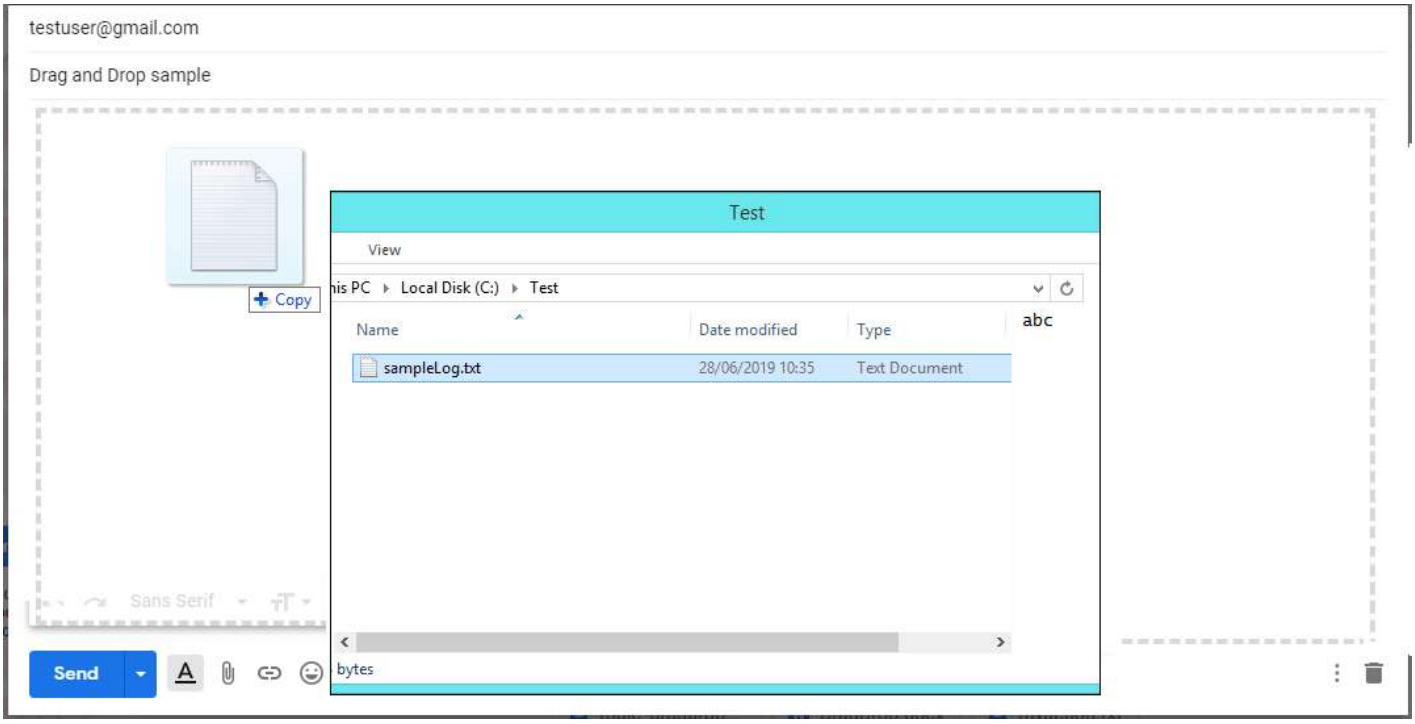
What is Drag and Drop Action?

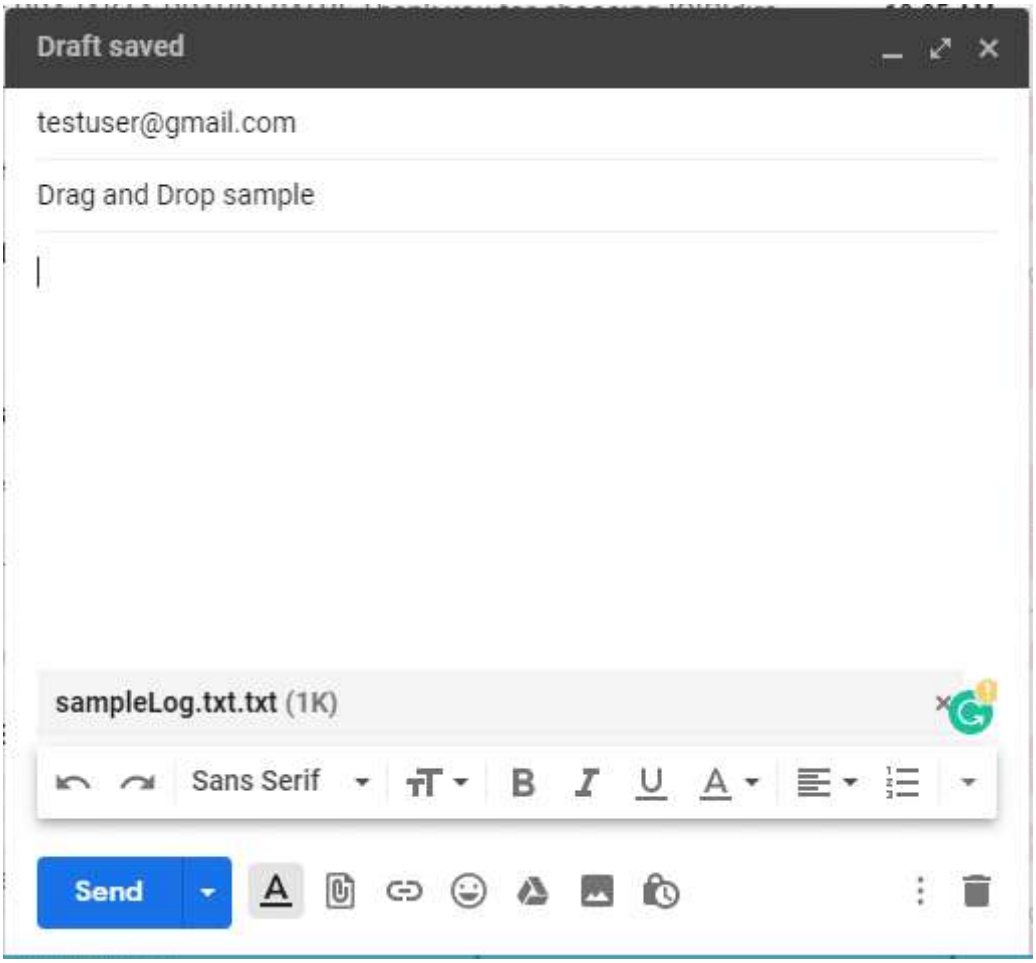
This is an action performed with a ***mouse*** when a user moves (*drags*) a web element and then places (*drops*) it into an alternate area.

E.g. This is a very common action used in Windows Explorer while moving any file from one folder to another. Here, the user selects any file in the folder, drags it to the desired folder and just drops it.

In Gmail, a file can be just dragged and dropped in new mail to send as an attachment like below:



Once the file is moved to compose an email, you can see it as an attachment. At the bottom, it is displayed that the file we dragged from windows to Gmail, is now attached as an attachment.



In automation test scripts at various instances, the same action needs to be emulated:

Select some element on the web page, drag it and then place it on the alternate area.

To perform the drag-drop action through a Selenium script, there is no direct drag-drop method available in *WebElement interface.* Unlike other commands like *click(), sendKeys()* there is nothing available for drag and drop. Here, we leverage the ***Actions class*** which provides various methods of emulating such complex interactions.

So, here are the methods Actions class provides for Drag-Drop action:

. ***dragAndDrop(WebElementsource****, WebElement target)*
. ***dragAndDropBy****(WebElementsource, int xOffset, int yOffset)*

Drag and Drop in Selenium

dragAndDrop(WebElement source, WebElement target): This method performs left click, hold the click to hold the source element, moves to the location of the target element and then releases the mouse click.

Let's see how to use Action class methods to perform drag-drop action:

First, instantiate an Actions class:

Actions actions = new Actions(driver);

As you can see, the *dragAndDrop(WebElement source, WebElement target)* method has two arguments to pass. One is a source web element and another is target web element. This source web element is any web element that needs to be dragged. Target web element is any web element on which dragged object needs to be placed or dropped. To find the source and target element use the below command:

WebElement source = driver.findElement(Any By strategy & locator);

WebElement target = driver.findElement(Any By strategy & locator);

Here, you can use any By strategy to locate the WebElement like find element by its *id, name attribute, etc.* To know more about all By strategies, please refer ***Find Elements*** tutorial.

Now, when we have got the actions class object and the element as well, just invoke ***perform()*** method for the drag & drop:

actions.dragAndDrop(source,target).perform();

Let's see what happens internally when invoke the *perform()* method above:

> ***Click And Hold Action:*** *dragAndDrop() method first performs click-and-hold at the location of the source element*
> ***Move Mouse Action:*** *Then source element gets moved to the location of the target element*
> ***Button Release Action:*** *Finally, it releases the mouse*
> ***Build:*** *build() method is used to generate a composite action containing all actions. But if you observe, we have not invoked it in our above command. The build is executed in the perform method internally*
> ***Perform:*** *perform() method performs the actions we have specified. But before that, it internally invokes build() method first. After the build, the action is performed*

Practice Exercise to Perform Drag and Drop using Actions Class in Selenium

Let's consider an example from an already available demo page on Toolsqa as ***http://demoqa.com/droppable/***

In the above image, *'Drag me to my target'* object can be selected and dragged and dropped at the *'Drop here'* object. Following message gets displayed once the object is dropped at '*Drop here*'

Now let's write a selenium script to drag-drop the object using drag and drop method.

Find below the steps of the scenario to be automated:

. *Launch the web browser and launch our practice page https://demoqa.com/droppable/*
. *Find the required source element i.e. 'Drag me to my target' object in our sample*
. *Find the required target element i.e. 'Drop Here' object in our sample*
. *Now Drag and Drop 'Drag me to my target' object to 'Drop Here' object*
. *Verify message displayed on 'Drop Here' to verify that source element is dropped at the target element*
. *Close the browser to end the program*

Selenium Code Snippet:

```
package com.toolsqa.tutorials.actions;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.interactions.Actions;

public class DragAndDrop1 {

        public static void main(String[] args) throws InterruptedException {

                 //Note: Following statement is required since Selenium 3.0,
                //optional till Selenium 2.0
                //Set system properties for geckodriver
                System.setProperty("webdriver.gecko.driver", "C:\\Selenium\\Toolsqa\\lib\

                WebDriver driver = new FirefoxDriver();

                String URL = "https://demoqa.com/droppable/";

                driver.get(URL);

                // It is always advisable to Maximize the window before performing DragND
                driver.manage().window().maximize();

                driver.manage().timeouts().implicitlyWait(10000, TimeUnit.MILLISECONDS);

                //Actions class method to drag and drop
                Actions builder = new Actions(driver);

                WebElement from = driver.findElement(By.id("draggable"));

                WebElement to = driver.findElement(By.id("droppable"));
                //Perform drag and drop
                builder.dragAndDrop(from, to).perform();

                //verify text changed in to 'Drop here' box
                String textTo = to.getText();

                if(textTo.equals("Dropped!")) {
                        System.out.println("PASS: Source is dropped to target as expected
                }else {
                        System.out.println("FAIL: Source couldn't be dropped to target as
                }

                driver.close();

        }

}
```

DragAndDropBy action in Selenium using OffSets

dragAndDropBy(WebElement source, int xOffset, int yOffset): This method clicks & holds the source element and moves by a given offset, then releases the mouse. Offsets are defined by x & y.

xOffset is horizontal movement
yOffset is a vertical movement

Let's see how to use ***Action class*** methods to perform drag-drop action using offsets.

First, instantiate an Actions class:

Actions actions = new Actions(driver);

As seen above, the dragAndDropBy() method has three arguments to pass. Source web element, *xOffset,* and *yOffset.* This WebElement source is any web element that needs to be dragged and dropped. To find the source element use the below command:

WebElement source = driver.findElement(Any By strategy & locator);

Now, the remaining two parameter values i.e. *xOffset* and *yOffset* are values in pixel.

For Example, if a xOffset value is set as 50, it means an object needs to be dragged and dropped by 50 pixels offset horizontal direction. Similarly, if a yOffset value is set as 50, it means an object needs to be dragged and dropped by 50 pixels offset vertical direction.

Here explain how to get the offsets of the element.

Now, when we have got all the objects, just invoke *perform()* method for the drag & drop:

actions.dragAndDropBy(source, xOffset, yOffset).perform();

This works exactly like *dragAndDrop(source, target)*, which we discuss first in the article. But the only difference is that in *dragAndDropBy,* it moves the source element not to the target element but to the offsets.

Practice Exercise

Let's consider an example from an already available demo page on Toolsqa as *https://demoqa.com/droppable/* . This is exactly the same sample that we have seen at the first practice exercise.

Now let's write a selenium script to drag-drop the object using *dragAndDropBy()* method.

Find below the steps of the scenario to be automated:

. *Launch the web browser and launch our practice page https://demoqa.com/droppable/*
. *Find the required source element i.e. 'Drag me to my target' object in our sample*
. *Calculate required xOffset and yOffset to drag source element in horizontal and vertical direction respectively [For this, calculate xOffset by getting the difference between x location of the source and target element. Similarly, get yOffset by getting the difference between y location of the source and target element]*
. *Now Drag and Drop 'Drag me to my target' object to 'Drop Here' object*
. *Verify message displayed on 'Drop Here' to verify that source element is dropped at the target element*
. *Close the browser to end the program*

Selenium Code Snippet:

```
package com.toolsqa.tutorials.actions;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;

import org.openqa.selenium.interactions.Actions;

public class DragAndDrop2 {

        public static void main(String[] args) throws InterruptedException {

                //Note: Following statement is required since Selenium 3.0,
                //optional till Selenium 2.0
                //Set system properties for geckodriver
                System.setProperty("webdriver.gecko.driver", "C:\\Selenium\\Toolsqa\\lib\
                WebDriver driver = new FirefoxDriver();

                String URL = "https://demoqa.com/droppable/";

                driver.get(URL);

                // It is always advisable to Maximize the window before performing DragND
                driver.manage().window().maximize();

                driver.manage().timeouts().implicitlyWait(10000, TimeUnit.MILLISECONDS);

                //Actions class method to drag and drop
                Actions builder = new Actions(driver);
```

```
                WebElement from = driver.findElement(By.id("draggable"));

                WebElement to = driver.findElement(By.id("droppable"));

                //Here, getting x and y offset to drop source object on target object loc
                //First, get x and y offset for from object
                int xOffset1 = from.getLocation().getX();

                int yOffset1 =  from.getLocation().getY();

                System.out.println("xOffset1--->"+xOffset1+" yOffset1--->"+yOffset1);

                //Secondly, get x and y offset for to object
                int xOffset = to.getLocation().getX();

                int yOffset =  to.getLocation().getY();

                System.out.println("xOffset--->"+xOffset+" yOffset--->"+yOffset);

                //Find the xOffset and yOffset difference to find x and y offset needed i
                xOffset =(xOffset-xOffset1)+10;
                yOffset=(yOffset-yOffset1)+20;


                //Perform dragAndDropBy
                builder.dragAndDropBy(from, xOffset,yOffset).perform();

                //verify text changed in to 'Drop here' box
                //Get text value of 'to' element
                String textTo = to.getText();

                if(textTo.equals("Dropped!")) {
                        System.out.println("PASS: Source is dropped at location as expect
                }else {
                        System.out.println("FAIL: Source couldn't be dropped at location
                }


                driver.close();

        }
}
```

Practice Exercise for You:

Now, you have just seen how to drag and drop by specifying *xOffset* and *yOffset,* can you try dragAndDropBy() on *http://demoqa.com/draggable/*

Hint: *You can drag and drop source object anywhere in the horizontal and vertical direction.*

If you are still finding difficulty, then only check the following code:

```
package com.toolsqa.tutorials.actions;
```

```
import java.util.concurrent.TimeUnit;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;

import org.openqa.selenium.interactions.Actions;

public class DragAndDrop3 {

        public static void main(String[] args) throws InterruptedException {

                //Note: Following statement is required since Selenium 3.0,
                //optional till Selenium 2.0
                //Set system properties for geckodriver
                System.setProperty("webdriver.gecko.driver", "C:\\Selenium\\Toolsqa\\lib\

                WebDriver driver = new FirefoxDriver();

                String URL = "https://demoqa.com/draggable/";

                driver.get(URL);

                // It is always advisable to Maximize the window before performing DragND
                driver.manage().window().maximize();

                driver.manage().timeouts().implicitlyWait(10000, TimeUnit.MILLISECONDS);

                //Actions class method to drag and drop
                Actions builder = new Actions(driver);

                WebElement from = driver.findElement(By.id("draggable"));

                //Perform dragAndDropBy
                builder.dragAndDropBy(from, 100,100).perform();

                System.out.println("Dropped");

                driver.close();

        }


}
```

Summary:

In this tutorial we have covered the concept of Drag and Drop elements in a web page and how to perform through following methods of Actions class:

dragAndDrop: Here, we saw how to use Drag and Drop Action in Selenium where we need to drag and drop the source element to target

dragAndDropBy: Here, we saw how to drag and drop the source element in a horizontal and vertical direction by the specified offset