**WebDriver Browser Commands**

The very first question which comes to my mind and has been asked in many interviews is ***What is Selenium WebDriver***? Is it an Automation Tool? Is it a Class? Is it an Interface or what actually it is? To answer this question we need to understand the ***Advance Java OOPs concepts*** first and then we would be able to visualise the ***WebDriver Implementation***. For the sake of simplicity, we will avoid this WebDriver Implementation topic for now and will cover this in later chapters. As of now we start with all the methods we get from WebDriver.

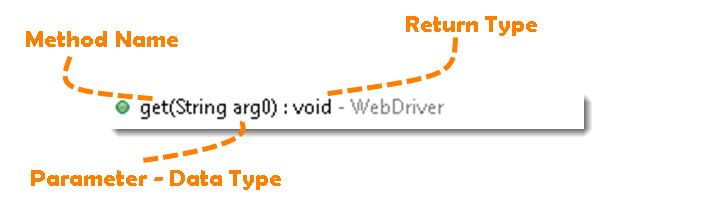
Now the next question is, How to access the methods of WebDriver? To check what all we have in WebDriver, create a driver object from WebDriver and press ***dot key.*** This will list down all the methods of WebDriver.



***Note:*** Methods followed by ***Object*** keyword are the generic methods gets from Object Class in Java. You will find these method for every object of java language.

* The suggestions marked in ***Blue Color*** are Nested Classes under WebDriver and will be covered in detail separately in the following chapters.
* The suggestions marked in ***Green Color*** are also Interfaces like WebDriver and will be covered in detail separately in the following chapters.
* The suggestions marked in ***Violet Color*** are similar methods like ***Orange*** but will be covered in detail separately in the following chapters.

Let’s just start discussing the ***Orange colored*** methods of ***Selenium WebDriver*** but before that try to understand the syntax of the suggestions display by Eclipse for WebDriver.



**Method:** A Java method is a collection of statements that are grouped together to perform an operation.

* ***Method Name:*** To access any method of any class, we need to create an object of class and then all the public methods will appear for the object.
* ***Parameter:*** It is an argument which is passed to a method as a parameter to perform some operation. Every argument must passed with the same data type. For e.g. ***get(String arg0) : void.*** This is asking for a ***String type*** argument.
* ***Return Type:*** Method can returns a value or returning nothing (void). If the ***void*** is mentioned after the method, it means the method is returning no value. And if it is returning any value, then it must display the type of the value for e.g. ***getTitle() : String***.

Now it would be very easy to understand the WebDriver commands in the below chapter. The very first thing you like to do with Selenium is to ***Opening*** a new browser, ***Perform*** few tasks and ***Closing*** the browser. Below are the numbers of commands you can apply on the Selenium opened browser.

## Get Command

***get(String arg0) : void*** – This method ***Load*** a new web page in the current browser window. Accepts String as a parameter and returns nothing.

**Command** – ***driver.get(appUrl);***

Where **appUrl** is the website address to load. It is best to use a fully qualified URL.



|  |  |
| --- | --- |
| 1  2  3  4  5  6 | driver.get("http://www.google.com");    //Or can be written as    String URL = "http://www.DemoQA.com";  driver.get(URL); |

## Get Title Command

***getTitle() : String*** – This method fetches the ***Title*** of the current page. Accepts nothing as a parameter and returns a String value.

***Command – driver.getTitle();***

As the return type is String value, the output must be stored in String object/variable.



|  |  |
| --- | --- |
| 1  2  3  4  5 | driver.getTitle();    //Or can be used as    String Title = driver.getTitle(); |

## Get Current URL Command

***getCurrentUrl() : String*** – This method fetches the string representing the ***Current URL*** which is opened in the browser. Accepts nothing as a parameter and returns a String value.

***Command – driver.getCurrentTitle();***

As the return type is String value, the output must be stored in String object/variable.



|  |  |
| --- | --- |
| 1  2  3  4  5 | driver.getCurrentUrl();    //Or can be written as    String CurrentUrl = driver.getCurrentUrl(); |

## Get Page Source Command

***getPageSource() : String*** – This method returns the ***Source Code*** of the page. Accepts nothing as a parameter and returns a String value.

***Command – driver.getPageSource();***

As the return type is String value, the output must be stored in String object/variable.



|  |  |
| --- | --- |
| 1  2  3  4 | driver.getPageSource();    //Or can be written as  String PageSource = driver.getPageSource(); |

## Close Command

***close() : void*** – This method **Close** only the current window the WebDriver is currently controlling. Accepts nothing as a parameter and returns nothing.

***Command – driver.close();***

Quit the browser if it’s the last window currently open.



|  |  |
| --- | --- |
| 1 | driver.close(); |

## Quit Command

***quit() : void*** – This method **Closes** all windows opened by the WebDriver. Accepts nothing as a parameter and returns nothing.

***Command – driver.quit();***

Close every associated window.



|  |  |
| --- | --- |
| 1 | driver.quit(); |

### Practice Exercise – 1

1. Launch a new Firefox browser.
2. Open Store.DemoQA.com
3. Get Page Title name and Title length
4. Print Page Title and Title length on the Eclipse Console.
5. Get Page URL and verify if the it is a correct page opened
6. Get Page Source (HTML Source code) and Page Source length
7. Print Page Length on Eclipse Console.
8. Close the Browser.

### Solution



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50 | package automationFramework;  import org.openqa.selenium.WebDriver;  import org.openqa.selenium.firefox.FirefoxDriver;  public class WebDriverCommands {    public static void main(String[] args) {  // Create a new instance of the FireFox driver  WebDriver driver = new FirefoxDriver();    // Storing the Application Url in the String variable  String url = "http://www.store.demoqa.com";    //Launch the ToolsQA WebSite  driver.get(url);    // Storing Title name in the String variable  String title = driver.getTitle();    // Storing Title length in the Int variable  int titleLength = driver.getTitle().length();    // Printing Title & Title length in the Console window  System.out.println("Title of the page is : " + title);  System.out.println("Length of the title is : "+ titleLength);    // Storing URL in String variable  String actualUrl = driver.getCurrentUrl();    if (actualUrl.equals(url)){  System.out.println("Verification Successful - The correct Url is opened.");  }else{  System.out.println("Verification Failed - An incorrect Url is opened.");  //In case of Fail, you like to print the actual and expected URL for the record purpose  System.out.println("Actual URL is : " + actualUrl);  System.out.println("Expected URL is : " + url);  }    // Storing Page Source in String variable  String pageSource = driver.getPageSource();    // Storing Page Source length in Int variable  int pageSourceLength = pageSource.length();    // Printing length of the Page Source on console  System.out.println("Total length of the Pgae Source is : " + pageSourceLength);    //Closing browser  driver.close();  }  } |

***Output***

Title of the page is : ONLINE STORE | Toolsqa Dummy Test site  
Length of the title is : 38  
Verification Failed – An incorrect Url is opened.  
Actual URL is : http://store.demoqa.com/  
Expected URL is : http://www.store.demoqa.com  
Total length of the Pgae Source is : 35646

### Practice Exercise – 2

1. Launch a new Firefox browser.
2. Open http://demoqa.com/frames-and-windows/
3. Use this statement to click on a New Window button “driver.findElement(By.xpath(“.//\*[@id=’tabs-1′]/div/p/a”)).click();”
4. Close the browser using close() command

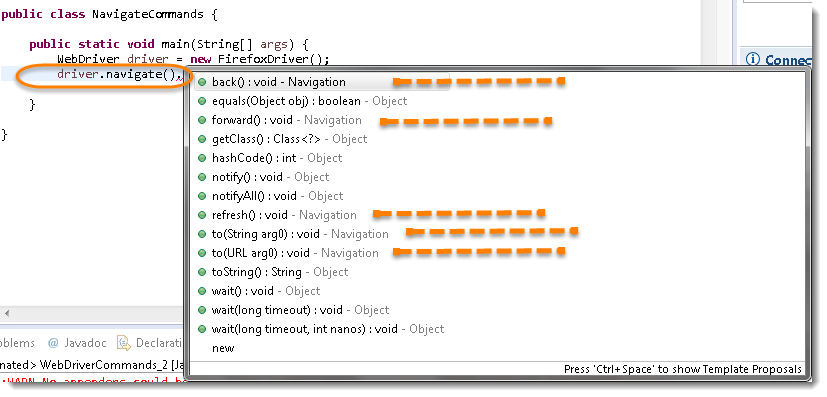
You will notice that only one window will close. Next time use quit() command instead of close(). At that time selenium will close both the windows.



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | package automationFramework;  import org.openqa.selenium.By;  import org.openqa.selenium.WebDriver;  import org.openqa.selenium.firefox.FirefoxDriver;    public class WebDriverCommands\_2 {  public static void main(String[] args) {  WebDriver driver = new FirefoxDriver();  driver.get("http://demoqa.com/frames-and-windows/");  driver.findElement(By.xpath(".//\*[@id='tabs-1']/div/p/a")).click();  driver.close();  }  } |

# Browser Navigation Commands

After successfully running our first test case on Firefox Browser now we are stepping towards grasping the essential ***Browser Navigation Commands*** in Selenium. Thus we are going to discuss about various navigation commands that we would be using in our day to day automation testing. The navigate interface exposes the ability to move backwards and forwards in the browser’s history.

To access the navigation’s method, just type ***driver.navigate().***. The intellisence feature of eclipse will automatically display all the public methods of **Navigate Interface** shown in the below image.  


***Note:*** Only methods which are followed by ***Navigation*** keyword are belongs to navigate. Rest followed by ***Object*** keyword are the generic methods gets from Object Class in Java. You will find these method for every object of java language.

## Navigate To Command

***to(String arg0) : void*** – This method ***Loads*** a new web page in the current browser window. It accepts a String parameter and returns nothing.

**Command** – ***driver.navigate().to(appUrl);***

It does exactly the same thing as the ***driver.get(appUrl)*** method. Where **appUrl** is the website address to load. It is best to use a fully qualified URL.



|  |  |
| --- | --- |
| 1 | driver.navigate().to("http://www.DemoQA.com"); |

## Forward Command

***forward() : void*** – This method does the same operation as clicking on the ***Forward Button*** of any browser. It neither accepts nor returns anything.

**Command** – ***driver.navigate().forward();***

Takes you forward by one page on the browser’s history.



|  |  |
| --- | --- |
| 1 | driver.navigate().forward(); |

## Back Command

***back() : void*** – This method does the same operation as clicking on the ***Back Button*** of any browser. It neither accepts nor returns anything.

**Command** – ***driver.navigate().back();***

Takes youback by one page on the browser’s history.



|  |  |
| --- | --- |
| 1 | driver.navigate().back(); |

## Refresh Command

***refresh() : void*** – This method ***Refresh*** the current page. It neither accepts nor returns anything.

**Command** – ***driver.navigate().refresh();***

Perform the same function as pressing F5 in the browser.



|  |  |
| --- | --- |
| 1 | driver.navigate().refresh(); |

### Practice Exercise

1. Launch new Browser
2. Open DemoQA.com website
3. Click on Registration link using “driver.findElement(By.*xpath*(“.//\*[@id=’menu-item-374′]/a”)).click();“
4. Come back to Home page (Use ‘Back’ command)
5. Again go back to Registration page (This time use ‘Forward’ command)
6. Again come back to Home page (This time use ‘To’ command)
7. Refresh the Browser (Use ‘Refresh’ command)
8. Close the Browser

### Solution



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32 | package automationFramework;  import org.openqa.selenium.By;  import org.openqa.selenium.WebDriver;  import org.openqa.selenium.firefox.FirefoxDriver;  public class NavigateCommands {  public static void main(String[] args) {  // Create a new instance of the FireFox driver  WebDriver driver = new FirefoxDriver();    // Open ToolsQA web site  String appUrl = "http://www.DemoQA.com";  driver.get(appUrl);    // Click on Registration link  driver.findElement(By.xpath(".//\*[@id='menu-item-374']/a")).click();    // Go back to Home Page  driver.navigate().back();    // Go forward to Registration page  driver.navigate().forward();    // Go back to Home page  driver.navigate().to(appUrl);    // Refresh browser  driver.navigate().refresh();    // Close browser  driver.close();  }  } |