**Handling keyboard events and mouse hover events using Webdriver**

In Webdriver, handling keyboard events and mouse events (including actions such as Drag and Drop or clicking multiple elements With Control key) are done using the advanced user interactions API . It contains Actions and Action classes which are needed when performing these events.  
In order to perform action events, we need to use org.openqa.selenium.interactions.Actions class.

Here is the sample code to work with Action Class

// Configure the Action

Actions action = new Actions(driver);

// To click on the element

action.moveToElement(element).click().perform();

**Note:** We need to use perform() to execute the action.

Using Action API, keyboard interactions are simple to do with webdriver. In Advanced User Interactions API, interaction with element is possible either by clicking on element or sending a Keys using sendKeys()  
[Check out the example](http://seleniumeasy.com/selenium-tutorials/how-to-perform-mouseover-action-in-selenium-webdriver)

To use mouse actions, we need to use current location of the element and then perform the action.

The following are the regularly used mouse and keyboard events :

**Method :clickAndHold()**  
Purpose: Clicks without releasing the current mouse location

**Method : contentClick()**  
Purpose: Performs a context-click at the current mouse location.  
[How to work with context menu by taking a simple example](http://seleniumeasy.com/selenium-tutorials/right-click-context-menu-webdriver-example)

**Method: doubleClick()**  
Purpose: Performs a double click at the current mouse location

**Method: dragAndDrop(source,target)**  
Parameters: Source and Target  
Purpose: Performs click and hold at the location of the source element and moves to the location of the target element then releases the mouse.

[Click here for Example](http://seleniumeasy.com/selenium-tutorials/drag-and-drop-using-webdriver-action-class)

**Method : dragAndDropBy(source,x-offset,y-offset)**  
Parameters: Source, xOffset - horizontal move, y-Offset - vertical move Offset  
Purpose: Performs click and hold at the location of the source element moves by a given off set, then releases the mouse.

**Method: keyDown(modifier\_key)**  
Parameters: Modifier\_key (keys.ALT or Keys.SHIFT or Keys.CONROL)  
Purpose: Performs a modifier key press, doesn't release the modifier key. Subsequent interactions may assume it's kept pressed

**Method: keyUp(modifier\_key)**  
Parameters: Modifier\_key (keys.ALT or Keys.SHIFT or Keys.CONROL)  
Purpose: Performs a key release.

**Method: moveByOffset(x-offset, y-offset)**  
Parameters: X-Offset , Horizontal offset, a negative value means moving the mouse to left side.  
Y-Offset, vertical offset, a negative value means moving the mouse to up.  
Purpose: Moves the mouse position from its current position by the given offset.

Check for the example here [Resizing a web element using movebyoffset](http://seleniumeasy.com/selenium-tutorials/resizing-a-web-element-using-webdriver-action-class-movebyoffset)

**Method: moveToElement(toElement)**  
Parameters: toElement - Element to Move to  
Purpose: It moves the Mouse to the middle of the element.

**Method: release()**  
Purpose: It releases the left mouse button at the current mouse location.

**Method: sendKeys(onElement, charSequence)**  
Parameters: onElement, which will receive the keyStrokes, usually text field.  
charsequence- any string value representing the sequence of keyStrokes to be sent.  
Purpose: It sends a series of keyStrokes onto the element