Selenium WebDriver- Handling Alerts

In this section, you will learn how to handle alerts in Selenium WebDriver.

Selenium WebDriver provides three methods to accept and reject the Alert depending on the Alert types.

1. void dismiss()

This method is used to click on the 'Cancel' button of the alert.

**Syntax:**

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

2. void accept()

This method is used to click on the 'Ok' button of the alert.

**Syntax:**

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

3. String getText()

This method is used to capture the alert message.

**Syntax:**

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

4. void sendKeys(String stringToSend)

This method is used to send some data to the alert box.

**Syntax:**

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

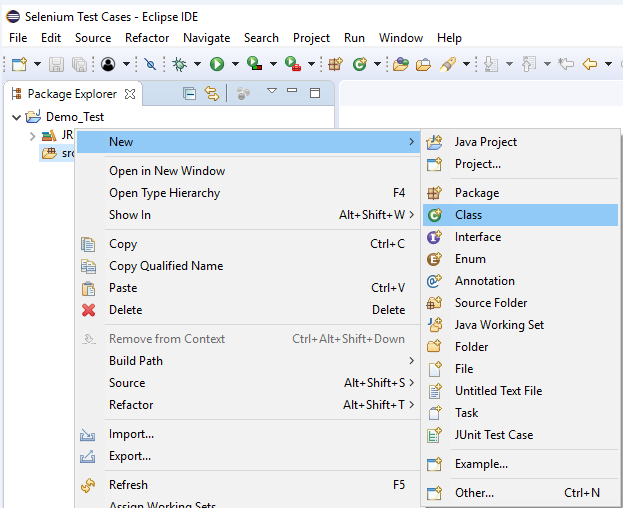
Let us consider a test case in which we will automate the following scenarios:

* Invoke Firefox Browser
* Open URL: <https://www.testandquiz.com/selenium/testing.html>
* Click on the "Generate Alert box" button
* Click on the "Generate Confirm box" button
* Close the browser

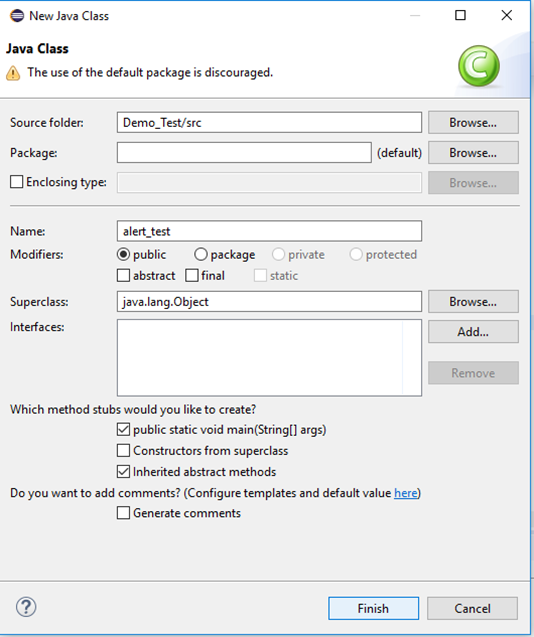
We will create our test case step by step in order to give you a complete understanding of how to handle alerts in WebDriver.

**Step1**. Launch Eclipse IDE and open the existing test suite "Demo\_Test" which we have created in earlier sessions of this tutorial.

**Step2**. Right click on the "src" folder and create a new Class File from **New >Class**.



Give your Class name as "alert\_test" and click on "Finish" button.



**Step3**. Let's get to the coding ground.

* To invoke Firefox browser, we need to download Gecko driver and set the system property for Gecko driver. We have already discussed this in earlier sessions of this tutorial. You can refer ["Running test on Firefox Browser"](https://www.javatpoint.com/selenium-webdriver-running-test-on-firefox-browser-gecko-driver) to learn how to download and set System property for Firefox driver.

Here is the sample code to set system property for Gecko driver:

1. // System Property for Gecko Driver
2. stem.setProperty("webdriver.gecko.driver","D:\\GeckoDriver\\geckodriver.exe" );

* After that we have to initialize Gecko Driver using Desired Capabilities Class.

Here is the sample code to initialize gecko driver using DesiredCapabilities class.

1. // Initialize Gecko Driver using Desired Capabilities Class
2. DesiredCapabilities capabilities = DesiredCapabilities.firefox();
3. capabilities.setCapability("marionette",**true**);
4. ebDriver driver= **new** FirefoxDriver(capabilities);

Combining both of the above code blocks, we will get the code snippet to launch Firefox browser.

1. // System Property for Gecko Driver
2. System.setProperty("webdriver.gecko.driver","D:\\GeckoDriver\\geckodriver.exe" );
4. // Initialize Gecko Driver using Desired Capabilities Class
5. DesiredCapabilities capabilities = DesiredCapabilities.firefox();
6. capabilities.setCapability("marionette",**true**);
7. WebDriver driver= **new** FirefoxDriver(capabilities);

* After that we need to write the code which will automate our second test scenario (navigate to the desired URL)

Here is the sample code to navigate to the desired URL:

1. // Launch Website
2. driver.navigate().to("https://www.testandquiz.com/selenium/testing.html");

The complete code till now will look something like this:

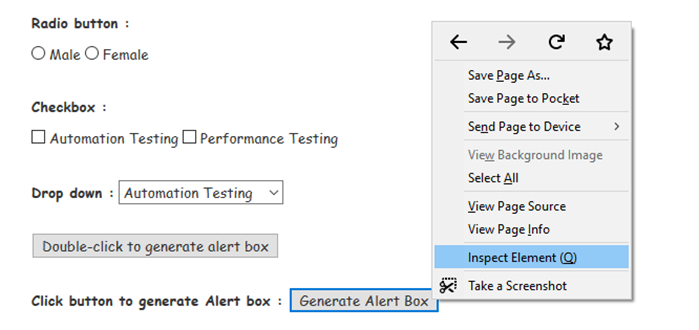
1. **import** org.openqa.selenium.WebDriver;
2. **import** org.openqa.selenium.firefox.FirefoxDriver;
3. **import** org.openqa.selenium.remote.DesiredCapabilities;
5. **public** **class** alert\_test {
7. **public** **static** **void** main(String[] args) {
9. // System Property for Gecko Driver
10. System.setProperty("webdriver.gecko.driver","D:\\GeckoDriver\\geckodriver.exe" );
12. // Initialize Gecko Driver using Desired Capabilities Class
13. DesiredCapabilities capabilities = DesiredCapabilities.firefox();
14. capabilities.setCapability("marionette",**true**);
15. WebDriver driver= **new** FirefoxDriver(capabilities);

18. // Launch Website
19. driver.navigate().to("https://www.testandquiz.com/selenium/testing.html");
21. }
23. }

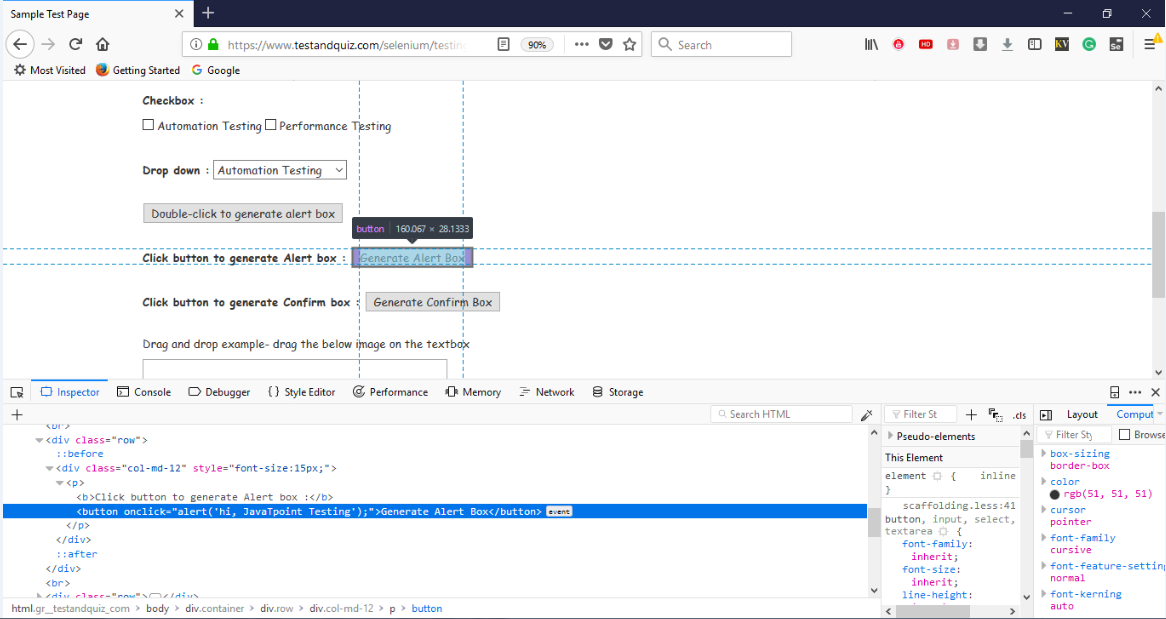
**Step4**. Now, we will try to locate the "Generate Alert Box" and "Generate Confirm Box" in order to perform alert handling operation. As we know that locating an element involves inspection of its HTML codes.

Follow the steps given below to locate the drop-down menu on the sample web page.

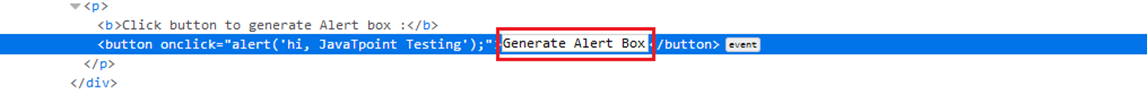
* Open URL: <https://www.testandquiz.com/selenium/testing.html>
* Right click on the "Generate Alert Box" button and select Inspect Element.



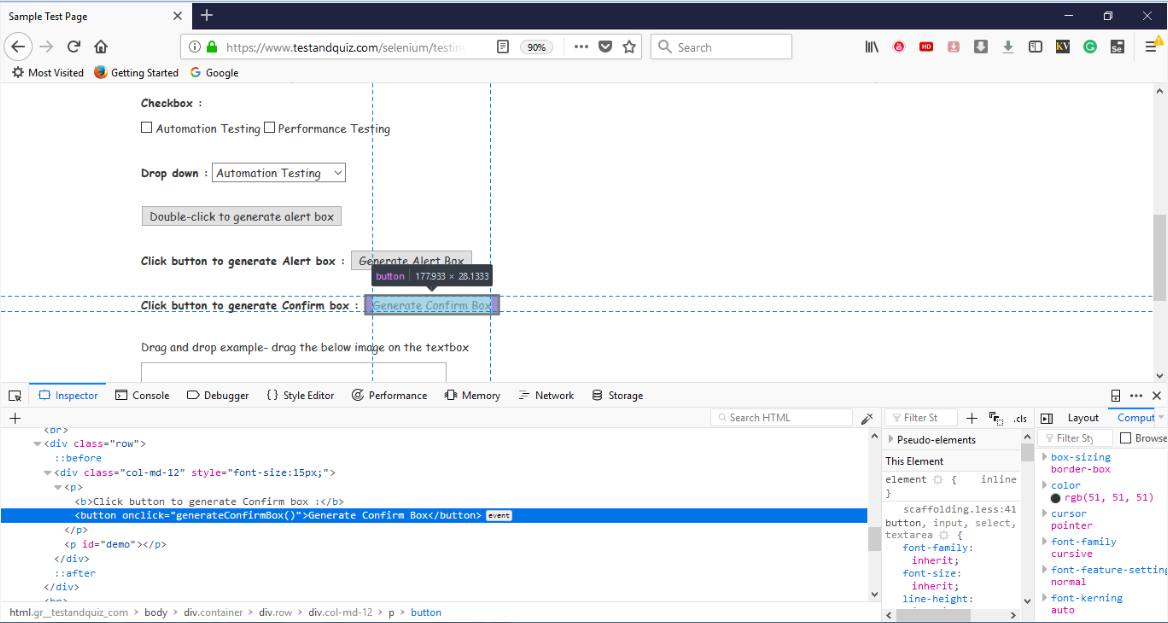
* It will launch a window containing all the specific codes involved in the development of the "Generate Alert Box" button.



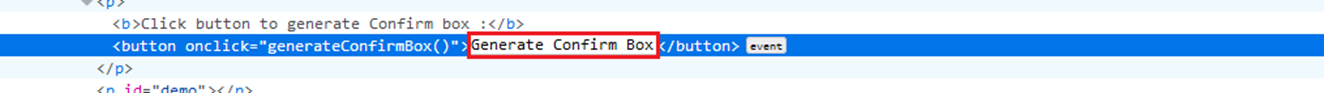
* Take a note of its link text i.e. "Generate Alert Box"



Similarly, we will inspect the "Generate Confirm box" button.



* Take a note of its link text i.e. "Generate Confirm Box"



**Step5.**

To automate our third and fourth test scenario, we need to write the code which click and accept the **Generate Alert box** as well as click and accept the Generate Confirm box.

Given below is the code snippet to automate our third and fourth test scenario.

1. //Handling alert boxes
2. //Click on generate alert button
3. driver.findElement(By.linkText("Generate Alert Box")).click();
5. //Using Alert class to first switch to or focus to the alert box
6. Alert alert = driver.switchTo().alert();
8. //Using accept() method to accep the alert box
9. alert.accept();
11. //Handling confirm box
12. //Click on Generate Confirm Box
13. driver.findElement(By.linkText("Generate Confirm Box")).click();

16. Alert confirmBox = driver.switchTo().alert();
18. //Using dismiss() command to dismiss the confirm box
19. //Similarly accept can be used to accept the confirm box
20. confirmBox.dismiss();

Thus, our final test script will look something like this:

1. **import** org.openqa.selenium.By;
2. **import** org.openqa.selenium.WebDriver;
3. **import** org.openqa.selenium.firefox.FirefoxDriver;
4. **import** org.openqa.selenium.remote.DesiredCapabilities;
5. **import** org.openqa.selenium.Alert;
6. **public** **class** alert\_test {
8. **public** **static** **void** main(String[] args) {
10. // System Property for Gecko Driver
11. System.setProperty("webdriver.gecko.driver","D:\\GeckoDriver\\geckodriver.exe" );
13. // Initialize Gecko Driver using Desired Capabilities Class
14. DesiredCapabilities capabilities = DesiredCapabilities.firefox();
15. capabilities.setCapability("marionette",**true**);
16. WebDriver driver= **new** FirefoxDriver(capabilities);

19. // Launch Website
20. driver.navigate().to("https://www.testandquiz.com/selenium/testing.html");
22. //Handling alert boxes
23. //Click on generate alert button
24. driver.findElement(By.linkText("Generate Alert Box")).click();
26. //Using Alert class to first switch to or focus to the alert box
27. Alert alert = (Alert) driver.switchTo().alert();
29. //Using accept() method to accept the alert box
30. alert.accept();
32. //Handling confirm box
33. //Click on Generate Confirm Box
34. driver.findElement(By.linkText("Generate Confirm Box")).click();

37. Alert confirmBox = (Alert) driver.switchTo().alert();
39. //Using dismiss() command to dismiss the confirm box
40. //Similarly accept can be used to accept the confirm box
41. ((Alert) confirmBox).dismiss();


45. }
47. }

Upon execution, the above test script will launch the Firefox browser and automate all the test scenarios.