**DAY 4 – SELENIUM BASICS ASSIGNMENT**

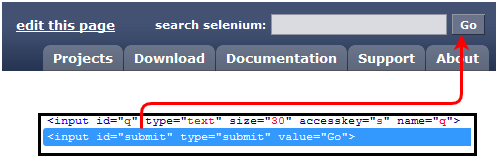
1. **Locate Your Element by ID:**

Format : *id=submit (Where “Submit” is the value of ID locator.)*

You can record the location of web element through Selenium IDE. In below example we will see how we can manually add locator with the use of Firebug.

Example :

– Hit the URL : http://docs.seleniumhq.org/ in Firefox browser.  
– Look for “search selenium” on Top right corner of the screen.  
– Open Firebug and click on Inspect button.  
– Select “Go” button. You will see something like this.



As you can see, id=”submit” has been associated with input button, which we can use as the locator in Selenium IDE.

– Open Selenium IDE and in Target box enter id=submit. Now Click on “Find” button, you’ll see the “Go” button gets highlighted.

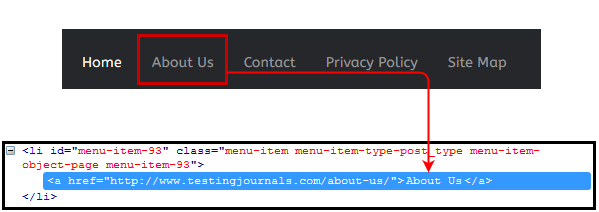
1. **Using Link Text Locator :**

This locator allows you to find the “Link” elements having hyperlink texts. The element is located by providing “link=” into IDE target field followed by hyperlink text.

Format : *link=About Us (Where “About Us” is the hyperlink text.)*

Example :

– Hit the URL : http://www.testingjournals.com in Firefox browser.  
– Right click on “About Us” from the header bar. As it is an hyperlink, you can get the locator value as shown below :



Using above mentioned format you can easily execute the command in IDE. Provide “link=About Us” in Target field and use command as “clickAndWait” and execute the command. You will be redirected to About Us page.

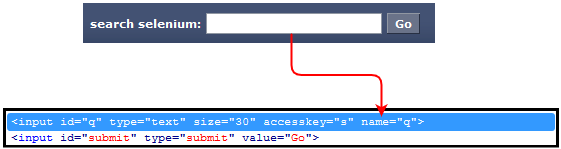
1. **Name Locator :**

Format : *name=q (Where “q” is the Name value assigned to the element.)*

Example :

– Hit the URL : http://docs.seleniumhq.org/ in Firefox browser.  
– Look for “search selenium” on Top right corner of the screen.  
– Open Firebug and click on Inspect button.  
– Select “Search” text-box. You will see something like this.

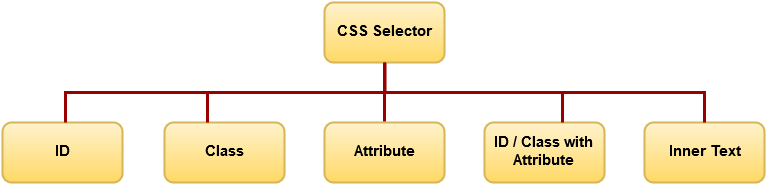
As shown in above image, name=”q” has been attached with search input box.



Same as ID locator, provide “name=” in the IDE target field followed by the value of Name attribute.

1. **CSS Selector :**

CSS locator can be formed in different ways. Most used and common are as shown below :



**CSS selector with Element ID :**

First step would be same as what we did in locating ID earlier in this tutorial. We will use the same example here.

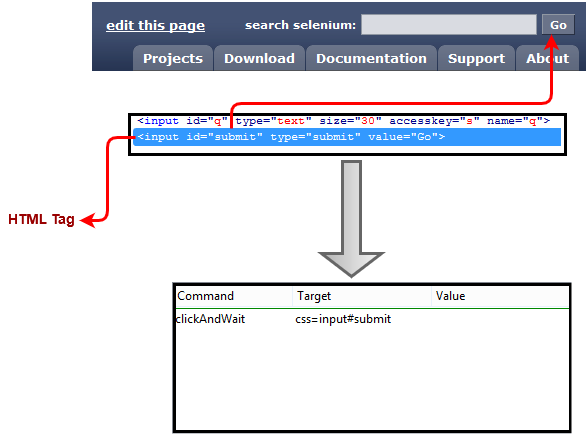
Format : *css=[HTML Tag]#[Element ID]*where,

HTML Tag = the HTML Tag of the interacted Element  
# = This Hash Tag should be present for each CSS locator using ID  
ID = ID locator of the interacted Element

Example :

– Hit the URL: http://docs.seleniumhq.org/ in Firefox browser.  
– Look for “search selenium” on Top right corner of the screen.  
– Open Firebug and click on Inspect button.  
– Select “Go” button.

Mentioned below is the generated CSS Locator  of “Go” button using Element’s ID. Don’t forget to use Find button to verify you have located proper element.



**CSS selector with Class :**

This CSS selector works with the class attribute of the web element. To showcase the example we will use Testing Journal’s global search button.

Format : *css=[HTML Tag].[Element Class Name]* where,

HTML Tag = the HTML Tag of the interacted Element  
. = This Dot should be present for each CSS locator using Class Name  
Class Name = Class name of the interacted Element which you can extract through Firebug

Example :

– Hit the URL : http://www.testingjournals.com  
– Open Firebug and locate the Search field from the Right side bar  
– Now generate the CSS locator for Search input field as mentioned in below image.



Hence, as mentioned in the above image we can generate the CSS locator using Class Name.

Note : Many times multiple elements have been allocated the same HTML Tag and Name. In such cases, the first element from the source code will be located. So, it better to avoid using Name in such cases.

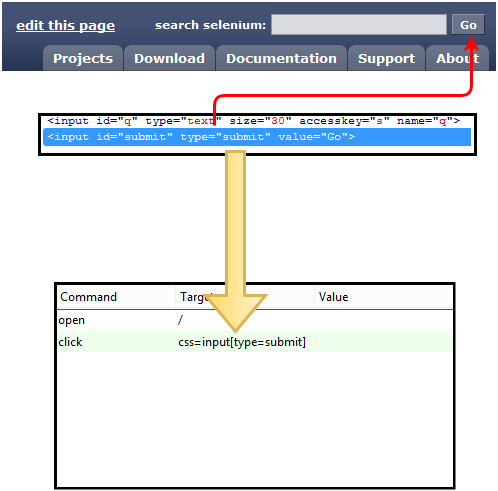
**CSS Selector with Attribute :**

Format : *css=HTML Tag[Attribute Value]* where,

HTML Tag = the HTML Tag of the interacted Element  
[] = Attribute value should be given in this bracket  
Attribute Value = Attribute value of the interacted Element which you can extract through Firebug

Example :

– Hit the URL : http://docs.seleniumhq.org/ in firefox browser.  
– Look for “search selenium” on Top right corner of the screen.  
– Open Firebug and click on Inspect button.  
– Select “Go” button. Now refer below image to generate CSS selector using element attribute value.



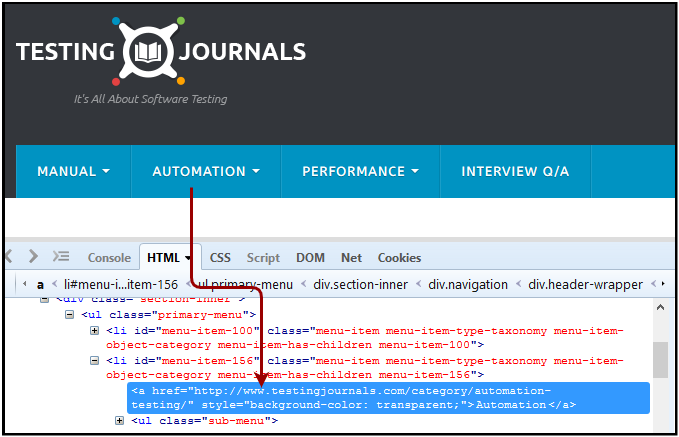
**CSS Selector with Inner Text :**

Format : *css=HTML Tag:contains(“Inner Text”)* where,

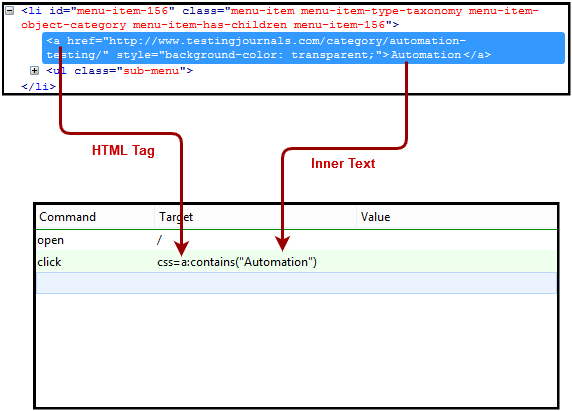
HTML Tag = the HTML Tag of the interacted Element  
:contains = Inner text should be provided in IDE command in this format  
Inner Text = Inner Text of the interacted Element which you can extract through Firebug

Example :

– Hit the URL : http://www.testingjournals.com  
– Open Firebug and locate the “Automation” as the Menu option in the header bar.



As you can see in above image, Inner text for the identified element is “Automation”. Now let’s locate this element through Selenium IDE using CSS with Inner text selector.



**LINK TEXT SELECTOR CODE -**

**import org.openqa.selenium.By;**

**import org.openqa.selenium.WebDriver;**

**import org.openqa.selenium.chrome.ChromeDriver;**

**public class MyClass {**

**public static void main(String[] args) {**

**String baseUrl = "http://demo.guru99.com/test/link.html";**

**System.setProperty("webdriver.chrome.driver","G:\\chromedriver.exe");**

**WebDriver driver = new ChromeDriver();**

**driver.get(baseUrl);**

**driver.findElement(By.linkText("click here")).click();**

**System.out.println("title of page is: " + driver.getTitle());**

**driver.quit();**

**}**

**}**

**PARTIAL LINK TEXT SELECTOR CODE -**

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

public class P1 {

public static void main(String[] args) {

String baseUrl = "http://demo.guru99.com/test/accessing-link.html";

System.setProperty("webdriver.chrome.driver","G:\\chromedriver.exe");

WebDriver driver = new ChromeDriver();

driver.get(baseUrl);

driver.findElement(By.partialLinkText("here")).click();

System.out.println("Title of page is: " + driver.getTitle());

driver.quit();

}

}

**USER INTERACTIONS CODE 1- SELENIUM FORMS CODE**

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.\*;

public class Form {

public static void main(String[] args) {

// declaration and instantiation of objects/variables

System.setProperty("webdriver.chrome.driver","G:\\chromedriver.exe");

WebDriver driver = new ChromeDriver();

String baseUrl = "http://demo.guru99.com/test/login.html";

driver.get(baseUrl);

// Get the WebElement corresponding to the Email Address(TextField)

WebElement email = driver.findElement(By.id("email"));

// Get the WebElement corresponding to the Password Field

WebElement password = driver.findElement(By.name("passwd"));

email.sendKeys("[abcd@gmail.com](mailto:abcd@gmail.com)

");

password.sendKeys("abcdefghlkjl");

System.out.println("Text Field Set");

// Deleting values in the text box

email.clear();

password.clear();

System.out.println("Text Field Cleared");

// Find the submit button

WebElement login = driver.findElement(By.id("SubmitLogin"));

// Using click method to submit form

email.sendKeys("[abcd@gmail.com](mailto:abcd@gmail.com)");

password.sendKeys("abcdefghlkjl");

login.click();

System.out.println("Login Done with Click");

//using submit method to submit the form. Submit used on password field

driver.get(baseUrl);

driver.findElement(By.id("email")).sendKeys("[abcd@gmail.com](mailto:abcd@gmail.com)");

driver.findElement(By.name("passwd")).sendKeys("abcdefghlkjl");

driver.findElement(By.id("SubmitLogin")).submit();

System.out.println("Login Done with Submit");

//driver.close();

}

}

**USER INTERACTIONS CODE 2 – RADIO BUTTONS**

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.\*;

public class Form {

public static void main(String[] args) {

// declaration and instantiation of objects/variables

System.setProperty("webdriver.chrome.driver","G:\\chromedriver.exe");

WebDriver driver = new ChromeDriver();

driver.get("http://demo.guru99.com/test/radio.html");

WebElement radio1 = driver.findElement(By.id("vfb-7-1"));

WebElement radio2 = driver.findElement(By.id("vfb-7-2"));

//Radio Button1 is selected

radio1.click();

System.out.println("Radio Button Option 1 Selected");

//Radio Button1 is de-selected and Radio Button2 is selected

radio2.click();

System.out.println("Radio Button Option 2 Selected");

// Selecting CheckBox

WebElement option1 = driver.findElement(By.id("vfb-6-0"));

// This will Toggle the Check box

option1.click();

// Check whether the Check box is toggled on

if (option1.isSelected()) {

System.out.println("Checkbox is Toggled On");

} else {

System.out.println("Checkbox is Toggled Off");

}

//Selecting Checkbox and using isSelected Method

driver.get("http://demo.guru99.com/test/facebook.html");

WebElement chkFBPersist = driver.findElement(By.id("persist\_box"));

for (int i=0; i<2; i++) {

chkFBPersist.click ();

System.out.println("Facebook Persists Checkbox Status is - "+chkFBPersist.isSelected());

}

//driver.close();

}

}

**USER INTERACTIONS CODE 3 – DROP DOWN BUTTONS**

package newpackage;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.openqa.selenium.support.ui.Select;

import org.openqa.selenium.By;

public class accessDropDown {

public static void main(String[] args) {

System.setProperty("webdriver.firefox.marionette","C:\\geckodriver.exe");

String baseURL = "http://demo.guru99.com/test/newtours/register.php";

WebDriver driver = new FirefoxDriver();

driver.get(baseURL);

Select drpCountry = new Select(driver.findElement(By.name("country")));

drpCountry.selectByVisibleText("ANTARCTICA");

//Selecting Items in a Multiple SELECT elements

driver.get("http://jsbin.com/osebed/2");

Select fruits = new Select(driver.findElement(By.id("fruits")));

fruits.selectByVisibleText("Banana");

fruits.selectByIndex(1);

}

}

**USER INTERACTIONS CODE 4 - MOUSE OVER CODE**

package newproject;

import org.openqa.selenium.\*;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.openqa.selenium.interactions.Action;

import org.openqa.selenium.interactions.Actions;

public class PG7 {

public static void main(String[] args) {

String baseUrl = "http://demo.guru99.com/test/newtours/";

System.setProperty("webdriver.firefox.marionette","C:\\geckodriver.exe");

WebDriver driver = new FirefoxDriver();

driver.get(baseUrl);

WebElement link\_Home = driver.findElement(By.linkText("Home"));

WebElement td\_Home = driver

.findElement(By

.xpath("//html/body/div"

+ "/table/tbody/tr/td"

+ "/table/tbody/tr/td"

+ "/table/tbody/tr/td"

+ "/table/tbody/tr"));

Actions builder = new Actions(driver);

Action mouseOverHome = builder

.moveToElement(link\_Home)

.build();

String bgColor = td\_Home.getCs sValue("background-color");

System.out.println("Before hover: " + bgColor);

mouseOverHome.perform();

bgColor = td\_Home.getCssValue("background-color");

System.out.println("After hover: " + bgColor);

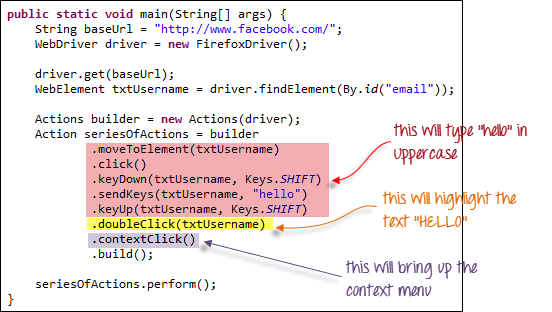
driver.close();

}

}

## **USER INTERACTIONS CODE 5 - MOUSE OVER AND KEY BOARD MULTIPLE ACTIONS**

**You can build a series of actions using the Action and Actions classes**. Just remember to close the series with the build() method. Consider the sample code below.

[](https://www.guru99.com/images/image053.png)

public static void main(String[] args) {

String baseUrl = "http://www.facebook.com/";

WebDriver driver = new FirefoxDriver();

driver.get(baseUrl);

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

Actions builder = new Actions(driver);

Action seriesOfActions = builder

.moveToElement(txtUsername)

.click()

.keyDown(txtUsername, Keys.SHIFT)

.sendKeys(txtUsername, "hello")

.keyUp(txtUsername, Keys.SHIFT)

.doubleClick(txtUsername)

.contextClick()

.build();

seriesOfActions.perform() ;

}

**USER INTERACTIONS CODE 6 - WINDOW HANDLE CODE**

import java.util.Iterator;

import java.util.Set;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

public class WindowHandle\_Demo {

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

WebDriver driver=new FirefoxDriver();

//Launching the site.

driver.get("http://demo.guru99.com/popup.php");

driver.manage().window().maximize();

driver.findElement(By.xpath("//\*[contains(@href,'popup.php')]")).click();

String MainWindow=driver.getWindowHandle();

// To handle all new opened window.

Set<String> s1=driver.getWindowHandles();

Iterator<String> i1=s1.iterator();

while(i1.hasNext())

{

String ChildWindow=i1.next();

if(!MainWindow.equalsIgnoreCase(ChildWindow))

{

// Switching to Child window

driver.switchTo().window(ChildWindow);

driver.findElement(By.name("emailid"))

.sendKeys("[gaurav.3n@gmail.com](mailto:gaurav.3n@gmail.com)

");

driver.findElement(By.name("btnLogin")).click();

// Closing the Child Window.

driver.close();

}

}

// Switching to Parent window i.e Main Window.

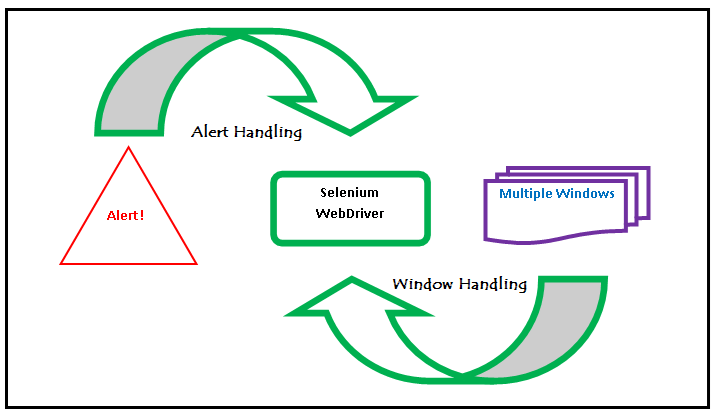
driver.switchTo().window(MainWindow);

}

}

**Output:**

When you execute the above code, it launches the site and on clicking the link "Click here," it opens up a child window in a new tab. You can close the child window, and switch to the parent window once the operation is completely done. Hence handling more than one window in the application.

[](https://www.guru99.com/images/3-2016/032216_1314_AlertPopuph18.png)

**USER INTERACTIONS CODE 7 - UPLOAD FILE CODE**

package newproject;

import org.openqa.selenium.\*;

import org.openqa.selenium.firefox.FirefoxDriver;

public class PG9 {

public static void main(String[] args) {

System.setProperty("webdriver.firefox.marionette","C:\\geckodriver.exe");

String baseUrl = "http://demo.guru99.com/test/upload/";

WebDriver driver = new FirefoxDriver();

driver.get(baseUrl);

WebElement uploadElement = driver.findElement(By.id("uploadfile\_0"));

// enter the file path onto the file-selection input field

uploadElement.sendKeys("C:\\newhtml.html");

// check the "I accept the terms of service" check box

driver.findElement(By.id("terms")).click();

// click the "UploadFile" button

driver.findElement(By.name("send")).click();

}

}

**References –**

1. Selenium Locators Examples - <http://www.testingjournals.com/understanding-selenium-locators/>
2. Selenium User Interactions - <http://guru99.com/checkbox-and-radio-button-webdriver.html>
3. Selenium User Interactions -<https://www.tutorialspoint.com/selenium/selenium_user_interactions.htm>
4. Advanced User Interactions -

<https://github.com/SeleniumHQ/selenium/wiki/Advanced-User-Interactions>

1. [Selenium Advanced User Interactions](http://www.theautomatedtester.co.uk/blog/2011/selenium-advanced-user-interactions.html) -http://www.theautomatedtester.co.uk/blog/2011/selenium-advanced-user-interactions.html
2. Know More About Selenium User Interactions -

<https://itelearn.com/blog/know-more-about-selenium-user-interactions/>

1. Actions Class in Selenium - <https://www.toolsqa.com/selenium-webdriver/actions-class-in-selenium/>

# How to handle Actions class in Selenium WebDriver?

<https://www.edureka.co/blog/keyboard-mouse-events-actions-class>

# Java-Selenium | Advanced User Interactions | Performing drag and drop operations

<https://cosmocode.io/java-selenium-advanced-user-interactions-performing-drag-and-drop-operations/>

1. Best selenium user Interactions of ‘2019’ **-** <https://onlineitguru.com/blog/best-selenium-user-interactions-of-2019>

# Java Code Examples for org.openqa.selenium.interactions.Actions -

<https://www.programcreek.com/java-api-examples/?api=org.openqa.selenium.interactions.Actions>