**Handling Mouse Movements**

As part of automation we might need to use mouse related operations

- Click
- Double Click
- Right Click
- Scroll
- Hover
- Drag And Drop

To handle these operations we have Actions class in selenium. Below are the equivalent methods for each operation in action class

- click : click()
- Double Click : doubleClick()
- Right Click: contextclick()
- Hover/scroll: moveToElement()
- Drag and Drop : dragAndDrop()s

Every operation can be performed on element or on location. We also have a method called sendkeys to send text on any control. Remember that Actions class can handle operations only inside html document. We have to build operations and perform all operation at once when using action class. Whenever we perform keyboard or mouse related operations keep some static wait before and after performing operations.

What is the difference between Actions class vs Robot class?

- Actions is given by selenium. Robot class is given java
- Using Actions class we can perform operations only in HTML document. Using Robot class we can perform operations in any location of the screen.
- We can perform operations on elements or on location using actions class. We can perform operations only on location using Robot class
- We can use combination of actions and robot to perform any operation in case if it is needed.

**Mouse Hover**

```
// goto naukri page
driver.get("https://www.naukri.com/");

WebElement elm = driver.findElement(By.xpath("//div[text()='Recruiters']"));

//Create action instance
Actions act = new Actions(driver);

//mouse hover on element
act.moveToElement(elm).build().perform();
```

**Mouse Scroll**

```
// goto redbus page
driver.get("https://www.redbus.in/");

//Create action instance
Actions act = new Actions(driver);

//mouse hover on element
act.moveToElement(driver.findElement(By.id("awards_div"))).build().perform();
```

**Double Click**

```
// goto jquery api page
driver.get("https://api.jquery.com/dblclick/");

// switch to frame
driver.switchTo().frame(driver.findElement(By.xpath("//*[@id='example-
1']//iframe")));

// Create action instance
Actions act = new Actions(driver);

// mouse hover on element
act.doubleClick(driver.findElement(By.xpath("//div"))).build().perform();
```

**Right Click on Element**

```
// goto jquery api page
driver.get("https://swisnl.github.io/jQuery-contextMenu/demo.html");

// Create action instance
Actions act = new Actions(driver);

// mouse hover on element
act.contextClick(driver.findElement(By.xpath("//span[text()='right click
me']"))).build().perform();

//click on option in context menu
act.click(driver.findElement(By.xpath("//span[text()='Edit']"))).build().perf
orm();
```

**Drag and Drop on Element**

```
// goto jquery api page
driver.get("https://jqueryui.com/droppable/");

//switch to frame
driver.switchTo().frame(driver.findElement(By.className("demo-frame")));

// Create action instance
```

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

//get src elem
WebElement elmSrc = driver.findElement(By.id("draggable"));

//get dst elem
WebElement elmDst = driver.findElement(By.id("droppable"));

//drag and drop on element
act.dragAndDrop(elmSrc, elmDst).build().perform();
```

**Drag and Drop on Location**

```
// goto jquery api page
driver.get("https://jqueryui.com/droppable/");

// switch to frame
driver.switchTo().frame(driver.findElement(By.className("demo-frame")));

// Create action instance
Actions act = new Actions(driver);

// get src elem
WebElement elmSrc = driver.findElement(By.id("draggable"));

// get dst elem
WebElement elmDst = driver.findElement(By.id("droppable"));

//get x, y co-ordinates
int dX = elmDst.getLocation().getX();
int dY = elmDst.getLocation().getY();

// drag and drop on location
act.dragAndDropBy(elmSrc, dX, dY).build().perform();
```

**Click element without using Click Method**

```
// goto orange hrm login page
driver.get("http://opensource.demo.orangehrmlive.com");

// enter text on username
driver.findElement(By.id("txtUsername")).sendKeys("admin");

// enter text on password
driver.findElement(By.id("txtPassword")).sendKeys("admin");

// click on login button
driver.findElement(By.id("btnLogin")).sendKeys(Keys.ENTER);
```

**Handling Elements Using JavaScript**

If we wanted work with Html DOM then we have to use javascript. When a web page is loaded, the browser creates a Document Object Model of the page.

With the object model, JavaScript gets all the power it needs to create dynamic HTML:

- JavaScript can change all the HTML elements in the page
- JavaScript can change all the HTML attributes in the page
- JavaScript can change all the CSS styles in the page
- JavaScript can remove existing HTML elements and attributes
- JavaScript can add new HTML elements and attributes
- JavaScript can react to all existing HTML events in the page
- JavaScript can create new HTML events in the page

The HTML DOM is a standard object model and programming interface for HTML. It defines:

- The HTML elements as objects
- The properties of all HTML elements
- The methods to access all HTML elements
- The events for all HTML elements

What is the advantage we get if we are using dom?

- We can find elements using javascript dom and return as webelement
- We can find elements using webdriver and pass it to javascript to do any operations that are not provided by webdriver
- You can see list of operations or properties that can be accessed using javascript in below link
    - https://www.w3schools.com/jsref/dom_obj_all.asp

**Using JavaScript In WebDriver**

- To execute javascript through webdriver there is JavascriptExecutor interface.
- All browser classes are implementing javascript executor
- javascriptexecutor js = new chromedriver();
- But to execute on webdriver instance we have to use casting
    - webdriver driver = new chromedriver();
    - javascriptexecutor js = (javascriptexecutor) driver;
- We can use executescript method to execute javascript
- executeScript takes minimum 1 parameter that is javascript and returns object type.
- The second parameter type for executeScript is object... and so we can pass any number of parameters of any type. You have to provide javascript as string
- The values passed from second parameter onwards can be accessed in javascript as arguments array.
- Below you can see list of operation that can be performed on an element
    - https://www.w3schools.com/jsref/dom_obj_document.asp

**Common Usage of Java Script Executor in Real Time**

- Finding the elements in case webdriver techniques are not sufficient in that situation
- Click operation will not perform in some situation using webdriver methods. We can use JS in that situation
- Scrolling to element
- Highlighting the elements

```
// goto orange hrm login page
driver.get("http://opensource.demo.orangehrmlive.com");

//create instance for java script executor
JavascriptExecutor jsDriver = (JavascriptExecutor) driver;

// find element using webdriver and perform operation using java script
WebElement elmUSerName = driver.findElement(By.id("txtUsername"));

//perform operation using java script
jsDriver.executeScript("arguments[0].value='admin';", elmUSerName);

// find element using java script and perform operation using webdriver
WebElement elmPAssword = (WebElement) jsDriver.executeScript("return
document.getElementById('txtPassword');");

// enter text on password
elmPAssword.sendKeys("admin");

// Find and perform operation using javascript
jsDriver.executeScript("document.getElementById('btnLogin').click();");
```

We can find element using javascript with below methods

- document.getElementById()
  - Finding by using ID(returns single element)
- document.getElementsByClassName()
  - Finding by using class name (returns multiple elements)
- document.getElementsByName()
  - Finding by using Name (returns multiple elements)
- document.getElementsByTagName()
  - Finding elements by using tag name (returns multiple elements)
- document.querySelector()
  - Finding elements by using CSS Selector(returns first matched element)
- document.querySelectorAll()
  - Finding elements by using CSS Selector(returns all matched elements)

**Scroll to an element or location**

```
// goto redbus page
driver.get("https://www.redbus.in/");

//get element
WebElement elmAwards = driver.findElement(By.id("awards_div"));

//scroll to an element
((JavascriptExecutor) driver).executeScript("arguments[0].scrollIntoView();",
elmAwards);

//scroll to location
((JavascriptExecutor) driver).executeScript("window.scrollTo(0,500);");
```

**Highlight elements using JavaScript**

```
// goto orange hrm login page
driver.get("http://opensource.demo.orangehrmlive.com");

// create instance for java script executor
JavascriptExecutor jsDriver = (JavascriptExecutor) driver;

WebElement elmUSerName = driver.findElement(By.id("txtUsername"));

// perform operation using java script
jsDriver.executeScript("arguments[0].style.border='5px dotted red';",
elmUSerName);
```