Working with Windows

- When we click on some link or button sometimes it opens a new window.
- Now we have two html documents opened in two different windows. But webdriver will be finding the elements in the first window opened using new chromedriver etc.
- To make webdriver to findelements in other windows opened by current window operations we have to switchto window.
- To switch to window we have to use handler id. The handler id will be there for every window.
- It is like session id and is unique. It will change everytime a window reopens.
- To get current window handler id use driver.getWindowHandle();
- To get other window handler id we dont have any specific method to get that by title. We can get all window handles opened by webdriver in current session using driver.getWindowHandles();
- This statement returns set of handler strings. Using forloop switch to one by one window and compare the window title. Whenever title matches stop switching the window and do operations on that window.
- In case if the title is dynamic then we can find the window by using any child element of that window.
- To switch back to window make sure you have the main window handler using getwindowHandle method. If we store this in a variable before switching the window it becomes easy for switching to main window in future.
- To close the current window use driver.close();
- To close all windows that are opened in current session use driver.quit();

```java
public class Wd23WorkingWithWindows {

    public static void main(String[] args) throws InterruptedException {
            // open new chrome window
            WebDriver driver = DriverFactory.getDriver("chrome");

            // specify page load
            driver.manage().timeouts().pageLoadTimeout(10, TimeUnit.SECONDS);

            // specify element wait
            driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

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

            // goto orange hrm login page
            driver.get("https://www.naukri.com/");

            // click on Login
            driver.findElement(By.xpath("//div[text()='Login']")).click();

            // click on facebook button
            driver.findElement(By.xpath("//span[text()='Facebook']")).click();

            String mWndHandle = driver.getWindowHandle();
            /*
```

```
		 * Set<String> strWndHandles = driver.getWindowHandles(); for(String
		 * wndHandle:strWndHandles) { driver.switchTo().window(wndHandle);
		 * if(driver.getTitle().equalsIgnoreCase("Log in to Facebook |
Facebook")){
		 * break; } }
		 */

		switchToWindowByTitle(driver, "Log in to Facebook | Facebook");

		// switchToWindowByChild(driver, By.id("email"));
		driver.findElement(By.id("email")).sendKeys("sudhakar@qtpsudhakar.com");

		driver.close(); // closes only current window

		driver.switchTo().window(mWndHandle);

		driver.quit(); // closes all windows opened in current session
	}

	public static void switchToWindowByTitle(WebDriver driver, String wndTitle) {
		Set<String> strWndHandles = driver.getWindowHandles();
		boolean wndFound = false;
		for (String wndHandle : strWndHandles) {
			driver.switchTo().window(wndHandle);
			if (driver.getTitle().equalsIgnoreCase(wndTitle)) {
				wndFound = true;
				break;
			}
		}

		if (wndFound) {
			System.out.println("switched to " + wndTitle + " Window");
		} else {
			System.out.println("Window Not Found");
		}
	}

	public static void switchToWindowByChild(WebDriver driver, By cElmlocator)
throws InterruptedException {
		Set<String> strWndHandles = driver.getWindowHandles();
		boolean wndFound = false;
		for (String wndHandle : strWndHandles) {
			driver.switchTo().window(wndHandle);
			Thread.sleep(2000);
			if (driver.findElements(cElmlocator).size() > 0) {
				wndFound = true;
				break;
			}
		}

		if (wndFound) {
			System.out.println("switched to Window using child");
		} else {
			System.out.println("Window Not Found");
		}
```

```
        }
    }
```

Working With Elements in Frame

- If developers wants to reuse html documents then they will use the concept of frames
- Using iframe tag they can reuse html documents
- <iframe src="url" />
- If they use frames in html, it becomes a html document in side another html document
- **What is problem with webdriver in working with frames?**
- Web driver will not directly find the elements which are there in frames by default.
- We must set the findelement scope to frame by using driver.switchTo().Frame
- We can switch by using frame id/frameElement/frameindex
- **How do we know that the element is in frame?**
- Right click on the element in chrome or firefox -> it will display frame related options like view frame source or this frame
- For other browsers we need to search in source code or see the hierarchy displayed by browser
- Most of the time in the first instance we get a failure and then we will check that the element is there in frame or not
- **What if we have frame in side of a frame?**
- We can use driver.switchTo().frame to switch from frame to frame
- To switch to parentframe there is driver.switchTo().parentFrame()
- **How do we switch back to main html document?**
- driver.switchTo().defaultContent

Note: Frames are also created by using frame and frameset tags in old web applications. But frame and frameset tags are not supported in html 5.

If an application is using frame and frameset then first we need to use switch frame and then again switch to frame because the frameset is combination of multiple frames

WebDriver driver = DriverFactory.getDriver("chrome");

DriverFactory.openApplication(driver, "https://www.redbus.in/");

driver.findElement(By.id("signin-block")).click();

driver.findElement(By.id("signInLink")).click();

//switch to frame

WebElement frmElm = driver.findElement(By.xpath("//iframe[@class='modalIframe']"));

//switch frame

```
driver.switchTo().frame(frmElm);


driver.findElement(By.xpath("//button[text()='signup with email']")).click();
driver.findElement(By.id("emailID")).sendKeys("abc@gmail.com");
```

Handling Alerts

- An alert is generated by javascript to alert the user
- When alert displays we cannot perform any other operation in application
- We must close or click on OK button to do any other operations in application
- We can see the alerts in below link
  - https://www.w3schools.com/js/js_popup.asp
- These alerts are not generated by html. That's why we cannot use driver.findelement to handle these alerts
- Selenium is giving alert class to handle these alerts
- we need to switch to alerts from html document
- driver.switchTo().alert();
- This statement will switch the focus to alert and return alert type
- we can use
  - alert.accept : to click on ok
  - alert.dismis : to click on cancel
  - alert.sendkeys : to send text in case it is asking an input
  - alert.getText() : to get text from alert window
- Sometimes we see the html alerts. They behave like javascript alerts but we can handle them using driver.findelement
- Javascript alerts are not colorful
- we cannot inspect javascript alerts
- we can inspect html alerts and these are colorful
- Alerts and notifications are different
- alert is generated by javascript when we click on some elements
- notification is like permission kind of activity from browser
- There is a different technique to handle notifications

Handling Notifications

- Handling notifications is not part of html
- These are browser level but not html document level
- We need to use some browser related classes to configure the browser while creating browser instance
- chrome is giving chromeoptions class
- firefox is giving firefox profiles
- we have to use those classes and input them to browser creation instance code

- chromeoptions co = new chromeoptions();
- co.add "required settings"
- webdriver driver = new chromedriver(co);
- //find chrome options in below link
- https://sites.google.com/a/chromium.org/chromedriver/capabilities
- https://peter.sh/experiments/chromium-command-line-switches/
- options.addArguments("--disable-notifications"); used for disablng notifications
- For firefox we can use firefox profiles
- we can create a profile for firefox and configure it manually with some settings
- We can use that profile for automation
- We can also create a new profile and setup with some settings using program
- FirefoxProfile p = new FirefoxProfile();
- p.setPreference("dom.webnotifications.enabled", false);
- We can use above class and pass that profile instance to firefox driver class
- whatever the setting we have added to chromeoptions or firefox profiles those settings will be applied while creating new chrome or firefox browser
- Note:
  - Some websites are displaying notifications as part of html document
  - we can handle them using findelement technique
  - ex: redbus

Handling File Inputs

- FileInput is a windows dialog and cannot be handled using webdriver
- If the file input is created using <input type="file"> then we sendkeys on that element instead of browsing the file. It gets updated on send keys.
- If not file type then we take the help of other tool which can automate technologies other than web
- There is a tool called SIKULI which can automate applications by finding elements using images not by attributes.
- We can use it for automating controls other than web
- We can also use another technique by automating mouse and keyboard. We can do this using JAVA Robot class.
- Using Robot class we can control mouse and keyboard.
- Using SIKULI we can automate anything which we can see on computer screen

Handling Using Robot Class

- Robot class is part of Java Abstract window toolkit
- Developers use these awt libraries to create java applications
- We can use this robot class for controlling key board/ mouse for automation purpose
- we don't need to add any extra libraries because this class is available in java sdk
- To create instance robot class
- Robot r = new Robot();
- It has methods to control keyboard or mouse

- We can use keyPress and KeyRelease methods to press and release keyboard keys. But the problem is we have to do for each key. If we have string it becomes difficult to press and release each key and we may miss a key while writing many statements for typing many keys.
- That's why we will store the text to type in clipboard and send ctrl+v keys to application to paste the entire string
- To get clipboard
- Clipboard cb = Toolkit.getDefaultToolkit().getSystemClipboard();
- to store contents
- cb.setContents(new StringSelection(flPath), null);
- below program will paste the file path and press enter key

```
//create instance for robot

Robot robo = new Robot();

//send file path text to clipboard

robo.keyPress(KeyEvent.VK_CONTROL);

robo.keyPress(KeyEvent.VK_V);

robo.keyRelease(KeyEvent.VK_V);

robo.keyRelease(KeyEvent.VK_CONTROL);


//send enter key to click on open button

robo.keyPress(KeyEvent.VK_ENTER);

robo.keyRelease(KeyEvent.VK_ENTER);
```

Note: When performing keyboard/mouse operations by using hardware we must specify static wait before and after performing operations

Handling Using SIKULI

- **configure sikuli:**
- Goto http://sikulix.com/
- click on Get version 1.1.1
- download sikulixsetup-1.1.1.jar file
- keep jar file in a safe and separate folder
- execute this jar file
- click on yes
- select first two options
- click on yes button until you see a message "have fun"
- after completion of setup there will be two jars created by setup jar file
- sikulix.jar : this is a runnable jar file which is sikuli IDE
- we can create sikuli tests within sikuli ide using jython language

- sikulixapi.jar : we have to use this for our eclipse project to automate applications using java by providing screen captures of elements
- Add this jar file to build path
- Start developing program using Screen class

import Screen class from org.sikuli.script.Screen;

```
//create instance for sikuli screen class

Screen sc = new Screen();

sc.type("element image path", "text to enter");

Thread.sleep(1000);

sc.click("element image path");

Thread.sleep(1000);
```

- You can capture elements using snipping tool
- while capturing the element images include label/supporting text also
- Keep some static wait before and after performing sikuli operations
- This will increase reliability