INDEX

* Ref → 08-09/01 ↗ Month (Dates)

S.No	CONTENTS (BASIC SELENIUM)	Ref.
08 1	Introduction to Automation, Advantages, Disadvantages	08-09/01
2	Intro to Selenium, Versions, Different Tools used	10-12/01
09 3	Types of Applications	12/01
4	Java - Selenium Architecture,	14/01
10 5	First Ever Selenium Test Script	15/01
6	WebDriver Architecture & Example	16,17/01
11 7	WebDriver Abstract Methods with Example	18-25/01
8	Intro to HTML, Basics, HTML Source Codes	25-30/01
12 9.	Inspection Methods, Locators & its types,	31/01-25/02
* 10.	Test Case to LOGIN into an Application to Test Script	26-28/02
01 11	Synchronization, Types, Examples	28/02-07/03
12	Handling Dropdown	07-12/03
02 13	Handling Duplicates, Dynamic Dropdown	13-18/03
14	Handling Keyboard and mouse hover actions	18-22/03
03 15	Handling Scroll Down	23-27/03
16	Handling Disabled Element	27-29/03
* 04 17	Taking Screenshots	30-3 to 04-4
18	Handling Web Table	04-06/04
05 19	Handling Pop-ups	08-17/04
* 20	Handling Frames	17-20/04
21	Handling new window | tab | child browser popup	20-25/04
22	Methods of WebElement Interface	25-30/04
07		
	ADVANCED SELENIUM	
1.	Automation Framework : Advantages, stages, Types	02-03/05
2.	Data-Driven Framework	03-13/05
3.	Key-word-Driven Framework	14-24/05
4.	StaleElementReferenceException, Page Object Model (POM)	24-28/05
5.	Method-Driven Framework	28/05-01/06
6	TestNG : Introduction	01-05/06
7.	Flags of TestNG	06-11/06

CONTENTS

S.No		Ref.
08 8	Annotations of TestNG	11-18/06
9	Assertion in TestNG	18-20/06
09 10	Data-Driven Concept, Grouping Execution in TestNG	20-27/06
11.	Data Parameterisation in TestNG	28/06
10 12	Parallel Execution in TestNG	29/06-06/07
13.	Modular Framework with Example	08-17/07
11 14.	Hybrid Framework Architecture & Notes	17-29/07
15	GitHub	29/07-01/08
12 16.	Maven	01-06/08
17.	Jenkins	06-09/08

AUTOMATION TESTING

Process of testing the Functionality of an Application by the help of automation tools is known as Automation Testing.

(OR)

Converting manual test Cases into automation Scripts, executing them through a framework and getting test results is known as Automation Testing.

Q: Why do we switch to Automation Testing? (or)
* Advantages of Automation Testing.

→ We switch to Automation Testing when we have to do regression testing.
→ While regression testing, there are lot of repetitive tasks and performing them manually is a boring job.
→ Also in regression, when the product size increases, time taken to test also will increase. Hence we switch to automation to reduce the time taken for testing.
→ To reduce the manpower / number of resources
→ To obtain the test results faster.
→ To get a quick ROI (Returns on Investment).
→ We can expect Accurate results.
→ Due to tough Competition, Software Companies need to deliver high quality product within less time. Hence we switch to automation testing.

* Drawbacks / Limitations of Automation Testing:
→ 100% Automation testing is not possible
→ Anything which is Completely dynamic Can'be automated.

→ Anything which requires manual intervention (interaction) cannot be automated eg.
What all cannot be automated ...

1. OTP, Capteha
2. Making payments through swiping the debit/ credit cards.
3. Barcode Scanner, QR Code Scanning
4. Animation
5. Testing the quality of audio/ video files
6. Game testing.

→ we need skilled resources to do automation testing.

→ Cost involved in automation testing is more compared to manual testing, because tools are licenced and automation skilled resources are paid higher.

→ We can't automate an unstable product, we have to wait for the product to become stable.

Note : Newly built features cannot be automated.

SELENIUM AUTOMATION TOOL

26.08.21 D2

Q: What is Selenium?

Selenium is a free, open-source automation tool. It can automate web-based application, but can't automate stand-alone and client-server applications.

Appium Tool : Used to automate mobile apps.
Winnium Tool, AutoIT, QTP : Used to automate stand alone apps
QTP : Quick Test Professional

→ Selenium jar file can be downloaded from https://www.selenium.dev/downloads.
→ Source Code is visible from https://github.com/SeleniumHQ/Selenium
https://github.com/selenium/blotmarker/java/client/Src/org/openqa/selenium.

Quick Notes:

Developed by - Jason Huggins
Year - 2004
Company (worked in) - Thoughtworks, Chicago, USA
Former Name - Javascript Test Runner
Renamed as - Selenium
Reason (for renaming) - To Compete with HP - Mercury Automation Tool.

Selenium Versions / Components / Flavours:

1. Selenium Core
2. Selenium IDE - Record and playback tool
3. Selenium RC (Remote Control), also called as Selenium 1.0.
4. Selenium Webdriver (2007) / Selenium 2.0.

Current Stable Version – 3.141.59
Upcoming (Alpha) – 4.0

5. Selenium Grid

Note:
Selenese : Selenium Commands which are used in Selenium IDE

→ Se Supports all the programming languages – Java, C#, Javascript, Perl, Ruby, Python, R, TCL, Elixir, Haskell.

→ Se Supports all the browsers: Google Chrome, Firefox, Opera, Safari

→ Se Supports all the Operating Systems: Windows, Mac, Linux, except for Unix.

TOOLS:

* Automation Testing Tools:
Selenium, QTP, Test optimize, Cypress, Silk Test, Appium, Winnium, Test Complete, Auto IT, TOSCA

* Defect Tracking Tools:
Jira, Bugzilla, Bugzero, Mantis, Bug-genic, rational clear quest.

* Test Management Tools:
Test Rail, Jira, QTest, Test Manager, Qc (quality Center), ALM (Application Life Cycle Management)

* Performance Testing Tools :

LoadRunner, JMeter, Neoload, Web load.

* Read Softwares & Files for Automating :

JDK, IDE (Eclipse), Selenium jar file.

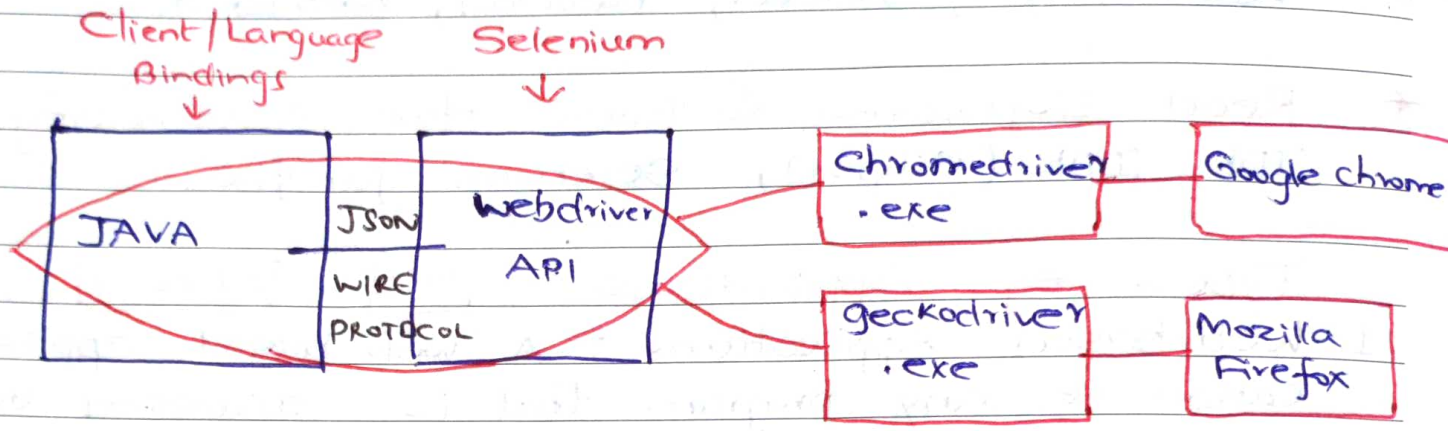
Types of Applications : D3 27.08.21

1. Web based Applications : A web based application is any program that is accessed over a network Connection using HTTP, rather than existing within a device's memory. It often runs inside a web browser.
Web based applications are also known as Web Apps.

2. Client Server Applications :
It Consists of a client program that Consumes Services provided by a server program. The Client requests Services from the Server by Calling functions in the Server application.

3. Stand Alone Applications :
It is an application that runs locally on the device and doesn't require anything else to be functional. All the logic is built into the app, So it doesn't need an internet Connection nor any other services installed. These type of applications are not bound to any specific platform.

JAVA - SELENIUM ARCHITECTURE :



JSON - Javascript object Notation
API - Application programming Interface.

→ Selenium webdriver API supports all the programming languages and it Communicates with language bindings like java by using JSON wire protocol.

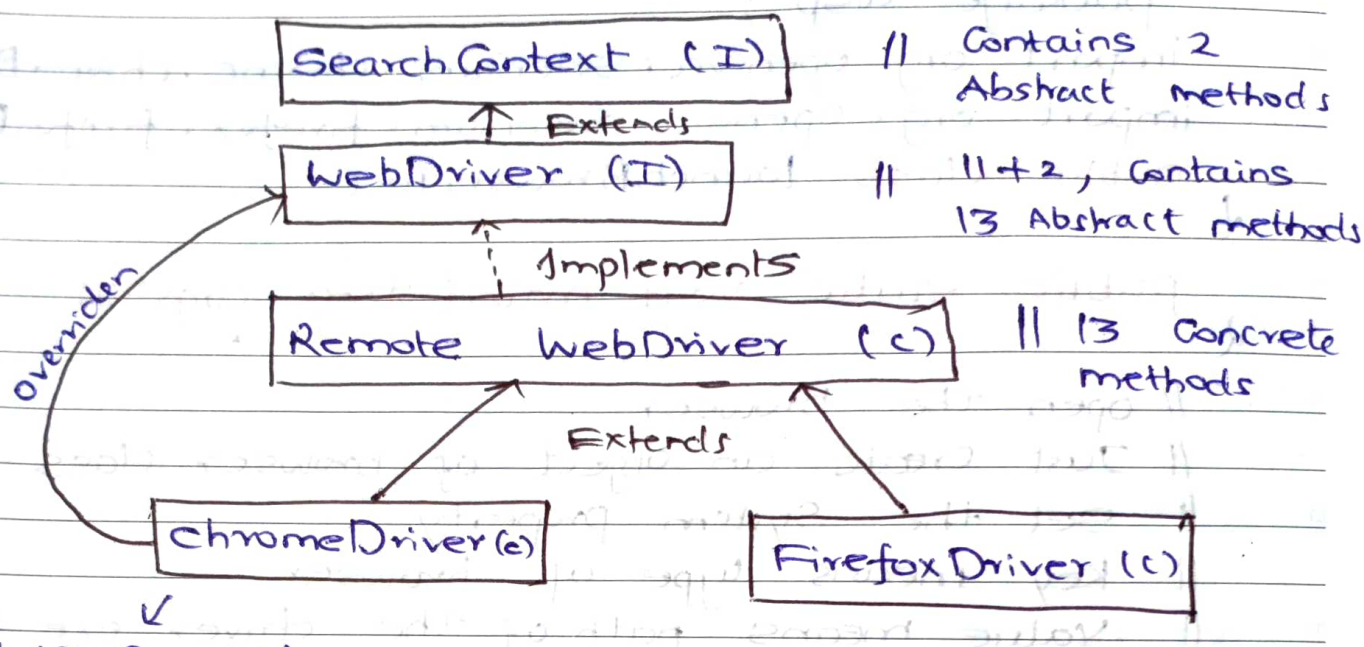
→ These two Combined to Communicate with the browsers by using respective driver executable files like Chrome driver.exe, geckodriver.exe etc.

Q. Opening browser and closing it through automation.

```
package arsp ;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
public class LaunchBrowserTest
{
public static void main(String args[])
{
// open the browser
// Just create an object of browser class
// set the System property
// key means type of browser
// value means path of the driver.exe file
System.setProperty("webdriver.chrome.driver",
"D:\\KCSMG\\AutomationProject\\drivers\\chrome
driver.exe");
new ChromeDriver().close();
System.setProperty("webdriver.gecko.driver",
"D:\\KCSMG\\AutomationProject\\drivers\\geckodriver
.exe");
FirefoxDriver driver = new FirefoxDriver();
driver.close();
}
}
```

O/P : Info: Detected dialect: W3C

WEB DRIVER ARCHITECTURE



// 13 Overridden Browser classes close()

Explanation: SearchContext is the super most interface, WebDriver interface extends it and all the 13 abstract methods of WebDriver are given implementation in RemoteWebDriver class and all those Concrete methods are overridden in respective browser classes like ChromeDriver class, FirefoxDriver class etc.

→ As per Se standards, we always upcast our browser classes to WebDriver interface to achieve

1. Generalization (not specifying any type)
2. Runtime Polymorphism (At runtime, we Can decide in which browser my code will run)
3. To get all the 13 methods required for

Automation Testing.

Q: How to open the browser ?

To open any browser, we need to create an object of that browser class.

```
new ChromeDriver().close();
```

Runtime Polymorphism: (program)

```
package arsp;
import java.util.Scanner;
import org.openqa.selenium.webdriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.firefox.FirefoxDriver;

public class RunTimePolymorphismInSelenium
{
public static void main(String args[])
{
System.Out.println(" Enter Browser Name: ");
Scanner Sc = new Scanner(System.in);
String str = sc.next();
WebDriver driver = null;

if (str.equals(" chrome"))
{
System.SetProperty (" webdriver.chrome.driver",
".\\drivers\\chromedriver.exe");
driver = new ChromeDriver();
}
```

```
        else if ( str. equals ( " Firefox "))
    {
        System. SetProperty ( " webdriver. gecko. driver",
            ". \/ drivers/ geckodriver. exe ");
        driver = new    FirefoxDriver ;
    }

    else
    {
        System .out. println (" Give proper browser name");
    }
        driver . close();
    }
}
```

03.09.2021 D6

WEBDRIVER ABSTRACT METHODS
(Browser window Related Methods)

Method Type	S.No	Method Name	Return Type	Method Use
Initiation Method (1)	1	get(String Url)	Void	Used to enter URL
Page Verification Methods / Data Capture Methods (2-4)	2	getCurrentUrl()	String	Used to get the URL of the Current web page.
	3.	getPageSource()	String	Used to get the Source Code of the Current web page
	4	getTitle()	String	Used to get the title of the Current web Page

Method Type	S.No	Method Name	Return type	Method Use
Data Capture Methods (5 & 6)	5	getWindowHandle()	String	Used to get the window handle of the Current browser window
	6	getWindowHandles()	Set<String>	Used to get window handle of all the browser window
Browser window Handling method (7)	7	manage()	Options	Used to manage the browser window.
Navigation Method (8)	8	navigate()	Navigation	Used to navigate from one page to another previous page, next page, Can refresh current webpage
Inspection methods (9 & 10)	9	findElement(By arg)	Web Element	Find the particular element on the web page
	10	findElements(By arg)	List <web Element>	Find multiple elements on the webpage.
Control Switching method (11)	11	SwitchTo()	Target Locator	Used to switch over location from webpage to pop-ups frames, windows etc
Termination Methods (12 & 13)	12	Close()	Void	Close the Current browser window
	13	quit()	Void	Close all the browser windows opend by Selenium

Program on 13 Methods - Concept

```
package asp;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.Point;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class WebDriverInterfaceMethods
{
  public static void main(String args[]) throws
      InterruptedException
  {
    System.SetProperty("WebDriver.chrome.driver",
        "./drivers/chromedriver.exe");

    webdriver driver = new ChromeDriver();
    driver.get("https://www.google.com/");
    String pageURL = driver.getCurrentUrl();
    System.out.println(pageURL);

    if (pageURL.equals("https://www.monsterIndia.com/"))
    {
      System.out.println("MonsterIndia URL is
          entered, PASS");
    }
    else
    {
      Sop("monster India URL is not entered, FAIL");
    }
    String Pgsrc = driver.getPageSource();
    Sop(pgsrc);
```

```
if (pgsrc . Contains( " Gmail "))
{
  Sop (" Gmail link is present, PASS");
}
else
{
  Sop (" Gmail link is not present, FAIL");
}
String title = driver. getTitle();
  Sop ( title);
if ( title. equals ( " Google "))
{
  Sop(" Google page is displayed, PASS");
}
else
{
  Sop(" Google page is not displayed, FAIL");
}
driver. manage(). window(). maximize();
  Thread . sleep (3000);
Dimension d = new Dimension (300, 400);
driver. manage(). window(). setsize (d);
  Thread . sleep (3000);
Point P = new Point (600, 400)
  driver . manage(). window(). SetPosition (P);
}
}
```

DIFFERENCE BETWEEN get() & navigate()

get()	navigate()
1. Enter URL	1. Enter URL Navigate to Previous page Navigate to Next page Refresh the Current page
2. get() will not take advantage of Brauser history	2. navigate() will take advantage of browser history
3. get() will wait until Complete page is loaded.	3. navigate will not wait until the Complete page is loaded

HYPER TEXT MARKUP LANGUAGE (HTML)

Scripting language used to develop web element.

In HTML, there are 3 required things.

1. Tag
2. Attribute
3. Text

Rules:

→ tag and attribute should be within <>
→ Text should be outside <>
→ Always start from <html>
→ Once a tag is opened, close it by using /tag.

Any editor – Save as .html

```
<html>
<head>
<title> My First Page </title>
</head>
<body>
<a href = 'https://www.facebook.com/'>
facebook </a>
</body>
</html>
```

Text

Link

Anchor tag

Text

→ href is an attribute of anchor tag [hyper text reference].

It is used to specify the URL of the page that the link goes to this attribute is used with anchor <a> tag

→ <a> - the anchor tag is used to create a hyperlink on the webpage. This hyperlink is used to link the webpage to other webpage. href attribute which indicates link's destination.

```
<a href = 'https://www.w3schools.com' target = '_blank'
visit W3Schools.com </a>.
```

attribute

Link

BASICS OF HTML

Tag	Description
<!DOCTYPE>	Defines the document
<html>	Defines html document
<head>	Contains meta data / information for document.
<body>	Defines document body
<h1> to <h6>	Defines HTML heading
<p>	Defines a paragraph
 	Inserts a single line break
<hr>	Defines a thematic change in the Content.
<!--,,-->	Defines a Comment

FORMS AND INPUT

Tag	Description
<input>	Defines an input Control
<text area>	Defines a multiple input Control (text area)
<button>	Defines a Clickable button.
<select>	Defines a dropdown list
<option>	Defines an option in dropdown.
<label>	Defines label for an <input> element.

Code – paragraph

<h1> heading 1 </h1>
<h2> heading 2 </h2>
⋮
<h6> heading 6 </h6>

Bold : Statements --

Italic : Statements --

Small (text) tag : <small> Statements </small>

Underline : <u> Statements </u>

image : <img src = " imagename.jpg" width = "40" height = "40" border = "0" >

link : <head>
<link ref = " stylesheet" type = "text" href = " style.css" >
</head>

Radio button :
<form method = post action = " /cgib/example.cg.i" >
Select an option :
<input type = "radio" name = "option" > option 1
<input type = "radio" name = "option" > option 2

Program:

```
<html>
<head>
<title> My signup page </title>
</head>
<body>
<h1> This is my page </h1>
<h2> This is my page </h2>
<h3> This is my page </h3>
<h4> This is my page </h4>
<h5> This is my page </h5>
User Name <input type = 'text'> </input> <br>
Password <input type = 'password'> </input> <br>
<span> Choose your gender </span> <br>
<input> type = 'radio' name = 'gender'> MALE
</input> <br>
<input> type = 'radio' name = 'gender'> FEMALE
</input> <br>
<input> type = 'radio' name = 'gender'> OTHERS
</input> <br>

<div> choose subjects, already completed </div>
<input type = 'checkbox'> Manual Testing </input><br>
<input type = 'checkbox'> SQL </input><br>
<input type = 'checkbox'> Core Java </input> <br>
<input type = 'checkbox'> Automation </input> <br>
<input type = 'button' value = 'signup'>
</input> <br>
</body>
</html>
```

INSPECTION METHODS

Before performing actions on any element, first we should locate the element on webpage.
Hence to find its address we have to find element using findElement() and findElements() inside webdriver interface.

findElement() - method of webdriver interface is used to get the address of first matching element on the webpage.

- Return type of findElement() is WebElement which is an interface.

- If findElement() is not able to find the element on the webpage then we get NoSuchElementException.

- findElement() takes locators as its argument

Locators :

- Locators are Static methods of abstract 'By' class.

- Locators are used to locate one or more elements on the webpage

- Locators act as argument for findElement() and findElements()

Types of Locators :

1. tagName (String arg)
2. id (String arg)
3. name (String arg)
4. className (String arg)
5. linkText (String arg)
6. partialLinkText (String arg)
7. cssSelector (String arg)
8. xPath (String arg)

HTML Code to Create a Link :

```
<html>
<body>
<a href = 'https:// www.google.com' id = 'i1',
name = 'n1' class = 'c1' value = 'v1' title = 't1'>
Google </a>
</body>
</html>
```

In the above Code, to create a link, we use <a> tag and mandatory attribute Called 'href' but other attributes like id, name, class, value, title are Optional.

So, in Selenium, we can use features like tag, attributes like id, name and class etc.

So, according to the above example,

1. tagName("a") - we use tagname of the element
2. id("i1") - we use id of the element
3. name("n1") - we use name of the element
4. ~~class Name~~ ("Google") - Here we use text but linkText make sure element is link only
5. className("c1") - we use class of the element
6. partialLinkText("Go") - Here we use a part of the complete link text.

Note: Through partialLinkText() locator, we handle partially dynamic linktext.

Program:

```
package asp;
import org.openqa.Selenium.By;
import org.openqa.Selenium.WebDriver;
import org.openqa.Selenium.WebElement;
import org.openqa.Selenium.Chrome.ChromeDriver;

public class InspectionMethods
{
public static void main(String[] args)
{
  System.SetProperty("webDriver.chrome.driver",
    "./drivers/chromedriver.exe");
  WebDriver driver = new ChromeDriver();
  driver.get("file://D://KCSMG/HTML/link.html");
//on the browser, find the element with
  tagname as 'a'.
  WebElement linkAddress = driver.findElement
                (By.tagname("a"));
```

```
// on the address, perform click action
    linkAddress.click();
// on the webpage, find the element with
the id as 'i1' and click on it.
driver.findElement(By.id("i1").click();
driver.findElement(By.name("n1").click();
driver.findElement(By.class("c1").click();
driver.findElement(By.linkText("Facebook")).
                                    click();
driver.findElement(By.partialLinkText("Fa")).click();
driver.findElement(By.cssSelector("a[title='t1']")).
                                    click();
driver.get("https://www.google.com/");
driver.findElement(By.xpath("//input[@name='q1']")).
                                    sendkeys("raghav");
}
}
```

7. CssSelector :

Here we identify any element by giving both tag and attribute.

Syn: tagName[AttributeName = 'AttributeValue']

Eg:

```
a[href = 'https://www.google.com]
a[id] = 'i1']
a[name] = 'n1']
a[class = 'c1']
a[value = 'v1']
a[title = 't1']
```

Shortcuts: Shortcut for id is #

div[id=i1] (or) div#i1 (or) #i1

Shortcut for class is .

div[class = c2]

div. c2

.c2

To verify the cssSelector expression and xpath expression, we download an add-on for FIREFOX browser called Try Xpath.

Steps:

→ open Firefox browser.

→ Go to Google and type 'download try xpath for firefox'.

→ Click on 'Add to Firefox' button and click on 'Add' button.

→ Try xpath will be installed and visible at top right corner of firefox browser.

→ click on Tx icon on top right corner.

→ For verifying cssSelector expression, change way to 'QuerySelectorAll'

→ write the cssSelector expression and click on execute button.

Limitation of cssSelector:

To use cssSelector minimum 1 attribute should be present which is unique.

If no attributes are present, we can't use cssSelector because cssSelector does not support text.

Note: Refer program on left page for example.

8. xpath :

→ xpath is the path travelled in the HTML tree to find an element.

→ xpath is one of the locator which covers all possible ways to find an element

There are two types of xpath
1. Absolute xpath
2. Relative xpath

Absolute xpath means complete path we travel from Start (html) to element -->
- It is achieved by '/' (single forward slash) --> immediate child / immediate descendant.

HTML Code

```
<html>
<body>

<div>
A <input type = 'text'> </input>
B <input type = 'text' > </ input>
</ div >

< div>
V < input type = 'text' > </ input >
D <input type = 'text' > </ input >
</ div >

</ body>
</ html>
```

HTML Tree Structure

```
html[1]
    body[1]
        div[1]
            input A[1]
            input B[2]
        div[2]
            input V[1]
            input D[2]
```

Note: In realtime applications, if we use absolute xpath, it will become very lengthy. So we use relative xpath.

Relative xpath - Shortest path --> achieved by '//' (Double forward slash) --> Any child / Any descendant.
- It is the shortest path in the HTML tree (Directly jump to required element)

Elements	Absolute Path
ABVD	html/body/div/input
AB	html/body/div[1]/input
A	html/body/div[1]/input[1]
AV	html/body/div/input[1]
BV	html/body/div[1]/input[2] ; html/body/div[2]/input[1]
AD	html/body/div[1]/input[1] ; html/body/div[2]/input[2]

Elements	Relative xpath
ABVD	//input
AB	//div[1]/input
A	//div[1] input[1]
AV	//input[1]
BV	//div[1]/input[2] ; //div[2]/input[1]
AD	//div[1]/input[1] ; //div[2]/input[2]

Note: Refer 04 Feb page for program.

15.09.21 D10

Cases of Relative xpath:

Case 1: Xpath by unique attribute

Syn: //tagName[@AttrName = 'AttrValue']

Eg 1: //input[@name='q'] (Google.Com)

Eg 2: //@[@href = 'https://www.facebook.com/praveendra~~...~~music/'] (https://www.yuvadhwaja.in/index.html)

Eg 3: //img[@Src = 'images/Kunwar Singh_Kannada.jpg'] (https://www.yuvadhwaja.in/index.html)

Eg 4: //input[@name = 'pwd'] (demo.actitime.Com)

Case 2: xpath by text() function

Syn: //tagName[text() = 'textvalue']

Eg 1: //div[text() = 'login']

Eg 2: //small[text() = 'Now']

Eg 3: //a[text() = 'Gmail']

Case 3 : xpath by multiple Attributes

~~<div fn = 'Deepika~~

Syn:

Unique Element (and) :

// tagName[@attr1 = 'attrval' and @attr2 = 'attrval' and ...]

Many Elements (or) :

// tagName [@attr1 = 'attrval' or @attr2 = 'attrval' or ...]

Concept :

< div fn = 'Deepika' ln = 'Rai'>

< div fn = 'Aishwarya' ln = 'Rai'>

// div [@fn = 'Deepika'] // 2 Counts (but Required only 1)

// div [@fn = 'Rai'] // 2 Counts (")

So, // div [@fn = 'Deepika' and @ln = 'Rai'] // 1 Count

↳ i.e. xpath by multiple attributes to achieve 1 Count or uniquely one element.

Eg 1 : // input [@type = 'password' and @name = 'pwd']
(demo.actitime.com)

Eg 2 : // a [@href = 'index.html' and @class = 'nav-link']
(https://yuvadhwaja.in/)

Eg 3 : // div [@class = '_4IiNRh_2mtKou' and @title = 'core i3'] (flipkart.com)

Eg 4 : // div [@class = '_1GEhLW' and text() = 'Core i7']
(flipkart.com)

Combination of Text and an attribute :

10 Sunday

Syn: // tagName [@AttrName = 'Attrvalue' and text() = 'textvalue']

Eg: // a [@href = 'index.html' and text() = 'Home']

Case 4 : xpath by Contains() function

Syn: // tagName[Contains(@AttrName, 'Attrvalue')]

Situation:

```
< div id = 'ui-id-1' >
< div id = 'ui-id-2' >
< div id = 'ui-id-3' >
< div id = 'ui-id-4' >
< div id = 'ui-id-5' >
```

Raghavendra equals Raghavendra
Raghavendra Contains Raghav

Write xpath expression to match all the above five:

1. a) Handle partially Dynamic Elements (partially dynamic attribute value)
   Eg: // div [Contains(@ id, 'ui-id')]

1. b) Handle partially Dynamic Elements (partially dynamic text value)
   Eg: //div [Contains(text(), 'we have')]
   for the example, we have 1,24,456 products

2. Avoid writing lengthy text
   Eg: //p [contains(text(), 'Uttara Karnataka')]

3. Avoid spaces present in the values
   Eg: //div [Contains (text(), 'Login')]

Case 5 : xpath by axis (relationship)

HTML TREE

```
html
    body
        table
            tbody
                tr id='i1'
                    td A
                    td B
                    td C
                    td D
                tr id='i2'
                    td E
                    td F
                    td G
                    td H
```

parent --> child

// parent / descendant :: which child

① html --> body

// html / descendant :: body

② table --> tr (i1)

// table / descendant :: tr[@id='i1']

Child --> parent

// child / ancestor :: which parent

① B --> tr(i1)

// td [text() = 'B'] / ancestor :: body

② F --> body

// td[text() = 'F'] / ancestor :: body

From one element to next element of same ~~element~~ parent.

// element / following-sibling :: which next element

① A --> D

// td[text() = 'A'] / following-sibling :: td[text() = 'D']

From one element to previous element of same parent

// element / preceding-sibling :: which previous element

① C --> A

// td[text() = 'C'] / preceding-sibling :: td[text() = 'A']

② i_2 --> i_1

// tr[@id = 'i_2'] / preceding-sibling :: tr[@id = 'i_1']

Cousin

B -> F

// td[text() = 'B'] / ancestor :: tr[@id = 'i_1'] / following-sibling :: tr[@id = 'i_2'] / descendant :: td[text() = 'F']

ASSIGNMENT 1: on Case 1: xpath by unique attribute

Syn: // tagName[@AttrName='AttrValue']

1. //input[@id='identifierId'] (Gmail.com)
2. //input[@placeholder='password'] (Facebook - login page)
3. //input[@name='verification'] (https://www.patanjaliayurved.net/customer/login)
4. //input[@name='Session[password]'] (twitter login page)
5. //input[@id='SignInsubmit'] (Amazon prime video login page)
6. //input[@id='ap_password'] (Amazon prime video login_passwd)
7. //input[@placeholder='Enter your mobile number'] (https://www.hotstar.com/in/subscribe/signin?returnURL=/in/subscribe/my.account)
8. //label[@class='checkbox'] (https://zoom.us/signup)
9. //a[@href='/signup'] (https://zoom.us/signin)
10. //input[@id='join-Confo'] (https://zoom.us/join)

ASSIGNMENT 2: on Case 2: xpath by text() function

Syn: // tagName[text()='textvalue']

1. //span[text()='Get access to your Orders, Wishlist and recommendations'] (https://www.flipkart.com)
2. //span[text()='Filters'] (https://www.flipkart.com/inverters/pr?sid=jqe,abm,7nd&otracker=undefined-footer-footer)
3. //h1[text()='Self Drive Car Rental in'] (https://www.zoomcar.com/bangalore/)
4. //p[text()='Earn 7 points for every booking and reedeem for additional discount'] (https://www.zoomcar.com/bangalore)
5. //div[text()='Email or phone'] (gmail signup page)

6. //button[text()='FACEBOOK'] (https://www.zoomcar.com/bangalore)

7. //button[text() = 'Login'] (https://www.facebook.com)

8. //span[text() = 'create your account']
(https://twitter.com/i/flow/signup)

9. //span [text() = 'Date of birth']
(https://twitter.com/i/flow/signup)

10. //label [text() = 'Email or phone']
(https://www.linkedin.com/login)

ASSIGNMENT 2 : on Case 3 : xpath by multiple attributes.

Unique Element (and) :

Syn: tagName[@attr1 = 'attrval' and @attr2 = 'attrval' and ...]

1. //input[@id='email' and @name = 'email']
(https://www.facebook.com)

2. //input [@id='pass' and @placeholder = 'password']
(https://www.facebook.com/password)

3. //button [@name = 'login' and @data_testid = 'royal_login_button']
(https://www.facebook.com/login)

4. //input[@title = 'Search for products, brands and more' or @placeholder = 'Search for products, brands and more'].
(https://www.flipkart.com/searchbar)

Many Elements (or)

5. //a[@class = '_2rpwqI' or @target = '_blank'] (Count : 127)
(https://www.flipkart.com/search?q=hair oil&otracker=Search&otracker1=Search&marketplace=Flipkart&as-show=on&as=off)

6. //img[@class='figure-href' or @height='100']
(Count : 18)
(https://www.patanjaliayurved.net/products/all)

7. //img[@class='img-responsive' or @title='Mine Diamond Ring KDRHCR111007']
(https://www.malabargolddiamonds.com/diamond-jewellery.html).

Combination of a text and an attribute:

8. //h3[@class='FormStandard_subheading' and text()='Additional information related to your business needs']
(https://explore.zoom.us/en/contactsales)

9. //h2[@class='section-title' and text()='Our Collection']
(https://www.malabargoldanddiamonds.com)

10. //p[@class='page-help-link_text and text()='Looking to create a page for a business?']
(https://www.linkedin.com/signup)

ASSIGNMENT 2: On Case 4: xpath by Contains() function

1. //h4[Contains(text(), 'Designed')]
(https://byjus.com/byjus-classes/

2. //h1[Contains(text(), 'Multi')]
(https://www.upgrad.com/blog/introduction-to-multivariate-regression-in-machine-learning/

3. //p[Contains(text(), 'transcend')]
(https://www.gitam.edu)

17 Sunday

4. //p[Contains(text(), 'workplaces')]
(https://www.tcs.com/tcs-chroma)

5. //p[Contains(text(), 'Capgemini operates in nearly')]
(https://www.capgemini.com/in-en/contact-us/)

6. //span[contains(text(), 'Basic')]
(https://www.amazon.com/)
7. //a[contains(@title, 'stupid')]
(https://www.reddit.com/)
8. //a[contains(@href, '/kukatpally/')]
(https://www.omnihospital.com/)
9. //span[contains(text(), '2 & 3 BHK')]
(https://www.myhomes.com/)
10. //img[contains(~~text~~ @src, '/krjs10 1.970x.pg')]
(https://www.malabar.com/)

16.09.21 D11

Case 6: xpath by dependent and independent element. (OR) Reference of surrounding unique element

→ First identify unique element which is nearby non-unique element.

→ From that unique element travel to non-unique element.

Eg: //p[text() = 'Ruby']/following-sibling::p[4]/descendant::a

//li[@class = 'nav-item']/descendant::a[text() = 'Home']

(selenium.dev)

Ruby Download
Java Download
Python Download
C# Download
JS Download

In our example,

1. Inspect independent element
   //td[. = 'Java']

2. Travel back until dependent element within the boundary
   //td[. = 'Java']/..

3. Inspect dependent and write continuously
   //td[. = 'Java']/../a[. = 'Download']

(or)

Alternate way to reach from Java --> Download.

//td[. = 'Java']/following-sibling::td[@data-label = 'Links']/descendant::a[. = 'Download]

//td[. = 'Java']/following-sibling::td[@data-label = 'Links']/a[. = 'Download'].

//td[text() = 'Java']/following-sibling::td[@data-label = 'Links']/descendant::a[text() = 'Download']

//td[text() = 'Java']/ancestor::tr/descendant::a[text() = 'Download']

//td[text() = 'Java']/../a[text() = 'Download']

(flipkart.com)

//div[text() = 'motorola']/preceding-sibling::div[@class = '_1p7h2j']

Case 7: Xpath by group index.

Here we group all similar elements and apply index on that group.

Q: What is the difference between //input, input[i], (//input)[i] ?

//input - Matches with all the inputs on the web-page.

//input[1] - matches with all the first inputs of their parent element (tag)

(//input)[57] - matches 57th input.

(//input)[last()] - matches last input

(//input)[last()-1] - matches second last (input before the last one)

(//input)[position() > last()-5] - matches last 5 inputs.

(//input)[position() mod 2 = 0] - matches all even inputs.

(//input)[position() mod 2 = 1] - matches all the odd inputs.

Explanation: Case 7:

html 1
- body 1
  - div 1
    - a 1
    - a 2
    - a 3
  - div 2
    - a 1
    - a 2
    - a 3

//a - matches all the 'a' -> 6

//a[1] - matches all the first 'a' -> 2

[//a][1] - matches very first 'a' -> 1

a 1
a 2
a 3
a 4
a 5
a 6

ASSIGNMENT 3 : on case 6 : Xpath by dependant and independant element.

1. //div[text()='INDUSTRIES']/ancestor::li[@id='om-leaf-om-u1-6764298 67-1']/following-sibling::li[@id='om-leaf-om-u1-6764298 67-2']/descendant::div[text()='SERVICES & PRODUCTS'] (https://www.hcltech.com)

2. //a[text()='Industries']/ancestor::li[@class='menu-item main-menu-items']/following-sibling::li[@class='menu-item main-menu-items']/descendant::a[text()='Services'] (https://www.tcs.com)

3. //span[text()='Helito']/ancestor::div[@class='a-section a-spacing-none a-spacing-top-small']/following-sibling::div[@class='a-section a-spacing-none a-spacing-top-small']/descendant::span[text()='399'] (https://www.amazon.in/)

4. //div[text()='SAMSUNG GALAXY F22']/ancestor::div[@class='_3ply_crow']/descendant::div[text()='14999'] (https://www.flipkart.com/)

5. //span[text()='Redme Note 10 (Aqua Green, 4GB RAM, 64 GB storage) - Amoled Dot Display 48MP Sony Sensor IM x 582 | snapdragon 678 processor']/ancestor::div[@id='title_features_div']/following-sibling::div[8]/descendant::span[text()='13,999.00'] (https://www.amazon.com/)

6. //span[text() = 'TRADE ZIE (pack of 4) MSWG-4 class set']/ ancestor :: div [@ class = 'aMaAEs'] / descendant :: div [text() = '₹499']
(https : www. amazon. com/)

7. //span[text() = 'VEEMIZO waterproof Shoe bags for Travelling Bag'] / ancestor :: div[@id = 'title_feature_div'] / following-sibling :: div[8] / descendant :: span [text() = '₹280.00']
(https: www. flipkart.com/)

8. // div [text() = 'APPLE iphone 12 (Black 128 GB)'] / ancestor :: div [@ class = 'Col Col-7-12'] / following-sibling :: div[1] / descendant :: div [text() = '₹68,999']
(https : www. flipkart. com/)

9. //span [text() = 'Redmi Earbuds ...'] / ancestor :: div [@id = 'title-feature-div'] / following-sibling :: div [8] / descendant :: span [text() = ₹ 998.00']
(https : www. amazon. com/)

10. // span [text() = ' realme Buds 2 Wired Headset'] / ancestor :: div [@ class = 'amaAEs'] / descendant :: div [text() = '₹599.00']
(https : www. amazon. com/)

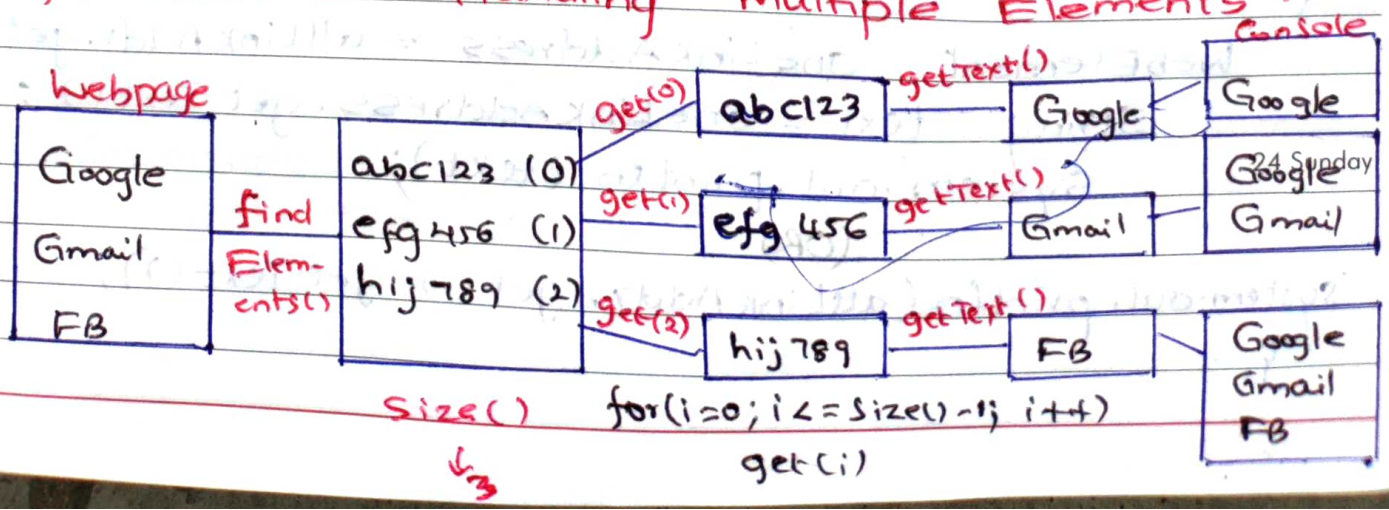
Handling Multiple Elements :

1. We handle multiple elements by using findElements() method of Webdriver Interface
2. Return type of findElements() is List<WebElements>
3. If findElements() method is not able to find the elements on the webpage, then we get Emptylist.

Difference between findElement() & findElements

findElement()	findElements()
1. findElement() is used to get the address of first matching element on the webpage.	1. findElements() is used to get the address of all the matching elements on the webpage.
2. Return is WebElement Interface	2. Return type is List< WebElement >
3. If findElement() method is not able to find the element on the webpage, we get, NoSuchElement Exception.	3. If findElements() method is not able to find the elements on the webpage then we get empty List

Explanation on Handling Multiple Elements :



Program :

```
package arsp;
import java.util.list;
import org.openara.Selenium.By;
import org.openara.Selenium.WebDriver;
import org.openara.Selenium.chrome.chromeDriver;

public class HandlingMultipleElements
{
public static void main(String[] args)
{
System.setproperty("webdriver.chrome.driver",
                   "./drivers/chromedriver.exe");
WebDriver driver = new chromeDriver();
driver.get("file:///D:/KCSMG/HTML/MultipleElements.html");
List<WebElement> allLinkAddr =
    driver.findElements(By.tagName("a"));
// 1. Count the number of links
int noofLinks = allLinksAddr.size();
System.out.println("No.of Links:" + noofLinks);

// 2. print the text of the all links
for (int i=0; i<= allLinkAddr.size()-1; i++)
{
WebElement oneLinkAddress = allLinkAddr.get(i);
String text = oneLinkAddress.getText();
System.out.println(text);
(OR)
System.out.println(allLinkAddr.get(i).getText());
}
```

```
// 3.  Click on last link
    allLinksAddr.get(allLinkAddr.size()-1).click();
  }
}
```

* TEST CASE to login to an Application

Step No.	Action Description	Input	Expected Result	Actual Result
1.	Open the browser Enter the test URL	https://demo.actitime.com/login.do	Login page should be displayed (actiTime - Login)	
2.	Enter valid User name, Password, & click on Login button	UN - admin Pwd - manager	Home page (Enter Time Track page should be displayed)	

* TEST SCRIPT for the above test case

```
package arsp;
import org.openara.Selenium.By;
import org.openara.Selenium.WebDriver;
import org.openara.Chrome.ChromeDriver;

public class ActiTIMELoginTest
{
  public static void main(String[] args) throws
    InterruptedException
      // Set the system property
System.setproperty("webdriver.chrome.driver",
  "./drivers/chromedriver.exe;
```

```
// open the chrome browser
WebDriver driver = new ChromeDriver();
// maximize the browser window
driver.manage().window().maximize();
// Enter the test URL
driver.get("https://demo.actiTime.com/login.do");
// verify Login page is displayed.
String expectedTitle = "actiTime - Login";
String actualTitle = driver.getTitle();
if (actualTitle.equals(expectedTitle))
{
System.out.println("Login page is displayed, PASS");
}
else
{
System.out.println("Login page is not displayed, FAIL");
}
Thread.sleep(2000);
// Enter valid Username into Username Textbox
driver.findElement(By.id("username")).sendKeys
("admin");
Thread.sleep(2000);
// Enter valid password into password Textbox
driver.findElement(By.name("pwd").sendKeys
("manager");
Thread.sleep(2000);
// click on login Button
driver.findElement(By.xpath("//div[text()='Login']").
click();
Thread.sleep(2000);
```

```
// verify login page is Displayed
String expectedHomeTitle = "actiTime - Enter Time-Track";
String actualHomeTitle = driver.getTitle();
if(actualHomeTitle.equals(expectedHomeTitle))
{
System.out.println("Home page is Displayed, PASS");
}
else
{
System.out.println("Home page is not displayed, FAIL");
}
}
}
```

18.09.21 D13

Synchronisation: Matching the speed of selenium according to the speed of application, to avoid synchronisation issue is called Synchronisation.

Q. What is Synchronisation issue?

Selenium ~~the~~ is too fast but application might take time to load, hence this time mismatch is called Synchronisation issue.

There are two types of waits to match Selenium speed.

1. Static wait
2. Dynamic wait

- Static wait means waiting time is fixed. here, we use Thread.sleep(15000)

means, Compulsorily we have to wait for 15 Seconds.

- Dynamic wait means waiting period is not fixed, here it changes according to the application speed.

Two types of Dynamic wait

Implicitlywait : Here, no need to give any Condition. It will wait for findElement() method and findElements() method.

Syn: driver.manage().timeouts().implicitlywait(20, TimeUnit.SECONDS);

20 - indicates long arg

Here 20 means maximum time until Control should wait before giving NoSuchElementException

Note: Implicitlywait statement Can be declared once at the top of the Code which Can work for all the findElement() method / findElements() method

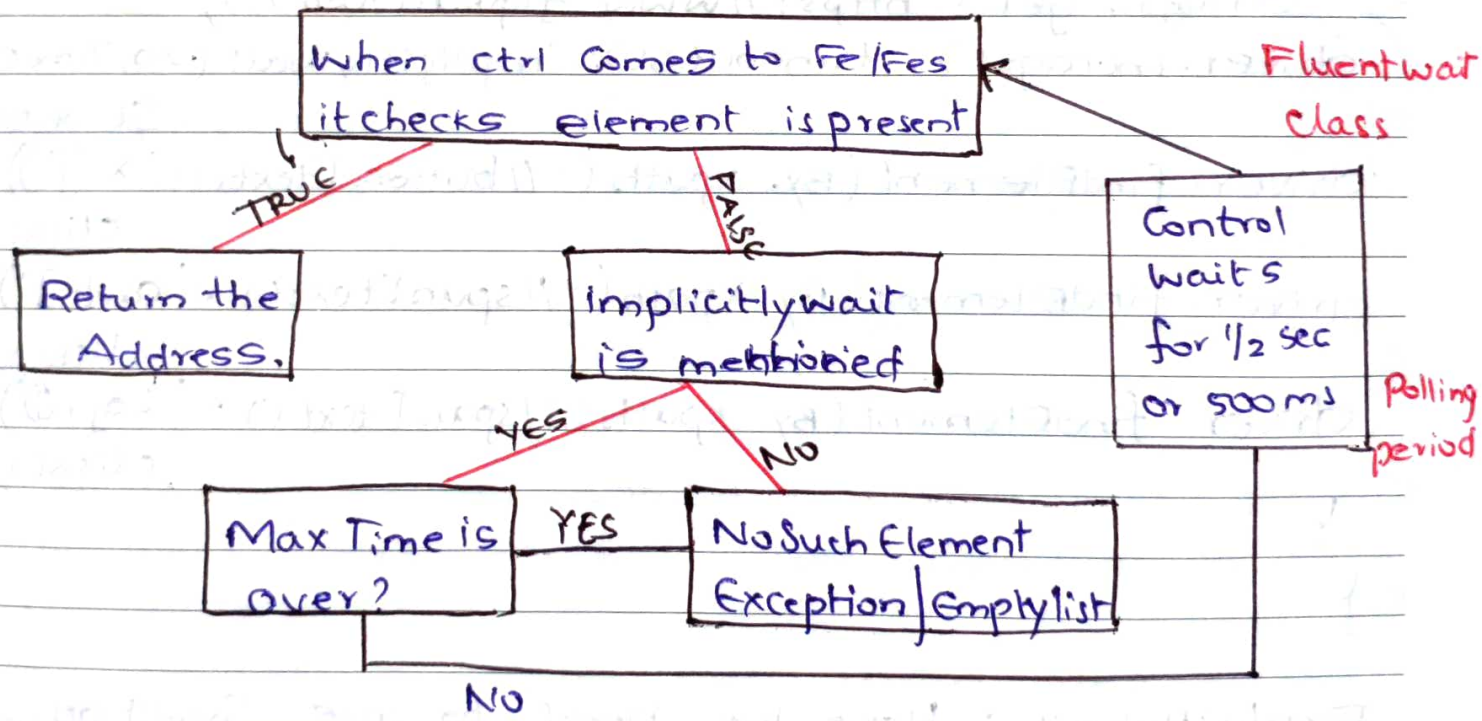
IMPLICITLYWAIT FLOW DIAGRAM ---↑

Explanation: When the Control Comes to findElement() / findElements() it will Check whether the element is present or not.. If the element is present, then findElement() will return the address..

But, if element is not present then it will check whether implicitlywait statement is mentioned or not.. If not then we will get NoSuchElementException / empty list. If mentioned, then it will check whether time is over or not?

If time is over but element not found, NoSuchElementException.. But if time is not over, then control will wait for 1/2 second or 500ms and again check whether element is present or not.

This 500 ms duration is called polling period and is mentioned in Fluentwait class.



Program:

```
package qsp;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.webdriver;
import org.openqa.selenium.chrome.ChromeDriver;
```

03 Sunday

```
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;

public class FlipkartTest
{
public static void main(String[] args) throws InterruptedException
{
System.setproperty("webdriver.chrome.driver", "./drivers/chromedriver.exe");
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://www.flipkart.com/");
driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
driver.findElement(By.xpath("//button[text()='X']")).click();
driver.findElement(By.xpath("//span[text()='Cart']")).click();
driver.findElement(By.xpath("//span[text()='Login']")).click();
}
}
```

Explicitly wait : Here we need to give Condition. It will work for any method.

Create an object of 'WebDriverWait' class.

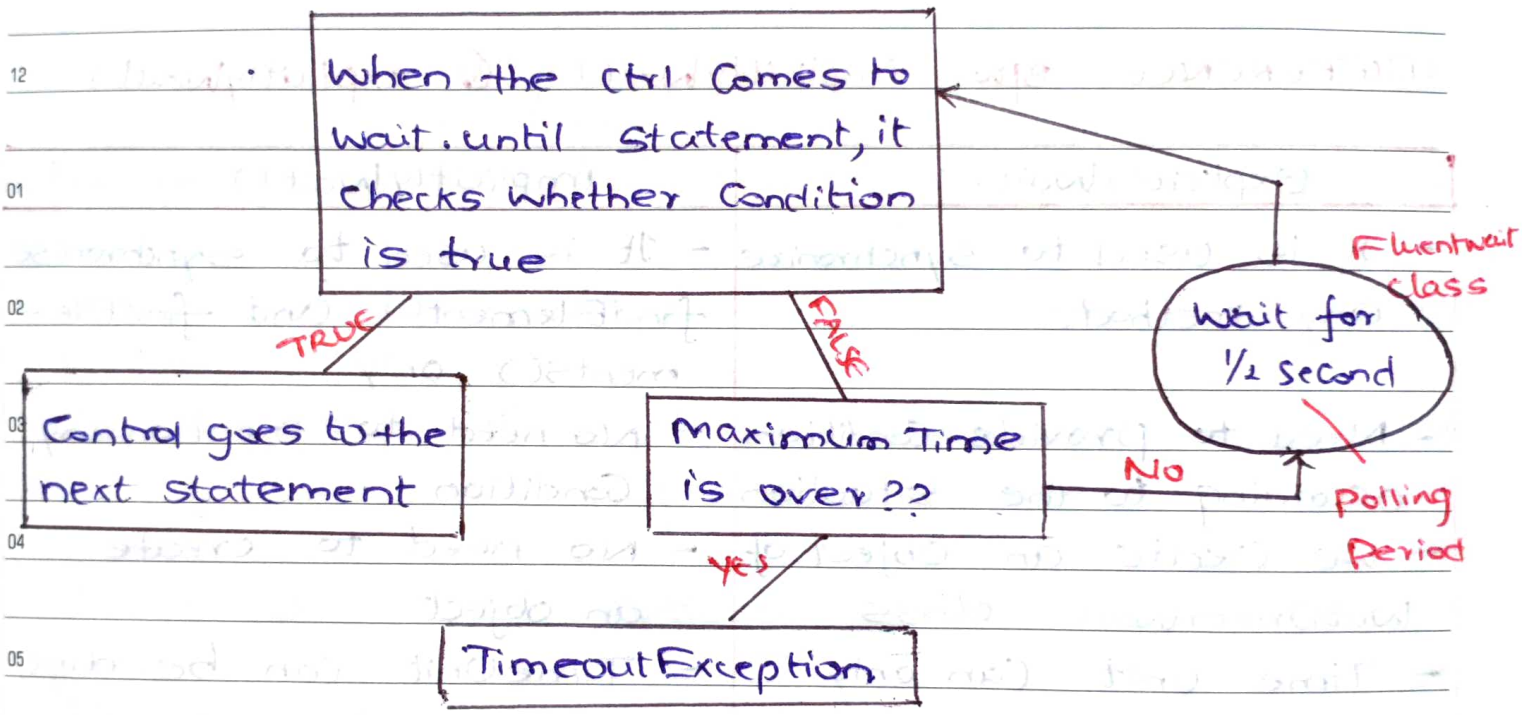
```
WebDriverWait wait = new WebDriverWait(reference Variable of WebDriver, timeout In Seconds);
```

Explicitly wait will work for any method including findElement() and findElements(). Here we need

to create an object of WebDriverWait class and we use the Static methods of ExpectedConditions class to give Condition.

WebDriverWait class and ExpectedConditions class is imported from 'Org.openqa.selenium.support.ui' package.

Explicitlywait Flow Diagram



When the Control Comes to wait.until() statement, it will Check whether the Condition is true or false.

If Condition is true, then Control goes to the next Statements and start executing them...

If Condition is false, then it will check maximum time is over or not.

If yes, that means Conditions never became true within the maximum time, it gives TimeOutException.

If no, it waits for ½ sec or 500 ms and then checks again Condition became true or not.

For 04 Mar program, we can also use Explicitly wait(),

Last statement but one,

```
WebDriverWait wait = new WebDriverWait(driver, 20);
Wait.until(ExpectedConditions.visibilityOfElementLocated(By.
xpath(" By.xpath("//span[text() =
```

DIFFERENCE B/W implicitlyWait() & explicitlyWait()

ExplicitlyWait()	implicitlyWait()
- It is used to synchronize any method.	- It is used to synchronise findElement() and findElements() only.
- Need to provide condition according to the situation	- No need to mention any condition.
- We create an object of WebDriverWait class	- No need to create an object
- Time unit can only be SECONDS	- TimeUnit can be days, hours, minutes, seconds, milliseconds, microseconds, nano seconds.
- Here, if condition doesn't become true within maximum time, then we get TimeOutException.	- Here, if element is not found, within maximum time, we get NoSuchElementException / Empty list
- It is a kind of "localwait" which has to be mentioned before every synchronization issue.	- It is a kind of "global wait" because we can mention implicitlywait once at the top of the code which works for every findElement()

explicitlyWait()	implicitlyWait()
– Syn: WebDriverWait = new WebDriverWait(WebDriver reference, TimeoutInSeconds) wait.until(ExpectedCondition Required Condition)	or findElements() through the Code: – Syn: driver.manage().timeouts() implicitlyWait(long arg, Timeout). Eg: driver.manage().timeout(). implicitlyWait(20, TimeUnit.SECONDS)

19.09.21 D14

Handling Dropdown (List Box / Web List / Custom Dropdown)

Dropdowns are used to save the space on the webpage.

Two types : 1. Static Dropdown
2. Dynamic Dropdown

Static Dropdown: options of the dropdown are fixed.

Two types of Static Dropdown.

1. Single-select Dropdown
2. Multi-select Dropdown

created by using <select>, <options> tags.

Q: How to handle the static dropdown?

Since it is created by using <select>, we handle it by using select class.

Select class is imported from org.openqa.Selenium.support.ui package.

select class Constructor is a parameterized Constructor. which takes webElement arg, where we need to pass address of the dropdown.

Select Class Methods:

Selection methods

1. SelectByVisibleText (String arg) : void
2. selectByValue (string arg) : void
3. SelectByIndex (int index) : void

Deselection Methods

1. deselectByVisibleText (string arg) : void
2. deselectByValue (String arg) : void
3. deselectByIndex (int index) : void
4. deselectAll () : void

Operational Methods

1. isMultiple() : boolean
2. getOptions() : List<WebElement>
3. getAllSelectedOptions() : List<WebElement>
4. getFirstSelectedOption () : WebElement
5. getWrappedElement() : WebElement

Note :

1. selectByIndex() can handle duplicate options
2. when we try to use deselection methods on a single-select dropdown, then we get UnsupportedOperationException. (Java) (RTE)

isMultiple(): Used to verify whether it is a Single Select-dropdown (or) multi-select drop-down. Return type is boolean. Returns true if it is multi-select dropdown.

getOptions(): Used to get the address of all the options present in the dropdown. Returntype is List<WebElement>.

If no option is present in dropdown, then we get EmptyList.

getAllSelectedOptions(): Used to get the address of the selected options. Return type is List<WebElement>. If no option is selected in dropdown then we get Empty List.

getFirstSelectedOption(): Used to get the address of first selected option in the dropdown. Return type is WebElement Interface. If no option is selected, then it returns NoSuchElementException

getWrappedElement(): Used to get the address of all the options wrapped into a single address. Return type is WebElement Interface. If no option is present in dropdown, then we get NoSuchElementException.

Note: To create a multi-select dropdown, add the attribute multiple in select tag.

Program:

```
package arsp;
import java.util.List;
import org.openqa.selenium.By;
import org.openqa.Selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.Selenium.chrome.ChromeDriver;
import org.openqa.Selenium.support.ui.select;

public class HandlingDropdown
{

public static void main(String[] args) throws
                InterruptedException.
```

```
System. Setproperty ("webdriver. chrome. driver",
                    "./drivers/chromedriver. exe");
WebDriver driver = new ChromeDriver();
driver. get ("File:///D:/KCSMG/HTML/dropdown. html");
driver. get ("file:///D:/KCSMG/HTML/MultiselectDropdown.
                                                html");

    Thread. Sleep (2000);
WebElement ddAddr = driver. findElement (By. id("SGM"))
    Select sel = new Select (ddAddr);
    // Sel. SelectByVisibleText (" CHAPATHI");
    // Sel. select By Value (" idli");
    // Sel. SelectByIndex (8);
    // ddAddr. Click();

    if (sel. isMultiple())
    {
        System.out. println (" It's a Multi-Select Dropdown");
} else
    { system. out. println (" It's a single-Select dropdown");
}
// Sel. SelectBy VisibleText("DOSA");
// List <WebElement> allOptions = sel. getOptions();
// for(int i= 0; i<= allOptions. size()-1; i++)
// {
//     System. out. println (allOptions. get(i). getText());
// }
// for (int i=0; i<= allOptions. size()-1; i++)
// {
//     sel. SelectBy Index(i);
//     Thread. Sleep (1000);
// }
```

```
// Thread.sleep(3000);
//    sel.deselectByVisibleText("DOSA");
//    sel.deselectByValue("dosa");
//    sel.deselectByIndex(1);
//    sel.deselectAll();
//    for(int i=1; i<=4; i++)
// {
//    sel.SelectByIndex(i);
// }
// System.out.println(sel.getFirstSelectedOption().getText());
// System.out.println(sel.getAllSelectedOptions().size());
System.out.println(sel.getWrappedElement().getText());
}
}
```

HTML Code for the above program.

```
<html>
<body>
<select id='novotel'> (OR) <select id='SGH' multiple>
<option value='idli'> IDLI </option>
<option value='dosa'> DOSA </option>
<option value='vada'> VADA </option>
<option value='poori'> POORI </option>
<option value='chapathi'> CHAPATHI </option>
<option value='samosa'> SAMOSA </option>
<option value='panipoori'> PANI POORI </option>
<option value='thatti idli'> THATTI IDLI </option>
<option value='idli'> IDLI </option>
<option value='double ka meeta'> DOUBLE KA MEETA
</option>
<option value='irani chai'> IRANI CHAI </option>
</select>
</body>
</html>
```

20.09.21

D15

Handling Duplicates

Inputs :

Idli
Dosa
Vada
poori
Idli
Chai

List (I)

Idli
Dosa
Vada
poori
Idli
chai

1. Duplicates are allowed
2. Insertion or is maintained
3. Prints as per the given order (Explan. to 2nd point)

SET (I)

HashSet (c)

Dosa
poori
Idli
Chai
Vada

1. Remove Duplicates
2. Insertion order is not maintained

~~Idli~~ ~~Dosa~~ ~~vada~~ ~~poori~~ ~~Idli~~

Dosa poori Idli ~~Chai~~ ~~Vada~~

LinkedHashSet(c)

Idli
Dosa
Vada
poori
Chai

1. Remove Duplicates
2. Order is maintained

Idli Dosa vada poori Chai

TreeSet(c)

Chai
Dosa
Idli
poori
Vada

1. Remove Duplicates
2. Sorted order

Chai Dosa Idli poori vada

Program :

```
package arsp1;
import java.util.Hashset;
import java.util.LinkedHashset;
import java.util.List;
import java.util.TreeSet;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.Select;

public class HandlingDuplicates
{
public Static void main(String[] args)
{
System.SetProperty("webdriver.chrome.driver",
        "./drivers/chromedriver.exe");
WebDriver driver = new ChromeDriver();
driver.get("file:///D:KCSM6/HTML/MultiselectDropdown.html");
WebElement ddAddr = driver.findElement(By.id("SGH"));
Select sel = new Select(ddAddr);
List<WebElement> allOptions = sel.getOptions();
for (int i=0; i<=allOptions.size()-1; i++)
{
    String text = allOptions.get(i).getText();
    System.out.println(text);
}
List<WebElement> allOptions = sel.getOptions();
HashSet<String> allText = new HashSet<String>();
for (int i=0; i<=allOptions.size()-1; i++)
{
```

```
//      String text = allOptions.get(i).getText();
08          allText.add(text);
// }
09// List<WebElement> allOptions = Sel.getOptions();
// HashSet<String> allText = new HashSet<String>();
10// for(int i=0; i<=allOptions.size()-1; i++)
// {
11// String text = allOptions.get(i).getText();
// allText.add(text);
12// }
// for(String oneText : allText)
01// {
//      System.out.println(oneText);
// }
// List<WebElement> allOptions = Sel.getOptions();
03  LinkedHashSet<String> allText = new LinkedHashSet<x>;
    for (int i=0; i<=allOptions.size()-1; i++)
04  {
        String text = allOptions.get(i).getText();
05          allText.add(text);
    }
06      for(String oneText : allText)
    {
07          System.out.println(oneText);
    }
    List<WebElement> allOptions = Sel.getOptions();
    TreeSet<String> allText = new TreeSet<>();
    for (int i=0; i<=allOptions.size()-1; i++)
    {
        String text = allOptions.get(i).getText();
            allText.add(text);
    }
```

```
for (string oneText: allText)
{
    System.out.println(oneText);
}
}
```

21.09.21 D6

Dynamic Dropdown / Custom Dropdown

In this kind of dropdown, the options are changing..

To create it we use tags like input, div, ul, li, a etc

Q: How to handle dynamic dropdown?

Since the options are inspectable, we use findElement(), findElements(), sendkeys(), click().

Since it is not created by select tag, hence we can't use Select class and its methods.

If we use, we get UnexpectedTagNameException.

TS:

```
package avspt;
import java.util.List;
public class HandlingDynamicDropdown
{
public static method main(String[] args) throws InterruptedException
{
System.setProperty("webdriver.chrome.driver",
"./drivers/chromedriver.exe");
```

```
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://www.flipkart.com/");
driver.findElement(By.xpath("//button[text()='X']
    ")).click();
WebElement ddAddr = driver.findElement(By.name("a
    ddAddr.Sendkeys("mobiles");
    ddAddr.sendkeys(Keys.Down);
    ddAddr.Sendkeys(Keys.ENTER);
driver.get("https://www.google.com/");
driver.findElement(By.name("q").sendkeys("carspiders")
    Thread.sleep(3000);
List<WebElement> allOptions = driver.findElements
    (By.xpath("//div[@class='wMGWTd']"));
    Thread.sleep(3000);
    System.out.println(allOptions.size());
  }
}
```

22.09.2021 D17

Handling Keyboard and Mouse Actions

Action class: Used to handle keyboard and mouse actions imported from org.openqa.selenium.interactions package.

→ Actions class Constructor takes WebDriver Reference or Browser class reference as its argument.

→ We handle mouse and Keyboard Actions by using non-static methods of Action class.

→ Also note for every method of Action class, we need to Compulsorily use .perform() at

the end.

→ Robot class is used to handle keyboard and mouse Actions. It is imported from java.awt.package

→ Robot class Constructor throws a Compiletime exception called AWTException

→ Robot class uses keyPress in which KeyEvent class is present.

→ In KeyEvent class, all the Keyboard Keys are stored as Virtual keys and since they are Static final variables we can access directly through classname.variableName (KeyEvent.VK_T)

→ Also remember, to release a key after it is pressed, so that it will be released virtually.

1. Handling mouse hover action

```
moveToElement(addr).perform()
```

2a. Perform Right Click

```
ContextClick(addr).perform()
```

2b. Handling Right Clicked Options

Robot class - keyPress / keyRelease

3 Perform Double click

```
doubleClick(addr).perform()
```

4. Perform Drag & Drop action

```
dragAndDrop(fromAddr, toAddr).perform()
```

S.No	Actions	methods	Mandatory method	Class
1.	Mouse Hover Action	moveToElement (address of an element)	perform()	Actions class
2.	Right Click Action	ContextClick (address of an element)	perform()	Actions class
3	Double click Action	doubleClick (address of an element)	perform()	Actions class
4	Drag and Drop Action	dragAndDrop (from address, to address)	perform()	Actions class
5	Handle Right clicked options	keyPress (KeyEvent.VK)	keyRelease (keyEvent.VK)	Robot class

Example :

```
package . asp1 ;
import java. awt . AWTException;
import java . awt . Robot;
import java. awt. event. KeyEvent;
import org. openqa. selenium . By ;
import org. openqa. Selenium. WebDriver ;
import org. openqa. Selenium. WebElement ;
import org. openqa. Selenium. chrome . ChromeDriver;
import org. openqa . Selenium . interactions. Actions;

public class HandlingMouseAndKeyboardActions
{
public static void main( String[] args) throws
    InterruptedException, AWTException
{
    // performing Mouse Hover Action
    System. setProperty( "webdriver. chrome. driver",
        ".\drivers\ Chromedriver. exe");
```

```
webDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://www.flipkart.com/");
driver.findElements(By.xpath("//button[text()='X']")).
click();

Actions ac = new Actions(driver);
WebElement fashion = driver.findElement(By.xpath("//
div[text()='Fashion']"));
ac.moveToElement(fashion).perform();
Thread.sleep(3000);
// performing Double click Action
driver.get("https://demo.actitime.com/login.do");
WebElement untb = driver.findElement(By.id("username"));
Untb.sendKeys("admin");
ac.doubleClick(untb).perform();
Thread.sleep(3000);
// Right Click and Handle Right clicked Options.
driver.get("file:///D:/kcsm6/HTML/Link.html");
Thread.sleep(3000);

WebElement linkAddress = driver.findElement(By.id("ii"));
ac.contextClick(linkAddress).perform();
Thread.sleep(3000);

Robot r = new Robot();
r.keyPress(KeyEvent.VK_T);
r.keyRelease(KeyEvent.VK_T);
Thread.sleep(3000);
// Perform Drag and Drop Actions
driver.get("https://www.dhtmlgoodies.com/scripts/
drag-drop-custom/demo-drag-drop-3.html");
```

```
WebElement src = driver.findElement(By.id("box7"));
WebElement dest = driver.findElement(By.id("box107"));
ac.dragAndDrop(src, dest).perform();
}
}
```

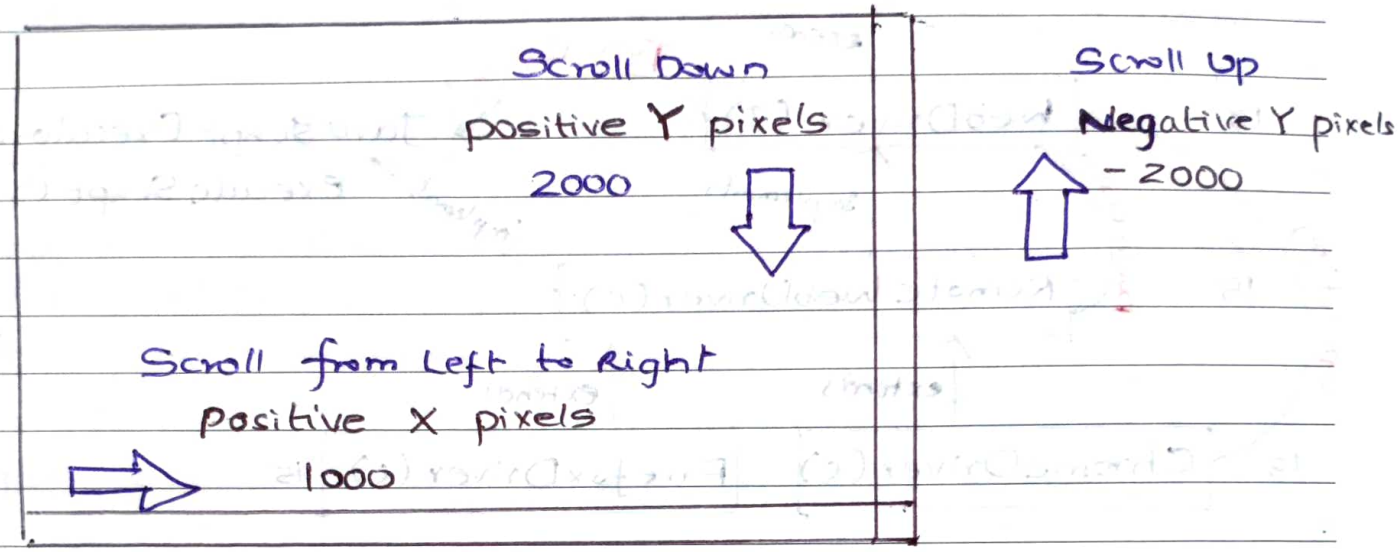
23.09.2021 D18

One more way to do the Assignment 6:

```
System.setProperty("webdriver.chrome.driver", "./drivers/chromedriver.exe");
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://demo.actitime.com/login.do");

WebElement untb = driver.findElement(By.id("username"));
untb.sendKeys("admin");
untb.sendKeys(Keys.CONTROL, "a");
untb.sendKeys(Keys.CONTROL, "c");
(or)
Thread.sleep(3000);
untb.sendKeys(Keys.CONTROL, "ac");
Thread.sleep(3000);

driver.findElement(By.name("pwd")).sendKeys(Keys.CONTROL, "v");
```

Q: How do we scroll down?

We use javascript



Scroll from Right to Left
Negative X pixels
-1000

Q: How to scroll down manually?

Step 1: Go to the required webpage and Press "Ctrl + Shift + I" or "F12 / Fn F12" "Developer Tools" will be displayed.

Step 2: Click on the "Console tab" and write the javascript as follows.

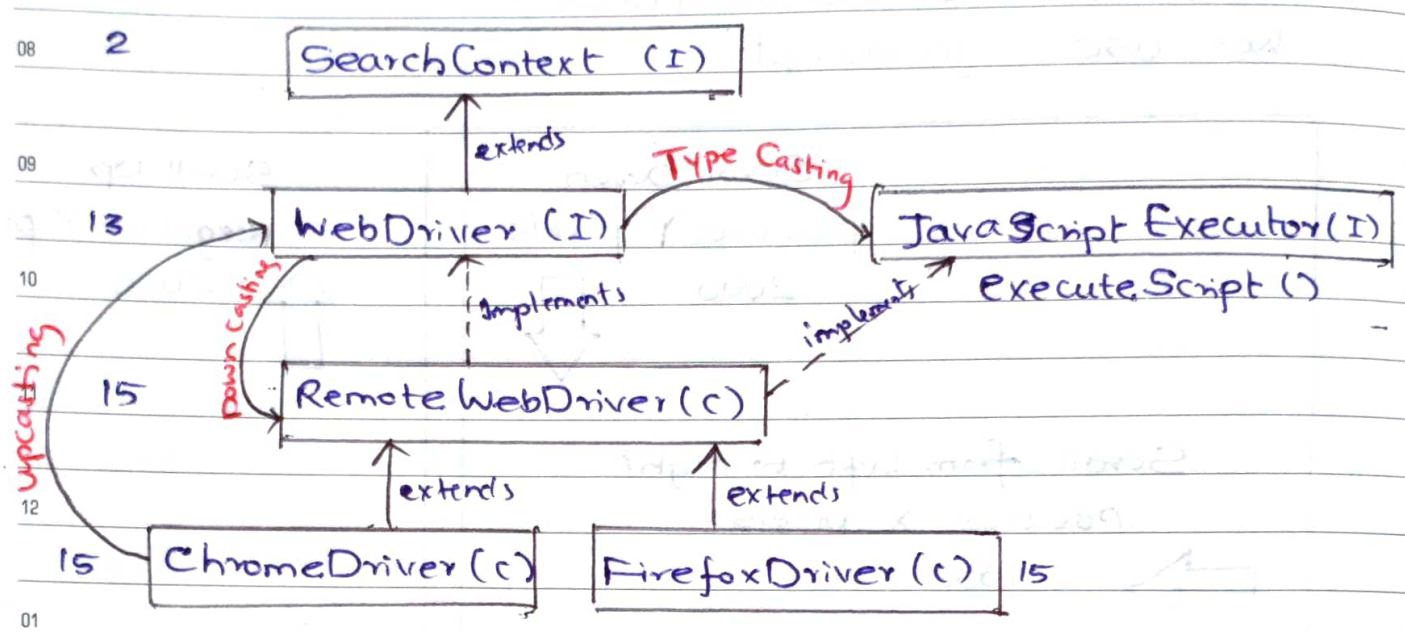
Syn: window.scrollBy(x axis, y axis)

Example: window.scrollBy(100, 800)

Q: How to scroll down through Automation(selenium WebDriver)???

24 Sunday

In Selenium we use executeScript() of JavaScriptExecutor interface to execute the required javascript automatically...



CD dr = new CD();
dr.eS();

RWD dr = new CD();
dr.eS() ✓

JSE dr = new CD();
dr.eS();

WD dr = new CD();
dr.eS(); ✗

Since we are upcasting our driver to webdriver interface, executeScript() method is not accessible.
Hence to access it we either,
TypeCast from WebDriver interface to JavaScript-Executor interface.
(OR) Downcast from WebDriver interface to RemoteWebDriver Class.

Q: How to scroll down to a particular element?
If we want to scroll down to a particular element, then we get the location of the element (location means X axis and Y axis)

by using getLocation() method of WebElement interface.

→ getLocation() returns Point (x axis, y axis) of the element.

Window.scrollBy + location of Element

Eg:

```
WebElement element = driver.findElement(By.xpath(
"//h2[.='Selenium Level sponsors']"));
Point loc = element.getLocation();
JavaScriptExecutor jse = (JavaScriptExecutor)driver;
jse.executeScript("window.scrollBy" + loc);
```

Example :

```
package asp1;
import org.openqa.selenium.By;
import org.openqa.selenium.JavaScriptExecutor;
import org.openqa.Selenium.Point;
import org.openqa.Selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.Selenium.chrome.ChromeDriver;
import org.openqa.Selenium.remote.RemoteWebDriver;

public class ScrollingDownAWebPage{
  public static void main(String[] args) throws InterruptedExcep...
{ System.setProperty("webdriver.chrome.driver", "./drivers/chromedriver.exe");
  // Upcasted from ChromeDriver class to webDriver Interface
  WebDriver driver = new ChromeDriver();
  driver.manage().window().maximize();
  driver.get("https://www.selenium.dev/downloads/");
  Thread.sleep(2000);
```

```
// Through Typecasting
JavascriptExecutor js = (JavascriptExecutor) driver;
js.executeScript("window.scrollBy(0,1000)");

// Through DownCasting
RemoteWebDriver rwd = (RemoteWebDriver) driver;
rwd.executeScript("window.scrollBy(0,1000)");

WebElement selspons = driver.findElement(By.xpath(
"h2[text()='Selenium Level Sponsers']"));
System.out.println(loc);
Point loc = Selspons.getLocation();
JavaScriptExecutor j = (JavascriptExecutor) driver;
j.executeScript("window.scrollBy"+loc);
}
}
```

HANDLING DISABLED ELEMENT

→ Disabled elements are created by 'Disabled' attribute.

→ It is handled by Javascript.
If we try to handle disabled element without javascript, i.e., by findElement(), click(), sendKeys() then we get **ElementNotInteractable** Exception.

⇒ Handling Disabled element manually:

1. Go to the required webpage, where disabled element is present.
2. Press F12/ Function + F12 (Ctrl + Shift + I) - Developer tools will be opened.

3. Click on console tab, write the required javascript and press enter (execute javascript)
document.getElementById('i2').value='manager'

⇒ Handling Disabled element through Automation By using executeScript() of JavascriptExecutor interface.
Since, we upcast our browser class to webDriver Interface, this method is hidden.
Hence we either Typecast from
WebDriver Interface to JavascriptExecutor Interface
or
Downcast from
WebDriver Interface to RemoteWebDriver class

HTML Document

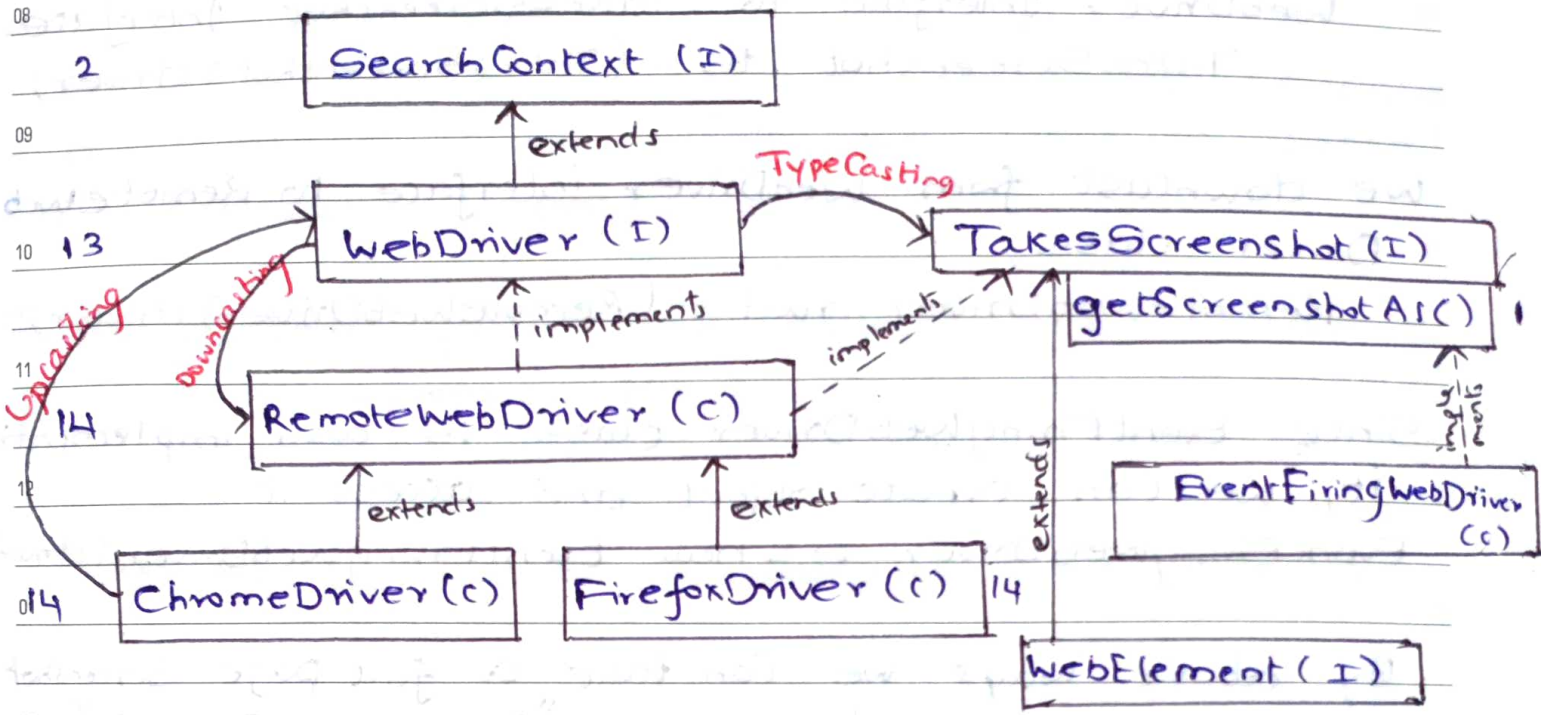
```
<html>
<body>
Username<input type='text' id='i1'> </input> <br>
Password < input type = 'text' id='i2' > disabled >
  </input>
</ body >
</ html >
```

EXAMPLE

```
package arsp1;
import org.openqa.selenium.By;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
```

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

public class HandlingDisabledElement
{

public Static void main(String[] args) throws InterruptedException
{
System.setProperty("webdriver.chrome.driver",
"./drivers/chromedriver.exe");
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("file:///D:KCSM6/HTML/DisabledElement.
html");

Thread.sleep(2000);
driver.findElement(By.id("i1")).sendKeys("admin");
Thread.sleep(2000);
WebElement pwtb = driver.findElement(By.id("i2"));
if(pwtb.isEnabled())
{
System.out.println("Password textbox is enabled,
Handle Normally");
pwtb.sendKeys("manager");
}
else
{
System.out.println("Password textbox is Disabled,
Handle through Javascript");
JavascriptExecutor jse = (JavascriptExecutor) driver;
jse.executeScript("document.getElementById('i2').
value="manager"");
}
}
```

TAKING SCREENSHOTS THROUGH AUTOMATION



CD dr = new CD();
dr.gSA(); ✓

RWD dr = new CD();
dr.gSA(); ✓

~~TS dr = new~~

TS dr = new CD();
dr.gSA(); ✓

WD dr = new CD();
dr.gSA(); ✗

We use getScreenshotAs() method of TakesScreenshot Interface to take a screenshot.
getScreenshotAs() is given implementation in RemoteWebDriver class and it is extended in browser classes.

But as a Selenium standard we always upcast our browser object to WebDriver Interface,
WebDriver driver = new ChromeDriver();
As a result getScreenshotAs() is hidden and inaccessible.

Hence to access it, we either typecast from WebDriver Interface to TakesScreenshot Interface.

```
TakesScreenshot ts = (TakesScreenshot) driver;
```

We downcast from WebDriver interface to RemoteWebDriver class,

```
RemoteWebDriver rwd = (RemoteWebDriver) driver;
```

Since EventFiringWebDriver class is also implementing it, we can create object and access it

```
EventFiringWebDriver e = new EventFiringWebDriver (driver)
```

By above ways we can take a full page Screenshot But if we have to take a "Particular Element" screenshot, Then, since WebElement interface extends TakesScreenshot interface, we can access through WebElement by getting the address of element and then Taking Screenshot of it.

Eg:

```
WebElement imgAddress = driver.findElement(By.id("hplogo"))
File src = imgAddress.getScreenshotAs(outputType.FILE)
```

Explaining TakesScreenshot Program in Interview

1. Upcasting Browser class to WebDriver Interface
2. Typecasting from WebDriver to TakesScreenshot interface.
3. Getting the Screenshot by Calling getScreenshotAs() which takes argument, OutputType which is file type.

4. We create an object of File class and in Constructor we pass three things.
   a. Path where we need to store the Screenshot
   b. FileName with which we want to store
   c. Extension (.png / .jpg)

5. Using third party class called "Files" which is imported from Com.google.Common.io package we copy the obtained screenshot and paste it in the specified path.
   Files.copy() throws a Compile time exception Called IOException.

Example :

```
package arsp1;
import java.io.file;
import java.io.IOException;
import org.openara.selenium.By;
import org.openara.selenium.OutputType;
import org.openara.Selenium.TakesScreenshot;
import org.openara.Selenium.WebDriver;
import org.openara.selenium.WebElement;
Import org.openara.selenium.chrome.ChromeDriver;
import org.openara.Selenium.remote.RemoteWebDriver;
import org.openara.Selenium.remote.support.events.
                                EventFiringWebDriver;
import Com.google.common.io.Files;
public class TakingScreenshot
{
Public static void main(String[] args) throws
  IOException
```

```
System.setProperty("webdriver.chrome.driver",
    "./drivers/chromedriver.exe");
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://www.google.com/");

// Through TypeCasting
TakesScreenshot ts = (TakesScreenshot) driver;
// Take Screenshot and store in heap area
File src = ts.getScreenshotAs(OutputType.FILE);
// But we want in our location
// path + Filename + extension
File dest = new File("D:\\KCSMG\\Screenshots\\
                GoogleImage.jpg");
// Copy from Src and paste it in dest
Files.Copy(src, dest);

// Through Downcasting
RemoteWebDriver rwd = (RemoteWebDriver) driver;
File Sre = rwd.getScreenshotAs(OutputType.FILE);
File dest = new File("D:\\KCSMG\\Screenshots\\
                GooglePhoto.png");
Files.Copy(src, dest);

// Taking screenshot of a particular element
WebElement googleLogo = driver.findElement(By.
    xpath("//img[@alt='Google']"));
File src = googleLogo.getScreenshotAs(OutputType.
                FILE);
File dest = new File("D:\\KCSMG\\Screenshots\\
                GoogleLogo.png");
Files.copy(src,dest);
```

```
// Through EventFiringWebDriver class
EventFiringWebDriver ef = new EventFiringWebDriver
                                        (driver);
File src = ef.getScreenshotAs(OutputType.FILE);
File dest = new File("D:\\\\KCSMG\\\\Screenshots\\\\
        GooglePhoto.png");
Files.copy(src, dest);
}
}
```

25.09.2021 D20

Handling Web Table :

Web table is created by using tags like
<table>
<th> / <thead> → table head
<tbody> → table body
<tr> → table row
<td> → table data

We can handle web table by using findElement() and findElements() of WebElement Interface

findElement() of	
WebDriver (I)	WebElement (I)
It will search full page and get address of first matching element on the webpage.	It will search within the current element and get the address of first matching element within the specified element.

findElements() of

WebDriver (I)	WebElement (I)
It will get the address of all the matching elements after searching on full webpage.	It will get the address of all the matching elements within the current element.

*Xpath to handle dynamic webtable
"//table[attrName = 'attrval']//tr[*]//td[Col num]"

EXAMPLE :

```
package arsp1;
import java.util.List;
import org.openqa.Selenium.By;
import org.openqa.Selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.Selenium.Chrome.ChromeDriver;

public class HandlingWebTables
{
public static void dynamicTable(WebDriver driver, int ColNum)
{
List<WebElement> Col = driver.findElement(By.Xpath("//table[@id = 't2']//tr[*]//td[" +ColNum+"]"));
System.out.println(Col.size());
}
```

```
public static void main(String[] args)
{
System.setProperty("webdriver.chrome.driver",
    "./drivers/chromedriver.exe");
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("file:///D:/KCSMG/HTML/webtable.html");
List<WebElement> allTrs = driver.findElements(By.
                                tagName("tr"));
System.out.println("Total rows on webpage:"
                                + allTrs.size());

WebElement table2 = driver.findElement(By.id("t2"));
List<WebElement> table2Rows = table2.findElements
                        (By.tagName("tr"));
System.out.println("table 2 rows:" + table2Rows.size());

WebElement tr3Row = driver.findElement(By.xpath("
    //table[@id='t2']//tr[3]"));
List<WebElement> tr3data = tr3Row.findElements
        (By.tagName("td"));
System.out.println("3rd row data:" + tr3data.size());

// Handling Dynamic Webtable
driver.get("file:///C:/Users/HP/Desktop/HTML/DynamicWeb
                table.html");
dynamicTable(driver, 7);
}
}
```

07 Sunday

webDriver (I)	Link Link input link link	Full page webElement (I)
Full webpage level,		Element Level
So, findElements - 5 (link)	link div	table So, findElements - 2 (link)

HANDLING POP-UPS

Pop-ups are anything which gives information to user or takes information / Confirmation from the user.

They are of 2 types of pop-ups
1. Web-based Popup
2. Window-based Popup

Web-based Popup:

Here, we have,
1. Javascript popup
2. Hidden Division Popup
3. Browser Notification popup

→ Javascript Popup:

Created by using Javascripts.. hence the name

Types:
1. Alert (only OK button)
2. Confirmation (Both OK and Cancel button)

Characteristics:
1. Is it Colorful ? – No
2. Is it Movable ? – No
3. Is it inspectable ? – No
4. Is it Until we handle this popup we can't Continue with webpage.

Q: How to Handle it ?

First we need to switch our Control from webpage to popup

```
Alert al = driver.switchTo().alert();
```

After switching we handle by using the abstract methods of Alert Interface,

al.accept() - Clicks on OK button

al.dismiss() - Clicks on Cancel button

al.getText() - Used to get the message on popup

al.sendKeys("string"); - Used to type any message too popup.

Note:

If there are no Javascript popups to handle, but still we try to switch and handle then we get NoAlertPresentException

→ Hidden Division Popup:

Characteristics:

1. Is it Colorful? - YES
2. Is it Movable? - ~~YES~~ No
3. Is it Inspectable? - YES

How to handle it?

Since we can inspect it, we can handle by using findElement(), click() and sendKeys().

```
<html>
<body>
<h2> JavaScript Alert </h2>
<button onclick="myFunction()"> Try It </button>
<Script>
function myFunction()
{
alert(" I am an Alert box! ");
}
```

```
</script>
</body>
</html>
```

```
<html>
<body>
<h2> JavaScript Confirm Box </h2>
<button onclick="myFunction()"> I Love you </button>
<p id="demo"></p>

<Script>
function myFunction()
{
var text;
If(Confirm("will you marry me???"))
{ txt = "You pressed OK! Aww... Cho chweet... Love you too;)";
}
else {
txt = " You pressed Cancel! Get lost Dabba Fellow;(";
}
document.getElementById("demo").innerHTML = text;
}
</ script>

</body>
</html>
```

EXAMPLE:

```
package arsp1;
import org.openqa.selenium.Alert;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class HandlingPopups
{
public static void main(String[] args) throws
                                InterruptedException
{
System.setProperty("webdriver.chrome.driver", "./dri-
  vers/chromedriver.exe");
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("file:///D:/KCSMG/HTML/Alert.html");
Thread.sleep(2000);
driver.findElement(By.xpath("//button[text()='Try It']")
  .click();

Alert al = driver.switchTo().alert();
Thread.sleep(2000);
System.out.println(al.getText());
Thread.sleep(2000);
al.accept();

driver.get("file:///D:/KCSMG/HTML/Confirm.html");
Thread.sleep(
driver.findElement(By.xpath("//button[text()='I love
you']")).click();
Thread.sleep(1000);
```

```
Alert al = driver.switchTo().alert();
Thread.sleep(1000);
System.out.println(al.getText());
Thread.sleep(1000);

// To click on OK button
    al.accept();

// To click on Cancel button
    al.dismiss();

// Handling Hidden Division popup

System.setProperty("webdriver.chrome.driver",
"./drivers/chromedriver.exe");
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://www.cleartrip.com/");
Thread.Sleep(3000);
driver.findElement(By.xpath("//button[@class=
'flex flex-middle flex-between t-all fs-2 focus:bc-
secondary-500 bg-transparent bc-neutral-100 c-poi-
nter pr-2 pl-3 pt-2 pb-2 ba br-4 h-8 c-neutral-
900"]")).click();
Thread.sleep(3000);
driver.findElement(By.xpath("//div[contains(@aria-
label, 'Oct')]/descendant::div[text()='28']")).click()
```

→ Browser Notification Popup:

characteristics

1. It is not Colorful
2. It is not movable
3. It is not Inspectable
4. But, we can continue with webpage without even handling it.

This is the only difference between Javascript Popup and Browser Notification popup.

Window Based Popup:

Selenium Can't automate window based applications. Hence we take help from third party tools like Robot Class or AutoIT.

Characteristics:

1. It is Colourful.
2. It is movable
3. It is not Inspectable.
4. We Can't go further before handling it.

Examples:

File upload
File Download
Print

Q: How to handle File upload?

Trick: Use sendkeys() (complete path of the file which has to be uploaded).

14 Sunday

If you try to handle file upload using Selenium, you get an exception - InvalidArgumentException

So, To copy the complete path of window popup,
" Shift + RightClick --> Copy as path.

Proper way: AutoIT

Q: How to handle File Download?
Trick: Use Robot class
Proper way: AutoIT
Note: No need to handle download popup in chrome browser. Because, it will not show popup but it auto-downloads any file.
But, to handle it in firefox browser, use Robot class.

EXAMPLE:

```
import java.awt.event.KeyEvent;
import org.openqa.selenium.By;
import org.openqa.Selenium.WebDriver;
import org.openqa.Selenium.WebElement;
import org.openqa.Selenium.chrome.ChromeDriver;
import org.openqa.Selenium.firefox.FirefoxDriver;
import java.awt.Robot; import java.awt.AWTException;
public class HandlingWindowPopups
{
  public static void main(String[] args) throws
    InterruptedException, AWTException
{
  // Handling File Upload popup
System.setProperty("webdriver.chrome.driver",
"./drivers/chromedriver.exe");
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
```

```
driver. get ("https://www.monsterindia.com/");
driver. findElement(By. xpath("//span[text()='upload Resume']")).click();
Thread. Sleep (2000);

// Hidden Division Popup will be opened.
WebElement uploadbtn = driver. findElement(By.id("file-upload"));
uploadbtn.sendKeys("C:\\Users\\HP\\Desktop\\My Resume.docx");

// File Download Popup
System. setProperty ("webdriver.gecko.driver",
".\drivers\geckodriver.exe");
WebDriver driver = new FirefoxDriver;
driver. manage(). window(). maximize();
driver. get("https://www.selenium.dev/downloads/");
Thread. sleep(2000);
driver. findElement(By. xpath("(//P[text()='Java']/following-sibling::p[@class=
Thread. Sleep (2000);
Robot r = new Robot;
r. keyPress (keyEvent. VK_DOWN);
Thread. sleep (2000);
r. keyPress (keyEvent. VK_ENTER);
r. keyRelease (keyEvent. VK_DOWN);
r. keyRelease (keyEvent. VK_ENTER);

// Handling Browser Notification Popup
System. setProperty ("webdriver. chrome. driver",
".\drivers\chromedriver. exe");
// Change the browser settings                    //FirefoxOptions
ChromeOptions co = new ChromeOptions();           //OperaOptions
```

```
co.addArguments("--disable-notifications");
co.addArguments("start-maximized");  // instead of using driver.manage() method chaining
// open the browser with the changes
WebDriver driver = new ChromeDriver(co);
driver.get("https://www.yatra.com/");
}
}
```

29.09.2021 D23

Handling Frames

Frame: One webpage present inside another webpage is called inner webpage | nested webpage or frames.

Frames are created by using iframe tag. By using src attribute we pass path of other webpage which we need on current webpage.

To check on the webpage whether an element is present inside frame. Right click and if we see word 'frame' in the options then the element is inside the frame (Firefox Browser).

Q: How to handle frames?

Handled by switching to frame by using overloaded frame() method.

Three ways to switch:

frame(int arg) - index of the frame, index - 0

frame(String arg) - either 'id' or 'name' of the frame

frame(WebElement arg) - address of the frame

Q: How to switch control back from a frame to parent frame?

To switch from a frame to its immediate parent frame, we use driver.switchTo().parentFrame()

Q: How to exit from all the frames and take our control back to the default webpage?

We use driver.switchTo().defaultContent();

Q: How to switch control back from one frame to another frame (not to immediate frame)?

To switch from one frame to another, first switch from a current frame to default webpage and then from default webpage, switch to the desired frame.

→ If an element is present inside the frame and we are trying to handle it <u>without</u> <u>switching</u> we get NoSuchElementException.

→ If there is no frame, but still we try to switch (or)

→ If we are switching to a frame by giving incorrect frame details then we get, NoSuchFrameException.

HTML Source Codes.

Frame : page2.html

```
<html>
<body>
PASSWORD < input type='text' id='i2'>
</body>
</html>
```

Main page :

```
<html>
<body>
USERNAME <input type='text' id='ii'></input>
<iframe Src='page2.html' id='frid' name='frnm'
class='frc1'></iframe>
</body>
</html>
```

EXAMPLE:

```
package arsp1;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class HandlingFrames
{
public static void main(String[] args) throws
InterruptedException
{
System.setProperty("webdriver.chrome.driver",
"./drivers/chromedriver.exe");
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("file:///C:/Users/HP/Desktop/HTML/Page1.html");
Thread.sleep(2000);

driver.switchTo().frame(0);        // Through index
driver.switchTo().frame("frid");   // Through id or name
driver.switchTo().frame("frnm");
driver.switchTo().frame("frc1");   // Through webpage address
```

```
WebElement frAddr = driver.findElement(By.id("frid"));
driver.switchTo().frame(frAddr);
driver.findElement(By.id("i2").sendkeys("manager");
Thread.Sleep(2000);

driver.switchTo().parentFrame();
driver.switchTo().defaultContent();
driver.findElement(By.id("i1")).sendkeys("admin");
```

HANDLING NEW WINDOW / NEW TAB (OR) HANDLING CHILD BROWSER POPUP

Characteristics:

1. It is Colorful.
2. It is movable.
3. It is inspectable.
4. It has minimize and maximize buttons.

Q: How to handle it?

First we need to switch our Control from webpage to Child browser window.
Hence we use, driver.switchTo().window(window handle)
To get window handle we use
getWindowHandle()
getWindowHandles()

Window Handle: It is an alpha-numeric string which is nothing but the unique address generated for every browser window whenever we open it.

getWindowHandle() : String

~~String~~ - Used to get the window handle of the current browser window.

getWindowHandles() : Set<String>

- Used to get the window handle of the all the browser windows.

Close() vs Quit()

Close() - closes the current browser window

Quit() - Closes all the browser windows which are opened through Selenium.

Important Interview Questions:

1. Can you close a parent browser window only?
Yes, by using driver.close()

2. Can you close all the browser windows at once?
Yes, by using driver.quit()

3. Can you close all the browser windows without using quit()?
Yes, first we get window handle of all the browsers by using getWindowHandles(). Then we switch to each and every browser and close it by using close()

4. Can you perform action on a particular browser?
Yes, we get the title of every browser we switch and compare to the one which we want.

5. Can you close all the child browsers only?
Yes, First we take parent browser window title, then we write condition if title not matching, close the browser.

NoSuchWindowException : Selenium - Runtime
Reason: If we are trying to switch to a window which is closed and session is expired then we get this exception.

EXAMPLE :

```
package arsp1;
import java.util.Set;
import org.openara.Selenium.By;
import org.openara.Selenium.WebDriver;
import org.openara.selenium.chrome.ChromeDriver;

public class HandlingChildBrowserPopup
{

public static void main(String[] args) throws Interr-
uptedException
{

System.setProperty("webdriver.chrome.driver",
"./drivers/chromedriver.exe");
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://www.haukari.com/");
```

```
driver.close();  // Closing parent browser window only
driver.quit();   // closing all browsers at once.

// To close all the browser windows without using quit():

Set<String> allwhs = driver.getWindowHandles();
System.out.println(allwhs.size());

for (String wh : allwhs)
{
    driver.switchTo().window(wh);
    driver.close();
    Thread.sleep(1000);

// To perform action on a particular browser?

Set<String> allwhs = driver.getWindowHandles();

for (String wh : allwhs)
{
    String title = driver.switchTo().window(wh).getTitle();
    if (title.equals("Cognizant"))
{
    driver.manage().window().maximize();
    Thread.sleep(3000);
    driver.findElement(By.xpath("//img[@alt = 'Cognizant']")).click();
    Thread.sleep(3000);
    driver.quit();
    break;
}
}
```

// To close all the child browsers only:

```
String parentTitle = driver.getTitle();
Set<String> allwhs = driver.getWindowHandles();

for(String wh : allwhs)
{
String title = driver.switchTo().window(wh).getTitle();
  if (!title.equals(parentTitle)
{
    driver.close();
    driver.switchTo().window(wh); // NoSuchWindowException
```

01.10.21 D24

WEB ELEMENT (I) – METHODS (Element Level)

Method Type	S.No	Method Name	Return Type	Method Use
Operational Methods	1.	Click()	void	Used to click on any element.
	2.	clear()	void	Used to clear the Content of the element.
	3.	SendKeys(charSequence)	Void	Used to send char-Sequence into the element
Data Capture Methods	4.	getTagName()	String	Used to get the tagName of the element.
	5	getAttribute(String attrName)	String	Used to get a particular attribute value of the element by giving the attribute name

Method Type	S.No	Method Name	Return Type	Method Use
Data Capture Methods	6.	getText()	String	Used to get the text of the element
	7.	getLocation()	Point	Used to get location of the element (means X pixel, Y Pixel)
	8	getRect()	Rectangle	Used to get X Axis, Y Axis, height, width
	9	getCssValue (String cssName)	String	Used to get a particular Css attribute value of the element by giving the css attribute name
	10	getSize()	Dimension	Used to get the size of the element (i.e. length & bx)
	11	getScreenshotAs(output Type)	File	Used to get the Screenshot of a particular element
Verification Methods	12	isDisplayed()	boolean	Used to verify if an element is present on the webpage.
	13	isEnabled()	boolean	Used to verify if an element is enabled or disabled (enabled means ready to interact)
	14.	isSelected()	boolean	Used to verify if an element is selected or not.

Method Type	S.No	Method Name	Return Type	Method Use
Inspector Methods	15	findElement (By arg)	WebElement	Used to find a particular element within the Current element
	16	findElements (By arg)	List <WebElement>	Used to find all the matching elements within the Current element
Operational Method.	17	Submit()	void	Used ~~like~~ click() method ~~only~~ but only can be used on buttons with type = 'submit'

EXAMPLE :

```
package arsp1;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class WebElementMethods
{
public static void main (String[] args) throws
InterruptedException
{

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

```
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://demo.actitime.com/login.do");

WebElement untb = driver.findElement(By.id("username"));
    untb.sendKeys("admin");
    Thread.Sleep(2000);
    untb.clear();

System.out.println("TagName: " + untb.getTagName());
System.out.println("Attribute value: " + untb.getAttribute
                                    ("placeholder"));

WebElement pwtb = driver.findElement(By.name("pwd"));
    pwtb.sendKeys("raghav");

System.out.println(pwtb.getAttribute("value"));
        "          (untb.getLocation());
        "          (untb.getRect().x);
        "          (untb.getRect().y);
        "          (untb.getSize());
        "          (untb.getRect().width);
        "          (untb.getRect().height);
        "          (untb.getCssValue("font-size"));

WebElement chkbox = driver.findElement(By.id("keepLog-
                                    ggedInCheckBox"));
    chkbox.click();

if (chkbox.isDisplayed())
{
    System.out.println(" checkbox is displayed");
```

```
if (chkbox.isEnabled())
{
    System.out.println("Checkbox is Enabled");

if(chkbox.isSelected())
{
    System.out.println("Checkbox is Selected, Don't touch now");
}
else
{
    System.out.println("Click now");
    chkbox.click();
}
}
else
{
    System.out.println("Checkbox is Disabled");
}
}
else
{
    System.out.println("Checkbox is not displayed");
}
}
}
```

AUTOMATION FRAMEWORK

Framework is set of rules and/or guidelines, Files and Folder structure (standard folder template) that every automation engineer should follow while automating the application

(OR)

Framework is Collection of reusable generic methods, which makes automation engineers life easy and also makes Coding efficient.

Advantages of Automation Framework :

1. Contains generic reusable methods which help in Code reusability.
2. Since every automation engineer follows the same format, we can achieve Consistency in Automation testing.
3. Since we follow framework which is very organised, it is easy to develop scripts and easy to understand framework.
4. Maintaining the developed scripts is easy.
5. Reduces Coding effort and reduces time.
6. Increases Coding efficiency.

Stages of Framework :

1. Design : (Generic)
   - Responsible for Design, mostly taken care by the any experienced member.
2. Implementation : (specific to Project)
   - Responsible for implementation, done by Seniors

3. Execution :

- Responsible for execution - Any Automation engineer.

Types of Framework :

1. Data-Driven Framework
2. Keyword-Driven Framework
3. Method Driven Framework
4. TestNG
5. Modular Framework
6. Hybrid Framework

Data-Driven Framework :

Storing test data in external files like Excel sheets, XML, Database, Code, CSV files, Property file and fetching the data through generic reusable methods and passing that data to the application and testing the application is known as Data-Driven Framework.

Q: Why Data-Driven ?

As a part of functional testing, we test each and every Component of the application with both valid and invalid data..
And since there will be many Components, the test data is huge,..
And if we hard-code test data then Code size will be more..
Also difficult to maintain the test data.
Hence we keep it in external files and fetch them from here.

Q: Why do we prefer to store test data in Excel?

1. Maintaining the test data in Excel is very easy (Adding, Modifying and deleting)
2. Already manual test engineers would have used excel to store test data, so we just write generic reusable codes to fetch them.

To achieve Data-Driven (read data) from excel sheet, we need to download Apache POI jars.

Steps:

1. Go to google, type 'Download Apache poi jars'.
2. Click on first link (https://poi.apache.org/download.html).
3. Click on 'The latest stable release is Apache POI 5.0.0' link.
4. Click on 'poi-bin-5.0.0-20210120.tar.gz (56.01 MB)', under Binary distribution.
5. Click on the first link..
6. Once downloaded, uncompress and add all the jars to build path
   a. Right click on Project, go to build-path -> Configure build path.
   b. Click on 'Libraries' tab.
   c. Click on 'Add External Jars'.
   d. Go to the location where poi jars are downloaded and select all the jars in poi-4.1.2 folder and also in 'lib', 'ooxml-lib'

To achieve data driven testing from excel sheet,

First we need to create an excel sheet and write our test data inside it..

Create a folder called 'data' under the project and create an excel under data folder.

- Right click on Project → New → Folder (name as 'data')
- Right click on data folder → Properties → Copy path → open window explorer → paste it in the address bar..
- Right click inside folder → New → Excel sheet (name as inputData).

→ To read the data, we open the file in read mode, we use POI libraries and guide our control to a particular cell and get the value in it.

→ To read multiple data, use loop to get data one after another.

→ To write the data, we open the file in read mode, prepare the cell and once we write we open in write mode and save it

Note: Create an Excel file, fill the data related to Freedom fighters.

Note :

WorkBook (I) ← WorkBookFactory (C)
getSheet() ← create()

Sheet (I)
getRow()

Row (I)
getCell()

Cell (I)
getStringCellValue()

* WorkBookFactory.create().getSheet().getRow().getCell().getStringCellValue()

EXAMPLE : (Read Data from Excel)

```
package datadrivenframework;

import org.apache.poi.ss.usermodel.Cell;
       "          "          "        .Row;
       "          "          "        .Sheet;
       "          "          "        .WorkBook;
       "          "          "        .WorkBookFactory;

public class ReadDataFromExcel
{
public static void main(String args[]) throws
                                        Throwable
```

```
// open file in read mode
FileInputStream fis = new FileInputStream
( " D:\\KCSM6\\Automation Project\\data\\inputData.
                                        xlsx");
    // Take Control of that file
WorkBook wb = WorkBookFactory.create(fis);

    // Go to that particular Sheet
Sheet sh = wb.getSheet("Sheet1");

    // Go to that Particular Row.
Row r = sh.getRow(8);

    // Go to that Particular Cell.
Cell c = r.getCell(1);

// Capture the string value present inside it
String xlVal = c.getStringCellValue();

    // Print it
System.out.println(xlVal);
}
}
```

(OR)

06.10.2021 D26

```
FileInputStream fis = new FileInputStream(".\data\input-
                                        Data.xlsx");
WorkBook wb = WorkBookFactory.create(fis);
System.out.println(wb.getSheet("sheet1").getRow(4).
        getCell(0).getStringCellValue());
}
}
```

Generic Method for Excel related

Read Multiple Data :

```
package datadrivenframework;
import java.io. FileInputStream;
import org. apache.poi.ss. usermodel. workbook;
import org. apache. poi. ss. usermodel. WorkbookFactory;

public class ReadMultipleData
{
public static void main(String[] args) throws Throwable
{
for (int i=0; i<=10; i++)
{
FileInputStream fis = new FileInputStream(".\data\input-Data. xlsx");
Workbook wb = WorkbookFactory. create (fis);
System. out. println (wb. getSheet ("sheet1"). get Row (i).
        get Cell (0). getStringCellvalue());
}
}
```

Write Data into Excel :

```
SC, import java. io. FileOutputStream;
public class WriteDataIntoExcel
{
public static void main(String[] args) throws Throwable
{
// Open file in read mode
FileInputStream fis = new FileInputStream(".\data\inputData.xlsx");
```

```
// Go to particular file
Workbook wb = WorkbookFactory.create(fis);

// Go to particular sheet
Sheet sh = wb.getSheet("Sheet1");

// Go to particular Row
Row r = sh.getRow(6);

// Keep the cell ready for writing purpose
Cell c = r.createCell(2);

// Provide data
c.setCellValue("Raghav Sir");

// open file in write mode
FileOutputStream fos = new FileOutputStream
                        (".\data\inputData.xlsx");

// Save the changes made in the file
wb.write(fos);
System.out.println("Done writing");
```

File Library: (Generic Methods)

```
package dataDrivenFramework;
  SC
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
```

```
public class FileLib
{
public string readExcelData(String excelPath, String sheetName,
                int row, int cell) throws Throwable

FileInputStream fis = new FileInputStream(excelPath);
    Workbook wb = workbookFactory. Create(fis);
String excelValue = wb. getSheet(sheetName). getRow(row).
                getCell(cell). getStringCellValue();
        return excelvalue;
}

public int getRowCount(String excelPath, String sheetName)
    throws Throwable
{
FileInputStream fis = new FileInputstream(excelPath);
Workbook wb = WorkbookFactory. create(fis);
int rowCount = wb. getSheet(sheetName). getLastRowNum();
    return rowCount;
}

public void writeExcelData(String excelPath, String sheetName,
                int row, int cell, String data)
                throws Throwable
{
    FileInputstream fis = new. FileInputStream(excelPath);
    Workbook wb = workbookFactory. create(fis);
    wb. getSheet(sheetName). getRow(row). create cell(cell).
                setCellvalue(data);

    FileOutputStream fos = new FileOutputstream(excelPath);
        wb. write(fos);
}
}
```

Data - Driven main class :

```
package dataDrivenFramework;
public class DataDrivenMainClass
{
public static void main(String[] args) throws
Throwable
{
FileLib flib = new FileLib();
int rc = flib.getRowCount("./data/inputData.xlsx",
                                    "Sheet1");
System.out.println(rc);
for (int i=0; i<=rc; i++)
{
String value = flib.readExcelData("./data/inputData.
                                    xlsx", "Sheet1", i, 1);
System.out.println(value);

flib.writeExcelData("./data/inputData.xlsx", "Sheet1",
                                    i, 2, "Raghav Sir");
}
}
}
```

KEYWORDDRIVEN FRAMEWORK

Storing the test data which is commonly or frequently used, as key-value pair and reading it by using generic reusable method is known as KeywordDriven Framework.

Here we store test data which we use very frequently like browserName, url, valid user-name and password to login etc, in every test case. as key-value pair in property file and we use a java class called Properties class and its method getProperty to get the data

getProperty() takes key as argument and returns its corresponding value.

Creating property file in our project.

1. Right click on the folder in which we want to create it ..> New --> File (name it as 'Config' and give extension as .property or .properties)

2. store test data in key-value pair as below.

Key	value
browser	chrome
url	https://www.actitime.com
username	admin
password	manager

Property File: (config.properties)

```
Trainer     Raghav Sir
browser     chrome
url         https://demo.actitime.com/login.do
username    admin
password    manager
```

IAutoConsts: Interface

```
package hybrid_datadriven_keywordDriven;

public interface IAutoConsts
{
String PROP_PATH = "./data/Config.properties";
String EXCEL_PATH = "./data/inputData.xlsx";

String CHROME_KEY = "webdriver.chrome.driver";
String CHROME_VALUE = "./drivers/chromedriver.exe";

String GECKO_KEY = "webdriver.gecko.driver";
String GECKO_VALUE = "./drivers/geckodriver.exe";
}
```

Read Data From Property File

```
package keywordDrivenFramework;
import java.io.FileInputStream;
import java.util.Properties;

public class ReadDataFromPropertyFile
{
public Static void main(string[] args) throws Throwable
{
    // Open file in read mode
FileInputStream fis = new FileInputStream("./data/
                                    Config.properties");

    // Create an Object of Properties class
Properties prop = new Properties();

// Keep the file ready for reading purpose
    prop.load(fis);

// Give the key to get its value
    String value = prop.getProperty("browser");
(OR) String value = prop.getProperty("browse", "Incorrect
                                    key");
    // Print it
      System.out.println(value);
  }
}
```

FileLib

```
package keywordDrivenFramework;
import java.io.FileInputStream;
import java.util.Properties;

public class FileLib
{
public String readPropertyData(String propPath,
        String key) throws Throwable
{
FileInputStream fis = new FileInputStream(propPath);
Properties prop = new Properties();
prop.load(fis);
String propValue = prop.getProperty(key, "InCorrect key");
return propValue;
}
}
```

KeywordMainclass

```
package keywordDrivenFramework;
public class keywordMainclass
{
public static void main(String[] args) throws Throwable
{
FileLib flib = new FileLib();
String br = flib.readPropertyData(".\data\Config.properties",
System.out.println(br);
"username");
}
}
```

(HYBRID DRIVEN)

BaseTest

```
package hybrid_datadriven_keywordDriven;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.firefox.FirefoxDriver;

public class BaseTest implements IAutoConsts
{

public static WebDriver driver;
public void openBrowser() throws Throwable
{

FileLib flib = new FileLib();
String browserValue = flib.readPropertyData(PROP_PATH,
                                            "browser");
if (browserValue.equalsIgnoreCase("chrome"))
{
System.setProperty(CHROME_KEY, CHROME_VALUE);
driver = new ChromeDriver();
}
else if (browserValue.equalsIgnoreCase("firefox"))
{
System.setProperty(GECKO_KEY, GECKO_VALUE);
driver = new FirefoxDriver();
}
else
{
System.out.println("Enter proper Browser name"));
}
driver.manage().window().maximize();
String appUrl = flib.readPropertyData(PROP_PATH, "url");
driver.get(appUrl);
}
```

```
    public void closeBrowser()
{
    driver.quit();
}
```

FileLib (Add-ons)

```
package hybrid_datadriven_KeywordDriven;

import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.util.Properties;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.ss.usermodel.WorkbookFactory;

public class FileLib
{
public String readExcelData(String excelPath, String sheet
Name, int row, int cell) throws Throwable
FileInputStream fis = new FileInputStream (excelPath);
Workbook wb = WorkbookFactory.Create(fis);
String excelValue = wb.getSheet(sheetName).getRow
(row).getCell(cell).getStringCellValue();
return excelValue;
}

public int getRowCount(String excelPath, String
sheetName) throws Throwable
{
FileInputStream fis = new FileInputStream(excelPath);
Workbook wb = WorkbookFactory.Create(fis);
int rowCount = wb.getSheet(sheetName).getLastRow
num();
return rowCount;
}
```

```
public void writeExcelData(String excelPath, String
    sheetName, int row, int cell, String data) throws
                                        Throwable
{
FileInputStream fis = new FileInputStream(excelPath);
Workbook wb = WorkbookFactory.create(fis);
wb.getSheet(SheetName).getRow(row).createCell(cell).
    setCellValue(data);

FileOutputStream fos = new FileOutputStream(excelpath);
wb.write(fos);
}
```

Valid Login Test

```
package hybrid_datadriven_keywordDriven;

public class ValidLoginTest extends BaseTest
{
public static void main(String[] args) throws Throwable
{
BaseTest bt = new BaseTest();
bt.openBrowser();
}
}
```

08.10.2021 D28

WebDriverCommonLib

```
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
```

```
public class WebDriverCommonLib
{
public String getPageTitle()
{
String pageTitle = BaseTest.driver.getTitle();
return pageTitle;
}
public void waitForTitle(String title)
{
WebDriverWait wait = new WebDriverWait(BaseTest.
                                        driver, 20);
wait.until(ExpectedConditions.titleContains(title));
}

public void verify(String expected, String actual,
                                        String page)
if (actual.equals(expected))
{
System.out.println(page + "is Displayed, PASS");
}
else
{
System.out.println(page + "is not Displayed, FAIL");
}
}
}
```

Config. properties file (Add-ons)

loginTitle actiTIME - Login
homeTitle actiTIME - Enter Time-Track

Base Test - Add ons before Closing Browser

```
WebDriverCommonLib wlib = new WebDriverCommonLib();
wlib.verify(flib.readPropertyData(PROP_PATH, "loginTitle"),
  wlib.getPageTitle(), "Login");
}
```

File Lib - (Add ons)

```
String excelValue = wb.getSheet(sheetName).getRow(row).
                    getCell(cell).toString();
```

Invalid Login Test:

```
package hybrid_datadriven_keywordriven;
import org.openqa.selenium.By;
public class InvalidLoginTest extends BaseTest
{
public static void main(String[] args) throws Throwable
{
BaseTest bt=new BaseTest();
  bt.openBrowser();
FileLib flib= new FileLib();
int rc = flib.getRowCount(EXCEL_PATH, "Invalid");
for (int i=1; i<=rc; i++)
{
String un = flib.readExcelData(EXCEL_PATH, "Invalid", i,0);
String pw = flib.readExcelData(EXCEL_PATH, "Invalid", i,1);

driver.findElement(By.id("username")).sendKeys(un);
Thread.sleep(2000);
driver.findElement(By.name("pwd")).sendKeys(pw);
driver.findElement(By.xpath("//div[text()='Login']"))
```

```
    .click();
    Thread.sleep(6000);
  }
}
```

input Data (EXCEL)

Username	Password
ad123	manager
ad#$%	manager
	manager
ad123#$%	manager
12345	manager
ad min	manager

11.10.2021 D29

StaleElementReferenceException

Q: What is it?

The reference or address of the element is old

Q: Why do we get this exception?

When we use findElement() / findElements() to get the address of an element, Store it in a reference variable, but before performing action, if the webpage is refreshed, then on the webpage element's address will change, but in reference variable we still have old address, and we will be trying to perform action on the old address only.

Hence we get this exception.

Q: How to handle it?

By using Page Object Model (POM)

Q: What is Page Object Model?

Page Object Model is one of the Java Design Patterns.

Mainly it's an Object Repository which is used to store all the WebElements in one place. Hence it is also called Element Repository / Page Object Repository.

Q: How does POM handle StaleElementReferenceException?

In page object model, we use @FindBy to get the address of the elements. @FindBy will always get the new address / Current address / latest address of the elements therefore address will never become old. Hence we can avoid getting this exception.

Q: How do we declare elements in POM class?

POM class means any Class which Contains @FindBy is known as POM class..

In this class we declare by using @FindBy.

Syn: @FindBy(locatorName = "locatorValue") AccessSpecifier ReturnType ElementName;

Eg: @FindBy (id = "username") private WebElement unTB;

26 Sunday

Q: How do we initialize all the elements in POM Class?

We use initElements() of PageFactory Class to initialize all the elements at once. initElements()

gives instructions to all the elements (@FindBy) stored to fetch the address and store it inside reference variable.

initElements() takes two Arguments

1. WebDriver reference variable - means browser class object reference.
   Eg: driver

2. Object Page - means in which class we have stored the elements. If the elements are in the same class/ current class only then we use 'this'.
   PageFactory.initElements(driver, this);

   If the elements are in some other class, Just create an object of that class and pass reference variable as argument

   PomClass pc = new PomClass();
   PageFactory.initElements(driver, pc);

Q: Is User defined Constructor Compulsory to initialize elements?
No, but initializing the elements with initElements() is mandatory. If not initialized then we get NullPointerException.

Q: How to handle multiple Elements in POM Class?
Use the return type as List<WebElement>

Q: How do we do utilization in POM class?

By creating public getter and setter methods.

Address returning method

```
public WebElement getUntb()
{
    return untb;
}
```

Action Performing method

```
public void setUsername(String un)
{
    untb.sendKeys(un)
}
```

Uses of Page Object Model

1. It is used to store all the elements in one place. Hence it acts as element Repository / Object Repository.
2. It can avoid StaleElementReferenceException
3. Code Reusability is achieved, which increases the efficiency of test script development.
4. Maintaining automation scripts is easy (Especially when requirements are frequently changing)
5. Helps in encapsulating the elements.
6. Can achieve abstraction
7. Can achieve method Driven testing

METHOD DRIVEN FRAMEWORK

Writing the implementation of the methods in different classes like generic classes or POM class and calling the methods in test class and executing the scripts by the help of these methods

is known as Method Driven Framework.

Login Page :

```
08 package sere;
10 import org.openqa.selenium.WebElement;
   import org.openqa.selenium.support.FindBy;
11 import org.openqa.selenium.support.PageFactory;
12 import hybrid_datadriven_keywordDriven.BaseTest;
01 public class LoginPage
{
02      // Declaration
   @FindBy(id="username") private WebElement untb;
03      // Initialization
04 public LoginPage
{
05 PageFactory.initElements(BaseTest.driver, this);
}
06      // Utilization
   public WebElement getUntb()
07 {  return untb;
   }
{ public void typeUn(String un)
   untb.sendKeys(un);
   }
}
```

① — ② line onwards

Handling StaleElementReferenceException

```
import org.openqa.selenium.By;

public class HandlingStaleElementReferenceException
extends BaseTest
{
public static void main(String[] args) throws Throwable
{
BaseTest bt=new BaseTest();
bt.openBrowser();

LoginPage lp=new LoginPage();
driver.navigate().refresh();
lp.typeUn("admin");
}
}
```

Login Page - (Add ons)

```
@FindBy(name="pwd") private WebElement pwtb;
@FindBy(xpath="//div[text()='Login']") private
                    WebElement loginBtn;

public LoginPage()
{
PageFactory.initElements(BaseTest.driver, this);
}

public WebElement getPwtb()
{
    return pwtb;
}

public void setPwtb(String pwd)
{
    pwtb.sendKeys(pwd);
}
```

```
    public webElement getLoginBtn()
08  {
        return loginBtn;
09  }
    public void clickLoginBtn()
10  {   loginBtn.click();
    }
11  public webElement getUntb()
    {
12      return untb;
    }
01  public void setUntb(String un)
    {
02      untb.sendKeys(un);
    }
03  public void login(String un, String pw)
    {
04      untb.sendKeys(un);
        pwtb.sendKeys(pw);
05      loginBtn.click();
    }}
06
```

Valid Login Test.

```
07
    package methodDrivenFramework;
    public Class ValidLoginTest
    {
    public Static Void main(String[] args) throws Throwable
    {
    // open Browser, enter URL, verify login page
```

```
    BaseTest bt = new BaseTest();
    bt.openBrowser();

    // Login into application and verify Homepage
    LoginPage lp = new LoginPage();
    lp.login("admin", "manager");
  }
}
```

12.10.2021 D30

Unit Testing Tools :

Java	-	JUnit
.NET	-	NUnit
Java + .NET	-	TestNG (TestNextGeneration)
PHP	-	PHPUnit
Python	-	PyDev
Ruby	-	Rspec

TestNG :

→ It is a unit testing tool which supports Java and .NET.

→ It is a third party tool developed by Cedric Beust which is the next version of JUnit.

→ Used by Developers in WBT and Automation Test Engineers in BBT which makes automation test engineers life easy.

02 Sunday

Uses :

1. Obtain the output in the form of emailable HTML reports.

2. Perform Batch execution.
3. (Running multiple Scripts with Single Click by Converting all the test Scripts into a test Suite (Collection of multiple Test Scripts --> Test Suite)).
3. Provides lot of Annotations and flag which are Very helpful in test Scripts development.
4. Perform Grouping Execution
5. Perform Data Parameterization
6. Achieve data-driven Concept
7. Perform Parallel Execution
8. Perform cross Browser testing
9. Provides better Assertion features
10. Provides many flags and test methods dependency features.
11. Provides Listeners feature which can Capture Screenshots dynamically.

Download and Install TestNG:

Since TestNG is an eclipse plug-in
1. In Eclipse → Help → Install New Software → Click on 'Add' button,
Provide name as TestNG and location as https://testing.org/testing-eclipse-update-site'
→ Select testNG and click on Install, follow instructions till end..
Restart the eclipse..

Adding TestNG to our project
1. Right click on the project → Go to BuildPath → Add Libraries → Select TestNG → Click on

Next and Finish button

Rules of TestNG:

1. Never use "Default Package"
   → Always create a class under some package
2. Never use main()
   → Use @Test
3. Never use SOP for printing
   → Use Reporter.log()

There are two types of reports

1. High-Level Reports : Overview Report which gives Status of the testscripts, Contains no. of tests passed, failed, skipped .. etc.

2. Low-level Reports : Detailed Report which is given by the programmar.

Reporter.log("abc") → abc will be printed in report only.

Reporter.log("abc", true) → abc will be printed in both report as well as Console.

Test Class :

Any class which Contains @Test is Called Test Class

(Class Demo)

Q: How to see the TestNG Report ?

1. Run the above Test Class
2. Refresh the Project (Right click on Java Project → Refresh).

3. New folder called test-output will appear.
4. Expand it --> Go to emailable-report.html
--> Right click --> Open with --> Web Browser

Batch Execution: means to execute multiple Test classes with single click by Converting all the required test classes into Test Suite.

Q: How to generate Test Suite?
Right Click on the project →
TestNG →
Converting to TestNG →
Finish.
A .xml file is generated as below which is nothing but the test suite / Batch Executor.

```
<suite name="Suite">
  <test name="Test">
    <classes>
      <class name="testNG.Sumo"/>
      <class name="testNG.Demo"/>
      <class name="testNG.Remo"/>
    </classes>
  </test>
</suite>
```

To execute the test suite, Right click on testNG.xml →
Run as →
TestNG suite.

Flags of TestNG.

1. priority - Used to control order of execution of test methods within a class
2. enabled - Controls the execution of test methods, will not execute if enabled = false.
3. invocationCount=3 - means test method will loop itself and run for 3 times.
4. dependsOnMethods - adds dependency feature, wherein a test method executes only if its dependant method is PASS.
5. parameters - Used to pick a specific parameter
6. groups - Used in grouping required test methods and executing them.
7. name - Used for identification
8. dataprovider - Used in data driven Concept of TestNG.
9. description - Used to give brief description about a test method.
10. alwaysRun - If true, then it will run always even if dependent methods are failed.

Q: Can we have multiple test methods in a test class?
YES

Q: What is the order of execution of test methods if multiple are present?
Alphabetical order of their method name.

Q: Can you execute a test method after another test method is executed?
YES, using priority flag.

Q: Can you execute a test method only after another test method is PASS?
YES, by using dependsOnMethods flag.

dependsOnMethods Flag:
By using this flag we can add dependency feature, means one test method can be dependent on 1 or more methods and only if all the methods are executed and passed, then dependent method will execute..

Example: If there are 2 methods, transfer1() and transfer2() and one more method called delete() is dependent on transfer1() and transfer2() then delete will execute only if transfer1() and transfer2() gets executed and passed.

Either or both of the methods are fail, then delete() gets "Skipped" and note that if alwaysRun flag is present then it will ignore dependsOnMethods flag.. thereby even if transfer1() fails delete() will execute..

when dependsOnMethods flag is present then priority flag will not have importance.

we Cannot disable register() because delete() is dependent on register(). So if register() wont run, then delete() also won't run.

Demo:

```
package testng;
import org.testing.Reporter;
import org.testing.annotations.Test;

public class Demo
{
  @Test
  public void raghav1()
  {
    System.out.println("I am SOP message");
    Reporter.log("I am report message");
    Reporter.log("Report + Console", true);
  }
}
```

Remo

```
public class Remo
{
  @Test
  public void raghav2()
  {
    Reporter.log("raghav2", true);
  }
}
```

Sumo.

```
public class Sumo
{
@Test
public void raghav3()
{
Reporter.log("raghav3", true);
}
}
```

Flags of TestNG:

```
package testng;
import org.testng.Reporter;
import org.testing.annotations.Test;

public class FlagsofTestNG
{
@Test(priority=1)
public void run()
{
Reporter.log("run method", true);
}
@Test(priority=2, enabled=false)
public void bun()
{
Reporter.log("bun method", true);
}
@Test(priority=3, invocationCount=10)
public void gun()
{
Reporter.log("gun method", true);
}
}
```

Dependency Feature

```
package testng;
import org.testng.Reporter;
import org.testng.annotations.Test;

public class DependencyFeature
{
    @Test(priority=1)
    public void register1()
{
    Reporter.log("Registered Successfully", true);
}
    @Test(priority=2)
    public void register2()
    {
        Assert.fail();
        Reporter.log("Registered Successfully", true);
    }
    @Test(priority=3, dependsOnMethods={"register1",
                                        "register2"})
    public void delete()
{
    Reporter.log("Deleted Successfully", true);
}
}
```

13.10.2021 D31

ANNOTATIONS

@BeforeSuite : It executes before start of every test Suite, Connects to Database Program.

@BeforeTest : It executes before every test runner, Custom report - keep ready.

@BeforeClass : It executes before every class, Open the browser, enter the URL.

@ BeforeMethod : It executes before every test method, Login to App program

@Test : This annotation is used for execution purpose, we write Test Cases..
CreateCustomerTest, EditCustomerTest.

@ AfterMethod : It executes after every test method, Logout program.

@ AfterClass : It executes after every class, Close Browser program

@AfterTest : It executes after every testRunner, Save the report program.

@ AfterSuite : It executes after the end of every test suite, Disconnect from database.

@BS - connectDB()
@BT - reportInitialization
@BC - openBrowser()
@BM - login()
@Test - CreateCustomer()
@AM - logout()

@BM - login()
08 @Test - editCustomer()
@AM - logout()
09 @BM - - login()
@Test - deleteCustomer()
10 @AM - logout
@AC - closeBrowser()
11 @BC - openBrowser()
@BM - login()
12 @~~Test~~ - CreateProject()
@AM - logout()
01 @AC - closeBrowser()
@AT - saveReports
02 @AS - DisconnectDB()

03 AnnotationsOfTestNG

```
package testng;
04 import. org.testing.Reporter;
import org.testing.annotations.AfterClass;
05 " " " " .AfterMethod;
" " " " .AfterSuite;
06 " " " " .AfterTest;
" " " " .BeforeClass;
07 " " " " .BeforeMethod;
" " " " .BeforeSuite;
" " " " .BeforeTest;

public abstract class AnnotationsOfTestNG
{
@BeforeSuite
public void a()
{
```

```
    Reporter.log("BeforeSuite", true);
08 }
   @AfterSuite
09 public void b()
   {
10 Reporter.log("AfterSuite", true");
   }
11 @BeforeTest
   public void c()
12 {
   Reporter.log("BeforeTest", true");
01 }
   @AfterTest
02 public void d()
   {
03 Reporter.log("AfterTest", true);
   }
04 @BeforeClass
Public void e()
05 {
   Reporter.log("BeforeClass", true);
06 }
   @AfterClass
07 public void f()
   {
   Reporter.log("AfterClass", true);
   }
   @BeforeMethod
   public void g()
{
   Reporter.log("BeforeMethod", true);
   }
```

```
@AfterMethod
public void h()
{
    Reporter.log("AfterMethod", true);
}
```

AnnotationsMainClass1

```
package testng;
import org.testng.Reporter;

public class AnnotationMainClass1 extends
                AnnotationsOfTestNG
{
    @Test
    public void runAnn1()
    {
        Reporter.log("I am runAnn test method1", true);
    }
    @Test
    public void runAnn2()
    {
        Reporter.log("I am runAnn test method2", true);
    }
}
```

AnnotationsMain ~~Metho~~ class2

```
package testing;
import org.testing.Reporter;
```



```
public class AnnotationMainClass2 extends
                AnnotationsOfTestNG
{
    @Test
```

```
    public void runAnn1( )
{
    Reporter.log ("I am runAnn test method3", true);
}
}

Suite:
    <Suite name = "Suite">
      <test name = "test">
        <classes>
          <class name = "testing.AnnotationMainClass"/>
        </classes>
      </test>
    </Suite>

Suite:
    <suite name = "Suite">
      <test name = "Test">
        <classes>
          <class name = "testing.AnnotationMainClass2"/>
          <class name = "testng.AnnotationMainClass1"/>
        </classes>
      </test>
    </suite>

Suite:
<suite name = "Suite">
  <test name = "TestRunner1">
    <classes>
      <class name = "testng.AnnotationMain Class1"/>
    </classes>
  </test>
```

```
<test name = "TestRunner2" >
  <classes>
    <class name = "testng.AnnotationMainClass2"/>
  </classes>
  </test>
</suite>
```

14.10.2021 D32

Assertion in TestNG:

Assertion means verification, it means verifying the actual result is same as Expected Result.

There are two types of Assertion in TestNG.

1. Assert class (HardAssert)
2. SoftAssert class (verify)

Difference between Assert and SoftAssert (verify)

Assert:

1. We use Assert to verify mandatory or critical features.
2. In Assert, if verification fails then execution stops of that test method, and we get Assertion Error.
3. All the methods are static, so no need to create any object.

   Assert.assertEquals(actual, expected);
4. Here we don't use assertAll().

SoftAssert (verify)

1. We use SoftAssert to verify non-mandatory or non-critical features.
2. In SoftAssert, even if verification fails the execution continues and at the end we get Assertion error.
3. All the methods are non-static, so we need to create object.

```
SoftAssert sa = new SoftAssert();
sa.assertEquals(actual, expected);
```

4. Here assertAll() is compulsory, and use as the last statement of the test method.

Methods of Assert:

1. Assert.assertEquals(actual, expected)
   Compares actual with expected, and stops the execution if not matching.
2. Assert.fail() - Used to intentionally fail a method for testing purpose.
3. Assert.assertTrue(Condition) - Expects Condition to become true, stops execution if false.
4. Assert.assertFalse(Condition) - Expects Condition to become false, stops execution if true.

Group Execution:

planning our execution in such a way that, we group the required test methods and execute the group to run those particular test scripts that we need.

We achieve grouping by two things

1. By groups flag:
We use groups flag to add test method to a particular group by giving groupname
@Test (groups = "SmokeTests")

2. In XML file, we use tags like
<groups> - Used for executing groups
<define> - Used for define all the groupNames in XML
<run> - Used for executing particular group included in it.
<include> - Used to include a group for execution purpose.
<exclude> - Used to exclude a group so that we don't want it to execute.

DATA - DRIVEN CONCEPT in TestNG

Process of storing the data in data banks/ data Repositories and fetching them into the test Scripts is Data Driven Concept in TestNG

To achieve this, we use two things
1. @DataProvider (DataProvider Annotation)
2. dataProvider flag

1. @DataProvider (DataProvider Annotation)
This annotation is used to create a data repository and supply it to test method.
Here we create an object of 2 Dimension Array and store data inside it.

```
Object[][] obj = new Object[2][3]
        obj[0][0] = "admin1";
        obj[0][1] = "manager1";
        obj[0][2] = "Goa";

        obj[1][0] = "admin2";
        obj[1][1] = "manager2";
        obj[1][2] = "vizag";
```

2 --> 2 rows also means 2 set of data is present.

3 --> 3 Columns also means no. of data present in each set.

Here 2 arrays are present, one for rows and one for Columns.

→ Rows represent how many set of data
→ Columns represent how many parameters we want in test method.
→ we need to give return type of databank method as Object[][]

```
@DataProvider (name = "set2")
public Object[][] dataBank2()
```

2. dataProvider flag:
This flag is used to identify a particular dataBank.
Here we can directly use databank method name / @Test (dataProvider = "dataBank2")

We can use name flag's value of dataBank method / @Test (dataProvider = "set2")

We need to parameterize the test method to catch the data from dataBank and pass it to the test method.

```
public void getCredentials(String un, String pwd,
                           String city)
```

Assertion in TestNG:

```
package testng;
import org.testng.Reporter;
import org.testng.annotations.Test;
import org.testng.asserts.SoftAssert;

public class AssertionInTestNG
{
  @Test
  public void verifyLogin()
  {
    System.setProperty("webdriver.chrome.driver",
        ".\drivers\chromedriver.exe");
    WebDriver driver = new ChromeDriver();
    driver.get("https://demo.actitime.com/login.do");

    Assert.assertEquals(driver.getTitle(), "actiTIME - Login");
    Reporter.log("Login page is Displayed, PAss", true);

    SoftAssert sa = new SoftAssert();
    sa.assertEquals(driver.getTitle(), "actiTIME - Login");
```

```
driver.findElement(By.id("username")).sendKeys("admin")
    Sa.assertAll();
}
}
```

Grouping Execution :

```
package testing;

import org.testing.Reporter;
import org.testng.annotations.Test;

public class GroupingExecution
{
    @Test(groups = "SmokeGrp")
    public void smoke1()
    Reporter.log("Smoke Test1", true);
}

    @Test(groups = "FTGrp")
    public void ft1()
    Reporter.log("FT1", true);
}

    @Test(groups = "ITGrp")
    public void it1()
    Reporter.log("IT1", true);
}

    @Test(groups = "SmokeGrp")
    public void smoke2()
{
    Reporter.log("Smoke Test2", true);
}
```

```
@Test (groups = "RegrGrp")
public void reg1()
{
    Reporter.log("Regr Test 1", true);
}

@Test (groups = "FTGrp")
public void ft2()
{
    Reporter.log ("FT2", true);
}
@Test (groups = "ITGrp")
public void it2()
{
    Reporter.log("IT2", true);
}
@Test (groups = "smokeGrp")
public void smoke3()
{
    Reporter.log ("Smoke Test3", true);
}
@Test( groups = "RegrGrp")
public void regr2()
{
    Reporter. log ("Regr Test 2", true);
}
@Test (groups = "SmokeGrp")
public void smoke4()
{
    Reporter. log ("Smoke Test4", true);
}
}
```

Suite :

```
<suite name = " Suite">
<test name = "Test">
<groups>
<define name = "allgroups">
<include name = "SmokeGrp"> </include>
<include name = "FTGrp"> </include>
<include name = "ITGrp"> </include>
<include name = "RegrGrp"> </include>
</define>
<run>
<include name = "allgroups"> </include>
<exclude name = "SmokeGrp"> </exclude>
</run>
</groups>
<classes>
<class name = "testng. GroupingExecution"/>
</classes>
</test>
</suite>
```

Data driven in TestNG

```
package testng;
import org. testng. Reporter;
import org. testng. annotations. DataProvider;
import org. testng. annotations. Test;

public class DataDrivenInTestNG
{
@DataProvider
public Object[][] dataBank1()
```

```
{
    Object[][]  obj = new  Object[2][2];
        obj[0][0] = "admin1";
        obj[0][1] = "manager1";

        obj[1][0] = "admin2";
        obj[1][1] = "manager2";
        return obj;
}
@DataProvider(name = "db2")
public Object[][]  dataBank2()
{
    Object[][]  obj = new  Object[2][3];
        obj[0][0] = "admin1";
        obj[0][1] = "manager1";
        obj[0][2] = "Goa";

        obj[1][0] = "admin2";
        obj[1][1] = "manager2";
        obj[1][2] = "Hyd";
        return obj;
}
@Test(dataProvider = "db2")
public void getCred(String un, String pw, String city)
{
Reporter.log(un+"---"+pw+"---"+city, true);
}
```

15.10.2021 D33

Data Parameterisation:

Fetching the data through the parameters of test method.

Reading Data from XML,
we use
1. <parameters> tag
2. @Parameters (Parameters annotation>

1. <parameters> tag:
It is used to Store the data.
We use attributes called 'name' and 'value'
<parameter name="browser" value="chrome">
</parameter>

@Parameters - Used to identify which parameter value we require. This is also Method Level Annotation just like @Test which should be written above the method.

@Parameters("Parameter Name") -
It will give parameter value which we can Catch by parameters of test method.

```
@Parameters ("browser")
@Test
public void getParameterValue(String browserName)
Reporter.log (browserName, true)
```

Note: Always run the .xml file because data is coming from it
write XML + Program

Parallel Execution:

Planning our execution so as run the scripts parallelly.

Uses of Parallel Execution:

1. To execute test scripts within less time, hence reducing execution time.
2. To obtain the test execution results faster.

Two types:

1. Distributed Parallel Execution
2. Cross Browser Parallel Execution

Distributed Parallel Execution: In this type, we distribute the test scripts across multiple threads and run all the threads parallelly.

Cross Browser Parallel Execution:
Cross browser means testing application across different browsers.
Cross Browser parallel Execution means executing the same set of test scripts in different browsers parallelly.

30 Sunday

Q: How to run parallelly?
By using the attribute called 'parallel' in <suite> tag.

To run test methods parallelly.

```
<suite name = "Suite" parallel = "methods">
```

To run test classes parallelly

```
<suite name = "suite" parallel = "classes">
```

To run test runners parallelly

```
<suite name = "Suite" parallel = "tests">
```

To stop running parallelly / to run sequentially,

```
<suite name = "Suite" parallel = "none">
```

Note: Maximum thread limit in testNG is '5'
Because if we try to use more than 5 threads, execution performance / speed becomes slow.

Reading Data from XML

```
package testng;

import org.testng.Reporter;
import org.testng.annotations.Parameters;
import org.testng.annotations.Test;

public class ReadingDataFromXML
{
  @Parameters({"browser", "username"})
  @Test
  public void getDataFromXML(String br, String un)
  {
    Reporter.log(br+"---"+run, true);
  }
}
```

Suite:

```
<suite name="Suite">
  <test name="Test">
    <parameter name="browser" value="chrome">
    </parameter>
    <parameter name="username" value="admin">
    </parameter>
    <classes>
      <class name="testng.ReadingDataFromXML"/>
    </classes>
  </test>
</suite>
```

Parallel methods

```
package testng;

import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.annotations.Test;

public class ParallelMethods
{
  @Test
  public void runChrome1()
  {
    System.setProperty("webdriver.chrome.driver",
        "./drivers/chromedriver.exe");
    new ChromeDriver();
  }
  @Test
  public void runChrome2()
  {
      "    "
  }
```

```
    @Test
    public void runChrome3()
  {
        "  "
  }
}
```

Suite :

```
<Suite name = "Suite" parallel = "methods">
  <test name = "Test">
    <classes>
      <class name = "testng.ParallelMethods"/>
    </classes>
  </test>
</suite>
```

Parallel Classes

```
package testng;

import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.annotations.Test;

public class Parallel1
{
  @Test
  public void runChrome1()
  {
    System.setProperty("webdriver.chrome.driver",
        "./drivers/chromedriver.exe");
    new ChromeDriver();
}}
```

Just like previous program,

```
    "       "
public class Parallel2
{
    @Test
    public void runChrome2()
    {
    "       "
    }
```

Just like,

```
public class Parallel3
    =
    =
```

Suite:

```
<suite name="Suite" parallel="classes">
  <test name="Test">
    <classes>
      <class name="testng.Parallel1"/>
      <class name="testng.Parallel2"/>
      <class name="testng.Parallel3"/>
    </classes>
  </test>
</suite>
```

Parallel Tests:

```
<suite name="Suite" parallel="tests">
  <test name="TestRunner1">
    <classes>
      <class name="testng.Parallel1"/>
    </classes>
  </test>
```

```
    <test name = "TestRunner2">
      <classes>
        <class name = "testng. Parallel2"/>
      </classes>
    </test>

  <test name = "TestRunner3">
    <classes>
      <class name = "testng. Parallel3"/>
    </classes>
  </test>
</suite>
```

Suite: CrossBrowser Parallel Testing

```
<Suite name = "Suite" parallel = "tests">
  <test name = "ChromeTest">
  <parameter name = "browser" value = "chrome">
    </parameter>
  <classes>
    <class name = "testng. CrossBrowserParallelTesting"/
  </classes>
  </test>

  <test name = "FirefoxTest">
  <parameter name = "browser" value = "firefox">
    </parameter>
    <classes>
      <class name = testng. CrossBrowserParallelTesting"/>
    </classes>
  </test>
</suite>.
```

Cross Browser Parallel Testing

```
package testng;

import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.annotations.Parameters;
import org.testng.annotations.Test;

public class CrossBrowserParallelTestng
{
  @Parameters("browser")
  @Test
  public void openBrowser(String br)
  {
    WebDriver driver;
    if (br.equals("chrome"))
    {
      System.setProperty("webdriver.chrome.driver",
          "./drivers/chromedriver.exe");
      driver = new ChromeDriver();
    }
    else if( br.equals("firefox")
    {
      System.setProperty("webdriver.gecko.driver",
          "./drivers/geckodriver.exe");
      driver = new FirefoxDriver();
    }
    else
    {
      System.out.println("Enter proper Browser Name");
    }
  }
}
```

Modular Framework :

For each and every module in the application we create a seperate package in the framework and we write all the test scripts related to that module inside that package.

Package Syntax:

domain. ApplicationName. ModuleName

Modules	Package names in our Framework
TimeTrack	Com. actitime. timeTrack
Tasks	Com. actitime. tasks
Reports	Com. actitime. reports
Users	Com. actitime. users
Settings	Com. actitime. settings

List of classes in Com. actitime. genericLib

BaseTest. java (Copy from previous project)
FileLib. java (" ")
IAutoConsts. java (" ")
WebDriverCommonLib. java (" ")

Add-ons in BaseTest.

Add - @Before class annotation, before openBrowser() and and at the end of the method, add - @AfterClass annotation for closeBrowser().

Add-ons for Config. properties file

taskTitle actiTIME - Task List

Addons for WebDriver Common Lib class

```
package Com. actitime .genericLib;

import java. io. File;
import java. io. IOException;
import org. openqa. selenium. OutputType;
   "      "      "      . TakesScreenshot;
   "      "      "      ". WebElement;
   "      "      "      ". Support. ui. ExpectedConditions;
   "      "      "      ". Support. ui. Select;
   "      "      "      ". support. ui. WebDriverWait;
import. org. testng. Assert;
import. org. testng. Reporter;
import Com. google. Common. io. Files;

* Copy - methods like getPageTitle(), waitForTitle(), from
  previous project.

public void verify(String expected, String actual, String page)
{
  Assert. assertEquals (actual, expected);
  Reporter. log (page + " is Displayed, PASS", true);
}
public void selectOption(WebElement element, int index)
{
   Select sel = new Select(element);
      Sel. selectByIndex(index);
}
```

```
public void selectOption(webElement element, String value)
{
    Select sel=new Select(element);
    sel.SelectByValue(value);
}
public void selectOption(String text, webElement element)
{
    Select sel = new Select(element);
    sel.selectByVisibleText(text);
}
public void SwitchToFrame(int index)
{
    BaseTest.driver.SwitchTo().frame(index);
}
public void SwitchTOFrame(String value)
{
    BaseTest.driver.switchTo().frame(value);
}
Public void SwitchToFrame(WebElement element)
    BaseTest.driver.SwitchTO().frame(element);
}
public void fullPageScreenshot(String path)
{
    TakesScreenshot ts = (TakesScreenshot) BaseTest.driver;
    File src = ts.getScreenshotAs(outputType.FILE);
    File dest = new File(path);
    try
    {
        Files.copy(src, dest);
    }
```

```
    catch ( IOException e)
{       e.printStackTrace();
    }
}
public void getElementScreenshot(webElement element,
                                 String path)
{
File src = element.getScreenshotAs(outputType.FILE);
File dest = new File(path);
try
{
Files.copy(src,dest);
}
catch( IOException e)
{
    e.printStackTrace();
}
}
public void acceptAlert()
{
BaseTest.driver.switchTo().alert().accept();
}
public void dismissAlert()
{ BaseTest.driver.switchTo().alert().dismiss();
}

}
```

List of classes in com.actitime.pages.

EnterTimeTrackPage.java
LoginPage.java.

Login Page:

```
package com.actitime.pages;

import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;
import com.actitime.genericLib.BaseTest;

public class LoginPage
{
@FindBy (id = "username") private WebElement untb;
@FindBy (name = "pwd") private WebElement pwtb;
@FindBy (xpath = "//div[text() = 'Login ']") private
 WebElement loginBtn;

public LoginPage()
{
 PageFactory.initElements(BaseTest.driver, this);
}
public WebElement getPwtb()
{
 return pwtb;
}
public void setPwtb (String pwd)
{
 pwtb.sendKeys(pwd);
}
public WebElement getLoginBtn()
{
 return loginBtn;
}
```

```
public void clickLoginBtn()
{
    loginBtn.click();
}
public WebElement getUntb()
{
    return untb;
}
public void setUntb(String un)
{
    untb.sendKeys(un);
}

public void login(String un, String pw)
{
    untb.sendkeys(un);
    pwtb.SendKeys(pw);
    loginBtn.click();
}
}
```

List of classes in Com.actitime.TimeTrack
validLoginTest.java

Valid Login Test:

```
package com.actitime.TimeTrack;
import org.testng.annotations.Test;
import Com.actitime.genericLib.BaseTest;
import Com.actitime.genericLib.FileLib;
import Com.actitime.genericLib.WebDriverCommonLib;
import Com.actitime.pages.LoginPage;
```

```
public class ValidLoginTest extends BaseTest
{
public void loginToApp() throws throwable
{
LoginPage lp= new LoginPage();
FileLib flib = new FileLib();
lp.login (flib. readPropertyData (PROP_PATH, "username"),
flib. readPropertyData(PROP_PATH, "password"));

WebDriverCommonLib wlib = new WebDriver CommonLib();
wlib. waitForTitle (flib. readPropertyData (PROP_PATH,
"homeTitle"));
wlib. verify( flib. readPropertyData (PROP_PATH, "homeTitle"),
wlib. getPageTitle(), "Enter Time Track Page");
}
}
```

Enter Time Track Page:

```
package com. actitime. pages;
import com. actitime. genericLib. BaseTest;
public class EnterTimeTrackPage
{
@FindBy( xpath = "//div [text() = 'Tasks']") private
WebElement tasksTab;

public EnterTimeTrackPage()
{
PageFactory. initElements (BaseTest. driver, this);
}
```

```
    public WebElement getTasksTab()
{
    return tasksTab;
}
    public void clickTasksTab()
{
    tasksTab. click();
}
}
```

List of classes in Com. actitime. tasks

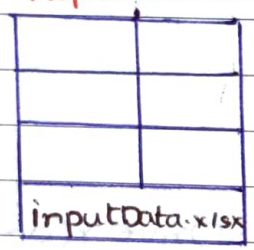
verifyTasksListPageTest. java

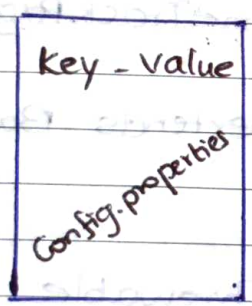
```
Verify Tasks List Page Test :
package Com. actitime. tasks;

import Com. actitime. genericLib. WebDriverCommonLib;
import Com. actitime. pages. EnterTimeTrackPage;

public class VerifyTasksListPageTest extends BaseTest
{
@Test
public void verifyTasks() throws Throwable
{
// Login to actitime and verify homepage
ValidLoginTest vl = new ValidLoginTest();
    vl. loginToApp();

// Click on Tasks tab
EnterTimeTrackPage et = new EnterTimeTrackPage();
    et. clickTasksTab();
```

```
// Verify Tasks Tab is Displayed

WebDriverCommonLib wlib = new WebDriverCommonLib();
FileLib flib = new FileLib();
wlib.verify(flib.readPropertyData(PROP_PATH, "taskTitle"),
    wlib.getPageTitle(), "Task Page");
}
}
```

19.10.2021 D35

③ Data Repositories





① Generic Libraries

- IAutoConsts
- FileLib
- MyListeners
- BaseTest
- WebDriverCommonLib

② Object Repositories

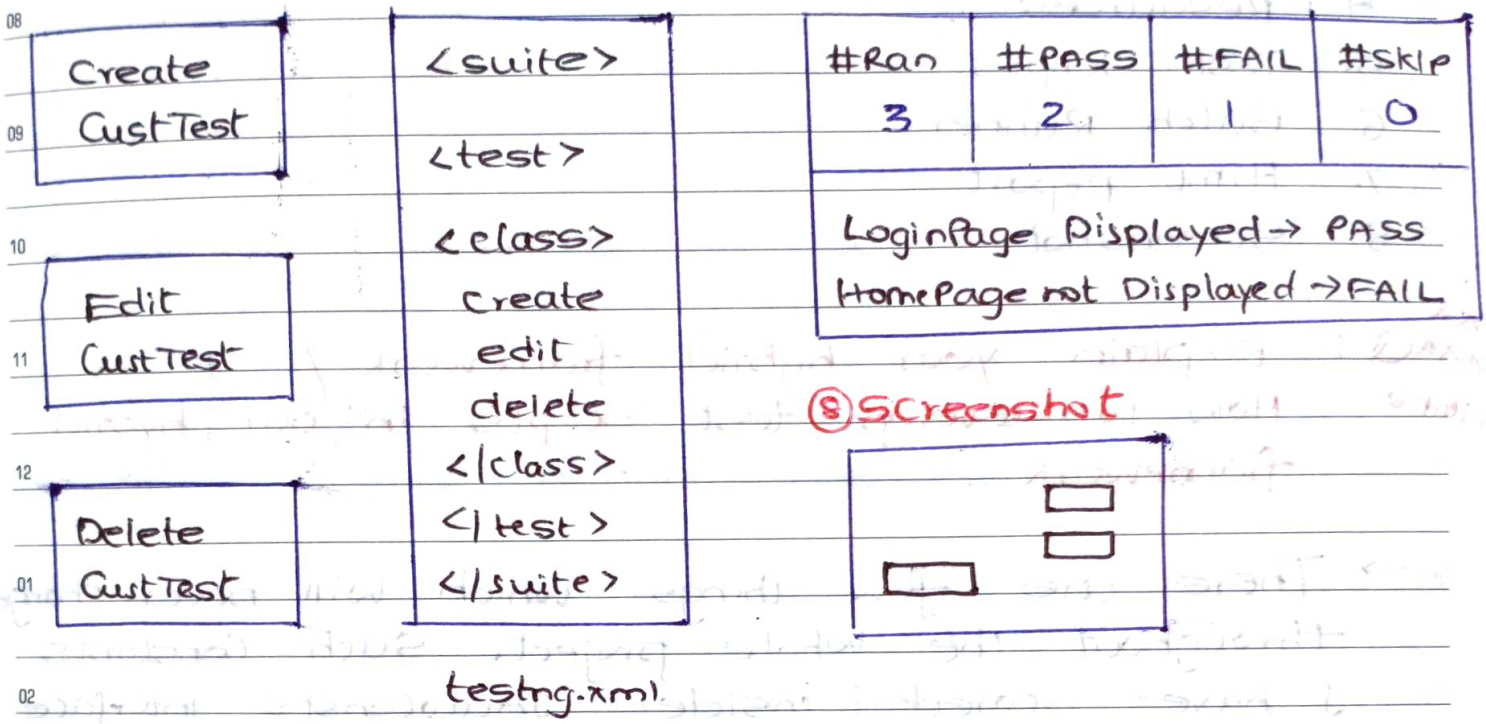
- LoginPage
- UserListPage
- TaskPage

④ Resources

Sel jars
POI Jars
driver exe files

HYBRID FRAMEWORK ARCHITECTURE ↑ ...→

⑤ Test Scripts ⑥ Batch Runner ⑦ Emailable HTML Report



HYBRID FRAMEWORK ARCHITECTURE

Rules of Writing scripts in Framework:

1. Run the test case manually
2. Identify the elements required for the test script and store it in respective pages.
3. Create getters and setters methods.
4. Use generic reusable methods in your scripts and run them.

Hybrid Framework Notes:

There are '8' Components in a framework.

1. Generic Library
2. Object Repository Library
3. Test Data

4. Resources
5. Test Scripts
6. Batch Runner
7. HTML Report
8. Screenshot

***
****Q: Explain your hybrid framework / How to develop test scripts in our hybrid framework.
VVIMPIQ

→ There are few things which will never change throughout the whole project. Such Constants I have stored inside IAutoConsts interface
Eg: Path of excel file and property file, key and value of the driver executable files.

→ There are few things which are repeated like open browser, enter url, verify loginPage and closeBrowser etc.. These repeated actions I am storing inside a Common class for all test scripts called as BaseTest, and I ensure that all the test scripts extends this BaseTest (Inheriting common properties from same place)

→ While automating the application, we require huge set of test Data.. So we store it in a "data repositories" like excel sheets (since easy to maintain data) and we write generic reusable file handling libraries to handle data from them.

→ All the WebDriver related actions will be written in one place as generic reusable libraries once and reused in testscripts whenever required.

→ All the elements on the webpage are stored in one place called Object repository (POM) wherein we declare, initialize and utilize the elements by writing public getters, setters and action methods.

→ In test scripts, we just call generic methods, getters, setters, action methods and execute the automation scripts through these methods. (Test Scripts become light-weight)

→ Since we need to run all my scripts with one click we converted all the test scripts into test suite (Batch executor) and run the suite. (testng.xml file)

→ Finally, we get HTML reports and Screenshots of failed test cases.

Q: Where have you used Java Concepts in your framework?

Inheritance:

Commonly repeated actions we have stored in BaseTest and we make sure every test class extends the BaseTest.

Encapsulation and Abstraction:

We use Object repository to store the web-Elements and we make the elements as private so as to encapsulate them within the POM class also we provide public getters & setters to access them outside the class but cannot be modified.

Also, here since we are hiding the functionality but just providing the public getters and setters, hence we use abstraction concept.

Polymorphism:

While selecting the browser, in the BaseTest, we provide if-else condition wherein depending upon the browser value it decides whether to load ChromeDriver Object or FirefoxDriver object into the same reference variable called driver.

Interface:

In my framework, there were few things, which were never changing like key and value of the driver exe files and excel path and property path. To indicate these are constants, we have created an interface and stored it inside it.

Method - overloading:

In WebDriverCommonLib class, we write overloaded methods for selecting an option from the dropdown.

public void select (WebElement ele, int index)
public void select (WebElement ele, String text)

public void select(String value, WebElement ele)

Method-Overriding :

In Listener, implementation class Called MyListener, we have implemented the methods of ITestListener interface. And, all the methods are getting overriden inside MyListener class.

Abstract Class :

In BaseTest class, though we have two methods, openBrowser() & closeBrowser() [tearDown()] but they cannot run independently.. They are dependent on @Test methods for execution. Since they can't run independently to denote that no automation engineer should run BaseTest class, we made BaseTest class as Abstract Class.

Collection :

While handling multiple elements we use List < WebElement > interface. Also, in a situation of printing elements text without duplicate etc, we use Set interface. So, List, Set are Collection topic.

Array :

To store the data in databanks, we create an object of two dimensional array in DataProvider Concept.

Static variable / Global public Variable:

public Static WebDriver driver;

In BaseTest, once we declare driver as WebDriver and make sure every test script inherits and understands this declaration from BaseTest. Hence to access it everywhere, we declare as public.
To load this declaration first, before every other class wants to access it to make it as static.

MyListeners: (Refer next pages for the notes)

```
package com.actitime.genericLib;

import org.testng.ITestContext;
import org.testng.ITestListener;
import org.testng.ITestResult;
import org.testng.Reporter;

public class MyListeners implements ITestListener
{
    @Override
public void onStart(ITestContext context)
{
Reporter.log(context.getName()+" started", true);
}
@override
public void onTestStart(ITestResult result)
{
Reporter.log(result.getName()+" method started", true);
}
```

```
@override
public void onTestSuccess(ITestResult result)
{
Reporter.log(result.getName()+" method PASSED, true);
}
@override
public void onTestFailure(ITestResult result)
{
Reporter.log(result.getName()+" method FAILED, true);
WebDriverCommonlib wlib = new WebDriverCommonlib();
wlib.fullPageScreenshot("./screenshots/"+result.getName()
                                    +".Png");
}
@override
public void onTestSkipped(ITestResult result)
{
Reporter.log(result.getName()+" method SKIPPED", true);
}
@override
public void onFinish(ITestContext Context)
{
Reporter.log(Context.getName()+" ended", true);
}
```

```
VerifyTasksListPageTest extends BaseTest:

package com.actitime.tasks;
import org.testng.Assert;

@Listeners(com.actitime.genericLib.MyListeners.class)
public class VerifyTasks~~Page~~ListPageTest extends BaseTest
{
```

```
@Test
public void verifyTasks() throws throwable
{
// Login to actitime and verify homepage

ValidLoginTest vl = new ValidLoginTest();
vl.loginToApp();

// Click on Tasks tab
EnterTimeTrackPage et = new EnterTimeTrackPage();
et.clickTasksTab();
Assert.fail();

// verify Tasks Tab is Displayed.

WebDriverCommonLib wlib = new WebDriverCommonLib();
FileLib flib = new FileLib();
wlib.verify(flib.readPropertyData(PROP_PATH, "taskTitle"),
wlib.getPageTitle(), "Task Page");
```

TestNG Listeners:

→ Listeners "listen" to every action/event defined in the selenium script and behave accordingly.
→ It is used in selenium by implementing ITestListener interface of TestNG.
→ It allows Customizing TestNG reports or logs.

Q: How do you integrate TestNG Listeners with our test methods?

We use a class level annotation called @Listeners (path of listener implementation class)

Eg: @Listeners (com.actitime.genericLib.MyListener.class)
public class CreateUser extends BaseTest.

⇒ TestNG Listener has following methods:

onStart - is called when every TestRunner Execution starts.

onTestSuccess - is called on the success of any test.

onTestFailure - is called on the failure of any test.

onTestSkipped - is called on skipped of any test.

onFinish - is called after every TestRunner execution is finished.

Steps:

1. Create a class and make sure it implements ITestListener (interface).
2. Right click inside class, source and click on 'Ad Unimplemented Methods'.
   All the incomplete (abstract) methods will automatically come and fall in this class.
3. Start writing the action you need to perform at each and every listener level.
   OnTestFailure - we can write taking screenshot program so that it will take a screenshot

every time when a test method fails.

4. There are interfaces like ITestContext (Test Runner level) and ITestResult (Test method level) which is helpful in getting test method names by using getName(). method.

20.10.2021 D36

SOURCE CONTROL TOOL / CONFIGURATION MANAGEMENT TOOL / VERSION CONTROL TOOL / SOURCE CODE MANAGEMENT TOOL.

→ GitHub
→ SVN (Sub Version)
→ CVS (Concurrent Version System)
→ Perforce.

It is a decentralised repository in which Source Code of the framework Can be stored. It provides web UI, can be easily accessible anywhere in the world through internet.

There are 2 plug-ins available in Git.
1. GitHub
2. Git

GitHub: It's a cloud based decentralised repo, where we should Create an account with github Community in order to use it.
Basically github is a Storage where we Can create our own repo, that repo can hold the data (Based on money you pay, you get space

10 GB - 10 TB etc). that is being shared across the world.

Advantages of GitHub:

1. It's a cloud based de-centralised repo.
2. No need to invest money for physical hardware.
3. No need of any admin to maintain the storage.
4. It provides web Interface.
5. Can be easily accessible anywhere in the world through internet.
6. It provides Security and backup facility.

Git: It's a client plug-in which should be installed in local system which is used to access github over the net(internet).
Without Git plug-in we can't Communicate to github.

There are many plugins available in the market.

1. E-Git (Eclipse-Git) - Default plugin for latest eclipse.
2. Git-Bash (cmd prompt)
3. Git-Desktop (cmd prompt)

Q: How to Create account with GitHub?

→ Go to google, search for Github login.
→ Navigate to https://github.com/login
→ Click on create an account by providing the details.

Q: How to create a repo in GitHub?

→ Login to GitHub
→ Go to '+' dropdown menu and click on new repo.
→ Give repo name and click on checkbox, initialize repo with read me.
→ Go to code dropdown and copy the git repo Url -
https://github.com/raghavendravinayb/ActitimeRepo.git
UserName :
Password :

Q: How to share existing framework in eclipse through git?

Select the project, Right click → Team → Share project, Create (Path) - finish - finish, NO-HEAD means successfully shared.

Q: How to transfer framework from working directory to git?

Select Project → Right click → Team → Commit, Go to Git staging window, drag all the files from unstaged area to staged area → Give Commit message → Commit.

Q: Transferring Framework from local repo to Global:

Right click on project → Team → push branchmaster.

Q: How to get the framework from Global repo into local system?

Pre-Condition : Make sure you have git username and password.

1. Go to Eclipse, file → import
2. expand git → Project from git
3. Clone Url → URL → UN PWD → next - next - finish

Q: How to transfer newly created file from local repo to global repo?

Select New file → right click → Commit.
Go to git staging window, drag all the files from unstaged area to staged area.
→ Commit message → click on Commit and push.

Q: How to get new files available in github to local?

Select the project → Right click → team → pull

Q: when will you get Conflicts in github and how to resolve?

Whenever two engineers modifies the same file, when you try to Commit that file into github, you'll get Conflict.
In order to resolve this problem, pull the framework using rebase option.

BUILD TESTING TOOLS:

Ant - previously used
Maven - Currently using

Maven: It is basically a dev tool used to test the applications build and create a build.

Maven Usage in Automation:

Maven is a build testing tool, which will check the integration issue with the framework Component whenever multiple engineers integrate their work with the same framework.

Integration issues might get when engineers modify the Code, which might affect other java file, which is created by other engineers.

Maven identifies such kind of issues and provide a message called build failure / build Success

There are 2 plug-ins available in framework

1. Maven cmd line plugin
2. Maven Integration plugin.

⇒ Maven Cmd line plugin: It should be explicitly installed in the local system which is used to handle the maven project in the Commond line.

Installation steps:

→ Go to Google, search for download maven
→ Click on first link, navigate to maven Community.
→ Beside the binary, click on apache-maven-3.6.1-bin.zip
→ Download and extract the zip file.
→ Get inside the folder and copy the bin location
→ Go to environment variables window in local System.
→ Go to System variable division, find and edit path variable, enter new path as C:\apache-maven-3.6.1\bin and click on ok.

→ Go to user variable div, click on new, and then create new M2-HOME = C:\apache-maven-3.6.1.

→ Ensure JAVA_HOME is also available otherwise maven will not work.

→ Go to cmd, type mvn -version to check if maven is installed or not.

⇒ Maven - Integration Plugin:

It's a default plug-in, available in latest eclipse which is used to create maven project through eclipse.

Advantages:

1. Has dependency feature, where we can download all dependent jar directly from the internet.
2. It is a build testing tool used to check the integration issue and provide the status of the build.
3. Using maven, we can run the test scripts in cmd line without eclipse.
4. Maven provides standard framework folder structure.

Q: What is maven dependency and how to handle maven dependency?

Dependency is a feature available only in Maven which is used to download all the dependency jar into local repo (c:/user/.m2).

04 Sunday

Global Repository (Google → mvn repository):

In order to add dependency we should write dependency code for webdriver, apache poi, testNG etc., inside POM.xml.

POM.xml :

Project object model is a configuration file for the maven which is used to handle maven project in cmd line.

→ Dependency code include group id, articraft id and version.

Group-Id - Name of the Group / Community - from where you are downloading.

Artifact-Id - Name of the project in that group/ Community - which file your are downloading

Version - indicates version of the file which you want to download.

Maven Life Cycle - Maven Goals

Clean - This phase involves cleaning of the project (for a fresh build and deployment)

Compile - Compilation of the project source code

Install - Installing the application in the local repository. Any other project can use this as a dependancy.

Test - Executing / Running the tests using some suitable test framework.

Note : For compiling, we download Compiler plugin
For test, we need to download Mvn Surefire plugin.

Q: How to solve MojoFailureException when it says you are running on JRE rather than JDK?

Right click on JRE → Properties → Installed JRE's → Add → Next → Directory → select path of JDK → Finish → select JDK instead of JRE → click on apply and close → select workspace default JRE(JDK..) → click on apply and close.

JENKINS:

→ It is a Continuous integration and Continuous deployment tool.

→ In case of development, jenkins will be used to automate build creation and build testing and build deployment process. So, it is called Continuous deployment tool.

Jenkins Usage in Automation:

→ In Automation, jenkins will be used in Continuous integration process, it means Jenkins will always monitor the framework in Git location, automatically creates a build and provides email notification if any changes had happened in the build.

→ Jenkins can also be used to execute all the test scripts whenever new build arrives to testing environment.

Jenkins provides 3 levels of execution:

1. On-demand
2. On-Schedule
3. Poll-scm (Source Code Management)

On-demand: Based on the customer demand, login to Jenkins and start the execution.

On-Schedule: Based on the Schedule time, which is given in the Configure area, Jenkins automatically triggers build.

Poll-SCM: Poll means Continuously monitor the framework or application in git location, it automatically starts the execution whenever any changes happen in the framework or testing server.

Note: Whenever jenkins starts the job execution, it will Connect to git and download the entire framework (build) into the local system and test the build by taking help of maven and execute all the test Scripts and sends out an email.

Installation steps of Jenkins:

→ Go to Google, search download jenkins, click on first link and click on windows option.

→ Download the exe file and install.

→ After installation, in order to open jenkins, http://localhost:8080.

Jenkins plugins:

→ Before Creating a project in Jenkins make sure below plugins are installed in Jenkins

→ Login to Jenkins, go to manage Jenkins, click on manage plugin → click on "Available" tab, Download

a) Maven Integration Plugin b) Github Integration Plugin

Jenkins Environment Variables :

→ After the installation of the plugin, make sure environment variables are added for Java, maven and git.

→ Login to Jenkins → manage Jenkins → Global tool Configuration → JDK → Unselect Install automatically → give name as JAVA_HOME and location of JDK.

Git : → C:\Program Files\Git\bin\git.exe

→ Do the same as above for Git and maven

Git.exe path

MAVEN_HOME

→ Click on Save to save all the changes.

Q: How to create a project or Job in Jenkins ?

1. Login to Jenkins
2. Click on new item, select maven project and click on OK.

Q: How to Configure git project/ framework location in Jenkins project or Job ?

→ Login to Jenkins → Click on Project Name → Click on Configure link.

→ Find the click git radio button under 'Source Code Management' section

→ provide git Url (Take it from 'Code' of Github).

→ Click on add button, provide git username and password.

→ Select username, password from the Credential dropdown.

→ Scroll down, find build division.

→ Enter in root POM : Project Name(Path)/pom.xm)

→ Goals and options : test

→ Click on save and click on build now.

Q: How to Configure Scedule ?

1. Login to Jenkins
2. Go to Configure file of the job
3. Scroll down and Select build periodically check-box under Build Triggers section.
4. Provide a time with respect to (min hour date month day)

(31 09 04 07 06)

Here it indicates 4th July Saturday 9:31 hours.

→ Minutes (0-59)
→ Hours (0-23)
→ Date of Month (1-31)
→ Months (1-12)
→ DOW (0-7) where 0 and 7 are Sundays)

Q: How to Configure Poll SCM ?

1. Go to project Configure file.
2. Select Poll SCM checkbox and provide 5 star seperated by space (* * * * *)

'*' means any minute, any day, any hour etc.

Q: How to Configure email ?

1. Go to Jenkins project Configure file.
2. Select email notification checkbox at the end of the stage.
3. Provide recipients

Note: In order to Configure email notification, we should provide Company SMTP Credentials in global Configuration file.