Date-19-8-19
Lecture-20

Automation Testing

Selenium

Page No.:
Date:
YOUVA

1) Definition of Automation

Testing an applications feature with the help of automation tool known as automation testing.

2) Important Automation tools

1) Selenium
2) QTP → paid
3) Sahi
4) Sahi pro
5) protector
6) Appium } mobile Automation
7) Selendroid }

*MNC mostly use their developed tool.

3) Disadvantage of manual

1) Require more resources.
2) It is time consuming.
3) Compatibility testing (Cross browser testing) is very difficult in manual testing. because if we have to check build on different different browser then if one browser take 10 min then 6 browser take 60 minutes.
4) Test cycle duration will be increased.
5) more human efforts are required.
6) If we do manual regression testing then it is time consumming

Page No.:
Date:
youva

4) Advantage of Automation

1) Less resources are required.
2) Compatibility testing is easy & less time consuming.
3) Regression testing is easy in Automation & less time consuming.
4) Test ~~life~~ cycle duration will be reduce.
5) Reusability of scripts:-
   Test cases are converted into program that is called as test scripts, we can use test scripts for multiple time.

5) Why we choose selenium

1) It is open source.
2) Supported by multiple languages
   1) Java, 2) Python, 3) C#, 4) pearl
3) compatibility cross browser test is possible.

6) Disadvantage of Selenium

1) we can automate only web base application.
2) we can not test captcha or barcode.
3) We can not do file uploading and downloading.
4) we can perform regression testing but can not perform Ad-hoc testing.

Page No.:
Date:

7) Installation

For installation we require jar file of selenium & ex file of browser.

1) Go to google
2) Type → selenium jar file download.
3) Click on first link
4) click on selenium download version 3.141.59 (selenium standalone server)

5) Scroll down
6) Select "latest" in front of google chrome.
7) then on new page click on big name at header
8) then click on chrome 76
9) then click on chrome driver win 32.zip.

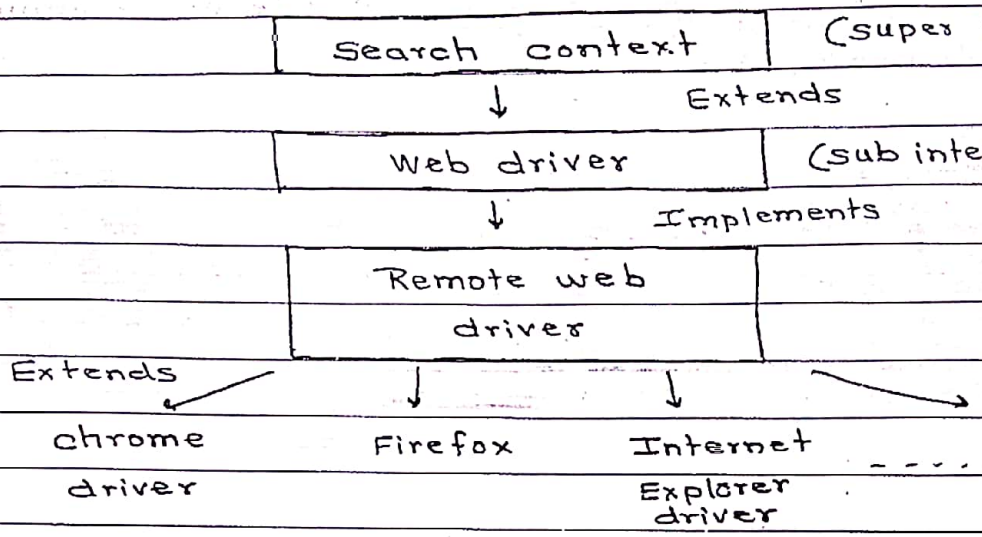
→ installation

1) open eclipse
2) new
3) Java project → Give project Name
4) Rig click on project
5) Build path
6) Configure build path
7) Libraries
8) add external jars
9) select selenium jar file
10) Apply & close.

Lecture 21
Date - 20/8/19

Page No.:
Date:

8) Selenium Architecture



1) Search context is super interface which is extended by web driver which is sub interface of search context.

2) Search context contains incomplete methods (i.e., abstract methods) these incomplete methods are extended by web driver so web driver contain its own incomplete method as well as incomplete methods of search context.

3) Remote web driver is implement class which provide defination to the incomplete methods of search context & web driver.

Page No.:
Date
YOUVA

4) Remote web driver

1) Remote web driver class is extended by diff diff browsers like chrome driver, Internet Explorer driver, firefox driver.

2) we write a script for a browser but we can run that script for multiple browser.

3) But to run the test scripts we need functions of web driver so we do upcasting.

Ex: we have to run script for google chrome, then we have to create object of chrome with refference to web driver

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

5) Web driver

i) web driver is an interface which perform action on browser like - open, close, maximize, resize.

* web driver VS web element

Web driver	web element
It is an interface which perform action on brawser	It is an interface which perform action on element of browser.
Ex: open, close, navigate, get, maximize	Ex: Dropdown, radio button

Page No.:
Date: YOUVA

8) methods of web driver

1) Open web browser

```
System.setProperty("webdriver.chrome.driver","URL");
WebDriver driver = new ChromeDriver();
```

rule:-

1) while creating object

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

These should be capital.

2) Open web site

```
driver.get("URL");
```

3) To maximize browser

```
driver.manage().windows().maximize();
```

4) There is no function present in web driver interface for minimize the browser. Instead of that we can change position & size.

5) To make delay

```
Thread.sleep(time in millisecond);
```

Ex. To make delay of 3 second we have to write.

```
Thread.sleep(3000);
```

This Thread.sleep(time) is not a function of web-driver interface, it is function of java.

6) To close browser

driver.close() or driver.quit();

close()	quit()
1) use to close browser	1) use to close browser
2) use to close current tab	2) use to close all tabs.

7) open two or more web site and navigate through them

1) To do this we use navigate function.

driver.navigate().to("URL");

(we can call this function multiple times)

(How many time we ~~open~~ call funct^n that many time new websites get open)

Ex: driver.navigate().to("https://google.com/");
driver.navigate().to("https://facebook.com/");

This open google first then fc.

we can add delay between them by Thread.sleep("-11-");

2) To go back from 2nd web site to first

```
driver.navigate().back();
```

→ This focus on 1st web site & go back from 2nd to 1st

3) To go from 1st to 2nd web site.

```
driver.navigate().forward();
```

→ This focus on 2nd web site & go to 2nd from 1st.

4) To refresh we current site

```
driver.navigate().refresh();
```

8) To get title of web site

```
String title = driver.getTitle();
```

1) Title of any web site is string formated.
2) so return /data type of title is string.
3) This return the title of URL which we enter.

Ex:-

```
System.setProperty("webdriver.chrome.dri
"https://google.com");

WebDriver driver = new ChromeDriver();
String title = driver.getTitle();
```

In testing for each script we check actual & expected result.

so for title we have to check actual & expected title.

so program is;

```
String act = driver.getTitle();
String Exp = "Name/title of site";

if (act.equals(Exp));
{
  S.O.P ("pass");
}
else
{
  S.O.P ("fail");
}
```

① String act = driver.getTitle();
→ this will return title of web site which is actually present.

② String Exp = "Name/title";
→ the Name/title we enter manually as per customer requirement.

③ These return types are string so to compare actual & Expected we use string functions.

String1 Name . equals (String2 name)

we can use this function in S.O.P / if else.

So to compare we use it in if else

→ Complete program

```
package demo;
public class test
{
  public static void main (String[] args)
  {
    System.setProperty ("webdriver.chrome.driver"
                        "URL");
    WebDriver driver = new ChromeDriver();

    Thread.sleep (3000);

    driver.get ("URL");

    driver.navigate().to get ("URL");
    driver.navigate().to get ("URL");

    driver.navigate().back();

    driver.navigate().forward();

    driver.navigate().refresh();

    driver.close();

    driver.close();

    String title = driver.getTitle();

    System.out.println (title);

    String act = driver.getTitle();
    String Exp = "title";
    if (act.equals (Exp));
    {
      s.o.p ("pass");
```

open google chrome

open web site

open two web sites

from 2nd to 1st

from 1st to 2nd

refresh current

close current tab

close all tabs

fetch title

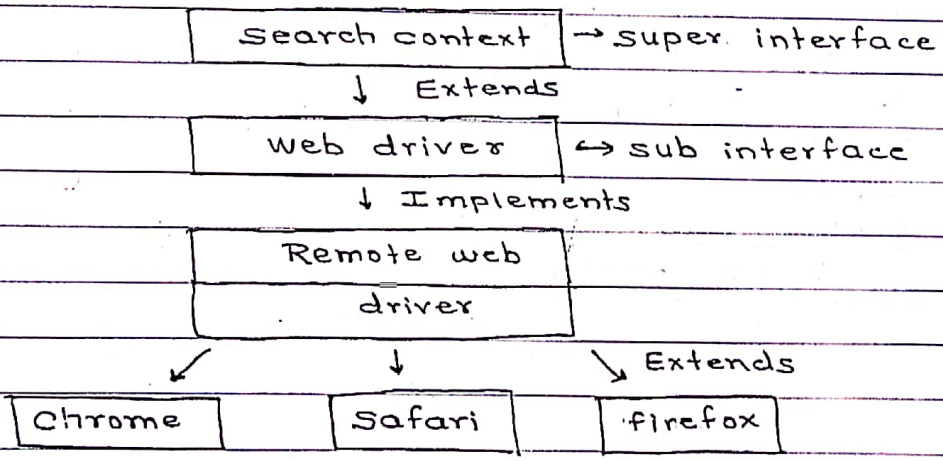
print title

compare actual title with user expected title.

```
        }
        else
        {
            System.out.println("fail");
        }
    }
}
```

→ Summary

1) Structure of selenium



2) methods in web driver

1) open browser

```
System.setProperty("webdriver.chrome.driver",
                   "URL");
WebDriver driver = new ChromeDriver();
driver.get("URL");
driver.close();
driver.quit();
```

```
driver.navigate().to get ("URL");
driver.navigate().to get ("URL");
driver.navigate().back();
driver.navigate().forward();
driver.navigate()$ refresh();

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

String title = driver.getTitle();
String act = driver.getTitle();
String exp = "Title";
if (act.equals (exp));
{
    S.o.p (act);
}
else
{
    S.o.p (Exp);
}
```

Lecture 22
Date - 21-8-19

Page No.:
Date:
YOUVA

1) Suppose if we open URL & Navigate through 2,3 pages & currently ~~as~~ we are on 3rd page then ~~to~~ we have to get URL then: we use

getCurrent~~Url~~Url()

```
String URL = ~~new~~driver.getCurrentUrl();
S.o.p (URL);
```

Program

```
package demo;
public class test
{
    public static void main (String [] args)
    {
        System.setProperty ("webdriver.chrome.driver",
                "path");

        WebDriver driver = new ChromeDriver ();

        driver.get ("URL");

        driver.~~N~~avigate().to(get ("url");
        driver.navigate().to get ("url");
        driver.navigate().forward ();

        String url = ~~new~~ driver.get~~C~~Current Url ();
        System.out.println (url);
    }
}
```

2) To change position of browser

Ans:-

Function used :-

driver . manage () . window () . setPosition (obj 1, ~~arg 1~~);

* Here args are pixels. for object

To use this function we have to create object of point first.

point d = new point (args, args);

width Height

```
package demo;
public class test
{
  public static void main (String [] args)
  {
    System.setProperty ("webdriver.chrome.drive
                        " path");
    WebDriver driver = new ChromeDriver ();

    point p = new point (200, 300);
    drive.manage ().window ().setPositon(
  }
}
```

Page No:
Date:

Q) To change size of browser.

Ans:-

Function used:-

```
driver.manage().window().setSize(obj);
```

To use this function we have to create object of diamension.

```
Diamension d = new Diamension(args, args);
                                width  Height
```

```
package demo;
public class test
{
  public static void main (String[] args)
  {
     System.setProperty("webdriver.chrome.driver",
                        "path");

     WebDriver driver = new ChromeDriver();

     driver.get("url");

     Diamension d = new Diamension(300,400);
     driver.manage().window().setSize(obj);
  }
}
```

Page No.:
Date:

4) Consider a HTML program

Write HTML code in notepad & save it with .html extension & then open it with browser.

1) Start & End of HTML code

```
<html>

</html>
```

2) To write title on any web page.

```
<html>
    <title>
        write title
    </title>
</html>
```

3) To add information we use body tag

```
<html>
    <title>
        write title
    </title>
    <body>
        write information
    </body>
</html>
```

Page No.:
Date:
YOUVA

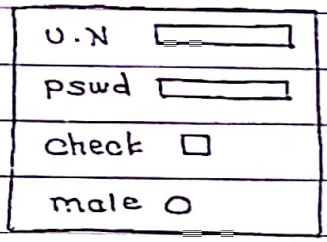
4) There are 3 main fields in html code

① Tag

② Attribute

③ text

5) consider a registration page.




```
<html>
    <body>
        UN <input type = 'text' > <\br>
        pwd <input type = 'password' > <\br>
        check <input type = 'checkbox' > <\br>
        male <input type = 'radio' > <\br>
    <\body>
<\html>
```

① Tag

Anything present after '<' (less than) symbox is called tag.

Tags in above code

html

body

input UN <input type = 'text' > <\br>

↑

tag

2) Attribute

Anything present after tagname with '=' symbol.

Attribute in above code.

type.

UN <input type = 'text' >

↑ Attribute Name

↑ Attribute value

3) text

Anything present after '>' & before '<' is calle text.

UN <input type = 'text' >
 pwd <input type = 'passwor

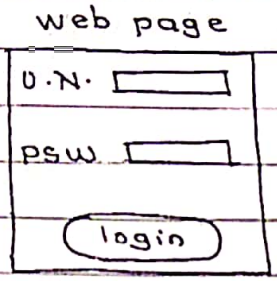
↑ Text.

Lecture 23
Date - 22-8-19

Page No.:
Date:
YOUVA

Locator

1) Locator



i) To login in the account we have to enter psw/uN.
ii) But to check whether these correct & working or not we have to enter credential through selenium.
iii) So we have to find ~~selentum~~ element through selenium
iv) To identify them, selenium uses locator.

Syntax

① For text box.

```
driver.findElement(locator).sendKeys("username text");
```

② For button

```
driver.findElement(locator).~~sendkeys~~ & click(" ");
```

Function of web driver } to use locator — web element function.

→ NOTES

1) Locator are use to find element by using locator type.

2) To identify element present in web page we need to use "find element" method present in web driver.

① To open browser.

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

② To get URL

```
driver . get ("URL");
```

③ To find element.

```
driver. findElement (locator) . sendkeys ("text");
```

3) Find element method find element with the help of By class, which contains static method.

4) All the static methods present in the By class are known as locator types.

5) Types of locator

1) Tag Name
2) ID
3) Name
4) Classname
5) linktext
6) partial link text
7) CSS
8) xpath ← important

xpath

→ How to use locator

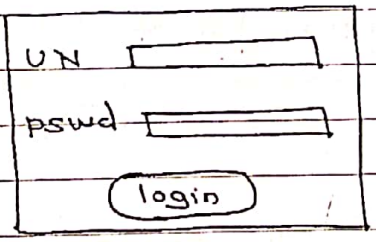
Suppose we are using xpath locator.

```
driver.findElement (By.xpath ("xpath Expression"))
                    .sendkeys ("username");
```

→ xpath

We use xpath locator to uniquly identify element.
Total six ptypes of xpath is there.

Type 1 : xpath by Attribute




```
<html>
  <body>
    UN <input type='text' id='1234'>
    pasword <input type='password' id='3547'>
    login <button type='submit' value='login'>
  </body>
</html>
```

Formula for xpath by attribute

```
(By.xpath ("//tag [@ atribute Name = Attribute value]"))
```

Page No.:
Date:

Example:-

UN <input type='text' id='1234'>

tag	attribute Name	Attribute value	Attribute Name	Attribute value.

→ when we use xpath by attribute then always give preference to id attribute & last prefere to type attribute.

so xpath expression is

(By.xpath("//input [@id='1234']"))

use in code

```
driver.findElement(By.xpath("//input[@id='1234'
    .sendKeys("prasad");
```

* This will enter "prasad" at username field.

Page No.
Date
YOUVA

⊛ How to create xpression

1) open facebook login page
2) right click anywhere & select last option inspect.
3) when html code opens, then at the upper left corner there is [↖] arrow. Click on that arrow & move cursor on any element, for that element code get open.

4) Then find code for UN, check tag names in code.
5) Then find attribute name where we find id, class, Name
6) Then give preference to id.
7) double click on attribute value of id & copy it.
8) press control + f
9) Then at bottom one box get open. write xpath Expression in that box.

10) ~~then~~ type ← paste this
//input [@ id = '1234']

11) infront of this code there should by 1 of 1.

12) copy this expression & put this in program.

Page No.:
Date:

Program

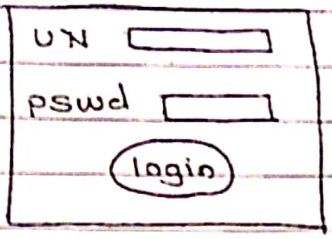
```
package demo;
public class test
{
  public static void main (String[] args)
   {
      system.setProperty ("webdriver.chromedriver.driver"
                                    "path");
      WebDriver driver = new ChromeDriver ();

      driver.get ("Mhttps://facebook.com");

firstName driver.findElement (By.xpath ("//input[@id='u_1_z"]")).
                sendKeys ("prasad");

setName driver.findElement (By.xpath ("//input[@name='lastname']"
                sendKeys ("kadam");

ign Up  driver.findElement (By.xpath ["//button (@id='u_0_z"].click

   }
}
```

Lecture 24
Date- 23-8-19

Page No.:
Date:
youvA

Type 2: xpath by text

If attribute is not available then in that case we xpath by text.




```
<html>
   <body>
        UN <input type = 'text'>
        pwd <input type = 'password'>
        <a href = 'link'> link </a>
   </body>
</html>
```

Here in this code text is link

Syntax

```
//tag [text() = 'text value']
```

Ex: `<a href = 'link'> link </a>`

```
//a [text() = 'link']
```

usage

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

Page No.:
Date:
YOUVA

Practical Example

Suppose we have to inspect signup button in fb.

```
<#button id='1234' > signup </button>
```

So,

```
driver.findElement (By.xpath ("//button [text() = 'signup
        .click();
```

Program

```
package demo;
public class test
{
  public static void main (String [] args)
    {
      System.setProperty ("webdriver.chrome.driver","path
      WebDriver driver = new ChromeDriver ();
      driver.get ("url");

driver.findElement (By.xpath ("//button [text()='sign
          .click();

    }
}
```

Type 3 : xpath by contains

If text is too large, then insted of passing full text we pass only substring.

Ex. prasadkadam1234tester is the text then we can pass 'adkad' & substring means continuous related words so we can not pass aak

syntax

// tagnam [contain (text() , 'text value')]

Ex: prasadkadam1234tester

// input [contain (text(), 'kadam')]

Usage

driver. findElement (By. xpath ("//[contain (text(), 'kadam')]
. SendKeys ('prasad');

When to use

1) If text is too large.
2) if spaces are there.

two type of spaces

Keybord stroke	Non breakable store
↓	↓
P_a_	P a
	↓
	we can not use xpath by text to use non breakable space text value

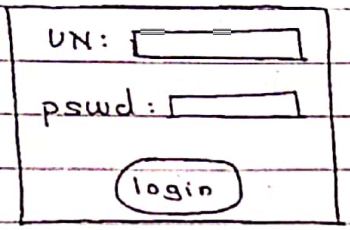
Page No.:
Date:

Program

```
package demo;
public class test
{
  public static void main (String[] args)
  {
    System.SetProperty ("Webdriver.chrome.driver", "pa
    WebDriver driver = new ChromeDriver ();
    driver.get ("url");
    driver.findElement (By.xpath ("//button[conta
          (text(), 'up')]")).click();

  }
}
```

Page No.:
Date:
YOUVA

Type-4 : xpath by index




```
<html>
  <body>
      UN <input type='text' name='name'>
      pwd <input type='text' name='name'>
  </body>
</html>
```

1) Here for user name & password attributes & their values are same.

2) So when we use xpath by attribute //input[@type='text'] then again and again it give i/p in 1st field.

3) Then to select appropriate element in this case, we use xpath by index.

1) How to find Index

→ 1st method

i) when we write //input[@type='text'] if 1 of 6 matches are there & our element is at 2nd position, then put cursor at 'text' and hit Enter button, then code get select & put cursor on code, the element related to code get heighlight, if it is desired ~~imp~~ element then see at 2 of 6 then 2 is index.

→ 2nd method

if we directly write xpath by attribute i.e.,
//input [@type ='text'][1] & change index value
then also new code- new element get hightight.

Syntax

//tag [@attribute_name = 'attribute_value'][index]

Usage

UN <input type='text'>

```
driver.findElement (By.path (//input [@type = 'text'][1]
        .sendKeys ('prasad');
```

program

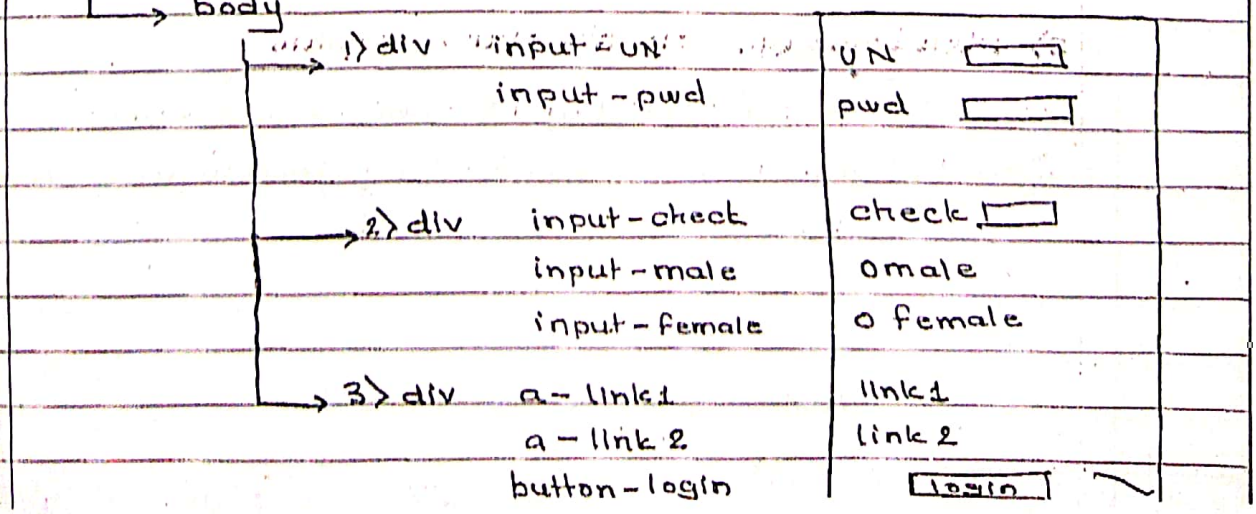
```
package demo;
public class test
{
 public static void main (String [] args)
  {
    System.setProperty ("webdriver.chrome.driver", "path"
    WebDriver driver = new ChromeDriver();
    driver.get ("url");
driver.findElement (By.xpath ("//input [@type='text'][2]
        .sendKeys ('prasad');
  }
}
```

Leoture 25
Date - 24-8-19

Page No.:
Date:
youva

```
<html>
  <body>
    <div>
        UN <input type = 'text' >
        pwd <input type = 'password' >
    </div>

    <div>
        check <input type = 'checkbox' >
        male <input type = 'radio' >
        female <input type = 'radio' >
    </div>

    <div>
        <a href = " " > link1 </a>
        <a href = " " > link2 </a>
        <button type = 'button' ; value = 'login' > </button>
    </div>
  </body>
</html>
```

HTML tree diagram



1) The tagname is use to identify the element on web page.

2) It is use for easy access.

3) If there are number of elements present in web page then by using tagname we can easily find the element.

Type 5 :- Absolute x path

1) In this focus is on the html tag (/html)

Example :- to select password field.

/html / body / div[1] /input[2]

2) We can not use it in real time app.

3) It is use to navigate from root of parent to immidiate child.

4) To achieve absolute x path we need to use "/"

5) Disadvantage of absolute x-path

(i) x path is too lengthy & time consuming.

(ii) Identifying an element by developing html tree diagram is difficult.

Page No.:
Date:
youva

Type 6 :- Relative x path

1) We can use single " / " as well as " // " to write the child class.

2) // is use to navigate from parent to any child.

3) In this / is use to navigate the parent child class

4) use to navigate from parent to immidiate child class

Example :- to select password field.

//div[1] // input [2]

5) Disadvantage

Identifying an element by developing HTML tree diagram is difficult.

Functions of Web Element

Page No.:
Date:

Web Elements

1) getText()

1) To get text on browser we use getText()

2) To use get text function we use xpath by co

Ex:- To get text of sign up button xpath is

```
By.xpath ("//button[contains(text(),'sign'
```

3) syntax of get text().

```
driver.findElement(By.xpath("Expath by contains"
                .getText();
```

4) But to print text we have to store this expre in the variable & the return type of the va is String.

```
String abc = driver.findElement(By.xpath("//b
    [contains(text(),'sign')]").getText
```

5) Then we have to print that variable.

```
System.out.println(abc);
```

Program

```
package demo;
public class test
{
    public static void main (String[] args)
    {
        System.setProperty ("webdriver.chrome.driver", "path");
        WebDriver driver = new ChromeDriver ();
        driver.get ("url");
String abc = driver.findElement (By.xpath ("//button
        [contains (text(), 'sign ')]")).getText()

System.out.println (abc);

}
}
```

o/p : sign up

2) isEnabled()

1) If the element is Enabled then return true & false.

2) We give return type of the isEnable() is boo

3) syntax:-

```
boolean text1 = driver.findElement(By.xpath("//butto
        [contains(text(),'sign')]")).isEnabled(
    System.out.println(text1);
```

4) A simple program

```
package demo;
public class test
{
  public static void main(String[] args)
  {
    System.setProperty("webdriver.chrome.driver",
    WebDriver driver = new ChromeDriver();
    driver.get("Url");

    boolean text1 = driver.findElement(By.xpath("//b
        [contains(text(),'sign')]")).isEnabled
    System.out.println(text1);
  }
}
```

Page No.:
Date:
youva

5) check whether element is enable or not, if true then write is enable otherwise is not enable.

```
package demo;
public class test
{
  public static void main (String [] args)
    {
      System.setProperty ("webdriver.chrome.driver", "path");
      WebDriver driver = new ChromeDriver();
      driver.get ("url");

    boolean text1 = driver.findElement (By.xpath ("//button
        [contains (text(), 'sign')]")).isEnabled();

      System.out.println (text1);

      if (text1 == true)
      {
        System.out.println ("is Enabled");
      }
      else
      {
        System.out.println ("is not Enabled");
      }
  }
}
```

Page No.:
Date:

2) is Selected

1) Use to check whether the radio button or checkbox is selected or not.

2) Syntax:-

```
boolean text2 = driver.findElement(By.xpath("xpath
        Expression")).isSelected();
    System.out.println(text2);
```

3) Simple program

```
package demo;
public class test
{
  public static void main(String[] args)
  {
     System.setProperty("webdriver.chrome.driver","pa
     WebDriver driver = new ChromeDriver();
     driver.get();
     boolean text2 = driver.findElement(By.xpath
       ("//label[@for='0_0_6']")).isSelected();
     System.out.println(text2);
  }
}
```

Page No.:
Date:

4) Write a program in which first check whether the radio button in front of male is selected or not if not then click/select it.

```
package demo;
public class test
{
 public static void main (String[] args)
  {
    System.setProperty ("webdriver.chrome.driver", "path");
    WebDriver driver = new ChromeDriver ();
    driver.get ("url");
    boolean text2 = driver.findElement (By.xpath ("//label
        [@for = 'u_0_6'] ")).isSelected();
    s.o.p (text2);

    if (text == = false)
    {
        driver.findElement (By.xpath ("//label[@for = 'u_0_6']
            .click();
    }
   else
    {
        s.o.p ("already selected");
    }
  }
}
```

Page No.:
Date:

4) Is displayed

1) We can get title but for every page some sites do not provide title, so we use display.

2) If component / Element is actually present or not is get checked in display.

3) If present then return true or false.

4) Syntax:-

```
boolean text = driver.findElement(By.xpath("//div[
                text() = 'login']")).isDisplayed();

System.out.println(text);
```

5) program

```
package demo;
public class test
{
 public static void main (String [] args)
 {
  System.setProperty("webdriver.chrome.driver","path
  WebElement driver = new ChromeDriver();
  driver.get("url");

  boolean text = driver.findElement(By.xpath("//div
          [text() = 'login']")).isDisplayed();

  System.out.println(text);
 }
}
```

Page No.:
Date
youva

5) code optimization

male O
female O

* if male & female radio buttons are there.
* we have to check if they are selected, Enable or display
* in normal code we writh 3 function separately.

* instead of that we declare xpath for them & store it in variable & call that variable to check these functions.

* Syntax :-

```
WebElemen male = driver.findElement (xpath);
```

(return type ← WebElemen; variable ← male; web drives function ← driver.findElement)

* in this we give return type as WebElement because later we will call functions of web element.

* function present next to the driver. is function of webdriver & function present after (xpath) is function of web Element.

* Example :-

```
WebElement male = driver.findElement (By.xpath ("//label
    [@for = 'U-0-6'] "));

    boolean a = male.isSelected ();
    boolean b = male.isDisplayed ();
    boolean c = male.isEnabled ();
    s.o.p (a);
    s.o.p (b);
    s.o.p (c);
```

Page No :
Date:

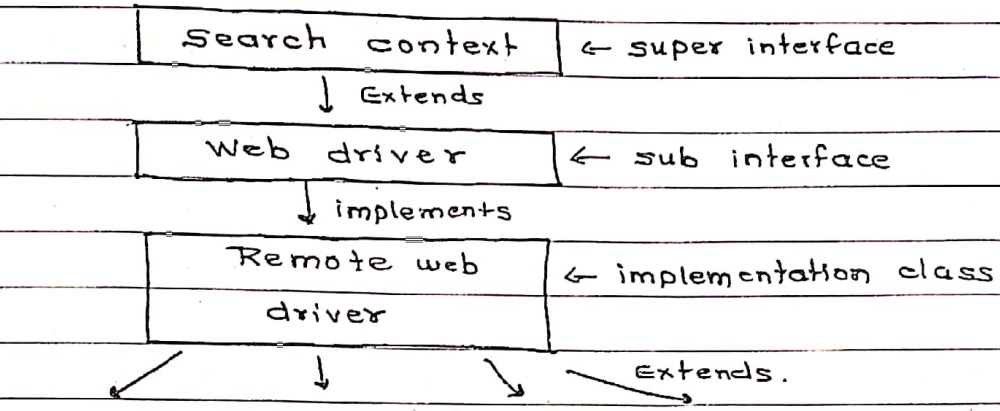
Program

```
package demo;
public class test
{
 public static void main (String [] args)
 {
   System.setProperty ("webdriver.chrome.driver",
   WebDriver driver = new ChromeDriver();
   driver.get ("Url");

  WebElement male = driver.findElement (By.xpath
              ("//lable [@for = 'u-0_6']"));

   boolean a = male.isSelected ();
   boolean b = male.isDisplayed ();
   boolean c = male.isEnabled ();
    S.o.p (a);
    S.o.p (b);
    S.o.p (c);
  }
}
```

Summary

Page No.:
Date:
youva

1) Architecture of Selenium.



Web driver contains method to do operation on browser so to use methods of web driver we have to use up casting

2) methods of web Driver

1) Open chrome browser

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

2) Close browser (current tab)

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

3) close browser (all tabs)

```
driver.quit();
```

4) put URL in browser

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

5) To navigate / open two sites :-

```
driver.navigate().toget("Url");
driver.navigate().to get("Url1");
```

6) To move forward from one site browser to other.

```
driver.navigate().forward();
```

7) To move back from one site browser to other.

```
driver.navigate().back();
```

8) To refresh current site

```
driver.navigate().refresh();
```

9) To maximize browser

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

10) Change size of browser

```
Diamention a = new Diamention (200,400);
driver.manage().window().setSize(a);
```

11) Change position of browser

```
Point a = new Point (200,400);
driver.manage().window().setPosition(a);
```

12) Get title of web page.

```
String text = driver.getText();
S.o.p (text);
```

Page No.:
Date:
YOUVA

13) To compare actual & Expected Title.

```
String act = driver.getTitle();
String exp = "Name of title";

if (act.equals(exp))
{
    S.O.P ("correct");
}
else
{
    S.O.P ("incorrect");
}
```

14) To get current url to confirm which page is open.

```
String url = driver.getCurrentUrl();
System.out.println(url);
```

Page No.:
Date:

3) Locator

1) Types:-

ID
name
classname
link text
partial link text
css
xpath.

4) Locator : xpath

Types of xpath

1) xpath by attribute
2) xpath by text
3) xpath by contains
4) xpath by index.

1) xpath by Attribute

```
driver.findElement(By.xpath("//tag[@attribute_Name='attribute_value']")).sendKeys("st
                                                                        .click();
```

* give preference to attribute ID.
* give least preference to attribute type

Page No.:
Date:
YOUVA

2) xpath by text

```
driver.findElement(By.xpath("//tag[text()='text_value']")).
    .sendKeys(" "); or click();
```

* use when we want to findElement using text.

3) xpath by contains

```
driver.findElement(By.xpath("//tag[contains
    (text(), 'text value')]")).click() or sendKeys("");
```

4) xpath by index

```
driver.findElement(By.xpath("//tag[@Attname=
    'Att value'][index]")).click();
                          sendKeys(" ");
```

5) web element

Types of web element :-

1) isSelected()
2) getText()
3) isEnabled()
4) isDisplayed()

Page No.:
Date:

1) isSelected()

Use to check whether the checkbox or radio button is selected or not.

```
boolean a = driver.findElement(By.xpath(expression)
                .isSelected();

    S.o.p(a);
```

2) isEnabled()

Use to check whether the element is enabled or not.

```
boolean b = driver.findElement(By.xpath(expression)).
                .isEnabled();
  S.o.p(b);
```

3) isDisplayed()

```
boolean c = driver.findElement(By.xpath(expression))
                .isDisplayed();
      S.o.p(c);
```

4) getText()

```
String text = driver.findElement(By.xpath by conta
                .getText();

    S.o.p(text);
```

Lecture 26
27-8-19

Page No.:
Date:

→ **Other locators**

		How to write ?
Tagname	attribute base locator	By.tagname ("tag's name");
id		By.id ("id value");
name		By.name ("name value");
classname		

→ we use whole text

linktext	text base locator	By.linktext ("text value")
partial link text		By.partiallinktext ("partial text value")

→ use partial text

Disadvantage :- if two tagname, attribute, text are same then it goes to first element.

css	Expression
xpath	

Function having "is" have return type boolean.

List Box

Page No.:
Date:

1) List Box

1) Single selectable list box → select one option at a time.

2) If "select" tag name is there then use this.

month	
	month
	Jan
	Feb
	!

```
<select>
    <option value = "123" > Jan </option>
    <option value = "456" > feb </option>
     ;
</select>
```

This is string value & can not be duplicate

1) How to select value or handle text box ?

Ans:- It has 3 steps.

STEP 1

Identify the list box which need to be handle & store it in refference variable.

```
WebElement day = driver.findElement(By.xpath
                ("//select [@id='day']"));
```

Page No.:
Date:
YOUVA

STEP 2

Create an object of select class, which accept WebElement argument (parameter)

```
Select $a= new Select (day);
```

STEP 3

By using select class function like select by index(); select By value(); , select By VisibleText(); we can select value from list box.

```
a.selectByIndex (Integer Index);
a.selectByValue ("string value");
a.selectByVisibleText ("string value");
```

→ Program

```
package demo;
public class test
{
  public static void main (String[] args)
  {
    System.setProperty ("webdriver.chrome.driver", "path");
    WebDriver driver = new ChromeDriver();
    driver.get ("url");
    WebElement day = driver.findElement (By.xpath
                ("//select [@id = 'day']"));
    Select a = new Select (day);
    a.selectByIndex (3);          ← ~~mar~~ 3
    a.selectByValue ("3");        ← 3
    a.selectByVisibleText ("3");  ← 3
  }
}
```

Page No.:
Date:

→ To print total options present in list.

STEP 1

Select the list box which we need to handle & store it in refference variable.

```
WebElement b = driver.findElement (locator);
```

STEP 2

Create an object of Select class which accept webelement argument (parameter)

```
Select s = new Select (b);
```

STEP 3

To get all options/element present in the list, we have to call getOptions function & store it in reff variable

```
List <WebElement> a = s.getOptions();
```

STEP 4

To print all option one by one we have to use for loop.

```
for (int i=0 ; i<= a.size -1 ; i++)
{
    String x = a.get (i) .getText ();
    System.out.println (x);
}
```

STEP 5

To get size of list we have to call getSize() function.

```
int y = a.size ();
System.out.println (y);
```

Page No.:
Date:

Program

```
package demo;
public class test
{
  public static void main (String[] args)
  {
    System.setProperty ("webdriver.chrome.driver", "path");
    WebDriver driver = new ChromeDriver ();
    driver.get ("url");

    WebElement a = driver.findElement (locator);

    Select s = new Select (a);

    List <WebElement> b = s.getOptions ();

    for ( int i=0 ; i<= b.size-1 ; i++)
    {
      String str = b.get(i).getText();
      System.out.println (str);
    }

    int z = b.size ();
    s.o.p (z);
  }
}
```

→ To check whether the given list box is single selectable or multi selectable.

STEP 1

Identify the list box on which we need to perform action.

```
WebElement a = driver.findElement ("locator");
```

STEP 2

Create object of select class which accept web element argument (parameter)

```
Select s = new Select (a);
```

STEP 3

To check the list box is single selectable or multi selectable we have to call isMultiple(); function & store it in refference variable having return type boolean.

```
boolean b = s.isMultiple();
System.out.println(b);
```

Page No.:
Date:
YOUVA

Program

```
package demo;
public class test
{
 public static void main (String [] args)
 {
   System.setProperty ("webdriver.chrome.driver", "path");

   WebDriver driver = new ChromeDriver ();

   ~~Select s = new Select~~
   WebElement a = driver.findElement (locator);

   Select s = new Select (a);

   boolean b = s.isMultiple ();
   System.out.println (b);

 }
}
```

→ To verify value which we selected.

STEP 1

Identify the list box which we have to perform action and store it in reff variable having return type web element.

```
WebElement a = driver.findElement (locator);
```

STEP 2

Create an object of Select class which accept argument of webElement (parameter)

```
Select S = new Select (a);
```

STEP 3

To identify the element we have to call getFirstSelected Option () function & store it in refference variable having return type WebElement.

```
WebElement b = S.getFirstSelected Option ();
System.out.println (b.getText ());
```

Page No.:
Date:
YOUVA

Program

```
package demo;
public class test
{
 public static void main (String [] args)
 {
  Web
  System.setProperty ("webdriver.chrome.driver","path");

  WebDriver driver = new ChromeDriver ();

  Select S = new
  WebElement a = driver.findElement (locator);

  Select s = new Select (a);

  WebElement
  boolean b = s.getFirstSelectedOption ();

  System.out.println (b.getText ());

 }
}
```

Page No.:
Date:

→ multiselectable List box.

1) To identify whether given list box is multiselectable or not

STEP 1

Identify the list box which we have to perform operation & store it in reff variable having return type WebElement.

```
WebElement a = driver.findElement(locator);
```

STEP 2

Create object of Select class which accept Argument of web element (parameter)

```
Select s = new Select(a);
```

STEP 3

To check whether the list is single or multiselect we have to call function isMultiple();

```
boolean b = s.isMultiple();
S.o.p(b);
```

Page No.:
Date:
YOUVA

program

```
package demo;
public class test
{
 public static void main (String [] args)
 {
  System.setProperty ("Webdriver.chrome.driver", "path");

  WebDriver driver = new ChromeDriver ();

  WebElement a = driver.findElement ("path");

  Select S = new Select (a);

  boolean b = S.isMultiple ();

  System.out.println (b);

  if (b == true)
  {
     System.out.println ("listbox is multi selectable");
  }
  else
  {
    S.o.p ("list box is single selectable");
  }
 }
}
```

Page No.:
Date:

2) How to deselect option from list box:

```
deselectByIndex();

deselectByValue();

deselectByVisibleText();

deselectAll();
```

Lecture 27
28-8-19

Screen Shot

Page No.:
Date:
youva

1) Screen Shot

When any test case get fail then we take screen shot.

STEP 1

To take screen shot in selenium first we have to down cast driver object to TakeScreenshot interface then we have to call function from interface i.e., getScreenshotAs() then pass i/p to it i.e., (OutputType.FILE);

```
FILE S = ((TakeScreenshot) driver).
                getScreenshotAs (OutputType.FILE);
```

STEP 2

To move screen shot from default location to desire location we have to create object screenshot having return type is file, because screenshot is img.

```
File d = new File ("path of folder");
```

STEP 3

To move actually we have to call function :

```
FileHandler.copy (S, d);
                  ↳ destinition.
                ↱ source
```

Page No.:
Date:

Program

```
package demo;
public class test
{
 public static void main (String [] args)
 {
  System.setProperty ("webdriver.chrome.driver", "path"
  WebDriver driver = new ChromeDriver ();
  driver.get ("url");

  driver.findElement (By.xpath ("//tag [@attr = 'value']
                .sendKeys ("  ");

  File S = ((TakeScreenshot) driver).getScreenshotAs
                (OutputType.FILE);

  File d = new File ("path of folder // name.jpg"

  FileHandler.copy (S, d);
 }
}
```

OR

```
File.utils.copyFile (Source, dest);
```

Page No.:
Date:

1) To give random names to ScreenShot.

STEP 1

To take screenshot in selenium first we have to down cast driver object to TakeScreenshot interface

Then to take screenshot we have to call function of interface i.e., getScreenshotAs() & pass OutputType.FILE argument to it.

Then we have to store it so we store it in reff variable having return type File.

```
File S = ((TakeScreenshot) driver).getScreenshotAs
                (OutputType.File);
```

STEP 2

~~To move screens~~ To give randome name to screenshot we have to call function RandomString.make(length) & store it in var having return type String.

```
String p = RandomString.make(3);
```

STEP 3

To move SS from default location, first we have to set path of new folder so we create object of ss having return type file & pass path of folder as arg.

```
File d = new File("path // name" + p + ".jpg");
```

STEP 4

```
FileHandler.copy(S, d);
```

Page No.:
Date:

Program

```
package demo;
public class test
{
 public static void main (String [] args)
  {
    System.setProperty ("webdriver.chrome.driver","p
    WebDriver driver = new ChromeDriver ();
    driver.get ("Url");
    driver.findElement (locator).click ();
    File S = ((TakeScreenshot) driver).getScreenshotAS
                        (OutputType.FILE);
    String p = RandomString.make(3);
    File d = new File ("path // name" + p + ".jpg");
    FileHandler.copy (S,d);
  }
}
```

Lecture 28
29-8-19

Excel Sheet.

1) Fetch data from Excel (single data)

We store input require for automation testing i.e., test data in Excel.

	0	1	2	3
0		abc		
1			xyz	
2				
3				

rows = row

column = cell

so index of abc = 0, 1

index of xyz = 1, 2

To run Excel program we have to add apache poi jar file to liabrary like selenium.

STEP 1

Create object of fileInputStream class having parameter/ Argument as path of Excel.

```
FileInputStream a = new FileInputStream ("path");
```

STEP 2

To open sheet we need to call static method of WorkbookFactory class i.e., create().

```
WorkBookFactory.create(a)
```

To open perticular sheet we need to call getSheet method & pass sheet name as an argument.

```
WorkBookFactory.create(a).getSheet("sheet1")
```

To select row in which our data is present, we need to call method getRow() & pass row index as an arg.

```
WorkBookFactory.create(a).getSheet("sheet1").getRow(0)
```

Page No.:
Date:

To select column in which our data is present we need to call method i.e., getcell(1)

```
WorkBookfactory.create(a).getSheet("sheet1").getrow(0).
        getcell(1)
```

To fetch data which is of String type we need to call method getStringCellValue()

```
WorkBookfactory.create(a).getSheet("sheet1").getrow(0)
    getcell(1).getStringCellValue();
```

OR

To fetch data which is in number/integer format we need to call method getnumericCellValue()

```
WorkBookfactory.create(a).getSheet("sheet1").getrow
  .getcell(1).getnumericCellValue();
```

Then we have to store it in reff variable having return type String or int.

```
String s = WorkBookfactory.create(a).getSheet("sheet1"
    .getrow(0).getcell(1).getStringCellValue();

Sop(s);
```

Page No :
Date
youva

Program

	0	1
0		Prasad
1		
2		

```
package demo;
public class test
{
public static void main (String [] args)
{

~~system. SetProperty~~
FileInputStream f = new FileInputStream ("path of Excell");
String a = WorkBookfactory.create(f).getsheet("sheet1").getrow(0).
           getcell(1).getStringCellValue();

System.out.println(a);

}
}
```

2) Fetch multiple data from Excel

	0	1	2
0	1	2	3
1	4	5	6
2	7	8	9
3	10	11	12
4	13	14	15

← This data we have to fetch

STEP 1

Create object of fileInputStream & pass path as an argument.

```
FileInputStream f = new fileInputStream ("path");
```

STEP 2

To call getLastRowNum() method to get last index of row & store it in variable having return type as integer.

```
int a = WorkBookfactory . create (f) . getSheet ("sh
        . getLastRowNum ();
```

STEP 3

Call getLastCellNum() method to get last index of column & store it in variable having return type as integer.

```
int b = WorkBookFactory . create (f) . getSheet ("shee
    . . getrow (0) . getLastCellNum ();
```

Page No.
Date
YOUVA

STEP 4

Then use for loop:

```
for (int i=0 ; i<=a ; i++)
{
   for(int j=0 ; j<=b ; j++)
     {


     }
   sopln(" ");
}
```

STEP 5

* To ~~eat~~ get data we have to call WorkBook factory's method.

```
for (int i=0 ; i<=a ; i++)
{
   for (int j=0 ; j<=b ; j++)
    {
      String c = WorkBook factory.create(f).
                 getSheet("Sheet 1").getrow(i).
                 getcell(j).getStringCellValue();
      System.out.print(c);
    }
  System.out.println( );
}
```

* If there is no data present then it will show Null Point Exception *

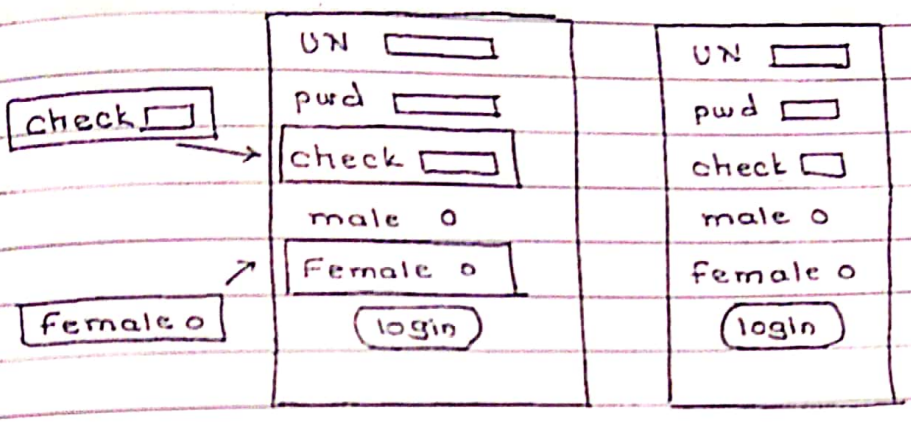
Page No.:
Date:
YOUVA

Program

```
package demo;
public class test
{
  public static void main (String[] args)
  {
   FileInputStream a = new fileInputStream ("path");

   int b = WorkBookfactory.create (a).getSheet
           ("sheet1").~~getrow~~ getLastRowNum();

   int c = WorkBookfactory.create (a).getSheet("sheet1
           .getrow (0).getLastCellNum ();

   for (int i=0 ; i<= b ; i++)
   {
     for (int j = 0; j<= bc ; j++)
      {
        String d = WorkBookfactory.create (a).
                   getSheet("sheet1").getrow(i).
                   getcell (j).getStringCellValue();
        System.out.print (d);
      }
   System.out.println ();
   }
  }
}
```

Lecture 29
30-8-19

iframe




```
<html>
  <body>
      UN <input type = 'text' >
      pwd <input type = 'password' >
          <iframe id = 'abc' name = 'xyz' >
            check <input type = 'checkbox' >
          </iframe>

      male <input type = 'radio' >
          <iframe id = 'abc123' name = 'xyz123' >
          f
      female <input type = 'radio' >
      </iframe>

      <button type = 'button' value = 'login' >
      </body>
  </html>
```

Sometimes some element like check are present so multiple page, then to write code for each page, instead of that only reff of it is provided to pages. So iframe is used.

Page No.:
Date:
YOUVA

1) When we try to create xpath of iframe it does not work & give exception NoSuchElement

2) To find path of iframe.

STEP 1

```
System.setProperty("webdriver.chrome.driver","pa
WebDriver driver = new ChromeDriver();
driver.get("url");
```

STEP 2

Find path of iframe.

```
driver.switchTo().frame("string ID");
```

OR

```
driver.switchTo().frame("string Name");
```

OR

```
driver.switchTo().fram(int index);
```

STEP 3

To switch to main page.

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

OR

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

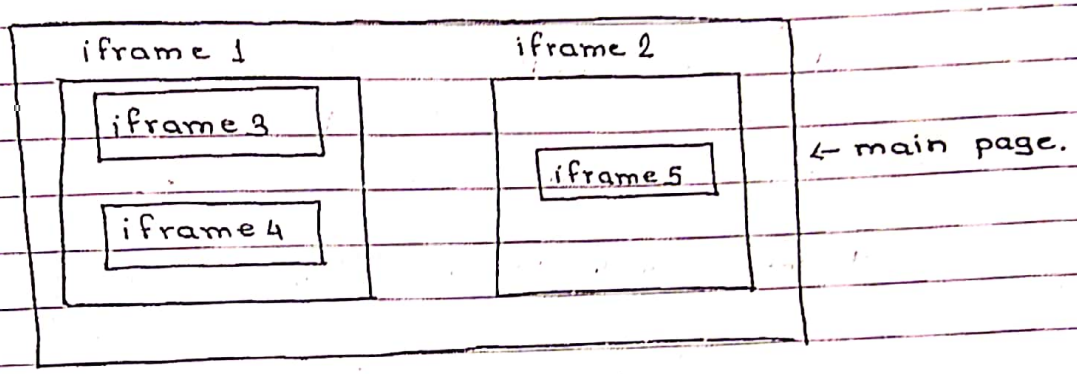
→ HOW TO HANDLE Iframe

First switch to iframe using syntax

driver.switchTo().frame("string ID");

Peform actions.

→



iframe 1 has iframe 3 & 4

if we have to go iframe 4 we go,

main page → iframe 1 → iframe 4

if we have to go to iframe 3 from iframe4

iframe 4 → iframe 1 → iframe 3

* for each iframe ID & Name are unique
* most of type ID & Name are used.

Page No.:
Date:

NOTES:-

1) Displaying web page as part of another web page is known as iframe.

2) iframe will be created using tagname iframe.

3) Procedure to handle iframe :-
To handle iframe we need to switch selenium ~~focum~~ focus from main page to frame by using:

driver . switchTo ( ) . frame ("string ID"/"str.Name"/ int index) ;

4) We can switch to frame by using 3 ways i.e., Name, ID, Index.

5) Once action perform on the component present in iframe selenium will not navigate by default to main page

6) To navigate frame iframe to main page we need to use methods like parent frame or default contaiNt

7) If we use the syntax :- driver . switchTo ( ) . parentframe then it will navigate from child frame to immidiate parent frame.

8) If we use the syntax :- driver . switchTo ( ) . default contaiNt ( ) , it will navigate from any child frame to main page.

9) We can identify iframe without checking HTML by rt8 click on that element & check option "view frame source" is appears or not.

Lecture 90
4-9-19

POP UP HANDLING

Page No.:
Date:
YOUVA

1) Pop up are small or separate window which will be displayed when we perform action on any component present in a web page.

2) These popup can be handle by Selenium Directly.

3) But sometimes we may need to use 3rd party tools to handle the popups.

4) Types of popup.
   - → Hidden division popup
   - → Alert popup
   - → Child browser popup
   - → Authentication popup
   - → File upload
   - → File download popup.

5) If we are able to inspect element present in popup then we can use selenium directly to handle that popup.

6) If we are unable to inspect element present in popup then we need to use 3rd party tool to handle that popup.

Page No.:
Date:

1) Hidden Division Popup

1) These popups are colorfull.
2) We can inspect element present in popup.
3) we can not perform drag & drop action.
4) As we can inspect element then using selenium we can handle it & no need to switch.

Ex: popup on flipkart.

2) Alert popup

1) We can not inspect element present in popup.
2) We can drag & drop this popup.
3) These popup will contain ok, cancel, button & result.
4) These type of popup also contain '?' or '!' Symbol

→ How to handle Alert popup.

1) To handle this popup we need to switch selenium focus from main page to alert popup by using

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

2) Alert is an interface which contain abstract method
   accept (use to click on ok button)
   dismiss (use to click on cancel button)
   getText (use to get text in alert popup)

→ To click on ok button:

```
alt.accept();
```

→ To click on cancel button:

```
alt.dismiss();
```

Page No.:
Date:
youvA

→ To get text present on popup.

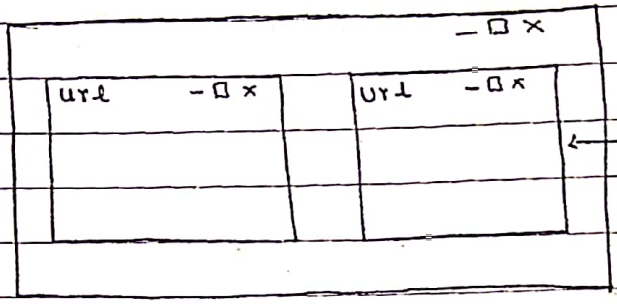
```
String a = alt.getText();
System.out.println(a);
```

program

```
package demo;
public class test
{
 public static void main (String[] args)
  {
    System.setProperty("webdriver.chrome.driver", "path");
    WebDriver driver = new ChromeDriver();
    driver.get("url");

    Alert alt = driver.switchTo().alert();
     alt.accept();
     alt.dismiss();
     String a = alt.getText();
     System.out.println(a);
  }
}
```

3) Child browser popup

1) These popup look like.



1) We can inspect element present in popup.
2) We can drag and drop.
3) These popup will contain address field (url), maximize, minimize, close options.

→ How to Handle Child browser popup?

1) To Handle Child browser popup we need to switch selenium focus from main page to child browser or window by using syntax:

```
driver.switchTo().window("String ID");
```

→ How to Handle multiple child browser popup?

1) If multiple child browser popup will occured, then we need to identify add field or ID's which is unique

```
Set<String> IDs = driver.getWindowHandles();
String A = Ids.get("ID value");

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

↓

Then perform actions using selenium.

Page No.:
Date:

* get Window Handles ():-

get address of main page as well as child popup & its return type is set<string> because it get multiple ID & set do not allow duplicate & ID are not duplicate.

* get Window Handle ()

get address of only main page so its return type is only String.

Leature 31
5-9-19

Page No.:
Date:

YOUVA

Customize list box

1) When any list box has "select" tag then use Select class actions on it.

2) When there is "div" tag then we use actions class & Keys class.

3) Function / methods of Actions class
   1) moveTo Element (reff variable of WebElement)
   2) Click () 5) Double click 6) context click.
   3) perform () 4) sendKeys ( )

4) Function / methods of Keys class
   1) ARROW_UP
   2) ARROW_DOWN
   3) ARROW_LEFT
   4) ARROW_Right
   5) ENTER

Q. How to handle customize list box:-

Ans:-

STEP 1

move focus to the element on which we have to perform action by finding its xpath & store it in variable having return type as WebElement.

```
WebElement a = driver.findElement ("xpath");
```

STEP 2

Create object of Actions class & in object constructor accept argument as ~~refference~~ object Name variable of WebElement.

```
Actions b = new Actions (driver(a));
```

STEP 3

Then we have to click on the object.

① `b.moveToElement(a).perform();`
`b.click().perform();`

OR

② `b.moveToElement(a).click().build().perform();`

OR

③ `b.click(a).perform();`

STEP 4

1) To move one option upward.

`b.sendKeys(Keys.ARROW_UP).perform();`

2) To move one option downward

`b.sendKeys(Keys.ARROW_DOWN).perform();`

3) To select option

`b.sendKeys(Keys.ENTER).perform();`

Page No.:
Date:

STEPS

1) To move arrow at top.
   → count how many options are present.
   → Ex. for month 12 options are there.

```
for (int i = 0 ; i <= 13 ; i++)
{
    a.SendKeys (Keys.ARROW-UP).perform();
    Thread.sleep (2000);
}
```

2) To move arrow at desire location.

```
for (int i=0 ;  i<=2 ;  i++)        → Select march. month.
 {
    a.sendkeys  (Keys.ARROW-DOWN).perform();
    Thread.sleep (2000);
 }
```

Page No.
Date:
youva

Program

```
package demo;
public class test
{
  public static void main (String [] args)
  {
    System.setProperty ("webdriver.chrome.driver", "path");
    WebDriver d = new ChromeDriver ();
    d.get ("url");

WebElement a = d.findElement (By.xpath ("\\ cj"));

  Actions b = new Actions (driver);

    b.click(a).perform();

  for (int i=0; i<=13; i++)
  {
      a.SendKeys (Keys.ARROW_UP).perform();
      Thread.sleep (2000);
    }

  for (int j=0; j<=2; i++)
   {
      a.sendKeys (Keys.ARROW_DOWN).perform();
      Thread.sleep (2000);
    }
  a.sendkeys (keys.ENTER).perform();
  }
}
```

Page No.:
Date:

Q. How to do double click on element.

Ans:-

STEP 1

Move focus to element on which we have to do work.

```
WebElement a = driver.findElement ("xpath");
```

STEP 2

Create object of Actions class which accept parameter as ~~ref var~~ obj name of WebDriver.

```
Actions b = new Actions (a);
                         driver
```

STEP 3

Click on the list in which we have to do actions.

```
b.click (a).perform ();
```

[This is single click]

STEP 4

To do double click we have to use double click function

```
b.doubleClick.perform ();
```

Program

```
package demo;
public class test
{
   public static void main (String [] args)
   {
      System.setProperty("webDriver.chrome.driver", "path");
      WebDriver driver = new ChromeDriver();
      driver.get ("Url");
      Actions a = new Actions (driver);
      a.doubleClick (b).perform ();
   }
}
```

WebEle b= findEle

Page No.:
Date:
YOUVA

Q. How to do Right-click on element.

STEP 1

Identify the element on which we have to do work.

```
WebElement a = driver.findElement(" ");
```

STEP 2

Create object of Actions class which accept argument as reff variable of WebElement driver.

```
Actions b = new Actions (driver);
```

STEP 4

To right click we have to use contextClick class function of Actions.

```
b.contextClick (a).perform();
```

Program

```
package demo;
public class test
{
 public static void main (String[] args)
 {
    System.setProperty (" webdriver.chrome.driver", " ");
    WebDriver a = new ChromeDriver();
    driver.get(url);
    WebElement z = driver.findElement ( );
    Actions b = new Actions (a);
    b.contextClick (z).
     b.contextClick (z).perform();
 }
}
```

Page No.:
Date:
YOUVA

Q. How to drag & drop element.

Ans:-

STEP 1

IDentify source on which we have to do operation.

```
WebElement a = driver.findElement ("xpath");
```

STEP 2

```
Actions b = new Actions (driver);
```

STEP 3

Identify new ~~p~~ element as destn.

```
WebElement c = driver.findElement ("xpath");
```

STEP 4

```
b.dragAndDrop (a, c).perform();
```

Lecture 32
6-9-19

Find Total Number of Links in Web Page.

Page No.:
Date:
YOUVA

1) Each link in web page has tag name a.

2) when we use //a then it will select each link one by one.

3) To focus on multiple links we use findElements method.

4) Return type of findElements is List <WebElement>

Q. How to find total Number of links in web page.

Ans:-

STEP 1

move focus to the elinks present in web page.

```
List<WebElement> a = driver.findElements(By.xpath("//a"));
```

STEP 2

Then find size of total number of links present.

```
int b = a.size();
System.out.println(b);
```

STEP 3

To print each link one by one use for loop.

```
for(int i=0 ; i<=b-1 ; i++)
{
    String c = a.get(i).getText();
    System.out.println(c);
}
```

Page No.:
Date:
YOUVA

Program

```
package demo;
public class test
{
  public static void main (String[] args)
   {
     ~~List <WebElement>~~ =
      System . setProperty ("webdriver.chrome.driver","");
       WebDriver driver = new ChromeDriver();
       driver.get ("url");

List<WebElement> a = driver.findElements (By.xpath("//a"));
 int b = a.size();
 System.out.println (b);


for (int i=0;  i<= b-1 ; i++)
{
   String c = a.get(i).getText();
   System.out.println (c);
 }
}
}
```

Page No.:
Date:

8. print all links present in web page in asc order.

ans:- * use TreeSet *

```
package demo;
public class test
{
 public static void main (String [] args)
 {
    System.setProperty ("webdriver.chrome.driver", "path");
    WebDriver driver= new ChromeDriver ();
    driver.get ("url");

List <WebElement> a = driver.findElements (By.xpath ("//a"));
TreeSet T = new TreeSet ();

int b = a.size ();

for ( int i=0; i<=b-1 ; i++)
{
   String c = a.get (i).getText ();
   T.add = c ;
 }

Iterator I = T.iterator ();
while ( I.hasNext ())
{
    System.out.println (I.next ());
 }
}
}
```

Page No.:
Date: one.

Identify checkbox & select them one by one.

Q. How to handle checkbox

Ans:-

STEP 1

Identify checkbox & find xpath of it & select that attribute which is common for all.

```
List<WebElement> a = driver.findElements(By.xpath("commonatt
```

STEP 2

Find size

```
int b = a.size();
```

STEP 3

use for loop & click function.

```
for(int i=0; i<= b-1 ; i++)
{
    a.get(i).click();
    Thread.sleep(2000);
}
```

Page No.:
Date:

→ program

```
package demo;
public class test
{
 public static void main (String [] args)
 {
   System.setProperty ("webdriver.chrome.driver", "path");
   WebDriver d = new ChromeDriver ();
   dr.get ("url");

 List (WebElement > l = d.findElements (By.xpath ( ));

  int b = l.size();

 for (int i=0 ; i<= b-1 ; i++)
 {
    l.get(i).~~get~~ click ();
    Thread.sleep(2000);
  }
 }
}
}
```

Lecture 33
9-9-19

Page No.:
Date:

Auto Suggesion

1) Auto suggesslons are like when we type something on google's search bar, it shows suggesion., while doing testing we have to test it.

2) Search box has tag name ul
Auto suggession has tag name li

Q How to handle Auto suggession

Ans:-

STEP 1

Focus on the search bar on which we have to do operation.

```
driver. findElement (By. xpath ("normal xpath")). sendKeys
                                                ("redmi");
```

STEP 2

Now site will show auto suggesions.
So we have to move focus to the options which are displayed.

```
List <WebElement > a = driver . findElements
                    (By. xpath ("//ul[ ]//li"));
```

STEP 3

Now we have to find size of how many options presse

```
int. size = a. size();
```

Page No.:
Date:
YOUVA

STEP 4

Now we have to declare a string who's value is same as expected result.

```
String exp = "Redmi note 7 pro";
```

STEP 5

Now use for loop & insid for loop use if.

```
for (int i=0 ; i<= size-1 ; i++)
{
    if (a equal
    String x = a.get(i).getText();
    if (x equals (exp))
    {
      S.o.p ("pass");
      break;
    }
    else
    {
      S.o.p ("fail");
    }
}
```

Page No.:
Date:

Program

```
package demo;
public class test
{
  public static void main (String[] args)
  {
     System . setProperty ("webdriver . chrome . driver", "path");
     WebDriver  driver = new ChromeDriver ( );
     driver . get ("url");

driver . findElement (By . xpath ("normal path" )) . sendKeys("redmi");

List <Web element> a = driver . findElements (By . xpath ("//ul[]\\li");

     int size = a . size ( );
      String exp = " redmi 7 ";
for (int i=0 ; i<= size-1 ; i++)
{
   String x = a . get (i) . getText ( );

   if (x equals (exp))
   {
      S.O.P (pass);
      break;
   }
}
}
}
```

Lecture :- 34
10-10-19

Page No.:
Date:
YOUVA

Framework

1) Types of framework
   1) Data Driven
   2) Keyword Driven
   3) Hybrid (combintn of both)

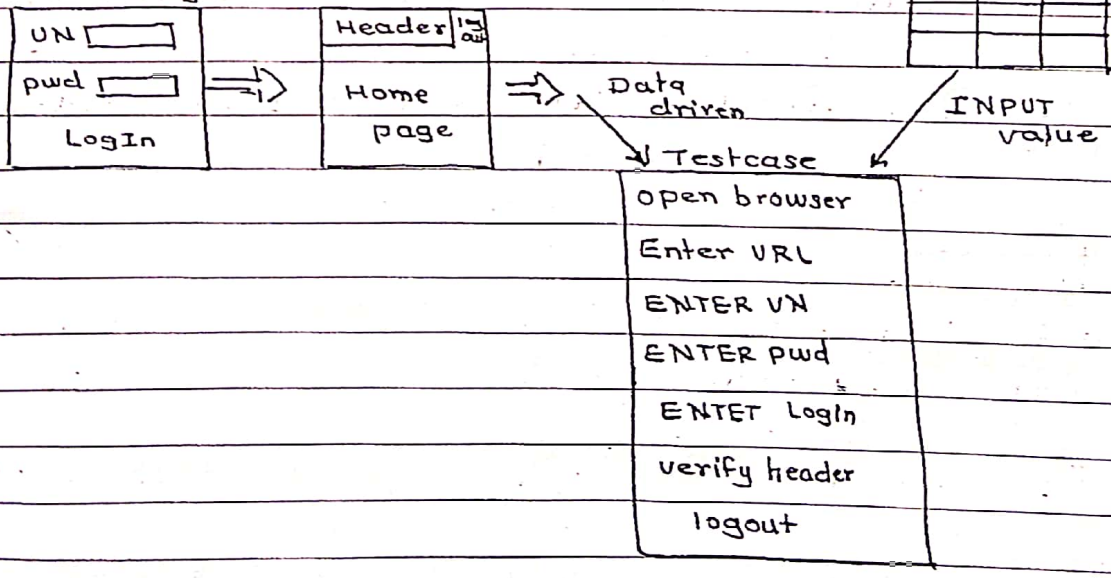
① Data Driven

Suppose application have multiple fields & we have to give them input by fetching data from excel sheet then we use data driven framework.

Data can be store in :
- Excel Sheet
- CSV
- Test ING data provider.

Example :-

actiTime login



Page No.:

FETCH DATA FROM EXCEL & put into desire location

[In same class]

STEP 1

1) open browser

```
System.setProperty("webdriver.chrome.driver", "path");
WebDriver driver = new ChromeDriver();
```

STEP 2

2) Enter URL

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

STEP 3

3) open Exell file.

```
FileInputStream excel = new FileInputStream("path");

Sheet a = Workbookfactory.create(excel).getSheet("Sheet1");
```

STEP 4

Store row & colum index in variable

```
String b = a.getrow(0).getcell(0).getStringCellValue();
                    //username
String c = a.getrow(1).getcell(0).getStringcellValue();
                    //password
```

STEP 5

Identify element which we want to handle & pass ~~row~~ variable to them.

```
driver.findElement(By.xpath(" ")).sendkeys(b);

driver.findElement(By.xpath(" ")).sendKeys(c);
```

Page No.:
Date:
YOUVA

STEP 6

To verify, find Header x path & use is display.

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

boolean e = d.isDisplayed();
S.O.P(e);
```

STEP 7

Then find path for logout button & click.

```
driver.findElement(By.xpath(" ")).click();
```

Page No.:
Date:

Program

```
package demo;
public class test
{
  public static void main (String [] args)
  {
    System.setProperty ( );
    WebDriver driver = new ChromeDriver();
    driver.get ("url");

    FileInputStream f = new FileInputStream ("path");
    Sheet s = Workbookfactory.create(f).getSheet("Sheet1");

    String UN = s.getrow(0).getcell(1).getStringCellValue();
    String pwd = s.getrow(1).getcell(0).getStringCellValue();

  driver.findElement(By.xpath(" ")).sendkeys(UN);
  driver.findElement(By.xpath(" ")).sendkeys(pwd);
  driver.findElement(By.xpath(" ")).click();

  WebElement E = driver.findElement ("Header xpath");
  boolean result = E.isDisplayed ();
  S.o.p (result);

  if (result == true)
  {
    S.o.p ("pass");
  }
  else
  {
    S.o.p ("fail");
    }
3 }
```

Page No.:
Date:

[fetch data by using code reusability]

```
package demo;
public class test
{
  public static String excel (int a, int b)
  {
  FileInputStream c = new FileInputStream ("path");
  String d = workbookfactory.create (a).getSheet ("Sheet1")
           .getrow (a).get cell (b).getStringCellValue();

    return d;
  }
}
```

```
package demo;
public class test1
{
  public static void main (String[] args)
  {
    System.setProperty ( );
    WebDriver driver = new ChromeDriver ();
    driver.get ("url");

    String e = test.excel (0,0);
    driver.findElement (xpath).sendKeys (e);

    String f = test.excel (1,0);
    driver.findElement (xpath).sendKeys (f);

    String g =
    driver.findElement (login).click ();
```

Lecture 33
11-9-19

Page No.:
Date:
YOUVA

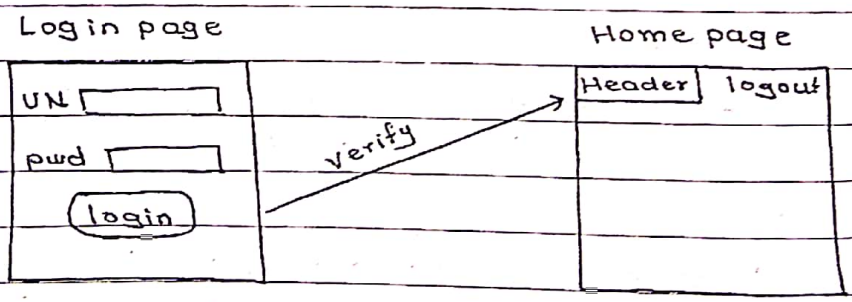
Page Object ~~Package~~ module (pom) with page factory

1) In normal java programming constructor are mainly use to initialize data members or variables,

like ex:

```
package demo;
public class test
{
  int a;          → declaration
  test
  {
    a=10;         } initialization
  }
  public void testI
  {
    S.O.P(a);     } usage.
  }
}
```

2) Page object module (pom) with page factory



Test cases

1) open login page
2) Enter username
3) Enter password
4) click on login button
5) verify Header on Home page
6) click on logout.

1) In normal program if we have to check this functionality then we use same class to check username, password Header.

2) But in page object module in test cases, test element related to how many number of web pages that many number of regular class we required.
Ex: In this testcase 2 pages are associate then 2 regular class we required.

3) Then in those regular classes we initialize, & & perform action (write functionality) on elements of web page.

4) Then we have to call these functionalities (regular method) from main method.class

~~main method = Test class~~

main class = Test class
Regular class = Pom class

5) we have to follow naming convensions:
Ex: → If we are checking login page of ActiTime website then class name is ActiTimeLoginPage

→ If we have to give input (sendKeys) then we use "set"
Ex. SetActiTimeLoginPageUsername

→ If we have to do click on any button then we use 'click'
Ex: ClickActiTimeLoginPageLoginButton

Page No.:
Date:

~~Beela~~

→ concepts use:-

1) Encapsulation
2) Annotation.

① Encapsulation

whenever in oop, we have to make any data, member of class usable for only that class that time we declare it as private. This is known as ~~private:~~ encapsulation.

② Annotation

Annotation contain some code, whenever we use annotation, then at time of execution that code is get executed.

Ex: @findBy (xpath = "xpath Expression")

This is the annotation we use it contains

driver.findElement (By.xpath ("xpath Ex"));

webdriver object	method of webdriver	By class contain method called as locators	method of by class i.e., xpath

Page No.:
Date:

1) Declaration of Variable

Rules:-

1) we have to declare variable as private.
2) Name of variable should be related to element.
3) data type of variable is WebElement.
4) we have to use annotation & focus on element by finding its xpath.

@findBy (xpath = "xpath Expression") Private WebElement UN;

Annotation — Encapsulation

2) Initialization

1) For initialization we have to use constructor.
2) constructor should be declare Public so that it can be accessible from other package in same project.
3) i) Then we have to pass driver (obj of WebDriver) to constructor.
   ii) This is because we have to create object of this class in main method & controller of browser is to WebDriver so to use data members we have to ~~eat~~ pass object of web driver here in constructor.

4) Then to initialize variable we have to use pagefactory class. This class contain initElements method which is static method.

5) This method will initialize element & by identifying their component by using @find By annotation.

6) parameters to ~~the~~ this method is driver, this ('this' is becz we use data members of same class which are ...

Page No.:
Date:

datatype of driver

```
public ActiTimeLoginPage (WebDriver driver)
{
    pagefactory.initElements (driver, this);
}
```

regular class — static method

3) Usage

1) To make action (write function) we need regular methods.

2) Number of methods = Number of variable declared

method Name

```
public void setActiTimeLoginPageUsername ()
{
    UN.sendkeys ("username string");
}
```

variable — web element method.

Program ① For login page

```
package demo;
public class ActiTimeLoginPage
{
    @FindBy (xpath = "xpath Expression") private WebElement UN;

    @FindBy (xpath = "xpath Expression") private WebElement pwd;

    @FindBy (xpath = "xpath Expression") private WebElement login;


    Public ActiTimeLoginPage (WebDriver driver)
    {
        pagefactory.initElements (driver, this);
    }


    public void setActiTimeLoginPageUsername ( )
    {
        UN.sendKeys ("prasad");
    }

    public void setActiTimeLoginPage Password ( )
    {
        pwd.sendKeys ("kadam");
    }

    public void clickActiTimeLoginPage LoginButton ( )
    {
        login.click ( );
    }
}
```

Page No.:
Date:

② For Home Page.

```
package demo;
public class ActiTimeHomePage
{
  @FindBy (xpath = "xpathEx") private WebElement Head;
  @FindBy (xpath = "xpathEx") private WebElement logout;

  public ActiTimeHomePage (WebDriver driver)
  {
    pagefactory.initElements (driver, this);
  }

  public void VerifyActiTimeHomePageHeader ()
  {
    Head- boolean present = Head.isDisplayed();
    s.o.p(present);
  }

  public void ClickActiTimeHomePageLogoutButton()
  {
    logout.click();
  }
}
```

Page No.:
Date:
YOUVA

⑤ For Test class

```
package demo1;
public class TestClass
{
   public static void main (String[] args)
   {
      System.setProperty (-11-);
      WebDriver driver = new ChromeDriver();
      driver.get ("url");

      ActiTimeLoginPage login = new ActiTimeLoginPage (driver);
      Login.set ActitimeLoginpageUsername();
      Login.set Act-11- Password();
      Login.click -11- LoginButton();

      ActiTimeHomePage Home = new ActiTime HomePage (driver)
      Home.verify -11- Header ();
      Home.click -11- LogoutButton();
   }
}
```

Lecture: 36
Date 12-8-19

Page No.:
Date.
YOUVA

→ Pom class by taking i/p from exel.

For Excel

```
package demo;
public class ~~test~~Excel
{
    public static String data (int a , int b)
    {
        FileInputStream data = new FileInputStream ("path");
        String data1 = workbookfactory.create (data)
                        .getSheet ("Sheet1").getrow(a).
                        getcell(b).getStringCellValue();
        return data1 ;
    }
}
```

For ActiTime Login

```
package demo;
public class ActiTimeLogin
{
    @FindBy (xpath= " ") private WebElement UN;
    @FindBy (xpath= " ") private WebElement pwd;
    @FindBy (xpath= " ") private WebElement login;

    public ActiTimeLogin (WebDriver driver)
    {
        pagefactory.initElements (driver , this);
    }
```

Page No.:
Date:

```
public void SetActitimeLoing Username (String user)
{
    UN.sendKeys (user);
}

public void SetActitimeLogin Password (String pass)
{
   pwd.sendKeys (pass);
}

public void Click ActiTime Login Button ():
{
   login.click();
}
}
```

For ActiTime Home page

```
package demo;
public class ActiTime Home
{
   @findBy (xpath = " ") private WebElement Header;
   @findBy (xpath = " ") private WebElement logout;

   public ActiTime Home (WebDriver driver)
   {
   pagefactory.initElements (driver, this);
   }
public void $Verify ActiTimeHome HeaderImage ()
{
   boolean verify = Header.IsDisplayed();
   S.o.p (verify);
}
```

Page No.:
Date:

```
public void click ActiTimeHome logoutbutton()
{
   logout.click();
}
}
```

For Test class

```
pub
package demo;
public class Test
{
   public static void main (String[] args)
   {
      System.setProperty("webdriver.chrome.driver",path
      WebDriver driver = new ChromeDriver();
      driver.get("url");
      String user data = excel.data(0,0);
      String pass data1 = excel.data(1,0);

      ActiTime_login loginpage = new ActiTimelogin(driver);
       loginpage.setActiTimeloginUsername(user);
       loginpage.setActiTimeloginPassword(pass);

      ActiTimeHome Homepage = new ActiTimeHome(driver);
       Homepage.setVerifyActiTimeHeader Image();
       Homepage.clickActiTimelogoutButton();
   }
}
```

Page No.:
Date:
youva

NOTES

1) Page object module

i) It is Java designed pattern use for designing of class in test scripts.

2) Pom follows encapsulation concept.

↳ Data member should declare globally with access level private. [private WebElement UN ;]

↳ Initialize within the constructor with access level public.

```
public ActiTimeLogin (WebDriver driver) {
    pagefactory.initElements (driver, this);
}
```

↳ Utilize within a method with access level public.

```
public void set () {
    UN.sendKeys ( );
}
```

3) Number of data members that need to be created under page object module class will depend on number of components that need to be handle in web page.

i.e., number of variable = number of component in page.

4) Pom class do not contain main method, so to run main method pom class we require main method, that main method is called test class.

5) Test class will contain all navigation step to test an application.

Page No.:
Date:
YOUVA

2) pagefactory

1) It is a class which contains static method like Init element.

2) To initialize data member in page factory we need to use, Init element method within the constructor.

pagefactory.InitElements(driver, this);

3) Init Elements method will initialize data member like username, password by identifying each component present in a web page by using @FindBy annotation which take locator type as an i/p.

@FindBy(xpath = " ") private WebElement UN;

3) Working of pagefactory

1) While executing test script, init element method will convert all the data members @FindBy annotation to find elements.
This process is known as early initializatn.

@FindBy (xpath = " ") private WebElement variable

↓ convert

WebElement variable = driver.findElement(By.xpath(" "));

Page No.:
Date:
YOUVA

2) To perform action on components we have to call the methods.

3) Before performing each action, initElement method will identify component present or not, then it will do complete initialization, this process is known as late initialization.

4) Disadvantage of Page object module

1) It will initialize all the data members before performing actions.

2) If page contain hidden elements, then it will not displayed while pom initializing, so it will throw no such element exception.

3) To overcome drawback of pom we need to use pom with page factory. which is an extension of pom.

Lecture 37
Date - 13-9-19

Test Ng.

1) How To Install

Eclipse
↓
Help
↓
Marketplace
↓
Find [Test NG]
↓
Install
↓
Restart Eclipse.

2) Test NG is Java framework method.

3) It will generate report of test. [pass, fail, skip]

4) Test Suite → Execute multiple test class one by one.

5) Execute fail test case:
   - ↳ If there are 5 suit & each contain 5 test class.
   - ↳ If 5 test case get fail, then we analyze them. correct them.
   - ↳ Then to recheck it we run suite, suit has fail test case file.
   - ↳ Then suit will execute only those fail test case.
   - ↳ fail test case file is named as fail.xml.

Page No.:
Date:

3) Topics in TestNG

1) Annotation
2) Key word
3) Verification
4) Report
5) Test suit.

4) Annotation

Suppose we have to print a message then in regular we have to create main method. But in TestNG we do not require main class.

```
package demo;
public class TestNG
{
    @Test    ← After annotation there should be a non st. class.
    public void test1()
                    ↑ Test case Name
    {                                    } Test case
        S.O.p ("Hii");
    }

    @Test
    public void test2()
    {
        S.O.p ("fail");
        Assert.fail();    ← To fail Test case.
    }
}
```

Page No.:
Date:

1) As Test NG generate report for Test cases. But system.out.println(" ") do not print in report.

2) To print on report we have to use static method log of Reporter class by passing message & True to it.

```
Reporter.log("message", true);

@Test
public void test1()
{
  Reporter.log("message", true);
}
```

2) How To open Emailable Report

1) Rig click on project.
↓
2) Refresh
↓
3) go to last folder testoutput.
↓
4) open folder
↓
5) emailable Report.HTML
↓
6) rig click
↓
7) open with browser.

Page No.:
Date:

3) Test cases

1) Precondition

Open browser
Username
password
login

} @Before method

2) Test

Header verify

} @Test

3) Post condition

logout.

} @After method

```
package demo;
public class TestNG
{
    @Before method
    public void precondition()
    {
        S.o.p (open browser);
        S.o.p (username);
        S.o.p (password);
        S.o.p (login);
    }
    @Test
    public void test()
    {
        Reporter.log (Header, true);
    }
    @Aftermethod
    public void postcondition(){ S.o.p (logout);}
```

Page No.
Date: YOUVA

Flow of Execution

@ Beforemethod
@Test
@ Aftermethod

* If there are two @test then as per alphabetical order they will execute.

Then flow of execution

Beforemethod
Test1
Aftermethod
Beforemethod
Test2
Aftermethod.

* If Before fail then execution stop.

4) Beforeclass & Afterclass

1) Before class & After class execute only once.
2) we write open browser, object creation, url. in Beforeclass.
3) we write browser close code in Afterclass.

Flow of Execution

Beforeclass
Beforemethod
Test
After ~~class~~method
Afler Class.

Page No.:
Date:
youva

5) Test NG program to take ilp from excel with pom.

① Create a class contain Excel.
② Create a class for login
③ Create a class to Home.

```
package demo;
public class TestNG
{
    @ WebDriver driver ;
      ActiTime login  login ;
      ActiTime Home  Home;

    @ BeforeClass
    public void pre ()
    {
        System.setProperty (" ");
        driver = new ChromeDriver();
        driver.get("url");
        login = new ActiTime login (driver);
        home = new ActiTimeHome (driver);
    }
    @ BeforeMethod
    public void login ()
    {
        String user = excel.data(0,0);
        String pwd = excel.data(0,1);
        login.setUsername(user);
        login.setPassword(pwd);
        login.clickbutton();
    }
```

Page No.:
Date:
YOUVA

```
@Test
public void verify(){
{
    home.verifyHeader();
}

@After method
public void logout()
{
    home.clicklogout();
}

@AfterClass
public void post(){
{
    driver.close();
    login = null;
    home = null;
}
```

} To free memory.

Lecture 38
Date - 14/9/19

Page No.:
Date:
YOUVA

KeyWords

→ Types

1) Priority
2) invocationCount
3) enabled = false
4) timeOut = time in millisecond
5) dependsOnMethods = {"method name"}

① Priority = integer

When we write two @Test methods & when at the time of execution their execution sequence is as per their alphabetic sequence.

To execute as per our sequence we define priority to them:

1) Priority can be by default zero (0) i.e., @Test
2) priority can be +ve integer i.e., @Test (priority = 1)
3) priority can be -ve integer i.e., @Test (priority = -1)
4) priority can be duplicate.

priority can not be decimal or variable.

```
package demo;
public class testClass
{
  @Test (priority = 1)
  public void first_method ()
  {
     Reporter.log ("message" , true);
  }
  @Test (priority = 2)
  public void second_method()
  {
     Reporter.log ("message" , true);
  }
```

② invocationCount = integer

When we have to execute same test case multiple time, then instead of writing @Test multiple time, we pass this keyword to @Test

```
package demo;
public class TestClass
{
    @Test (invocationCount = 5)          ← execute 5 times.
    public class test ()
    {
        Reporter.log ("message", true);
    }
}
```

③ enabled = false

If test class contains multiple test methods to skip one of the test method execution we need to use keyword false.

```
package demo;
public class TestClass
{
    @Test (enabled = false)
    public class test ()
    {
        Reporter.log ("message", true);
    }
    @Test ()
    public class testing ()
    {
        Reporter.log ("message", true);
    }
}
```

Page No.:
Date:
YOUVA

④ timeOut = time in milliseconds

If a test class contain multiple test methods, if one of the test method is time consuming to execute then TestNG by default fail that test method & execute other test methods, which can be possible using timeOut.

```
package demo;
public class TestClass
{
    @Test (timeOut = 5000)
    public void Testmethod()
    {
        Reporter.log ("message", true);
        Thread.sleep (6000);
    }
}
```

⑤ dependsOnmethod = {"method-name"}

If a test method execution depends on multiple test method then we need to use depends on method.

```
package demo;
public class Testclass
{
  @Test
  public class test()
  {
    Reporter.log ("message", true);
  }

  @Test (dependsOnMethod = {"test"})
  public class testclass()
  {
    Reporter.log ("message", true);
  }
}
```

Lecture 29
Date : 16-9-19

Page No.:
Date:

NOTES

1) Test NG

It is a Java Unit framework. use for writting or designing of test classes.

Advantage of Test NG

1) It generate report.
2) multiple test cases executed at same time (test suit)
3) We can execute only fail test cases using fail.xml file.

2) Annotation

1) @Before Class

This annotation is use for execution of few test methods before executing test class.
It get executed only once.

2) @After Class

This annotation is use for execution of few test method After executing test class.

3) @ Before method

It is use for execution of few test method before executing every test method with an annotation @Test.

4) @ After method

Use for execution of few test method after executing every test method with @Test.

5) @Test

use for execution of tert case / method.

Page No.:
Date:
YOUVA

INHERITANCE CONCEPT IN FRAMEWORK

We write open browser, url in @BeforeClass, if we have 100+ test cases, then if there is change in url then we have to change it at each class, it is time consumming so we define a regular class & extends that code into a test class, use method with written type.

```
package demo;
public class ~~test~~ open
{
    WebDriver driver;
    public WebDriver browser ( )
    {
        System. setProperty ("");
        driver = new ChromeDriver ( );
        driver. get ("url");
        return driver;
    }
}
```

```
package demo;
public class open2 extends open
{
    ~~@~~ WebDriver driver;
    @Before class
    public void ~~open~~ precondith ( )
    {
        driver = browser ( );
        -"-
    }
    -"-
}
```

Lecture 40
Date: 17/9/19

Page No.:
Date:

HOW TO TAKE SCREENSHOT of FAIL TEST CASES. ONLY.

```
package demo;
public class Screenshot
{
    public static void take (WebDriver driver , int TCID)
    {
        File src = ((TakeScreenshot)driver).getScreenshotAs
                                        (OutputType.file);

        File dest = new File ("path//name" + TCID + ".jpg");

        FileHandler.copy (src, dest);
    }
}
```

```
package demo;
public class TestNG
{   int TCID;
    @Test
    public void Testing ()
    {   TCID=1;
        Reporter.log ("message" , true);
        Assert.fail();
    }
    @AfterMethod
    public void postcondtn (~~ITestReport~~ ITestResult result)
    {
        If ( ~~Assert~~ ITestResult.failure == result.getStatus())
        {
            screenshot.take (driver , TCID)
        }
    }
}
```

Page No.:
Date:
YOUVA

ITestResult

To take screenshot of fail test cases, we first require result of test cases.
This result store in listner., this listner is known as ITestResult.
We have to use ~~After~~ ITestResult in Aftermethod.

ot As
file)
'jpg'