Selenium WebDriver is a core component of the Selenium suite, designed for automating web browsers. It provides a programming interface to interact with web elements, simulate user interactions, and perform assertions. [Unlike its predecessor, Selenium RC (Remote Control), WebDriver communicates directly with the browser, offering a more stable and efficient means of automation1](https://www.selenium.dev/documentation/webdriver/)[2](https://www.geeksforgeeks.org/introduction-to-selenium-webdriver/).

[WebDriver supports multiple programming languages, including Java, Python, C#, and JavaScript, making it versatile for developers working in different technology stacks2](https://www.geeksforgeeks.org/introduction-to-selenium-webdriver/). [It allows for cross-browser testing, ensuring consistent behavior across various browsers like Chrome, Firefox, Safari, and Internet Explorer](https://www.selenium.dev/documentation/webdriver/)[2](https://www.geeksforgeeks.org/introduction-to-selenium-webdriver/).

Selenium WebDriver is a powerful tool for automating web browsers. It provides a programming interface to create and run test scripts across various browsers, like Chrome, Firefox, and Internet Explorer, and supports multiple programming languages, including Java, C#, Python, and Ruby.

Selenium WebDriver - 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. Get Command

**Method:**

1. get(String arg0) : **void**

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

Backward Skip 10sPlay VideoForward Skip 10s

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

1. driver.get(URL);
3. // Or can be written as
5. String URL = "URL";
6. driver.get(URL);

**Example:** For instance, the command to load the official website of javaTpoint can be written as:

1. driver.get("www.javatpoint.com")

2. Get Title Command

**Method:**

1. getTitle(): String

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:

1. driver.getTitle();
3. // Or can be written as
5. String Title = driver.getTitle();

3. Get Current URL Command

**Method:**

1. 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:

1. driver.getCurrentUrl();
3. //Or can be written as
5. String CurrentUrl = driver.getCurrentUrl();

4. Get Page Source Command

**Method:**

1. 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:

1. driver.getPageSource();
3. //Or can be written as
4. String PageSource = driver.getPageSource();

5. Close Command

**Method:**

1. 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*.

ADVERTISEMENT

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

1. driver.close();

6. Quit Command

**Method:**

1. 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.

ADVERTISEMENT

ADVERTISEMENT

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

1. driver.quit();

Let us consider a sample test script in which will cover most of the Browser Commands provided by WebDriver.

In this sample test, we will automate the following test scenarios:

* Invoke Chrome 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

For our test purpose, we are using the home page of "Google" search engine.