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();

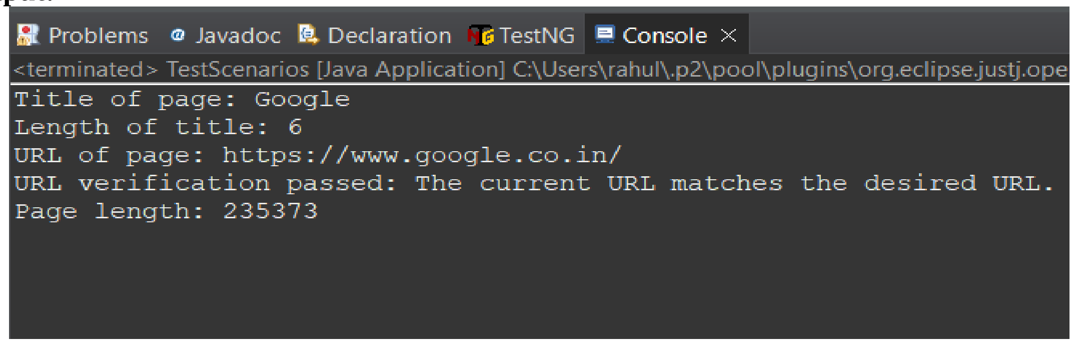
Implementation

1. **Write a selenium script to navigate to https://www.toolsqa.com/selenium-training/ website and performfollowing 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)**

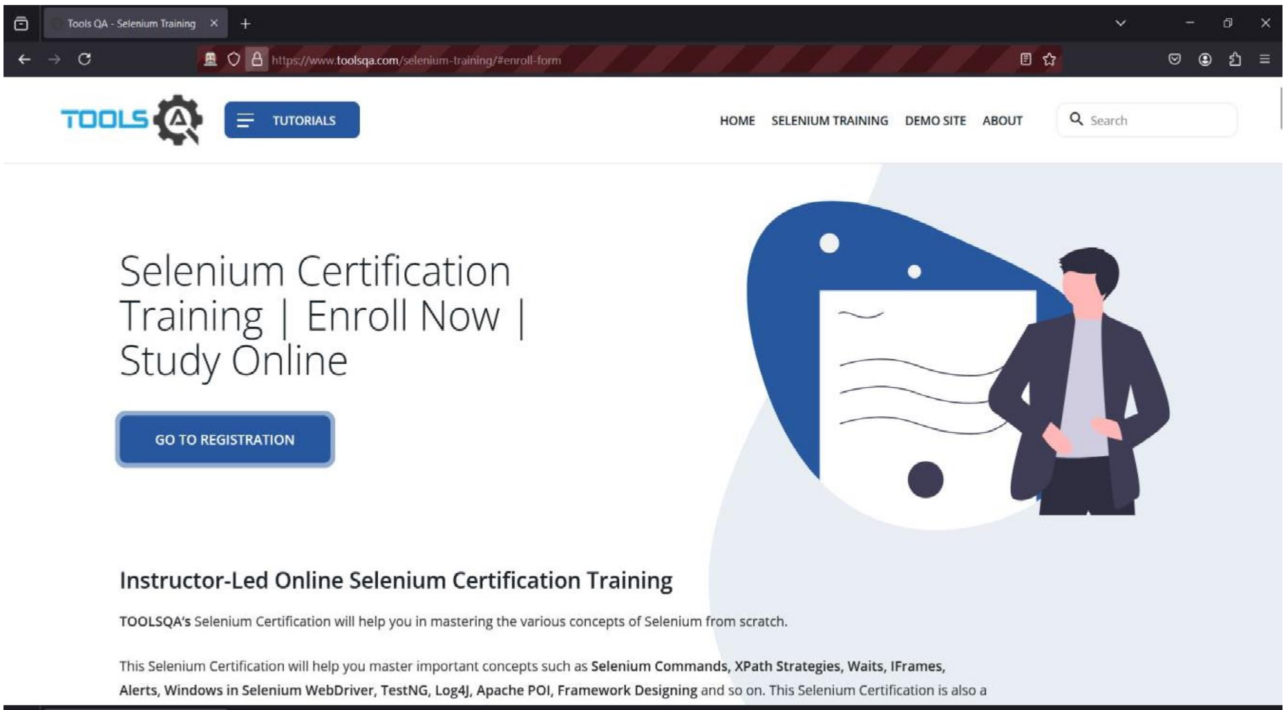
Program :

```
package selenium_test;

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.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import java.time.Duration;

public class ExploringNavigate {
    public static void main(String[] args) throws InterruptedException
{
        // Set path to Gecko driver
        System.setProperty("webdriver.gecko.driver",
"E:\\Selenium_setup\\geckodriver.exe");

        // Create instance of Firefox WebDriver
        WebDriver driver = new FirefoxDriver();

        // Set implicit wait
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(10));

        // Navigate to the ToolsQA Selenium Training webpage
        driver.get("https://www.toolsqa.com/selenium-training/");
        System.out.println("Navigated to ToolsQA Selenium Training page");

        // Wait for 2 seconds
        Thread.sleep(2000);

        // Locate and click on the Registration button using CSS Selector
       WebElement registrationButton = driver.findElement(By.cssSelector(".btn.btn-
    primary- shadow.btn-block"));
        registrationButton.click();
        System.out.println("Clicked on Registration button");

        // Wait for 2 seconds
        Thread.sleep(2000);

        // Come back to the Home page using the 'Back' command
        driver.navigate().back();
        System.out.println("Navigated back to the Home page");

        // Wait for 2 seconds
        Thread.sleep(2000);

        // Go back to the Registration page using the 'Forward' command
        driver.navigate().forward();
        System.out.println("Navigated forward to the Registration page");

        // Wait for 2 seconds
        Thread.sleep(2000);

        // Come back to the Home page again using the 'To' command
        driver.navigate().to("https://www.toolsqa.com/selenium-training/");
        System.out.println("Navigated to Home page using the 'to()'
command");

        // Wait for 2 seconds
        Thread.sleep(2000);

        // Refresh the browser
        driver.navigate().refresh();
        System.out.println("Refreshed the browser");

        // Quit the browser
        driver.quit();
        System.out.println("Closed the browser");
    }
}
```

Output:



```
<terminated> TestScenarios [Java Application] C:\Users\rahul\.p2\pool\plugins\org.eclipse.justj.ope
Title of page: Google
Length of title: 6
URL of page: https://www.google.co.in/
URL verification passed: The current URL matches the desired URL.
Page length: 235373
```

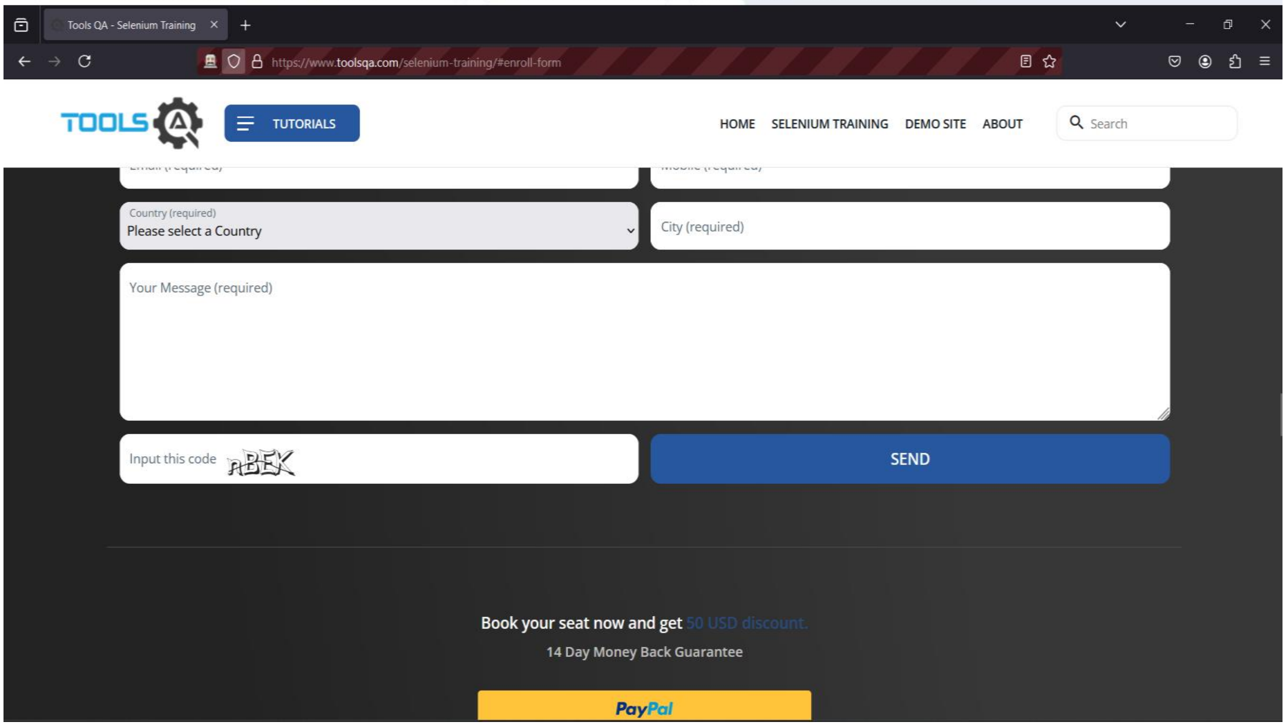

Tools QA - Selenium Training
https://www.toolsqa.com/selenium-training/#enroll-form
TOOLS
TUTORIALS
HOME
SELENIUM TRAINING
DEMO SITE
ABOUT
Search
GO TO REGISTRATION

Selenium Certification Training | Enroll Now | Study Online

Instructor-Led Online Selenium Certification Training

TOOLSQA's Selenium Certification will help you in mastering the various concepts of Selenium from scratch.

This Selenium Certification will help you master important concepts such as **Selenium Commands, XPath Strategies, Waits, IFrames, Alerts, Windows in Selenium WebDriver, TestNG, Log4J, Apache POI, Framework Designing** and so on. This Selenium Certification is also a



Book your seat now and get 50 USD discount.

14 Day Money Back Guarantee

2. Automate the following test scenarios:

a) Invoke firefox Browser

b) Open URL: https://www.google.co.in/

c) Get Page Title name and Title length

d) Print Page Title and Title length on the Eclipse Console

e) Get page URL and verify whether it is the desired page or not

f) Get page Source and Page Source length

g) Print page Length on Eclipse Console.

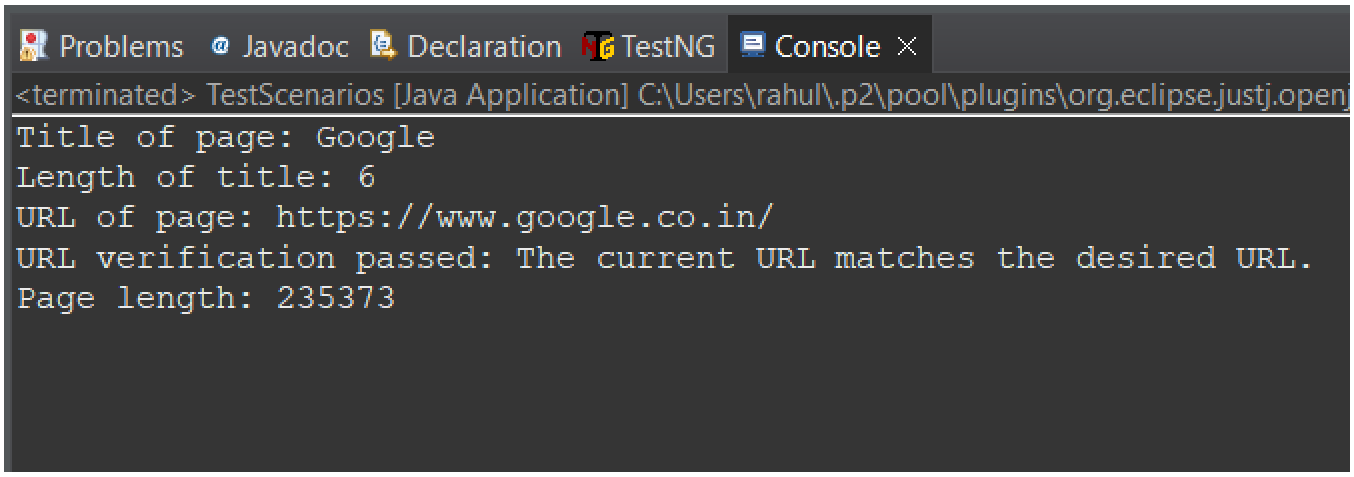
h) Close the BrowserProgram

Code:

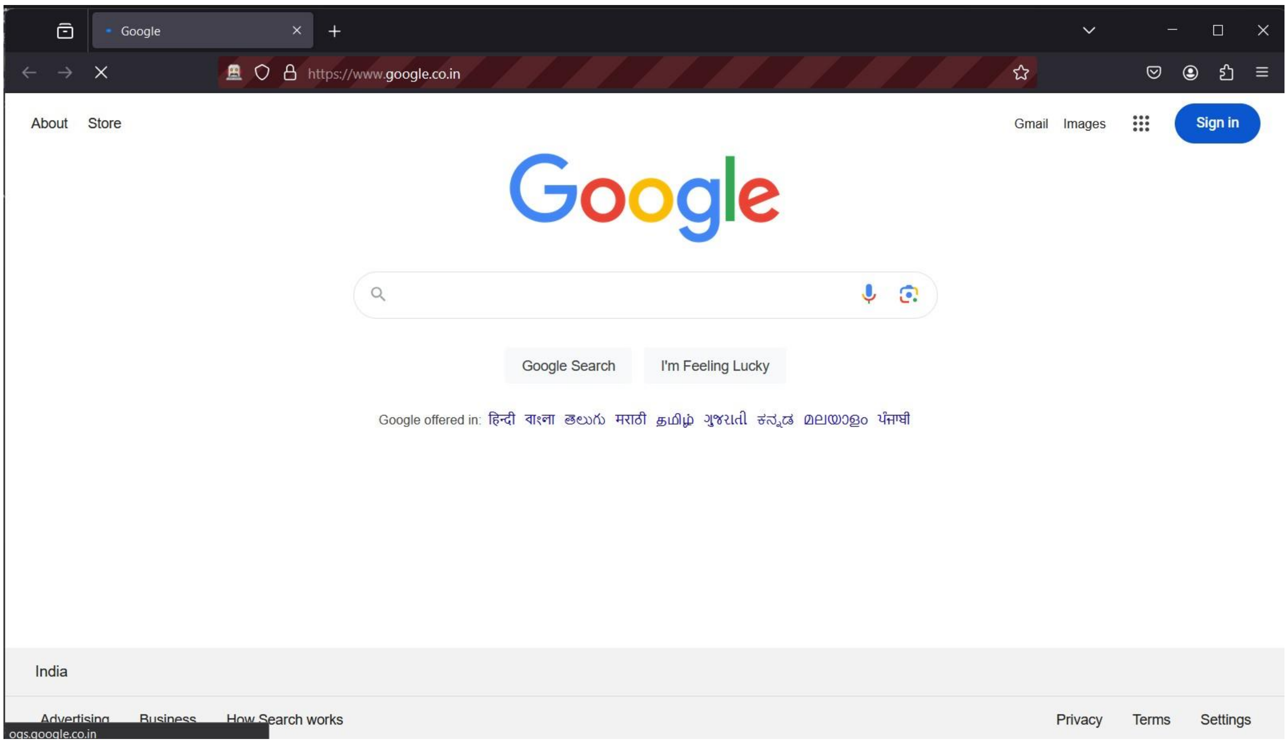
```
package selenium_test;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;

public class TestScenarios {
  public static void main(String[] args) {
    // Set the path to the Gecko driver
      System.setProperty("webdriver.gecko.driver", "E:\\Selenium_setup\\geckodriver.exe");


    // Create instance of Firefox WebDriver
    WebDriver driver = new FirefoxDriver();

    try {
      // Open URL: https://www.google.co.in/
      driver.get("https://www.google.co.in/");
      // Get Page Title name and Title length
      String title = driver.getTitle();
      int titleLength = title.length();

      // Print Page Title and Title length on the Eclipse Console
      System.out.println("Title of page: " + title);
      System.out.println("Length of title: " + titleLength);


      // Get page URL and verify whether it is the desired page or not
      String currentUrl = driver.getCurrentUrl();
      System.out.println("URL of page: " + currentUrl);
```

```
            if (currentUrl.equals("https://www.google.co.in/")) {
                System.out.println("URL verification passed: The current URL matches the desired URL.");
            } else {
                System.out.println("URL verification failed: The current URL does not match the desired URL.");
            }
            // Get page Source and Page Source length
            String pageSource = driver.getPageSource();
            int pageSourceLength = pageSource.length();

            // Print page Source length on Eclipse Console
            System.out.println("Page length: " + pageSourceLength);
        } finally {
            // Close the Browser
            driver.quit();
        }
    }
}
```

Output:



```
<terminated> TestScenarios [Java Application] C:\Users\rahul\.p2\pool\plugins\org.eclipse.justj.openj
Title of page: Google
Length of title: 6
URL of page: https://www.google.co.in/
URL verification passed: The current URL matches the desired URL.
Page length: 235373
```



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