**Q. What is cross browser testing?**

**Cross Browser Testing** is a type of functional test to check that your web application works as expected in different browsers.

## Q. Why do we need Cross Browser Testing?

## Web-based applications are totally different from Windows applications. A web application can be opened in any browser by the end user. For example, some people prefer to open [****https://twitter.com****](https://twitter.com/) in ****Firefox browser,**** while other's can be using ****Chrome browser**** or****IE****, Mozilla, safari, opera etc.

## Q.How to perform Cross Browser Testing?

## 

## 

## Only tell comments.

## Q. What is Selenium?

## 

* **SELENIUM** is a free (open-source) automated testing framework used to validate web applications across different browsers and platforms.
* You can use multiple programming languages like Java, C#, Python etc to create Selenium Test Scripts. Testing done using the Selenium tool is usually referred to as Selenium Testing.
* It is also a suite which fulfills a different testing needs of an organization.

1. Selenium can support different type of browsers for automation.
2. Selenium can be integrated with TestNG to perform Multi Browser Testing.
3. From parameters in testing.xml we can pass browser name, and in a test case, we can create Web Driver reference accordingly.

Q. What are the components/tools of selenium?

* Selenium Integrated Development Environment (IDE)
* Selenium Remote Control (RC)
* WebDriver
* Selenium Grid

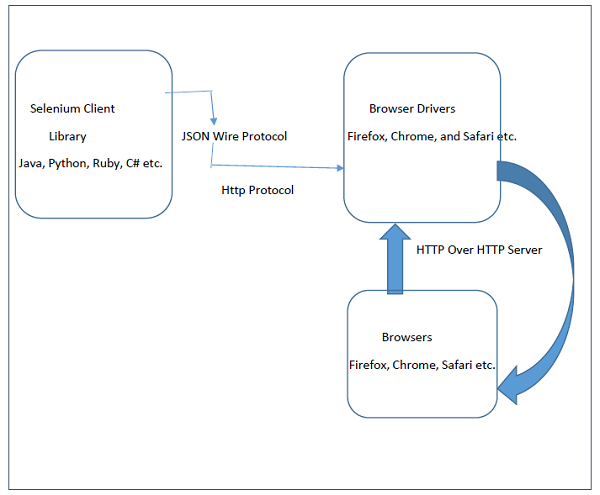
Q. What is Selenium Webdriver?

* **Selenium Webdriver** is an open-source collection of APIs which is used for testing web applications.
* Selenium Web driver tool is used for automating web application testing to verify that it works as expected or not.
* Web driver is an interface .It mainly supports browsers like Firefox, Chrome, Safari and Internet Explorer. It also permits you to execute cross-browser testing.
* It uses different drivers chromedriver, gecodriver, IE driver, safari driver

Following programming languages are supported by WebDriver

* [Java](https://www.guru99.com/java-tutorial.html)
* .Net
* [PHP](https://www.guru99.com/php-tutorials.html)
* [Python](https://www.guru99.com/python-tutorials.html)
* [Perl](https://www.guru99.com/perl-tutorials.html)
* Ruby

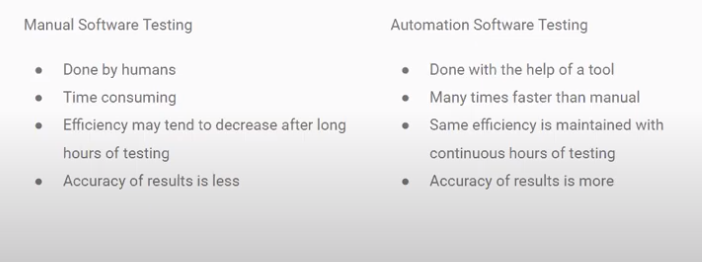
Q. What is the Selenium Web Driver Architecture?



Selenium WebDriver API enables interaction between browsers and browser drivers. This architecture consists of four layers namely the Selenium Client Library, JSON Wire Protocol, Browser Drivers and Browsers.

* Selenium Client Library consists of languages like Java, Ruby, Python, C# and so on. After the test cases are triggered, entire Selenium code will be converted to Json format.
* JSON stands for Javascript Object Notation. It takes up the task of transferring information from the server to the client. JSON Wire Protocol is primarily responsible for transfer of data between HTTP servers. Generated Json is made available to browser drivers through http Protocol.
* Each browser has a specific browser driver. Browser drivers interact with its respective browsers and execute the commands by interpreting Json which they received from the browser. As soon as the browser driver gets any instructions, they run them on the browser. Then the response is given back in the form of HTTP response.

Q. Difference bet Manual and Automation testing?



## Q. What are driver commands?

## WebDriver can be broadly classified in following categories:

Browser Commands

Navigation Commands

WebElement Commands

**1. Browser Commands**

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

### **1. Get Command**

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

driver.get(URL);

driver.get([www.facebook.com](http://www.facebook.com));

### **2.** **Get Title Command**

getTitle(): String

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

driver.getTitle();

### **3. Get Current URL Command**

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.

driver.getCurrentUrl();

### **4. Get Page Source Command**

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.

driver.getPageSource();

### **5. Close Command**

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 Command**

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.

# **2. Navigation Commands**

WebDriver provides some basic Browser Navigation Commands that allows the browser to move backwards or forwards in the browser's history.

### **1. Navigate To Command**

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/navigate a new web page can be written as:

driver.navigate().to("www.javatpoint.com");

It maintains the browser history and cookies

### **2. Forward Command**

In WebDriver, this method enables the web browser to click on the **forward** button in the existing browser window. It neither accepts anything nor returns anything.

driver.navigate().forward();

### **3. Back Command**

In WebDriver, this method enables the web browser to click on the **back** button in the existing browser window. It neither accepts anything nor returns anything.

driver.navigate().back();

### **4. Refresh Command**

In WebDriver, this method refresh/reloads the current web page in the existing browser window. It neither accepts anything nor returns anything.

driver.navigate().refresh();

# **3. WebElement Commands**

The term web element refers to a HTML element. The HTML documents are composed of HTML elements.

### **1. Clear Command**

1. element.clear();
2. //Or can be written as
4. driver.findElement(By.id("UserName")).clear();

### **2. Sendkeys Command**

1. element.sendKeys("text");

//Or can be written as

driver.findElement(By.id("UserName")).sendKeys("JavaTpoint");

### **3. Click Command**

element.click();

//Or can be written as

driver.findElement(By.linkText("javaTpoint")).click();

### **4. IsDisplayed Command**

**Method:**

isDisplayed() : **boolean**

**Command:**

element.isDisplayed();

//Or can be written as

**boolean** staus = driver.findElement(By.id("UserName")).isDisplayed();

### **5. IsEnabled Command**

**Method:**

isEnabled() : **boolean**

**Command:**

element.isEnabled();

//Or can be written as

**boolean** staus = driver.findElement(By.id("UserName")).isEnabled();

### **6. IsSelected Command**

**Method:**

isSelected() : **boolean**

**Command:**

element.isSelected();

//Or can be written as

**boolean** staus = driver.findElement(By.id("Sex-Male")).isSelected();

### **7. Submit Command**

**Method:**

submit() : **void**

**Command:**

element.submit();

//Or can be written as

driver.findElement(By.id("SubmitButton")).submit();

[**next →**](https://www.javatpoint.com/selenium-webdriver-running-test-on-chrome-browser)[**← prev**](https://www.javatpoint.com/selenium-webdriver-navigation-commands)

# **Selenium WebDriver - WebElement Commands**

Before proceeding with this section, first we should know the basic terminology related to web elements in WebDriver.

## What is Web Element?

The term web element refers to a HTML element. The HTML documents are composed of HTML elements. It consists **a start tag**, **an end tag** and the **content** in between. For instance, a HTML element is written as: "<tagname> content </tagname>"

In WebDriver, we have several commonly used web element commands and actions. The following screenshot displays the eclipse web element command panel.



#### **Note: To get the web element object, we have to write the statement as:**

1. WebElement element = driver.findElement(By.id("UserName"));

Here, the UserName is the value of the **id** attribute, used as a unique identification for the desired web element.

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

### **1. Clear Command**

**Method:**

1. clear() : **void**

**Command:**

1. element.clear();

**Code snippet:**

1. WebElement element = driver.findElement(By.id("UserName"));
2. element.clear();
4. //Or can be written as
6. driver.findElement(By.id("UserName")).clear();

### **2. Sendkeys Command**

**Method:**

1. sendKeys(CharSequence? KeysToSend) : **void**

**Command:**

1. element.sendKeys("text");

**Code snippet:**

1. WebElement element = driver.findElement(By.id("UserName"));
2. element.sendKeys("JavaTpoint");
4. //Or can be written as
6. driver.findElement(By.id("UserName")).sendKeys("JavaTpoint");

### **3. Click Command**

**Method:**

1. click() : **void**

**Command:**

1. element.click();

**Code snippet:**

1. WebElement element = driver.findElement(By.linkText("javaTpoint"));
2. element.click();
4. //Or can be written as
6. driver.findElement(By.linkText("javaTpoint")).click();

### **4. IsDisplayed Command**

**Method:**

1. isDisplayed() : **boolean**

**Command:**

1. element.isDisplayed();

**Code snippet:**

1. WebElement element = driver.findElement(By.id("UserName"));
2. **boolean** status = element.isDisplayed();
4. //Or can be written as
6. **boolean** staus = driver.findElement(By.id("UserName")).isDisplayed();

### **5. IsEnabled Command**

**Method:**

1. isEnabled() : **boolean**

**Command:**

1. element.isEnabled();

**Code snippet:**

1. WebElement element = driver.findElement(By.id("UserName"));
2. **boolean** status = element.isEnabled();
4. //Or can be written as
6. **boolean** staus = driver.findElement(By.id("UserName")).isEnabled();
8. //Or can be used as
9. WebElement element = driver.findElement(By.id("userName"));
10. **boolean** status = element.isEnabled();
11. // Check that if the Text field is enabled, if yes enter value
12. **if**(status){
13. element.sendKeys("javaTpoint");
14. }

### **6. IsSelected Command**

**Method:**

1. isSelected() : **boolean**

**Command:**

1. element.isSelected();

**Code snippet:**

1. WebElement element = driver.findElement(By.id("Sex-Male"));
2. **boolean** status = element.isSelected();
4. //Or can be written as
6. **boolean** staus = driver.findElement(By.id("Sex-Male")).isSelected();

### **7. Submit Command**

**Method:**

1. submit() : **void**

**Command:**

1. element.submit();

**Code snippet:**

1. WebElement element = driver.findElement(By.id("SubmitButton"));
2. element.submit();
4. //Or can be written as
6. driver.findElement(By.id("SubmitButton")).submit();

### **8. GetText Command**

**Method:**

getText() : String

**Command:**

element.getText();

### **9. GetTagName Command**

**Method:**

getTagName() : String

**Command:**

element.getTagName();

 //Or can be written as

 String tagName = driver.findElement(By.id("SubmitButton")).getTagName();

### **10. getCssValue Command**

**Method:**

1. getCssvalue() : String

**Command:**

1. element.getCssValue();

### **11. getAttribute Command**

**Method:**

1. getAttribute(String Name) : String

**Command:**

1. element.getAttribute();

**Code snippet:**

1. WebElement element = driver.findElement(By.id("SubmitButton"));
2. String attValue = element.getAttribute("id"); //This will return "SubmitButton"

### **12. getSize Command**

**Method:**

1. getSize() : Dimension

**Command:**

1. element.getSize();

**Code snippet:**

1. WebElement element = driver.findElement(By.id("SubmitButton"));
2. Dimension dimensions = element.getSize();
3. System.out.println("Height :" + dimensions.height + "Width : "+ dimensions.width);

### **13. getLocation Command**

**Method:**

1. getLocation() : Point

**Command:**

1. element.getLocation();

**Code snippet:**

1. WebElement element = driver.findElement(By.id("SubmitButton"));
2. Point point = element.getLocation();
3. System.out.println("X cordinate : " + point.x + "Y cordinate: " + point.y);

Q. What are selenium Features?

* Selenium is an open source and portable Web testing Framework.
* It can be considered as the leading cloud-based testing platform which helps testers to record their actions and export them as a reusable script with a simple-to-understand and easy-to-use interface.
* Selenium supports various operating systems, browsers and programming languages. Following is the list:
  + Programming Languages: C#, Java, Python, PHP, Ruby, Perl, and JavaScript
  + Operating Systems: Android, iOS, Windows, Linux, Mac, Solaris.
  + Browsers: Google Chrome, Mozilla Firefox, Internet Explorer, Edge, Opera, Safari, etc.
* It also supports parallel test execution which reduces time and increases the efficiency of tests.
* Selenium can be integrated with frameworks like Ant and Maven for source code compilation.
* Selenium can also be integrated with testing frameworks like TestNG for application testing and generating reports.
* Selenium requires fewer resources as compared to other automation test tools.

Q. Difference between close and quit?

| **Serial No.** | **close()** | **quit()** |
| --- | --- | --- |
| 1 | close() method shall close the browser which is in focus. | quit() method closes all the browsers. |
| 2 | close() method closes the active WebDriver instance. | quit() method closes all the active WebDriver instances. |

Q. Difference between get and Navigate to?

The differences between get () and navigate () methods are listed below.

| **sl.no.** | **get()** | **navigate()** |
| --- | --- | --- |
| 1 | It is responsible for loading the page and waits until the page has finished loading. | It is only responsible for redirecting the page and then returning immediately. |
| 2 | It cannot track the history of the browser. | It tracks the browser history and can perform back and forth in the browser. |

## Q. Difference between Find element Vs Find elements

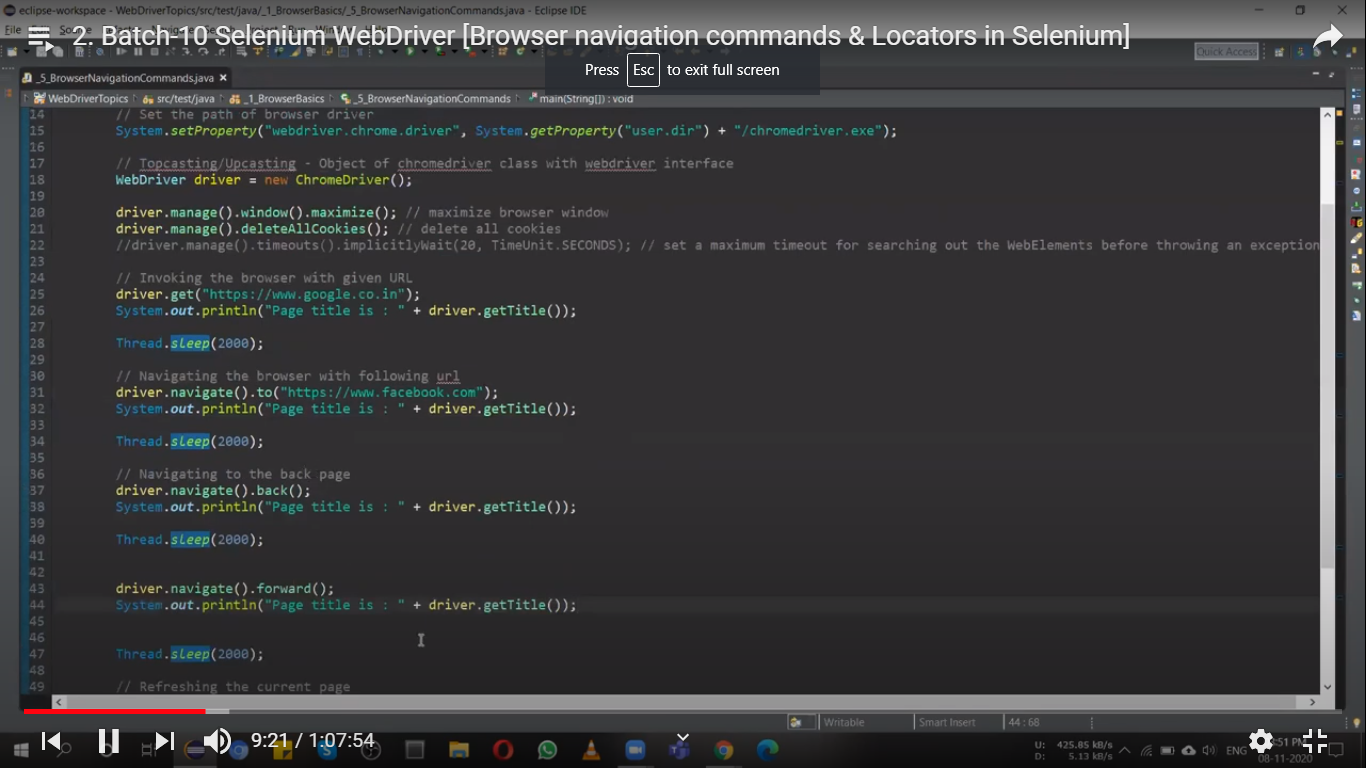
|  |  |
| --- | --- |
| **Find Element** | **Find Elements** |
| Returns the first most web element if there are multiple web elements found with the same locator | Returns a list of web elements |
| Throws exception NoSuchElementException if there are no elements matching the locator strategy | Returns an empty list if there are no web elements matching the locator strategy |
| It will only find one web element | It will find a collection of elements whose match the locator strategy. |
| Not Applicable | Each Web element is indexed with a number starting from 0 just like an array |

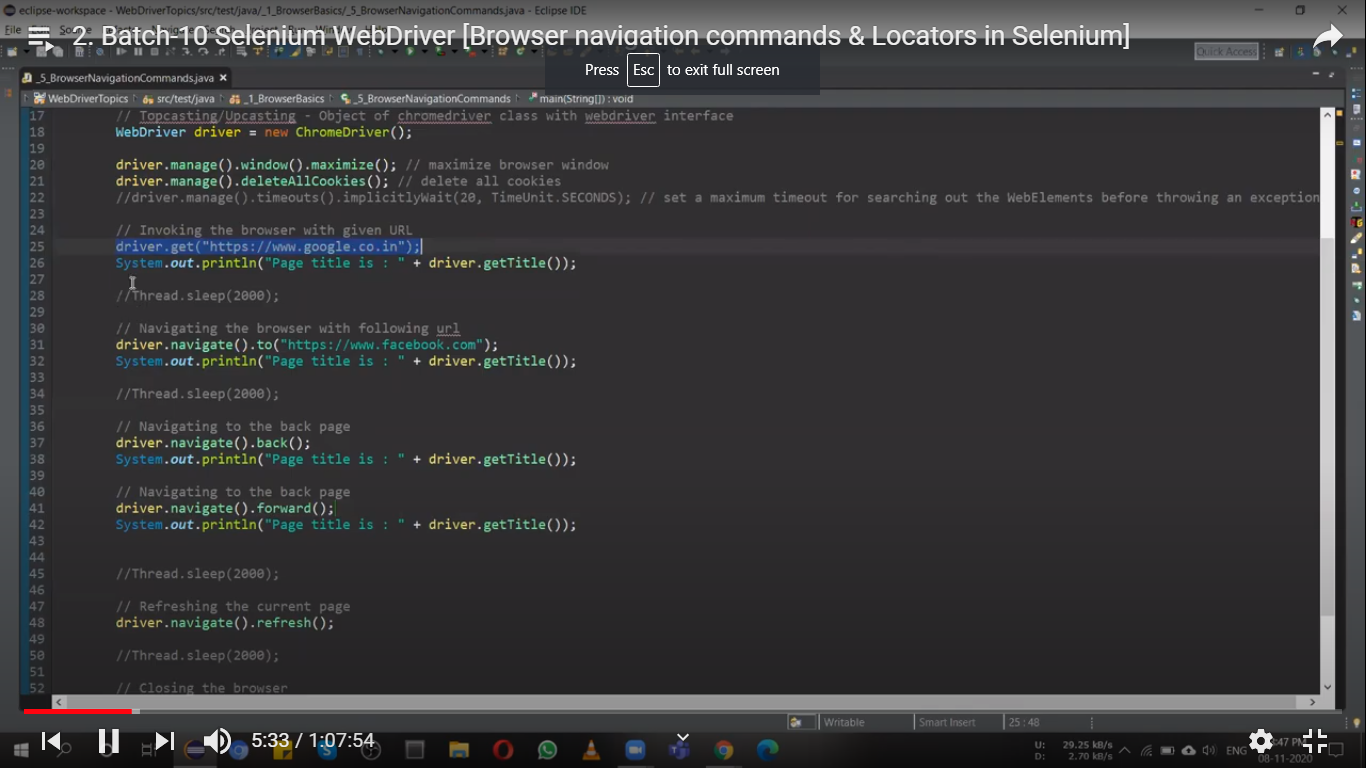
## Q. What are Locators?

## 

## 

Q . How you perform practical on Browser and Navigation commands?





Thread .sleep(2000): mechanism to pause your script for 2 sec, it gives warning and throws exception Interrupted exception.

Navigation commands: diver.navigate to