iamneo

EventHandling in Selenium Webdriver

www.iamneo.ai

Performing mouse hover action

✔ Mouse hover action is basically an action where a user places a mouse over a designated area like a hyperlink.

✔ Moving the mouse over an element on web page displays some pop-up windows or maybe description boxes.

Syntax :

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

a.moveToElement(driver.findElement(By.xpath(“”))). build().perform();
```

Example for Mouse Hover Action

- ✔ Actions actions = new Actions(driver);
- ✔ //Retrieve WebElement 'Music' to perform mouse hover
- ✔ WebElement menuOption =

  driver.findElement(By.xpath(".//div[contains(text(),'Music')]"));
- ✔ //Mouse hover menuOption 'Music'
- ✔ actions.moveToElement(menuOption).perform();
- ✔ System.out.println("Done Mouse hover on 'Music' from Menu");

Handling Context Menu

- ✔ Right Click operation is also called Context Click in Selenium.
- ✔ Right click action in Selenium web driver can be done using Actions class.
- ✔ Syntax :

```
1)Actions actions = new Actions(driver);
2)WebElement elementLocator =  driver.findElement("Locator");
3)actions.contextClick(elementLocator).perform();
```

Handling Drop down Menu

- ✔ A drop-down menu is a graphical control element that allows the user to choose one value from a list.
- ✔ 'Select' class in Selenium WebDriver is used for selecting and deselecting option in a dropdown.

Example

```
Select dropdown = new Select(driver.findElement(By.id("testingDropdown")));
dropdown.selectByVisibleText("Database Testing");
```

Ways to select an option from drop-down menu

selectByIndex - It is used to select an option based on its index, beginning with 0

for eg:

```
WebElement element = driver.findElement(By.id("year"))
Select select = new Select(element)
select.selectByIndex("Index number")
```

selectByValue - It is used to select an option based on its 'value' attribute

for eg:

```
WebElement element = driver.findElement(By.id("year"))
Select select = new Select(element)
select.selectByValue("Value")
```

Ways to select an option from drop-down menu

selectByVisibleText - It is used to select an option based on the text over the option

for eg:

```
    WebElement element = driver.findElement(By.id("year"))
    Select select = new Select(element)
    select.selectByVisibleText("Text")
```

Performing Drag and Drop action

- ✔ This action is performed using a mouse when a user moves (drags) a web element from one location and then places (drops) it at another point.
- ✔ This is a common action used in Windows Explorer when moving a file from one folder to another. Here, the user selects a file in the folder, drags it to the desired folder, and drops it.

Syntax:

```
Actions action = new Actions(driver); action.dragAndDrop(Sourcelocator,
Destinationlocator).build().perform();
```

Achieve Perform Drag and Drop action

- ✔ **clickAndHold(WebElement element)** - Clicks a web element at the middle(without releasing).
- ✔ **moveToElement(WebElement element)** - Moves the mouse pointer to the middle of the web element without clicking.
- ✔ **release(WebElement element)** - Releases the left click (which is in pressed state).
- ✔ build() - Generates a composite action

Syntax to Perform Drag and Drop action

```
WebElement fromElement = driver.findElement(By Locator of fromElement);

WebElement toElement = driver.findElement(By Locator of toElement);

Actions builder = new Actions(driver);

Action dragAndDrop = builder.clickAndHold(fromElement)
moveToElement(toElement).release(toElement).build();

dragAndDrop.perform();
```

Performing Keyboard actions

A Keyboard Event describes a user's interaction with the keyboard.

When a user presses single or multiple keys, keyboard events generate.

Type in capital/Camel case letters: Wherever user need to type a word or letter in caps, he/she will press the "SHIFT" key and will type the necessary characters,

Copy & Paste Text: When we need to copy some text from one text box to another, we select the text by pressing "CTRL+A" they copy the text using "CTRL+C" and paste the text in the new text box by simply clicking in the text box and pressing keys "CTRL+V".

Example Keyboard actions

```
// Select the Current Address using CTRL + A

        actions.keyDown(Keys.CONTROL);

        actions.sendKeys("a");

        actions.keyUp(Keys.CONTROL);

        actions.build().perform();

            (OR)

    actions.keyDown(Keys.CONTROL).sendKeys("a").keyUp(Keys.CONTROL).build().perfor
    m();
```

Example Keyboard actions

```
// Copy the Current Address using CTRL + C
    actions.keyDown(Keys.CONTROL);
    actions.sendKeys("c");
    actions.keyUp(Keys.CONTROL);
     actions.build().perform();
       (OR)
 actions.keyDown(Keys.CONTROL).sendKeys("c").keyUp(Keys.CONTROL).build().perform();
```

Example Keyboard actions

```
//Press the TAB Key to Switch Focus to Permanent Address

      actions.sendKeys(Keys.TAB);

      actions.build().perform();

            (OR)

  actions.sendKeys(Keys.TAB).build().perform();
```

Example Keyboard actions

```
//Paste the Address in the Permanent Address field using CTRL + V
actions.keyDown(Keys.CONTROL);
actions.sendKeys("v");
actions.keyUp(Keys.CONTROL);
 actions.build().perform();
   (OR)
actions.keyDown(Keys.CONTROL).sendKeys("v").keyUp(Keys.CONTROL).build().perform();
```

Alerts/popups

- *Alerts* are small popup boxes/windows which display the *messages/notifications* and notify the user with some information seeking some permission on certain kinds of operations.
- We can also use them for warning purposes.
- The user can enter a few details in the alert box as well.

(for eg: Windows alert)

Types of Alerts/popups

Simple alert: These alerts are just informational alerts and have an OKbutton on them.

Prompt Alert: the user can enter his/her username and press the OK button or Cancel the alert box without entering any details .

Confirmation Alert: these alerts get some confirmation from the user in the form of accepting or dismissing the message box.

Methods in Alerts/popups

To switch the control from the parent window to the Alert window,,

```
driver.switchTo( ).alert( );
```

To handle *Javascript alerts*, *Selenium WebDriver* provides the package *org.openqa.selenium.Alert* and exposes the following methods:

***Void accept()*:** This method clicks on the '*OK*' button of the alert box.

```
driver.switchTo( ).alert( ).accept();
```

***Void dismiss()*:** We use this method when the '*Cancel*' button clicks in the alert box.

```
driver.switchTo( ).alert( ).dismiss();
```

Methods in Alerts/popups

String getText(): This method captures the message from the alert box.

```
driver.switchTo().alert().getText();
```

Void sendKeys(String stringToSend): This method sends data to the alert box.

```
driver.switchTo().alert().sendKeys("Text");
```

Syntax of Simple Alerts/popups

```
// This step will result in an alert on screen
   driver.findElement(By.id("alertButton")).click();
   Alert simpleAlert = driver.switchTo().alert();
     simpleAlert.accept();
```

Syntax of Prompt Alerts/popups

```
WebElement element = driver.findElement(By.id("promtButton"));
  ((JavascriptExecutor) driver).executeScript("arguments[0].click()", element);
  Alert promptAlert  = driver.switchTo().alert();
  String alertText = promptAlert.getText();
  System.out.println("Alert text is " + alertText);
 //Send some text to the alert
  promptAlert.sendKeys("Test User");
    promptAlert.accept();
```

Syntax of Confirmation Alerts/popups

```
driver.findElement(By.id("timerAlertButton")).click();
        WebDriverWait wait = new WebDriverWait(driver,10);
        wait.until(ExpectedConditions.alertIsPresent());
        Alert simpleAlert = driver.switchTo().alert();
        simpleAlert.accept();
```

Iframe

iFrame is a HTML document embedded inside an HTML document. iFrame is defined by an *<iframe></iframe>* tag in HTML.

```
<html>

 <body>

  <div class="box">

   <iframe name="iframe1" id="IF1" height="600" width="400" src="https://toolsqa.com"></iframe>

  </div>

    </body>

</html>
```

Way to switch to Iframe

To Switch between iFrames we have to use the driver's *switchTo().frame* command. We can use the *switchTo().frame()* in three ways:

switchTo.frame(int frameNumber): Pass the frame index and driver will switch to that frame.

switchTo.frame(string frameNameOrId): Pass the frame element Name or ID and driver will switch to that frame.

switchTo.frame(WebElement frameElement): Pass the frame web element and driver will switch to that frame.

Syntax to switch to Iframe

```
//Switch by Index

        driver.switchTo().frame(0);

        driver.quit();

//Switch by frame name

        driver.switchTo().frame("iframe1");

        driver.quit();
```

Likewise for other options.

Handling WebTables

All you need to is to inspect the table cell and get the HTML location of it. In most cases, tables contain text data and you might simply like to extract the data given in each row or column of the table

Example:

```
String sRow = "2";

String sCol = "1";

driver.findElement(By.xpath("/html/body/div[1]/div[2]/div/div[2]/article/div/table/tbody/tr["+sRow+"]/td["+sCol+"]")).getText();
```