

Practical No. 4. Browser command and navigation Commands in Selenium.**Date:** _____**Aim:**

To learn Browser command and navigation Commands in Selenium.

Theory:

As we know, WebDriver talks to individual browsers natively. This way it has better control, not just on the web page, but on the browser itself. Navigate is one such feature of WebDriver that allows the test script developer to work with the browser's Back, Forward, and Refresh controls. Using the WebDriver's navigation feature, you can emulate those actions.

Navigation Commands**1. navigate()**

The method that is used for this purpose is **navigate()**.

The following is its API syntax:

WebDriver.Navigation navigate()

There is no input parameter for this method, but the return type is the WebDriver.Navigation interface, which contains all of the browser navigation options that help you navigate through your browser's history.

2. to();

the to() method is used to navigate to a web URL.

The API syntax is as follows:

void to(java.lang.String url)

The input parameter for this method is the url string that has to be loaded in the browser. This method will load the page in the browser by using the HTTP GET operation, and it will block everything else until the page is completely loaded. This method is the same as the driver.get(String url) method

3. back()

This method is used to emulate our browser's Back button.

The API syntax for this method is pretty straightforward, as follows:

void back()

This method doesn't take any input and doesn't return anything as well, but takes the browser one level back in its history.

4. forward()

This method is pretty much similar to the back() method, but takes the browser one level in the oppositedirection.

The API syntax for the method is as follows:

```
void forward()
```

This method doesn't take any input and doesn't return anything as well, but takes the browser one level forward in its history

5. refresh()

This method will reload the current URL to emulate the browser's refresh (F5 key) action.

The API syntax is as follows:

```
void refresh()
```

Browser Commands

The very basic browser operations of WebDriver include opening a browser; perform few tasks and then closing the browser.

Given are some of the most commonly used Browser commands for Selenium WebDriver.

1. void get(String arg0)

In WebDriver, this method loads a new web page in the existing browser window. It accepts String as parameter and returns void.

The respective command to load a new web page can be written as:

```
driver.get(URL);
```

// Or can be written as

```
String URL = "URL";
```

```
driver.get(URL);
```

2. String getTitle();

In WebDriver, this method fetches the title of the current web page. It accepts no parameter and returns a String.

The respective command to fetch the title of the current page can be written as:

```
driver.getTitle();
```

// Or can be written as

```
String Title = driver.getTitle();
```

3. getCurrentUrl(): String

In WebDriver, this method fetches the string representing the Current URL of the current web page. It accepts nothing as parameter and returns a String value.

The respective command to fetch the string representing the current URL can be written as:

```
driver.getCurrentUrl();  
//Or can be written as  
String CurrentUrl = driver.getCurrentUrl();
```

4. getPageSource(): String

In WebDriver, this method returns the source code of the current web page loaded on the current browser. It accepts nothing as parameter and returns a String value.

The respective command to get the source code of the current web page can be written as:

```
driver.getPageSource();  
//Or can be written as  
String PageSource = driver.getPageSource();
```

5. close(): void

This method terminates the current browser window operating by WebDriver at the current time. If the current window is the only window operating by WebDriver, it terminates the browser as well. This method accepts nothing as parameter and returns void.

The respective command to terminate the browser window can be written as:

```
driver.close();
```

6. quit(): void

This method terminates all windows operating by WebDriver. It terminates all tabs as well as the browser itself. It accepts nothing as parameter and returns void.

The respective command to terminate all windows can be written as:

```
driver.quit();
```

Implementation

1. Write a selenium script to navigate to <https://www.toolsqa.com/selenium-training/> website and perform following actions-1)Click on Registration button link,2) Come back to Home page (Use 'Back' command), 3)Again goback to Registration page (This time use 'Forward' command), 4) Again come back to Home page (This time use 'To' command), 5) Refresh the Browser (Use 'Refresh' command)

Program:

```
package SeleniumScript;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;

public class seleniumTraining {
    public static void main(String[] args) throws InterruptedException {
        // TODO Auto-generated method stub
        System.setProperty("webdriver.gecko.driver", "E:\\Selenium\\geckodriver.exe");
        WebDriver driver = new FirefoxDriver();
        driver.get("https://www.toolsqa.com/selenium-training/");

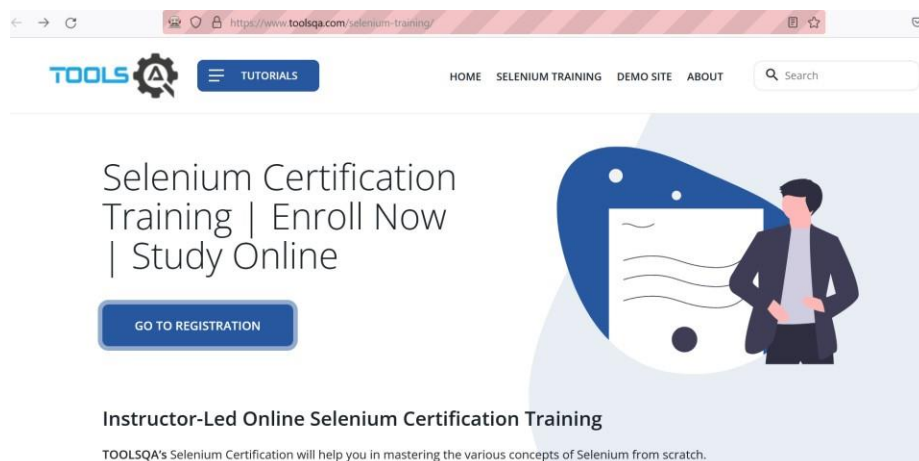
        WebElement regiBtn=driver.findElement(By.cssSelector("a[href='#enroll-form']"));
        regiBtn.click();
        Thread.sleep(2000);

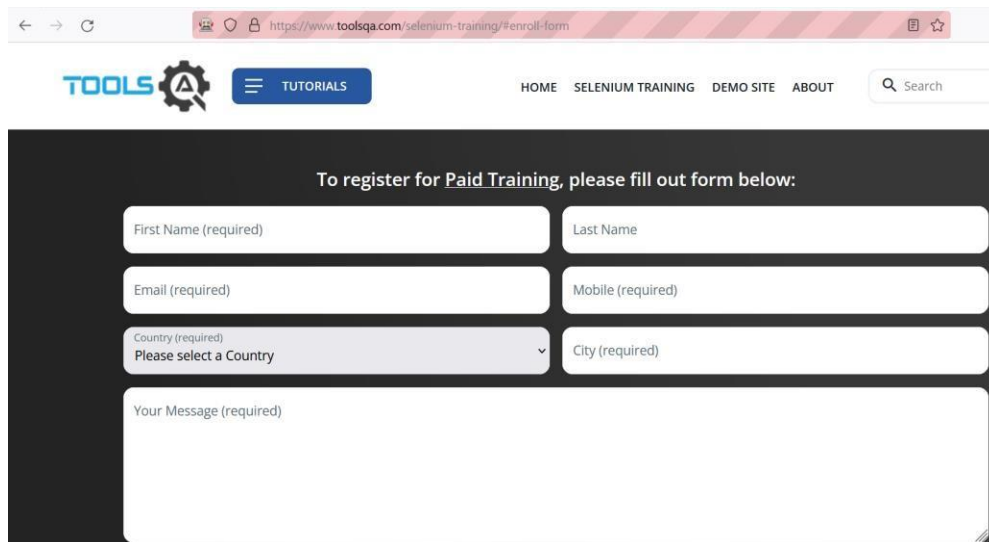
        driver.navigate().back();
        Thread.sleep(2000);

        driver.navigate().forward();
        Thread.sleep(2000);

        driver.navigate().to("https://www.toolsqa.com/selenium-training/");
        Thread.sleep(2000);
        driver.navigate().refresh();
    }
}
```

Output:





The screenshot shows a web browser window with the URL <https://www.toolsqa.com/selenium-training/#enroll-form>. The page has a dark header with the 'TOOLSQA' logo, a 'TUTORIALS' button, and navigation links for 'HOME', 'SELENIUM TRAINING', 'DEMO SITE', and 'ABOUT'. A search bar is also present. The main content area is a dark box with the text 'To register for Paid Training, please fill out form below:'. Below this text is a registration form with the following fields: 'First Name (required)', 'Last Name', 'Email (required)', 'Mobile (required)', 'Country (required)' (a dropdown menu with 'Please select a Country' as the selected option), 'City (required)', and 'Your Message (required)' (a large text area).

2. Automate the following test scenarios:

- Invoke firefox Browser
- Open URL: <https://www.google.co.in/>
- Get Page Title name and Title length
- Print Page Title and Title length on the Eclipse Console
- Get page URL and verify whether it is the desired page or not
- Get page Source and Page Source length
- Print page Length on Eclipse Console.
- Close the Browser

Program:

```
package SeleniumScript;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;

public class q2 {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        System.setProperty("webdriver.gecko.driver", "E:\\Selenium\\geckodriver.exe");
        WebDriver driver = new FirefoxDriver();
        driver.get("https://www.google.co.in/");

        System.out.println("Title of page "+driver.getTitle());
        System.out.println("Length of Title Page:"+driver.getTitle().length());

        System.out.println("Page Url "+driver.getCurrentUrl());

        if(driver.getCurrentUrl().equalsIgnoreCase("https://www.google.co.in/"))
        {
            System.out.println("First name textbox displayed");
        }
        else {
            System.out.println("First name textbox not displayed");
        }
    }
}
```

```

    }
    String pageSource=driver.getPageSource();

    System.out.println("Page Source Length:"+pageSource.length());
    driver.close();
    System.out.println("Browser is closed");
}
}

```

Output:

```

<terminated> q2 [Java Application] C:\Users\Asus\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\bin\javaw.exe (28-No
DevTools listening on ws://127.0.0.1:54130/devtools/browser/750593bc-9bcb-4357-87
Nov 28, 2022 11:42:44 AM org.openqa.selenium.remote.ProtocolHandshake createSessi
INFO: Detected upstream dialect: W3C
Title of page Google
Length of Title Page:6
Page Url https://www.google.co.in/
First name textbox displayed
Page Source Length:152605
1669615967982 RemoteAgent INFO Perform WebSocket upgrade for incoming cc
JavaScript error: chrome://remote/content/server/WebSocketHandshake.sys.mjs, line
System info: host.info: {browserName: 'chrome', browserVersion: '96.0.4664.45', chrome: {devtools: {page: {source: 'chrome://remote/content/server/WebSocketHandshake.sys.mjs', line: 1}}}},
Driver info: driver.version: RemoteWebDriver
    at org.openqa.selenium.remote.http.netty.NettyWebSocket.<init>(NettyWebS
    at org.openqa.selenium.remote.http.netty.NettyWebSocket.lambda$create$3(
    at org.openqa.selenium.remote.http.netty.NettyClient.openSocket(NettyCli
    at org.openqa.selenium.devtools.Connection.<init>(Connection.java:77)
    at org.openqa.selenium.firefox.FirefoxDriver.maybeGetDevTools(FirefoxDri
    at org.openqa.selenium.remote.RemoteWebDriver.close(RemoteWebDriver.java
    at Selenium_Scripts.q2.main(q2.java:26)

1669615967996 Marionette INFO Stopped listening on port 62545
Browser is closed

```

Conclusion: Learnt to use Browser command and navigation Commands in Selenium