27/3/2023
Mon

Selenium

Content

Chapter - 1

1. Introduction to Automation
   - what ?
   - when ?
   - Advantages & Disadvantages
   - Automation tools.
2. Introduction to selenium
   - what ?
   - Advantages & Disadvantages
   - Components of Selenium
   - Selenium Architecture
3. Basics
   - Launching empty browsers
   - Selenium Webdriver class diagram
4. Web Driver Methods
5. Window Management
6. Navigation Api's
7. Locators (xpath - syntax) — Interview
8. Synchronization — Interview (Impli W& WD.Wait)
9. Web Element Methods — Explicit/implicit
   (getters)

theory

chapter - 2

Handling WebElements

1. Auto Suggestions
2. Mouse Actions — Interview
   - Mouse Hover
   - Right click
   - Double click
   - Drag and drop
3. Dropdowns — Interview
4. Frames and Windows — Interview
5. Screenshot — Int
6. Scroll Bar — Inter
   - JavaScript Executor
7. pop ups
   - Alert
   - calendar
   - child Browsers
   - Notification ask 1 guy
   - File upload

practical

Selenium is the nick name of Selenium.

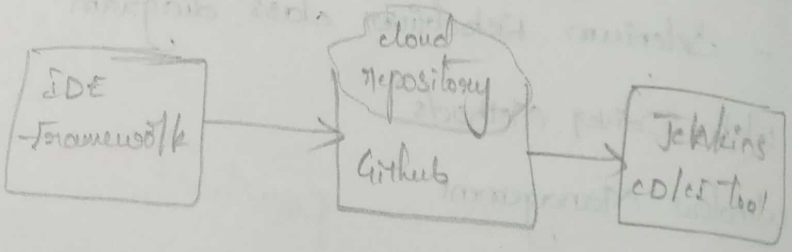
Chapter - 3

1. Data Driven Testing
   - What ?
   - Why ?
   - Advantages
   - properties file
   - Excel file
2. POM - page Object Model
   - What ?
   - Why ?
   - Advantages
   - steps to develop POM class
3. TestNG (Test Next Generation)
   - What ?
   - Installation
   - Annotations
   - Batch Execution
   - Group Execution
   - parallel Execution
   - Assertions
   - Reports
   (optional
      - Listeners
      - Dataproviders)

Chapter - 4

Framework

1. Maven
2. Framework
   - Design
   - Development
   - Execution
3. Github
4. Jenkins
5. Framework Architecture
6. Oops w.r.t. framework



Languages

programming language

- Develop an application

Ex:- Java, C, C++, C#

- Now-a-days we use it for testing as well.

Scripting Language

- Validating an application

Ex:- VB Scripting, python, Javascript.

- Now-a-days used in development as well.

Markup & other Language

- To Enable Communication b/w two applications

Ex:- HTML, XML, JSON

Applicati

Application

- Standalone
- client-Server
  - client-Server
  - web

Application:- It is a Set of programming Instructions which is used to perform Specific task.

1. Standalone :- An app which can be installed in local devices and can be accessed without internet is called standalone app.

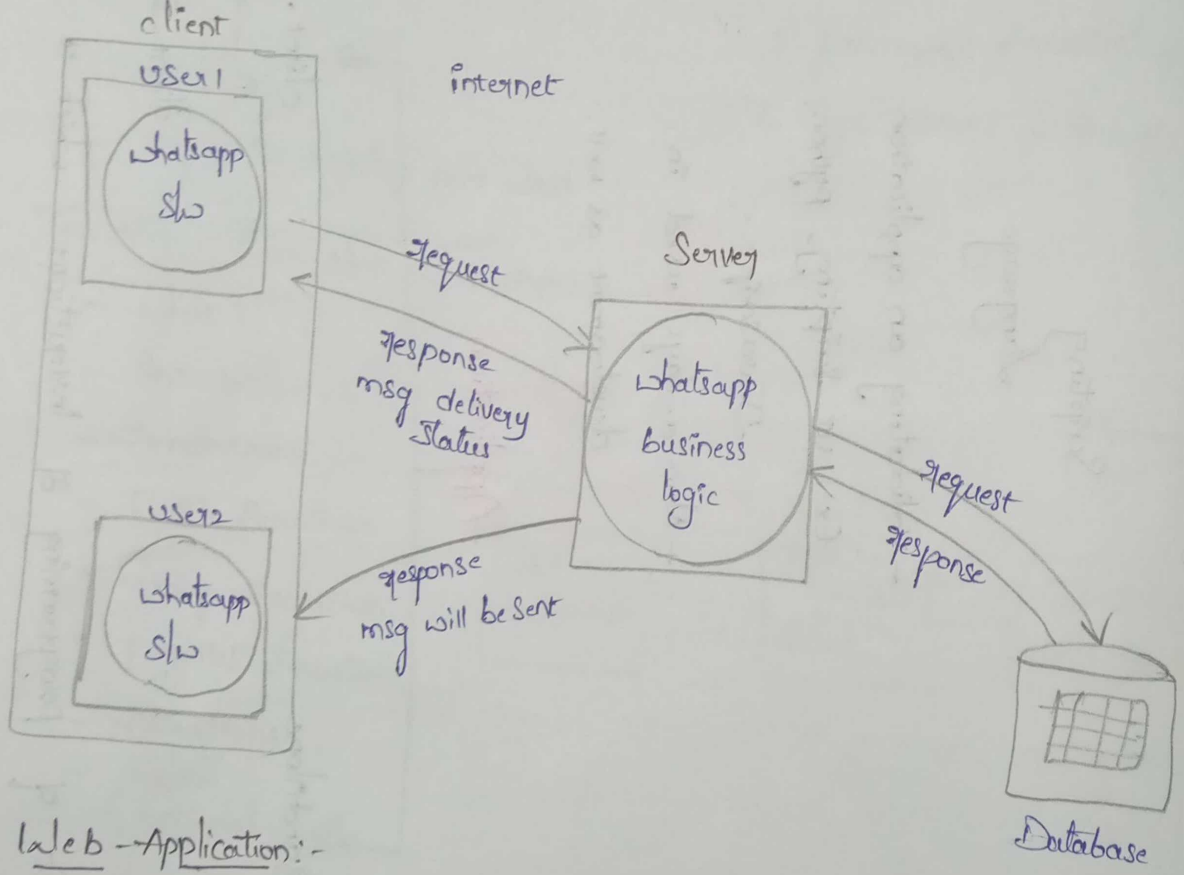
- Only single user can access it at a time.

Ex:- Calculator, Notepad, Ms office.

2. Client Server App :-

An app in which a part of Software is installed in local System and can be accessed via internet without browser.

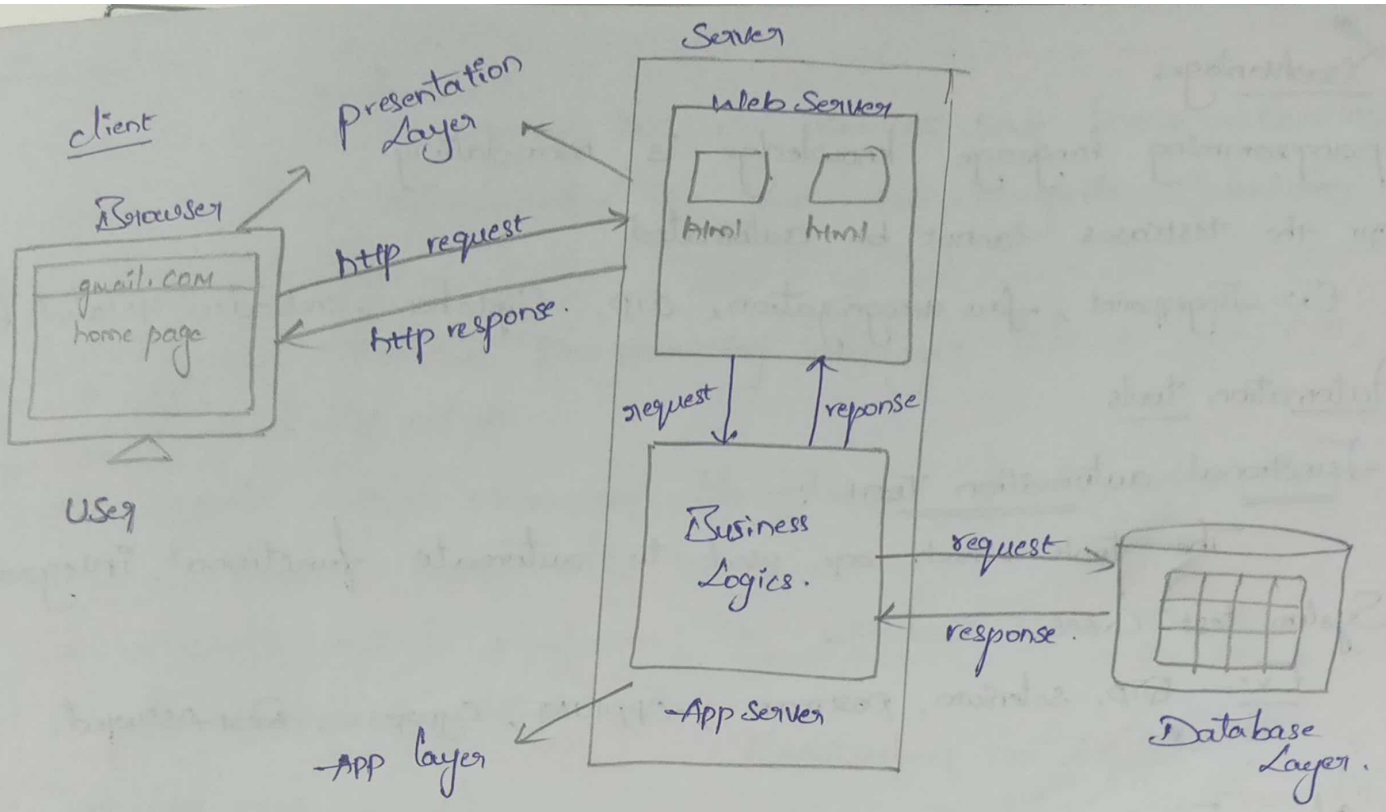
- Multiple users can be accessed at a time.



Web-Application :-

An app which can be accessed via Internet and browser using url is called as Webapplications.

- Multiple users can access the app at a time.
- Browser is a standalone app.



28/3/23 Tues

Introduction to Automation :-

Automation :- The process of Converting manual test cases into test Scripts using programming language of Scripting language and automation tool & executing testcases without any manual intervention.

When we Start Automation ?

- When the application is functionally stable.
- Resources are ready - test cases & automation tool.
- When we have more repetitive tasks.
- When we have more regression cycles.

Advantages

1. Time Saving
2. Accuracy in results.
3. Resources can be reusable or reusuability of resources.
4. Batch Execution.
5. Avoids repetitive tasks
6. Less resources
7. Generates reports automatically

Disadvantages

- programming language knowledge is mandatory.
- All the testcases cannot be automated.

Ex:- Fingerprint, face recognization, OTP, Captcha, embeded related

Automation tools

1. Functional automation Tool :-

The tools which are used to automate functional, integr System test Cases.

Ex:- QTP, selenium, postman, APPIUM, Cypress, Rest-Assured. (Quick Test professional)

2. Non Functional Automation Tool:-

The tools which are used to automate non-functional & p related test cases.

Ex:- Neoload, Loadrunner, J-meter, Appload.

Introduction to Selenium:-

Selenium:-

It is an open source web application test automation tool

Open Source - 1. Need not buy license (free of Cost).
2. Can view Source Code.
3. Can Customize it.

Web application:-

An application which is rendered over web.

Test automation:-

Automating test Cases

Tool:- Softwage.

Advantages

- Since it is an Open Source tool we neednot buy license - Cost effective.
- Since it is an open source tool we can Integrate Selenium with any thirdparty tools.
- It Supports multiple programming languages. like C, C++, C#, Java, python, JavaScript.
- It supports multiple browsers like chrome, firefox, edge, Opera, Safari etc. - Browser Compatibility testing is easy.
- It Supports multiple platforms like windows, Linux, IOS etc.- System Compability testing is easy.

Disadvantages :-

Same as Automation.

Selenium is the most popular tool

Components of Selenium :-

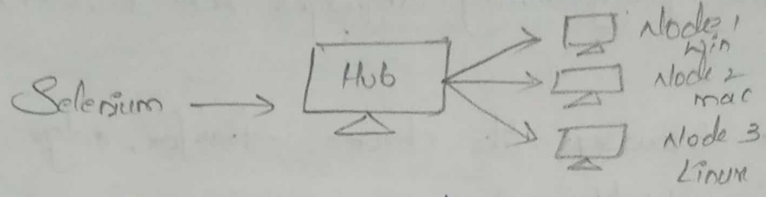
Enterprise

1. Selenium IDE (Integrated development Environment).
   - It is used for record and playback actions.
   - we automate quick bug reproductive tasks.
   - It is also used in automation aided exploratory system.
   - It is developed as plugged in for the browsers like firefox, Chrome & recently in edge.
2. Selenium WebDriver :-
   - It is the Successor of Selenium Rc (Remote Control).
   - It is used to build Robust frameworks.
   - It Supports multiple browsers.
   - It is used for test Suite executions.
3. Selenium Rc :-
   - It is also present in Selenium Community.
   - It is the deprecated version (Currently not in use).

- It Supports multiple browsers and programming languages.
- It does not automate https protocol.

4. Selenium Grid:-

It is the Collection of libraries Selenium Webdriver + Selenium RC + Selenium Grid Server.



Selenium Grid is used for remote execution. (other devices).

remote execution - Executing test scripts on other devices in the same network, Cloud Computing (virtual execution).

For mobile based applications:- 1. Selendriod → for Andriod based application.

2. Appium - For both IOS and Andriod based applications.

3. Winium - (For windows) It is used to automate Window based app.

Task:-

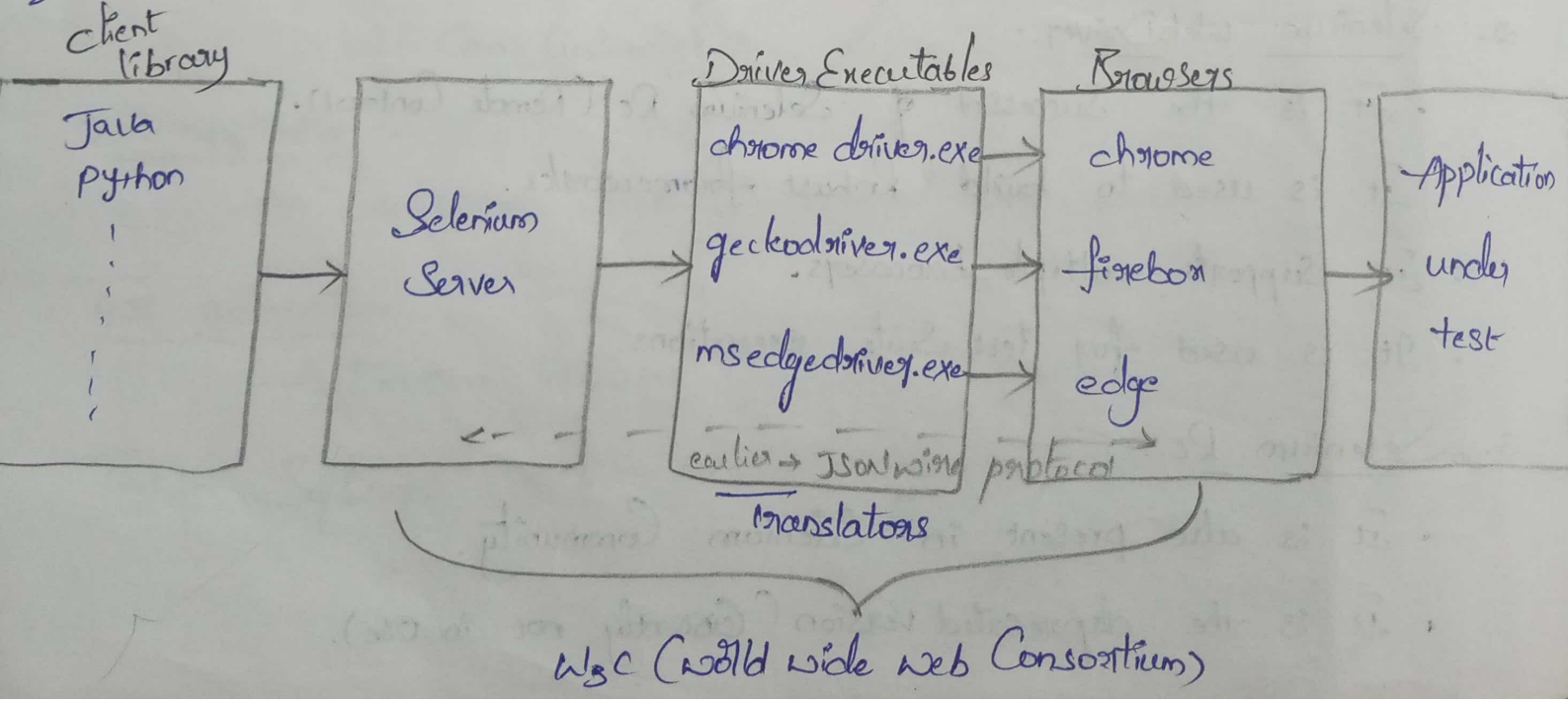
Which language Selenium Rc is not Supported?

Ans:- asp.

JSON - Java Script Object Notation

29/8/23 Wed

Selenium Architecture



- Selenium Supports multiple programming languages called as client library.
- When we write test scripts in one of the languages and trigger the execution. Selenium Server Communicates with drivey executables.
- Drivey executables act like translatorys between Selenium Server and bqowseys (in previous Selenium veysions this Communication was happening via Although JSON wire protocol).
- Bqowseys is the place wheye actual test execution takesplace.
- Selenium sevey, dqivey executables and bqowseys is togethey called as W3C procotol (World wide web Constorium).

Steps to download Selenium Server.

- Open the bqowsey & type Selenium.dev.
- click on downloads.
- Scroll the page till previous releases & click on releases link.
- click on '4.6.0' veysion.
- click on 'Selenium-Sevey-4.6.0.jar'. (jar - java aychieve files).

Steps to download Drivey Executables:-

Chyome

- check veysion
  - Open chyome bqowsey and click on ':' on top right Cornes.
  - Go to help.
  - click on 'About Google chyome
  - Note down the veysion (111.0.5563.146)
- Open the bqowsey and type Selenium.dev/downloads/
- Scroll the page till 'platforms Supported by Selenium'.
- click on '+ Bqowseys'.
- Undey chyome, click on 'documentation' link

- click on 'Downloads' link
- Under Current Releases, click on respective chrome driver Version link
- click on 'chrome driver - win32.zip' file.

Edge

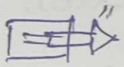
- check Version
  - Open edge browser and click on '...' on top right Corner.
  - Go to 'Help and feedback'.
  - click on 'About Microsoft Edge'.
  - Note down the version Completely. (Version - 111.0.1661.54)
- Open the browser and type selenium.dev/downloads/.
- Scroll the page till 'platforms Supported by Selenium'.
- click on '+ Browsers'.
- Under edge, click on 'documentation' link.
- click on 'Downloads' link.
- Click on repective version 'x64' link.

Firefox

- check version.
  - Open firefox browser and click on '...' on top right Corn
  - click on 'Help'
  - click on 'About firefox'.
  - Note down the version Completely
- Open the browser & type Selenium.dev/downloads/.
- Scroll the page till 'platforms Supported by selenium'.
- click on '+ Browsers'.
- Under firefox, click on 'documentation' link.
- Under 'Supported platforms' click on 'geckodriver releases' link.
- Scroll till 0.32.0, click on 'Assets'.
- click on 'geckodriver - v0-32.0-win64.zip.

30/3/23
Thurs

Steps to add Selenium and .exe files to Java project.

- Extract all driver executable zip files.
- Create java project in eclipse.
  - Open eclipse.
  - click on file → New → Java project.
  - Name the project.
  - De-select the checkbox 'create module-info.java'.
  - click on 'Finish'.
- Copy Selenium-Server-4.6.0.jar file from 'downloads' folder and paste it on java project in eclipse.
- Right click on Selenium jar file in project → Build path → Add to build path.
- Add Driver executables to the project.
  - Go to project folder location in file explorer.
    - Right click on java project in eclipse → properties.
    - click on the location icon "  "
    - Open project folder.
  - Copy driver executable .exe files from extracted folders and paste it into project folder.

Launch Empty Driver

1) Chrome

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

ChromeDriver → class in org.openqa.selenium.chrome

driver → ref Variable.

= → assignment operator

new → keyword to create object.

ChromeDriver() → Constructor of chromeDriver class.

; → Separator to end java statement.

Firefox

FirefoxDriver driver = new FirefoxDriver();

3) Edge

EdgeDriver driver = new EdgeDriver();

Tasks

1. Write a Script to launch multiple browsers.
2. Write a Script to launch user desired browser.

1) package locators;

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.edge.EdgeDriver;
public class MultipleBrowsers throws InterruptedException {
    WebDriver driver = new ChromeDriver();
    System.setProperty("webdriver.chrome.driver", "./chromedriver.exe");
    driver.get("http://google.com/");
    Thread.sleep(3000);

    System.setProperty("webdriver.edge.driver", "./msedgedriver.exe");
    WebDriver driver1 = new EdgeDriver();
    driver.get("http://instagram.com/");
    Thread.sleep(3000);
    driver1.close();
    driver.close();
  }
}
```

31/3/23
Fri

Launching Empty browser fixing problem Exception.

Chrome (import.

```
package basics;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.chrome.ChromeOptions;
public static void class EmptyChromeBrowserLaunch {
  public static void main (String [] args)
      ChromeOptions options = new ChromeOptions();
      options.addArguments ("--remote-allow-origins=*");
       ChromeDriver driver = new ChromeDriver(options);
  }
}
```

→ Steps to remove older version.

- Right click on your project.
  - Select Build path.
    - click on Configure Buildpath.
- click on Libraries.
  - select Selenium Server and click on remove button.
  - click on apply & close.
- Right click on selenium-server jar file in project and click on delete.

→ Steps to add Selenium 3 (New Server Version)

- Open the browser type maven repository.
- click on the first link.
- Now type Selenium-Server in Search bar, click on Search button.
- click on first link Selenium Server.
  - click on 3.141.59 version.
  - click on jar (62kB) file. & it will get download.
- Go to downloads Copy se Server 3.141.59

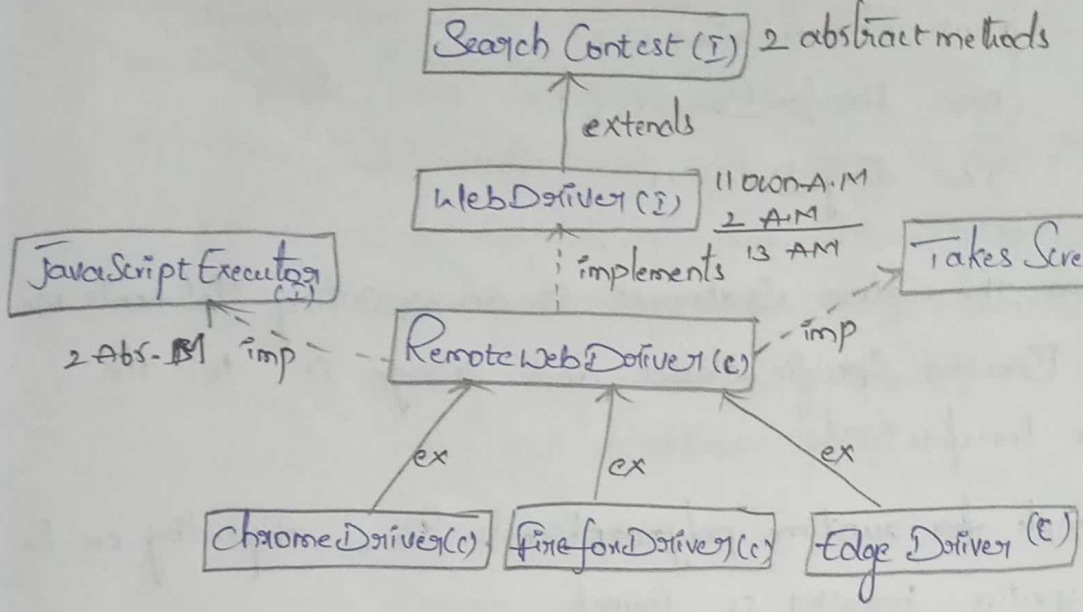
wrong

Edge

```
package basics;
import org.openqa.selenium.edge.EdgeDriver;
Public class EdgeDriverLaunch {
 public static void main (String [] args) {
System.setproperty ("webdriver.edge.driver", ".\msedgedriver.exe");
    EdgeDriver driver = new EdgeDriver ();
  }
}
```

5/4/2023
Monday

WebDriver class Diagram



1. SearchContext is the Supermost interface in WebDriver class Diagram.
   It has 2 abstract methods. 1) findElement()
   2) findElements()

2. WebDriver is the SubInterface of SearchContext Interface. It has 11 Own abstract methods and 2 are Inherited from SearchContext Interface. All together it has 13 abstract methods:-
   1. get()
   2. getTitle()
   3. getCurrentUrl()
   4. getPageSource()
   5. getWindowHandle()
   6. getWindowHandles()
   7. manage()
   8. Navigate()
   9. SwitchTo()
   10. close()
   11. quit()

3. RemoteWebDriver is the Concrete implementing class of WebDriver Interface. It also implements JavaScriptExecutor Interface and TakesScreenshot Interface.

*. JavaScriptExecutor interface has 2 abstract methods.
   1. executeScript()
   2. executeAsyncScript().

* TakesScreenshot Interface has One abstract methods.
   1. getScreenshotAs()

* All the browsers specific classes like ChromeDriver, FirefoxDriver and EdgeDriver extend to RemoteWebDriver class.

4/4/23 WebDriver Methods

WebDriver driver = new ChromeDriver();
new FirefoxDriver();
new EdgeDriver();

→ With respect to java the above statement is an upcasting statement. Create Instance for Browser specific classes and assign it to the Super type WebDriver Interface (reference Variable)

It is also an example for runtime polymorphism. Since depending on Instance created respective browser is launch.

→ w.r.to Selenium the above statement launches Empty Browser.

WebDriver Methods

Get(): It is used to navigate to an application and wait until the page is loaded. 2) return type is void.

- It accepts url like String type url as argument.

GetTitle():

- It is used to fetch the title of Current webpage
- return type is String.

GetCurrentUrl():

- It is used to fetch the url of Current webpage.
- return type is String.

GetPageSource():

- It is used to fetch the ~~(url of)~~ Source Code of the Current We
- return type is String.

closeMethod()

- It is used to close the tab of window.
- return type is void.

→ package Webdrivermethods;

import org.openqa.selenium.WebDriver;

public class GetTitleAndURL{

public static void main (String [] args){

WebDriver driver = new ChromeDriver();

driver.get ("https://www.amazon.com/");

String title = driver.getTitle();

String url = driver.getUrl();

S.o.pln (title);

S.o.pln (url);

}

}

o/p:- Webpage open

login page title

url print.

→ package webdrivermethods;

import org.openqa.selenium.WebDriver;

public class GetpageSourceMethod{

public static void main (String [] args){

WebDriver driver = new ChromeDriver();

driver.get ("https://www.google.com/");

S.o.pln (driver.getpageSource());

}

}

→ package Webdrivermethods;

import org.openqa.selenium.WebDriver;

/** Scenario
* Open the browser
* entry ebay.com
* fetch the title & Url
* close the browser
*/

public class CloseMethod {

public static void main (String [] args) {

WebDriver driver = new ChromeDriver();

driver.get ("https://www.ebay.com/");

System.out.println (driver.getTitle());

S.o.pln (driver.getUrl());

driver.close();

}

}

Window management

- We can manage the window dimensions using the webdriver method, manage().
- Manage() is used to do all the browser menu settings.
- Return type of manage method is 'Options' (Static Interface).

manage() → options (s.I).
↳ window()

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

minimize → driver.manage().window().minimize(); (Not present in Sel 3 present in sel 4

fullScreen → driver.manage().window().fullScreen();

To set window Size → Dimension d = new Dimension (height, width);

driver.manage().window().setSize(d);

① package practice ;

```
import org.openqa.Selenium.WebDriver ;
import org.openqa.Selenium.chrome.ChromeDriver ;
public class Practice1 {
    public static void main (String [] args) throws InterruptedException {
        WebDriver driver = new ChromeDriver();
        driver.get ("https://www.ebay.com/");
        driver.manage().window().maximize();
        Thread.sleep (4000);
        driver.manage().window().fullscreen();
        Thread.sleep (4000);
        driver.close();
    }
}
```

o/p:- webpage open
maximize
fullscreen
close

② package practice ;

```
import org.openqa.Selenium.Dimension;
import org.openqa.selenium.WebDriver;
import org.openqa.Selenium.chrome.chromeDriver;
public class SetWindowSize {
    public static void main (String [] args) throws InterruptedException {
        WebDriver driver = new ChromeDriver();
        driver.get ("https://www.ebay.com/");
        Dimension d = new Dimension (900, 720);
        driver.manage().window().SetSize(d);
        Thread.sleep (4000);
        driver.close();
    }
}
```

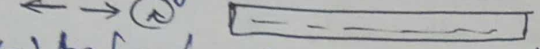
o/p:- webpage open
page open
Size Expands
close

By default Statement :- driver.manage().window().maximize();

5/4/23
Wed

Navigation Api's :-

- In older to perform navigations in the browser we have webdriver method navigate().
- return type of navigate method is "navigation."
- Navigation is a static interface which handles all browser navigations.



→ webbrowser.

(←) before/previous page → driver.navigate().back();

(→) next page → driver.navigate().forward();

refresh → driver.navigate().refresh(); — Reload the current page

navigate to application → driver.navigate().to(url);

→ package navigationapis

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

public class Navigation {
    public static void main(String[] args) throws InterruptedException{
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.google.com/");
        Thread.sleep(2000);
        driver.navigate().to("https://www.facebook.com/");
        Thread.sleep(2000);
        driver.navigate().back();
        Thread.sleep(2000);
        driver.navigate().forward();
        Thread.sleep(2000);
        driver.navigate().refresh();
        Thread.sleep(2000);
        driver.close();
    }
}
```

1 Differences b/w Get() & Navigate()

Get()	Navigate()
. It is used to navigate to an app & wait until the app is loaded.	. It is used to navigate to an application.
. It doesnot store browsing history.	. It stores browsing history.
. return type is void.	. return type is navigation.

*
```
package navigationapis;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
public class Navigation {
    public class static void (String[] args) throws InterruptedException {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.instagram.com/");
        Thread.sleep(2000);
        driver.navigate().to("https://www.twitter.com/");
        Thread.sleep(2000);
        driver.navigate().back();
        Thread.sleep(2000);
        driver.navigate().forward();
        Thread.sleep(2000);
        driver.navigate().refresh();
        Thread.sleep(2000);
        driver.navigate().to("https://www.flipkart.com/");
        Thread.sleep(2000);
        driver.navigate.back();
        Thread.sleep(2000);
        driver.navigate.forward();
        Thread.sleep(2000);
        driver.close();
    }
}
```

HTML

- HTML - Stands for "Hyper text Markup Language".
- It is used by web Developers & web designers to develop web
- It has 3 elements. 1. Tags
  2. Attributes
  3. Text

1. Tags : The first word which is enclosed within angular brackets <Html> is called Tags.

Ex:- <a> - links
<select> - dropdown
 - list
 - division
<table>
<h1> - heading
<input>
<button>
<textarea>

Tags in html are predefined.

2. Attributes :-

Anything which is enclosed in angular brackets as key value pairs.

Ex:- <input id = "username" type = "text" />.

key value pairs

3. Text :-

Anything which is not enclosed within angular brackets.

Ex:- <a> Skillbox .

text

Locators :-

- In order to locate an element in the webpage we use locators.
- Locators are the static methods of "By" class - where By is abstract class in Selenium.
- We have different locators.

1. id
2. Name
3. ClassName
4. TagName
5. LinkTest
6. partialLinkText.
7. cssSelector
8. xpath.

7/4/23
Tue

Id locator

If we have id attribute in the element node we can go for id locator.

Name locator :-

If we have name attribute in the element node we can go for name locator.

→ package locators ; (Script)

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
    /* Scenario
     * open the browser
     * enter facebook.com
     * enter username and password.
     * click on login
     * close the browser
    */
public class IdAndNameLocator {
  psvm (s[] a) throws InterruptedException {
   WebDriver driver = new ChromeDriver();
   driver.manage().window().maximize();
   driver.get("https://www.facebook.com/");
   driver.findElement(By.id("email")).sendKeys("abcdefgh");
   driver.findElement(By.id("pass")).sendKeys("abdefgh");
   driver.findElement(By.name("login")).click();
   Thread.sleep(4000);
   driver.close();
  }
}
```

findElement

- This method is used to fetch the first matching element from webpage.
- Return type is WebElement(I).
- it takes locators as arguments.
- if it does not fetch the element it throws NoSuchElementsException.

Package locators:

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
public class IdAndLogin2 {
    /* Scenario
     * open the browser
     * enter demo.actitime.com
     * enter username as 'admin' and password as 'manager'
     * validate the homepage
     * close the browser
     */
    public static void main(String[] args) throws InterruptedException {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://demo.actitime.com/login.do");
        driver.findElement(By.id("username")).sendKeys("admin");
        driver.findElement(By.name("pwd")).sendKeys("manager");
        driver.findElement(By.id("loginButton")).click();
        Thread.sleep(3000);
        if(driver.getTitle().contains("Enter Time-Track"))
            System.out.println("Testpass");
        else
            S.O.pln("Test-fail");
        driver.close();
    }
}
```

Tasks

1) Scenario

Open the browser

-Enter ebay.com

Type 'headphones' in Search text field.

click on Search button

close the browser.

```
package locators;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
public class IdAndNameLocators {
    psvm (S[] args) throws InterruptedException {
        WebDriver driver = new ChromeDriver(); (FirefoxDriver());
        driver.manage().window().maximize();
        driver.get("https://ebay.com/");
        driver.findElement(By.id("gh-ac")).sendKeys("headphones");
        Thread.sleep(3000);
        driver.findElement(By.id("gh-btn")).click();
        Thread.sleep(3000);
        driver.~~findElement~~ close();
    }
}
```

2) Scenario.

open the browser

-Enter 'google'.com

Type your name in Search text field.

click on Search button

close the browser.

```
package locatorss ;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
public class IdAndNameLocators {
    psvm (s[] args) throws InterruptedException {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://google.com/");
        driver.findElement(By.name("q")).SendKeys("Vanalakshmi");
        Thread.sleep(3000);
        driver.findElement(By.name("btnk").click();
        Thread.sleep(3000);
        driver.close();
    }
}
```

8/4/23 Sat

Link Text Locator :-

It is used to identify an element uniquely using text on the link.

partial Link Text :-

It is used to identify an element uniquely using partial Text on the link.

className Locator :-

It is used to identify an element uniquely using class attribute va

```
/* Scenario
* Open the browser
* Enter facebook.com
* click on 'forgotten password ?" Link
* enter email in text field.
* click on 'Search' button
* close the browser
*/
```

LinkText Locator:-

```
package locators;

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

public class LinkTextLocator {

    psvm (S[] args) throws InterruptedException{
    ChromeOptions options = new ChromeOptions();
    options.addArguments("--remote-allow-origins=*");
     WebDriver driver = new ChromeDriver();
      driver.manage().window().maximize();
      driver.get("https://www.facebook.com/");
      driver.findElement(By.LinkText("Forgotten password?")).click();
      driver.findElement(By.name("email")).sendKeys("vaga");
      driver.findElement(By.id("did_submit")).click();
      Thread.sleep(2000);
      driver.close();

    }
}
```

2) partialLinkText :-

```
package locators;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
public class PartialLinkTextLocator {

    psvm (S[]args) throws InterruptedException {
    WebDriver driver = new ChromeDriver();
      driver.manage().window().maximize();
      driver.get("https://demo.actitime.com/");
```

```
        Thread.sleep (2000);
        driver.findElement (By.partialLinkText ("Forgot")).click();
        Thread.sleep (2000);
        driver.close();
    }
}
```

3) ClassName :-

```
package locators;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
public class classNameLocator {
    psvm(S[] args) throws InterruptedException {
    WebDriver driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.get("https://snapdeal.com/");
    driver.findElement (By.className ("searchformInput")).sendKeys("Sno
                                                          .waitfo
    Thread.sleep (2000);
    driver.findElement (By.className ("searchTextSpan")).click();
    driver.close();
    }
}
```

Using className we can also fetch multiple elements from the we

10/4/23 Mon

findElements() :-

. It is an abstract method of SearchContext(I). It is used to fetch the list of matching elements from the webpage.

. return type of is List<WebElement>.

- If the elements are not found it returns empty arraylist.

Differences b/w findElement() & findElements():

- It fetches 1st matching WebElement.
- return type is WebElement.
- If the element is not found it throws NoSuchElementException.

- It fetches the list of all matching WebElements.
- return type is List<WebElement>.
- If the elements are not found it writtens Empty arraylist.

TagName Locator :-

- It is used to identify an element or list of element using the tag of the element node

Css Selector :- (Cascading style sheet)

- It is one of the locators which is used to identify an element using unique attributes.
- It is unidirectional (forward but not supported backward traversi
- It is faster compared to any locators.
- It doesn't support text.

Syntax :-

1. id → tagname # id-attribute-value
2. class → tagname . class-attribute-value
3. Generic → tagname [attributeName = 'attributeValue']

4) ClassName (using findElements)

```
package locators;
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;
/* Scenario.
 * Open the browser
 * enter flipkart.com
 * type iphones & click on search
 * fetch the list of iphones from first page
 * close the browser.
 */
public class ClassNameLocator1 {
    psvm (s[] args) throws InterruptedException {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.flipkart.com/");
        Thread.sleep(2000);
        driver.findElement(By.className("_2doB4z")).click();
        driver.findElement(By.name("q")).sendKeys("iphones");
        driver.findElement(By.className("L0Z3Pu")).click();
        Thread.sleep(2000);
    List<WebElement> iphoneList = driver.findElements(By.className("_4
for (WebElement iphone : iphoneList) {
    String name = iphone.getText();
        S.O.pln(name);

    }
        Thread.sleep(2000);
        driver.close();
}
```

5) TagNameLocator

```
package locators;
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;
    /*  Scenario
     * open the browser
     * fetch all the links present in the webpage
     * close the browser
     */
public class TagNameLocator {
    psvm (SC] args) throws InterruptedException{
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get ("https://www.facebook.com/");
        Thread.sleep (2000);
        List<WebElement> links = driver.findElements (By.TagName ("a"
for( WebElement link : links) {
        String textOnLink = link.getText();
            S.o.pln (textOnLink);
    }
      driver.close();
}
}
```

6. CssSelector Locator :- (using generic, id tag)

```
package locators;
    import org.openqa.selenium.By;
    import org.openqa.selenium.WebDriver;
    import org.openqa.selenium.chrome.ChromeDriver;
    /* Scenario:
    * open the browser
    * enter demo.actitime.com
    * enter username as 'admin' & password as 'manager'
    * click on login button
    * Verify homepage
    * close the browser
    */
public class CssSelectorLocator {
    psvm(S [] args) throws InterruptedException {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("www.demo.actitime/login.do");
        Thread.sleep(2000);
    driver.findElement(By.className("input#username")).
                sendKeys("admin");
    driver.findElement(By.className("input Cname='pwd'J")).
                sendKeys("manager");
        Thread.sleep(2000);
    driver.findElement(By.cssSelector("a#loginButton")).click;
    Thread.sleep(2000);
    if(driver.getTitle().Contains("Enter Time-Track"))
        S.o.pln("Testpass");
    else
        S.o.pln("Test fail");
    driver.close();
}
}
```

11/4/22
Tues

CssSelector Locator (using class tag)

```
package locators;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
public class CssSelector2 {
    psvm(S[] args) throws InterruptedException {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.Snapdeal.com/");
    driver.findElement(By.cssSelector("input.Search-form-input")).
                                sendKeys("Smartwatches");
    driver.findElement(By.CssSelector("span.SearchTextSpan")).click();
    Thread.sleep(2000);
    if(driver.getCurrentUrl().Contains("watches"))
        S.o.p("Test pass");
    else
        S.o.p("Test fail");
    driver.close();
}
}
```

X-path

- Absolute X-path
- Relative X-path
  - Basic relative X-path
    - X-path by attributes
    - X-path by text()
    - X-path by Contains()
      - Using attributes
      - Using text()
  - Advanced relative X-path
    - X-path by group finder
    - Xpath by traversing
      - Independent & dependent path
        - forward traversing
        - backward traversing
      - X-path by axes
        - parent
        - child
        - ancestor
        - descendant
        - Sibling functions
          - preceding sibling
          - following sibling

X-path keywords - and, or

X-path functions

1. text()
2. Contains()
3. last()
4. normalize-space()
5. starts-with()
6. Name()

X-path

It is one of the locators in Selenium which is used to find an element using attributes of text by traversing in HTML tree-structure.

We have 2 types of X-path. — 1. Absolute X-path
2. Relative X-path

Absolute X-path :-

The Complete path from the root of the HTML tree structure to till the element node is called as "Absolute Xpath".

. Here we use '/' for traversing.

```
<html>
    <head> </head>
    <body>
        <div>
            <input type = "text" id = "abc" /> → /html/body/div[1]/input[1]
            <input type = "text" id = "bcd" /> → /html/body/div[1]/input[2]
        </div>
        <div>
            <input type = "text" id = "cde" /> → /html/body/div[2]/input[1]
            <input type = "text" id = "def" /> → /html/body/div[2]/input[2]
        </div>
    </body>
</html>
```

Drawbacks of Absolute X-path

1. X-path can be lengthy.
2. It is time-Consuming.
3. Even if we miss one node in the element is not located.
4. In AGILE Methology due to frequent requirement changes GUI changes. which effects absolute X-path.

Relativexpath

- X-path written directly to an element using '//' is called relative X-path.
- We can also traverse from any point in html tree till the element node.

```
<html>
  <head> </head>
  <body>
    <div>
      <input type ="text" id = "A" />
      <input type = "text" id = 'B" />
    </div>
    <div>
      <input type = "text" id = "C"/> → //input[3] or //div[2]
      <input type = "text" id = "D"/> → //input[4] or //div[2]/in
    </div>
  </body>
</html>
```

Basic relative X-path

1. X-path by attributes :-

X-path written directly to an element using attributes of the element

Here we can use single / double quotes

Syntax :-

```
//tagname[@attributeName = 'attributeValue']
```

→

```
package locators;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
public class XpathLocator {
    ps vm (s[] args) throws InterruptedException {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://demo.actitime.com/login.do");
        driver.findElement(By.xpath("//input[@placeholder=\"username\"]")).
                sendKeys("admin");
        driver.findElement(By.xpath("//input[@placeholder=\"password\"]")).
                sendKeys("manager");
        driver.findElement(By.path("//a[@id=\"loginButton\"]")).click();
        Thread.sleep(3000);
        driver.close();
        if (driver.getTitle().Contains("Enter Time-Track")
            S.o.pln("Test pass");
        else
            S.o.pln("Test fail");
        driver.close();
    }
}
```

2. X-path by text():

. X-path written directly to an element using text on the element

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

→
```
package locators ;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
/* Scenario :
 * Open the browser
 * Enter demo.actitime.com
 * click on forgot password
 * click on return to login page
 * close the browser
 */
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
public class XpathByText{throws InterruptedException}
    public static void main(String[] args) throws InterruptedException{
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://demo.actitime.com/login.do");
        driver.findElement(By.xpath("//a[text()='Forgot your password']")).click();
        driver.findElement(By.xpath("//span/div[text()='Request Login info']")).
                                                                    click();
        Thread.sleep(3000);
        driver.close();
    }
}
```

13/4/23
Thurs

3) X-path by Contains()

X-path written directly to an element using attributes & text.

Syntax :- i) using attributes

//tagname [contains(@attributeName, 'attributeValue')]

ii) Using text()

//tagname [contains (text(), 'textvalue')]

Advantages

1. Handles lengthy text and attribute values.
2. Handles spaces. Including unbreakable spaces.
3. Handles partially changing elements.

For 1st advantage :-

Contains (attributeName)

```
① package locators;
  import org.openqa.selenium.By;
  import org.openqa.selenium.WebDriver;
  import org.openqa.selenium.chrome.ChromeDriver;
  public class XpathByContains1 {
      psvm(S[] args) throws InterruptedException {
          WebDriver driver = new ChromeDriver();
          driver.manage().window().maximize();
          driver.get("https://www.amazon.com/");
          driver.findElement(By.Xpath("//img[contains(@alt,'Fitness')]")).click
          if (driver.getCurrentUrl().Contains("fitness"))
              S.o.pln("Test pass");
          else
              s.o.pln("Test fail");
          Thread.sleep(3000);
          driver.close();
      }
  }
```

Contains(text())

```
package locators;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
public class XpathByContains2 {
psvm(S[]args) {
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://www.facebook.com/");
String getext = driver.findElement(By.xpath("//h2[contains(text(),'Facebook')]")
                                   .getText();
S.O.p(text);
driver.close();
}
}
```

14/4/23 Friday

X-path Keywords

And :-

Whenever we try to fetch an element using different Combinations of attributes, text() and Contains(), Such that all the Conditions should satisfy then we use "and" keyword.

OR :-

To fetch multiple elements using different Combinations of attributes, text(), Contains(), Such that any one of the Condition satisfies then we go for 'OR' keyword.

//tagname [@AN1 = 'AV1' and @AN2 = 'AN2']
or

//tagname [@AN = 'AV' and text() = 'Tv']
or

//tagname [@AN = 'AV' and Contains(Attributes(@), Text())]
or

//tagname [@ text() and text()]
or

// tagname [text() and/or Contains()]

// tagname [Contains() and/or Contains()].

Advanced relative Xpath

Xpath by group of Index:-

Whenever we have multiple matching elements use write Xpath to locate an element uniquely using the position value of the element.

Syntax :-

(Basic - Relative - x-path - expression) [positive - value].

Xpath by Traversing:-

Traversing:- moving from a node to another node in HTML.

Steps to locate dynamic element :-

1. Identify static element & write Xpath for it.
2. Identify Common parent & traverse to it.
3. Write attributes of tagname of dynamic element.

Static Element :-

The element which is fixed & doesn't change.

Dynamic Element :-

The element which changes frequently.

Independent & dependent Xpath :-

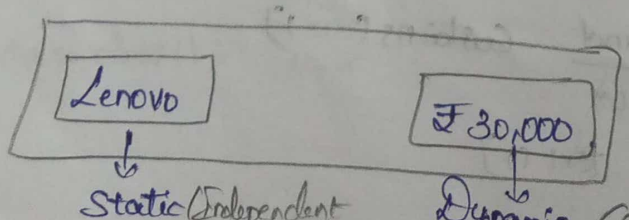
Independent Xpath :-

Xpath written to a Static element.

Dependent Xpath :-

Xpath written to a dynamic element.

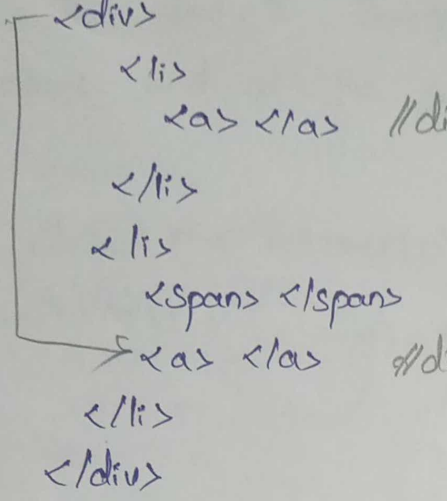
Ex:-



Forward Traversing:-

Traversing from a node to its immediate child using '/'.

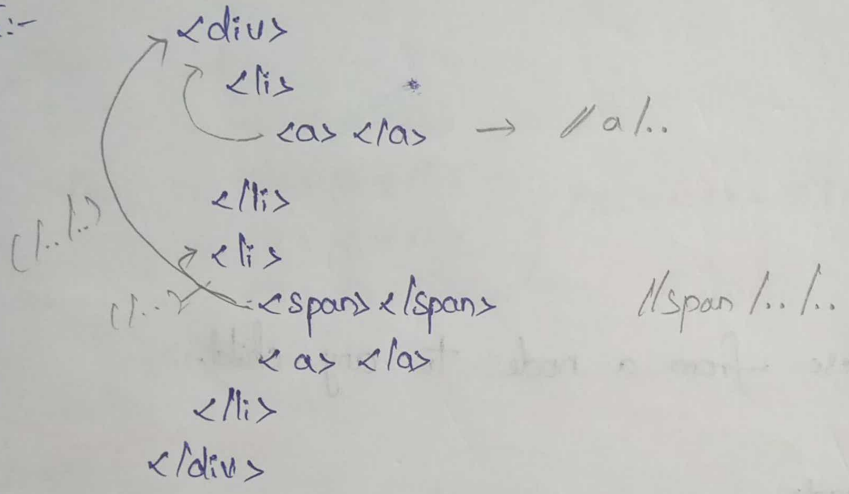
Ex:-



Backward Traversing:-

Traversing from a node to its immediate parent using '/..'.

Ex:-



15/4/23 Satur

Xpath by Axes:-

Syntax:- /axes-name :: tagname

Parent axes:-

It is used to traverse from a node to its Immediate parent. (it is Similar to backward traversing).

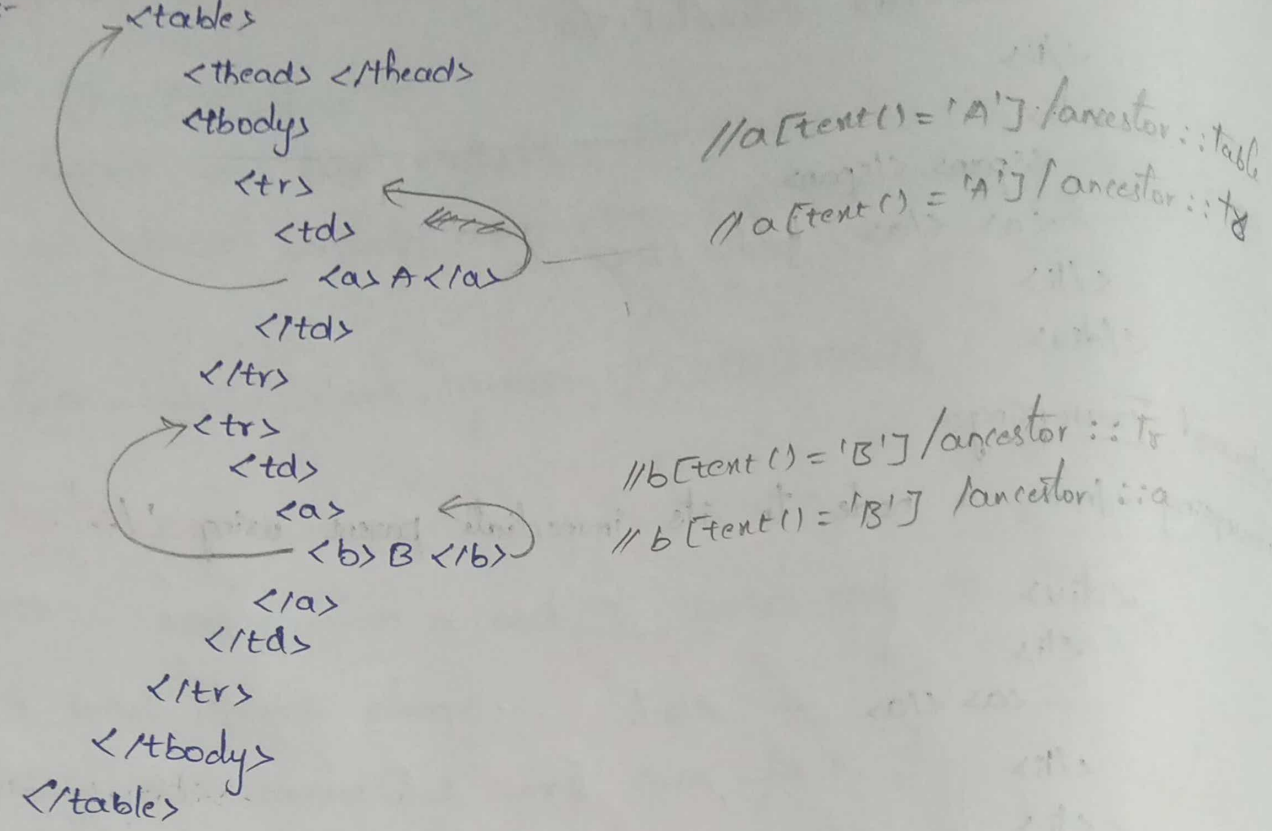
Child axes:-

It is used to traverse from a node to its Immediate child. (it is Similar to forward traversing).

Ancestor axes

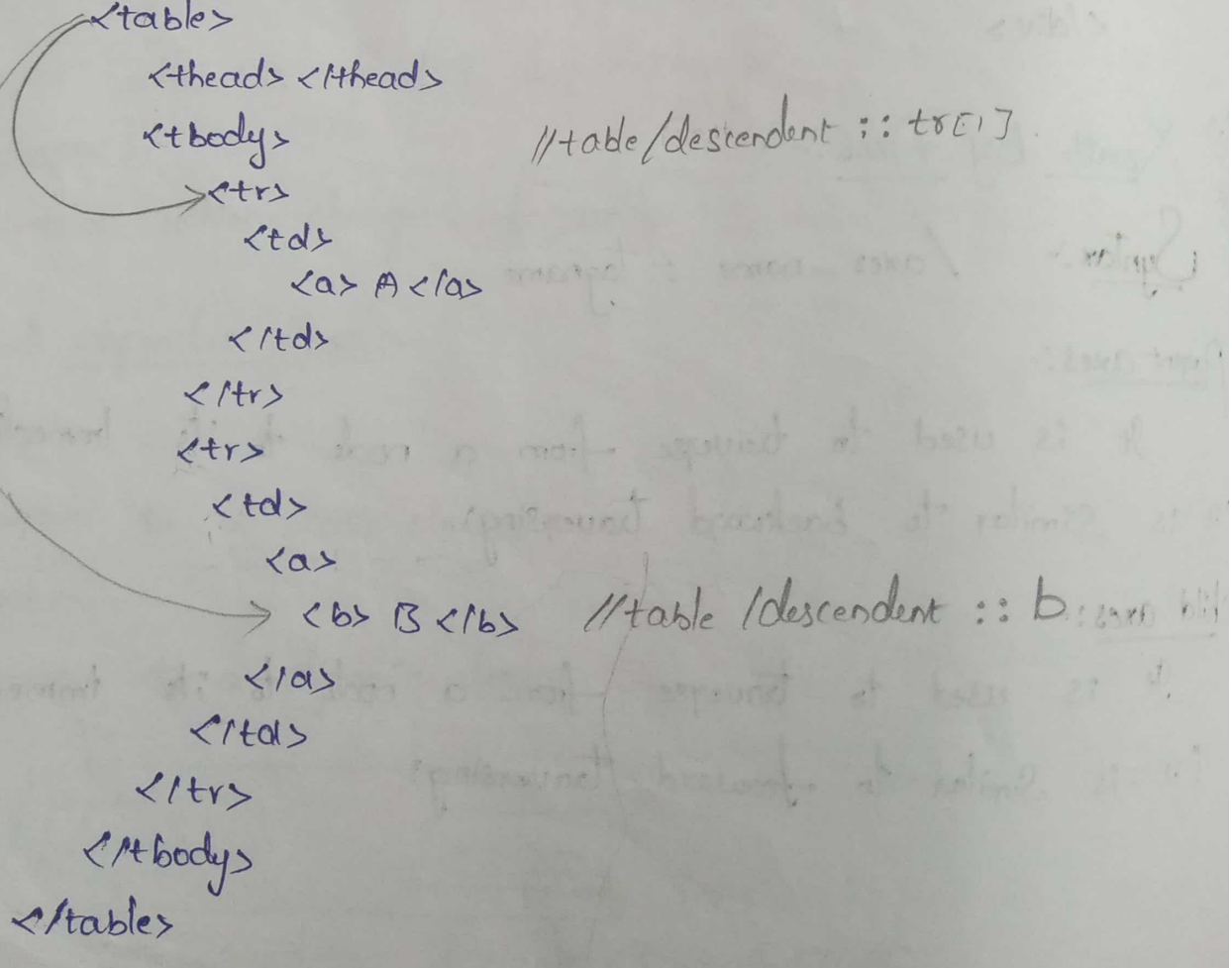
Traversing from a node to any parent can be done using ancestor axes.

Ex:-



Decendant axes:-

It is used to traverse from a node to any child.



17/4/23 Monday

Sibling - functions :-

① preceding - Sibling :-

It is used to traverse from a node to the nodes which are lying above and at the same level having Common parent.

Ex:-
```
<div>
  <li> A </li>
  <li> B </li>
  <li> C </li>
  <li> D </li>
</div>
```

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

//li[text() = 'D']/preceding-sibling

li[text() = 'A']

② following - Sibling :-

It is used to traverse from a node to the nodes lying below at the same level having Common parent.

Ex:-
```
<div>
  <li> A </li>
  <li> B </li>
  <li> C </li>
  <li> D </li>
</div>
```

//li[text() = 'B']/following-sibling::

li[text() = 'D']

Preceding axes

It is used to traverse from a node to the nodes lying above at the same level having uncommon parents.

Ex:-
```
<table>
  <tbody>
    <tr>
      <a>
        <b> ABC </b>
      </a>
    </tr>
    <tr>
      <a>
        <b> BCD </b>
      </a>
    </tr>
  </tbody>
</table>
```

//b[text() = 'BCD']/preceding::b.

following axes:-

It is used to traverse from a node to the nodes lying at the same level having uncommon parent.

Ex:-

```
<table>
  <tbody>
    <tr>
      <a>
        <b>ABC</b>
      </a>
    </tr>
    <tr>
      <a>
        <b>BCD</b>
      </a>
    </tr>
  </tbody>
</table>
```

//b[text() = 'ABC']/following::b

Xpath Functions:-

Name():-

It is used to handle unidentifable tags such as Svg.

Syntax:- //*[name() = 'svg']_attributes

18/4/23 tues

last():-

It is used to fetch the last element in the list of similar elements

Syntax:- tagname[last()]

```
<table>
  <tbody>
    <tr></tr>
    <tr></tr>
    <tr></tr>
    <tr></tr>
  </tbody>
</table>
```

→ //tr[last()]

Ex:- icc-cricket.com — //table/tbody/tr[last()].
bollymoviereviewz.com — //table[last()]/tbody/tr[last()].

normalize-space ():-

This function is use to remove all unnessecary spaces in the attribute values or text values.

Ex:- Syntax :- //tagname [normalize-space (text ()) = 'TV']

//tagname [normalize-space (@AN) = 'AV']

Starts-with ():-

It is used to fetch the elements which starts with specified attribute value or text value.

Syntax :- //tagname [starts-with (text (), 'start-value-of text')]

//tagname [starts-with (@AN, 'start-value-of-attribute')].

Synchronization :-

The process of matching Selenium Speed with application Speed is called Synchorization.

Why Synchorization?

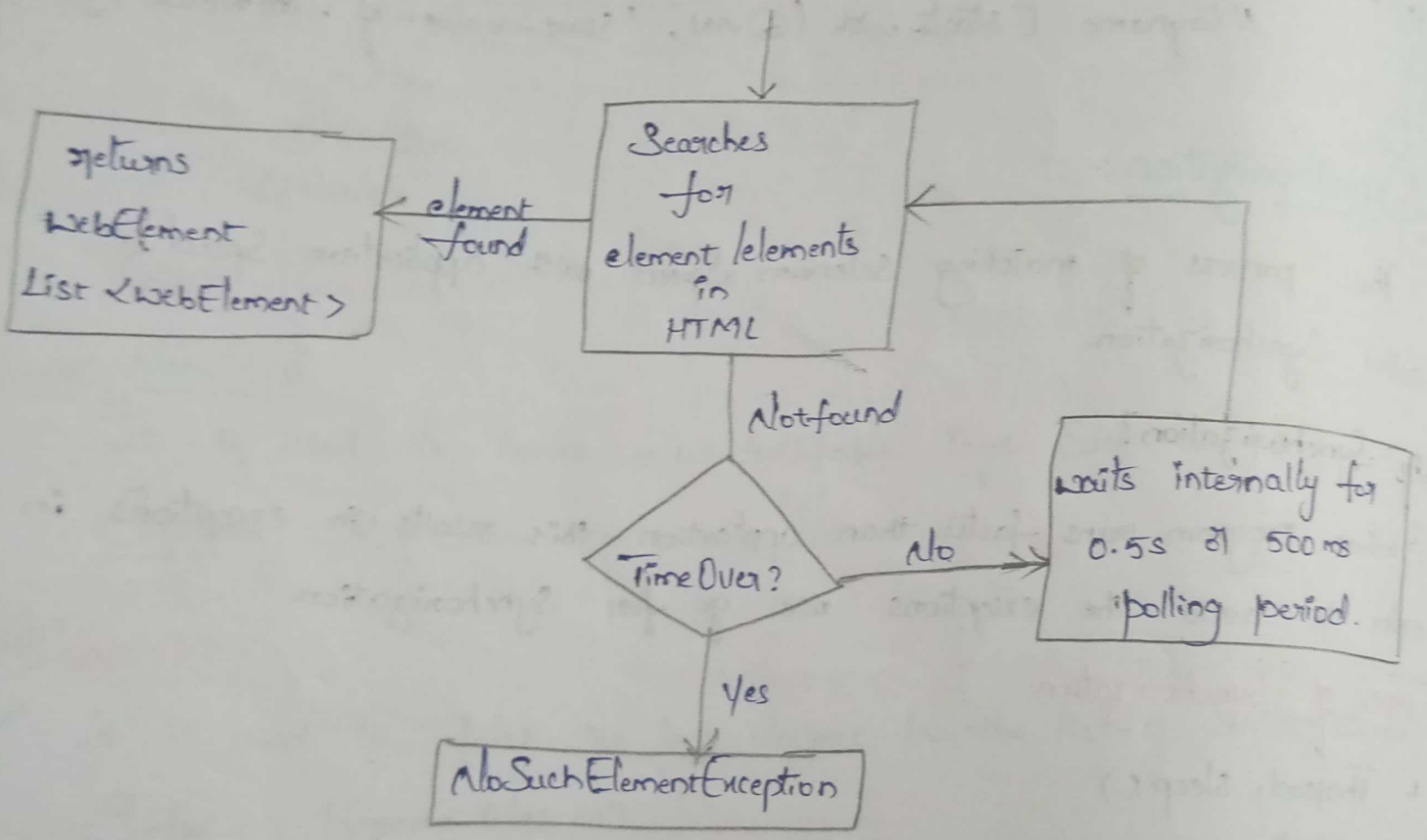
Selenium program runs faster than application this results in exceptions in order to avoid the exceptions we go for Synchorization.

Types of Synchorization

1. Thread.sleep()
2. implicitly wait.
3. Explicit wait
    1. WebDriver wait
    2. fluent Wait.
4. Custom wait

1. Thread.Sleep()
- It is java wait statement which waits for specified amount of time
- It takes time in milliseconds (long datatype as argument)
- It throws InterruptedException.

2. Implicitly wait
- It is Selenium wait statement which Synchronizes findElement & findElements methods only.
- It is global wait statement which is given only once in the beginning of the test script, it Synchronizes findElement & findElements methods present throughout the script.



Syntax:-

Version 3 → driver.manage().timeouts().implicitlyWait(time_in_sec, TimeUnit.SECONDS);

Version 4 → driver.manage().timeouts(), implicitlyWait(Duration.ofSeconds(time_in_sec));

. Whenever implicitlywait is given it Searches for the element/ elements in HTML.

. If the elements is found it writtens returns WebElement or List<WebElement>.

. If the element is Not found it checks if the Specified time is Over or not, if the time is Over it throws NoSuchElementException.

. If the time is not Over it waits Internally for 0.5s or 500ms this waiting period is called as polling period.

. Once the polling period is done it Searches for the element agai

. This process repeates until the element is found (or) time is Over

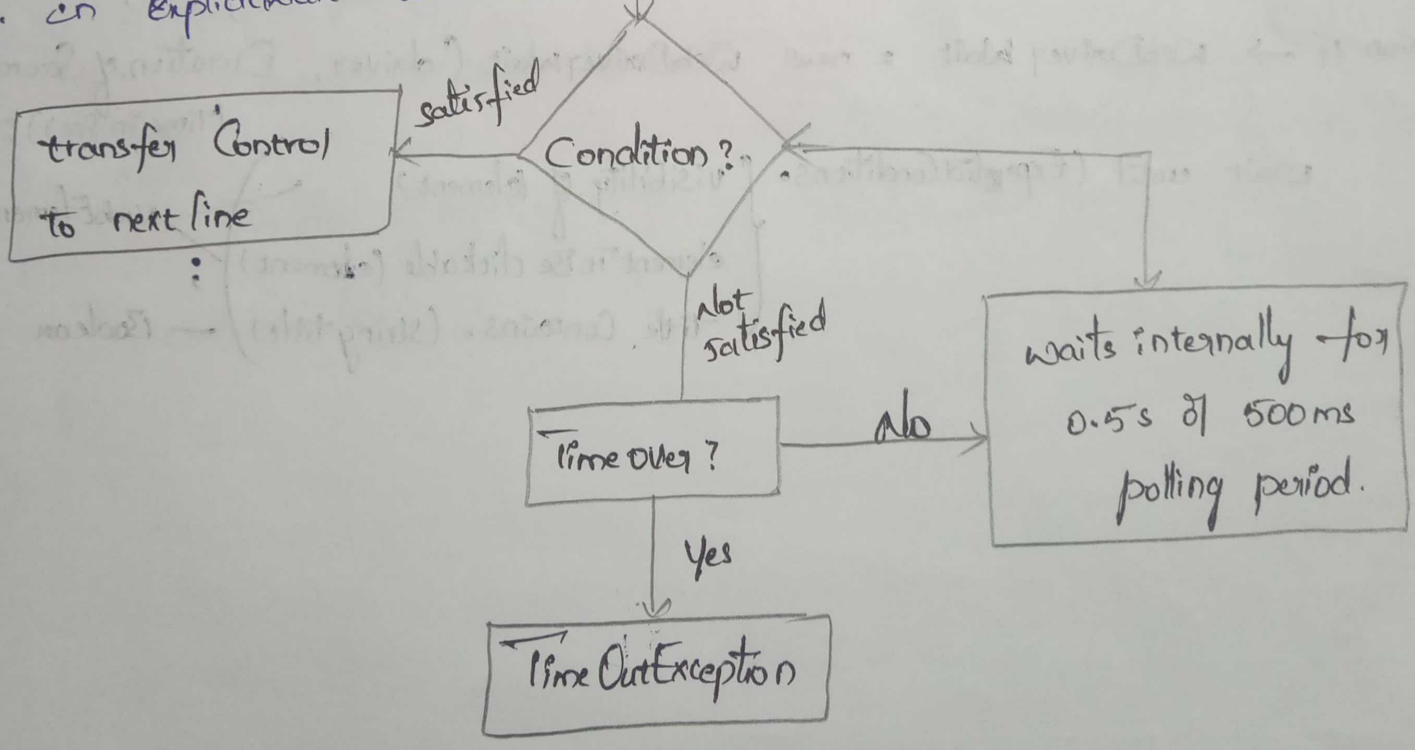
14/2/23

3. Explicit Wait

. It is Selenium wait Statement which Synchorinizes all the methods including findElement & findElements methods.

. Unlike implicit wait, explicit wait should be mentioned everytim when Synchorinization is needed.

. In ExplicitWait we should mention the Condition.



- Whenever explicitWait is given it checks for Condition, if the Condition is satisfied it traverse the control to the next line.
- If the Condition is not satisfied it checks if the specified time is Over or not.
- If the time is Over it throws TimeOutException.
- If the time is not Over it waits Internally for 0.5s or 500ms this waiting time period is called as polling period.
- Once the polling period is done it checks for the Condition again.
- These process repeats until the Condition is satisfied or time is Over.

Syntax:-

WebDriverWait :-

Version 3 → WebDriverWait wait = new WebDriverWait (driver, timeinSec);

wait.until(ExpectedConditions.{ visibility of (element) / elementToBeClickable (element) / titleContains (String title) }; → WebElement / Boolean)

Version 4 → WebDriverWait wait = new WebDriverWait (driver, Duration.ofSeconds (Time in Sec));

wait.until(ExpectedConditions.{ visibility of (element) / elementToBeClickable (element) / titleContains (String title) }; → WebElement / Boolean)

Implicit Wait :-

```
package Synchronization;
import java.util.Concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class ImplicitWait {
    psvm (s[] args) {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://demo.actitime.com/login.do");
        driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
        driver.findElement(By.id("username")).sendKeys("admin");
        driver.findElement(By.name("pwd")).sendKeys("manager");
        driver.findElement(By.id("loginButton")).click();
        WebElement header = driver.findElement(By.xpath("//div[@class=
                                \"pagetitle\"]/descendant::td[1]"));

        if(header.getText().Contains("Enter Time-Track"))
            S.o.pln("login Success");

        else
            S.o.pln("login fail");

        driver.close();

    }

}
```

ExplicitWait

WebDriverWait Case visiblity of Element))..

```
package Synchronization;
import java.util.concurrent.TimeUnit;
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.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;

    public class ExplicitWait1 {
     psvm (S[] args) {
    WebDriver driver = new ChromeDriver();
    WebDriver.manage().window().maximize();
    driver.get("https://the-internet.herokuapp.com/dynamic_loading/1");
    driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
    driver.findElement(By.xpath("//button[text()="
                                        "Hello World!"]));

    WebDriverWait wait = new WebDriverWait(driver, 10);
    WebElement element = wait.until(ExpectedConditions.visibilityOf(header));
        S.o.p(element.getText());
        driver.close();

    }

}
```

2)

```
package Synchronization;
import java.util.concurrent.TimeUnit;
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.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
```

```
public class ExplicitWait2 {
  ps vm (S [] args) {
    WebDriver driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.get("https://www.instagram.com/");
    driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
    WebElement loginButton = driver.findElement(By.Xpath(
                                "//button[.='Log in']"));
    WebDriverWait wait = new WebDriverWait(driver, 5);
    try
    {
      wait.until(ExpectedConditions.elementToBeClickable(loginButton)).click();
    }
    catch (Exception e) {
       S.o.pln("Disabled");
    }
    driver.close();
  }
}
```

3)

```
package Synchorinization;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
public class WebDriverWait {
  psvm (S[] args) {
  WebDriver driver = new ChromeDriver();
  driver.manage().window().maximize();
  driver.get("https://demo.actitime.com/");
```

```
driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
driver.findElement(By.id("username")).sendKeys("admin");
driver.findElement(By.name("pwd")).sendKeys("manager");
driver.findElement(By.id("loginButton")).click();
WebDriverWait wait = new WebDriverWait(driver, 10);
boolean status = wait.until(ExpectedConditions.titleConditions
                                ("Enter Time-Track"));
if(status)
    s.o.pln("home page displayed");
else
    s.o.pln("home page not displayed");
driver.close();
}
}
```

(Function method - it has 1 abstract method
Markup " - it doesn't have any abstract method

20/4/23 Thurs

2) Fluent Wait

- it is a Selenium wait statement and its a part of ExplicitWait.
- it is used to customize polling period.
- it also ignores any exception that might occur before the timeouts.
- Unlike webDriver wait, we have to give user-defined method to provide the condition.

Syntax:-
Version 4 → Wait(I) ← --- FluentWait(C)

Version - 3

```
Wait<WebDriver> wait = new FluentWait<WebDriver>(driver)
        .withTimeout(Duration.ofSeconds(time-in-sec), TimeUnit.SECONDS)
        .pollingEvery(Duration.ofSeconds(time-in-sec))
        .ignoring(Exception e);
WebElement element = wait.until(new Function<WebDriver, WebElement>(){
        public WebElement apply(WebDriver driver){
    WebElement e = driver.findElement(<locator>);
    if(e.isDisplayed())
        return e;
```

```
        else
            return null;
        }
    });
```

(1)

```
package Synchorinization;
import java.time.Duration;
import java.concurrent.TimeUnit;
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.FluentWait;
import org.openqa.selenium.ui.Wait;
import com.google.common.base.Function;
public class FluentWaitPractice {
    psvm(S[] args) {
    WebDriver driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.get("https://the-internet.herokuapp.com/dynamic_loading/2");
    driver.manage.timeouts().implicitlyWait(10, TimeUnit.SECONDS);
    driver.findElement(By.Xpath("//button(text() = 'start']")).click();

    Wait<WebDriver> wait = new FluentWait<WebDriver>(driver)
                .withTimeout(Duration.ofSeconds(10))
                .pollingEvery(Duration.ofSeconds(2))
                .ignoring(Exception.class);
WebElement header = wait.until(new Function<WebDriver, WebElement>(){
    public WebElement apply(WebDriver driver) {
        WebElement e = driver.findElement(By.Xpath("//h4[text()='Hello World!']"));
        if(e.isDisplayed())
            return e;
        else return null; }});
        S.o.p(header.getText());
    }  driver.close();
```

CustomWait

- It is User defined wait Statement.
- Here we use loops & Exception handling to develop wait method.

Syntax:

```
int Count ;
while (count <20)
{
  try {
     WebElement e = driver.findElement (<locators>);
     break;
  } catch (Exception e) {
     Count ++;
     Thread.sleep (1000);
  }
}
```

Ex:-

```
package Synchronization ;
import java.util.concurrent.TimeUnit ;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver ;
public class CustomWait {
 p s v m (S [] args) throwss InterruptedException {
  WebDriver driver = new ChromeDriver ();
  driver.manage().window().maximize();
  driver.get ("https://the-internet.herokuapp.com/dynamic_loading/
  driver.manage().timeouts().implicitlyWait (10, TimeUnit.SECONDS);
  driver.findElement (By.xpath ("//button[text() = "Start"]")).click
    int Count =0 ;
    WebElement header = null ;
```

```
while (count < 20)
{
  try
    {
      header = driver.findElement (By.xpath ("//h4[text()='Hello World!']"));
      break;
    }
    catch (Exception e)
    {
      count++;
      Thread.sleep (1000);
    }
  }
  S.O.P (header.getText());
  driver.close();
 }
}
```

Note :-

FluentWait and WebDriverWait are Implementing classes of Wait (I).

Differences b/w ImplicitWait & ExplicitWait.

ImplicitWait	ExplicitWait
• It Synchronizes only findElement & findElements methods.	• It Synchronizes all the methods along with findElement & findElements methods.
• Need not Specify Condition	• Condition is Mandatory.
• It is given only once in the beginning of testScript.	• It should be Specify everytime whenever Synchonization is needed.
• If the element / elements is not found it throws NoSuchElementException.	• If the Condition is not Satisfied it throws TimeOutException.
It is used to wait 50sec (polling period)	It is used to halting the execution (stop there itself).

Differences b/w WebDriverWait & FluentWait

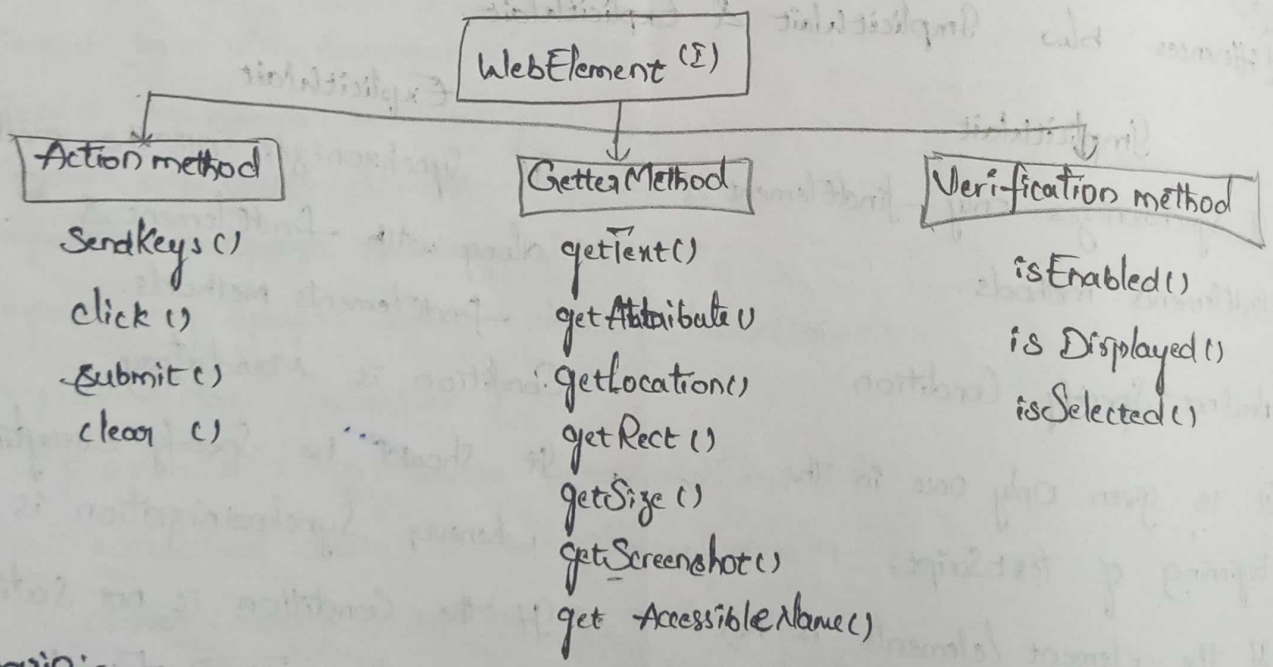
WebDriverWait	FluentWait
• It is a part of ExplicitWait.	• It is a part of ExplicitWait
• polling period can't be Customized.	• polling period can be Customized.
• It has pre-defined methods to give Condition.	• User should develop the method to provide Condition.

21/4/23
Fri

WebElement & Methods

WebElement :-

- The elements which appear on the webpage are called as WebElements.
- In Selenium WebElement is an Interface which Contains abstract methods to perform operations on WebElements.



Scenario:-

- Open the browser
- Enter Amazon.com
- Type dress in text field.
- clear the data in the text field.
- Type Smartwatches click on search button.
- close the browser.

(4) Using - SendKeys(), click(), clear()

```
package webelementmethods;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class WebElementMethods {
    psvm (String[] args) throws InterruptedException {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.amazon.com/");
        driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
        WebElement element = driver.findElement(By.id("twotabsearchtextbox"));
        element.sendKeys("dress");
        Thread.sleep(2000);
        element.clear();
        Thread.sleep(2000);
        element.sendKeys("Smartwatches");
        Thread.sleep(2000);
        driver.findElement(By.id("nav-search-submit-button")).click();
        // driver.findElement(By.id("nav-search-submit-button")).submit();
        driver.close();
    }
}
```

for Submit method

Action Methods

SendKeys() :-

- It is used for typing action or passing text to textfield.
- It takes character sequence type argument.
- The return type is void.

Click() :-

- It is generic method which is used to perform click action on an element.
- return type is void.

3. Submit ():-

- It is similar to click method but it is used only in forms
- It should be given only when the element node as the attribute type = "submit".

4. clear () :-

- It is used to delete the Content in the text field.

GetterMethod

GetText ():-

- It is used to fetch the text on the element.
- return type is String.

getTagname ():-

- It is used to fetch the tagName of the element in HTML.
- return type is string.
- Usage :- getTagname ( )

getAttribute () :-

- It is used to fetch attribute value when the attribute name is passed as argument from the element node.
- Usage :- getAttribute (string AttributeName);
- return type is String.

getAccessibleName ():-

- It is used to fetch the accessibleName of the element from the webpage.
- return type is String.

Scenario:-

- Open the browser
- Enter facebook.com
- Locate login button & Implement the above methods.

```
package webElementMethods;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
    public class GetterMethod{
     psvm(S[] args){
     WebDriver driver = new ChromeDriver();
     driver.manage().window().maximize();
     driver.get("https://www.facebook.com/");
     driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
     WebElement loginButton = driver.findElement(By.name("login"));
     S.o.pln("Text is :"+loginButton.getText());
     S.o.pln("TagName is :"+loginButton.getTagName());
work in  //S.o.pln("Accessible name is :"+loginButton.getAccessible~~Name~~());
version 4
     S.o.pln("Attribute name is :"+loginButton.getAttribute("type"));
       //here in place of type we give name, id.
     driver.close();
    }
}
```

2/4/23 later

getLocation():-

- It is used to get the location of WebElement on the Webpage.
- return type is point class, which is present in org.openqa.Selenium package.
- Point class has two methods 1. getX()
  2. getY()

getX() - It is used to get X coordinate of the element.

- return type is int.

getY() - It is used to get Y coordinate of the element.

- return type is int.

* Scenario :-

- open the browser
- Type facebook.com & get the location of login button.

```
package WebElementMethods;
import java.util.Concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.Point;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
public class GetterMethods{
psvm(s[] args){
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://www.facebook.com/");
driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
WebElement loginButton = driver.findElement(By.name("login"));
Point p = loginButton.getLocation();
S.o.pln(p.getX());
S.o.pln(p.getY());
driver.close();
}
}
```

* Scenario :-

- open the browser
- Type Amazon.com and get the location of dresses.

```
package webelementmethods;
import java.util.Concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.Point;
import org.openqa.selenium.WebDriver;
```

```
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
public class GetMethod2 {
    public static void main (String [] args) {
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get ("https://www.amazon.com/");
driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
WebElement loginButton = driver.findElement(By.xpath
                ("img [@alt = 'Dresses']"));
    Point p = loginButton.getLocation();
        S.O.pln (p.getX());
        S.O.pln (p.getY());
        driver.close();
}
}
```

GetSize():-

- It is used to fetch the size of a WebElement.
- return type is Dimension class, which is parent in org.openqa.selenium package.
- Dimension class has 2 methods. 1) getHeight()
  2) getWidth().
  - getHeight - it is used to fetch the height of the WebElement.
    - return type is int.
  - getWidth - it is used to fetch the width of the WebElement.
    - return type is int.

Scenario:-

- Open the browser
- Type amazon.com get the size of dresses image in the home page.
- close the browser.

```
package webelementmethods;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
public class GetterLocation{
  psvm(s[] args){
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://www.amazon.com/");
driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
WebElement loginButton = driver.findElement(By.Xpath
                         ("//img[@alt='Dresses']"));

Dimension p = loginButton.getSize();
   So.pln(p.getHeight());
   So.pln(p.getWidth());
   driver.close();
  }
}
```

Scenario :-

- Open the browser
- Type facebook.com
- And get the size of the login button.
- close the browser.

```
package webelementmethods;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class GetSize {
    psvm(S[] args) {
    WebDriver driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.get("https://www.facebook.com/");
    driver.mange().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
    WebElement loginButton = driver.findElement(By.xpath
                                ("//button[@name=\"login\"]"));

    Dimension p = loginButton.getSize();
        S.O.pln(p.getHeight();
        S.O.pln(p.getWidth();
        driver.close();
    }
}
```

getRect();-

- It is used to fetch the rectangular dimension as well as location of the WebElement.
- The return type is rectangle class, which is present in org.openqa.selenium. package.
- It has 4 methods
  1. getHeight()
  2. getWidth()
  3. getX()
  4. getY().

(3)

Scenario

- open the browser
- type amazon.com
- get the rectangular dimension & location of dresses image in the homepage
- close the browser.

```
package webelementmethods;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.Rectangle;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
    public class GetRect{
        psvm(S[] args){
    WebDriver driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.get("https://www.amazon.com/");
    driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
    WebElement loginButton = driver.findElement(By.Xpath
                                ("//img[@alt='Dresses']"));
        Rectangle r = loginButton.getRect();
        s.o.pln(r.getHeight());
        s.o.pln(r.getWidth());
        s.o.pln(r.getSize());
```

```
        S.o.pln(r.get.y());
        driver.close();
    }
}
```

Scenario :-

(4)

- open the browser
- type facebook.com.
- get the rectangular dimension & location of login button in the homepage.
- close the Browser.

```
package webelementmethods;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.Rectangle;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
    public class GetRect2 {
    psvm(S[] args) {
    WebDriver driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.get("https://www.facebook.com/");
    driver.manage.timeouts().implicitlyWait(10, TimeUnit.SECONDS);
    WebElement rect = driver.findElement(By.Xpath
                                ("//a[text()='create new account']"));
        Rectangle r = rect.getRect();
        S.o.pln(r.getHeight());
        S.o.pln(r.getWidth());
        s.o.pln(r.getX());
        S.o.pln(r.getY());
        driver.close();
    }
}
```

24/4/23 ~~Step to Capture~~ How to Maven repository & Apache Commons io

- Open the browser & type maven repository.
- click on the 1st link.
- Now type apache commons io in search bar & click.
- click on the 1st option. then u should click on 2.11.0 version
- Now u should click on jar (319kB). , automatically it is downloaded.
- Go to downloads & copy commons io & paste it on the project.
- Right click on jar file & add to buildpath.

GetScreenshotAs():-

- This method is used to fetch the Screenshot of an element.. we use apache commons io libraries to fetch the Screenshot.

Steps to Capture ElementScreenshot.

Java.io. 1. Capture Screenshot

File Src = element.getScreenshotAs(OutputType.FILE);

2. Create a file to store ss.

File dest = new file("./Elementss/myj.png");

3. Copy Scr to dest.

apache.commons.io library

↖ FileUtils.copyFile(src, dest); → Throws IOException.

→ package webelementmethods;

import java.io.File;

import java.io.IOException;

import java.util.concurrent.TimeUnit;

import org.apache.commons.io.FileUtils;

import org.openqa.selenium.By;

import org.openqa.selenium.OutputType;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

```
import org.openqa.selenium.chrome.ChromeDriver;
public class GetScreenshot{
  psvm (S[] args) throws IOException{
    WebDriver driver = new ChromeDriver();
     driver.manage().window().maximize();
     driver.get("https://www.swiggy.com/");
    driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
    WebElement login = driver.findElement(By.Xpath("//button[.'Login in']"));
  WebElement signUp = driver.findElement(By.Xpath("//a[text()='Sing up']"));
     File src = sinUp.getScreenshotAs(OutputType.FILE);
     File dest = new File("./ElementScreenshot/SinUp.png");
     FileUtils.copyFile(src,dest);
     driver.close();
  }
}
```

Verification methods:-

Is-Displayed():- isDisplayed() is used to check if an element is display on the webpage or not.

- return type is boolean.

isEnabled():-

- This method is used to verify if an element is enabled or not.
- return type is boolean.

isSelected():-

- This method is used to check if an element is selected or not.
- return type is boolean.

IsDisplayed

```
package webelementmethods;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class IsDisplayed{
    psvm (S[] args) {
    WebDriver driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.get ("https://www.Swiggy.com/");
    driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
    WebElement logo = driver.findElement(By.Xpath("//*[name()='svg']"));
    if(logo.isDisplayed())
        S.o.pln ("logo displayed");
    else
        S.o.pln("logo not displayed");
    driver.close();
}
}
```

IsEnabled

```
package webelementmethods;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver();
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class IsEnabled {
    psvm (S[] args) {
```

```
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://www.instagram.com/");
driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
WebElement login = driver.findElement(By.xpath("//button[.='Log in']"));
if(login.isEnabled())
    S.o.println("enabled");
else
    S.o.println("disabled");
driver.findElement(By.name("username")).SendKeys("Vaaa");
driver.findElement(By.name("password")).sendKeys("12345");
if(login.isEnabled()){
    S.o.println("enabled");
    login.click();
}
else
    S.o.println("disabled");
    driver.close();
}
}
```


IsSelected :-

```
package webelementmethods;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
public class IsSelected{
    psvm(S[]args){
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://www.facebook.com/");
```

```
driver.manage().timeOuts().implicitlyWait(10, TimeUnit.SECONDS);
driver.manage().findElement(By.xpath("//a[@data-testid=\"open-registration-
                                    form-button\"]")).click();
WebElement female = driver.findElement(By.xpath("//input[@name='sex'
                                    and @value=\"1\"]"));

    female.click();
if(female.isEnabled())
{
    s.o.pln("Selected");
}
else
    s.o.pln("disabled");
    driver.close();
}
}
```

Limit the Scope of Driver

```
package webelementmethods;
import java.util.List;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class ScopeMethods {
    ps vm(S[] args) {
    WebDriver driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.get("https://www.facebook.com/");
    driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
WebElement footer = driver.findElement(By.id("pageFooter"));
List<WebElement> links = footer.findElements(By.tagName("a"));
for(WebElement link:links){
    S.o.pln(link.getText());
}
}  driver.close();
```

25/4/22

GetCssSelectorValue :-

- This method fetches the properties of an element like Colour, font, background colour etc,.
- return type is String. • It accepts String type arguments.

→
```
package webelementmethods;
import java.util.selenium.By; Concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.Selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
public class GetSelectorValue {
    psvm(S[] args) {
    WebDriver driver = new ChromeDriver();
    driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
    WebElement createAccount = driver.findElement(By.Xpath
                ("//a[@data-testid='open-registration-formbutton']
        S.o.pln(createAccount.getCssValue("color"));
        S.o.pln(createAccount.getCssValue("font"));
        S.o.pln(createAccount.getCssValue("background-color"));
        driver.close();
    }
}
```

1) Differences b/w CssSelector & Xpath.

CssSelector	Xpath
• It is unidirectional. only forward traversing is possible.	• It is Bidirectional. Both forward & backward traversing is possible.
• It doesn't support text.	• It supports text.
• It is faster.	• It is bit slower Compared to CssSelector.

driver.close

1. It is an abstract method.
2. kill the ~~pa~~ current browser.

driver.quit

1. It is an abstract method
2. kill the instance of the browser. (closing parent & child browser)

25 July 23 Tuesday

Chapter - 2

Handling WebElements

Auto Suggestions :-

Auto suggestions can be handled using findElements method.

```
package handlingwebelements;
import java.util.List;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
public class AutoSuggestions {
    psvm (String[] args) throws InterruptedException {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.amazon.com/");
        driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS
        driver.findElement(By.id("twotabsearchtextbox")).sendKeys("gifts"
        Thread.sleep(2000);
        List<WebElement> list = driver.findElement(By.xpath
                ("div[contains(@aria-label,'gifts')]"));
        for(WebElement e:list)
        {
            S.o.pln(e.getText());
        }
        Thread.sleep(2000);
        driver.close();
    }
}
```

Task:-

Scenario:-

- Open the browser
- Enter the browser
- type your name Search texefield.
- fetch all autoSuggestions
- print the 5th element in Console.

```
package actions;
import java.time.Duration;
import java.util.List;
import org.openqa.selenium.By;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.chrome.ChromeOptions;*
    public class Actions{
    ps vm (s[] args) throws InterruptedException{
    ChromeOptions options = new ChromeOptions();*
     chromeDriver driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.get("https://www.google.com/");
   driver.manage().timeouts().implicitlyWait(Duration.of(10, TimeUnit.SECONDS));
   driver.findElement(By.xpath("//textarea[@name='q']")).
                                        sendKeys("vagalaxmi");
    Thread.sleep(2000);
List<WebElement> list = driver.findElements(By.xpath
                        ("//li[contains(@data-view-type,'1')][5]"));
   for(WebElement e: list)
   {
       S.o.pln(e.getText());
   }
   Thread.sleep(2000);
   driver.close();
}
}
```

Mouse Actions:-

- Mouse actions can be handled using Actions class.
- Actions class is present in org.openqa.selenium.interactions. package.

Steps to handle mouse Actions:-

1. Mouse b/ Actions a = new Actions (driver);
2. mouse hovey → a.moveToElement (element).perform();
3. right click → a.contextClick (element).perform();
4. Double click → a.doubleClick (element).perform();

drag & drop → a.dragAndDrop (element, target).perform();

perform:-

perform is the mandatory method it should be given with the action methods.

26/4/23 Wed

MouseHovey:-

```
package autosuggestions;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.interactions.Actions;

public class MouseHover {
psvm(s[] args) {
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://www.myntra.com/");
driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
WebElement kids = driver.findElement(By.xpath
("//a[text()='kids' and @class]"));
Actions a = new Actions(driver);
a.moveToElement(kids).perform();
```

```
driver.findElement(By.xpath("//a[text()='Soft Toys']")).click();
WebElement header = driver.findElement(By.xpath
                    ("//div[@class='filter-Summary-filter']"));
if(header.getText().contains("Soft Toys")){
    S.o.pln("test pass");
}
else
    S.o.pln("test fail");
driver.close();
}
}
```

Rightclick:-

```
package autoSuggestions;
import java.awt.AWTException;
import java.awt.Robot;
import java.awt.event.KeyEvent;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.interactions.Actions;
public class Rightclick{
psvm(s[] args) throws AWTException, InterruptedException{
    WebDriver driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.get("https://www.amazon.com/");
    driver.manage().timeouts().implicitlyWait(10, TimeUnits.SECONDS);
WebElement todaysDeals = driver.findElement(By.xpath
                    ("//a(text()=\"Today's Deals\"]"));
driver.findElement(By.xpath("//input[@data-action-type=
                    \'DISMISS\"]")).click();
Actions a = new Actions(driver);
a.ContextClick(todaysDeals).perform();
```

```
Robot r = new Robot();
r.keyPress(KeyEvent.VK_RIGHT);
r.keyRelease(KeyEvent.VK_RIGHT);
r.keyPress(KeyEvent.VK_ENTER);
r.keyRelease(KeyEvent.VK_ENTER);
Thread.sleep(3000);
driver.close(); //driver.quit();
}
}
```

(16)

Doubleclick :-

```
package actions;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.interactions.Actions;
public class DoubleClick{
psvm(String[] args){
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://demoapp.skillrary.com/");
driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
WebElement courselink = driver.findElement(By.id("Course"));
Actions a = new Actions(driver);
a.moveToElement(courselink).perform();
driver.findElement(By.Xpath("//span/a[text()='selenium Training']")).click();
WebElement value = driver.findElement(By.id("quantity"));
(Select on (-) Button)
int intialQualtity = Integer.parseInt(value.getAttribute("value"));
WebElement plusButton = driver.findElement(By.id("add"));
a.doubleClick(plusButton).perform();
```

```
int finalQuantity = Integer.parseInt(value.getAttribute("value"));
if(finalQuantity == intialQuantity+1)
    S.o.pln("Quantity increased pass");
else
    S.o.pln("Fail");
driver.close();
}
}
```


Drag & Drop

```
package actions;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.interactions.Actions;

public class Drag-andDrop {
    psvm(S[] args) throws InterruptedException{
    WebDriver driver = new ChromeDriver;
    driver.manage().window().maximize();
    driver.get("http://www.dhtmlgoodies.com/Submitted-Scripts/
                i-google-like-drag-drop/index.html");
driver.manage().window().ImplicitlyWait(10, TimeUnits.SECONDS);
WebElement block1 = driver.findElement(By.Xpath
                ("//h2[text()='Block 1']"));
WebElement block4 = driver.findElement(By.Xpath
                ("//h2[text()='Block 4']"));
    Actions a = new Actions(driver);
    a.dragAndDrop(block1, block4);
    Thread.sleep(2000);
    driver.close();
}
}
```

Frames

- These are the webpages which are developed separately and kept inside the main webpage. (page inside page)
- To identify frame.
  - Right click any where in the frame, new frame Source Option.
  - When frame element is inspected, we get the following.

```
<iframe>
  <html>
    :
  </html>
</iframe>
```

Handling frames:-

- Frames can be handled, by Switching the Control to the frame.

```
driver.SwitchTo() { frame(int index);
                    frame(String id or Name);
                    frame(WebElement, frameElement); }
```

} overloaded methods

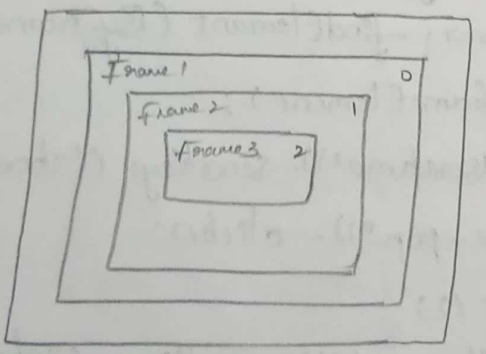
- To Switch Back from the frame.

```
driver.SwitchTo().defaultContent();
```

- SwitchTo - return type

TargetLocator - static Interface

Frame Indexing:-



Frames program

```
package actions;
import java.awt.Robot;
import java.awt.event.KeyEvent;
import java.util.concurrent.TimeUnit;
```

Frames

```
package handling-webelements;
import java.awt.Robot;
import java.awt.event.KeyEvent;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chromeDriver;
import org.openqa.selenium.interaction.Action;

public class Frames {
    psvm (s[] args) {
        WebDriver driver = new chromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.snapdeal.com");
        driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS
        WebElement signin = driver.findElement(By.xpath
                ("//span[text()='Sign in']"));
        Actions a = new Actions(driver);
    a.moveToElement(signin).perform();
// driver.switchTo().frame(0); // index - 0.
// driver.switchTo().frame("login_iframe");
WebElement frameElement = driver.findElement(By.name("login_frameLogin
 driver.switchTo().frame(frameElement);
 driver.findElement(By.id("username")).sendKeys("shoes");
driver.findElement(By.id("close-pop")).click();
driver.switchTo().defaultContent();
driver.findElement(By.name("keyword")).sendKeys("shoes");
driver.findElement(By.className("searchTextSpan")).click();
String header = driver.findElement(By.xpath("//div[@id="SearchMess
                Container']/descendant::span")).getText();
if(header.contains("shoes"))
        s.o.pln("test pass");
else
        s.o.pln("test fail");
driver.quit();
}}
```

Windows :-

- driver.SwitchTo().newWindow(WindowType.TAB);

The above Statement is used to open a new tab & Switch the Control to it.

- driver.SwitchTo().newWindow(WindowType.WINDOW);

the above Statement is used to open a new window & Switch the Control to it.

TAB — This programs will work in Version (4).

```
package selenium;
~~public class import java.util.concurrent.TimeUnit;~~
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.chrome.ChromeOptions;
import org.openqa.selenium.WindowType;
    public class SwitchToTab{
    psvm(s[] args) throws InterruptedException{
        ChromeOptions options = new ChromeOptions();
        options.addArguments("--remote-allow-origins=*");
        WebDriver driver = new ChromeDriver(options);
        driver.manage().window().maximize();
        driver.get("https://www.google.com/");
        Thread.sleep(2000);
TAB     driver.SwitchTo().newWindow(WindowType.TAB);
WINDOW  // driver.SwitchTo().newWindow(WindowType.WINDOW);
        driver.get("https://www.Swiggy.com/");
        driver.quit();
    }
}
```

Screenshot Of Window :-

1. Type caste the

   TakesScreenshot ts = (TakesScreenshot) driver;

2. File Src = ts.getScreenshotAs (OutputType.FILE);

3. File dest = new File (". /screenshot / webpage . png");

4. FileUtils.copyFILE (Src, dest); — IO Exception.

   apache Commons io

→

```
package handlingwebelements;
import java.io.File;
import java.io.IOException;
import org.openqa.selenium.OutputType;
import org.apache.commons.io.FileUtils;
import org.openqa.selenium.TakesScreenshot;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
public class ScreenshotOfAWindow {
    psvm (S[] args) throws IOException {
    WebDriver driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.get ("Https://www.swiggy.com");
    TakesScreenShot ts = (TakesScreenshot) driver;
    File Src = ts.getScreenshotAs (OutputType.FILE);
    File dest = new File (". /Windowss / swiggy.png");
    FileUtils.copyFile (src, dest);
    driver.close();
}
}
```

Screenshot of Webpage :-

```
package handlingwebelements;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

    public class
import java.io.File;
import java.io.IOException;
import javax.imageio.ImageIO;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import ru.yandex.qatools.ashot.Ashot;
import ru.yandex.qatools.ashot.Screenshot;
import ru.yandex.qatools.ashot.shooting.shootingStrategies;
  public class ScreenshootOffullWebpage {
      psvm (S[] args) throws IOException{
      WebDriver driver = new ChromeDriver();
      driver.manage().window().maximize();
      driver.get("https://www.amazon.com/");
    Screenshot screenshot = new Ashot().shootingStrategy(ShootingStrategies.viewportPasting(1000))
                            .takeScreenshot(driver);
    File dest = new File("./Webpage Screenshot/amazon.png");
    ImageIO.write(Screenshot.getImage(), "PNG", dest);
      driver.close();
  }
}
```

8/14/21

Dropdowns :-

How do we perform dropdown if there is no select tag in facebook.

-A:- using Robot classes

Static dropdowns :-

- These are the webelements which when click on dropsdown predefined set of options.
- To identify static dropdowns element node Contains.

```
<select>
  <option value = " "> text </option>
  <option value = " "> text </option>
  - - - - - - - - - - - - - -
</select>
```

- Static dropdowns are 2 types. 1. Single Select dropdown
  2. Multi Select dropdown

Single Select dropdown :-

- The Static dropdown which allows us to select single option at a tim
- We cannot deselect single Select dropdown.

Multiselect dropdown :-

- The static dropdown which allows us to select more than one option is called multiselect dropdown.
- We can deselect multi select dropdown.

* Static dropdowns can be handled using Select©.

- Select© is present in org.openqa.selenium.support.ui. package.

Steps to handle static dropdown :- (11 methods)

Create an Instance of class.

* Select s = new Select (element);

1. To select an option —
   1. s.selectByIndex(int index);
   2. s.selectByValue(String value);
   3. s.selectByVisibleText(String text);
   - Return type is void.

4. To check if it is multiselect → s.isMultiple();
   - return type is boolean.

5. To get all the options from dropdown → s.getOptions();
   - return type is List<WebElement>.

6. To get first selected option → s.getFirstSelectedOption();
   - return type is WebElement.

7. To get all selected options → s.getAllSelectedOptions();
   - return type is List<WebElement>.

8. To deselect an option → s.deselectByIndex(int index);
   9. s.deselectByValue(String value);
   10. s.deselectByVisibleText(String text);
   - return type is void.

11. To deselect all options → s.deselectAll();
   - return type is void.

StaticDropdown :-

```
package dropdowns;
import java.util.concurrent.TimeUnit;
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 StaticDropdown{
        psvm(s[] args) throws InterruptedException{
          WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.amazon.com/");
      driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECOND
```

```
WebElement dropdown = driver.findElement(By.id
                    ("SearchDropdownBox"));
Select s = new Select(dropdown);
if(s.isMultiple())
    s.o.pln("multi select dropdown");
else
    s.o.pln("single select dropdown");
 s.selectByIndex(7);
 Thread.sleep(2000);
 s.selectByValue("search-alias-digital-text");
   Thread.sleep(2000);
 s.selectByVisibleText("Kindle Store");
 Thread.sleep(2000);
 driver.close();
}
}
```

Select & Deselect

```
package dropdowns;
import java.util.List;
import java.util.concurrent.TimeUnit;
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 DeselectedOptions{
      psvm(S[] args) throws InterruptedException{
      WebDriver driver = new ChromeDriver;
      driver.manage().window().maximize();
      driver.get("https://demoapp.skillrary.com/");
    driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
```

```
WebElement dropdown = driver.findElement(By.id("Cars"));
Select s = new Select(dropdown);
List<WebElement> list = s.getOptions();
for(WebElement e : list)
    S.o.pln(e.getText());
if(s.isMultiple())
{
    s.selectByIndex(0);
    s.selectByIndex(1);
    s.selectByIndex(2);
Thread.sleep(2000);
    S.o.pln("\n First option :" + s.getFirstSelectedOptions().
                                    getText());
List<WebElement> selectedList = s.getAllSelectedOptions();
for(WebElement opt = selectedList)
{
    S.o.pln(opt.getText());
}
    s.deselectAll();
}
else
    S.o.pln("Single select dropdown");
    driver.close();
}
}
```

29/4/23
Saturday

JavaScriptExecutor

- Javascript executor is an Interface in WebDriver class diagram it is used to execute javascript code in selenium.
- It has 2 abstract methods 1- executeScript()
  2. executeAsyncScript().

1. Handle Scrollbar
   - Using hardcoded coordinates
   - Using element location
   - Using element reference
2. Navigate to an application
3. pass data to an text field.
4. click on an element
5. fetch title & Url of Current Webpage.
6. Refresh the webpage.
7. Handle shadow root elements
8. Handle disabled elements.

① Using hardcoded coordinates

a)
```
package javascriptexecutor;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class ScrollByUsingHardcoded {
    psvm (String[] args) throws InterruptedException {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.amazon.com/");
```

```
driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
Thread.sleep(2000);
JavascriptExecutor js = (JavascriptExecutor) driver;
js.executeScript("window.ScrollBy(0, 2000)");
Thread.sleep(2000);
js.executeScript("window.ScrollBy(0, -2000)");
Thread.sleep(2000);
driver.close();
}
}
```

(5)

(b) Using ElementLocation.

```
package javascriptexecutor;
import java.util.concurrent.TimeUnit;
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;
public class ScrollUsingElementLocation {
psvm(String[] args) throws InterruptedException {
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://www.amazon.com/");
driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
Thread.sleep(2000);
WebElement electronics = driver.findElement(By.Xpath
("//h2[text()='Electronics']"));
Point loc = electronics.getLocation();
int x = loc.getX();
int y = loc.getY();
```

```
JavascriptExecutor js = (JavascriptExecutor) driver;
    js.executeScript ("window.scrollBy ("+x+", "+y+")");
      Thread.sleep (2000);
      driver.close ();
  }
}
```

© Using ElementReference :-

```
package javascriptexecutor;
import java.util.concurrent.TimeUnit;
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 ScrollUsingElementReference {
  psvm (S[] args) throws InterruptedException {
    WebDriver driver = new ChromeDriver ();
    driver.manage().window().maximize();
    driver.get ("https://www.amazon.com/");
  driver.manage().timeouts().implicitlyWait (10, TimeUnit.SECONDS);
    Thread.sleep (2000);
  WebElement fitnessNeeds = driver.findElement (By.xpath
                ("//h2[text() = 'For your FitnessNeeds']"));
    JavascriptExecutor js = (JavaScriptExecutor) driver;
      js.executeScript ("arguments[0].ScrollIntoView (true)",
                          fitenessNeeds);
      Thread.sleep (2000);
      driver.close ();
  }
}
```

→ Pass Data to text-field & click on element :-

```
package javascriptexecutor;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.By;
public class passDataAndclickOnElement {
    psvm (s[]a) throws InterruptedException {
    WebDriver driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.get ("https://www.amazon.com/");
    driver.manage().timeouts().implicitlyWait (10, TimeUnit.SECONDS
    Thread.sleep (2000);
    WebElement searchTF = driver.findElement (By.id
                ("twotabsearchtextbox"));
    WebElement searchButton = driver.findElement (By.id
                ("nav-search-submit-button"));
    JavascriptExecutor js = (JavascriptExecutor) driver;
    js.executeScript ("arguments[0].value = arguments[1]",
                searchTF, "shoes");
    js.executeScript ("arguments[0].click()", searchButton);
    Thread.sleep (2000);
    driver.close();
    }
}
```

285

```
package javascriptexecutor;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
public class NavigateAndFetchTitleUrlRefresh{
    psvm(s[]a) throws InterruptedException{
    WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
JavascriptExecutor js = (JavascriptExecutor) driver;
js.executeScript("window.location = arguments[0]",
                 "https://www.flipkart.com/");
S.O.pln(js.executeScript("return document.title"));
S.O.pln(js.executeScript("return document.URL"));
Thread.sleep(2000);
    js.executeScript("history.go(0)");
    Thread.sleep(2000);
      driver.close();
  }
}
```

1/5/23
Mon

Steps to handle shadow root Elements :-

1. Inspect the element and right click on element node → Copy → Copy js path
2. JavascriptExecutor js = (JavascriptExecutor) driver;
   Object elementObj = js.executeScript("return <jspath>");
3. Typecast eleObj to WebElement.
   WebElement element = (WebElement) eleObj;
4. Use element to perform webelement operations.

```
package javascriptexecutor;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
public class ShadowRootElement {
    psvm(S[] args) throws InterruptedException {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("chrome://downloads/");
    driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
     JavascriptExecutor js = (JavaScriptExecutor) driver;
    Object elementObj = js.executeScript("return document.querySelector
    (\"body > downloads-manager\").shadowRoot.querySelector
    (\"#toolbar\").shadowRoot.querySelector(\"#toolbar\").
      shadowRoot.querySelector(\"#search\").shadowRoot.
       querySelector(\"#searchInput\")");
   WebElement SearchTF = (WebElement) elementObj;
       SearchTF.SendKeys("Vaya");
        Thread.sleep(2000);
        driver.close();
    }
}
```

8) →

```
package javascriptexecutor;
import java.util.concurrent.TimeUnit;
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 DisabledElements {
    psvm(s [] args) throws InterruptedException {
    WebDriver driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.get("file:///c:/Users/hgaur/oneDrive/Documents/
            disabledExample.html");
    driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
→ //driver.findElement(By.id("user")).sendKeys("varalakshmi");
            // not work for normal id's.
    WebElement textbox = driver.findElement(By.id("user"));
    JavascriptExecutor js = (JavascriptExecutor) driver;
js.executeScript("arguments[0].value = arguments[1]",
            textbox, "varalaxmi");

    Thread.sleep(2000);
    driver.close();

}

}
```

Popups

popups are the small windows which appear on the webpage that gives vital information to the user.

Types of popups:-

1. Alert / confirmation / Javascript popup
2. Hidden division / calendar popup
3. child browser popup
4. Notification popup
5. File upload popup (nowadays it is directly download).

(1) Alert / confirmation / Javascript popup

- This popups are develop using javascript.
- This popups are not inspectable nor movable.

Alert (1) — * Alert popup is handle using Alert (1).

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

II. al.accept(); → To click on 'Ok'
al.dismiss(); → To click on 'cancel'
al.sendKeys("data"); → To pass data to popup.
al.getText(); → To fetch Text on popup.

→ package popups;

```
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.Alert;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
```

```
public class Alertpractice {
    psvm (s [] a) throws InterruptedException {
    WebDriver driver = new chromeDriver();
    driver. manage (). window(). maximize();
    driver. get ("https://the-internet. herokuapp. com /
                javascript – alerts");
    driver. manage (). timeouts (). implicitlyWait (10, TimeUnit. SECONDS);
driver. findElement (By. Xpath ("// button [text () =
                'click for Js Alert']"). click();
    Alert al = driver. switchTo (). alert();
      S.O.pln (a1. getText());
       al. accept();
       Thread. sleep();
      driver. close();
  }
}
```

2)

```
package popups;
import java. util. concurrent. TimeUnit;
import org. openqa. selenium. Alert;
import org. openqa. selenium. By;
import org. openqa. selenium. WebDriver;
import org. openqa. selenium. chrome. ChromeDriver;
    public class Alertpractice2 {
     psvm(s [] a) throws InterruptedException {
     WebDriver driver = new chromeDriver();
      driver. manage(). window(). maximize();
     driver. get(" https://the-internet. herokuapp. com / javascript –
                                                alerts");
```

```
public class AlertPractice {
    psvm (s[] a) throws InterruptedException {
    WebDriver driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.get("https://the-internet.herokuapp.com/
                javascript_alerts");
    driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
driver.findElement(By.Xpath("//button[text()=
                'click for Js Alert']")).click();
    Alert al = driver.switchTo().alert();
     S.o.pln (a1.getText());
     al.accept();
     Thread.sleep();
     driver.close();
  }
}
```

2)

```
package popups;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.Alert;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
    public class AlertPractice2 {
    psvm(s[] a) throws InterruptedException {
    WebDriver driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.get("https://the-internet.herokuapp.com/javascript_
                alerts");
```

```
driver.manage().timeouts().implicitlyWait(10, TimeUnits.SECONDS);
driver.findElement(By.Xpath("// button[text() =
        'click for Js Confirm 'J'")).click();
Alert al = driver.switchTo().alert();
    S.O.pln(a1.getText());
        al.dismiss();
        Thread.sleep(2000);
        driver.close();
    }
}
```

(3)

```
package Popups;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.Alert;
import org.openqa.selenium.By;
import org.opena.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
public class Alertpractice3 {
    psvm(S[] args) throws InterruptedException {
    WebDriver driver = new ChromeDriver;
     driver.manage().window().maximize();
driver.get("https://the-internet.herokuapp.com/
                    javascript_alerts");
driver.manage().timeouts().implicitlyWait(10, TimeUnits.SECONDS
driver.findElement(By.Xpath("// button[text() = 'click for
                    Js Prompt 'J'")).click();
    Alert al= driver.switchto().dlert();
    Ale  S.O.pln(al.getText());
        al.SendKeys("ammu");
        al.accept();
        Thread.sleep(2000);
```

```
        driver.close();
    }
}
```

15/2023
ries

2) Calender / hidden division popup:-

- Calender popups are Inspectable but not movable.
- We can handle this popups using findElement().

→
```
package popup;
import java.util.Concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
    public class Calenderpopup
    {
     public static void main (String[] args)
     {
       WebDriver driver = new ChromeDriver();
       driver.manage().window().maximize();
       driver.get ("https://www.yatra.com/");
      driver.manage().timeouts.implicitlyWait (10, TimeUnit.SECONDS);
      driver.findElement (By.id ("BE_flight_origin_date")).click();
      driver.findElement (By.id ("24/06/2023")).click();
      Thread.sleep (2000);
      driver.close();
    }
}
```

3) Child browser popup :-

Imp Interview

- It is Inspectable & movable.
- child browser popup can be handle using following webdriver methods - 1. GetWindowHandle() - return type is String
  2. getWindowHandles() - return type is Set<String>

getWindowHandle() :-

- This method returns parent window address.
- return type is String.

getWindowHandles() :-

- This method returns both parent & child window address.
- return type is Set<String>.

→

```
package popup;
public class import java.util.Set;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
    public class ChildBrowserpopup {
        psvm (S[] args) throws InterruptedException {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.flipkart.com/");
    driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
    driver.findElement(By.XpathC"//button[@class='_2kpZ61-2doB4z
        click();
    WebElement SearchTF = driver.findElement(By.name("q"));
    SearchTF.SendKeys("iphone");
```

```
driver.findElement(By.XpathClassName("LDz3pu")).click();
driver.findElement(By.Xpath("//span[text()='APPLE iphone 13 (Blue, 128 GB)']")).click();

String parentID = driver.getWindowHandle();
Set<String> ids = driver.getWindowHandles();

for(String id : ids)
{
  driver.switchTo().window(id);
}
String rating = driver.findElement(By.Xpath("//span[text()='APPLE iphone 13 (Blue, 128 GB)']/ancestor::div[@class='aMaAEs']
/descendant::div[@class='_3LWZ1k']")).getText();
S.O.println(rating);
driver.close();
driver.switchTo().window(parentID);
Thread.sleep(2000);
searchTF.clear();
searchTF.sendKeys("Smart watch");
driver.findElement(By.className("LDz3pu")).click();
Thread.sleep(2000);
}} driver.close();
```

4) File upload popup:-

- This popups are movable but not Inspectable.
- we can handle this popups in following ways
  1. Using sendkeys.
  2. Using Robot ©
  3. Using AutoIT

(1) Using SendKeys

- Whenever we have type = "file" attribute in the element node, we can use sendKeys to upload the file.

```
package popups;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
public class FileUploadUsingSendkeys {
    psvm(String[] args) throws InterruptedException {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.foundit.in/");
    driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
    driver.findElement(By.Xpath("//div[contains(text(), 'Upload Resume')]
            .click();
    driver.findElement(By.id("file-upload")).sendKeys
        ("C:\\Users\\deeks\\Downloads\\TCS application form.pdf");
    Thread.sleep(2000);
    driver.close();
}
}
```

3/10/18 Wed

FileUploadUsingRobotClass :-

Website - Naukri.com , click on register.

→
```
package popup;
import java.awt.AWTException;
import java.awt.Robot;
import java.awt.Toolkit;
import java.awt.datatransfer.StringSelection;
import java.awt.event.KeyEvent;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
  public class FileUploadUsingRobotClass {
      psvm (String[] args) throws AWTException, InterruptedException {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.naukri.com/registration/createAccount?
                                    otherscrp=22836");
        driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
    driver.findElement(By.Xpath("//button[text()='upload Resume']")).click();
     Thread.sleep(5000);
  StringSelection str = new StringSelection("c:\\users\\nganesh\\Downloads\\vara resu
                                    .doc");
   Toolkit.getDefaultToolkit().getSystemClipboard().setContents(str, null);
     Robot r = new Robot();
       r.keyPress(KeyEvent.VK_CONTROL);
       r.keyPress(KeyEvent.VK_V);
       r.keyRelease(KeyEvent.VK_CONTROL);
       r.keyRelease(KeyEvent.VK_V);
       r.keyPress(KeyEvent.VK_ENTER);
       r.keyRelease(KeyEvent.VK_ENTER);
        Thread.sleep(6000);
} }     driver.close();
```

Notes :- Steps FileUploadUsingRobotclass

- Robot class is used to automate all keyboard related actions
- it is available in java.awt.package
- awt - Abstract window Toolkit.

Step-1 :- create an Instance for Robot class.

Robot robot = new Robot();

Step2 :- create an Instance for StringSelection class & pass the path of the file to be Copied to the file upload popup.

StringSelection path = new StringSelection ("file-path");

StringSelection class is available in java.awt.datatransferpath

It is used for all data transfer purposes.

Step-3 :- Set the Contents to the clipboard using Toolkit class

Toolkit.getDefaultToolkit().getSystemClipboard().SetContents(path,null);

Step-4 :- press ctrl+v to ~~Copy~~ Paste file path from the clipboard.

robot.keyPress(KeyEvent.VK_CONTROL);

robot.KeyPress(KeyEvent.VK_V);

Step-5 :- Release the pressed keys

robot.keyRelease(KeyEvent.Vk_CONTROL);

robot.keyRelease(KeyEvent.Vk_V);

Step-6 :- Press and release enter button.

robot.KeyPress(KeyEvent.Vk_ENTER);

robot.KeyRelease(KeyEvent.Vk_ENTER);

SciTE Script Editor

Steps to launch SciTE Script Editor

1. Type Scite Script Editor in Search
2. Open the app.

Standalone application Scenario Steps:-

1. Switch to the opened file popup window
2. Switch the Control to filename - text box
3. Type file path
4. click on Open button

Step-1:- WinWaitActive ("title")

This method is used to wait until the file upload popup window is active and switches the Control to it.

Step-2:- Sleep (2000)

In autoIT Sleep() is the only wait statement.

Step-3:- ControlFocus ("title", "text", "ControlID")

In order to Inspect an element in 'Stand alone' application

1. Search for 'autoIT v3 Window Info' in Computer.
2. Drag & drop the 'Finder Tool' to the element to be Inspected.

click on Control in AutoIT v3 Window Info.

ControlFocus method is used to switch the Control over to the element.

Control ID : It is the Combination of class and Instance.

Step-4:- Send ("path_of_the_file_to_be_uploaded")

Send method is used to type data in to the element.

Step-5:- sleep (2000)

Step-6:- ControlClick ("title","text","controlID")

ControlClick method is used for click on the element.

--> After writing the program, save it in a folder in desktop with .au3 extension.

--> To Compile the program in Scite ScriptEditor, click on 'Compile' click on 'Compile Script' button.

We can see .exe file auto-saved to the folder on dekstop.

Now integrate the above code with Selenium program:

Runtime.getRuntime().exec("path of .exe file");

AutoIT

Website - Naukri.com - click on register

```
package popup;
import java.io.IOException;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
public class FileUploadUsingAutoIT {
    psvm(S[] args) throws InterruptedException, IOException {
    WebDriver driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.get("https://www.naukri.com/registration/createAccount?
                                    otherSocp=22636");
    driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
    driver.findElement(By.xpath("button[text()='Upload Resume']")).click();
    Runtime.getRuntime().exec("c:\\users\\ngane\\OneDrive\\Documents\\
                Naukrifileupload.exe");
    Thread.sleep(5000);
    drive.close();
    }
}
```

22/5/2023 Thurs
Prep for Interview

Notification popup :-

- These are browser Specific popups
- They are not Inspectable nor movable.
- These popups can be handled using browser Specific classes as follows.
  - chrome → ChromeOptions ©
  - firefox → FirefoxOptions ©
  - edge → EdgeOptions ©

- Website - ajio.com (we disable notification popup).

```
package popup;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.chrome.ChromeOptions;

public class Notificationpopup {
    psvm (String[] args) throws InterruptedException {
        ChromeOptions options = new ChromeOptions();
        options.addArguments("--disable-geolocation");
        options.addArguments("--disable-notification");

        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.ajio.com/");
        Thread.sleep(2000);
        driver.close();
    }
}
```

// HashMap Object

Content.settings.put({notifications, geolocation, media-stream}, {0 → Ask, 1 → Allow, 2 → Block});

profile.put("managed_default_content_settings", ContentSettings);

preference.put("profile", profile);

chromeOptions

Options.setExperimentalOption("prefs", preference);

website – Ajio.com

1.
```
package popups;
import java.util.HashMap;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.chrome.ChromeOptions;
public class HandleAnyNotification{
psvm(SC[] a) throws InterruptedException{
HashMap<String, Integer> ContentSettings = new HashMap<>();
HashMap<String, Object> profile = new HashMap<>();
HashMap<String, Object> preference = new HashMap<>();
ContentSettings.put("geolocation", 1);
ContentSettings.put("notifications", 2);
profile.put("managed_default_content_settings", ContentSettings);
preference.put("profile", profile);
ChromeOptions options = new ChromeOptions();
options.setExperimentalOption("prefs", preference);
WebDriver driver = new ChromeDriver(options);
driver.manage().window().maximize();
driver.get("https://www.ajio.com/");
Thread.sleep(2000);
driver.close();
}
}
```

2) Website - mictests.com

```
package popups;
import java.util.HashMap;
import java.util.concurrent.TimeUnit;
import org.Openqa.selenium.By;
import org.Openqa.selenium.WebDriver;
import org.Openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.chrome.ChromeOptions;
public class HandleAnyNotification1 {
psvm(SC]a) throws InterruptedException {
HashMap <String, Integer> ContentSettings = new HashMap<>();
HashMap <String, Object> profile = new HashMap <>();
HashMap <string, Object> preference = new HashMap <>();
ContentSettings.put("media_stream", 1);
profile.put("managed_default_Content_settings", ContentSettings);
preference.put("profile", profile);
ChromeOptions.Options = new ChromeOptions();
options.setExperimentalOption("prefs", preference);
WebDriver driver = new ChromeDriver(Options);
driver.manage().window().maximize();
Driver.get("https://mictests.com/");
driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
driver.findElement(By.id("mic-launcher")).click();
Thread.sleep();
driver.close();
}
}
```

3. Website - Webcamtests.com

Same as above.

```
driver.get("https://webcamtests.com/");
driver.findElement(By.id("webcam-launcher")).click();
```

Chapter-3

5/5/23
9am

Data Driven Testing:-

. The process of driving the data into testScripts from external resources & performing test execution is called data driven Testing.

Why data driven Testing?-

As per the rule of automation testing data shouldnot be hardcoded also maintanence & modification of data becomes tough/difficult which is why we go for data driven testing.

Advantages

1. Avoids hardcoding
2. Maintanence & modification of data is easy.
3. Time saving
4. Data reusuability
5. Data sharing among team members &

We have 2 types of data - 1. Common data
2. Test data

Common data:-

The data which is common to all the test-Scripts.

Ex:- url, browser, timeouts, database url.

Test data:-

The data which is test-Script Specific.

Ex:- Naukri.com

RegistuTest

- fullname
- DoB
- Yop
- ~~Resume~~
- phone.no.
- Email

Properties file:-

- It is java featured file which is used to store data in the form of key, value pairs.
- Both key & value are in String format.
- properties file has .properties as extension.
- We store common data in properties file.
- It is faster to read data from properties file compared to any other file. bez java has it's own libraries to read data from properties file.

Ways to store data

```
1 url = https://www.google.com/
2 browser = chrome
3 time = 10
```

I (=)

```
1 url https://www.google.com/
2 browser chrome
3 time 10
```

II (giving one space)

```
1 url : https://www.google.com/
2 browser : chrome
3 time : 10
```

III (giving colon)

Website - demo.actitime.com.

```
package datadriven;

import java.io.FileInputStream;
import java.io.IOException;
import java.util.properties;

public class FetchDataFromProperties {

    psvm(sc[]args) throws IOException{
    //Step-1: Convert physical file into java readable file.
      //new FileInputStream(); throws FileNotFoundException
    FileInputStream fis = new FileInputStream("./testResources
                                                .prop
```

```
// Step 2: create an instance for properties Object.
    Properties property = new Properties();
// Step 3: load all key value pairs into properties object.
    // load() throws IOException.
    property.load(fis);
    // Step 4: fetch data to properties using key.
    property.put("usernam
    S.o.pln(property.getproperty("url"));
    S.o.pln(property.getproperty("browser"));
    S.o.pln(property.getproperty("username"));
    S.o.pln(property.getproperty("password"));
    S.o.pln(property.getproperty("time"));
  }
}
```

Java Concept used Internally in properties file:-

- Internally properties file utilizes map Interface Concept to load data to properties Object.
- When property.load(fis) method is called, it Internally creates a HashTable and stores all the data from fis into HashTable in the form of key-value pairs.
- HashTable is underlying datastructures of HashMap.

Drawbacks of properties file:-

1. We can fetch only single data at a time.
2. It doesnot allow duplicate keys
3. It is not organized.
4. When we have lot of data it is tedious to fetch particular key from the file.

Excel:-

In excel the data is stored in the form of tables.

Write to properties program:-

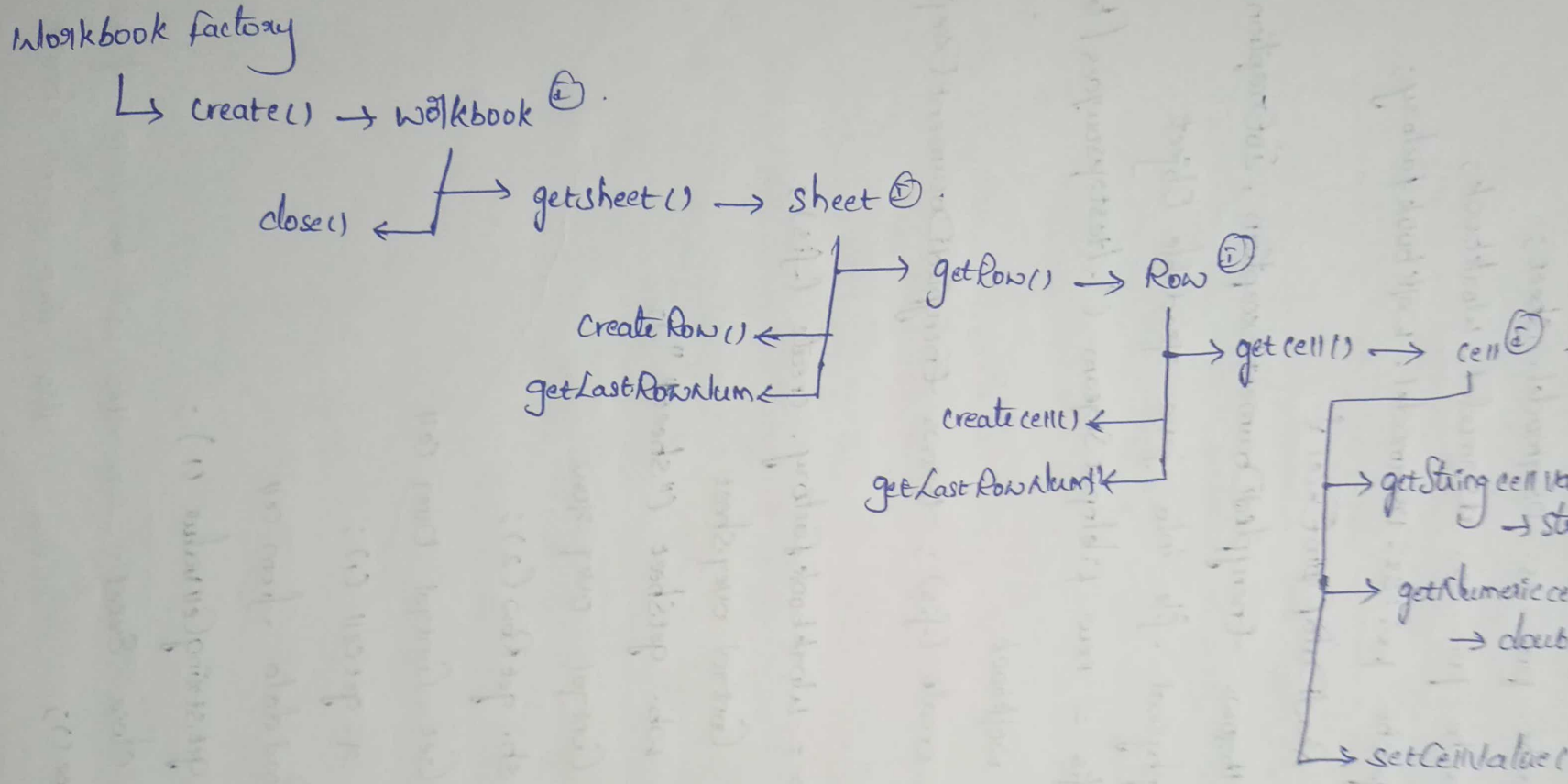
```
package datadriven;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.util.Properties;
public class WriteToProperties {
    psvm(String[] args) throws IOException {
// Step 1: Convert physical file into java readable file
//new FileInputStream(); throws FileNotFoundException
FileInputStream fis = new FileInputStream("./testresources/data.properties");
// Step 2: Create an instance for properties Object.
Properties property = new properties();
// Step 3: load all key value pairs into properties Object.
// load(); throws IOException.
property.load(fis);
// Step 4: Write data to properties
property.put("username2", "trainee");
property.put("password2", "trainee");
// Step 5: Save data to properties file
FileOutputStream fos = new FileOutputStream("./testresources/data.properties");
property.store(fos, "updated Successfully");
}
}
```

6/05/23
Saturday

```
package datadriven;

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.properties
import java.util.Concurrent.TimeUnit;

import org.openqa.Selenium.By;
import org.openqa.selenium.webDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.Support.ui.ExpectedConditions;
import org.openqa.selenium.Support.ui.WebDriverWait;

public class ActitimeLoginScript{

  psvm(S[] args) {
[// Step 1: Convert physical file into java readable file.
  // new FileInputStream();]
    WebDriver driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.get(readData("url"));
    long time = LongparseLong(readData("time"));
  driver.manage().timeOuts.implicitlyWait(time, TimeUnit.SECONDS);
  driver.findElement(By.id("username")).SendKeys(readData("username"));
  driver.findElement(By.name("pwd")).SendKeys(readData("password"));
  driver.findElement(By.id("loginButton")).click();
  WebDriverWait wait =new WebDriverWait(driver, time);
boolean status = wait.until(ExpectedConditions.titleContains
                                ("Enter Time-Track"));
```

```
if (status)
    s.o.pln ("pass");
else
    s.o.pln ("fail");
driver.close();
}
public static String readData (String key)
{
// Step 1: Convert physical file into java readable file.
// new FileInputStream(); Surrounded with try/catch block.
    FileInputStream fis = null;
    try{
        fis = new FileInputStream (".\\testresources\\data.properties");
    }
    catch (FileNotFoundException e) {
        e.printStackTrace();
    }
// Step 2: Create an instance for properties object
    Properties property = new Properties();
// Step 3: load all key value pairs into properties file
    // load(); surrounded with try/catch block.
    try {
        property.load(fis);
    }
    catch (IOException e)
    {
        e.printStackTrace();
    }
    return property.getProperty(key);
}
```

Apache poi class diagram



Excel

- We store data in tabular form in excel.
- It is well organised way to store the data in excel.
- Generally testdata is stored in excel.
- We use apache poi libraries to read data from excel.

8/5/2023
Monday

```
package datadriventesting;
import java.io.FileInputStream;
import java.io.IOException;
import org.apache.poi.EncryptedDocumentException;
import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.Row;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.ss.usermodel.WorkbookFactory;
  public class ReadDataFromExcel {
  psvm (SC[]a) throws EncryptedDocumentException, IOException {
// Step 1 :- Convert physical file into java readable Object.
  FileInputStream fis = new FileInputStream (".\\testdata\\testdata.xlsx");
// Step 2 :- Open Workbook
// WorkbookFactory.create(fis); throws EncryptedDocumentException,
    IOException.
  Workbook wb = WorkbookFactory.create(fis);
// Step 3 :- Get Control over sheet.
  Sheet sh = wb.getSheet("sheet1");
// Step 4 :- Get Control over row
  Row r = sh.getRow(2);
// step 5 :- Get Control over cell
  Cell c = r.getCell(1);
// Step 6 :- Read data from cell
  S.O.println(c.getStringCellValue());
// Step 7 :- close Excel ---> this will be imp, if we won't give
  wb.close();                this then Excel should Corrupted.
  }
}
```

```
package datadriventesting;
import java.io.FileOutputStream;
import java.io.IOException;
import org.apache.poi.EncryptedDocumentException;
import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.Row;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.ss.usermodel.WorkbookFactory;
public class WriteDataToExcel {
psvm(String[] args) throws EncryptedDocumentException, IOException {
// Step 1:- Convert physical file into java readable Object.
FileInputStream fis = new FileInputStream("./testresources/
                                          testData.xlsx");

//step 2:- Open workbook.
// WorkbookFactory.create(fis); throws EncryptedDocumentException,
   IOException.
   Workbook wb = WorkbookFactory.create(fis);
//step 3:- Get Control over sheet
   Sheet sh = wb.getSheet("Sheet1");

//Step 4:- Create new row
   Row r1 = sh.createRow(4);
//step 5:- Create new Cell
   Cell c = r1.createCell(0);
//Step 6:- Write to Cell
   c.setCellValue("Tool");
//step 7:- Save
FileInputStream fos = new FileOutputStream("./testresources/
                                          testData.xlsx");
```

```
        wb.write(fos);
    //step 8 :- close workbook
        wb.close();
    }
}
```

(3)

```
package datadriventesting;
import java.io.FileInputStream;
import java.io.IOException;
import org.apache.poi.EncryptedDocumentException;
import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.Row;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.ss.usermodel.WorkbookFactory;
public class WriteDataToExcel {
psvm(String[] a) throws EncryptedDocumentException, IOException {
//step-1:- Convert physical file into java readable Object
FileInputStream fis = new FileInputStream(".\\testresources\\
                                    testData.xlsx");
//step-2: Open workbook
// WorkbookFactory.create(fis); throws EncryptedDocumentException, IOException
Workbook wb = WorkbookFactory.create(fis);
//step-3:- Get Control Over sheet
Sheet sh = wb.getSheet("Sheet1");
//step-4:- Get row
Row r = sh.getRow(4);
//step-5:- Create new Cell
Cell c = r.createCell(1);
c.setCellValue("Selenium");
```

```
// step 6 :- Save
FileOutputStream fos = new FileOutputStream("./testresources/
                                    testData.xlsx");
    wb.write(fos);
//step 7 :- close workbook
    wb.close();

}

}
```

A)

```
package datadriventesting;
import org.apache.poi.EncryptedDocumentException;
import java.io.FileInputStream;
import java.io.IOException;
import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.Row;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.ss.usermodel.WorkbookFactory;
  public class DataFormatterPractice {
psvm(String[]a) throws EncryptedDocumentException, IOException{
//Step 1:- Convert physical file into java readable Object.
FileInputStream fis = new FileInputStream("./testresources/
                                    testdata..xlsx");
//Step 2 : Open Workbook
// WorkbookFactory.create(fis); throws EncreptedDocumentException,IOException
  Workbook wb = WorkbookFactory.create(fis);
//Step 3:- Get Control Over Sheet
  Sheet sh = wb.getSheet("Sheet1");
//Step 4:- Get Control Over row
    Row r = sh.getRow(3);
```

// Step 5 :- Get Control Over Cell

Cell c = s.getCell(1);

// Step 6 :- Read data from Cell

DataFormatter df = new DataFormatter();

String data = df.formatCellValue(c);

(3) S.o.pln(data);

// step 7 : close Excel

wb.close();

}

}

5) package datadriventesting;

import java.io.FileInputStream;

import java.io.IOException;

import org.apache.poi.EncryptedDocumentException;

import org.apache.poi.ss.usermodel.Cell;

import org.apache.poi.ss.usermodel.Row;

import org.apache.poi.ss.usermodel.sheet;

import org.apache.poi.ss.usermodel.Workbook;

import org.apache.poi.ss.Usermodel.WorkbookFactory;

public class ~~Read~~ ActitimeLoginUsingExcel {

psvm(s[]a){

Map<String, String> map = readfromExcel();

WebDriver driver = new ChromeDriver();

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

driver.get(map.get("url"));

long time = Long.parseLong(map.get("time"));

driver.manage().timeouts().ImplicitlyWait(10, TimeUnit.SECONDS);

driver.findElement(By.id("username")).SendKeys(map.get("username

```
driver.findElement(By.name("pwd")).sendKeys(map.get("password"));
driver.findElement(By.id("loginButton")).click();
WebDriverWait wait = new WebDriverWait(driver,time);
wait.until(ExpectedConditions.titleContains("Enter Time-Track"));
    driver.close();
}
public static Map<String, String> readFromExcel() {
    FileInputStream fis = null;
    try {
    fis = new FileInputStream("./testresources/testData.xlsx");
    }
    catch (FileNotFoundException e){
        e.printStackTrace();
}
    Workbook wb = null;
    try {
      wb= WorkbookFactory.create(fis);
}
catch (EncryptedDocumentException |IOException e){
    e.printStackTrace();
}
  Sheet sheet = wb.getSheet("Sheet1");
Map<String,String> map = new HashMap();
DataFormatter df = new DataFormatter();
for(int i=0; i<=sheet.getLastRowNum(); i++){
  String key = df.formatCellValue(sheet.getRow(i).getCell(0));
  String value = df.formatCellValue(sheet.getRow(i).getCell(1));
    map.put(key, value);
}
```

```
        S.o.p (map);
        return map;
    }
}
```

9/5/2023 **POM - Page Object Model**

- POM stands for page Object Model
- It is java design pattern preffered by google Company to develop object repository.

Object Repository:-

It is the Collection of elements their locators and respective business libraries in One place.

Why POM?

In Agile Methodogy, due to frequent requirement changes - GUI also changes which effects the elements also modifying the locators becomes tedious job, Hence we go for POM.

- In POM elements are stored in pagewise.

Advantages of POM

1. Maintainance & modification of elements is easy.
2. Test Script development is faster.
3. Increased code readability.
4. Reusability of libraries.
5. Debugging is easy.
6. Handles StaleElementReferenceException.
7. Test Script Optimization is achieved.
8. No need for external libraries.

** Steps to achieve / develop pom class

1. Declaration :-

@FindBy (LocatorName = "Locator Value")

private WebElement / List < WebElement > elementRef ;

@FindBy

. It is Selenium Annotation.

. It is used to fetch WebElement of list < WebElement > using one locator as well as declares it as private.

2. Initialization :-

public classname (WebDriver driver) → Constructor

{

pagefactory.initElements (driver, this);

}

→ StaleElementReferenceException Handled here.

old (or) Expired. Stale Element Reference Exception :-

It is Selenium Exception which occurs when we access an Element using old or expired address.

→ In this page, the driver is initialized to Current webpage address which avoids StaleElementReference Exception.

3) Utilization :-

Business libraries are defined here.

Loginpage

```
package pom;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.findBy;
import org.openqa.selenium.support.pagefactory;

public class Loginpage {
// Declaration
@findBy (className = "atLogaimg").
  private WebElement logo;
@findBy (id = "username")
  private WebElement usernameTF;
@findBy (name = "pwd")
  private WebElement pwdTF;
@findBy (id = "loginButton")
  private WebElement loginButton;
@findBy (id = "keepLoggedInCheckBox")
  private WebElement keepMeLoggedInCB;
@findBy (partialLinkText = "forgot")
  private WebElement forgotpwdlink;
@findBy (xpath = "//span[contains(text(), 'username or password')]")
  private WebElement errorMsg;
// Initilization
public WebElement getLogo() {
    return logo;
}
public Loginpage (WebDriver driver) {
    pagefactory.initElements (driver, this);
}
```

// Utilization :-

```
public WebElement getLogo() {
    return logo;
}
public void setUsername(String username);
    usernameTF.sendKeys(username);
}
public void setPassword(String password);
    pwdTF.sendKeys(password);
}
public WebElement getLoginButton() {
    return loginButton;
}
public void clickLogin() {
    loginButton.click();
}
public void clickKeepMeLoggedIn() {
    keepMeLoggedInCB.click();
}
public void clickforgotpwdLink() {
    forgotpwdLink.click();
}
public String getErrorMsg() {
    return errorMsg.getText();
}
}
```

Login Test

```
package pom;
import java.util.Map;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
```

```
import datadriven.ActitimeLoginUsingExcel;
public class LoginTest {
    public static void main (String[] args) {
    Map<String, String> map = ActitimeLoginUsingExcel.readFromExcel();
     WebDriver driver = new ChromeDriver();
     driver.manage().window().maximize();
     driver.get (map.get ("url"));
      long time = Long.parseLong (map.get ("time"));
     driver.manage().timeouts().implicitlyWait (time, TimeUnit.SECONDS);
      LoginPage login = new LoginPage (driver);
     if (login.getLogo1().isDisplayed())
       S.o.pln (" Login page is Displayed");
   • else
       S.o.pln (" Login page is not found");
     login.setUsername1 (map.get ("username"));
     login.setPassword1 (map.get ("password"));
     login.clickKeepMeLoggedIn1();
     login.clickLogin1();
    WebDriverWait wait = new WebDriverWait (driver, time);
     wait.until (ExpectedConditions.titleContains ("-Enter Time-Track"));
      driver.close();
  }
}
```

01/5/23 Thurs

TestNG

Test Next Generation.

- It is an open source unit testing framework tool used by both developers & automation engineers
- Developers use it for white box testing & automation test engineers use it to develop framework for testSuite execution.
- TestNG is developed as separate plugin for IDE's. (Integrate development Environments)
- TestNG is headless tool, IDE's provide userInterface for TestNG.
- It is the Combination of JUnit & NUnit + Advanced features.

  JUnit → Unit testing framework tool for java.

  NUnit → Unit testing framework tool for .net.

Features of TestNG:-

1. prioritizing testCases.
2. Rerunning the same testScripts multiple times.
3. Disabling the testScripts.
4. Create dependency b/w testScripts.
5. Control test execution flow using Annotations.
6. Batch Execution
7. Group Execution.
8. parallel Execution
9. Reports
10. Assertions

Annotation Test :- (@Test) :-

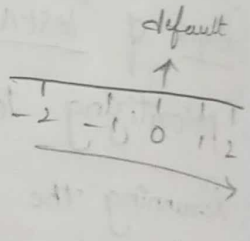
- It is the driving factor in TestNG.
- It acts like main method in java.
- One @Test Method is equal to one test Script.
- In a class we can have more than one @Test methods.
- In real-time maximum 15 @Test methods are given in a class.

1. priority :-

1. It is used to prioritized the testCases
2. Usuage

```
@Test (priority = int)
public void demo()
{
  //test Script
}
```

3. priority follows Integer no. line order of execution.



4. Default priority is zero.
5. If the priority of testCases is same it Considers ascii values of method names.

* Invocation Count

- It is used to re-run the test Script multiple times.
- Usuage

```
@Test (invocationCount = Int)
public void demo(){
  // test Script
}
```

- Default Invocation Count is 1.
- If the Invocation Count is zero or -ve number it will Consider the test Script in execution queue, but it willnot execute.

priority Ex:-

```
package testng;
import org.testing.Reporter;
import org.testing.annotations.Test;
public class prioritizeTestCases {
@Test
public void demo1 () {
  Reporter.log ("demo1", true);
}
@Test (priority = -1)
  public void demo2 () {
  Reporter.log ("demo2", true);
}
@Test (priority =1)
public void demo3 () {
  Reporter.log ("demo3", true);
}
@Test (priority = -1)
  public void demo4 () {
  Reporter.log ("demo4", true);
}
}
```

o/p:- demo2
demo4
demo1
demo3.

InvocationCount

- (It is used to return the @testScript multiple times).

→

```
package testing;
import org.testing.Reporter;
import org.testing.annotations.Test;
public class InvocationCount {
@Test (invocationCount =0)
```

```
public void demo1 (){
  Reporter.log ("demo1", true);
}

@Test (invocationCount = 1)
public void demo2 (){
  Reporter.log ("demo2", true);
}
@Test (invocationCount = 3)
  public void demo3() {
  Reporter.log ("demo3", true);
}
@Test (invocationCount = -1)
  public void demo4 (){
  Reporter.log ("demo4", true);
}
@Test (invocationCount = 1)
  public void demo5 (){
  Reporter.log ("demo5", true);
}
}
```

12/5/23 Friday

Enabled

- In order to disabled the testScript we use enabled parameter.
- Usage

```
@Test (enabled = false)
  public void demo()
  {
    // test script
  }
```

- By default enabled = true.

```
package testng;
import org.testng.annotations.Test;
public class DisablingTestScript {
@Test(enabled = false)
public void demo(){
  S.o.pln("demo");

}
}
```

Combination of priority, InvocationCount, enabled

```
package testng;
import org.testng.annotations.Test;
public class priorityInvocationEnabled {
@Test(priority = -1, invocationCount = 2)
  public void testcase1(){
    S.o.pln("testCase 1");

}
@Test(priority = 1, invocationCount = 0)
  public void testCase 2(){
    S.o.pln("test case 2");
}
@Test(priority = 2, enabled = false)
  public void testCase 3(){
    S.O.pln("test case 3");
}
@Test(priority = -1, enabled = false)
public void testCase 4(){
    S.o.pln("test case 4");
}
@Test(priority = +1, invocationCount = 1)
  public void testCase 5(){
    S.o.pln("testCase 5");
}
```

op:- Test case 1
" 1
Test Case 5

4) dependsOnMethods :-

- It is used to create dependency blw the methods.
- Usage

```
@Test
public void tc1(){
//testScript
}
@Test (dependsOnMethods = "tc1")
public void tc2(){
// testScript
}
```

more than one dependency

```
@Test (dependsOnMethods
= {  ,  }).
```

→

```
package testng;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.Keys;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.annotations.Test;
public class DependsOnMethods{
WebDriver driver;
@Test
public void navigateToActitime(){
driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://demo.actitime.com/login.do");
WebElement logo = driver.findElement(By.className("atLogoImg"));
if(logo.isDisplayed())
S.o.println("pass");
else
```

```
        s.o.pln ("fail");
        driver.close();
    }
    @Test (dependsOnMethods = "navigateToActitime")
    public void loginToActitime () throws InterruptedException {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get ("https://demo.actitime.com/login.do");
        driver.manage().timeouts().implicitlyWait (10, TimeUnit.SECONDS);
        driver.findElement (By.id ("username")).sendKeys ("admin" + Keys.TAB +
                                                    "manager" + Keys.ENTER);
        Thread.sleep (3000);
        driver.close();
    }
}
```

5) Annotations :-

. Annotations are the metadata provided to JVM to perform some functions / task.

. In TestNG annotations are used for control test execution flow.

* @ BeforeSuite

. It is executed before <Suite>, here we give actions such as establishing database connections & configuration settings.

. It is executed only once per execution of xml file.

* @ BeforeTest

@Before - preCondition
@Test - Actual test Script
@After - post Condition

. It is executed before <test> in Xml.

. It is generally used for parallel execution (SimultaneousExecution).

. No. of times of execution of annotation before Test depends on no. of <test>

* @ BeforeClass

. It is executed before each Test class. / <class>

. We generally give functions such as launching the browser

.

- The no. of times of execution of @BeforeClass = no. of testclass

* @BeforeMethod
  - it executes before each & every @TestMethod.
  - Here we give functions such as login to the application,
  - The no. of times of execution of @BeforeMethod = no. of @Test methods
* @AfterTestMethod → @Test - we already written in front page.
  - it executes after each and every @TestMethod.
  - Here, we give functions such as logout to the application.
  - The no. of times of execution of @AfterMethod = no. of @TestMethods.
* @AfterClass
  - it is executed after each </class>
  - We generally give functions such as launching the browser.
* @AfterTest
  - it is executed after </test> in xml.
  - it is generally used for parallel executions
  - No. of times of execution of @AfterTest depends on no. of </test> tags
* @AfterSuite
  - it is executed After</suite>
  - Here we give actions such as establishing database Connections and Configuration settings.
  - it is executed only once per execution of xml. file.

Example:-

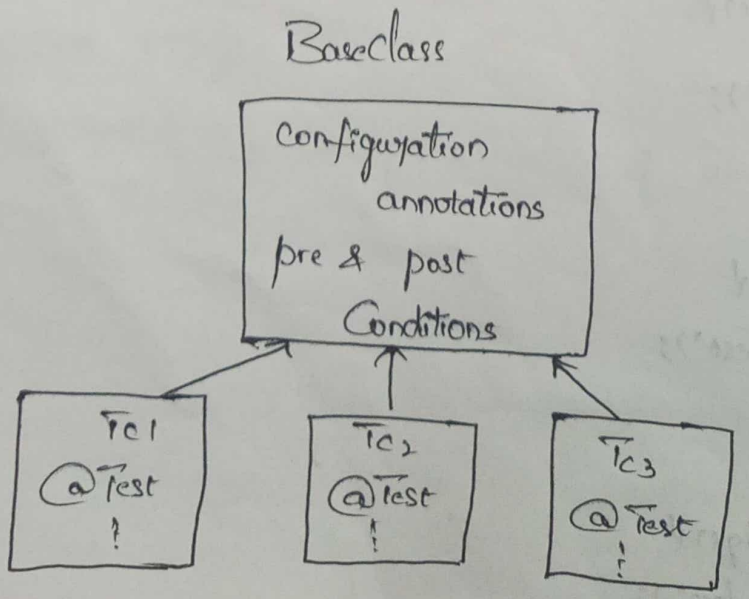
```
package testng;
import org.testng.Reporter;
import org.testng.annotations.Test;
public class Testclass1 extends Baseclass{
@Test
public void tc1(){
  Reporter.log("@Test", true);
}
}
```

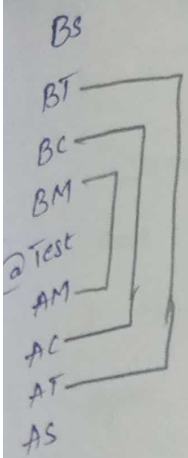
```
package testng;
import org.testng.annotations.AfterClass;
import org.testng.annotations.AfterMethod;
import org.testng.annotations.AfterSuite;
import org.testng.annotations.AfterTest;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.BeforeSuite;
import org.testng.annotations.BeforeTest;
public class Baseclass{
@BeforeTestSuite
public void SuiteConfig(){
   S.o.println("@BeforeSuite");
}
@BeforeTest
public void TestConfig(){
  S.o.println("@BeforeTest");
}
@BeforeClass
public void classConfig(){
} S.o.println("@Beforeclass");
```

```
@BeforeMethod
public void methodConfig(){
    S.o.pln("@BeforeMethod");
}

@AfterMethod
public void methodTeayDown(){
    S.o.pln("@AfterMethod");
}

@Afterclass
public void methodTeayDown(){
    S.o.pln("@Afterclass");
}

@AfterTest
public void testTeayDown(){
Sopln("@ AfterTest");
}
@AfterSuite
public void suiteTeayDown(){
S.o.pln("@ AfterSuite");
}
}
```

Base Class :-

- it Consists of all Configuration annotations.
- there will be only one Baseclass for project.
- All the testclasses extends to baseclass.
- Java Concept involved here is Hierarichal Inheritance.





@Test(2) – 1 class.

BS
BT
BC
BM
@Test
AM
BM
@Test
AM
AC
AT
AS

2 classes
1 @Test in each

BS
BT
BC
BM
@Test
AM
AC
BC
BM
@Test
AM
AC
AT
AS

2 <test>
1 class & 1 @Test in each

BS
BT
BC
BM
@Test
AM
AC
AT
BT
BC
BM
@Test
AM
AC
AT
AS

6) Batch Execution

- Executing all the testScripts sequentially is called Batch Execution.

Steps

1. Convert all the testclasses into one xml file.
2. Execute the xml file.

Group Execution:-

- Grouping similar testCases & performing execution of Specific group is called group execution.

Steps

1. Specify the group for each & every @Test method.

```
@Test(groups = "Smoke")
public void test1(){
}

@Test(groups = {"Smoke", "Sanity"})
public void test2()
{
}
```

2. Convert all the testclasses into xml file.

3. Include & exclude the groups from the execution, after Sca

```
<suite>
  <groups>
    <run>
      <include name = "smoke" />
      <exclude name = "system" />
    </run>
  </groups>
  <test>
    <classes>
      <class -- />
      :
    </classes>
  </test>
</suite>
```

Batch Execution Example:-

```
package batchexecution;
import org.testing.annotations.Test;
public class TestClass1 {
@Test
 public void Test1() {
   S.o.pln ("Testclass1 ⇒ test1");
 }
 @Test
 public void Test2() {
    S.o.pln ("Testclass2 ⇒ test2");
  }
}
package batchexecution;
import org.testing.annotations.Test;
 public class TestClass2 {
 @Test
 public void Test2() {
   S.o.pln ("Testclass2 ⇒ test2");
 }
 @Test
```

```
public void Test2(){
    S.O.pln ("Testclass 2 => test 2");
}
}
```

Same as 6 Testclasses

then right click on package name go to TestNG then Convert to TestNG & change location testng to batch.xml then click on finish. double click on ~~open~~ batch.xml then run the xml file.

GroupExecution:-

```
package groupexecution;
import org.testng.annotations.Test;
public class Test1{
@Test (groups = "Smoke")
public void tc1(){
    S.O.pln ("test 1 - Smoke");
}
@Test (groups = "system")
public void tc2(){
    S.O.pln ("test 1 - system and regression");
}
@Test (groups = "Sanity")
public void tc3(){
    S.O.pln ("test 1 - Sanity and Smoke");
}
}
```

```
package groupexecution;
import org.testng.annotations.Test;
public class Test2{
@Test (groups = "Regression"):
public void tc1(){
    S.O.pln ("test 2 - regression and Sanity");
}
```

```
@Test (groups = "system")
public void test2() {
    S.o.pln ("test 2 - System");
}
@Test (groups = "sanity")
public void test3() {
    S.o.pln ("test 2 - sanity");
}
}
```

like as above 3 testclass

then right click on package name go to TestNG then Convert to TestNG then change location testing to group.xml. then click on finish. then double click on group.xml then add some points below <suite> tag like <groups>, <run> <include> in this we <include> whatever we want like string then run the group.xml file it gives the output.

15/5/23 Monday — Installation maven & Jenkins

18/5/23 Thurs

Parallel Execution:-

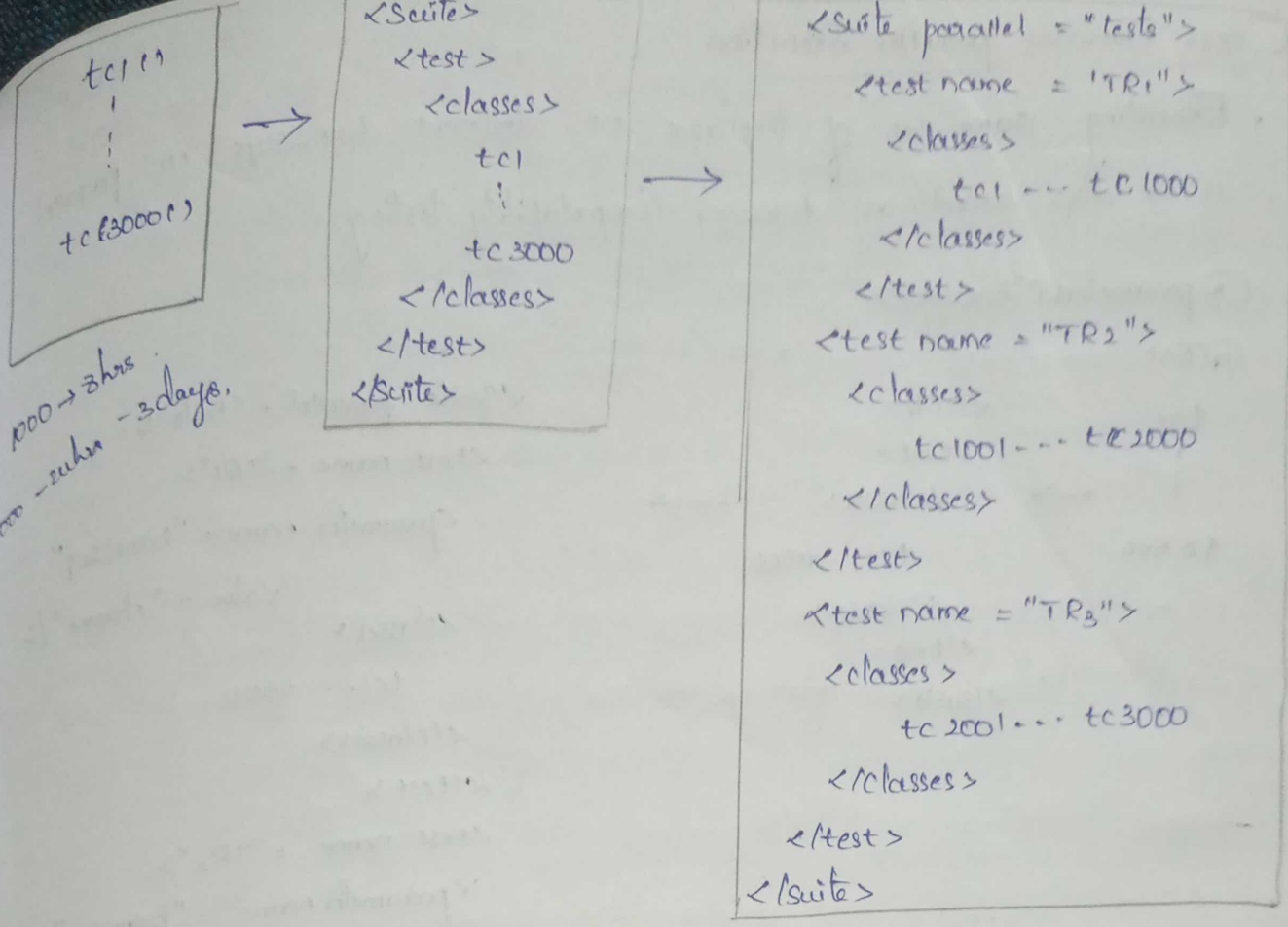
- Executing testcases simultaneously (or) in parallel is called parallel execution.

Types of parallel Execution:-

1. Distributed parallel execution
2. Cross Browser parallel Execution
3. cross platform parallel Execution

1. Distributed parallel Execution:-

Distributing the test cases into different test runners & executing them in parallel.



https://testng.org/testng-1.0.dtd. (doctype)

```
<?xml version = "1.0" encoding = "UTF-8"?>
<!DOCTYPE Suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name = "Suite" parallel = "tests">
  <test thread-count = "5" name = "Test 1">
    <classes>
      <class name = "batchexecution.TestClass1"/>
      <class name = "batchexecution.Testclass2"/>
    </classes>
  </test> <!--Test-->
  <test thread-count = "5" name = "Test 2">
    <classes>
      <class name = "batchexecution.Testclass3"/>
      <class name = "batchexecution.Testclass4"/>
    </classes>
  </test> <!--Test-->
  <test thread-count = "5" name = "Test 3">
    <classes>
      <class name = "batchexecution.Testclass5"/>
      <class name = "batchexecution.Testclass6"/>
    </classes>
  </test> <!--Test-->
</suite> <!--Suite-->
```

Cross Browsers parallel Execution

- Executing same set of testCases on different browsers in parallel
- It is also called as browsers Compatability testing.

```
@parameters("browser")
@Test
tc1
 :
tc500
```

→

```
<suite>
 <test>
  <classes>
   tc1 -- tc500
  </classes>
 </test>
</suite>
```

→

```
<suite parallel = "tests">
 <test name = "TR1">
  <parameter name = "browsers"
             value = "chrome"/>
  <classes>
   tc1 -- tc500
  </classes>
 </test>
 <test name = "TR2">
  <parameter name = "browser"
             value = "fire-fox"/>
  <classes>
   tc1 -- tc500
  </classes>
 </test>
 <test name = "TR3">
  <parameter name = "browser"
             value = "edge"/>
  <classes>
   tc1 -- tc500
  </classes>
 </test>
</suite>
```

```
package crossBrowserparallel;
import org.openqa.selenium.WebDriver;
public class Baseclass {
    protected WebDriver driver;
    @parameters ("BROWSER")
```

```
@BeforeClass
public void classConfig (String browser) {
  switch (browser) {
    case "chrome":
      driver = new ChromeDriver();
      break;
    case "firefox":
      driver = new FirefoxDriver();
      break;
    case "edge":
      driver = new
      System.setProperty ("webdriver.edge.driver", "./msedgedriver.exe");
      driver = new EdgeDriver();
      break;
    default:
      System.out.println ("invalid browser info");
  }
  driver.manage().window().maximize();
}
@AfterClass
public void classTearDown() {
  driver.close();
}
}
```

→
```
package crossBrowserParallel;
import org.testng.annotations.Test;
public class Googletest extends Baseclass
{
  @Test
  public void google() throws InterruptedException.
    driver.get("https://www.google.com/");
    Thread.sleep(2000);
  }
}
```

```
package crossBrowserparallel;
import org.testng.annotations.Test;
public class Swiggy extends Baseclass{
@Test
  public void Swiggy() throws InterruptedException{
    driver.get("https://www.swiggy.com/");
    Thread.sleep(3000);
  }
}
```

Now select googletest & Swiggy class file right click go to TestNG Convert to testng. now modify the script like add parameters and then run.

```
https://testng.org/testng-1.0.dtd (doctype)
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="Suite" parallel="tests">
  <test thread-count="5" name="Test1">
    <parameter name="BROWSER" value="chrome"></parameter>
    <classes>
      <class name="crossBrowserparallel.Googletest" />
      <class name="crossBrowserparallel.Swiggy" />
    </classes>
  </test><!--Test-->
  <test thread-count="5" name="Test2">
    <parameter name="BROWSER" value="firefox"></parameter>
    <classes>
      <class name="crossBrowserparallel.Googletest" />
      <class name="crossBrowserparallel.Swiggy" />
    </classes>
  </test><!--Test-->
```

```
<test thread-count = "5" name = "Test 3">
    <parameters name = "BROWSER" value = "edge"></parameters>
    <classes>
        <class name = "crossBrowserparallel.GoogleTest"/>
        <class name = "crossBrowserparallel.Swiggy"/>
    </classes>
</test> <!-- Test -->
</Suite> <!-- Suite -->
```

Reports

TestNG has the a feature to generate HTML reports automatically when xml file is executed.

Steps to View report:-

1. run.xml file.
2. refresh the project
3. Go to testoutput folder then open emailable-report.html file with web browser.

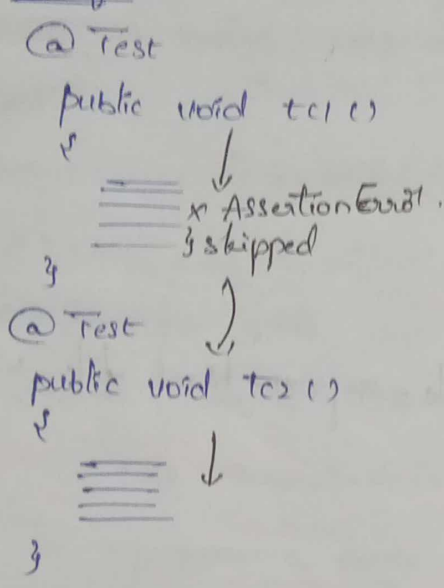
Assertions

- TestNG has a feature called assertions to validate test Scripts.
- We have 2 type of assertions - 1) Hardassertion
  2) Softassertion.

1. Hardassertion :-

- Hardassert can be given using the class Assert.
- Assert© is present in org.testNG.Assert package.

Workflow



- whenever hardassert is given if the validation fails in the middle of the method it throws assertionError immediately & skipps the remaining statements in the block & transveys the control to the next method.
- Assert© has all static methods

```
assertEquals (act, exp);
assertNotEquals (act, exp);
assert True (Condition);
assertfail (Condition);
assertNull ();
assertNotNull ();
fail ();
```

2. Softassertion

- Softassertion can be given using the class SoftAssert.
- SoftAssert class is present in org.testNG.Assert package.

Workflow

```
@Test
public void tc1()
{
    ≡ ⊗
    ≡ ↓
}  ——— AssertionError.

@Test
public void tc2()
{
    ≡ ↓
}
```

- Whenever SoftAssert is given if the validation fails it will continue with the execution of remaining statements in the method & throws assertionError at the end of the block.
- SoftAssert © has all NonStatic Methods.

```
assertEquals (act, exp);
assertNotEquals (act, exp);
assertTrue (condition);
assertFail (condition);
assertNull ();
assertNotNull ();
fail ();
```

assertAll(); — It is a mandatory method should be given at the end of the block.

```
package assertions;
import org.testng.annotations.Test;
public class IfElseStmt{
@Test
public void demo(){
String s1 = "Hello";
String s2 = "hello";
if (s1.equals(s2))
{
S.O.pln("pass");
}
else
{
S.O.pln("fail");
}
}
}
```

HardAssertion:-

```
package assertions;
import org.testng.Assert;
import org.testng.annotations.Test;
public class Hardassertion{
@Test
public void test(){
String s1 = "Hello";
String s2 = "hello";
Assert.assertEquals(s1, s2);
S.O.pln(s1);
S.O.pln(s2);
}
@Test
public void test2()
{
S.O.pln("test2");
}
}
```

O/p:- passed : test 2
Failed : test 1

SoftAssertion

```
package assertions;
import org.testng.Assert;
import org.testng.annotations.Test;
import org.testng.asserts.SoftAssert;
public class SoftAssertion{
@Test
public void test1(){
SoftAssert soft = new SoftAssert();
    String s1 = "hi";
    String s2 = "hlo";
     soft.assertEquals(s1,s2);
       s.o.pln(s1);
       s.o.pln(s2);
       soft.assertAll();
}
@Test
  public void test2(){
    s.o.pln("test 2");
}
}
```

o/p:- hi
hlo
test 2
passed : test 2
failed : test 1

Listeners

- Listeners are advanced features in testNG which monitors actual test execution and Captures runtime events like pass, fail or skipped status & performs necessary actions.

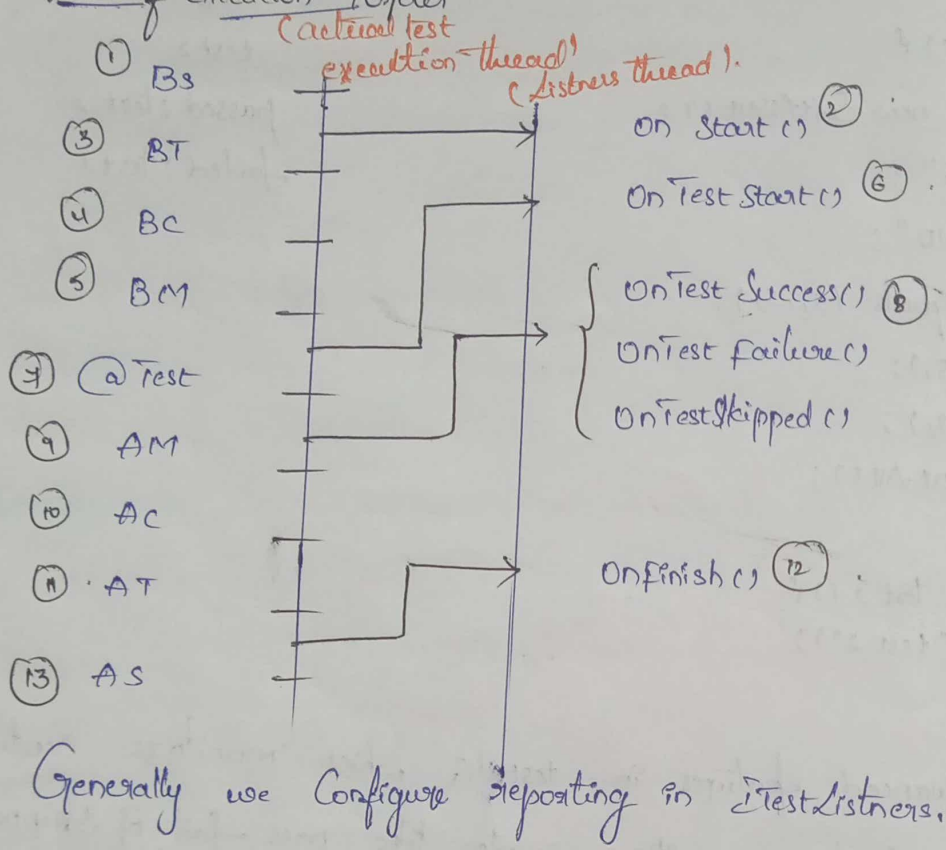
Interfaces in testNG

- ITestListner
- ITestResult
- ITest
- IRetryAnalyser

- ITestListner :-
  - It is an interface in testNG which monitors the test execution & performs necessary actions.
  - It has Several abstract methods.

- onStart()
- onTestStart()
- onTestSuccess()
- onTestFailure()
- onTestSkipped()
- onFinish()

Flow of execution / Order



- Generally we Configure reporting in ITestListners.

2) ITestResult :-

- it Captures the result of the testCase.
- it is used as argument for most of the methods in ITestListner (i).

→ package listners ;

import org.testng.annotations.AfterClass;
import org.testng.annotations.AfterMethods;
import org.testng.annotations.AfterSuite;
import org.testng.annotations.AfterTest;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.BefoleMethod;
import org.testng.annotations.BeforeSuite;
import org.testng.annotations.BefoleTest;

```
public class Baseclass {
@BeforeSuite
public void suiteConfig() {
  s.o.p("BeforeSuite");

}
@BeforeTest
public void testConfig() {
  s.o.pln("Before test");

}
@Beforeclass
public void classConfig() {
  s.o.pln("Before class");

}
@BeforeMethod
public void methodConfig() {
  s.o.pln("Before method");

}
@AfterMethod
public void methodTearDown() {
  s.o.pln("AfterMethod");

}
@Afterclass
public void classtearDown() {
  s.o.pln("After class");

}
@AfterTest
public void testtearDown() {
  s.o.pln("After Test");

}
@AfterSuite
public void suitetearDown() {
  s.o.pln("After suite");

}
}
```

Op:-

→ package listeners;

import org.testng.annotations.Listeners;

import org.testng.annotations.Test;

@Listeners (Listeners.ListenerImplementation.class)

public class TestScript extends Baseclass{

@Test

public void test(){

S.o.pln ("test");

}

}

→ package listeners;

import org.testng.ITestContext;

import org.testng.ITestListner;

import org.testng.ITestResult;

public class ListnerImplementation implements ITestListener{

@Override

public void onStart (ITest Context){

S.o.pln (" on Start");

}

@override

public void onTestStart (ITestResult result){

S.o.pln (" on test Start");

}

@override

public void onTest Success (ITestResult result){

S.o.pln (" on test Success");

}

@override

public void onTest failure (ITestResult result){

S.o.pln (" on test failure");

}

@override

public void onTest Skipped (ITestResult result){

S.o.pln (" on test Skipped");

}

```
@Override
public void onFinish(ITestContext Context) {
    S.O.println("onFinish");

}
}
```

Rerunning failed test scripts

Steps

- Execute .xml file and refresh the project.
- Open test-output folder & open testNG-failed.xml file.
- Fix the errors in the respective methods given in testNG-failed.xml & execute it.

IRetryAnalyser:-

it is an functional interface.
having 1 interface abstract method.

- It is an interface in testNG which is used to re-run the failed testScripts automatically.
- TestScript failure might occur due to network issues, Server issues, Synchronization issues, but these testscripts might pass when we Re-run it for 2 to 5 times.
- We use IRetryAnalyser for this kind of testScripts.

```
package retryAnalyzer;
import org.testng.Assert;
import org.testng.annotations.Test;
public class TestScript {
    @Test
    public void test1(){
        S.o.pln("Test 1");
    }
@Test(retryAnalyzer = retryAnalyzer.RetryAnalyzerImp.class)
    public void test2(){
        S.o.pln("Test 2");
        Assert.fail();
    }
}
```

If we have doubt, it will fail then we pass arguments here
Right click copy qualified name

→

```
package retryAnalyzer;
import org.testng.IRetryAnalyzer;
import org.testng.ITestResult;
public class RetryAnalyzerImp implements IRetryAnalyzer{
        int Count;
        int maxRetries = 4;
    @Override
        public boolean retry(ITestResult result){
            if(Count < maxRetries){
                Count++;
                return true;
            }
```

O/p:-
Test 1
Test 2
Test 2
Test 2
Test 2
Test 2

return false;
}
}

* If we have more than one @Test → we need not give multiple times listeners implementation → directly we give the / convert Test script into testNG there we give it give. xml file. Open .xml file add <listeners> tiles.

<listners>
<listener class_name = " " > </listners>.
</listeners>

this will give after the open. Suite tag.
</suite>

@DataProvider :-

- It is a testNG feature which is used to Re-run a test script multiple times with different data.
- We use dataproviders in datadriven frameworks
- In dataproviders multiple sets of data is stored in twodimensional object (Object[][]) array.
- Dataprovider method always returns 2 dimensional object array.
- The Dataprovider method should be given in the same class where @TestMethod is given.
- Object [2] [2] → no. of arguments in each set.
  ↓
  no. of datasets.
- @Test (dataprovider = "data")
  public void tc1 (String data1, String data2)
  {
  ==
  }
  @Dataprovider.
  public Object [][] data ()
  {
  Object [][] obj = new Object [2][2];

```
Obj[0][0] = "  ";
obj[0][1] = "  ";
obj[1][0] = "  ";
obj[1][1] = "  ";
    return obj;
}
```

Ex:-

```
package testng;
import org.testng.annotations.DataProvider;
import org.testng.annotations.Test;
public class Dataproviderpractice{
@Test (dataprovider Test (String src, String dest){
    S.o.pln ("from: "+src+"\t"+"To: "+dest);
}
@ Dataprovider.
public Object [][] dataforTickets (){
    Object [][] obj = new Object [3][2];
    obj [0][0] = "Bangalore";
    obj [0][1] = "Mumbai";
    obj [1][0] = "Bangalore";
    obj [1][1] = "Hyderabad";
    obj [2][0] = "Bangalore";
    obj [2][1] = "Vizag";
    return obj;
}
}
```

Chapter-9

Framework

- It is the well organised structured Collection of reusable Components such as classes, methods & object repositories. which enables faster testscript development & test execution easier & report generation.

Stages of Framework development :-

1. Design
   - Add plugins & dependencies in pom.xml. -(project Object model)
   - Add Testdata template in src/test/resources
     - properties file
     - Excel file
   - Develop Generic libraries in src/main/java.
     - custom IConstantpath (I) → to store external file paths.
     - propertiesUtility (C) → to store reusable methods of properties file.
     - ExcelUtility (C) → to store reusable methods of Excel file.
     - WebDriverUtility (C) → to store reusable methods of WebDriver actions
     - JavaUtility (C).
     - Baseclass (C) → pre& post Conditions.

2) Development :-
   - pom classes ↔ src/main/java
   - Test Scripts → src/test/java

3. Execution :-
   - → xml files
     Github
   - → Jenkins

Src/test/resources

In these we can right click on these go to properties & change the path to CommonData.properties.

In that we can give. browser = chrome
url = https://skillrary.com/
time = 10.

* Open Excel & give the
full name	varalakshmi
Email	neeli@gmail.com.
Subject	regarding testing Courses
Message	Need details regarding testing Courses

& the save the TestData.

Src/Main/java :-

We can create pkg/package like Generic Libraries. In that we create Interface. & also create another class like propertiesUtility in this we can create Instance, .fileInputStream etc. & property file.

→
```
package genericLibraries;
public interface IConstantpath{
    String PROPERTIES_PATH = ".\src\test\resources\CommonData.properties";
    String EXCEL_PATH = ".\src\test\resources\Testdata.xlsx";
```

```
package genericLibraries;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.Properties;
  public class propertiesUtility {
    private properties property;
      public void propertiesInit (String filepath) {
        FileInputStream fis = null;
      try {
        fis = new FileInputStream (filepath);
      }
```

```
catch (FileNotFoundException e) {
    e.printStackTrace();
}
property = new Properties();
try {
    property.load(fis);
} catch (IOException e) {
    e.printStackTrace();
}
}
public String readDataFromProperty(String key) {
    return property.getProperty(key);
}
}
```

29/5/23
Mon

→ Convert all testcases into testNG.xml file.

- pom.xml changes:
  - remove <scope>test</scope> from testNG dependency.
  - In surefire plugin change version 3.00-M8
  - After version we have to add, see

```
<configuration>
    <suiteXmlFiles>
        <suiteXmlFile>testNG.xml</suiteXmlFile>
    </suiteXmlFiles>
</configuration>
```

- Save pom.xml & update maven project.
  - right click on project.
  - maven → update project.
- Select force update checkbox
  - click on ok.

Steps to create repository in Github

- Open Github.com & Sigin to ur account.

Token generations steps

1. click on the profile icon on top right Corner.
2. click on settings.
3. Scroll down & click on Developer Settings in the left side bar.
4. click on personal access tokens ⟹ tokens (classics)
5. Now click on generate new token → Generate new token (classic)
6. provide the Note - HCSE10 then click on repo checkbox. next scroll down click on generate token.
7. Copy the Token Save it in Notepad.
8. Token generation is done.

Steps to create cloud. repository

→ . click on + button on top right Corner.

- click new repository & give repository name (skillbay (CHCSE10) & click on create repository.

Steps to push the framework to Github repository

- Right click on project → Team → share project.
- Select use of create repository check box.
- Select the project. → click on create repository & Finish.
- Right click on the project again → Team → Commit.
- we can see git staging beside Console.
- click on double ++icon
- ~~Right~~ write Commit message
  Ex:- First Commit
- click on Commit
- click on push HEAD.

. Now go to Github repository Copy repository url.

. paste in it Eclipse URI. https://

Note:- Once after pasting URI host & repository path will be auto filled.

. give github user name & token as password.

. click on preview

. again click on preview.

. click on push

. Give github user name & Github token as password & click on login

. click on close.

. Go to github refresh the page.

. click on close.
. Go to github refresh the page.

Github

. It is distributed decentralised cloud based repository.
. It is used by developers, manual test engineers, automation test engineers & devops.

Developers Usage :-

. To maintain source code.

Manual Testers Usage :-

. To ~~main~~ store & maintain CRS documents & testcase documents.

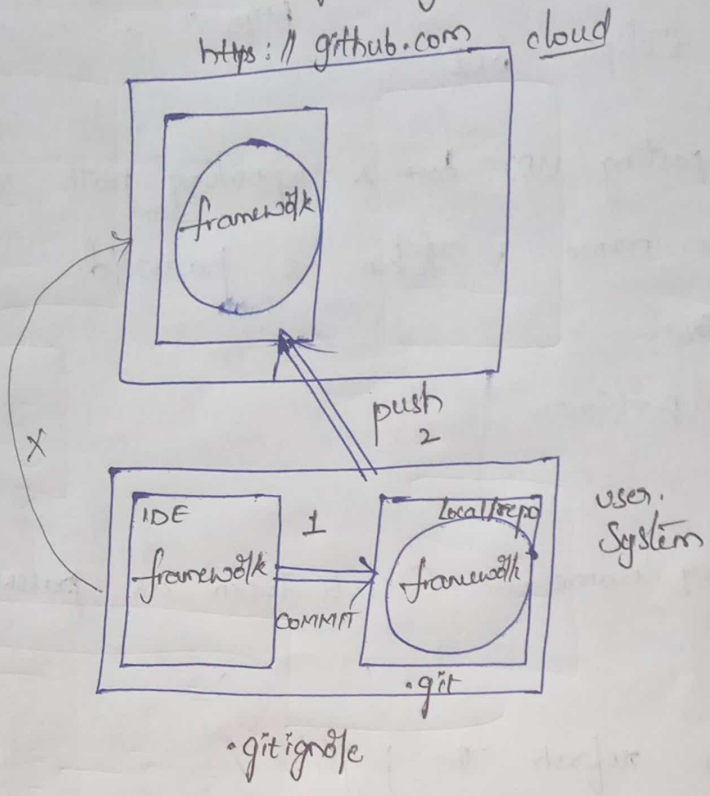
Automation test Engineers Usage :-

. To maintain automation framework.

Devops Usage :-

. To maintain build version files.

Why github is decentralised repository?



- Whenever we wanted to push the framework to github it initially creates local repository (.git).
- It stores code in local repository first only then the framework is push to global repository.

  That's why we it as decentralised repository.

Advantages :-

- Since it is cloud based repository maintanence team is not required.
- We can access the repository from any place using Internet.
- No initial Investment needed for storage.
- Jenkins also gets the new build from github.

Jenkins - Steps to trigger ms - framework

1. Open the browser login to jenkins.
   - Jenkins url - localhost:8080
   - login
2. Create a new job in jenkins.
   - click on New Item
   - Enter an Item name - HCSE10
   - click on maven project.
   - click on ok.
3. Integrate github repository to the job created
   - click on Source code Management
   - Git radio button
4. Open ur github repository includes credentials.
   - Open skillary HCSE10.
5. click on Code. → Copy the url.
6. Go back to jenkins paste the url under Repository URL.
7. Go to build → click on build.
   - Type (test) under goals and options.
   - click on apply
   - click on Save.

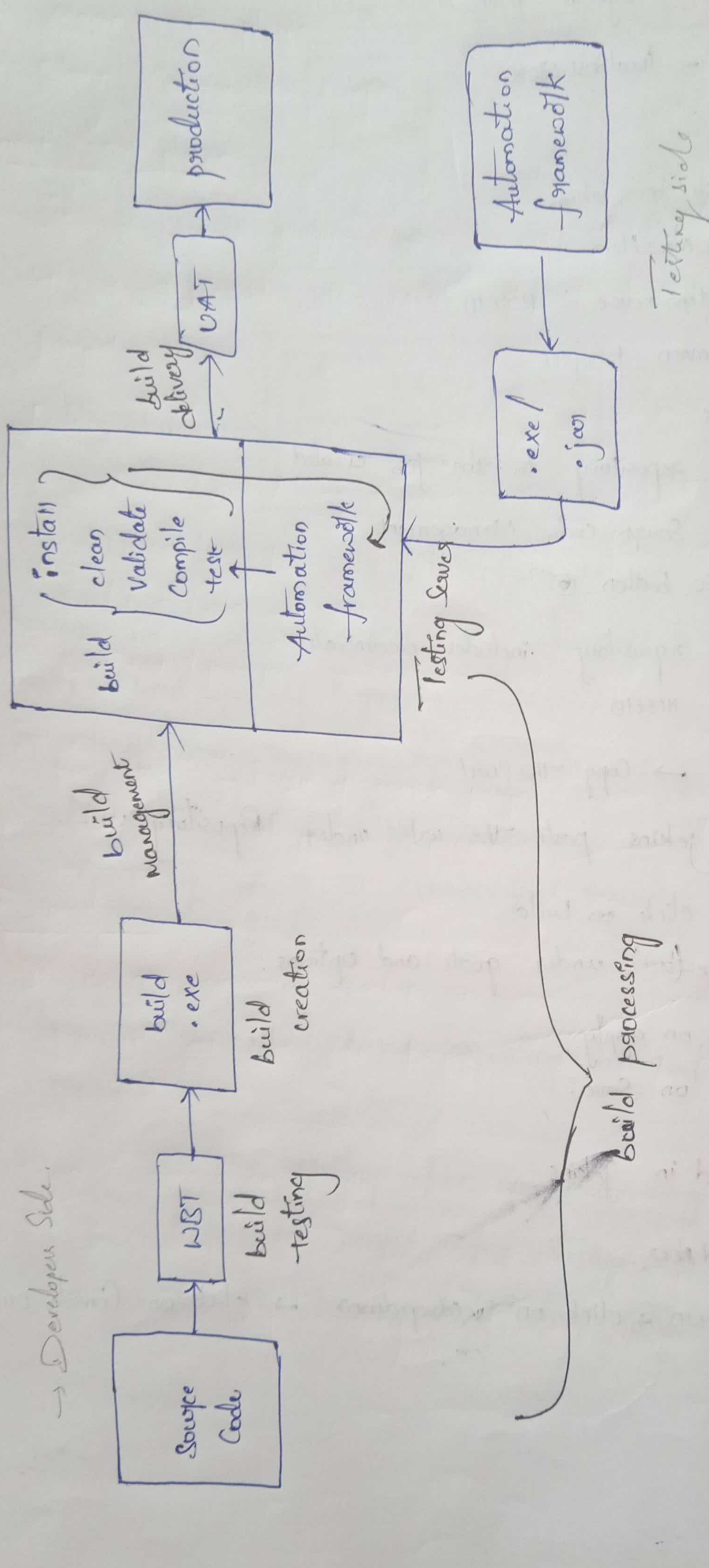
Trigger the build in Jenkins

- click on build Now.
  - go to down → click on v (dropdown) → click on Console output.

31/5/23
Wednes

Maven-Build Management Tool:-

Other build Management Tools } In out-side Market - They use - Ant, Gradle



Maven is a build management tool used in both development & testing.

Development Usuage :-

- It is used for build processing such as build testing, build creation & build develop deployment.

Build creation:-

- converting Source code to .exe file

Build testing:-

- Testing the Compilation issues in the project & providing the status.

Build deployment:-

- Installing build into testing environment.

Automation testing Usuage:-

- In automation it is used to test the automation framework build & testScripts.
- It is also use to check Integration issues blw framework Components.

Advantages of Maven:-

- Handles all dependencies jar file.
- checks Integration blw framework Components.
- Creates framework Configuration quick Setup.
- Supports Commandline execution.
- Supports jenkins
- provides default ~~project~~ folder structures for project.

Softwares in Maven

- Maven IDE plugin
- Maven Commandline plugin

1> Maven IDE plugin:

- It is inbuilt plugin available in IDE's (Eclipse intellij)
- It is responsible to create default folder structure in maven project.

Folder structure -

1. src/main/java - Generic libraries, Object Repository
2. src/main/resources - driver executable files (.exe)
3. src/test/java - test scripts.
4. src/test/resources - external files such as properties file, excel file.
5. pom.xml - heart of maven project where we provide required plugins & dependencies.

Dependencies

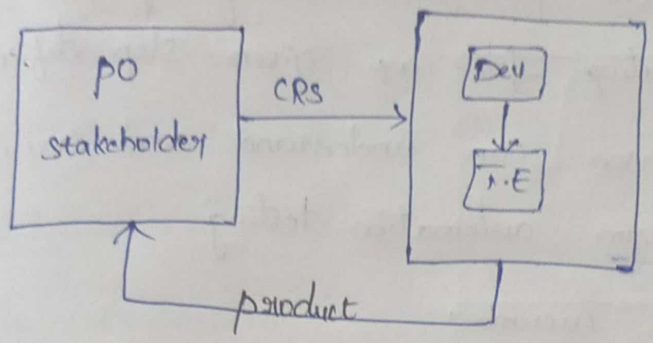
- It is an advanced feature in maven which downloads the required jar files into project automatically based on the dependencies provided.

2> Maven Commandline plugin:

- This plugin should be installed explicitly.
- It is used for executing the build / frame works via Commandline

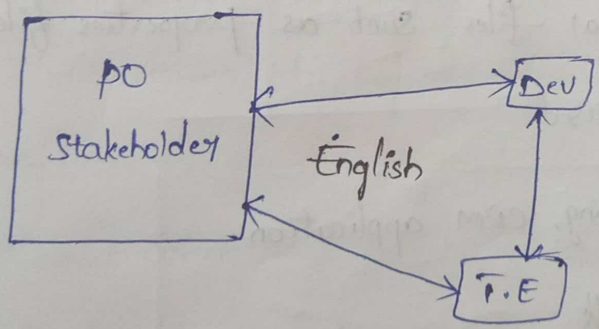
Framework approaches

1. Test Driven Development (TDD)



- In traditional development the priority is given for architecture, skills, resources and technicalities.
- Communication blw the team members is not enforced.
- Here manual testcases are developed by manual testengineer only then automation testengineer develops automation testscripts looking into testcase document.
- Tools used here are
  - TestNG
  - JUnit
  - NUnit
- We will have more reusuable methods in this approach.
- Test is the driven factor here.

2) Behaviour Driven Development (BDD)



- In this approach Communication blw the team members is given the first priority.
- changes of developing wrong product is less.
- Here automation testengineer will develop feature files which

act as manual test case document.

- featured files are developed using Gherkin language., it is normal english language with predefined set of keywords.
- All the steps in feature file are given step. defination. (code)
- Even a nontechnical person can understand and execute the feature files to perform automation testing.
- Tools used in BDD - Cucumber
  Specflow
  Kahlan
  JBehave.
- feature files are the driving factor here.
- We will have less reusuable methods.

1/6/2023
Thurs

Types of Frameworks

1. Data Driven Framework.
   - We go for data driven framework, whereever we have huge amount of data to be tested in the application.
   - We store data in external files such as properties file, Excel file, data providers are also used.
     Ex:- E-commerce, banking, CRM application.
2. Method Driven Framework
   - When the app has lot of reusuable methods we go for method driven framework.
   - All the reusuable methods are stored in generic libraries.
     Ex:- CRM, ERP applications.

3. Modular driven framework

When the application has lot of modules to be tested then we go for modular driven framework.

We store the testcases, testdata and xml files, module wise.

Ex:- CRM, E-commerce, healthcare application.

4. Keyword Driven Framework

The automation test engineer of developer will develop the program for every action of element. and store it in the form of keys in Excel of csv files.

This keys are used by the test engineers to automate the testCases

This type of framework is called as Keyword Driven Framework.

It is used only for Smaller applications such as Education related applications.

5. Hybrid framework

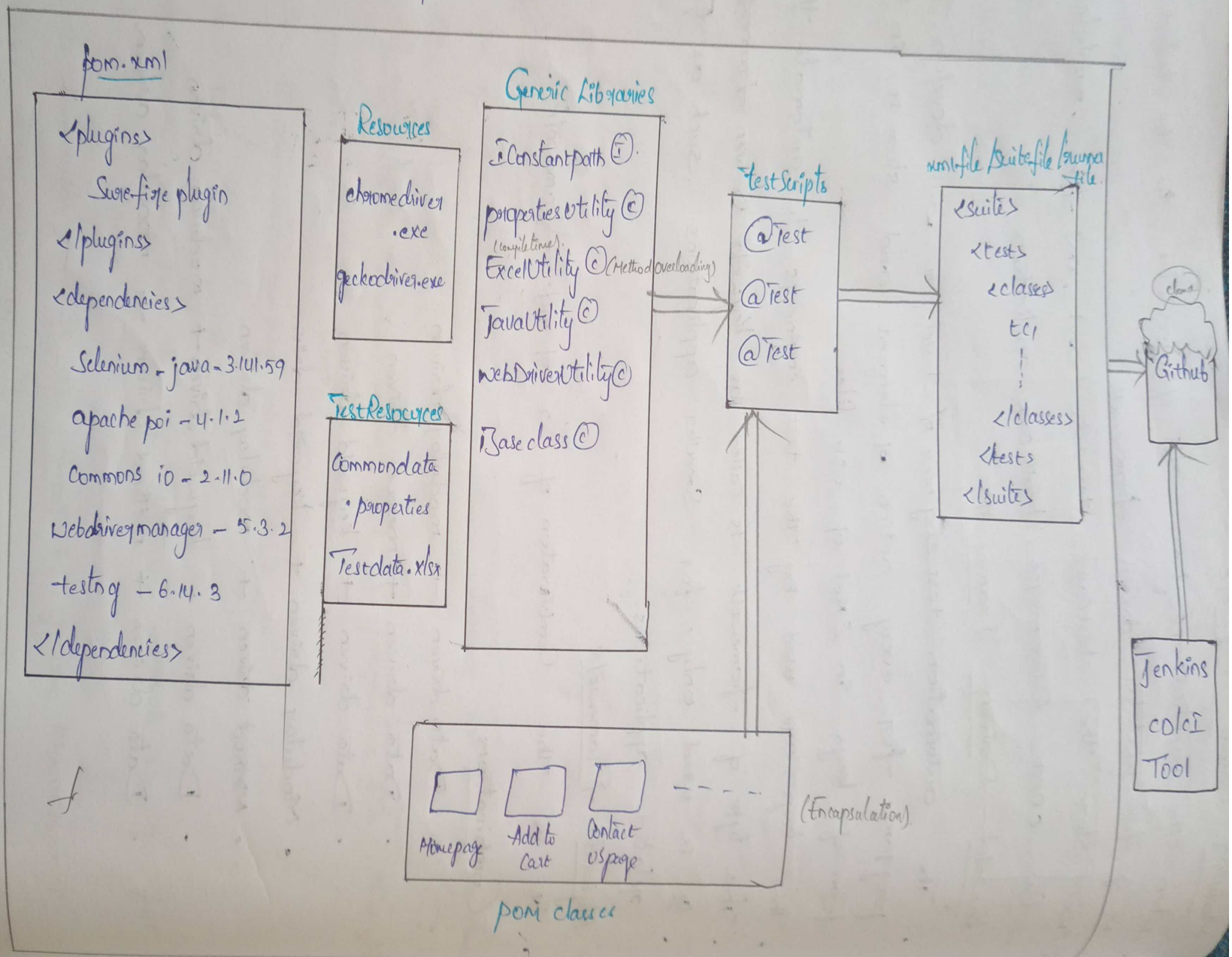
It is the Combination of 2 of more frameworks.

Combinations

- Data driven + modular driven
- Data driven + method driven ✓ we work this
- Data driven + keyword driven
- Modular driven + keyword driven
- Method driven + modular driven
- Data driven + keyword driven + modular driven
- Data driven + method driven + modular driven.

Framework Architecture

- worked on E-Commerce application.



I worked in Ecommerce application, which is of Hybrid project.

In this i used 2 frameworks like Datadriven framework, method driven framework.

It is a client Here i used client binding (Java).

In this i used tools like Eclipse IDE
Selenium
Apachepoi
Testng.
pom design pattern.

then Explain generic libraries

then Explain pom with the help of these we write testscripts which will be used in @Test, after writing the testScripts will Convert to xml files, after Converting these we push these into Github repository.

If we good enough, in jekins explain jekins indetail.

Srivalli