## LOCATORS

**CSS Selectors**

1. **using class Attribute = . class value…( . represent class attribute)**
2. **using id Attribute = # id value…( # represent id Attribute)**
3. **using Attribute = tag name [attribute = attribute value] OR tag name [attribute =’ attribute value’]**

**this works without tag name as well……. [attribute = attribute value]**

using css methods

1. **Prefix (starts with)(^) = [attribute ^= attribute value]**
2. **Suffix (ends with) ($) = [attribute $= attribute value]**
3. **Substring (contain) (\*) = [attribute \*= attribute value]**

## ****XPATH****

**Relative X-path**

1. **Normal = //tag name [@attribute = ‘attribute value’]**
2. **Using and operator = // tag name[@attribute = ‘attribute value’ and @attribute = ‘attribute value’ ]**
3. **Using or operator = // tag name[@attribute = ‘attribute value’ or @attribute = ‘attribute value’ ]**
4. **Using index value = (// tag name [@attribute = ‘attribute value’])[index value]…ex (//input[@type= 'text'] )[2]**

**Relative X-path Methods**

1. **Contains () = // tag name [contains (@attribute, ‘attribute value’)]**
2. **Stars-with () = // tag name [stars-with (@attribute, ‘attribute value’)]**
3. **Text () = // tag name [text() = ‘ inner text ’]**

**Launch Browser**

|  |  |  |
| --- | --- | --- |
| **BROWSER** | **WEB DRIVER** | **Keys** |
| **Chrome** | **Chromedriver.exe** | **“webdriver.chrome.driver”** |
| **Firefox** | **Geckodriver.exe** | **“webdriver.gecko.driver”** |
| **Edge** | **Edgedriver.exe** | **“webdriver.edge.driver”** |

**System. setProperty ( key, “value”); ……………………Value = Location of web browser exe**

**ChromeDriver obj = new ChromeDriver(); OR WebDriver obj = new ChromeDriver ();**

**METHOD 2 … *using web driver manager API***

**WebDriverManager.chromedriver.setup();**

**ChromeDriver obj = new ChromeDriver(); OR WebDriver obj = new ChromeDriver ();**

**Browser Command**

1. **driver.close() = close current opened browser**
2. **driver.quit() = close all opened browser**

**.get Commands**

**….( return String value)**

1. **driver. get (“url”); ……open Mentioned Url**
2. **driver . getTitle (); …….. Fetch currently open web page title**
3. **driver . getCurrentUrl (); ………Fetch Currently open web page url**
4. **driver. getPageSource(); ………. Fetch Currently open web page HTML dom**
5. **webElement.getText(); …….. Fetch inner text value in mentioned web element**
6. **webElement.getAttribute(“ attribute name ”); ……fetch mentioned attribute value ,from mentioned webElement**

**Active Commands**

**…….(Return Boolean value)**

1. **Webelement.isDisplayed(); ……. Return Boolean value True if webelement is displayed on webpage**
2. **Webelement.isEnabled(); ……. Return Boolean value True if webelement is Enabled on webpage**
3. **Webelement.isSelected(); ……. Return Boolean value True if webelement is Selected on webpage**

**Navigation Commands**

1. **driver.navigate().to(“ URL ”); ………….open url in current browser window**
2. **driver.navigate().forward(); …………..open next web page in current browser window**
3. **driver.navigate().back(); …………..open previous web page in current browser window**
4. **driver.navigate ().refresh (); ………….. refresh web page**

**GERNAL COMMANDS**

1. **driver.findElement( locator type(“locator”) ); ……..return single webElement**
2. **driver.findElements( locator type(“ locator”) ); ……..return multiple webElement**

**input box**

1. **webElement .clear(); …………………………… Clear the text box**
2. **webElement .sendKeys(“ value ”); …………….. send mentioned value into input text box**

**DropDown**

**Using selenium Select class**

**Select obj = new Select ( web Element(drp button locator)) ;**

1. **Obj.selectByVisiableText(“text”) ; ……… text – this is inner text present in select tag list**
2. **Obj.selectByIndex( index value ); …………. Index value(int) – index value of select tag**
3. **Obj.selectByValue(“value of value attribute”); ……. Basically this methods works when in select tag value attribute present**

**Ex :- < option value = “10”> Algeria</option> ….. here value of value attribute is 10**

**4. Obj.getOptions(); = return List of webElement**

**Handles Window**

**(return String OR list of String)**

**Open Url In New Tab OR In New Window**

**1. String tab = Keys.chord(Keys.CONTROL,Keys.ENTER); = open a link in a new tab**

**………………..driver.findElement(By.LocatorType (" Locater ")).sendKeys(tab); ………..link locator**

**2. driver.switchTo().newWindow(WindowType.WINDOW); = it will open new url in new window**

**……….driver.get(" URL "); ………… url**

**Windows handling Commands**

1. **driver.getWindowHandle(); = Fetch the currently open Window Id (Return single window id only)**
2. **driver.getWindowHandles(); = Fetch the All open Window Id (Return List of window id )**
3. **driver.switchTo().window(WinId); = Switch one window to Another (Different Tab’s in same browser)**

**Using Switch To Command Handle the windows**

**Ex:- String mulwindowID = driver.getWindowHandles();**

**ArrayList<String> mulwindowIDlist = new ArrayList (mulwindowID);// get list of windowsID**

**String parentWinId = mulwindowIDlist.get(0); //return parent window id**

**String childWinId = mulwindowIDlist.get(1); //return child window id**

**driver.switchTo().window(parentWinId);**

**System.*out*.println("Parent Window Title:" + driver.getTitle());**

**driver.switchTo().window(childWinId);**

**System.*out*.println("Child Window Title:" + driver.getTitle());**

**Alerts**

1. **driver.switchTo().alert().accept(); = Switch to alert and Accept the alert**
2. **driver.switchTo().alert().dismiss(); = Switch to alert and Dismiss the alert**
3. **driver.switchTo().alert().getText(); = Fetch Alert window Text**
4. **driver.switchTo().alert().sendKeys(“value”); = sent string value in alert input box**

**Authentication pops :- Need To Enter in a WebApplication (username and password)**

**Syntax :- http:// username : password@.......Application url……**

**Ex:- driver.get( “ http://usr : pwd @ …url… “);**

**Permission pops :- block the notification pop ups (for chrome browser)**

**Syntax :- ChromeOptions obj1 = new ChromeOptions();**

**obj.addArguments(“ –disable-notifications”);**

**WebDriver obj2 = new ChromDriver(obj1);**

**Iframes**

**Some of the web elements present in frame to Access them , First We need to, switch to the iframe by locting them And Perform different operations**

**For Switching to iframe we used different locators like…. By id OR By name, By index, By WebElement**

**EX: -**

1. **driver.switchTo().frame("id or name"); = Using By id or By name**
2. **driver.switchTo().frame(0); = Using By index we Switch to the iframe ( html coading frame's count from zero index)**
3. **driver.switchTo().frame(WebElememt); = using webElemnt**
4. **driver.switchTo().parentFrame(); = switch to parent frame**
5. **driver.switchTo().defaultContent(); = Inner frame to Root of HTML**

**Actions class operations**

**Mouse operations**

**Actions act =new Actions (driver);**

1. **act.contextClick(WebElement).perform(); = perform right click Action**
2. **act.doubleClick(WebElement).perform(); = Perform Double click Action**
3. **act.dragAndDrop(Source Web-Element , Target Web-Element).perform(); = perform drap and drop action**
4. **act.moveToElement(WebElement).click().perform(); = Mouse hover action + click action perform on mentioned web Element.**

**Key-Board Actions**

1. **act.sendKeys(Keys.ENTER).perform(); = press & release….. enter key**
2. **act.sentkeys(“a”).perform(); = press & release…. “a” key**

**Combination Keys :-**

**CTRL + A**

**act.keysDown(Keys.CONTROL); = press down ctrl key**

**act.sendKeys(“a”); = press & release…. “a” key**

**act.keysUp(Keys.CONTROL); = release Ctrl key**

**act.perform(); = To carry out All above Action ………… build().perform() And .perform()…works as a same**

**Capture Screenshot**

**We need to use File Class + takeScreenShot interface**

**1. For a Full Page :-**

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

**TakeScreenshot obj = driver ;**

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

**TakeScreenshot obj = (TakeScreenshot) driver ;**

**(While using WebDriver interface we need to casting)**

1. **obj.getScreenshotAs(OutPut.FILE); = this method take screen shot as file format**

**Ex:-**

**File src = obj.getScreenshotAs(OutPut. FILE);**

**File trg = new File (“ location to be save + file name + file Extension”); = obj creation + deicide save file location**

**File.Utils.copyFile(src ,trg ); = used for copy to folder**

**2. For a Particular Web Element:-**

**WebElement ele = locator();**

**File src = ele . getScreenshotAs(OutPut. FILE);**

**File trg = new File (“ location to be save + file name + file Extension”); = obj creation + deicide save file location**

**File.Utils.copyFile(src ,trg ); = used for copy to folder**

**Java Script Executor**

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

**JavascriptExecutor obj = driver;**

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

**JavascriptExecutor obj = (JavascriptExecutor) driver;**

**(While using WebDriver interface we need to casting)**

**Syntax :-**

1. **obj.executeScript (“arguments [0] . style . border = ‘3 px solid red’ ” , web Element ); = highlight the mentioned web element**
2. **obj.executeScript (“ window . scrollTo(0, document.body.scrollHeight)”) ; = Scroll down the page**
3. **obj.executeScript (“ window . scrollTo(0,- document.body.scrollHeight)”) ; = Scroll Up the page**
4. **obj.executeScript(“ arguments[0] . click(); ” , webElement) ; = click on webElement**
5. **obj.executeScript(“ document . body . style . zoom = ‘50%’ ”) = Page zoom 50%**

**WAIT’S**

**1. Implicit Wait :- Wait 10 sec for all HTML gets load**

**driver.manage ().timeouts() . implicitlyWait (Duration .ofSeconds(10));**

**2. Explicit Wait :- Wait 10 sec for Specified Web Element gets load**

**WebDriverWait obj1 = new WebDriverWait (driver , Duration. ofSeconds(10)) ;**

**obj1.until (ExpectedConditions.visibilityOfElementLocated (By. (locator) ) );**

**3. Fluent Wait:- Wait 10 sec for Specified Web Element gets load with customize polling time and exception handling**

**Wait<WebDriver> fwait = new FluentWait<WebDriver>(driver)**

**.withTimeout(Duration.ofSeconds(15))**

**.pollingEvery(Duration.ofSeconds(5))**

**.ignoring(NoSuchElementException.class);**

**WebElement foo = fwait.until (new Function<WebDriver, WebElement>()**

**{ public WebElement apply (WebDriver driver)**

**{ return driver.findElement(By.name("q")); } } );**

**Up Load File**

**Using send Keys Method : -**

**this method use when uploadFile Choose or Select button ..in there html code written like...< type ='file' >**

**then and then it only work other wise is not**

**……………….driver.findElement (By . locator Type (" Locater ")) . sendKeys ( " File path + File name + . File Extension " );**

**Using Robot class : -**

**it will work with out any condition**

**1. Find the upload file path**

**2. copy those path in clip-board using..**

**3. then paste this path in file selected box using..**

**4. and press enter**

**Robot rb = new Robot ();**

**copy file to clip board**

**StringSelection ss = new StringSelection ("File path + File name + . File Extension " (where file) );**

**Toolkit.getDefaultToolkit ().getSystemClipboard ().setContents ( ss, null);**

**perform control + v action to paste file in selected box**

**3. Enter key press and release**

**rb.keyPress(KeyEvent.VK\_ENTER);**

**rb.keyRelease(KeyEvent.VK\_ENTER);**

**2. CONTROL + V... key Release**

**rb.keyRelease(KeyEvent.VK\_CONTROL);**

**rb.keyRelease(KeyEvent.VK\_V);**

**1. CONTROL + V... key press**

**rb.keyPress(KeyEvent.VK\_CONTROL);**

**rb.keyPress(KeyEvent.VK\_V);**

**Excel**

**1. Create object of file And read the file**

**FileInputStream fis = new FileInpurStream (“ file Location + File name + .File Extension ”);**

**2. Create object of Work book Factory And get file**

**Workbook wb = new WorkbookFactory . create ( fis) ;**

**1. wb.getNumberOfSheets () ; = it will return all sheet count present in Mentioned Excel Work-Book And it count**

**start from 1 not from zero (return type is int)**

**2. wb.getSheetName ( Index number ) ; = it will return sheet NAME for that we need to mentioned index number**

**here indexing start from zero (return type is String)**

**3. wb.getSheetIndex (“ String ”); = it will return you SHEET INDEXING.. for that we need mentioned sheet name**

**(return type is int)**

**3. To Work on particular Sheet : -**

**SHEET is interface Here we choose particular sheet for work For that we have method...wb.getSheet("sheet name");**

**And it will return you object.. and using this object we perform different different operations**

**Sheet sh = wb.getSheet("Sheet Name"); = Select mentioned Sheet**

1. **sh.getLastRowNumber () ; = his method return int value of rows present in sheet It will count from 0 and count**

**EMPTY cell Also.**

1. **sh.getPhysicalNumberOfRows (); = This method return int value of active(Filled with data) rows present in sheet**

**It will count from 1 and DOES NOT count Empty cell.**

**4. ROW interface: - s.getRow (row count) return the row as object**

**Row rw = sh.getRow (1); = point out first row**

1. **rw.getPhysicalNumberOfCells(); = method count only active cell present in mentioned rows(It will count from 1)**
2. **rw.getLastCellNum(); = method count EMPTY cell as well present in mentioned columns (It will count from 1)**

**5. CELL interface:-**

**to get cell value first we need to point the cell at particular row After pointing to cell... call the particular...cell methods with respective to their data type ..rw.getCell(1).....just point cell which is present at row 1(s.getRow(1))**

**Cell cel = rw.getCell(0); = just point cell which is present at row 1(s.getRow(1))**

1. **cel. getStringCellValue(); = To get only string type data**
2. **cel. getNumericCellValue(); = To get only Numeric type data**
3. **cel. getBooleanCellValue(); = To get only Boolean type data**
4. **cel. getCellFormula(); = To get only Formula type data**

**Properties File**

**Properties pro = new Properties ();**

**FileInputStream fis = new FileInputStream (“Property File path + Property File Name + Property File Extension ”) ;**

1. **pro.load (fis) ; = load the file.**
2. **pro.getProperty (“ Key ”); = Always mentioned key here + this method return ‘value’ in the form of String.**

**TEST NG (Next Generation)**

**Annotations :- Control The Flow Of execution .**

1. **@Test :- Executes Methods Below According To ASCII Value, A-Z (Uppercase) = 65-90, a-z ( Lowercase ) = 97 -122**
2. **@BeforeSuite :- It will run only once before all tests in the suite Are executed.**
3. **@BeforeTest :- It will be executed before the all Test Annotations in particular class or irrespective class( when run file through XML).. and it will runs at once**
4. **@BeforeClass:- it will be executed before class, but in the mentioned class only**
5. **@BeforeMethod:- Before every test Method which is present in same class.**
6. **@AfterMethod :- After every test Method which is present in same class.**
7. **@AfterClass :- It will be executed After class, but in the mentioned class only**
8. **@AfterTest :- It will be executed before the all Test Annotations in particular class or irrespective class( when run file through XML).. and it will runs at once**
9. **@AfterSuite :- It will run only once before all tests in the suite Are executed**

### ****Test Attributes :-****

1. **@Test (priority = 1) :- Set method execution priority it would be (–ve, 0 , or +ve ), Always execute low priority first, default priority will be zero**
2. **@Test ( enabled = false ) or @Test (priority = 1, enabled = false) :- by default it true ,make it false it will won’t execute method**
3. **@Test(groups={"group name"}) :- executes group of methods together, through <group tag > under xml file**
4. **@Test (invocationCount = num ) :- is used to run single test case multiple time…. (thread pool Size = num, jst like thread count)**

### ****Parallel testing :-****

**It will help us to execute……… Tests, Classes , & Methods…. Simultaneously**

**Syntax :- <test name = "Parallel Tests" parallel = "classes" thread-count = "2">**

**1. Using parallel='classes'**

Class 1

Class 2

**ex :- thread Count = 2 ,**

**Thread 1 Allocated to class 1**

**T1 = M1, T1 = M2, T1 =M3**

**& Thread 2 Allocated to class2**

**T2 = M1, T2 =M2 .**

* **All thread will be equally distributed to all classes mentioned in <tag>**

M2

M1

* **If odd thread num i.e. thread count =3 the first two Allocated for class1**
* **And remaining one allocated for class 2**
* **Those thread complete first they help rest of them**

**2. Using parallel = ‘Tests’**

**Syntax :- <suite name = "Parallel Testing Suite" parallel = "tests" thread-count = "2">**

**Suite**

Test Tag 2

Test Tag 1

Class1

Class2

M3

M1

Class 1

Class 2

M2

M1

M2

M1

M2

M1

**Thread Count :- 2,**

**\* Thread 1 Allocated to test tag 1**

**T1 = M1 (Class1) , T1 = M2(class1), T1 = M3(class1)**

**T1 = M1 (class2) , T2 = M2 (class2)**

**\* & thread 2 Allocated test tag 2**

**T2 = M1(class1) , T2 = M2(class1)**

**T2 = M1 (class2)**

**3. Using Methods :- <test name = "Parallel Tests" parallel = "methods">**

* **All Thread will be allocated to first class as per the Sequence in testing.xml file.**
* **In class According to ascii value test methods will be executed.**
* **Once it will be execution gets complete in first class then it will move to the next class as per the sequence in testing.xml file**

**Ex:- thread Count = “ 2 ”,parallel = ‘methods’**

Class 1

Class 2

**T1 = M1 (class1), T2 = M2 (class1), T1= M3(class1)**

M2

M1

**T2= M1 (class2), T1 = M2(class2)**

M2

M1

M3

### ****Parameters :-****

* Test-NG parameters are present in XML file
* Parameters defined in <test tag > applicable to only that particular tag but if parameters defined in < suite tag > applicable to all test tag’s in the xml file
* **Syntax** :-

**Suite tag :-**

**<suite name="Suite\_Name">**

**<parameter name = "browser" value = "chrome"/>**

**<parameter name = "url" value = "https://opensource-demo.orangehrmlive.com/"/>**

**<test name="test name">**

**<classes>**

**<class name=" Package Name.ClassName " />**

**</classes>**

**</test >**

**</suite>**

**Test Tag :-**

**<suite name="Suite\_Name">**

**<test name="test name">**

**<classes>**

**<parameter name = "browser" value = "chrome"/>**

**<parameter name = "url" value = "https://opensource-demo.orangehrmlive.com/"/>**

**<class name=" Package Name.ClassName " />**

**</classes>**

**</test >**

**</suite>**

**public class para\_ex1**

**{**

**@BeforeClass**

**@Parameters({"browser", "url"}) // parameter need to decleare in XML file and pass the key .**

**public void setup(String browser, String application)**

**{**

**if(browser.equalsIgnoreCase("chrome"))**

**{**

**WebDriverManager.chromedriver().setup();**

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

**}**

**else if ( browser.equalsIgnoreCase("firefox"))**

**{**

**WebDriverManager.firefoxdriver().setup();**

**WebDriver driver = new FirefoxDriver();**

**}**

**driver.get(application);**

**}**

## ****Data Provider :-****

1. **It has only one ‘attribute name ’ if you do not Specify the name of attribute then the data provider’s name will be same as the corresponding method name.**
2. **Return type is 2 dimensional array of object ,if the data provider returns an array of 2\*3 objects , the corresponding test case will be invoked 2 times with 3 parameter each time**
3. **It will executes inside the class as well as out side the class but not for the out side the package**
4. **data provider is just provided data from one method to another method.**

**Syntax :-**

In the different class bt with in pckg

Within class

**@Test(dataProvider = “xyz”)**

**public void loginTest (String email ,String pwd )**

**{**

**System.out.println (email +” ”+pwd)**

**}**

**// data provider define and declaration**

**@DataProvider (name = “xyz”)**

**public Object[][] getdata()**

**{**

**Object [][] data = {{“abc@gmail.com” , “123”},{“xyz@gmail.com”,”456”}}**

**return data;**

**}**

**@Test ( dataProvider = “xyz”, dataProviderClass = Class name of data provider method .class )**

**public void loginTest (String email , String pwd )**

**{**

**System.out.println (email +” ”+pwd)**

**}**

### ****Reporter :-****

**Is an inbuilt class in test-NG which is available under the org.testng pckg this methods to log msg that will be include in the HTML reports generated by testing. ( Jst write it down in method it will appear in the reports)**

**Syntax :-**

**Reporter.log (“ msg”) ;**

### ****Assertion : -****

1. **Assertions in test NG are verify that actual result and expected result matched or not , And returns Boolean value**
2. **There are two types Soft Assertion And Hard Assertion**

Soft Assertion:-

1. **It is allow to next line code after found Exception (it wont terminate the program ), found Exception handle at the end of mentioned method completion.**
2. **It is non Static method so we need to create object of it**
3. **At end of method it is mandatory to call assertion method obj.assertAll(); ….so they can handle all exception found at method**
4. **We can add customize msg explicitly & It’s work when Boolean value = false**

**Syntax :-**

**SoftAssert obj = new SoftAssert ();**

**obj.assertEquals(Excepted Result, actual Result, "customize msg");**

**obj.assertAll(); ….At the end of mentioned method**

Hard Assertion:-

1. **It won’t allow next line code after found exception**
2. **It is static method so there is no need to create object of it**
3. **Do not need to call any kind of method at end mentioned method, To the wrap up the exception handling**

**Syntax :-**

**Assert.assertEquals (Excepted Result, actual Result, "customize msg”);**

**Few method used in both Soft and hard assert**

1. **.assertEquals (Ex result, Actual result ); :- It takes three string arguments and checks whether both are equal, if not it will fail the test and throws the message which we provide.**
2. **.assertTrue(condition, message) :- It takes one boolean argument and String message. It Asserts that a condition is true. If it isn't, an Assertion Error, with the given message, is thrown.**
3. **.assertFalse(condition, message) :- It takes one boolean argument and String message. It Asserts that a condition is false. If it isn't, an Assertion Error, with the given message, is thrown.**
4. **.assertNotNull([Object](http://java.sun.com/j2se/1.5.0/docs/api/java/lang/Object.html?is-external=true" \o "class or interface in java.lang) object ,**[**String**](http://java.sun.com/j2se/1.5.0/docs/api/java/lang/String.html?is-external=true)**message) :- Asserts that an object isn't null. If it is, an Assertion Failed Error, with the given message, is thrown.**
5. **.assertNull([Object](http://java.sun.com/j2se/1.5.0/docs/api/java/lang/Object.html?is-external=true" \o "class or interface in java.lang) object,** [**String**](http://java.sun.com/j2se/1.5.0/docs/api/java/lang/String.html?is-external=true)**message) :- Asserts that an object is null. If it is not, an Assertion Failed Error, with the given message, is thrown.**

**Listeners**

**It is listens to the desired events and executes the methods accordingly.**

**IT listeners is an interface in which predefine 7 methods to use this simply implemented interface in the class**

**And override this predefined methods.**

1. **onStart( ITestContext obj (Result) ); =**  **it will executed before all test method get started (at once).**
2. **onTestStart (ITestResult obj (Result)); = is invoked only when every test method gets started .**
3. **onTestSuccess ( ITestResult obj (Result)); =** **method is executed on the success of a test method.**
4. **onTestFailure(ITestResult obj (Result)); = invoked when test method fails**.
5. **onTestSkipped (ITestResult obj (Result)); =** **run only when any test method has been skipped.**
6. **onTestFailureButWithinSuccessPercentage(ITestResult obj (Result)); = This method is invoked each time when the test method fails but within success percentage.**
7. **onFinish(ITestContext obj (Result)); =it will execute after all test method finished .**

**There are two ways to use the listeners , 1. Using @Listeners , 2. Using xml file**

**1. inherited defined interface method class, Where to be use**

**@Listeners (pckg name + .class name (xyz) + .class extension )**

**public class class name {}**

**0. Create class And implement interface , override methods**

**public class class name(xyz) implements ITestListener**

**{ override methods… }**

**2. using xml file**

**<suite name="Suite\_name">**

**<listeners>…..mentioned interface methods defined class**

**<listener class-name="pckg name . class name (xyz)" />**

**</listeners>**

**<test name="Test\_Listener" thread-count="5" >**

**<classes>…mentioned all classes where to use**

**<class name= "pckg name . class name" />**

**</classes>**

**</test>**

**</suite>**

Page Factory:-

1. **Its class provided by selenium web Driver to support the page object model pattern.**
2. **@FindBy () Annotation used for to locate and declare web Element using different types of locators**
3. **.initElements() it is static method in page factory class, used for to initialize the all web element’s which is located by @FindBy Annotation .**

**Syntax :-**

**@FindBy (locator Type = “ locator ” ) WebElement obj ;**

**Class Abc {**

**WebDriver driver;**

**Abc(WebDriver d)**

**{ driver = d;**

**PageFactory.initElements(driver, this);**

**}**

**}**

**POM :- Page Object model**

**By Attribute**

1. **Id = driver.findElement (By.id ("id value"));**
2. **Name = driver.findElement (By.name (" name value"));**
3. **linkText = driver.findElement (By.linkText("inner Text"));**
4. **partialLinkText = driver.findElement (By.partialLinkText (" half inner Text"));**

**\*\*\*\* Tag and className are not unique so they will use for make list or where multiple data need to be use \*\*\*\*\*\***

1. **className = driver.findElement (By.className(" classNameAttribute value "));**
2. **tagName = driver.findElement (By.tagName("tag value"))**

**css Selector = (.) represent class Attribute , (#) represent id Attribute ,**

**driver.findElement (By.cssSelector(" # Id Attribute value "));**

**driver.findElemet (By.cssSelector (" .class Attribute value "));**

**driver.findElement (By.cssSelector("[attribute name = Attribute value]"));**

**\*\*\*\*\*\*\*\* OR using tag + cssSelector \*\*\*\*\***

**driver.findElement (By.cssSelector (" tag name # id Attribute value"));**

**driver.findElement (By.cssSelector(" tag name . class Attribute value "));**

**driver.findElement (By.cssSelector("tagName[Attribute = Attribute Value]"));**

**\*\*\*\*\*\*\*\*\*\*\*\*\* combo\*\*\*\*\*\*\*\*\*\*\*\***

**tag + class + attribute**

**driver.findElement (By.cssSelector (" tag . classAttribute value [attribute = 'attribute value']"));**

**tag + id + attribute**

**driver.findElement(By.cssSelector ("tagName# Id Attribute Value [Attribute = Attribute Value]"))**

**\*\*\*\*\*\*\* css selector methods\*\*\*\*\*\*\*\*\*\*\***

**1. Start With ^ = [Attribute name ^= ' initial Attribute value'] , Ex :- name="firstname"**

**........ driver.findElement(By.cssSelector (" [ name ^= 'first']"));**

**2. End With $ =[Attribute name $= 'ends up Attribute value'] ex:-id="password\_step\_input"**

**........ driver.findElement(By.cssSelector('[ id $= 'input']'))**

**3. Contains with \*= [Attribute \*= 'Attribute value'], Ex :- class="inputtext \_58mg \_5dba \_2ph-"**

**........... driver.findElement(By.cssSelector(" [ class \*= '58mg '] "));**

**(used for find out multiple element)**

## ****XPATH****

**Relative X-path**

1. **Normal = //tag name [@attribute = ‘attribute value’]**
2. **Using and operator = // tag name[@attribute = ‘attribute value’ and @attribute = ‘attribute value’ ]**
3. **Using or operator = // tag name[@attribute = ‘attribute value’ or @attribute = ‘attribute value’ ]**
4. **Using index value = (// tag name [@attribute = ‘attribute value’])[index value]…ex (//input[@type= 'text'] )[2]**

**Relative X-path Methods**

1. **Contains () = // tag name [contains (@attribute, ‘attribute value’)]**
2. **Stars-with () = // tag name [stars-with (@attribute, ‘attribute value’)]**
3. **Text () = // tag name [text() = ‘ inner text ’]**

**Genric = //a[contains(@href ,'fashion-best-seller')]….xpath**

**[href\*='fashion-best-sellers']……css**