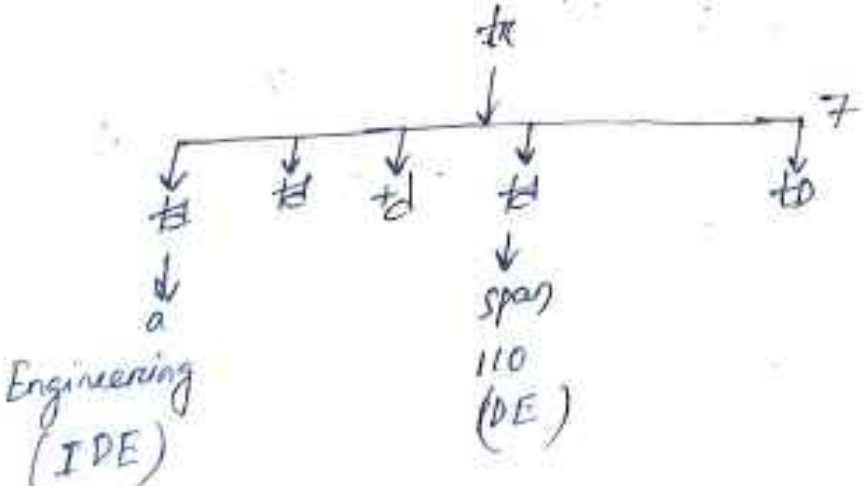
Ex-3 :-



2) //a [text() = 'python'] /../../td[2]/a

3) //a [text() = 'samsung'] /../../../div/span/a

Q// Write an xpath to identify the price of engineering type of work present in the active under settings dropdown.



//a [text() = 'engineering'] /../../td[4]/span

1) Write an xpath to identify the states for manufacturing type of work presenting in actitime

2) Write an xpath to identify the category for testing type of work

3) Write an xpath to identify the set by default link for support type of work.

steps to write xpath without html tree:

i) identify the independent and dependent element

ii) Inspect the independent element and write the xpath by looking at the html code.

iii) move the mouse pointer to upward direction step by step till it ~~house~~ highlights both independent and dependent element.
will be common parent.

iv) At the both of common parent next to the above xpath using /..

v) use down arrow key & navigate from common parent to dependent element.

vi) ~~And it~~ add it's path next to the common parent

4) Write an xpath to identify the version of C# language present in the download page of selenium under language bindings.

5) Write an xpath to identify the changlog for ruby language

6) Write an xpath to identify for API docs link for python language

---

1) Write an xpath ~~for~~ to identify the stickes for manufacturing type of work in actitime.

→ //a[text()='manufacturing']/../../td[2]

2) Write an xpath to identify the category for testing type of work.

→ //a[text()='testing']/../../td[3]

3) Write an xpath to identify the set by default link for support type of work.

→ //a[text()='support']/../../td[5]/a

4) Write an xpath to identify the version of C# language present in the download page of selenium under Language binding

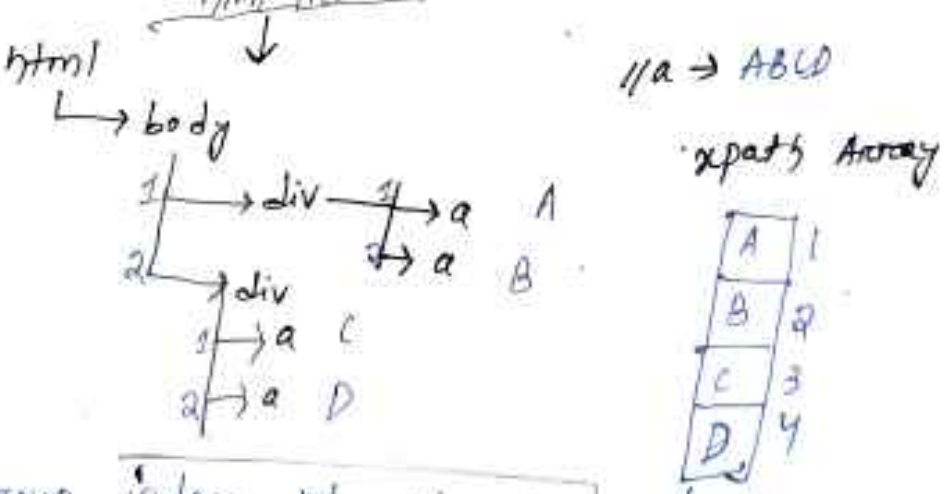
→ //p[text()='c#']/../p[2]/a

5) //p[text()='Ruby']/../p[3]/a

6) //p[text()='Python']/../p[4]/a

xpath By Group Index:

i) While writing xpath expression we can write the expression within the brace and we can specify the index outside the brace which is called as group index.



ii) When we use group index 1st it will execute the expression present inside the brace,

In this example 1st it will execute //a and it will store the matching elements that is ABCD links into xpath Array. where index starts from 1.

iii) Then it will identify the matching elements based on the index which is specified out side the bra

In this example it maches with 1st Link A

(//a)[1] → A (Maches with 1st link only)

(//a)[2] → B (" " 2nd Link)

(//a)[3] → C (" " 3rd Link)

(//a)[4] → D

(//a)[last()] → D (Maches with last link only)

//a[1] → AC (Maches with all the Links having index 1)

(//a[1])[1] → A

(//a[1])[2] → C

//a[2] → BD (Maches with all the link having Index 2

(//a[2])[1] → B

(//a[2])[2] → D

(//a)[1] | (//a)[last()] → A | D → AD

↓

Maches with 1st and last link

Q// Write an xpath to identify the price of 2nd selenium book ?

selenium	100
selenium	300

(//td[text()='selenium'])[2]/../td[2]

1) Write an xpath to identify the price of HRX by hirtik roshon tshirt present in Myntra.

(//h3[text()='HRX by Hrithik Roshan'])[1]/../../div/span[1]/span[1]

2) Write an xpath to identify the prize Libas kurta present in myntra under women section.

(//h3[text()='Libas'])[1]/../../div/span[1]/span[1]

3) Write an xpath to identify the price of H & M jeans present under kids section

(//h3[text()='H&M'])[1]/../../div/span[2]

4) Write an xpath to identify the price of samsung s23 ultra present in amazon.in. → (contains) Q.

5) Write an xpath to identify the price of Apple watch SE present in flipkart

(// div [contains (@class, '_2B099V')] [2] / a[2] / div[2] / div

Q// what are the frequently used locators in selenium. !

1) id ( )
2) name ( )
3) linkText ( )
4) xpath ( )

Imp) Write a script to remove the text present in the textbox.

```
public class RemoveText {
static {
System.setProperty ("webdriver.chrome.driver", "./drive/chromedriver.exe");
}
P.S.V.M (S [] args) throws InterruptedException
WebDriver driver = new ChromeDriver ();
driver.get ("files // C:/users/ajits/Desktop/Demo2.html");
Thread.sleep (3000);
driver.findElement (By.xpath ("// input [@type = 'text'] [2]")
                    .clear ();
}
}
```

// write a script to check whether facebook logo is displayed or not.

```
class VerifyLogo {
static {

psvm(s[] args){
WebDriver driver = new ChromeDrive();
driver.get("https://www.facebook.com/");
boolean logo = driver.findElement(By.xpath("//img[@alt='facebook']")).isdisplayed();
if (logo == true) {
s.o.println("logo is displayed and pass");
}
else {
s.o.println("logo is displayed and fail");
}
```

// Write a script to check whether login button present in facebook is enable or not.

```
public class loginButton {
static {
System.setProperty("webdriver.chrome.driver", "./drive/
                                    chromedriver.exe");
}
  P S V m(S [] args) {
  WebDriver driver = new ChromeDriver();
  driver.get("https://www.facebook.com/");
boolean b enable = driver.findElement(By.name("login")).
                                    isEnabled();
    if (enable == true) {
  S.o.pln("login button is enabled and pass");
    }
    else {
  S.o.pln("Login button is not enabled and fail");
    }
  driver.close();
  }
}
```

Q// Write a script to print the states of the checkbox present in Actitime

```
class statusCheck {
static {
    path =
}
    p s v m (s [ ] args) throws InterruptedException {
    WebDriver d = new ChromeDriver ();
    d. get ("https://demo. actitime com /");
boolean select = d. findElement (By. nam ("username")). isSelected ();
    Thread - sleep ( 3000);
if (select = = true) {
s-o pln ( "checkbox is selected and pass");
}
else {
    s-o pln ( " checkbox is not selected and fail");
}
d. close ();
}
}
```

Q/ Write a script to print the text of forgot your password link present in actitime.

```
class PrintText {
static {
System.setProperty("webdriver.chrome.driver", "./drivers/chromedriver.exe");
}
p s v m (s [] args) {
WebDriver driver = new ChromeDriver();
driver.get("https://Demo.actitime.com/");
String text = driver.findElement(By.id("toPasswordRecoveryPageLink")).getText();
s.o.pln(text);
driver.close();
}
}
```

Q/ WAS to get the tagname of forgot your password link present in actitime.

```
String text = driver.findElement(By.id("toPasswordRecoveryPageLink")).getTagName();
s.o.pln(text);
```

Q/ WAS to print the value of href attribute for the FYP link present in actitime.

```
String text = driver.findElement(By.id("toPasswordRecoveryPageLink")).getAttribute("href");
s.o.pln(text);
```

~~// Write a script to g~~

// How do you click on a button without using click () method.

→ By using submit () method.

Submit () method works only if type = 'submit' attribute is present

Write a script to click on the login button present in facebook without using click method.

```
public class withoutClick {
static {
System.setProperty("webdriver.chrome.driver", "./driver/chromedriver.exe");
}
    p s v m (s [] args) {
    WebDriver driver = new ChromeDriver();
     driver.get ("https://www.facebook.com/");
driver.findElement(By.name("login")).submit();
    driver.close();
  }
}
```

imp

Write a script to print height and width of the email textbox present in the facebook?

```
class PrintHeightAndWidth {
static {
System.setProperty("webdriver.chrome.driver","./driver/chromedriver.exe");
}
  P S V m(S[] args) {
    WebDriver driver = new ChromeDriver();
    driver.get("https://www.facebook.com/");
  WebElement emailTbx = driver.findElement(By.id("email"));
  int height = emailTbx.getSize().getHeight();
  int width = emailTbx.getSize().getWidth();
    S.o.pln("Height:"+height);
    S.o.pln("Width:"+width);
  driver.close();
  }
}
```

// WAS to check whether height and width of username & password textbox present in actitime is equal or not.

```
public class Heeight and width HW
    p s v m (s [] args) {
    WebDriver ~~new~~ driver = new ChromeDriver ();
     driver.get ("https://demo.actitime.com/login.do");
WebElement uname = drive findElement (By.name("username"));
 int h1 = uname.getSize().getHeight();
 int w1 = uname.getSize().get ~~Height~~ Width();
WebElement pwd = driver.findElement(By.name("pwd"));
 int h2 = pwd.getsize().get Height();
 int w2 = pwd.getsize().get width();
  if (h1 == h2 && w1 = w2) {
    s.o.pln("H and W of textbox is equal and pass");
  }
  else {
  s.o.pln(" H and W of textbox is not equeal and
                                    fail");
  }
  driver.close();
 }
}
```

// WAS to print x & y axis of login button present in facebook.

```
Class xandYaxis
    P S V M (S [] args) {
WebDriver driver = new ChromeDriver();
driver.get("https://www.facebook.com/");
int xaxis = driver.findElement(By.name("login")).getLocation()
                                                  .getX();
int yaxis = driver.findElement(By.name("login")).getLocation()
                                                  .getY();
    S.o.pln(xaxis);
    S.o.pln(yaxis);
    driver.close();
    }
}
```

// WAS to check whether username & password textbox present in actitime is properly ~~align~~ align or not

```
Class VerifyAlignment {
static {

}

    P S V M (S [] args) {
WebDriver driver = new ChromeDriver();
driver.get("https://demo.actitime.com/");
int x1 = driver.findElement(By.id("username")).getLocation().
                                                  getX();
int x2 = driver.findElement(By.name("pwd")).getLocation()
                                                  .getX();
```

```
if (x1 == x2) {
S.o.pln("username and password textbox are properly aligned
                                        and pass");
}
else {
S.o.pln("username and password textbox are not
                                properly aligned and fail");
}
driver.close();
}
}
```

Note:-

By using getRect() method we can get x and y axis along with height and width element.

Automate the following scenario

1) open the browser
2) Enter the url. (demo.actitime.com)
3) Enter admin in the username textbox.
4) Enter manager in password textbox
5) click on login button.

Ans is in next Page →

```
class Actitime {
static {
System.setProperty("webdriver.chrome.driver", "./driver/
                                        chromedriver.exe");
}
    P S V m(S[] args) {
    WebDriver d = new ChromeDriver();
    d.get("https://demo.actitime.com/login.do");
d.findElement(By.name("username")).sendKeys("admin");
d.findElement(By.name("pwd")).sendKeys("manager");
d.findElement(By.xpath("//div[text()='Login']")).
                                        click();
    driver.close();
    }
}
```

// WAS. to print the colour of forgotten password link present in facebook.

```
public class PrintColour {
static {
System.setProperty("webdriver.chrome.driver", "./driver/chromedriver.exe");
}
P S V M (String[] args) {
WebDriver d = new ChromeDriver();
d.get("https://www.facebook.com/");
String colour = d.findElement(By.linkText("forgotten password?")).getCssValue("color");
S.o.println(colour);
driver.close();
}
}
```

// WAS to print the font-size of F-P link present in facebook

```
String fontsize = d.findElement(By.xpath("//div[text()='forgotten password?']")).getCssValue("font-size");
S.o.println(fontsize);
```

// WAS to print

// WAS to copy the text present in one textbox and paste it into another textbox

```
class copyPast {
static {
System.setProperty ("                    ");
}
P S V m(S[] args){
WebDriver d = new ChromDriver
d.get ("file:///C:users/ajits/Desktop/Demo1.html");
d.get find Element "(By.xpath((//input[@type='text'])[1]')).
                                    sendkeys (keys.CONTROL + "ac");
d.get findElement (By.xpath("(//input[@type='text'])[2]")).
                                    SendKeys (keys.CONTROL + "av");

driver.close ();
}
}
```

Handling Multiple Elements :-

i) In order to handle multiple elements we use findElements() method.

ii) The return type of findElements is list of WebElements.

iii) List should be imported from java.util package

iv) if the locators are matching with multiple elements then it will return the address of all the matching elements.

v) if the locators are not matching with any of the element then findElements method will returns EmptyList

Ex→

```
import java.util.List;
    class HandlingMultipleElements {
static {
    System.setProperty("  ...  ");
}
    P S V M (S [] args) {
WebDriver d = new ChromeDriver();
d.get("file

List<WebElement> allLinks = d.findElements(By.tagName("a"));
int count = allLinks.size();
S.o.pln("count");
WebElement link = allLinks.get(0);
String text = link.getText();
S.o.pln(text);
}
}
```

Vvimp

WAS to print all the links present in amazon.in

```
import java.util.List;
class AmazonLink {
static {

}

P S V M (S [] args) {
WebDriver d = new ChromeDriver();
d.get("https://www.amazon.in");
List<WebElement> allLinks = d.findElements(By.xpath("//a"));
int count = allLinks.size();
S.o.pln(count);
for(int i=0; i<count; i++){
String text = allLinks.get(i).getText();
    S.o.pln(text);
}
d.close();
}
}
```

Note:

In order to take the input or url from the user we have to use scanner class as shown below.

Ex→ import java.util.Scanner;

```
class Assign1 {
static {"para"
}
P S V M (S [ ] args) {
S.o.pln ("Enter the url");
Scanner sc = new Scanner (System.in);
String url = sc.nextLine();
WebDriver d = new ChromeDriver();
d.get(url);
```

* VVimp

Auto Suggestion

Q// Automate the following scenarios

i) open the browser

ii) Go to Google.com

iii) type java in search textbox.

iv) capture all the auto suggestion & print the count of auto suggestion

v) Print the text of autosuggestion.

vi) Select the 5th auto suggestion

```
class AutoSuggestion {
static {
System.setProperty("path");
}
p s v m (s [] args) throws InterruptedException {
// open the browser
WebDriver driver = new ChromeDriver();
// go to google.com
driver.get("https://www.google.com/");
// type java in search textbox
driver.findElement(By.name("q")).sendKeys("java");
Thread.sleep(2000);
// capture All the Auto suggestion and print the count
List<WebElement> allSugg = driver.findElements(By.xpath
("//span[contains(text(),'java')]"));
int count = allSugg.size();
S.O.println(count);
```

```
// print the text of Auto suggestion
for (int i=0; i<count; i++){
  String text = allsugg.get(i).getText();
      s.o.pln(text);
  }
  allsugg.get(0).click();
  driver.close();
  }
}
```

vvimi

// WAS to navigate google.com and search for selenium and capture all the auto suggestion and print it on the console.
- then select the last Auto suggestion.

```
import java.util.List;
  class AutoSelenium {
  static {
  System.setProperty("path");
  }
    p s v m(s[] args) throws InterruptedException {
  WebDriver d = new chromeDriver();
  d.get("https://www.google.co.in/");
- ~~List<WebElement> as~~ d.findElement(By.name
                          ("q")).sendKeys("selenium");
    Thread.sleep(3000);
List<WebElement> as = d.findElements(By.xpath("//span
                      [contains(text(),'selenium')]"));
```

```
for (int i=0; i<as.size(); i++) {
    String text = as.get(i).getText();
    }
        S.o.ptln(text);
    as.get(as.size()-1).click();
    d.close();
  }
}
```

Q) Difference betn findElement & findElements :

findElement	findElements
i) The returntype of findElement is WebElement	i) The returntype of findElements is List of WebElement.
ii) If the locator is not maching it will throw no such Element Exception	ii) returns EmptyList
iii) If locators are maching with multiple elements, it returns the address of 1st maching elements	iii) It returns all the maching elements.
iv) used to handle single elements	iv) used to handle multiple elements.

@imp // WAS to navigate to BBC com and capture top 4 and latest bussiness news and print it on the console.

```
import java.util.List;
class Bbc {
static {
System.setProperty("webdriver.chrome.driver", "./driver/chromedriver.exe");
}
psvm(s[] args) throws InterruptedException {
WebDriver d = new ChromeDriver();
d.get("https://www.bbc.com/");
List<WebElement> as = d.findElements(By.xpath("//a[@rev='most-popular/link']"));
for(int i=0; i<as.size(); i++) {
String news = as.get(i).getText();
s.o.println(news);
}
d.close();
}
}
```

Synchronigation :

i) The process of maching the selenium speed with the application is called as Synchronigation.

ii) Most of the time selenium is faster than the application

Because of this reason we may not get expected result or we may get Exception such NSEException.

Ex →

```
class Synchronigation {
    static {
System.setProperty("path");
    }
    p s v m (s[] args) {
WebDriver d = new ChromeDriver();
d.get("https://www.actitime.com/");
d.findElement(By.id("username")).sendKeys("admin");
d.findElement(By.name("pwd")).sendKeys("manager");
d.findElement(By.xpath("//div[text()='Log in']")).click();
  Thread.sleep(5000);
d.findElement(By.id("logoutLink")).click();
    }
}
```

Implicit Wait:

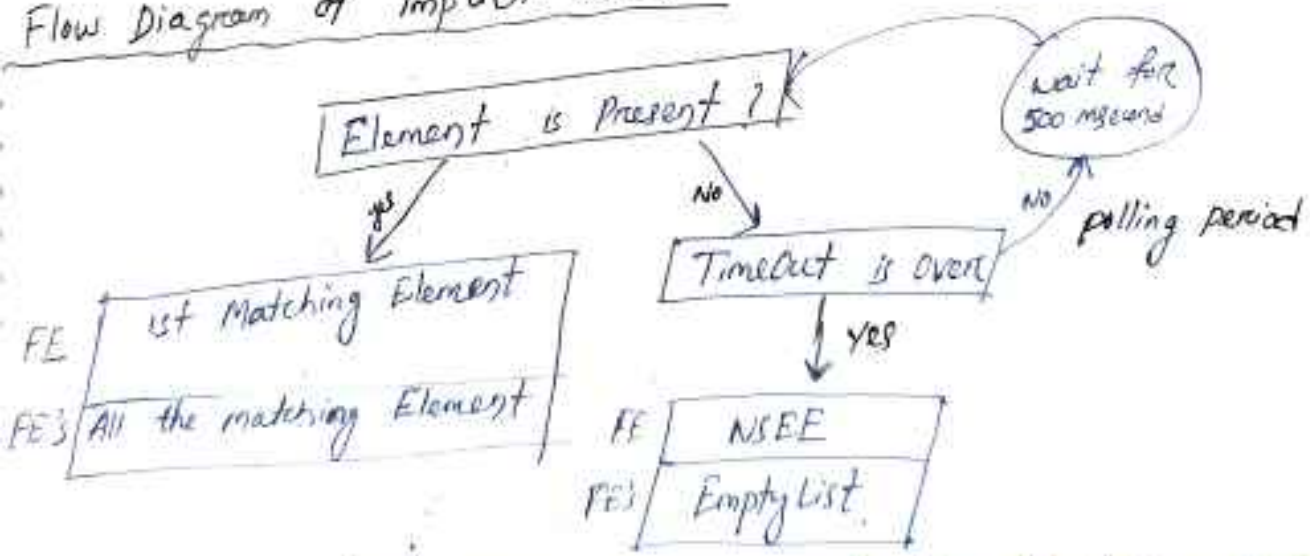
In selenium there are different ways to synchronize the script one of the frequently used option is implicitly wait.

SYNTAX IS: driver.manage().timeouts().implicitlyWait(10, TimeUnit.seconds);

it takes two arguments
1) time duration (Long)
2) TimeUnit

i) TimeUnit can be days, hours, minutes, seconds ... etc.
ii) The specified duration is used only by findElement and findElements statement means implicit wait will work only for findElement & findElements, not for any other function or methods.

Flow Diagram of Implicit wait:



→ when the control comes to any of the findElement and findElements statement. it will check wether the element is present or not.
→ if the element is present then findElement method returns 1st matching element. Whereas findElements returns all the matching elements.

→ if the specified element is not present then it will check for the time out.

→ if the time is over then findElement throws NSEException whereas findElements method returns Empty List

→ if the time is not over it wait for 500ms (half second) which is called as polling period

→ then it will continues to check whether the element is present or not.

Ex→

```
public class Implicitwait {
    static {
System.setProperty("webdriver.chrome.driver","./drivers/chromedriver.exe");
}
    P S V M (S [] args) {
  WebDriver d = new ChromeDriver();
    d.manage().timeouts().implicitlyWait(10, TimeUnit.seconds);
    d.get("https://www.actitime.com/");
  d.findElement(By.id("username")).sendKeys("admin");
d.findElement(By.name("pwd")).sendKeys("manager");
d.findElement(By.xpath("//div[text()='Log in']")).click();
  d.findElement(By.id("logoutLink")).click();
    d.close();
  }
}
```

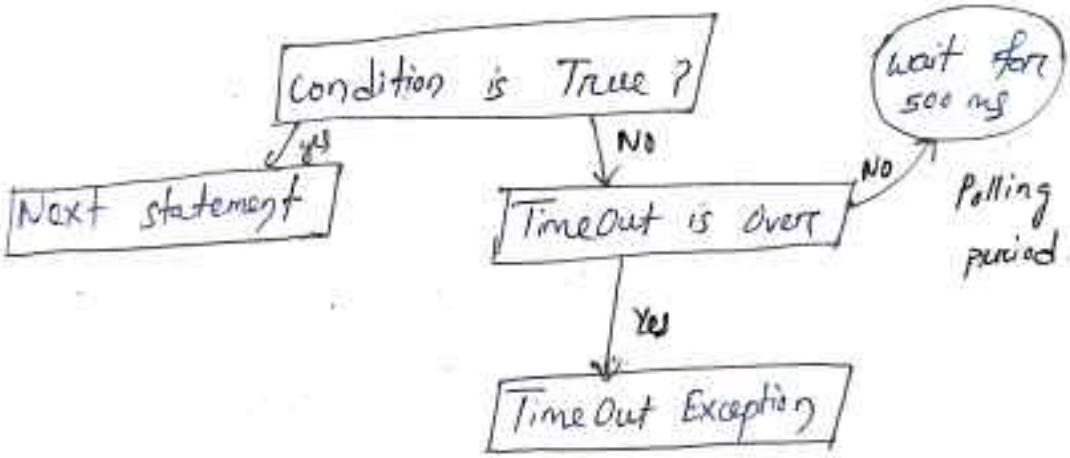
Explicit Wait:

In order to handle the synchronization of any method we can use Explicit wait

WebDriver wait itself is called as Explicit wait because we have to specify the waiting condition explicitly

SYNTAX: WebDriverWait wait = new WebDriverWait(driver, 10);
wait.until(ExpectedConditions.titleIs("Enter Title"));

Flow Diagram:



→ When the control comes to wait.until statement it will check the specific condition.

→ if the condition is true it will go to the next statement, if the condition is false it will check for the timeout

→ if the Timeout is over it will throw timeout exception, else it will wait for 500 ms and it will continues to check the condition.

Note:-

i) In the above flow Diagram titleIs ~~the~~ is the condition.

ii) By using Explicit wait we can handle the synchronization of any method but only one at a time.

```
Ex→   class ExplicitWait {
        static {
          System.setProperty("path")
        }
        p s v m (s [] args) {
      WebDriver d = new ChromeDriver();
      ~~driver~~.get("https://demo.actitime.com/");
d.findElement(By.id("username")).sendKeys("admin");
d.findElement(By.name("pwd")).sendKeys("manager");
d.findElement(By.xpath("//div[text()='Log in']")).click();
WebDriverWait wait = new WebDriverWait(driver, 10);
wait.until(ExpectedConditions.titleIs("actitime - Enter Time trac
// we can use also
wait.until(ExpectedConditions.titleContains("Enter");
      String title = d.getTitle();
      S.O.println(title);
    }
}
```

Note:-

Expected conditions is an Interface and ~~Expected conditions~~ Expected Conditions is a class

// can we specify the implicit wait statement multiple times in selenium script. ?

→ Yes

// Is it necessary to specify implicit wait statement multiple times ?

→ No

imp// can we handle the synchronization of findElement method using Explicit wait ?

→ Yes

Ex → WebDriver wait = new WebDriverWait(driver, 10);
wait.until(ExpectedConditions.presenceOfElementLocated
(By.id("logoutlink")));

or

wait until(ExpectedConditions.visibilityOfElementLocated
(By.id("logoutLink")));

Custom Wait:

The process of handling the synchronigation of the automation script by writing our own java code is called custom wait.

```
Ex → class customWait {
 static {
System.setProperty ("path");
 }
    p s v m (S [] args) {
WebDriver d = new ChromeDriver ();
 d.get ("https://demo.actitime.com/");
d.findElement (By.id ("username")).sendKeys ("admin");
d.findElement (By.name ("pwd")).sendKeys ("manager");
d.findElement (By.xpath ("//div[text()='Log in']")).click();
  int i=0;
 while (i<=100) {
   try {
    d.findElement (By.id ("logoutLink")).click();
     break;
   }
   catch (NoSuchElementException e) {
    i++;
   }
 }
}
}
```

ifference bet'n Explicit and Implicit wait. Imp very very

mplict Wait	Explict Wait
) We do not specify the waiting condition explicitly	i) We should specify the waiting condition explicitly
i) We can handle the Synchronigation of only findElement & findElements method.	ii) We can handle the synchronigation of any method but only one at a time.
iii) After the duration we get NoSuchElement Exception	iii) After the duration time is over we get TimeOut Exception.
iv) Time Duration can be days, hours, minutes seconds .. etc	iv) Time Duration will be only seconds.

Q/ What are the different ways of synchronigation?

→ Implict wait, Explict wait, custom wait and Thread.sleep, fluent wait

Handling Listbox :-

i) In Listbox there are two types

1) Single select Listbox (Dropdown)
2) Multi select Listbox

ii) Whenever listbox is created using select tag and content of the listbox is created using option tag they to handle that listbox we use select class of selenium.

iii) Select class should be imported from
org.openqa.selenium.support.ui package

iv) Select class as a parameterized constructor where it takes an argument of type webElement (address of the listbox)

v) In order to select the required option present in the listbox we can use any one of the following methods.

Methods

1) selectByIndex(int)
2) selectByValue(string)
3) SelectByVisibleText(string)
4) deselectByIndex
5) deselectByValue
6) deselectByVisibleText
7) deselectAll
8) isMultiple()
9) getFirstSelectedOption
10) getAllSelectedOptions
11) getOptions
12) getWrappedElement

Ex→ class HandlingListbox {
static {
System.setProperty ("path");
}

P S V M (S [] args) throws InterruptedException {
WebDriver driver = new ChromeDriver ();
driver.get ("https://www.facebook.com/");
driver.findElement (By.LinkText ("create new account")).click();
Thread.sleep (3000);
WebElement monthListbox = driver.findElement (By.id ("month"));
Select s = new Select (monthListbox);
s.selectByIndex (7);

s.selectByValue ("12");

s.selectByVisibleText ("Feb");
}

}

// wrote a script to select your DOB present in facebook after clicking crete new account button

class DropdownSingle {

static {
System.setProperty ("webdriver.chrome.driver", "./driver/chromedriver.exe");
}

```
P S V m (S [ ] args) throws InterruptedException {
WebDriver d = new ChromeDriver ( ) {
  d. get ( " https :// www facebook. com / ");
d. findElement ( By. linkText ("Create new account ")). click ();
Thread. sleep (3000);

WebElement dayLb = d. findElement ( By. id (" day "));
  Select s = new select ( dayLB);
  s. selectBy Index (0);

WebElement monthLb = d. findElement ( By. id ("month"));
  Select s1 = new select ( monthLB);
  S1. selectBy Value ("4 ");

WebElement yearLB = d. find Element ( By. id ("year "));
  Select s2 = new select (yearLB);
    S2. selectBy VisibleText ("2000 ");

  }
 }
}
```

Multi Select Listbox:

Simple HTML code to create Multi Select List box;

```
MTR: <br>
<select id = "mtr" multiple>
<option value = "i"> idly </option>
<option value = "d"> dosa </option>
<option value = "v"> vada </option>
<option value = "p"> poori </option>
<option value = "g"> gulugula </option>
<option value = "p"> pongal </option>
<option value = "D"> dahibara </option>
<option value = "k"> khara bath </option>
<option value = "i"> idly </option>
</select> <br>
CheckPost : <br>
<select id = "cp" multiple>
<option value = "l" selected> Lemon Rice </option>
<option value = "p"> puliyogare </option>
<option value = "b" selected> bisi bele bath </option>
<option value = "c"> chapati </option>
<option value = "p"> parotha </option>
<option value = "v"> Vegbiriyani </option>
<option value = "po"> poha </option>
<option value = "v"> Vadapav </option>
</select>
```

Ex→ class MultiselectListBox

```
static {
System.setProperty("webdriver.chrome.drive","/driver/
                                    chromedriver.exe");
}
  P S V m (S [] args) throws InterruptedException {
  WebDriver d = new ChromeDriver
d.get("file:///C:/Users/
WebElement mtrListBox = d.findElement(By.id("mtr"));
Select s = new Select(mtrListbox);
  s.selectByIndex(0);
  s.selectByValue("2");
  s.selectByVisibleText("Vada");
  s.deselectByValue("v");
 s.deselectByIndex(1);
 s.deselectByVisibleText("Idly");
 s.deselectAll();   // for deselect all we can use this
                                          method
  s.opln(s.isMultiple());
  }
}
```

// WAS to print firstSelected option present in checkpost Listbox

```
class PrintFirstSelected {
static {
}
  p s v m (s [] args) {
WebDriver d = new ChromeDriver
d.get("
WebElement cplistbox = d.findElement(By.id("cp"));
Select s = new Select(cplistbox);
// use String text = s.getFirstSelectedOption().getText();
                                    // code optimization
s.opln(text);
```

we can write in this way also =>

```
WebElement fsoption = s.getFirstSelectedOption();
String text = fsoption.getText();
s.opln(text);
```

Print all the selected option present in checkPost Listbox:

```
class PrintAllSelected {
static {
System.setProperty("path");
}
P S V m(S[]args){
WebDriver d = new ChromeDriver();
d.get(
WebElement cplistbox = d.findElement(By.id("cp"));
Select s = new Select(cplistbox);
List<WebElement> allSOption = s.getAllSelectedOptions();
int count = allsoption.size();
s.o.pln(count);
for(int i = 0; i<count; i++){
String text = allSOption.get(i).getText();
s.o pln(text);
}
d.close();
}
```

Q// WAS to print all the options present in MTR listbox :

```
        class PrintAlloptions {
  static {
  System.setProperty ("path");
}
      p s v m (s [] args) {
  WebDriver d = new ChromeDriver();
  d.get(
WebElement mtrListbox = d.findElement(By.id("mtr"));
  Select s = new Select (mtrListbox);
List<WebElement> alloptions = s.getOptions();
  for (int i = 0; i< alloptions.size(); i++) {
    String text = alloptions.get(i).getText();
      s.o.println(text);
  }
  d.close();
 }
}
```

Q// WAS to to select all the option present in MTR
imp** listbox and deselect them in reverse order.

```
class selectAllOptions {
  static {
System.setProperty ("path");
}
  p s v m (S[] args) throws InterruptedException {
  WebDriver d = new ChromeDriver ();
  d.get (
WebElement mtrLb = d.findElement (By.id ("mtr"));
Select s = new Select (mtrLb);
List<WebElement> alloptions = s.getOptions();
    int count = alloptions.size();
    for (int i=0; i<count; i++) {
        Thread.sleep (500);
        s.selectByIndex(i);
    }
  for (int i = count-1; i>=0; i--) {
        Thread.sleep(500);
        s.deselectByIndex (i);
    }
  }
```

1) // WAS to print All the options present in MTR Listbox without duplicate

2) // WAS to print only duplicate options present in MTR Listbox

3) // WAS to print all the option in alphabetical order

4) // WAS to print all the options present in MTR Listbox without using for loop.

```
class PrintAllOption {
static {
System.setProperty("path");
}
    p s v m (S[] args) {
    WebDriver d = new ChromeDriver();
    d.get(
  WebElement mtrLb = d.findElement(By.id("mtr"));
    Select s = new Select(mtrLb);
    String text = s.getWrappedElement.getText();
      s.o.println(text);
      d.close();
    }
  }
```

Handling Popup :-

i) In selenium depending on popup we write different types of code to perform action on the popup

ii) Popups are generally categorized as follows

1) Alert or javascript Popup
2) Hidden division or calender popup
3) File upload popup
4) File Download popup
5) Print popup
6) Notification Popup
7) child window & child browser Popup

Alert Popup (javascript or conformation)

Characteristic :

i) We can not move this popup
ii) we can not inspect this popup
iii) This popup will be having ok button (alert) or it contains ok & cancel button (conformation)
iv) It will be present below the address bar in the middle section of the browser

Solution :

We handle this alert popup by using

driver.switchTo().alert(). statement

ii) After switching to alert popup we can use

1) accept() → for clicking on ok button
2) dismiss() → for clicking on cancel button
3) getText() → To get the text present on the popup.
4) sendKeys() → To type the text on the popup

iii) All the above code will works on conformation popup also

Ex →

```
class AlertPopup {
    static {
    }
    p s v m (s[] args) { throws InterruptedException {
    WebDriver driver = new ChromeDriver();
driver.get("https://demo.automationtesting.in/Alerts.html");
driver.findElement(By.xpath("//button[@class='btn btn-danger']")).
                                                    click();
    Thread.sleep(2000);
    Alert a = driver.switchTo().alert();
    String text = a.getText();
    a.accept();
    S.o.println(text);
    driver.close();
    }
}
```

Note :-

if the popup is alert then their will not be any difference bet'n accept & dismiss, both of them clicks on OKK button only

Hidden division Popup (Calender Popup)

Characteristic :-

i) We can not move this popup

ii) We can inspect this popup

Solution :-

We handle this popup by using findElement method

Ex →

```
class HiddenDivisionPopup {
    static {
    }
    p s v m (s [] args) throws InterruptedException {
    WebDriver d = new ChromeDriver();
    d.get ("https://www.flipkart.com/");
        Thread.Sleep (2000);
    d.findElement (By.xpath("(//button)[2]")).click();
    }
}
```

Automate the following scenario

1) open the browser
2) Enter the url (careinsurance.com)
3) Enter the policy number as 123
4) click on DOB
5) select your DOB 11) Enter the contact number as 9845098450
6) click on lets renew button.

```
class careInsurance {
PSVm (S[] args) {
WebDriver d = new chromeDriver()
d.get("https://www.careinsurance.com/#/renew/");
d.findElement(By.id("policy number")).sendKeys("123");
d.findElement(By.id("dob")).click();
WebElement sm = d.findElement(By.xpath("//select[@class
='ui-datepicker-month']"));
Select s = new Select(sm);
s.selectByIndex(3);
WebElement sy = d.FE(By.xpath("//select[@class='ui-datepicker-
year']"));
Select s1 = new select(sy);
s1.selectByValue("2000");
d.FE(By.xpath("//a[text()='1']")).click();
d.FE(By.id("alternative_number")).sendkeys("8327759027");
d.FE(By.id("renew_Policy_submit")).click();
}
}
```

File Upload Popup:

Characteristic :

i) We can not inspect this popup

ii) We can move this Popup

iii) This popup will be having open & cancel button

iv) Title of the popup will be open as file Upload

Solution :

To handle file upload popup we specify the absolute path of the file as an argument for sendkeys method

Note :

```
<input type="file" id="cv" />
```

Save this as naukri.html

Ex →

```
class FileuploadPopup {
    static {
    }
    p s v m(S[]args) throws InterruptedException {
        WebDriver d = new ChromeDriver();
        d.get("file:///C:/Users/ajits/Desktop/naukri.html");
        Thread.sleep(3000);
        File f = new File("./data/Resume.docx");
        String abPath = f.getAbsolutePath();
        d.findElement(By.id("cv")).sendKeys(abPath);
    }
}
```

Note :

In order to make the code generic we use relative path of the file but sendKeys method will take only Absolute path of the file. So we use getAbsolute path method of File class

File Download Popup :

characteristic : (w.r.t to older version of firefox)

i) We can move this popup.

ii) We can not inspect this popup.

iii) It will be having OK & cancel button along with open with and save file radio button.

Solution :

By using Robot class

Note :

1) When we click on any type of download options in ~~Chrome~~ chrome browser or newer version of firefox browser it will not display file download popup instead of that it will start downloading the file directly

Hence we don't need to handle file download popup.

Robot class :

i) Robot class is a java class present in java.awt package (abstract window toolkit)

ii) whenever we want to perform any keyboard operations in widows we go for robot class.

iii) Here we commonly use two methods :
1) keypress
2) key Release

iv) Robot class works almost similar to sendkeys method.

Q WAP to type @Qsp in the Notepad (or) WAP to demonstrate Robot class

```
class DemoRobotClass {
P S V m (S [] args) throws AWTException, IOException {
Runtime.getRuntime().exec("notepad");
Robot r = new Robot();
r.KeyPress(KeyEvent.VK_SHIFT);
r.KeyPress(KeyEvent.VK_Q);
r.KeyRelease(KeyEvent.VK_SHIFT);
r.KeyPress(KeyEvent.VK_S);
r.KeyPress(KeyEvent.VK_P);
}
}
```

Print Popup :

Characteristic : (w.r.t firefox driver)

i) We can not move this Popup

ii) We can not inspect this popup

iii) This Popup will be having print & cancel button

Solution :

We handle this Popup by using Robot class

Q// Automate the following scenario

i) open the firefox browser

ii) Go to download page of selenium

iii) Press control + P to get the print popup

iv) Press up Arrow key 1 time.

v) Take the 1st 2 pages of the printout

vi) Press tab to go to the pages option.

vii) Press down Arrow key 4 time for selecting custom

viii) Press 1-2 for taking 1st two pages

ix) Click on print button by pressing Enter.

x) enter any file name & save.

Ans →

```
class PrintPopup {{
    static {

System.setProperty("webdriver.gecko.driver", "./drivers
                    /geckodriver.exe");

    }
```

```
P S V M (S[] args) throws AWTException, InterruptedException {
WebDriver d = new FirefoxDriver();
d.get("https://www.selenium.dev/downloads/");
    Robot r = new Robot();
    Thread.sleep(2000);
    r.keyPress(KeyEvent.VK_CONTROL);
    r.keyPress(KeyEvent.VK_P);
    r.keyRelease(KeyEvent.VK_CONTROL);
    Thread.sleep(2000);
    r.keyPress(KeyEvent.VK_UP);
    Thread.sleep(2000);
    r.keyPress(KeyEvent.VK_TAB);
    Thread.sleep(2000);
    r.keyPress(KeyEvent.VK_DOWN);
    Thread.sleep(1000);
    r.keyPress(KeyEvent.VK_DOWN);
    Thread.sleep(1000);
    r.keyPress(KeyEvent.VK_DOWN);
    Thread.sleep(1000);
    r.keyPress(KeyEvent.VK_DOWN);
    Thread.sleep(1000);
    r.keyPress(KeyEvent.VK_1);
    r.keyPress(KeyEvent.VK_MINUS);
    r.keyPress(KeyEvent.VK_2);
```

```
Thread.sleep(1000);
r.keyPress(KeyEvent.VK_ENTER);
r.keyRelease(KeyEvent.VK_ENTER);
r.keyPress(KeyEvent.VK_A);
r.keyPress(KeyEvent.VK_ENTER);
r.keyRelease(KeyEvent.VK_ENTER);
```

Note :-

i) We use Robot class to handle print popup in all the browsers except chrome.

ii) In chrome we can inspect the popup which we can handle it by using findElement method.

Notification Popup :

Characterstic :

i) We can not move this popup
ii) We can not inspect this popup
iii) This popup will be having allow & block button
iv) It will be displayed bellow the address bar in the beginning

Solution :

i) To handle notification pupup to change the setting of the browser so that notification popup will not be displayed

ii) To change the setting of the browser we use add arguments method of 'chromeOptions' class.

Note :

i) addArguments is an example for method overloading where it takes string or list of strings as an argument.

ii) For every browser we have respective options class

Ex → ChromeOptions , FirefoxOptions , EdgeOptions ...etc

Ex → class NotificationPopup {

```
static {
System.setProperty("webdriver.chrome.driver", "./drivers/
                                    chromedriver.exe");
}

P S V m (S[] args) {
ChromeOptions option = new ChromeOptions();
option.addArguments("--disable-notifications");
WebDriver driver = new ChromeDriver(option);
driver.get("https://www.getkra.com/");
}
}
```

Note :-

In order to open the browser with modified settings we use parameterized constructor in the respective browser class.

new ChromeDriver() → open the browser in default settings.

new ChromeDriver(options) → will open the browser in modified setting

↑

The above statement is an example for constructor overloading

Child window or child Browser Popup :

Characterstic :

i) We can move this popup
ii) We can inspect this popup
iii) This popup will be having minimize, maximize and close button along with address bar.

Solution :

To handle child window popup we use getWindowsHandle() method and driver.switchTo().window() statement.

Note :

i) Address of the browser present on the desktop is called as window Handle.
ii) In order to retrive it we use getWindowHandle() method.

Ex:

```
class ChildWindow {
    static {
    ("path")
    }
    p s v m (S[] args) {
    WebDriver d = new ChromeDriver();
     d.get("https://secure.indeed.com/");
    String wh = d.getWindowHandle();
        S.o.p(wh);
     d.close();
    }
}
```

Q// WAS to print the address of all the browsers present in inded.com after clicking facebook and apple buttons.

```
class PrintAllWHandles {
static { System.setProperty(" ") }
  p s v m(S[]args) {
WebDriver d = new ChromeDriver();
 d.get(" https://secure.indeed.com/");
d.findElement(By.id("apple-signin-button")).click();
d.findElement(By.id("login-facebook-button")).click();
Set<String> allWh = d.getWindowHandles();
  int count = allWh.size();
    S.o.println(count);
for (String wh : allWh) {
    S.o.println(wh);
  }
d.quit();
}
}
```

Q/ What is difference bet" getWindowHandle() and getWindowHandles ||

getWindowHandle	getWindowHandles
i) return type of getWindowHandle is string	i) return type of getWindowHandles is set of string
ii) getWindowHandle will return the address of current browser	ii) getWindowHandles will return the address of all the browsers

Q// WAS to print the title of all the browsers present in indeed after clicking apple & facebook buttons.

```
class PrintAllTitle {

P S V m (S[] args) {
WebDriver d = new ChromeDriver();
d.get("https://secure.indeed.com/");
d.findElement(By.id("apple-signin-button")).click();
d.findelement(By.id("login-facebook-button")).click();
Set<String> allwh = d.windo getWindowHandles();
for(String wh: allwh) {
d.switchTo().window(wh);
String title = d.getTitle();
S.o.pln(title);
}
d.quit();
}
```

Q: WAS to close all the browser without using quit method

```
class withoutQuit {
p s v m (S[] args) {
WebDriver d = new ChromeDriver();
d.get("https://secure.indeed.com/");
d.findElement(By.id
d.findElement(By.id
Set<String> allWh = d.getWindowHandles();
for (String wh: allWh) {
d.switchTo().window(wh);
d.close();
}
}
}
```

Q/ WAS to close all the browsers except the parent browser.

```
class closeAllChild {
static {
System.setProperty("webdriver.chrome.driver", "./driver/
chromedriver.exe");
p s v m (S[] args) {
WebDriver d = new ChromeDriver();
d.get("https://secure.indeed.com");
```

```
d.findElements(By.id("login-signin-button")).click();
d.findElements(By.id("login-facebook-button")).click();
String pwh = d.getWindowHandle();
Set<String> allWh = d.getWindowHandles();
    for(String wh : allWh) {
    d.switchTo().window(wh);
        if(pwh.equals(WH)) {
        }
        else {
        d.close();
        }
}}
```

Q// WAS to close all the browser except the specific browser (By taking the input from the user)

Q// WAS to close only the specific browser.

1) e class CloseAllBrowser

```
p s v m(s[] args) throws InterruptedException {
WebDriver d = new ChromeDriver();
d.get("https://secure.indeed.com/");
d.findElement(By.xpath/id("login_facebook_button")).click();
d.findElement(By.id("apple-signin-button")).click();
Set<String> wh = d.getWindowHandles();
```

```
Scanner sc = new Scanner (System.in);
S.o.pln ("Enter the title");
String sb= sc.nextLine();
for (String w : wh) {
 String title = d.switchTo().window(w).getTitle();
  Thread.sleep(3000);
 if (title.contains(sb)) {
 }
 else {
  d.close();
 }
}
}
}
```

Handling Tabs :

i) Tab is also treated as new window in selenium.

ii) Here we are going to handle the tab in the same way as we handle child window & child browser popup

Q WAP to count the number of tabs open after clicking actitime Inc link present in actitime and print all the windowHandles

```
class TabHandle {

p s v m (S[] args) {
WebDriver d = new chromeDriver();
d.get("https://demo.actitime.com/");
d.findElement(By.linkText("actiTIME. Inc.")).click();

Set<String> at = d.getWindowHandles();
  int count = at.size();
    S.o.pln(count);

for (String t : at) {
    S.o.pln(t);
  }
  d.quit();
 }
}
```

Q// How do you close the current tab

→ By using d.close() method

Q// How do you close all the tab

→ By using d.quit() method.

Q// How do you close all the tab without using quit() method. (after clicking actitime.inc)

→ By using d.close() method in for each loop.

Handling Mouse Action :-

By using mouse we can perform the following actions
i) Mouse hover (Dropdown menu)
ii) Right click → contextClick()
iii) Drag & Drop
iv) Double click

Q// How do you handle mouse hover or dropdown menu.

→ mouse hover means moving the mouse pointer to a particular location.

→ Drop down menu means it is an element in which if we move the mouse pointer it will display the list of option.

→ To handle this we use move to element method of actions class.

→ In selenium Action is an Interface and Actions is a class

→ Acting class is present in interaction package

→ Actions class ~~accept~~ as a parameterize constructor where it takes webDriver as a parameter

→ Actions class is mainly used for mouse actions

→ Whenever we call any methods of Actions class we have to use perform() method at the last

8) Automate the following scenario.

1) open the browser
2) Go to Vtiger.com
3) mouse over to resources & select contact us in the dropdown menu
4) Get the bangalore India ph no and print it on the console

→

```
class Mousehover {
static {"path"}
p s v m (s[] args) throws InterruptedException {
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://www.vtiger.com/");
WebElement resources = driver.findElement(By.partialLinkText
                                ("Resources"));
Actions a = new Actions(driver);
a.moveToElement(resources).perform();
driver.findElement(By.partialLinkText("Contact us")).click();
Thread.sleep(3000);
String phno = driver.findElemnt(By.xpath("//p[contains
                                (text(),'Bangalore')]/../p[2]"))
                                .getText();
s.o.println(phno);
driver.close();
}
}
```

Q/ How do u perform context click (Right click)

i) Right clicking in mouse is called as context click

ii) When we right click on any element we get list of options which is called context menu

iii) To right click on the element we use context click of method of Actions class

iv) To select the required options we press shortcut key such as T for new Tab and w for new window

Q/ Write to open actitime-inc link in new window

```
class RightClick {
static {
System.setProperty("webdriver.chrome.driver",
        "./driver/chromedriver.exe");
}
p s v m (S[] args) throws AWTException
WebDriver d = new ChromeDriver();
d.get("https://demo.actitime.com/");
WebElement target = d.findElement(By.linkText("actitime inc"));
Actions a = new Actions(d);
a.contextClick(target).perform();
Thread.sleep(2000);
Robot r = new Robot();
r.keyPress(KeyEvent.VK_W);
}}
```

Q// How do you perform drag & drop

→ By using dragandDrop() method of Actions class

→ It takes an agrument as src and target destination

Ex→

```
class DragAndDropEx {
static { (path) }
p s v m (S [] args) throws InterruptedException {
    WebDriver d = new ChromeDriver();
d.get("http://www.dhtmlgoodies.com/submitted-scripts/i-google");
WebElement src = d.findElement(By.xpath("//h1[text()='Block 1']"));
WebElement target = d.findElement(By.xpath("//h1[text()='Block 4']"));
  Actions a = new Actions(d);
  a.dragAndDrop(src, target).perform();
    d.close();
}
}
```

when we write perform method it is internally call build() perform.

build will return only action it will not perform

How do you perform double-click in selenium.

By using doubleClick() method of Actions class.

Ex → a.doubleClick(webElement).perform();

WAS to navigate to vtiger.com and mousehover to resources and click on customers then click on double click on login and verify login page is displayed or not.

```
class Vtiger {
ps vm(s[] args) {{
WebDriver d = new chromeDriver();
d.get("https://vtiger.com/");
WebElement rc = d.findElement(By.partialLinkText("Resources"));
Actions a = new Actions(d);
a.moveToElement(rc).perform();
```

Note:-

i) All the methods of Actions class is an example for Method overloading.

ii) In order to end the action we need to use Perform method at the last.

iii) The meaning of Perform is execute.

Handling frames (Embedded webpage)

i) A web page inside another webpage is called as Embedded webpage (frames)

ii) Developers create embedded webpage using iframe tag

iii) while automating the element if it is present inside the frame we should transfer the driver control into the frame using :- driver.switchTo().frame('')

statement

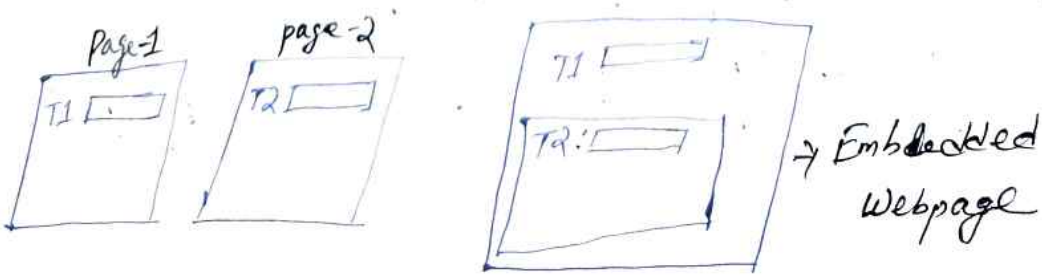
Sample HTML code:-

HTML code to create 1st frame (Page1.html)

```
T1: <input type="text" id="t1" /><br>
<iframe src="page2.html" id="f1" />
```

HTML code to create 2nd frame (Page2.html)

```
T2: <input type="text" id="t2" />
```



// WAS to enter JSP in T2 textbox and QSP in T1 textbox.

```
class Handling frames {
P S V m (S [] args) {
WebDriver d = new ChromeDriver();
d.get("file:///C:/users/ajits/Desktop/page1.html");
d.switchTo().frame(0);
d.findElement(By.id("t2")).sendKeys("JSP");
d.switchTo().parentFrame();
d.findElement(By.id("t1")).sendkeys("QSP");
}
}
```

// WAS to type abcd in T1 and T2 texbox alternatively character by character

```
class Handling frame 2 {
P S V m (S[] args)
WebDriver d = new ChromeDriver();
d.get("file:///C:/users/ajits/Desktop/page1.html");
d.findElement(By.id("t1")).sendkeys("a");
d.switchTo().frame("f1");
d.findElement(By.id("t2")).sendkeys("b");
d.switchTo().defaultContent();
d.findElement(By.id("t1")).sendKeys("c");
```

```
WebElement e = d.findElement(By.xpath("//iframe"));
d.switchTo().frame(e);
d.findElement(By.id("ta")).sendKeys("d");
}
}
```

Note :-

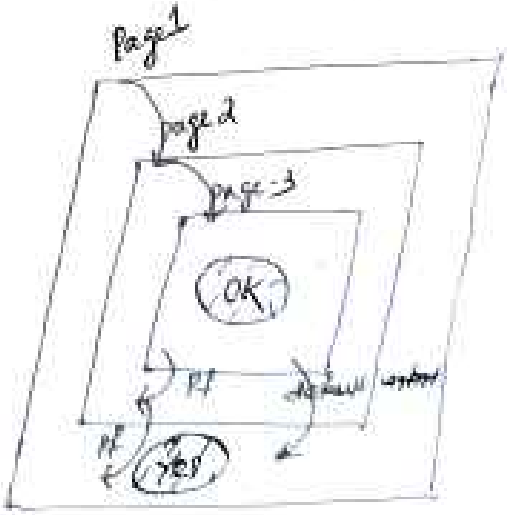
i) In the above example frame method is overloaded

ii) It takes only one argument of any of the following 3 types.

i) int - (Index of the frame)

ii) String (Attribute or id of the frame)

iii) WebElement (address of the frame)



Ex→

```
d.switchTo().frame(0);
d.switchTo().frame(0);
d.findElement().click();
d.switchTo().pf();
d.switchTo().pf();
d.switchTo().defaultContent();
d.PF().click() → Yes
```

Handling disabled Elements & Scroll bar :

i) In selenium we don't have a method to handle disabled element & scroll bar (By using java lang)

ii) In order to handle we use executeScript() method

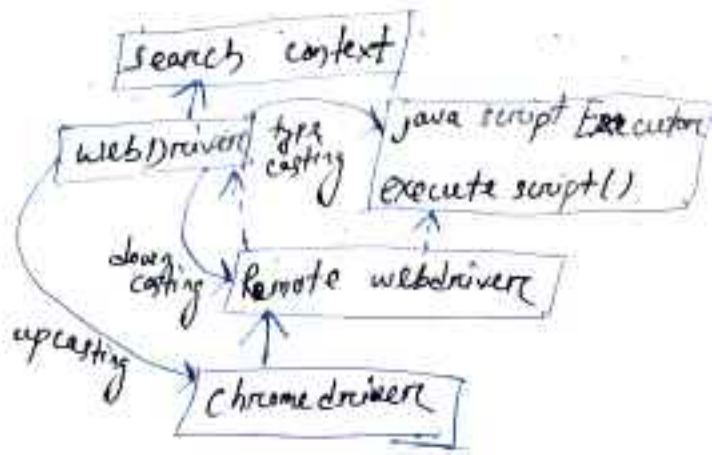
iii) The executeScript() method is declared in JavaScript executer Interface and implemented in Remote WebDriver class

iv) Since we already upcasted chromeDriver class to WebDriver Interface. This method will be hidden

v) In order access this method we should downcast it to RemoteWebDriver or typecast the object to javaScript executer.

Sample HTML code :

```
cen: <input type="text" id="d1" /><br>
pw: <input type="text" id="d2" disabled /><br>
<input type="button" id="d3" value="Login">
```



Ex → class HandlingDisableElement

```
psvm(s[]args){

ChromeDri
WebDriver d = new chromeDriver();
d.get("file:///C:/users/ajits/Desktop/disabled.html")
d.findElement(By.id("d1")).sendkeys("admin");
RemoteWebDriver r = (RemoteWebDriver) driver;
r.executeScript("document.getElementById("d2").value="manager");

}
}
```

Note:-
In order to validate Java script statement in the browser, Inspect the element & click on console tab present in the developers toolbar & type the script

document.getElementById("d2").value = '' → for deleting the text present in textbox

d.getElementById("d3").click() ⟹ for clicking on the element or button

Q/ Handling scrollbar → was to scroll 3000 pixel vertically in bbc.com.

```
class HandlingScrollBar {
static {
}
  p s v m (s [] args) {
WebDriver d = new ChromeDriver();
 d.get("https://www.bbc.com/");
javascriptExecutor j = (javascriptExecutor) d
 j.executeScript("window.scrollBy(0,3000)");
}
}
```

Q/ was to scroll to the particular element

→
```
class HandlingScrollBar2 {
p s v m (s [] args) {
WebDriver d = new ChromeDriver();
d.get("https://www.bbc.com/");
int y = d.findElement(By.xpath("//span[text()='Future Planet']"))
                                    .getLocation().getX();
JavascriptExecutor J = (javascriptExecutor) d;
 J.executeScript("window.scrollBy(0,"+y+")");
}
}
```

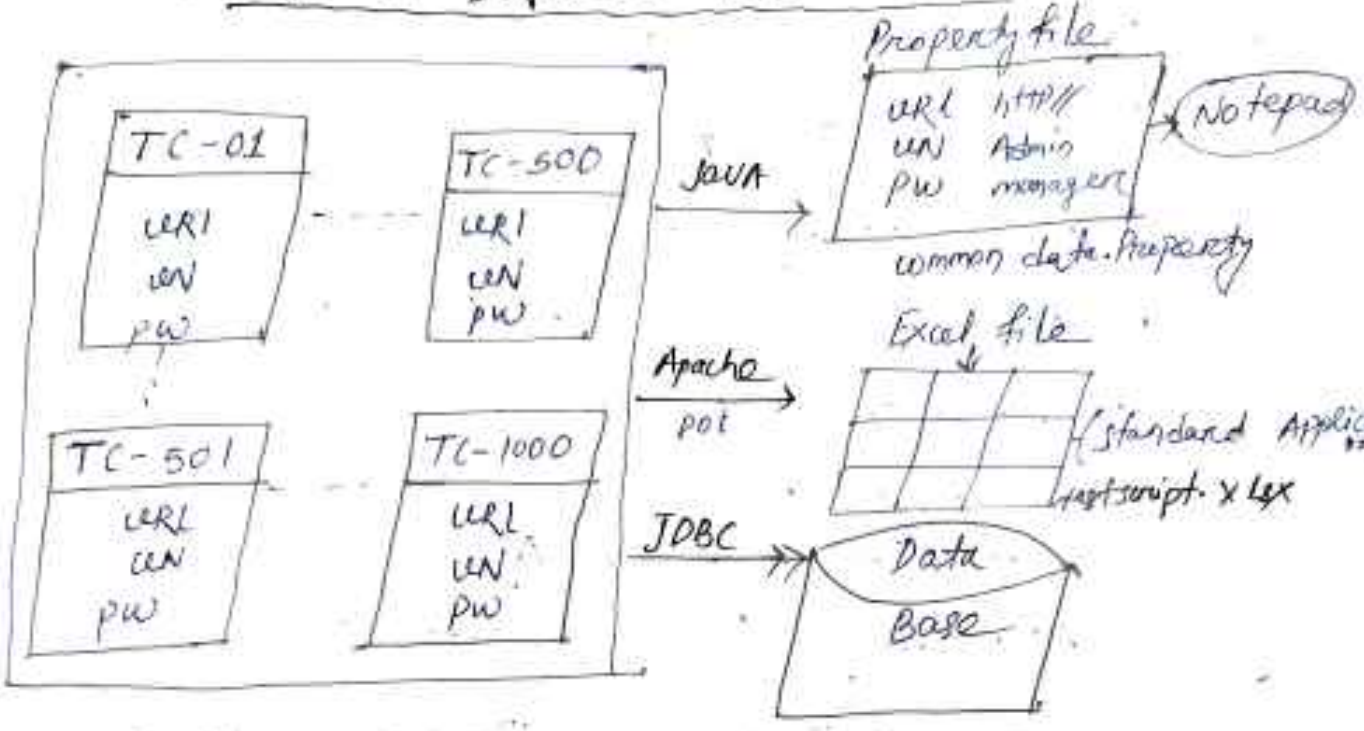
Q// WAP to scroll to the bottom of the webpage & scroll to top of the web page

```
class ScrollToBottom {
p s v m (S[]args) throws InterruptedException {
WebDriver driver = new ChromeDriver();
d.get("https://www.bbc.com/");
Thread.sleep(3000);
//scroll to bottom of the webpage.
JavascriptExecutor J = (JavascriptExecutor) d
J.executeScript("window.scrollTo(0, document.body
.scrollHeight)");
Thread.sleep(3000);
//scroll to top of the webpage
J.executeScript("window.scrollTo(0,0)");
}
}
```

ScrollTo :- It will scroll from the top of the webpage Always

ScrollBy :- It will scroll from wherever the scroll bar or control is present.

DATA DRIVEN TESTING



Def^n

Testing the application with multiple inputs or test data which is kepting external resources file like property file, Excel file, Database is called as Data Driven Testing.

Handling Property File :

i) In property file data will stored in the form of key value concept of Map.

ii) key and value should be separated by single space

iii) By default all the data present in property file is string.

Note:-
copy the below code in the Notepad and save it as commondata.property

```
url      https://demo.actitime.com/
username admin
password manager
```

vimp
// WAP to read the data from the property file.

```
class HandlingPropertyFile {
    P S V M (S[] args) throws IOException
// get the java representive object of the physical file
    FileInputStream fis = new FileInputStream("./data/commondata");
// create an object of properties class
    Properties p = new Properties();
    // load the file
    p.load(fis);
// get the ~~value~~ data (value) By passing key
    String url = p.getProperty("url");
    String un = p.getProperty("username");
    String pw = p.getProperty("password");
    S.O.println(url);
    S.O.println(un);
    S.O.println(pw);
  }
}
```

Asst) WAS To login to Actitime by taking the test data from the property file.

```
class FileData {
static { }
P S V m (S [ ] args) throws FileNotFoundException, IOException {
FileInputStream fis = new FileInputStream (" file path");
Properties p = new Properties ();
    p. load (fis)
string url = p. getProperty ("url");
string un = p. getProperty ("username");
string pw = p. getProperty ("password");
WebDriver d = new ChromeDriver ();
d. get (url);
d. findElement (By. name ("username")). sendkeys (un);
d. findElement (By. name ("pwd")). sendkeys (pw);
d. findElement (By. xpath ("//div [text() = 'Login']")). click();
d. close ();
}
}
```

Advantages of Property file :

i) It is very faster in execution

ii) It is very light weight when compare to any other external resources file.

Handling Excel File : (Data Driven from excel file)

i) We need to use Apache poi plugin or third party tool in order to read the data from All microsoft documents like .xls, .xlsx, .docx, .ppt etc.

ii) Apache file is a free and open source tool similar to selenium.

Installation of Apache poi

→ Go to Google then search for Apache poi download

→ click on 1st link & navigate to Apache poi website.

→ Scroll down and click on binary Artifacts link

→ Then it take to download page

→ Scroll down and click on the latest zip file. (poi-bin-5.2.3-zip)

→ It will download the zip file.

→ unzip the downloaded folder & copy all the 50 jars present in all the folders and paste it into jar folders of eclips.

→ Add all the jars into the build path.

imp Q// WAP to read the data from the excel file.

```
class HandlingExcelfile {

p s v m (s [] args) throws EncryptedDocumentException,
                           IOException {

FileInputStream fis = new FileInputStream

// create a workbook and open the excel in read mode

Workbook wb = WorkbookFactory.create(fis);

// get the control of the sheet, then row, then cell
   then read the data

String data= wb.getSheet("createCustomer").getRow(1)
            .getCell(3).getStringCellValue();

   s.o.println(data);
  }
}
```

Q// WAP to write the data back to excel file

```
class WDataToExcel {

p s v m (s[] args) throws EncryptedDocumentException,
                          IOException {

FileInputStream fis = new FileInputStream(

Workbook wb = WorkbookFactory.create(fis);
```

```
wb.getSheet("create customer").getRow(1).getCell(4)
.setCellValue("fail");
// get the physical file of the java representative object
FileOutputStream fos = new FileOutputStream(
// save the workbook
wb.write(fos);
// close the workbook
wb.close();
}
}
```

WAP to read the multiple data from the excel

```
class HandlingMultipleData {
psvm(S[] args) throws EncryptedDocumentException,
                      IOException.{
FileInputStream fis = new FileInputStream(
Workbook wb = WorkbookFactory.create(fis);
// Get the last row number which contains data
int rowCount = wb.getSheet("InvalidLogin").getLastRowNum();
for(int i=1; i<=rowcount; i++) {
String un = wb.getSheet("InvalidLogin").getRow(i).getCell(0)
            .getStringCellValue();
String pw = wb.getSheet("InvalidLogin").
            getRow(i).getCell(1).
            getStringCellValue();
S.o.println(un+" "+pw);
}}
```

Note:-

i) Whenever we are reading the data from excel the file extension should be .xlsx

ii) All the apache poi classes should be imported from org.apache.poi.ss.usermodel package.

Advantages of Data Driven Testing:-

i) Maintenance of data in excel or external resources file is easier.

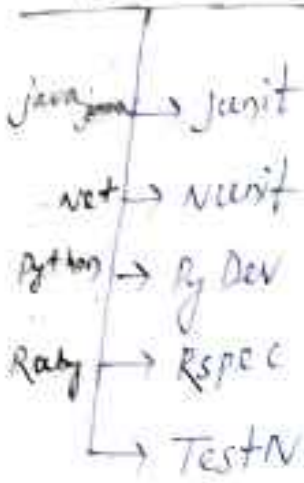
ii) Modification of data in excel or external resources file is easier

iii) Reusability of common data and testscript data.

iv) We can test the application with huge volume of data.

v) We can create the test data before writing the testscript.

UNIT TESTING FRAMEWORK TOOL



TestNG (Test Next Generation)

1) TestNG is a unit Testing framework Tool which is mainly used for Batch execution.

or

2) It is a plugin for Eclipse which is inspired by Junit & Nunit with some additional features.

3) Basically TestNG is used by developers to perform unit Testing & it is also used in selenium to perform black box Testing.

Advantages or Additional features of TestNG

1) Batch Execution
2) Parllal Execution (cross browse Testing)
3) Group Execution
4) Generate the Reports Automatically (HTML)
5) Run only failed Test scripts
6) Additional Annotations
7) Listners features

Installation of TestNG :-

1) In Eclipse go to help and select Eclipse Market Place.
2) Type TestNG in the search Textbox & click enter
3) click on the install button present under TestNG for eclipse.
4) click on confirm & finish button.
5) select the unsigned ~~button~~ checkbox & click on trust selected.
6) Then Restart now the eclipse.

Note:
After Installing TestNG to the eclipse we need to add TestNG to the java project present in Eclipse

i) Right click on java project go to build path, then select ~~Add~~ Add libraries then select TestNG and click on next then finish.

ii) In TestNG we should not use following these thing

1) No Main Method
2) No default Package
3) Don't ~~use~~ use s.o.p().

TestNG Example :-

```
package com.actitime.testscript
import org.testng.Reporter;

public class DemoTest {
  @Test
  public void test() {
      Reporter.log("welcome to Testing", true);
  }
}
```

When we execute the above code it will automatically generate the execution result in HTML (Report)

In order to see the report

1) Refresh the Java Project (F5)
2) which will display Test output folder
3) expand the folder & do right click on emailable-report.html
4) Go to open with and select web browser.

Note :-

```
S.O.println("hi"); // Prints only on console
Reporter.log("bye", true); // prints on report & console
Reporter.log("hello", false); // prints only on report
R.l("Welcome"); // Prints only on report.
```

Interview Questions

1) if a class contains multiple test methods in which order they are executed

→ Alphabetical Order

2) How to execute the test methods in required order ?

→ By using Priority

SYNTAX: @Test(priority=1)

Note :- i) Default priority is @0. if priority is duplicate then those methods will be executed in alphabetical order

ii) We can specify -ve value for priority and it will execute them in assending order

iii) Variable & decimals are not allowed.

3) How Do you Run a test method multiple times ?

→ By using invocationCount

SYNTAX: @Test(invocationCount=3)

Note :- i) Default invocation count is 1

ii) if we use 0 or -ve value it will not execute that test method.

iii) Variables and decimals are not allowed.

4) How do you make a test depends on another test ?

By using dependsOnMethods option

SYNTAX : @Test (dependsOnMethods = "create customer")

@Test (dependsOnMethods = {"create customer", "Modify customer"})

Note : if both priority and dependency are specified it will consider the dependency.

5) What if two test methods are depends on each other

→ we get TestNG Exception (Error is Cyclic dependencies)

6) How do you disable the testCase or testMethod

By using Enabled = false or invocationCount = 0 or -ve

SYNTAX : @Test (enabled = false)

Batch Execution :

i) Executing the collection of multiple test script together via TestNG.xml suit file is called Batch Execution.

ii) To perform Batch execution we need to create TestNG.xml suite file.

In order to create the Test suite

1) Right click on the project, go to to testNG and select convert to TestNG

2) click on finish it creates TestNG.xml file inside the project.

To execute it :

i) Right click on xml file &
2) go to run as & select TestNG suite

or

open the suit file & click on run button.

content of TestNG.xml suite file

```
<suite name = "suite">
  <test thread-count = "5" name = "Test">
  <classes>
<class name = "com.actitime.testscipt.ProjectModule"/>
<class name = "  "  "  "  .CustomerModule"/>
<class name = "  "  "  "  .TaskModule"/>
</classes>
</test>
</suite>
```

Q. How Do u execute only failed test cases?

By using TestNG-failed.xml file present in the test-output folder.

Q. How Do u failed the test case Intentionally?

→ By using Assert.fail()

Assertion :-

i) Assertion is a feature present in testNG which is used to verify the actual & expected result of the test script.

Ex→

```
class DemoAssertion {
    static {
        (path")
    }
    @Test
    public void verifyTitle() {
    WebDriver d = new ChromeDriver()
        d.get("https://www.google.com/");
    String aTitle = d.getTitle();
        String eTitle = "google";
        if (aTitle.equals(eTitle)) {
            S.o.println("Title is matching & pass");
        }
        else {
            S.o.println("Title is not match & fail");
        }
}}
```

As per the rule of Automation every expected result should be varified with Assert statement instead of java if else statement bcz if else block doesn't have the capacity to fail the testscript.

In Assertion There are 2 types

1) Assert (Hard Assert) → static
2) Soft Assert → Non-static

Q/ What are the important methods present in Assert class

1) assertEquals()
2) assertNotEquals()
3) assertSame()
4) assertNotSame()
5) assertNull()
6) assertNotNull()
7) assertTrue()
8) assertFalse()
9) fail()

All the above methods are static methods of assert class.

Q/ How Do u compare actual value with expected value without using if else statement

→ By using assertEquals method of Assert class

Ex of using assertEquals method :

```
class Assert {
static {("path")}

@Test
public void verifyTitle() {
WebDriver d = new ChromeDriver();
d.get("https://www.google.com/");
String eTitle = "Google"
String aTitle = d.getTitle();
Assert.assertEquals(aTitle, eTitle);
d.close();
}
}
```

Note :

→ if comparision fails then the statements which are present after the assert statement of the current test method will not be executed

→ In the above example it will not close the browser if comparision fails.

→ In order to continue the execution even after failure of the comparision. we can use soft Assert

→ But here all the methods are non-static

Ex of using Soft Assert :-

```
@Test
public void verifyTitle() {
WebDriver d = new ChromeDriver();
d.get("https://www.google.com/");
 String eTitle= "Soogle";
 String aTitle = d.getTitle();
SoftAssert s = new SoftAssert();
s.assertEquals(aTitle, eTitle);
 d.close();
 s.assertAll();
 }
}
```

Note :-

- To update the states of the comparision into the result window we should use assertAll method at the last
- Any statement after assertAll method will not be executed if comparision fails.

Difference bet^n Assert & soft Assert :-

Assert (Hard)	soft Assert
i) All the methods are static	i) All the methods are non-static.
ii) if the comparision fails remaining statement will not be executed in current method	ii) Exeeeetes remaining statements even if the comparision fails.
iii) We do not call assertAll method	ii) We should call assertAll method at the last

Annotation

Annotation is a block of data which provides special instruction to the java compiler during runtime.

or

It is one of the block in java which is used to develop or provide custom instruction.

Important Annotation of TestNG :

1) @ Beforesuite
2) @ Before Test
3) @ Before class
4) @ Before Method
5) @ Test
6) @ After Method
7) @ After class
8) @ After Test
9) @ After Suite

Optional Annotations :

1) @ Listeners
2) @ DataProvider
3) @ Parameters
4) @ Before Groups
5) @ After Groups
6) @ Ignore
7) @ Factory

Ex →

```
public class CustomerModule {
    @BeforeMethod
    public void login() {
        Reporter.log("login", true);
    }
@AfterMethod
    public void logout() {
        R.l("logout", true)
    }
    @Test
        public void createCustomer() {
            R.l("createCustomer", true)
        }
    @Test
        public void deleteCustomer() {
            R.l("deleteCustomer", true);
        }
    }
}
```

output

login
createCustomer
logout
login
deleteCustomer
logout.

Example →

```
public class cm {
@BeforeClass
 public void openBrowser () {
  Reporter.log ("open browser", true);
}
@Afterclass
public void closeBrowse () {
  R.l("closeBrowser", true);
}
@BeforeMethod
 public void login() {
   R.l("login", true);
 }
@AfterMethod
  public void logout() {
     R.l ("logout", true);
  }
@Test (priority=1, invocationCount=2)
   public void editCustomer () {
      R.l("editcustomer", true);
   }
  @Test
   public void registerCustomer() {
      R.l("registerCustomer", true);
   }
  @Test
     P. V deleteCustomer () {
        R.l("deleteCustomer", true);
  } }
```

output

```
open browser
  login
deleteCustomer
  logout
  login
RegisterCustomer
  logout
  login
editCustomer
  logout
  login
editcustomer
  logout
  closebrowser
```

Annotation usage in Realtime :

@BeforeMethod

i) @BeforeMethod Annotation will be executed Before the execution of every @Test Method

ii) In Realtime it will be used to develop common preconditions like login program

@AfterMethod:

i) @AfterMethod Annotation will be executed after the execution of every @Test method

ii) In RealTime it is used to develop common postconditions like logout program

@AfterClass:

i) @AfterClass Annotation will be executed after the every @Test class

ii) In Real Time it is used to close the browser

@Beforeclass

i) @Beforeclass Annotation will be executed before the execution of the every @Test class

ii) In RealTime it is used to open the browser

@Test : i) @Test Annotation indicates @test method

ii) In RealTime @Test method indicates 1 test case or Test script

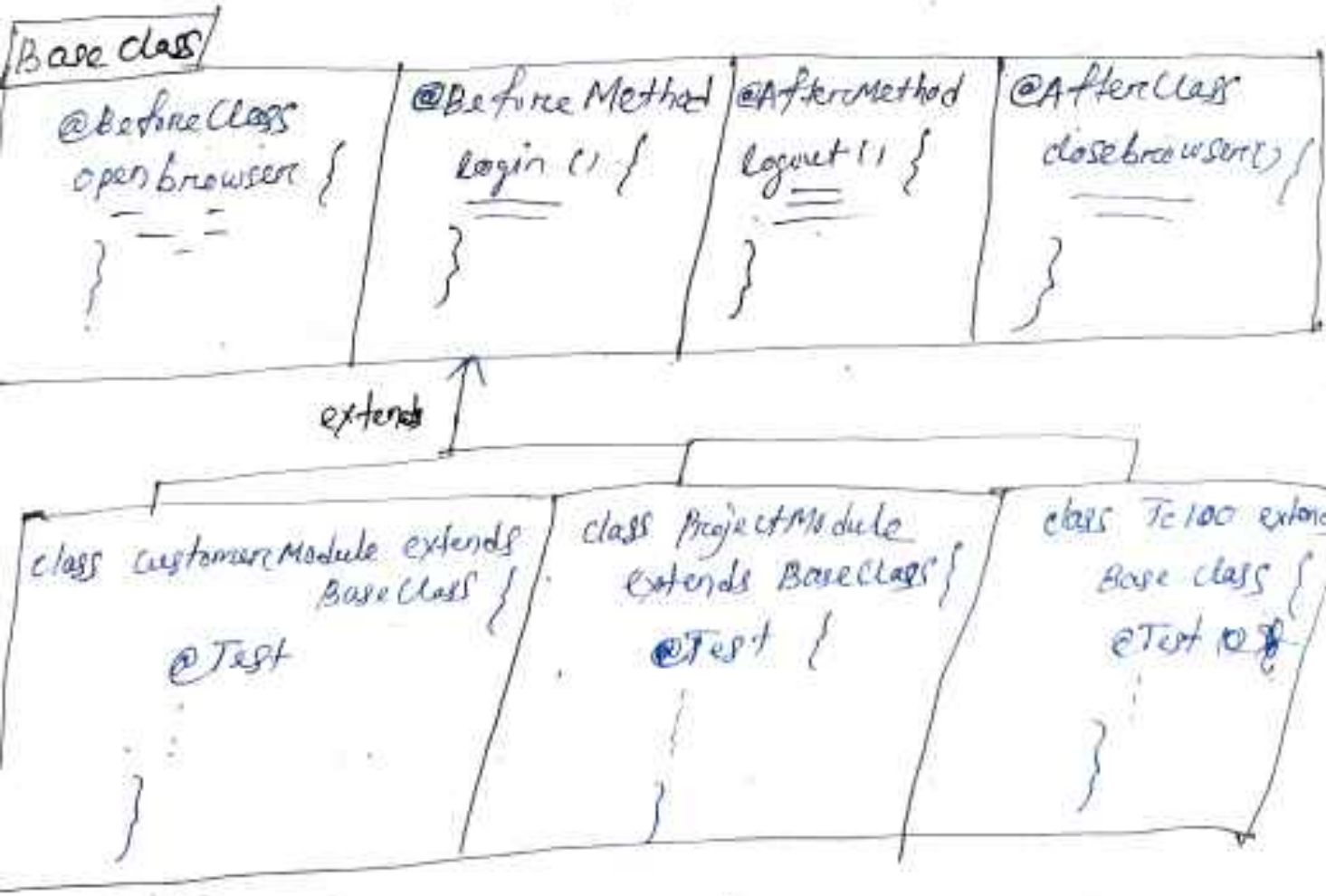
Base Class :

i) Base class is the supermost class in the framework and it will be created under com.actitime.generic package

ii) It contains all the configuration method like open browser, login, logout and close browser etc

iii) All the test classes will be created under com.actitime.testscript ~~class~~ package

iv) every test class should extends base class as shown below.



```
public class BaseClass {
  @BeforeClass
    public void openBrowser() {
        Reporter.log("openbrowser", true);
    }
  @AfterClass
    public void closeBrowser() {
      R.L("closebrowser", true);
  }
  @BeforeMethod
    public void login() {
        R.L("Login", true);
  }
  @AfterMethod
      public void logout() {
        R.L("logout", true);
      }
}
```

Content of Base class:

```
public class BaseClass
static {

}
public WebDriver driver;
    @BeforeClass
        public void openBrowser() {
        Reporter.log("openBrowser", true);
        driver = new ChromeDriver();
        driver.manage().window().maximize();
driver.manage().window().timeouts().implicitlyWait(10, TimeUnit.SECONDS
}
    @AfterClass
        public void closeBrowser() {
            R.l("closeBrowser:", true);
            driver.close();
        }
    @BeforeMethod
        public void login() {
            R.l("login", true);
        driver.get("demo.actitime.com");
driver.findElement(By.name("username")).sendKeys("admin");
driver.findElement(By.name("pwd")).sendKeys("manager");
driver.findElement(
```

```
@AfterMethod
  public void logout() {
    Reporter.log("logout", true);
    driver.findElement(By.id("logout link")).click();
  }
}
```

```
public class CustomerModule extends BaseClass {
  @Test
    public void createCustomer() {
      R.L("create customer", true);
    }
}
```

Q. Write to log in to actitime to read the property file using TestNG Method.

```
public class Base1 {
static {
System.setProperty("webdriver.chrome.driver", "./drivers/
                                    chromedriver.exe");
}
  WebDriver driver;
@BeforeClass
  public void ob() {
    R.L("open browser", true);
    driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
```

```
@After Class
  public void cb() {
     R.l("closebrowser", true);
     driver.close();
  }
@Before Method
  public void login() throws IOException {
  R.l("Login", true);
FileInputStream fis = new FileInputStream("./data/commondata
                                          .property");
Properties p = new Properties();
     p.load(fis);
string url = p.getProperty("url");
string un = p.getProperty("username");
string pw = p.getProperty("password");
     driver.get(url);
driver.findElement(By.name("username")).sendKeys(un);
driver.findElement(By.name("pwd")).sendkeys(pw);
driver.findElement(By.xpath("//div[text()='Login']")).click();
  }
@AfterMethod
     public void logout() {
        R.l("logout", true);
driver.findElement(By.id("logoutlink")).click();
}
}
  public class Createcustomer extends Base1
  {
    @Test
      public void creatcustomer() {
        R.l("create customer", true);
    }
```

Q/ WAS to take the screenshot of the web page ?

```
class DemoScreenShot {
static {("path")
} @Test
  public void screenshot() throws IOException {
   WebDriver driver = new ChromeDriver();
   driver.get
   TakeScreenshot t = (TakeShreenshot) driver;
  File src = t.getScreenshotAs(outputType.FILE);
  File dest = new file("./Screenshot/ss.png");
   FileUtils.copyfile(src, dest);
    driver.close();
  }
}
```

Encapsulation :-

i) The process of hiding the data and binding with method in order to hide internal implementation is called as encapsulation.

ii) we making this by making the variable as private and access outside the class with the help of getter and setter method

Ex →

```
package com.actitime.pom;
class A
{
 private int i; // declaration
  A (int j)
  {
    i = j     // initialization
  }
 int getValues ( ) // giving read access
 {
  return i;
 }
 public void setValues(int k) // giving write access
  {  i = k;
  }
      class B {
      p s v m (s[] args) {
     A  a1 = new A (10);
     int x = a1.getValue(); // utilization
       s.o.pln(x);
     a1.setvalue(30);
     s.o.pln ( a1.getValue() ); // utilization
  } }
```

Program

```
Package com.actitime.pom;
public class LoginPage {
  private WebElement untbx; //declaration
  public LoginPage(WebDriver driver) {
  untbx = driver.findElement(By.id("username")); //initialization
  }
  public void setloginuser(String un) {
  untbx.sendKeys(un); //utilization.
  }
  }
}
  class MainMethodclass {
static {
  ("path")
}
  public psvm(String[] args) {
  WebDriver d= new ChromeDriver();
  d.get("actitime.com");
  LoginPage l = new LogthPage(driver)
    l.setUser("admin");
  }
  }
}
```

Note :

Whenever data is stored in a variable, in java for any given variables we should perform following steps:

i) declaration
ii) Initialization
iii) Utilization.

i) There are two classes in the above example. the purpose of class A is only to manage the variable i.

ii) Whereas the purpose of class B is only to execute the code

Using Encapsulation in selenium

→ selenium code to enter admin in the username textbox

d.findElement(By.id("username")).sendkeys("admin");

above code can also be written as

WebElement usrtbx = driver.findElement(By.id("username"));
usrtbx.sendkeys("admin");

OR

WebElement usrtbx; // declaration
usrtbx = d.findElement(By.id("username")); //initialization
usrtbx.sendkeys("admin"); //utilization

Program

```
public class LoginPage {
    private WebElement untbx;
    private WebElement pwtbx;          Declaration
    privat WebElement lgBtn;

    public LoginPage(WebDriver driver) {
  untbx = driver.findElement(By.id("username"));      initialiati
pwtbx = driver.findElement(By.name("pwd"));
lgBtn = driver.findElement(By.xpath("//div[.='login']"));
}
    // business logic method
    public void setLogin(String un, String pw) {
      untbx.sendKeys(un);
      untbx.sendKeys(pw);                    // utilization
      lgBtn.click();
    }
}
```

com.actitime.pom

```
    public class MainMethodcls {

static {
System.setProperty("pass")
 }
```

```
p s v m (s [ ] args) {
WebDriver driver = new ChromeDriver ();
  driver.get ("https://demo.actitime.com");
   LoginPage l = new LoginPage (driver);
   l.setLogin ("admin", "manager");
  }
}
```

Note :-

. The setlogin method present loginPage class can be used to enter invalid and valid inputs

```
p s v m (s [ ] args) {
WebDriver d = new ChromeDriver ();
  d.get ("actitime.com");
LogioPage l = new LoginPage (driver);
  l.setLogin ("admin1", "manager");
    Thread.sleep (4000);
     l.setLogin ("admin", "manager");
  }
}
```

When we execute the above code we may get. stale Element reference exception because when it clicks on login button after entering invalid username and password page will be reloaded and address of the element will be changed. but the reference variable such as wntbx will be holding old address

will try to enter valid username using old address which is no longer exist (invalid) Hence we get the exception

script to Explain StaleElementReferenceException

```
public class DemoPom {

static {

}

    p s v m (S[]args) {
    WebDriver d = new ChromeDriver();
    d.get("https://demo.actitime.com/");
  // stores the username textbox address as @a1 in untbx
WebElement untbx = d.findElement(By.id("username"));
// refresh and username textbox gets new address like
                                        @b1

d.navigate().refresh();
//try to enter admin using old address i.e @a1
    untbx.sendKeys("admin");
}

}
```

stale means old or no longer fresh

8 Methods are there in ITestListener

1) onTestStart(ITestResult result)

```
    ITestListener.super.onTestStart(result);
}
```

2) onTestSuccess
3) onTestFailure
4) onTestSkipped
5) onTestFailedButWithinSuccessPercentage
6) onFinish
7) onStart
8) onTestFailedWithTimeout

PAGE OBJECT MODULE (POM & Object Repository)

i) Page object Module is one of the java designed pattern which is used to store the objects (or elements)

ii) POM concept is used by the developers to develop the web pages and it is also used in Automation to test the web pages.

iii) It is also used to avoid stale element reference exception.

iv) In pom class we declare the element by using @FindBy Annotation

v) @FindBy Annotation should be imported from org.openqa.selenium.support.FindBy; package

vi) The SYNTAX is as followed.

SYNTAX For Single Element:

```
@FindBy (Locator = "locator value")
private WebElement ElementName;
```

SYNTAX For Multiple Element:

```
@FindBy ( Locator = "Locator value")
private List<WebElement> ElementName;
```

To initilice the element we use:

i) initElements Method of PageFactory class

ii) It will take two arguements 1) WebDriver (driver)
2) object of POM class (this)

iii) InitElements Method will only loads the element (only declare) But it will not initialize actually

iv) Elements are actually initialized during the runtime when we try to perform any action on the elements. this process is called as Lazy Initialization.

v) This will avoid StaleElementReferenceException.

Ex →

```
public class LoginPage {
    @FindBy (id = "username")
    private WebElement usrtbx;

    @FindBy (name = "pwd")
    private WebElement pwdtbx;

    @FindBy (xpath = "//div[. = 'Login']")
    private WebElement lgBtn;
```

```
public LoginPage (WebDriver driver) {
    PageFactory.initElements (driver, this);
}
public void setLogin (String un, String pw) {
    untbx.sendKeys (un);
    pwtbx.sendKeys (pw);
    lgbtn.click();
}
}
class TestMethodCls {
static {
System.setProperty ("path");
}
@Test
public void validLogin() {
WebDriver d = new ChromeDriver();
d.get ("https://demo.actitime.com/");
LoginPage l = new LoginPage (driver);
l.setLogin ("admin", "manager");
```

Note:

The class in which elements of the webpage are stored By using @FindBy Annotation is called as Pom class. (page class)

Ex → LoginPage, HomePage, taskListPage

The class which we execute using @Test is called as Test class.

Ex → TestMethodCls, customerModule,

We creating this classes under com.actitime.testScript

Advantages of POM

i) WebElements are maintained based on the web pages so maintaince of the webElements are easy.

ii) Modification of webElement locators is easy whenever ui is getting changed.

iii) xpath and locators of the webElements are reuseable so that no need to put same effort to rewrite the xpath again & again.

iv) We can avoid staleElementReferenceException.

Note:

i) All the pom classes will be create under com.actitime.pom package.

ii) No of pom classes will eqeal to no of web pages present in application.

Interview Questions ?

1) How will you declare an element in POM
→ By using @FindBy Annotation

2) How do you initialize an element in POM
→ By using initElements() of the PageFactory class
PageFactory.initElements();

3) What is object Repository ?
It is a location where we store the objects
~~page object~~ POM is also called as object Repository.

4) What happens if we don't use initElements() in POM class or anywhere else.
→ we get Null pointer Exception.

5) Can we develop a POM class without a constructor
→ Yes.

1) Can we use initElements in test or method class
→ Yes but we should explicitly call initElements() in Test class or main method.

7) What is Pagefactory ?
It is a class which implements the POM concept

Framework :-

i) Framework is a set of rules and guidlines or best practice to be followed by automating only application

ii) There are 6 types of frameworks present in the industry such :- i) Data Drives Framework
ii) Modular Drives framework
iii) Method " "
iv) Hybrid Driven " "
v) Keyword Drives Frame Work
vi) BDD Framework (Behaviour driven Development).

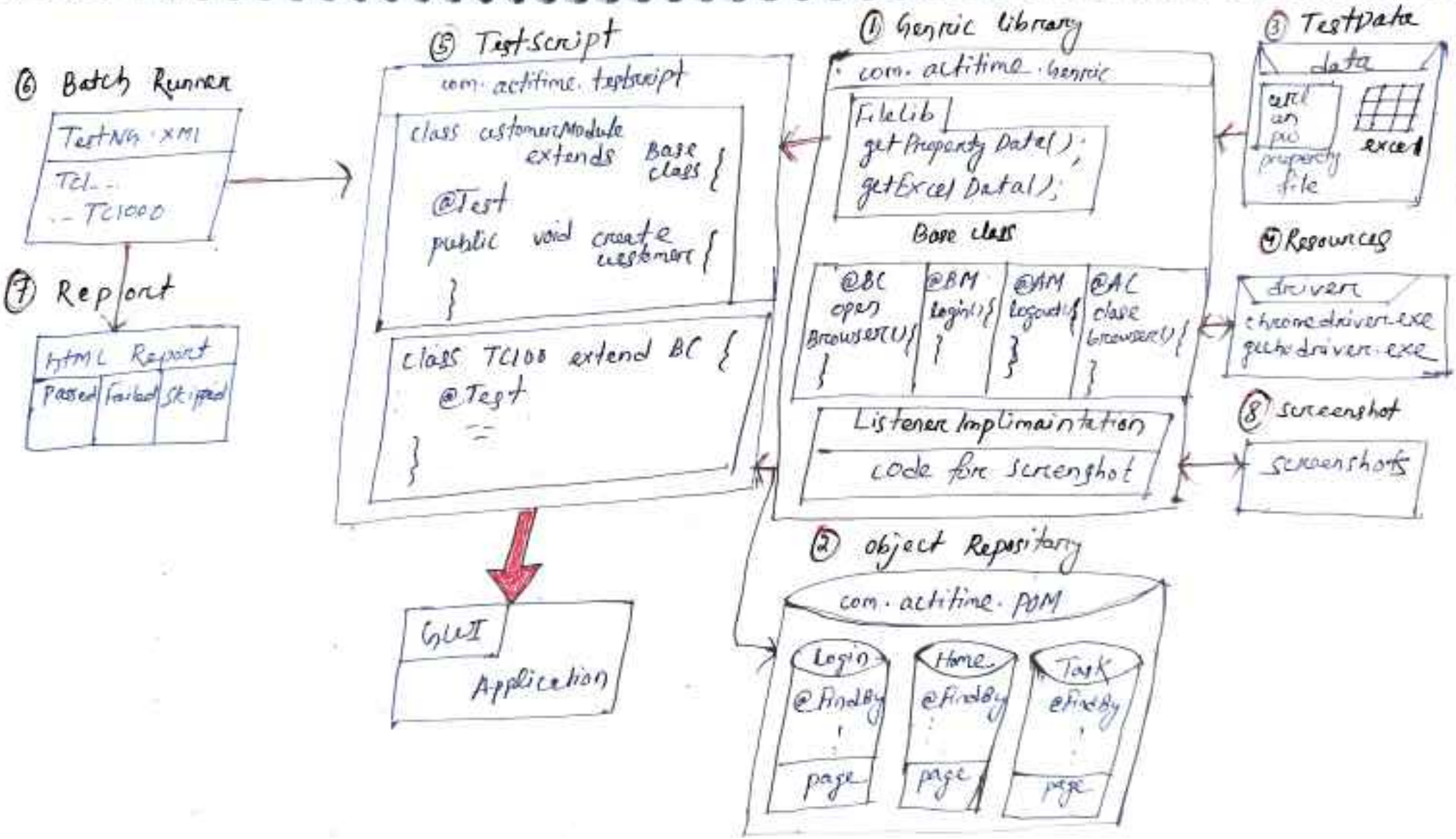
Architecture of Hybrid Drives Framework

i) Framework is a set of reeles and guidelines or best practice to be followed by automating only Application.

ii) In order to execute the test case with multiple inputs or test data we use excel and property file so we call our framework Data driven Frame work

iii) Since we maintain our framework module wise so we also call our framework as modelar driven framework.

iv) In order to avoid writing repeatative state again and again we use lot of reusable methods so we call our framework as method driven framework


⑥ Batch Runner
TestNG.xml
Tc1...
..Tc1000
⑦ Report
HTML Report
Passed
Failed
Skipped
⑤ Testscript
com.actitime.testscript
class customerModule extends Base class {
@Test
public void createCustomer{
}
class TC100 extend BC {
@Test
}
GUI
Application
① Generic library
com.actitime.generic
FileLib
getPropertyData();
getExcelData();
Base class
@BC
openBrowser(){
@BM
login(){
@AM
logout(){
@AC
closeBrowser(){
Listener Implimaintation
code for screenshot
② object Repository
com.actitime.POM
Login
@FindBy
page
Home
@FindBy
page
Task
@FindBy
page
③ Testdata
data
util
no
property
file
excel
④ Resources
driver
chromedriver.exe
gechodriver.exe
⑧ screenshot
screenshots

5) Since it is the combination of two or Three framework hence we call our framework was hybrid driven framework.

6) In the begining of the execution 1st it executes the base class which is present in the generic package which contains all the configuration methods like @Before class, @BeforeMethod, @AfterMethod, @After class

7) 1st it execute @Before class which contains the code for opening the browser, then it will executes login code which is present in @BeforeMethod then it will executes the @TestMethod where actual Testscripts are written in the Testscripts package.

8) while executing the testscripts it will take the test Data from the excel file with the help of apache file jars. & performs action on the GUI Application by calling ~~resseenesse~~ methods present in the POM classes.

9) Once it executes the test Method it will executes the logout code which is present in the @AfterMethod Annotation.

10) Like this it will executes all the testcases one after the other with the help of Batch runner. (TestNG.xml file)

11) After the execution of all the testcases it will closes the browser.

12) Since we are using TestNG it will automatically generates the default HTML Report (Test output folder) which contains no of Test cases passed, failed and skip etc

13) Since we have implemented the Listeners feature of TestNG it will automatically takes the screenshot of failed Test cases. ⊕ in the Screenshot folder.

Keyword Driven Framework:

A keyword DF is one in which the actions are associated with keywords. and kept external files ex) lauching browser launchBrowser().

Parallel Execution: \ Cross browser Testing

i) Executing the multiple test scripts with multiple browsers concurrently or at the same time is called as parallel execution.

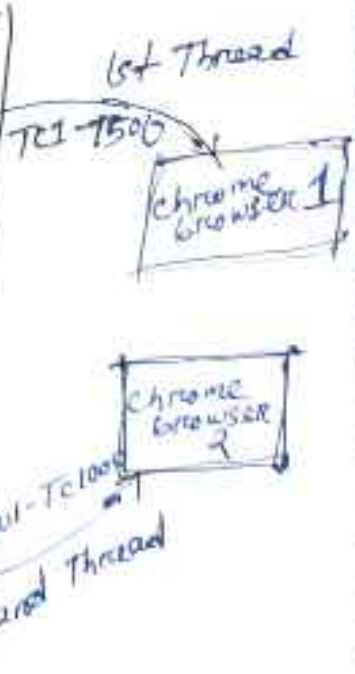
ii) TestNG provides two types of parallel execution
   1) Distributed parallel execution
   2) compatibility Parallel execution (cross browser Testing)

Distributed Parallel Execution:

i) In real time if we execute all the testscripts sequencially (1000 test cases) it will take more time to get the result

ii) In order to get the result in early stages we should go for distributed parallel execution

```
<suit name = suite Parallel = 'tests'>
  <test thread count = 2 name = 'test 1'>
    <classes>
      TC 1 ... TC 500
    </classes>
  </test
  <test thread count = 2 name = 'test 2'>
    <classes>
      TC 501 ... TC 1000
    </classes>
  </test
</suite>
```



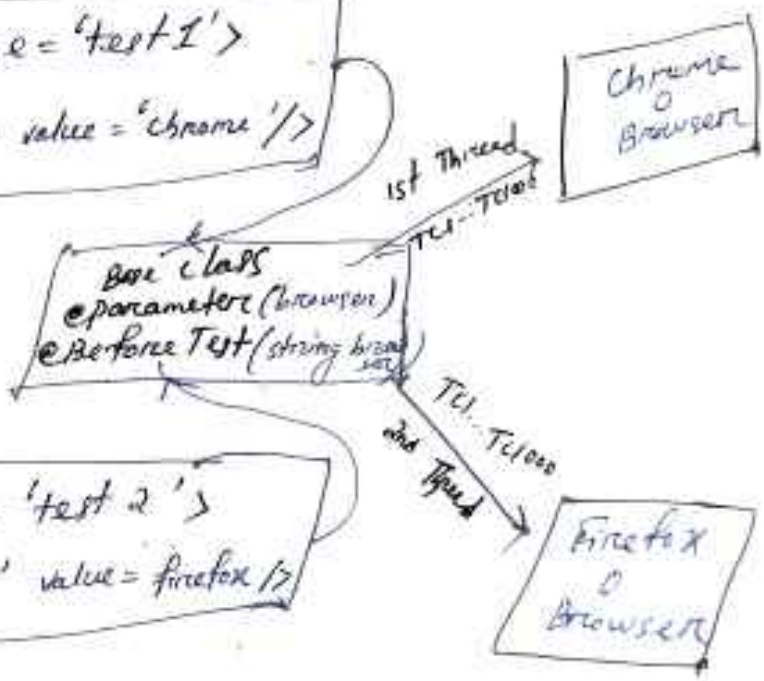
Note :-

i) To perform distributed parllal Execution. We should change or customize the testNG. xml suit file. as shown in previous page.

ii) Here we should create multiple test block (Test Runner) and distribute the test script and use the attribute parllal = 'tests' in the suit tag

Compatibility Parllal Execution (cross browser Testing)

```
<suite name = 'suite' Parllal = 'tests' >
  <test thread-count='2' name = 'test1'>
    <parameter name = 'browser' value = 'chrome' />
    <classes>
      TC1 -- TC1000
    </classes>
  </test>
  <test thread count = '2' name = 'test 2'>
    <parameter name = 'browser' value = firefox />
    <classes
      TC1 ... TC1000
    </classes>
  </test>
</suite>
```



To perform cross browser Testing we should create multiple test blocks and use browser parameter per each test block. so that whenever we execute the xml file each test block lunches the specific browser using threads.

Then executes all the test scripts in different browsers parallaly at the same time. In case of parllel execution it is mandatory to use @parameter Annotation along @BeforeTest Annotation in the base class

@parameter annotation will be used to recive the browser parameter from xml file into the base class.

Note:-

i) In real time to perform cross browser testing we will use a separate tool called selenium grid.

ii) Selenium grid is a another automaton tool present in selenium community which is used to run in another virtual machines.

Note

It is mandatory to use BeforeTest for opening the browser instead of before class in case of parllel execution.

Q// Difference bet? @BeforeTest & @Before Method

→ @Before Method will be executed before the execution of every @Test Method

→ whereas @Before Test will be executed Before the execution of the test block present in the suit file

Run Time Polymorphism in selenium :-

```
class RunTimeP {
static {


}
}
    public WebDriver driver;
    p s v m (s[] args) {
    s.o.pln("Enter the browser name");
    Scanner sc = new Scanner (System.in);
    String browser = sc.nextLine();
      if (browser.equals("Chrome")) {
          driver = new ChromeDriver();
      }
      else if (browser.equals("firefox")) {
          driver = new FirefoxDriver();
      }
```

```
driver.get("https://www.google.com/");
String title = driver.getTitle();
s.o.println(title);
driver.close();
}
}
```

Fluentwait

i) In order to change the polling period from 500ms to any time we go for fluent wait.

ii) It is not widely used in the industry because of complexity in the syntax of Fluentwait

SYNTAX 1:

Instead of implicit wait → `Fluentwait wait = new Fluentwait<>(driver);`

```
wait.withTimeout(Duration.ofSeconds(10)).pollingEvery
(Duration.ofSeconds(1));
```

SYNTAX 2:

Instead of explicit wait → `Fluentwait wait = new Fluentwait<>(driver);`

```
wait.withTimeout(Duration.ofSeconds(10)).pollingEvery
(Duration.ofSeconds(1)).
until(ExpectedConditions.titleContains("Enter"));
```

// How to handle Autocomplete google places drop down wherever we can't inspect the location then how to handle:

```
class A
static { };
p s v m (s [] args){{
WebDriver d = new ChromeDriver ();
    d. get ("https://www.twopluges.com/");
d. findElement (By.xpath ("//a[text()='Live Posting']")).click();
WebElement sb = d.FE (By.id ("autocomplete"));
  sb. clear ();
  sb. sendkeys ("Toronto");
String text;
do {
    sb. sendkeys (Keys. Arrow_Down);
  text= sb. getAttribute ("value");
    if (text. equals ("Toronto, OH, USA")) {
        sb. sendkeys (keys. Enter);
        break;
    }
  while (! text. isEmpty ());
  }
}
```

1) How to select specific checkbox & all the checkboxes : https:

```
class A {
static {
};
P.S V m(S[] args) {
WebDriver d = new ChromeDriver();
d.get("https://itera-qa.azurewebsites.net/home/automation");
d.findElement(By.xpath("//input[@id='monday']")).click();

// select All the checkbox
for (WebE
List<WebElement> cb = d.fES(By.xpath("input[@type='checkbox' and contains(id,'day')]");
for (WebElement w : cb) {
w.click();
}
}
```

// How to select last two checkboxes.

```
int totalCb = checkbox.size();
for(int i= totalCb-2; i<totalCb; i++)
{
checkboxes.get(i).click();
}
}
}
```

// How to select 1st Three checkboxes;

```
for(int i=0; i<totalCb; i++){
if(i<2)
{
checkboxes.get(i).click();
}
}
}
```

Broken Link

if the code is greater than 400 then it is broken link;

Authentication Pop-up :

When user access any protected web url, an authentication pop-up is displayed to enter credentials.

How to handle :

To handle the authentication pop-up we can pass the username and password along with the web page's url

SYNTAX : http://username : password URL

```
class A
@ static { path };
p s v m (s [ ] args) {
WebDriver d = new ChromeDriver ();
d. get("http://admin : admin @ the - internet. herokuapp
                    . com / basic - auth");
d. close();
```

How to handle slider :

By using Drag and drop By ( ) method;

```
d. get("https://jquerysciript...");
WebElement s1 = d. findElement (By. xpath("//span[2]"));
Actions a = new Actions (d);
a. dragAndDropBy (s1, 100, 0). perform();
```

Q How to capture tooltip of the webElement.

```
WebDriver d = new ChromeDriver();
d.get("url");
String title = d.findElement(By.LinkText("Them")).getAttribute("title");
s.o.println(title);
```

Q// How to open the link in new tab;

By using contextClick method of Actions class then use Robot class. OR

We can use keys class methods like →

```
d.get("http://demo.nopcommerce.com/");
String s = Keys.chord(Keys.CONTROL, Keys.Return);
d.FE(By.LinkText("Register")).sendKeys(s);
```

Q// How to open multiple url in multiple tab or in new window.

By using driver.switchTo().newWindow(WindowType.window);
.newWindow(WindowType.TAB);

```
d.get("url");
```

It will open in new windows and new Tab;

Q/ How to capture all cookies, add cookies, delete cookies from the browser.

By using driver.manage().getCookies(); Set<Cookies>
driver.manage().addcookies();
driver.manage().deleteCookie(obj)
driver.manage().deleteAllCookies();

for adding we have to create cookie object

```
Cookie c = new Cookie("Ajit", "1234");
driver.manage().addCookies(c);
Set<Cookie> CK = driver.manage().getCookies();
for(Cookie cl: CK){
    s.o.println(cl.getName()+cl.getValue());
}
```

Interview Q?

1) What are some limitations of Selenium?
   i) We can not test desktop application using selenium
   ii) We can't test web services.
   Web services are headless so we can not test.

2) Disadvantage of Absolute x-path?
   if their is slightest change in the UI or any element the whole absolute x-path fail.

3) Which x-path you will prefer to use? Why?
   i) we prefer Relative x-path
   ii) Because it identify element even after some UI changes.

4) Difference bet'n xpath & css Selector?
   using x-path we can traverse top to bottom or bottom to top. whereas using css selector we can only move downward.

5) How to switch bet'n multiple windows?
   d.switchTo().window("{windowHandleName}")

6) For refresh the page we can use also.
   sendKeys(Keys.F5);

Q// How to check which option in the drop down is selected.

By using isSelected() method

```
Select s= new Select(d.fE(By.id("continys")));
s.selectByVisibleText("india");
s.o.println(d.fE(By.id("india")).isSelected());
```

Q// HTMl Unit Driver ?

HTMl Unit Driver is the fastest web Driver, it is non-GUI, while running no browser gets launched

SQ// How can we make sure a test method runs even if the test methods which it depend fail or skipped;

```
@Test
public void parentTest(){
Assert.fail("failed Test");
}
@Test(dependsOnMethods={"parentTest"}, alwaysRun=true)
public void dependsTest(){
S.o.println("Running");
}
```

Q// which seenarios we can't automate ?

captcha, Bar code, image, word / pdf

Q// How to set the size of browser window using Selenium.

By using setSize() method.

```
s.o.pln(d.manage().window().getSize());
Dimension d = new Dimension(420, 600);
d.manage().window().setSize(d);
```

Q// Advantage & benifits of Automation Testing ?

i) Save Time & money & it is faster in execution
ii) Reusability of code, create one time and execute multiple time.
iii) Low-cost maintenance.
iv) Maximum test coverage

Q// What are the open source frameworks supported by Selenium WebDriver ?

1) Junit 2) TestNG 3) Cucumber 4) JBehave.

Q// Why do you prefer Selenium Automation tool ?

1) Free and open source
2) Cross browser compatibility
3) Platform compatibility
4) Multiple programming language support

Maven :

i) Maven is a build tool which is provided by Apache software foundation. It is open source

ii) It will provides project structure and pom.xml → dependencies → downloading for libraries & drivers
plugin → configuration.
→ Reports, Documentation, packaging

iii) When we download maven it will internally create maven local repository. → from Remote maven repository it will download all the third party libraries

pom.xml

<dependencies>
all libraries version update → To upgrade we have to modify the version number in dependency
</dependencies>

<plugin>
→ configuration project related
It will organize complete project
</plugin>

go to file - click on new then - other - maven - maven project → next

For adding dependencies we have to go MvnRepository.com search TestNG → click on latest version copy and paste in dependencies

Note: Whichever the dependency or Libraries we need we just add to the our project dependencies.

Maven Build Life Cycle:

i) Validate
ii) Compile
iii) Test
iv) Package
v) Integration Test
vi) Verify
vii) Install - Local App
viii) Deploy - Install in remote repository that we can use by project