Selenium WebDriver - Event Listeners

- Event listener in Selenium WebDriver is generally used to collect logs of WebDriver, by tracking all the activities that the WebDriver is performing say 'Navigating to page', 'Clicking a button' etc.
- To implement event listener in Selenium WebDriver, we have to know how to use the below:
  - **EventFiringWebDriver** (Selenium Class)
    - Will throw the events around WebDriver
    - We will be using this in our Selenium Code to throw the events
  - **WebDriverEventListener** (Selenium Interface)
    - WebDriverEventListener will catch the events that are thrown around WebDriver in Selenium code
- Create a new class say Demo, in which we will write the Selenium Code - Demonstrate here
  - Provide the dependencies for selenium webdriver in pom.xml file
- Create a new class say 'MyEventsHandler' which implements **WebDriverEventListener**
  - As WebDriverEventListener is an interface, all its predefined methods dont contain any method body.
  - Class implementing the interface needs to define the methods body for all the methods of WebDriverEventListener interface
    - Select to add the unimplemented methods
- Start implementing the methods which we need to track as part of our Selenium code
  - Go to Seleniumhq.org > Javadocs > WebDriverEventListener interface and view all the predefined methods of WebDrivenEventListener interface
  - Write the Selenium code for navigating to http://www.omayo.blogspot.com page in **two ways** - Demonstrate here
  - Implement EventFiringWebDriver class in the above selenium code
    - Go to Seleniumhq.org > Javadocs and view all the classes implementing the WebDriver interface
    - FirefoxDriver, ChromeDriver, InternetExplorer classes implementing WebDriver interface won't have the capability to throw the events
    - EventFiriingWebDriver class implementing the WebDriver interface can be used to throw the events
      - As EventFiriningWebDriver implements WebDriver interface, all the predefined methods of WebDriver like get(), findElement(), findElements() etc can be accessed using EventFiringWebDriver class
      - Hence EventFiringWebDriver classes uses the predefined methods of WebDriver interface to throw the events
    - Apart from the predefined methods of WebDriver interface, **EventFiringWebDriver** class has its own predefined methods
      - **register()** is used to register the Class which implements the predefined methods of **WebDriverEventListener** interface
      - **unregister()** is used to unregister the Class which implements the predefined methods of **WebDriverEventListner** interface
    - Create an object for EventFiringWebDriver class in Demo class haivng Selenium Code
      - Provide the driver object as an input to EventFiringWebDriver() constructor
      - Instead of using WebDriver object reference for performing automation operations, we are going to use the object reference of EventFiringWebDriver class going a head.
    - Register the class which is implementing the predefined methods of WebDriverEventListener interface, using register() predefined methods of EventFiringWebDriver class
      - Create an object for the class which is implementing the predefined methods of WebDriverEventListener interface and pass it to the above register() method
    - Using object reference of EventFiringWebDriver class, perform the automation tasks navigating to a page using get() and navigate().to() methods
      - Replace the driver object with the object of EventFiringWebDriver - Demonstrate here

- Implement the beforeNavigateTo() and afterNavigateTo() methods in MyEventsHandler class - Demonstrate here
  - **beforeNavigateTo()**
    - An event will be captured by this method before navigating to any page in Selenium Automation code.
    - **get()** method and **navigate().to()** in Selenium API will help us navigate to the pages in Selenium Automation.
  - **afterNavigateTo()**
    - An event will be captured by this method after navigating to any page in Selenium Automation code.
    - **get()** method and **navigate().to()** in Selenium API will help us navigate to the pages in Selenium Automation.
- Execute the program to check whether the WebDriver logs are getting printed in the output console
- Implement the beforeFindBy() and afterFindBy() methods in MyEventsHandler class - Demonstrate here
  - Find and Create a WebElement for search box field in omayo.blogspot.com - Demonstrate here
  - **beforeFindBy()**
    - An event will be captured by this method before finding any element.
    - **driver.findElement(By)** and **driver.findElements(By)** in Selenium API will help us in finding the element or elements.
  - **afterFindBy()**
    - An event will be captured by this method after finding any element.
    - **driver.findElement(By)** and **driver.findElements(By)** in Selenium API will help us in finding the element or elements.
- Execute the program to check whether the WebDriver logs are getting printed in the output console
- Implement the beforeChangeValueOf() and afterChangeValueOf() methods in MyEventsHandler class - Demonstrate here
  - Clear and Enter text into the search box field in omayo.blogspot.com - Demonstrate here
  - **beforeChangeValueOf()**
    - An event will be captured before changing the value of any elements value attribute.
    - **clear()** and **sendKeys()** commands in Selenium API will help us in changing the value of elements.
  - **afterChangeValueOf()**
    - An event will be captured after changing the value of any elements value attribute.
    - **clear()** and **sendKeys()** commands in Selenium API will help us in changing the value of elements.
- Execute the program to check whether the WebDriver logs are getting printed in the output console
- Implement the beforeClickOn() and afterClickOn() methods in MyEventsHandler class - Demonstrate here
  - Click on 'ClickToGetAlert' button in omayo.blogspot.com - Demonstrate here
  - **beforeClickOn()**
    - An event will be captured by this method before clicking any element.
    - **click()** in Selenium API will help us in clicking the element.
  - **afterClickOn()**
    - An event will be captured by this method after clicking any element.
    - **click()** in Selenium API will help us in clicking the element.

- Execute the program to check whether the WebDriver logs are getting printed in the output console
- Implement the beforeAlertAccept() and afterAlertAccept() methods in MyEventsHandler class - Demonstrate here
- Accept the dispalyed alert in omayo.blogspot.com - Demonstrate here
- Execute the program to check whether the WebDriver logs are getting printed in the output console
- Implement the beforeAlertDismiss() and afterAlertDismiss() methods in MyEventsHandler class - Demonstrate here
- Accept the dispalyed alert in omayo.blogspot.com - Demonstrate here
- Execute the program to check whether the WebDriver logs are getting printed in the output console
- Implement the beforeNavigateRefresh() and afterNavigateRefresh() methods in MyEventsHandler class - Demonstrate here
- Refresh the page in omayo.blogspot.com - Demonstrate here
- Execute the program to check whether the WebDriver logs are getting printed in the output console
- Implement the beforeScript() and afterScript() methods in MyEventsHandler class - Demonstrate here
- Refresh the page uinsg JavaScript in omayo.blogspot.com - Demonstrate here
- Execute the program to check whether the WebDriver logs are getting printed in the output console
- Implement the beforeGetText() and afterGetText() methods in MyEventsHandler class - Demonstrate here
- Retrieve the text in omayo.blogspot.com - Demonstrate here
- Execute the program to check whether the WebDriver logs are getting printed in the output console
- Implement the beforeSwitchToWindow() and afterSwitchToWindow() methods in MyEventsHandler class - Demonstrate here
- Switch to a window in omayo.blogspot.com - Demonstrate here
- Execute the program to check whether the WebDriver logs are getting printed in the output console
- Implement the beforeGetScreenshotAs() and afterGetScreenshotAs() methods in MyEventsHandler class - Demonstrate here
- Take a screenshot in omayo.blogspot.com - Demonstrate here
- Execute the program to check whether the WebDriver logs are getting printed in the output console
- Implement the beforeNavigateBack(), afterNavigateBack(), beforeNavigateForward() and afterNavigateForward() methods in MyEventsHandler class - Demonstrate here
- Navigate back and forth in omayo.blogspot.com - Demonstrate here
- Execute the program to check whether the WebDriver logs are getting printed in the output console
- Implement the onException() - Demonstrate here
- Intentionally get an exception - Demonstrate here
- Execute the program to check whether the WebDriver logs are getting printed in the output console

- Unregister the class which is implementing the predefined methods of WebDriverEventListener interface, using unregister() predefined method of EventFiringWebDriver class - Demonstrate here
- View complete Selenium Code here
- View complete EventHandler Code here
