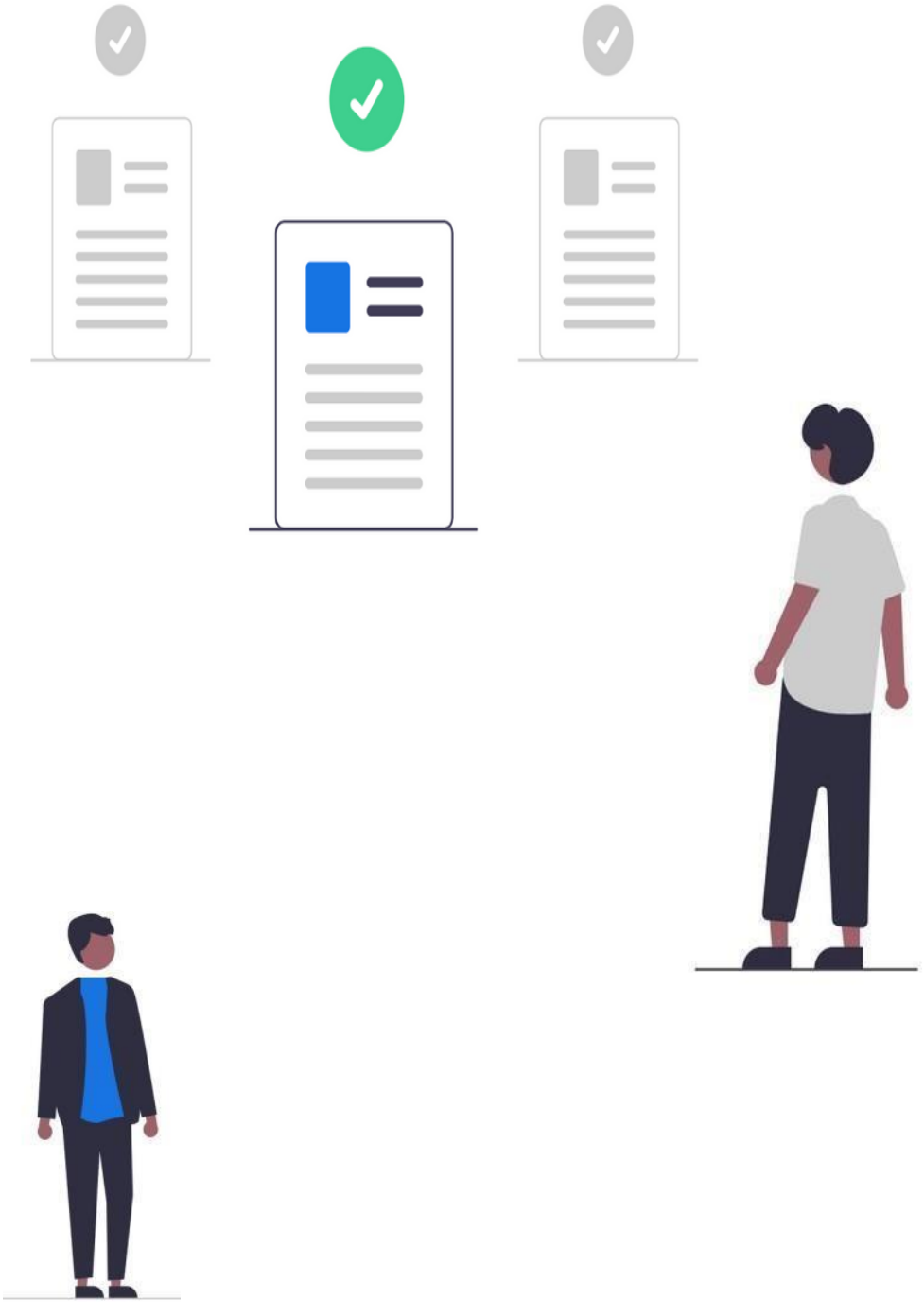
www.qastack.net
+92 300 1404 303

AUTOMATION TESTING NOTES



WHAT IS AUTOMATION TESTING?

Automation testing is the testing process which makes use of test automation tool and executes our test script. It compares our actual result with the expected result and reports it to the testers

Benefits:

1. Saves the time
2. Faster
3. Requires less manual effort
4. Accuracy will be more
5. Multitask
6. Requires less human resource

SELENIUM

- Selenium is an open-source automation testing tool which is used to automate web-based application.

Advantages of Selenium

1. Supports multiple programming languages like Java, C#, Ruby, Python, pearl etc.
2. Framework support
3. Free and Opensource
4. It supports multiple platform such as Windows, Linux, MacOS etc.
5. It have cross browser compatibility like Firefox, Chrome, IE, Safari etc.
6. Less hardware usage

Drawback:

We can't automate desktop application using selenium.

Selenium Components:

1. Selenium IDE
2. Selenium RC (Remote control)
3. Selenium WebDriver
4. Grid

Selenium IDE: -

- Selenium IDE comes with Firefox plugin, thus it supports Firefox browser only.

Selenium Grid: -

- Selenium Grid is used for parallel testing or distributed testing. It allows us to execute test scripts parallelly on different machines.

Selenium RC: -

- Selenium Remote Control (RC) is used to write test cases in different Programming languages
- In Selenium IDE, we can run the recorded scripts only in Firefox browser, whereas, in Selenium RC, we can run the recorded script in any browser like IE, Chrome, Safari, Opera and so on

Selenium WebDriver: -

- Selenium WebDriver is a tool used to automate testing for web application
- It allows us to execute tests against different browsers like Firefox, Chrome, IE & Safari
- Selenium WebDriver eliminated the use of Selenium Server thus making it work faster than RC

Latest version of selenium

selenium-server-standalone-3.141.59

To configure Selenium jar:

Create new project → Right click → New → New folder {**copy selenium jar file**}

↓

Name the folder (lib will be best practicing)

↓

Paste jar file

↓

Right Click jar file

↓

Build Path

↓

Add to Build Path

Browser	**Driver**	**Key**	**Class**

Chrome	chrome driver	webdriver.chrome.driver	ChromeDriver
Firefox	gecko driver	webdriver.gecko.driver	FirefoxDriver
Internet Explorer	ie driver	webdriver.ie.driver	InternetExplorerDriver

To configure Driver:

Create new project → Right click → New → New folder → Name the folder (driver will be best practicing) → Paste driver exe file → **{copy driver file}**

To configure browser:

System.setProperty(key , path of the driver);

- System is a class.
- ***setProperty()*** is a method in this we have to pass key and value.
- It is used to set the key and path location of driver.

WebDriver

- WebDriver is an interface.
- WebDriver is a web automation framework that allows you to execute your tests in different browser.
- WebDriver also enables you to use a programming language in creating your test script.
- It is mainly used for providing the connection between the browser and local system.
- It acts as a bridge.

Instantiate for webdriver:

WebDriver refName = new DriverClassName();

WebDriver Methods:

Method	Return Type	Description
driver.get("url");		To launch given URL in the configured browser
driver.gettitle();	String	To get the title of the webpage launched
driver.getCurrentUrl();	String	To get the URL of the current webpage
driver.quit();		To quit the browser
driver.manage().window().maximize()		To maximize the browser tab
driver.switchTo().alert()	Alert	Switch to alert

Sample Program:
==========

```
package org.day.one;

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

public class Sample {

	public static void main(String[] args) {

		//configure your browser
		System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A.
R\\eclipse-workspace\\SCalss1\\Drivers\\chromedriver.exe");

		//Instantiate for webdrivers
		WebDriver d=new ChromeDriver();

		//to maximize browser
		driver.manage().window().maximize();

		//to launch given url
		d.get("https://www.facebook.com");

		//to get the title of the webpage launched
		String title = d.getTitle();
		System.out.println(title);

		//to get the url of the current page
		String url = d.getCurrentUrl();
		System.out.println(url);

		//to quit the browser
		driver.quit();

	}

}
```

Locator:

When we need to perform any action in the browser, we have to find the locator. Locators used to find and match the elements of your page that it needs to interact with.

In By class all the locator methods are available

By→ Abstract Class **Package→** selenium.By

Static Methods of **By**

- **id**
- **name**
- **className**
- **xpath**
- **tagName**
- **linkText**
- **partiallyLinkText**
- **cssSelector**

WebElement

WebElement → Interface **Package** →selenium.WebElement

Anything **that** is present on the web page is a WebElement such as text box, button, etc. WebElement represents an HTML element. **Selenium WebDriver** encapsulates a simple form element as an object of the WebElement**.** It basically represents a DOM element and all the HTML documents are made up by these HTML elements.

WebElements in Selenium can be divided into different types, namely:

- **Edit box**: It is a basic text control that enables a user to type a small amount of text.
- **Link**: *link* is more appropriately referred to as a *hyperlink* and connects one web page to another. It allows the user to click their way from page to page.
- **Button**: This represents a clickable button, which can be used in forms and places in the document that needs a simple, standard button functionality.
- **Image**: It helps in performing actions on images like clicking on the image link or the image button, etc.
- **Text area**: It is an inline element used to designate a plain-text editing control containing multiple lines.
- **Checkbox**: This is a selection box or a tick box which is a small interactive box that can be toggled by the user to indicate an affirmative or a negative choice.
- **Radio button**: It is an option button which is a graphical control element that allows the user to choose only one predefined set of mutually exclusive options.
- **Dropdown list**: It is a graphical control element, similar to the *list* box, which allows the user to choose one value from the list. When this *drop-down list* is inactive, it displays only a single value.

To find Locator:

<input type="text" class="inputtext _55r1 _6luy" name="email" id="email" data-testid="royal_email" placeholder="Email address or phone number" autofocus="1" aria-label="Email address or phone number">

input → tag name (at first near to open arrow <)

Class → attribute name (left side of the equal symbol)

inputtext _55r1 _6luy → attribute value (right side of the equal, enclosed within double quotes)

Find locator

```
WebElement txt = driver.findElement(By.id("email"));
```

Methods of WebElement:

txt.sendKeys("sequence")		To pass any values in the textbox
btn.click()		To perform click option for button

txt.getText()	String	To print the text in the WebElement
txt.getAttribute("AttribiuteName)	String	To print the attribute value in the web element
txt.getAttribute("value")	String	To print the values we have passed in text box

1) By.id()

id value is "email". and it's a textbox, then we can find the locator by

```
WebElement txt = driver.findElement(By.id("email"));
```

2) By.name()

name value is "email". and it's a textbox, then we can find the locator by

```
WebElement txt = driver.findElement(By.name("email"));
```

3) By.className()

class value is "inputtext _55r1 _6luy". and it's a textbox, then we can find the locator by

```
WebElement txt = driver.findElement(By.className("inputtext _55r1 _6luy"));
```

```
package org.day.two;

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

public class One {

	public static void main(String[] args) {



	System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A.
R\\eclipse-workspace\\SDay2\\driver\\chromedriver.exe");

		WebDriver driver=new ChromeDriver();

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

		//to find locator of the user name
		WebElement txtUserName = driver.findElement(By.id("email"));
		//to pass values in the text box
		txtUserName.sendKeys("azarara321@gmail.com");

		//to find locator of the password field
		WebElement txtPassword = driver.findElement(By.name("pass"));
		txtPassword.sendKeys("123456789");

		//to find locator of login button
		WebElement btnLogin = driver.findElement(By.className("login"));
```

```
            //to click button
            btnLogin.click();
        }
}
```

4) By.xpath()

Xpath is one of the locator available in webpage.

/ → absolute path (It find the WebElement from the top of the downstream.)

// → relative path (It find the WebElement from that particular tag.)

Reason for going to Xpath:

- For validating the locator. (We can check this in webelements by **ctrl+F)**
- When id,classname,name is not present.

General syntax:

//tagName [@attributeName = ' attributeValue ']

If there is more than one attribute value is same for same attribute names in the web element then we have to go for index(matching with more than one loctor)

General syntax:

(//tagName [@attributeName = ' attributeValue ']) [2]

When we try to find locator, if only text is present there then we go for text

text()

//tagName [text( ) = ' text name ']

contains()

//tagName [contains (text( ) , 'partially text')]

contains() *using attributes*

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

Example Program

```
package org.com;

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

public class hai {
    public static void main(String[] args) {

        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspace\\Zpractice

        WebDriver driver=new ChromeDriver();
        driver.get("http://demo.automationtesting.in/Register.html");

        //to find xpath
        WebElement txtMail = driver.findElement(By.xpath("//input[@type='email']"));
        txtMail.sendKeys("abc@gmail.com");

        //to find path when more locators found
        WebElement txtFst = driver.findElement(By.xpath("(//input[@type='text'])[1]"));
        txtFst.sendKeys("Azar");

        WebElement txtLst = driver.findElement(By.xpath("(//input[@type='text'])[2]"));
        txtLst.sendKeys("AR");

        }

    }
```

```
package org.com;

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

public class hai {
    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\'

        WebDriver driver=new ChromeDriver();
        driver.get("https://www.facebook.com/");

        WebElement txtmail = driver.findElement(By.id("email"));
        txtmail.sendKeys("azarara321@gmail.com");

        //to print the entered values
        String mail = txtmail.getAttribute("value");
        System.out.println(mail);

    }
}
```

```java
package org.com;

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

public class hai {
    public static void main(String[] args) {

        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspace\\SDay

        WebDriver driver=new ChromeDriver();

        driver.get("http://greenstech.in/selenium-course-content.html");

        //contains text
        WebElement txtAdd = driver.findElement(By.xpath("//h6[contains(text(),'Greens')]"));
        String text = txtAdd.getText();
        System.out.println(text);

        //text
        WebElement txtAdd1 = driver.findElement(By.xpath("//p[text()='mail-info']"));
        String text2 = txtAdd1.getText();
        System.out.println(text2);

    }
}
```

==========

```java
package org.com;

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

public class hai {
    public static void main(String[] args) {

            System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspa

            WebDriver driver=new ChromeDriver();

            driver.get("http://greenstech.in/selenium-course-content.html");

            //contains text
            WebElement txtAdd = driver.findElement(By.xpath("//h6[contains(text(),'Greens')]"));
            String text = txtAdd.getText();
            System.out.println(text);

            //text
            WebElement txtAdd1 = driver.findElement(By.xpath("//p[text()='mail-info']"));
            String text2 = txtAdd1.getText();
            System.out.println(text2);
    }
}
```

DEBUG

- It is the step by step verification.
- We can easily identify the step where the code getting exception.

Steps for debug:

1. Set a break point(double click at the line number.
2. R+click
3. Debug as
4. Java application
5. Switch

F6=> stepover

F8=>close debug

Types of Debug

- Eclipse debugger.
- Firefox JavaScript debugger.
- Dynamic debugging technique
- On line debugging tool.

For page loading time issue

Thread.sleep(milliseconds);

Thread* => *class

sleep()*=>*static method

It throws ***Interrupted Exception***

1000 milli second = 1 second

ACTIONS

Actions → Class

Package → selenium.interactions

MouseOverAction

- When we place a mouse on some option it will display a list of subOption.
- For mouseOverAction we can use Actions class.

Declare Actions

Actions a = new Actions(WebDriver ref name);

Methods in Action

a.moveToElement(WebElement)		To move mouse point or curser to webelement
a.dragAndDrop(source,target)		For drag and drop option
a.sendKeys(source,sequence)		To pass any values in the textbox
a.sendKeys(char sequence)		To pass any values in the textbox
a.doubleClick(WebElement)		To perform double click
a.contextClick(WebElement)		To perform right click

The menu list disappears within the fractions of seconds before Selenium identify the next submenu item and perform click action on it.

So, it is better to use .perform() method.

Whenever Actions methods takes place we use or end that method with ***perform()*** to perform that method.

```
package org.com;

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

public class hai {
    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspa
        WebDriver driver=new ChromeDriver();

        driver.manage().window().maximize();
        driver.get("http://demo.guru99.com/test/drag_drop.html");

        Actions ac=new Actions(driver);

        //to perform drag and drop operation
        WebElement sour = driver.findElement(By.xpath("//a[text()=' BANK ']"));
        WebElement dest = driver.findElement(By.xpath("(//li[@class='placeholder'])[1]"));
        ac.dragAndDrop(sour, dest).perform();

        Thread.sleep(3000);

        WebElement sour2 = driver.findElement(By.xpath("//a[text()=' 5000 ']"));
        WebElement dest2 = driver.findElement(By.id("amt7"));
        ac.dragAndDrop(sour2, dest2).perform();
    }
}

        WebElement weight = driver.findElement(By.xpath("//a[text()='Weight Gainers']"));
        weight.click();
    }

}
```

ROBOT

- **Robot → class**
- **Package → java.awt**
- **Exception →AwtException(Abstract window toolkit)**

Robot class is a class which is used to perform the keyboard action in java.

Declare Robot

Robot r=new Robot();

Mehods of Robot:

r.keyPress()		To press any key
r.keyRelease()		To release key(whenever keyPress() takes place keyRelease() should also mentioned)
KeyEvent => Class(takes place inside the robot method where it consists of all keyboard keys).		

Example Program

```
package org.com;
import java.awt.AWTException;
import java.awt.Robot;
import java.awt.event.KeyEvent;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.interactions.Actions;
public class hai {
    public static void main(String[] args) throws AWTException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspac
        WebDriver driver=new ChromeDriver();
        driver.get("http://www.greenstechnologys.com/");
        Actions a=new Actions(driver);
        //Robot class declaration
        Robot r= new Robot();
        WebElement paragraph = driver.findElement(By.xpath("(//p[@style-size:18px;'])[2]"));
        //double click in actions
        a.doubleClick(paragraph).perform();
        //right click in actions
        a.contextClick(paragraph).perform();
        //for using down key
        r.keyPress(KeyEvent.VK_DOWN);
        r.keyRelease(KeyEvent.VK_DOWN);
        //for using enter key
        r.keyPress(KeyEvent.VK_ENTER);
        r.keyRelease(KeyEvent.VK_ENTER);
    }
}
```

ALERT

Alert → interface

Alert is a small message box displayed on the screen to give some information to the user. Alert and webpage are different Alert has no locators. When alert appeared first we need to switch into the alert to handle the alert, then only user can perform the next operation in the webpage.

To switch into the alert

Alert a=WebDriver.switchTO().alert();

Types of Alert

1. Simple Alert → Contains only ok button
2. Confirm Alert → Contains both ok and cancel button
3. Prompt Alert → Contains text box with ok and cancel button

Methods in Alert

r.accept()	**Accept the alert**
a.dismiss()	**Dismiss the alert**
a.sendKey()	**To insert the values**
a.getText	**To print the text in the alert**

Simple Alert

```
package org.com;

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

public class hai {
    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\

        WebDriver driver=new ChromeDriver();
        driver.manage().window().maximize();

        driver.get("http://demo.automationtesting.in/Alerts.html");

        //Switching into Simple alert
        Alert a = driver.switchTo().alert();

        //to accept in the alert
        a.accept();
    }

}
```

Confirm Alert

```
package org.com;

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

public class hai {
    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. F

        WebDriver driver=new ChromeDriver();
        driver.get("http://demo.automationtesting.in/Alerts.html");

        //Switching to confirm alert
        Alert a = driver.switchTo().alert();

        // accept the alert
        a.accept();
    }
}
```

Confirm Alert

```
package org.com;

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

public class hai {
    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\

        WebDriver driver=new ChromeDriver();
        driver.manage().window().maximize();

        driver.get("http://demo.automationtesting.in/Alerts.html");

        //Switching to Prompt  alert
        Alert a = driver.switchTo().alert();

        //passing the values in alert
        a.sendKeys("Azar");

        //accept the alert
        a.accept();
    }
}
```

JAVASCRIPT EXECUTOR

Java Script → Interface

Package → selenium.JavaScriptExecutor

- JavaScriptExecutor is an Interface that helps to execute JavaScript through Selenium Webdriver.
- In Selenium Webdriver, locators like XPath, CSS, etc. are used to identify and perform operations on a web page.
- In case, these locators do not work you can use JavaScriptExecutor. You can use JavaScriptExecutor to perform a desired operation on a web element.

To implement JavaScript

1. Type casting

   JavaScriptExecutor js = (JavaScriptExecutor) WebdriverRef.Name
2. Use methods in JavaScript

Methods of JavaScript

js.executeScript ("JavaSript code" ,WebElement reference);

JavaScript Codes

arguments[0].setAttribute('value','input txt')		To pass any input text in text box
return arguments[0].getAttribute('value')	String	Retrieve the values of user entered text
arguments[0].click()		For button click
arguments[0].scrollIntoView(false)		Scroll down
arguments[0].scrollIntoView(true)		Scroll up

When ever we want retrieve the user entered values the method returns Object. With that object reference name we can do upcasting to get string values to be printed.

String s1=(String)object;

Example

```
package org.com;

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

public class hai {
    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspace\\S
        WebDriver driver=new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.facebook.com/");
        JavascriptExecutor js=(JavascriptExecutor)driver;
        //pass values in the text box using Java Script
        WebElement txtEmail = driver.findElement(By.id("email"));
        js.executeScript("arguments[0].setAttribute('value','azarara321@gmail.com')", txtEmail);
        //retrieve the user entered text in the webelement
        Object object = js.executeScript("return arguments[0].getAttribute('value')", txtEmail);
        //upcasting
        String s1=(String)object;
        System.out.println(s1);
        WebElement txtPass = driver.findElement(By.id("pass"));
        js.executeScript("arguments[0].setAttribute('value','12345')", txtPass);
        //button click using Java Script
        WebElement btnLogin = driver.findElement(By.name("login"));
        js.executeScript("arguments[0].click()", btnLogin);
        }
    }
```

```
package org.com;

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

public class hai {
    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspace\\SDay7\\driv

        WebDriver driver=new ChromeDriver();
        JavascriptExecutor js=(JavascriptExecutor)driver;
        driver.get("http://toolsqa.com/");

        WebElement txtScroll = driver.findElement(By.xpath("//span[text()='Selenium Training Benefit']"));
        //scroll up
        js.executeScript("arguments[0].scrollIntoView(false)", txtScroll);
        //scroll down
        js.executeScript("arguments[0].scrollIntoView(true)", txtScroll);
    }
}
```

TAKES SCREENSHOT

TakesScreenshot → Interface

TakesScreenshot ts=(TakesScreenshot)WebDriverRef.Name;

Methods

File src = ts.getScreenshotAs(OutputType.FILE);

File → **Return type for *getScreenshotAs( );***

Steps:

1. Typecast
2. Store screenshot in default
3. Create a screenshot

The output format of the screenshot will be Base64, Bytes, Class, FILE.

In order to store the screenshot in our project folder

1. After taking the scerrnshot, create a file class
2. Create a folder in the package and give the path of the folder in the File class
3. In the path at last add the name of the screen shot.
4. Use ***FileUtils.copyFile(source, destination);*** to copy ***src*** file and past it in the desired project folder. ***Source*** → ref name of ***getScreenshotAs( ).***
5. copyFile() is a static method in the FileUtils class.

Example

```
package org.com;

import java.io.File;
import java.io.IOException;
import org.apache.commons.io.FileUtils;
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class hai {
    public static void main(String[] args) throws IOException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspace\\SDay7\\d

        WebDriver driver=new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("http://www.greenstechnologys.com/");

        //typecasting
        TakesScreenshot ts=(TakesScreenshot)driver;

        //take screenshot
        File src = ts.getScreenshotAs(OutputType.FILE);

        //copy from src and save it in destination folder
        File dest= new File("C:\\Users\\Azar. A. R\\eclipse-workspace\\SDay7\\Screenshot\\Greesn1.png");
        FileUtils.copyFile(src, dest);
    }

}
```

VISIBLITY OF WEBELEMENT

To check visibility of the WebElements

isEnabled() - method to check whether the web element is enabled or not.

isDisplayed() - method to check whether the web element is displayed(present) or not

isSelected() -method to check whether the web element is selectable or not

It is widely used in radio button, dropdown, checkbox

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

public class Task_02 {

    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspace\\
        WebDriver driver =new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.facebook.com/");
        WebElement btnLogin = driver.findElement(By.name("login"));
        //to check whether WebElement is displayed
        boolean displayed = btnLogin.isDisplayed();
        System.out.println(displayed);
        //to check whether WebElement is enabled
        boolean enabled = btnLogin.isEnabled();
        System.out.println(enabled);
        WebElement btnCreate = driver.findElement(By.xpath("//a[text()='Create New Account']"));
        btnCreate.click();
        Thread.sleep(3000);
        WebElement rdoGender = driver.findElement(By.name("sex"));
        rdoGender.click();
        //to check whether WebElement is selected
        boolean selected2 = rdoGender.isSelected();
        System.out.println(selected2);
    }
}
```

FRAMES

html embedded inside another html

When any locator is placed inside the frame we cannot directly access the locator.

First we need to switch into the frame, then only we can access frame.

To switch into frame

First we have to check frame is available in DOM or not.

R + click → view frame source

Or

Inspect→cntrl + F →//iframe or //frameset etc.

Methods to switch into frame (method-overloading)

driver.switchTo().frame(string id);

driver.switchTo().frame(string name);

driver.switchTo().frame(web element);

driver.switchTo().frame(index);

Methods to switch out of frame

To switch from current frame to immediate parent frame (frame inside frame concept)

driver.switchTo().parentframe();

To switch the control from any frame to main.

driver.switchTo().defaultContent();

Example

```
package org.com;

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

public class hai {
    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspace\\SDay

        WebDriver driver=new ChromeDriver();
        driver.manage().window().maximize();

        driver.get("https://netbanking.hdfcbank.com/netbanking/?_ga=2.176378149.1819882415.153388321

        //switching into frame using string id
        driver.switchTo().frame("login_page");
        Thread.sleep(2000);

        WebElement login = driver.findElement(By.xpath("//img[@src='/gif/continue_new1.gif?v=1']"));
        login.click();

        //to switch the contril from frame to main
        driver.switchTo().defaultContent();
    }
}
```

WINDOWS HANDLING

Whenever we execute any program it can access current window webelement only.

When we have multiple windows to switch control between windows we go for windows handling.

To switch into other window

driver.switchTo().window(String URL)

driver.switchTo().window(String title)

driver.switchTo().window(Window ID)

To find window ID

Parent id:

driver.getWindowHandle() → ***String***

Child id:

driver.getWindowHandles() → ***Set<String>***

```
package org.com;

import java.util.Set;

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

public class hai {
    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspace\\SD

        WebDriver driver= new ChromeDriver();
        driver.manage().window().maximize();

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

        WebElement txtSearch = driver.findElement(By.id("inputValEnter"));
        txtSearch.sendKeys("Hand sanitizer");
        WebElement btnSearch = driver.findElement(By.xpath("//span[text()='Search']"));
        btnSearch.click();
        WebElement btnPro = driver.findElement(By.xpath("(//img[@class='product-image ']) [1]"));
        btnPro.click();

          //to get parent window id
          String parentWind = driver.getWindowHandle();

          //to get all window id including parent
          Set<String> allWind = driver.getWindowHandles();

          //to switch into child window by iteration
          for (String cd : allWind) {

              if(!(parentWind.equals(cd))) {

                  //switch to child window
                  driver.switchTo().window(cd);
              }
          }

          WebElement btnAdd = driver.findElement(By.id("add-cart-button-id"));
          btnAdd.click();

          //to switch to parent window
          driver.switchTo().defaultContent();
    }
}
```

WEBTABLE

To access the elements in the webtable we need to go for it.

Every table must be in the pattern of

```
<table id="customers">
  <tbody>
    <tr>
      <th>Company</th>
      <th>Contact</th>
      <th>Country</th>
    </tr>
    <tr>
      <td>Alfreds Futterkiste</td>
      <td>Maria Anders</td>
      <td>Germany</td>
    </tr>
    <tr>…</tr>
    <tr>…</tr>
    <tr>…</tr>
    <tr>…</tr>
    <tr>…</tr>
  </tbody>
</table>
```

table - tag name

tr - table row

th - table heading

td - table data

To find the web table

```
// Find the table
WebElement table = driver.findElement(By.xpath("//table[@id='customers']"));

// (or)

WebElement table = driver.findElement(By.xpath("(//table)[4]"));

// (or)


List<WebElement> tables = driver.findElements(By.xpath("//table"));
WebElement table = tables.get(3);

// (or)

List<WebElement> tables = driver.findElements(By.tagName("table"));
WebElement table = tables.get(3);
```

```
        //iterating each rows
        for(int i=0;i<tr.size();i++) {
            WebElement row = tr.get(i);

            //to get heading in that table
            List<WebElement> th = row.findElements(By.tagName("th"));
            for(int j=0;j<th.size();j++) {
                WebElement head = th.get(j);
                String text = head.getText();
                System.out.println(text);
            }

            //to get data in that table
            List<WebElement> td = row.findElements(By.tagName("td"));
            for(int j=0;j<td.size();j++) {
                WebElement data = td.get(j);
                String text1 = data.getText();
                System.out.println(text1);
            }
        }
        driver.quit();
    }
}
```

To print the first row of the table

```
public class hai{
    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclips
        WebDriver driver=new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.w3schools.com/html/html_tables.asp");
        //to get the table id useing xpath
        WebElement main = driver.findElement(By.xpath("//table[@id='customers']"));
        //to get the rows elements using tagname
        List<WebElement> tr = main.findElements(By.tagName("tr"));
            //to get 4th row of the table
            WebElement row = tr.get(3);
            //to get data in that table
            List<WebElement> td = row.findElements(By.tagName("td"));
            for(int j=0;j<td.size();j++) {
                WebElement data = td.get(j);
                String text1 = data.getText();
                System.out.println(text1);
            }
        driver.quit();
    }
}
```

DROPDOWN

Whenever dropdown takes place we need to go for Select class.

Select s=new Select(WebElement)

Dropdown Syntax

```
Drop Down syntax:
---------------------

<select>
<option value="IND">India</option>
<option value="US">United States</option>
</select>

Select:C
--------
selectByIndex(10)
selectByValue("IND")
selectByVisibleText("India")
```

Methods:

```
Select:class
-----------
1.selectByValue()-m
2.selectByVisibleText()-m
3.selectByIndex()-m
4.getOptions()-m
5.getAllSelectedOptions()-m
6.getFirstSelectedOption()-m
7.isMultiple()-m
8.deSelectByIndex()-m
9.deSelectByValue()-m
10.deSelectByVisibleText()-m
11.deSelectAll()-m
```

Example:

```
import org.openqa.selenium.support.ui.Select;

public class hai {

    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R\\eclipse-workspace\\S
        WebDriver driver=new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.facebook.com/");
        WebElement btnCreate = driver.findElement(By.xpath("//a[text()='Create New Account']"));
        btnCreate.click();
        WebElement year = driver.findElement(By.id("year"));
        //Select class to select the webelement
        Select s=new Select(year);
        //to select the required webelement using index
            s.selectByIndex(2);
        //to select the required webelement using index
            s.selectByValue("1996");
        //to select the required webelement using index
            s.selectByVisibleText("1998");
        //to select the required webelement using index
            s.deselectByIndex(2);
        //to select the required webelement using index
            s.deselectByValue("1996");
        //to select the required webelement using index
            s.deselectByVisibleText("1998");
        driver.quit();
    }
}
```

CSS VALUE

```
CSS(Cascading Style sheets) VALUE:
==================================
<p style="color:green;font-size:24px;">Greens Technologies</p>

-->"style" attribute is way of declaring CSS properties of any HTML element.
-->Each attribute has a name and a value, separated by a colon (:).
-->Each property declaration is  separated by a semi-colon (;).

Disadvantage of getAttribute("attributename")
---------------------------------------------------------
We cannot get the value of each attribute separately using getAttribute()

WebElement f1 = driver.findElement(By.xpath("//p[text()='Greens Technologies']"));
String text=f1.getAttribute("style");
System.out.println(text);

OutPut:
----------
color:green;font-size:24px
```

```
Instead of getAttribute(),we go for getCssValue("name")
-------------------------------------------------------
System.out.println(f1.getCssValue("font-weight"));
System.out.println(f1.getCssValue("color"));
System.out.println(f1.getCssValue("font-size"));
System.out.println(f1.getCssValue("background-color"));
System.out.println(f1.getCssValue("text-align"));

output:
------
400
rgba(28, 30, 33, 1)
12px
rgba(255, 255, 255, 1)
center
```

If the tag name changes dynamically we can use *.

Eg:

//button[@name='login'] →here the button name changes dynamically

//*[@name='login'] → use *

Highlighting text

```
package org.com;

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

public class Test_01 {
    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver","C:\\Users\\Azar. A. R\\eclipse-workspace\\Zpractice\\
        WebDriver driver= new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.facebook.com/");
        WebElement txtHighlight = driver.findElement(By.xpath("//h2[contains(text(),'Facebook helps')]"));
        JavascriptExecutor js=(JavascriptExecutor) driver;

        //Highlighting the text in webpage
        js.executeScript("arguments[0].setAttribute('style','background:yellow')",txtHighlight);
    }

}
```

```
package org.com;

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

public class Task_02 {

    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar.
        WebDriver driver =new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.facebook.com/");
        WebElement btnLogin = driver.findElement(By.name("login"));
        String color = btnLogin.getCssValue("background-color");
        String fontSize = btnLogin.getCssValue("font-size");
        String width = btnLogin.getCssValue("width");
        String fam = btnLogin.getCssValue("font-family");
        System.out.println(color);
        System.out.println(fontSize);
        System.out.println(width);
        System.out.println(fam);

    }

}
```

NAVIGATION COMMANDS

driver.navigate().to("url")	To navigate to given url
driver.navigate().forward()	To move to the next page
driver.navigate().backward()	To move to the current page
driver.navigate().refresh()	To refresh the particular page

get() Will wait till the webpage loaded completely and it will not maintain cookies(history)

navigate().to() Will not wait till the page load completely. But it will maintain browser history so that we can move to the previous page and next page.

```
package org.com;

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

public class Task_02 {

    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A.
        WebDriver driver =new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.facebook.com/");

        //navigate to next url
        driver.navigate().to("https://www.redbus.in/");
        //to go to previous page (moves to facebook again)
        driver.navigate().back();
        //to go to next page (moves to redbus again)
        driver.navigate().forward();
        //to refresh the page (refreshes the page)
        driver.navigate().refresh();
    }
}
```

WAITS (synchronisation)

Types:

1. unconditional wait
2. conditional wait

1. Unconditional Wait

Thread.sleep() → For the given time script will pause its execution. Even the webelement is found earlier the program will not resume until the given time completes.

Disadvantages

1. Application will get slow
2. Performance will be reduced.

2. Conditional Wait

For a given condition we can make our script to wait is known as conditional wait.

Types

1. Implicit Wait
2. Explicit Wait

2.1 Implicit Wait

Whenever we need to find webelement in webpage, if the webelement is not present, before throwing the exception it will wait for the given time. When the webelement appeared the program will resume and it wont wait for the time to complete

If it could not find the webelement it will throw TimeOutException.

It is applicable for all the locators.

Default polling time is 250 ms(milli seconds)[every 250ms it will go and check the webelement found or not)

```
import java.util.concurrent.TimeUnit;

public class Task_02 {

    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Azar. A. R
        WebDriver driver =new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.facebook.com/");

        //implicit wait
        driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);

        WebElement txtmail = driver.findElement(By.id("email"));
        txtmail.sendKeys("azar");
    }
}
```

2.2 Explicit Wait

It is applicable for particular locator/ condition.

For the given condition to be satisfied or for finding the webelement till that we can make our program to wait.

Types

1. WebDriverWait
2. FluentWait

- ***Wait*** (Wait is the interface)
- ***FluentWait implements Wait***
- ***WebDriverWait extends FluentWait***

All the methods in fluent wait will also be in WebDriverWait

2.2.1 WebDriverWait

Whenever the time interval we give it will take only in SECONDS (so we cannot overload).

It cannot handle TimeOutException.

Default polling time is 500 ms

WebDriverWait w = new WebDriverWait(driver, 10);

```
import org.openqa.selenium.Alert;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;

public class Test02 {

    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver","C:\\Users\\Azar. A. R\\ecl
        WebDriver driver= new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.facebook.com/");
        //instantiate WebDriverWait
        WebDriverWait w = new WebDriverWait(driver, 10);
        w.until(ExpectedConditions.alertIsPresent());

        Alert alert = driver.switchTo().alert();
        alert.accept();

        WebDriverWait w1 = new WebDriverWait(driver, 20);
        w1.until(ExpectedConditions.elementToBeClickable(By.name("login")));

        WebElement login = driver.findElement(By.name("login"));
        login.click();
    }
}
```

2.2.2 FluentWait

We can give the time interval in terms of MILLISECONDS, MACROSECONDS etc.

It can handle TimeOutException.(because here we have additional method)

We can set the polling time.

```
import org.openqa.selenium.Alert;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.FluentWait;

public class Test02 {

    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver","C:\\Users\\Azar. A. R\\eclipse-workspace\\
        WebDriver driver= new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.facebook.com/");
        //instantiate FluentWait
        FluentWait<WebDriver> w = new FluentWait<>(driver).withTimeout(Duration.ofSeconds(20)).
                            pollingEvery(Duration.ofSeconds(1)).ignoring(Throwable.class);
        w.until(ExpectedConditions.alertIsPresent());

        Alert alert = driver.switchTo().alert();
        alert.accept();

        w.until(ExpectedConditions.elementToBeClickable(By.name("login")));

        WebElement login = driver.findElement(By.name("login"));
        login.click();
    }
}
```

Methods

WebDriver d=new ChromeDriver();		
WebDriver = d		
d.get("url")		To get into / launch webpage
d.getTitle()	String	To print the title of the webpage
d.getCurrentUrl	String	To print the current page url
d.manage().window().maximize()		To maximize the window screen
d.findElement()	WebElement	Find the webelement in the webpage
d.findElements()	List<WebElement>	Find all the xpath with similar xpath name
d.quit()		To quit the browser
ALERT		
d.switchTo().alert()	Alert	To switch into alert when alert takes place
FRAMES		
d.switchTo().frame()		to switch into frame
d.switchTo().parantFrame()		to switch from child frame to parent frame
d.switchTo().defaultContent()		to switch from frame to main content

WebElement e=driver.findElement(By.id("email"))		
WebElement=e		
By.locator()		to get the locator
e.sendKeys		to pass the values in the text box
e.click		to click button in web page
e.getText()	String	print existing text in the webpage
e.getAttribute("AttributeName")	String	to print the atribute value
e.getAttribute("value")	String	to print the user enterd value in the webpage
e.findElement()		Find the webelement in the webpage
e.findElements()		Find all the xpath with similar xpath name

Locator	
By.Locator	
By.id()	id value
By.name()	name value
By.className()	class value
By.Xpath()	**attributes**

	//tagName[@attributeName='attributeValue']
	text
	//tagName[text()='textName']
	partial text
	//tagName[contains(text(),'partial text']
	partial attribute
	//tagName[contains(@atttributeName,'attributeValue')]

VISIBLITY OF WEBELEMENT		
e.isSelected()	Boolean	to check whether the webelement is selected or not
e.isEnabled()	Boolean	to check whether the webelement is enabled or not
e.isDisplayed()	Boolean	to check whether the webelement is displayed or not
CSS VALES		
e.getCssValue("details tag");		

Actions a = new Actions(d)		
a.moveToElement(e)		To move curser/ mouse point
a.dragAndDrop(source e, target e)		For drag and drop option
a.doubleClick(e)		double click option
a.contextClick(e)		right click option
a.sendKeys(source e, target words)		pass values in text field
perform()		mentioned at last of actions method

Robot r =new Robot()		
r.keyPress()		Key press operation
r.keyRelease()		Key release operation
	KeyEvent.VK_DOWN	eg for downkey operation inside robot method
	KeyEvent.VK_UP	eg for upkey operation inside robot method
	KeyEvent.VK_SHIFT	eg for shiftkey operation inside robot method

Alert ar=d.switchTo().alert();		
ar.accept()		accept the alert
ar.dismiss()		to dismiss the alert
ar.sendKeys()		to insert the value in alert
ar.getText()	String	to print text in the alert

JavaScriptExecutor js = (JavaScriptExecutor)d		
js.executeScript (“JavaSript code” ,WebElement reference);		to excecute JavaScript
JavaScript Codes		
arguments[0].setAttribute('value','input txt')		to pass any input text in the text box
return arguments[0].getAttribute('value')	String	to retrieve the user enterd values
arguments[0].click()		to click any button
arguments[0].scrollIntoView(false)		for scroll down operation
arguments[0].scrollIntoView(true)		for scroll up operation
arguments[0].setAttribute('style','background:color')		to heighlight the webelement in the wepage

TakesScreenshot ts=(TakesScreenshot) d;		
ts.getScreenshotAs(OutputType.FILE)	File	to take screen short and return file directiry stored
FileUtils.copyFiles(source,destination)		to copy from the default storage and paste it in the destination

d.switchTo().frame()	
d.switchTo().frame(String id)	To switch into frame
d.switchTo().frame(String name)	To switch into frame
d.switchTo().frame(webelement)	To switch into frame
d.switchTo().parantFrame(String id)	to switch out of child frame
d.switchTo().defaultContent()	to switch to main page of DOM

d.switchTo().window()		
d.switchTo().window(string url)		to switch into other window using url
d.switchTo().window(string title)		to switch into other window using page title
d.switchTo().window(string id)		to switch into other window using window id

to find id		
d.getWindowHandle()	String	to get parent id
d.getWindowHandles()	Set<String>	to get all the windows id
d.switchTo().defaultContent()		to switch to main parent window

Select s=new Select(WebElement)	
s.selectByValue()	
s.selectByVisibleText()	
s.selectByIndex()	
s.getOptions();	List<WebElement>
s.getAllSelectedOptions();	List<WebElement>
s.getFirstSelectedOptions();	WebElement
s.isMultiple()	Boolean
s.deSelectByValue()	
s.deSelectByVisibleText()	
s.deSelectByIndex()	
s.deSelectAll()	

.

7Navigation	
d.navigate().to("url")	to navigate to given url
d.navigate().forward()	to move to the next page
d.navigate().backward()	to move to previous page
d.navigate().refresh()	to refresh the current page

Waits
Unconditional wait
Thread.sleep(milliseconds)
Conditional Wait
Implicit wait

d.manage().timeouts().implicitlyWait(10,TimeUnit.SECONDS)
d.manage().timeouts().implicitlyWait(10,TimeUnit.MINUTES)
d.manage().timeouts().implicitlyWait(10,TimeUnit.MILLISECONDS)
d.manage().timeouts().implicitlyWait(10,TimeUnit.MICROSECONDS)
d.manage().timeouts().implicitlyWait(10,TimeUnit.HOURS)
d.manage().timeouts().implicitlyWait(10,TimeUnit.DAYS)
Explicit Wait
WebDriverWait
WebDriverWait w = new WebDriverWait(d, 10);
w.until(ExpectedConditions.elementsToBeClickable(By.Name("login");
FluentWait<WebDriver> w = new FluentWait<>(driver).withTimeout(Duration.ofSeconds(20)). pollingEvery(Duration.ofSeconds(1)).ignoring(Throwable.class);
w.until(ExpectedConditions.alertIsPresent());