

#### 6: Browser Native Popup:

- >The Popup which is given by browser not by application
- >We cannot drag this popup
- >We cannot inspect this popup

Script:

```
//Handling notifications popup in chrome browser
    //user setting
    ChromeOptions opt = new ChromeOptions();
    opt.addArguments("start-maximized");
    opt.addArguments("disable-notifications");

    //open browser with user setting
    WebDriver driver = new ChromeDriver(opt);
    driver.get("https://www.irctc.co.in/ndt/train-search");
```

---

Script: //Handling notification popup in firefox Browser

```
    //user setting
    FirefoxOptions opt = new FirefoxOptions();
    opt.addPreference("dom.webnotifications.enabled", false);
    //Open browser with user setting
    WebDriver driver = new FirefoxDriver(opt);
    driver.get("https://www.yatra.com");
```

---

#### 7: Frames Popup:

- >One webpage element will be displayed as a part of another webpage
- >To create frame we use iframe tag

Examp: webpage\_1

```
<html>
<body>
    pwd<input type="password" id="pass"><br>
    Email<input type="text" id="email">
</body>
</html>
```

```
<html>
<body>
    name<input type="text" id="user"><br>
    <iframe src="webpage_1.html" id="q"></iframe><br>
    contact<input type="text" id="number">
</body>
</html>
```

-->to handle iframe element we need to switch selenium focus from main page to frame by using foll cmd:

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

- 1: int index
- 2: String(id/name)
- 3: WE arg.

-->After handling iframe element we need to switch back from frame to main page using foll. cmd:  
driver.switchTo().parentFrame();  
driver.switchTo().defaultContent();

Ques: Diff. between defaultContent() and parentframe()

- 1: ParentFrame: This method is used to switch from child frame to parent frame
- 2: defaultContent: This method is used to switch from any child to main page

Script:

```
WebDriver driver = new FirefoxDriver();
    driver.get("file:///C:/Users/Alpha/Desktop/HTML/E32/webpage_2.html");
    //enter name
    driver.findElement(By.id("user")).sendKeys("abc");
    //switch to frame
    //driver.switchTo().frame(0);
    //driver.switchTo().frame("q");
    WebElement frame = driver.findElement(By.xpath("//iframe"));
    driver.switchTo().frame(frame);
    //enter pasxword
    driver.findElement(By.id("pass")).sendKeys("mno");
    driver.findElement(By.id("email")).sendKeys("abc@gmail.com");
    //switch to main page
    //driver.switchTo().parentFrame();
    driver.switchTo().defaultContent();
    driver.findElement(By.id("number")).sendKeys("12345");
```

---

Script:

```
WebDriver driver = new FirefoxDriver();
    driver.manage().window().maximize();
    driver.get("https://demo.automationtesting.in/Frames.html");
    //switch using int index
    //driver.switchTo().frame(0);

    //driver.switchTo().frame("singleframe");

    WebElement frame = driver.findElement(By.xpath("//iframe[@id='singleframe']"));
    driver.switchTo().frame(frame);
    driver.findElement(By.xpath("//input[@type='text']")).sendKeys("hello");
```

---

Script:

```
WebDriver driver = new FirefoxDriver();
driver.get("https://jqueryui.com/droppable/");
driver.switchTo().frame(0);
Actions a = new Actions(driver);
WebElement drag_Ele = driver.findElement(By.xpath("//div[@id='draggable']"));
WebElement drop_Ele = driver.findElement(By.xpath("//div[@id='droppable']"));
a.dragAndDrop(drag_Ele, drop_Ele).perform();
driver.close();
```

---

\*\*\*Nested Frame\*\*\*

1: Enter Username:

```
driver.switchTo().Frame(0);
driver.switchTo().Frame(0);
driver.findElement().sendKeys();
```

2: Enter Password:

```
driver.switchTo().ParentFrame();
driver.switchTo().Frame(1);
driver.findElement().sendKeys();
```

3: Enter Email:

```
driver.switchTo().defaultContent();
driver.switchTo().Frame(1);
driver.switchTo().Frame(0);
driver.findElement().sendKeys();
```

4: Enter Contact:

```
driver.switchTo().defaultContent();
driver.switchTo().Frame(2);
driver.findElement().sendKeys();
```

---

Practice Ques: [https://the-internet.herokuapp.com/nested\\_frames](https://the-internet.herokuapp.com/nested_frames)(Display LEFT & MIDDLE text as an output)

---

\*\*\*\*Handling Synchronization Issues\*\*\*

- >While navigating between pages application may take time to load webpage
- >findElement method will not wait until page loaded
- >If element is not visible, displays NoSuchElementException
- >To avoid this exception we need to match teste script execution speed & Application running speed.

Definition: Matching of application running speed with script execution speed.

-->to achieve synchronization following wait commands:

1: Thread.sleep()---->dead wait/BlindWait

2: implicit wait

3: explicit wait

4: Fluent wait

1: Thread class- By using static method sleep(), we can stop the execution of script for a period of time

Syntax: Thread.sleep(milliseconds);

\*\*Limitation:

1: If application is loaded early script execution will not start until mentioned time is completed.

2: If application takes more time than mentioned, script execution will starts & returns

NoSuchElementException

3: It will increase the length of Test Script.

---

2: implicitWait():

-->It will control all findElement and findElements method in test Script  
-->findElement will search element is present in webpage or not  
-->If it is present perform action otherwise wait for 0.5 sec and search for the element. This process will continue until timeout.  
-->If element is not visible within given time. NoSuchElementException  
\*\*Searching for element for every 0.5 sec is known as "Polling time"

Syntax: driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(Time in sec));  
3rd line of code

Note: It is also called as global wait because if we declare it once it will take care of entire script

Script:

```
WebDriver driver = new FirefoxDriver();
    driver.get("https://www.facebook.com/");
    driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(15));
    driver.findElement(By.linkText("Forgotten password?")).click();
    driver.findElement(By.id("identify_email")).sendKeys("abc");
    driver.findElement(By.id("did_submit")).click();
```

---

3: explicitlyWait:

-->It works based on condition, if condition satisfied execute next command  
-->If condition not satisfied wait for 1/2 sec again verify the condition satisfied or not  
-->This process will continue till timeout and once timeout happened it throw TimeOutException & terminate script.

Syntax: WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(time))
 wait.until(ExpectedConditions.condition(Locator));

-->create Object of WebDriverWait[which is a class in selenium]  
using this reference variable call the method until  
until takes ExpectedConditions as an argument(class in selenium)

1: titleContains()  
2: UrlContains()  
3: alertIsPresent  
4: elementToBeSelected()  
5: elementToBeClickable

Examp:

```
WebDriver driver = new FirefoxDriver();
    driver.get("http://localhost/login.do");
    //verify target page displayed or not
    WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));
    wait.until(ExpectedConditions.titleContains("Login"));
    System.out.println("Login Page Displayed");
```

---

4: Fluent wait:

-->Similar to explicit wait  
-->It is used to customize the polling time

```
Syntax: FluentWait wait = new FluentWait(driver);
        .pollingEvery(Duration.ofSeconds(1));
        .withTimeout(Duration.ofSeconds(15));
        .until(ExpectedConditions.condition(WebElement));
```

Examp:

```
WebDriver driver = new FirefoxDriver();
    driver.manage().window().maximize();
    driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(15));
    driver.get("https://www.shoppersstack.com/products_page/25");
    driver.findElement(By.id("Check Delivery")).sendKeys("123456");
    WebElement check = driver.findElement(By.id("Check"));
    //fluent Wait
    FluentWait wait = new FluentWait(driver);
    wait.pollingEvery(Duration.ofSeconds(1));
    wait.withTimeout(Duration.ofSeconds(15));
    wait.until(ExpectedConditions.elementToBeClickable(check));
    check.click();
```

---