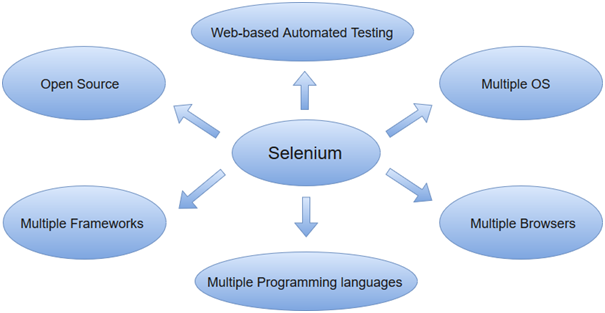
What is Selenium:

Selenium is one of the most widely used open source Web UI (User Interface) automation testing suite.It was originally developed by Jason Huggins in 2004 as an internal tool at Thought Works. Selenium supports automation across different browsers, platforms and programming languages.

Selenium can be easily deployed on platforms such as Windows, Linux, Solaris and Macintosh. Moreover, it supports OS (Operating System) for mobile applications like iOS, windows mobile and android.

Selenium supports a variety of programming languages through the use of drivers specific to each language.Languages supported by Selenium include C#, Java, Perl, PHP, Python and Ruby.Currently, Selenium Web driver is most popular with Java and C#. Selenium test scripts can be coded in any of the supported programming languages and can be run directly in most modern web browsers. Browsers supported by Selenium include Internet Explorer, Mozilla Firefox, Google Chrome and Safari.

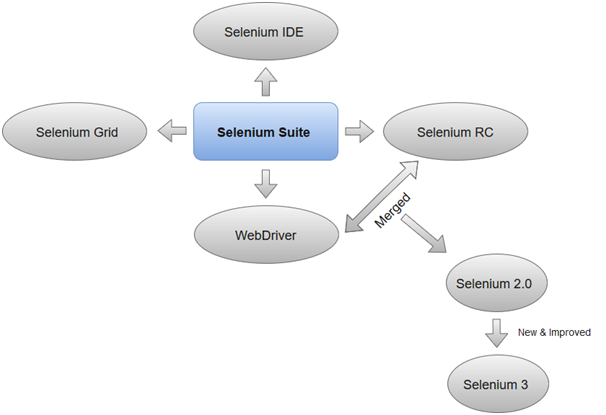


Selenium can be used to automate functional tests and can be integrated with automation test tools such as **Maven**, **Jenkins**, **& Docker** to achieve continuous testing. It can also be integrated with tools such as **TestNG**, & **JUnit** for managing test cases and generating reports.

# Selenium Tool Suite

Selenium is not just a single tool but a suite of software, each with a different approach to support automation testing. It comprises of four major components which include:

1. Selenium Integrated Development Environment (IDE)
2. WebDriver
3. Selenium Grid



### 1.Selenium Integrated Development Environment (IDE)

Selenium IDE is implemented as Firefox extension which provides record and playback functionality on test scripts. It allows testers to export recorded scripts in many languages like HTML, Java, Ruby, RSpec, Python, C#, JUnit and TestNG. You can use these exported script in Selenium RC or Webdriver.

>Selenium IDE has limited scope and the generated test scripts are not very robust and portable.

### 2. Selenium Remote Control

Selenium RC (officially deprecated by selenium)allows testers to write automated web application UI test in any of the supported programming languages. It also involves an HTTP proxy server which enables the browser to believe that the web application being tested comes from the domain provided by proxy server.

Play Videox[](https://campaign.adpushup.com/get-started/?utm_source=banner&utm_campaign=growth_hack)

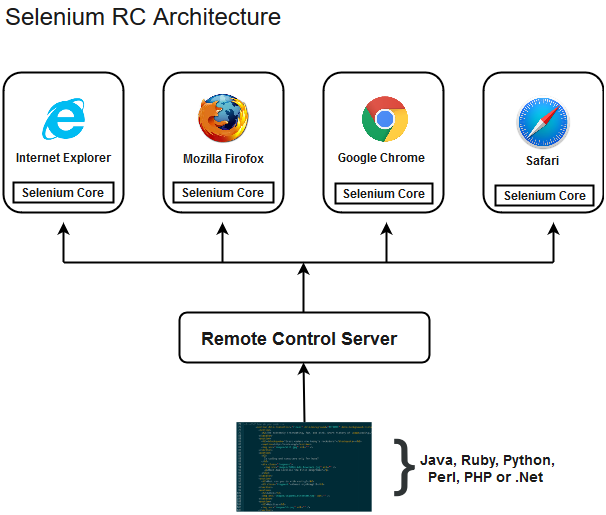
Selenium RC comes with two components.

1. Selenium RC Server (acts as a HTTP proxy for web requests).

>

1. Selenium RC Client (library containing your programming language code).

The figure given below shows the architectural representation of Selenium RC.



# Selenium Integrated Development Environment (IDE)

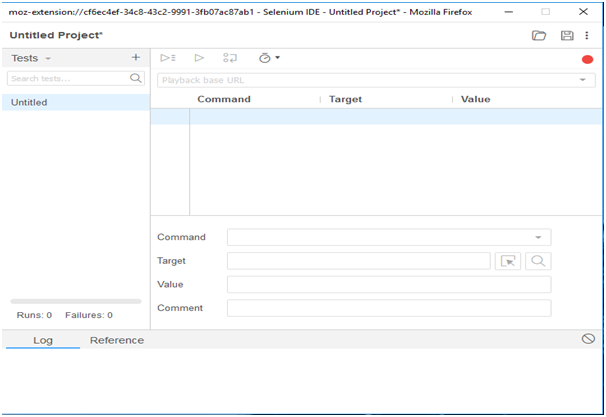
Selenium IDE (Integrated Development Environment) is an open source web automation testing tool under the Selenium Suite. Unlike Selenium WebDriver and RC, it does not require any programming logic to write its test scripts rather you can simply record your interactions with the browser to create test cases. Subsequently, you can use the playback option to re-run the test cases.

Perhaps, creating test cases on Selenium IDE does not require any programming language but when you get to use selenese commands like **runScript**, a little knowledge prior to JavaScript would prove beneficial for you to understand the concepts more clearly. You can also refer our JavaScript tutorial provided under the URL:

[https://www.javatpoint.com/javascript-tutorial](https://www.javatpoint.com/javascript-tutorial" \t "_blank)

#### Note: Selenium IDE is available only as Mozilla Firefox and Chrome plug-in, which means you can't record your test cases on browsers other than Firefox and Chrome. The recorded test scripts can also be exported to programming languages like C#, Java, Ruby or Python.

The following image shows the default interface of Selenium IDE:



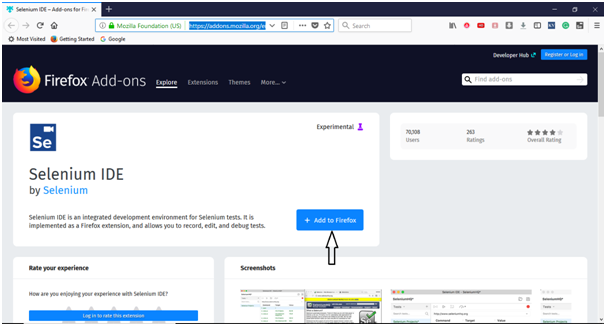
# Selenium IDE-Installation

Since, Selenium IDE is available only as Firefox and Chrome plug-in, we assume that you have already installed Mozilla Firefox browser in your system. However, you can download the latest version of Firefox through their official website provided under the link given below.

<https://www.mozilla.org/en-US/firefox/new/>

## Selenium IDE Download and Install

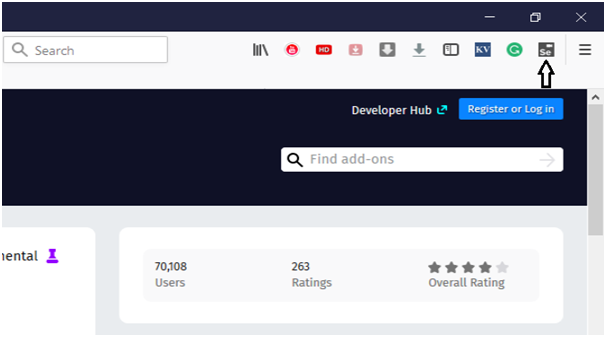
* Launch Mozilla Firefox browser.
* Open URL<https://addons.mozilla.org/en-us/firefox/addon/selenium-ide/>It will redirect you to the official add-on page of Firefox.
* Click on "Add to Firefox" button.



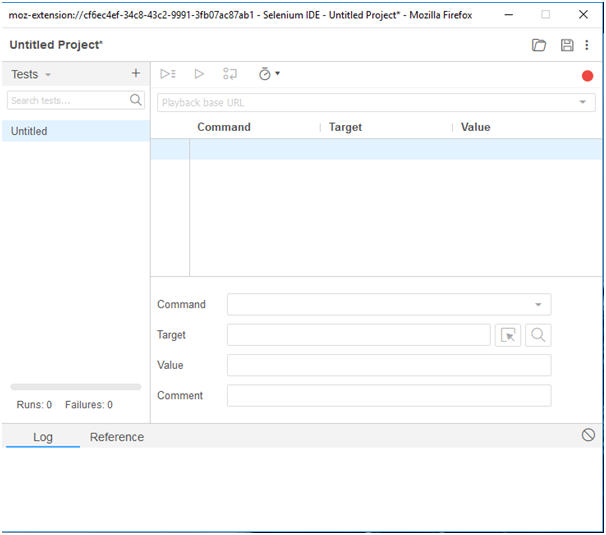
* A pop-up dialog box will be appeared asking you to add Selenium IDE as extension to your Firefox browser.
* Click on "Add" button.



* Restart you Firefox browser.
* Go to the top right corner on your Firefox browser and look for the Selenium IDE icon.



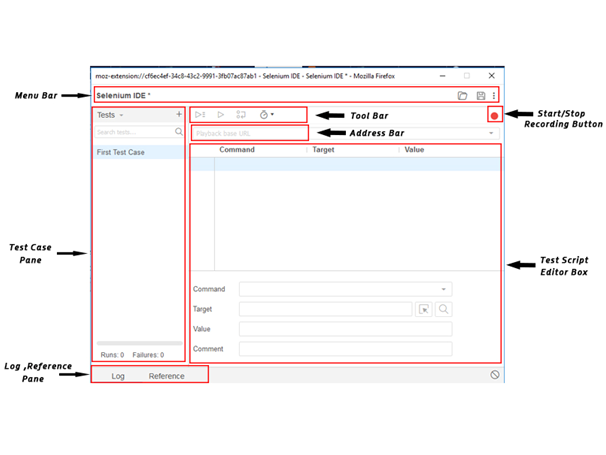
* Click on that icon to launch Selenium IDE.



# Selenium IDE-Features

Selenium IDE is divided into different components, each having their own features and functionalities.We have categorized seven different components of Selenium IDE, which includes:

1. Menu Bar
2. Tool Bar
3. Address Bar
4. Test Case Pane
5. Test Script Editor Box
6. Start/Stop Recording Button
7. Log, Reference Pane



#### Now, we will look at the features and functionalities of each component in detail.

### 1. Menu Bar

Menu bar is positioned at the top most portion of the Selenium IDE interface. The most commonly used modules of menu bar include:

* Project Name  
  It allows you to rename your entire project.

IDE-Features

* Open Project  
  It allows you to load any existing project from your personal drives.

IDE-Features

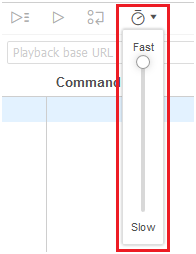
* Save Project  
  It allows you to save the entire project you are currently working on.

IDE-Features

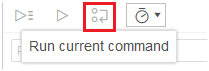
### 2. Tool Bar

The Tool bar contains modules for controlling the execution of your test cases. In addition, it gives you a step feature for debugging you test cases. The most commonly used modules of Tool Bar menu include:

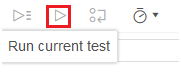
* Speed Control Option  
  It allows you to control the execution speed of your test cases.



* Step Feature  
  It allows you to "step" through a test case by running it one command at a time. Use for debugging test cases.



* Run Tests  
  It allows you to run the currently selected test. When only a single test is loaded "Run Test" button and "Run all" button have the same effect.



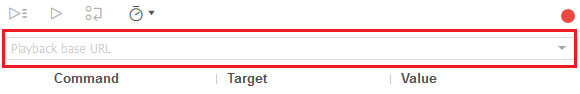
* Run All  
  It allows you to run the entire test suite when a test suite with multiple test cases is loaded.



### 3. Address Bar

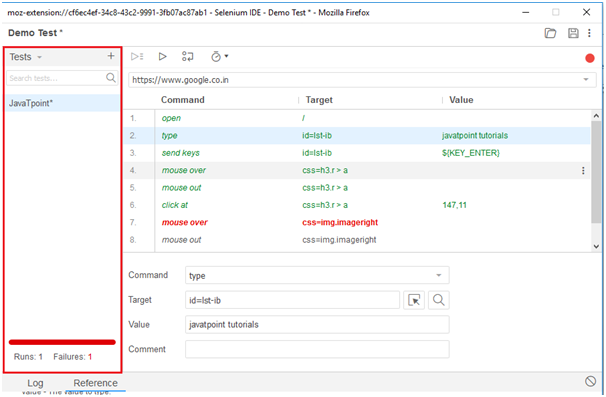
This module provides you a dropdown menu that remembers all previous values for base URL. In simple words, the base URL address bar remembers the previously visited websites so that the navigation becomes easy later on.

Play Videox[](https://campaign.adpushup.com/get-started/?utm_source=banner&utm_campaign=growth_hack)



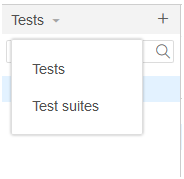
### 4. Test Case Pane

This module contains all the test cases that are recorded by IDE. In simple words, it provides the list of all recorded test cases at the same time under the test case pane so that user could easily shuffle between the test cases.



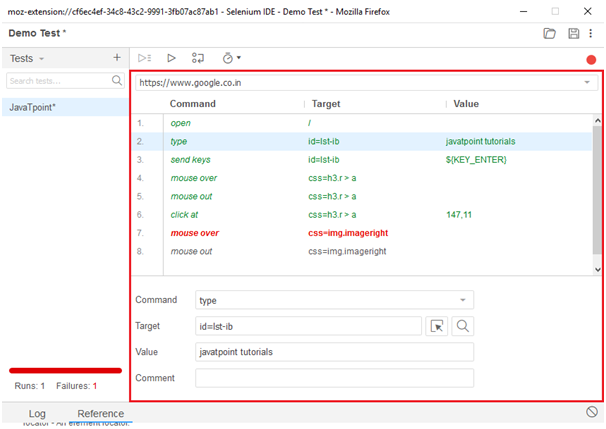
At the bottom portion of the Test Case Pane, you can see the test execution result summary which includes the pass/fail status of various test cases.

Test Case Pane also includes features like Navigation panel which allow users to navigate between test cases and test suites.

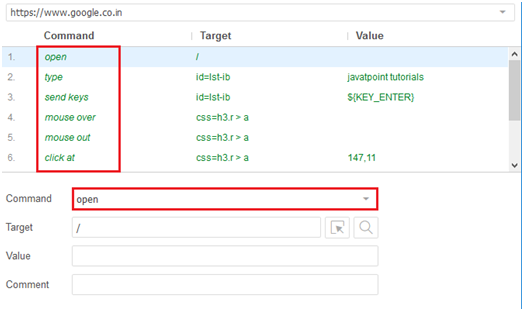


### 5. Test Script Editor Box

Test Script Editor Box displaysall of the test scripts and user interactions that were recorded by the IDE. Each user interaction is displayed in the same order in which they are performed. The Editor box is divided into three columns:Command, Target and Value.



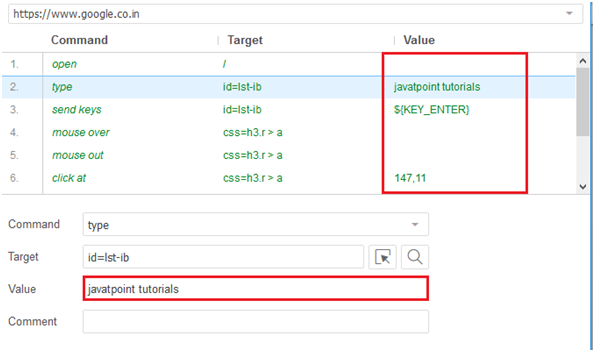
* Command:  
  Command can be considered as the actual operation/action that is performed on the browser elements. For instance, if you are opening a new URL, the command will be 'open'; if you are clicking on a link or a button on the web page, then the command will be 'clicked'.



* Target:  
  Target specifies the web element on which the operation has to be performed along with a locator attribute. For instance, if you are clicking on a button called javaTpoint, then the target link will be 'javaTpoint'.



* Value:  
  Value is treated as an optional field and can be used when we need to send some actual parameters. For instance, if you are entering the email address or password in a textbox, then the value will contain the actual credentials.



### 6. Start/Stop Recording Button

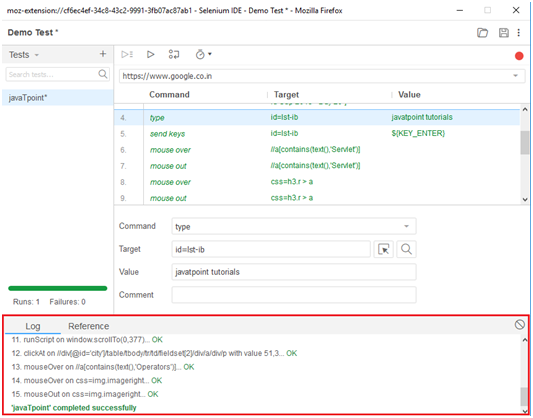
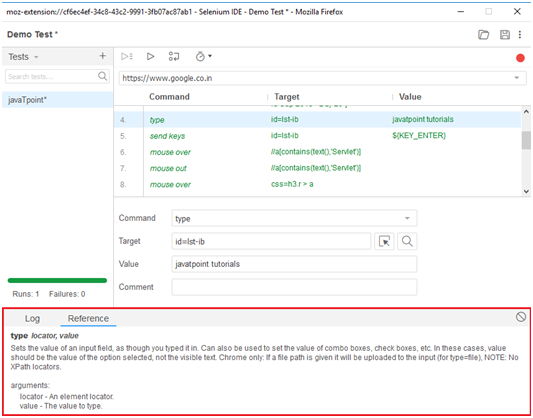
Record button records all of the user actions with the browser.



### 7. Log, Reference Pane

The Log Pane displays the runtime messages during execution. It provides real-time updates of the actions performed by the IDE. It can be categorized into four types: info, error, debug and warn.

The reference Pane displays the complete detail of the currently selected selenese command in the editor.

# Selenium IDE- First Test Case

In this section, you will learn how to create a basic test case in Selenium ide.

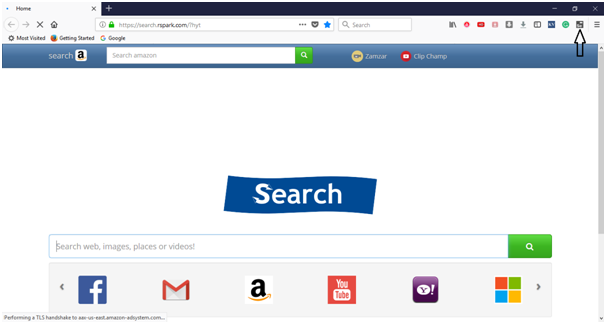
The entire test script creation process in Selenium IDE can be classified into three steps:

1. Recording (recording user interactions with the browser)
2. Playing back (executing the recorded script)
3. Saving the test suite

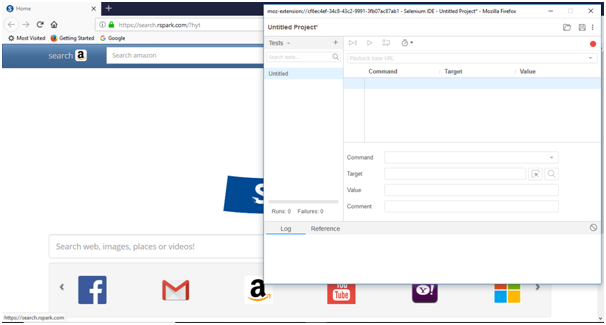
Now, we will see the implementation of the above three steps.

### 1. Recording

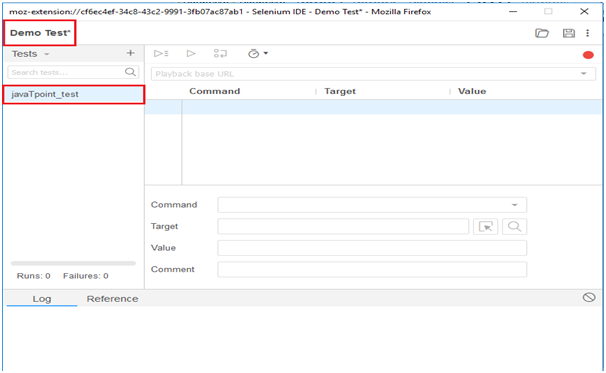
* Launch Firefox browser.
* Click on the Selenium icon present on the top right corner on your browser.



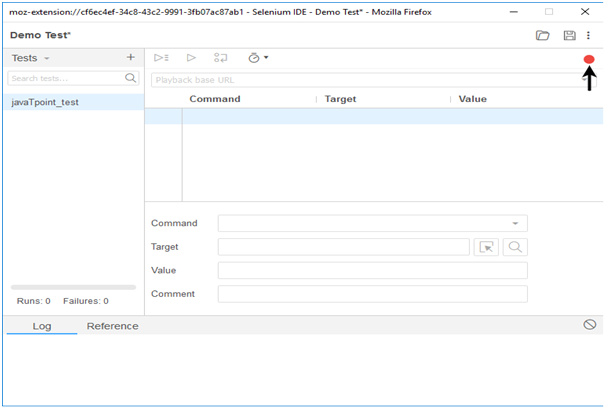
* It will launch the default interface of Selenium IDE.



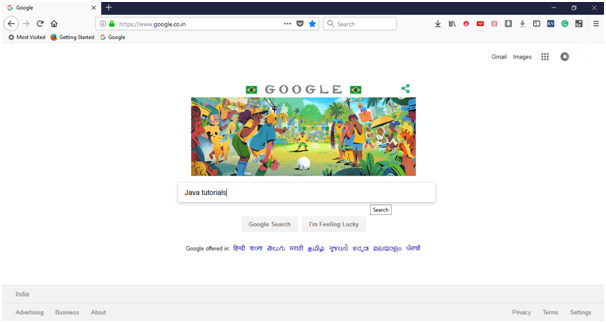
* Rename the project as "Demo Test".
* Rename the test case as "javaTpoint\_test".



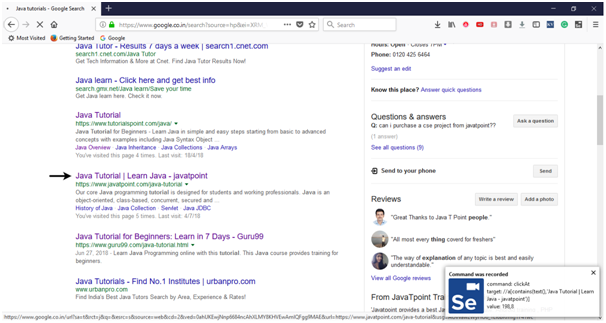
* Click on the "Start Recording" Button present on the top right corner on the IDE to start recording the test case.



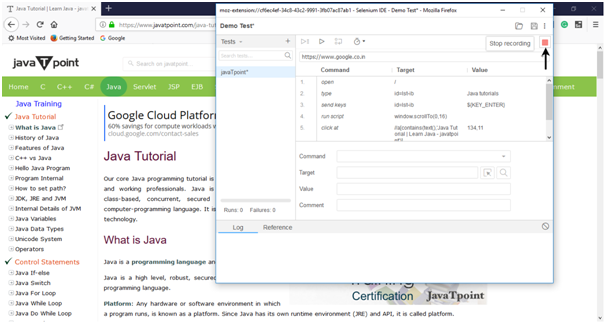
* Go to your Firefox browser and open URL:[www.google.com](https://www.javatpoint.com/www.google.com)
* It will redirect you to the Google search engine page.
* Type "Java Tutorials" in the Google search box.



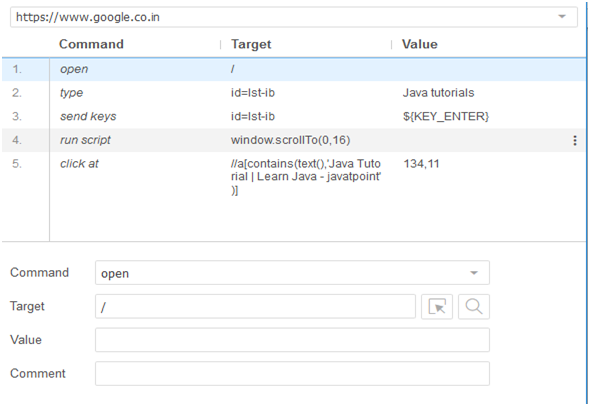
* Hit enter to get the search results.
* Click on the link "Java Tutorial" provided under the URL<https://www.javatpoint.com/java-tutorial>



* It will redirect you to the javaTpoint's Java tutorial web page. Meanwhile, you will get the notifications of the actions performed by the IDE at the extreme right corner of your web browser.
* Now, go the IDE and click on the "Stop Recording" button to stop recording your actions further.



* The Test Editor box now contains the list of all of your interactions with the browser.

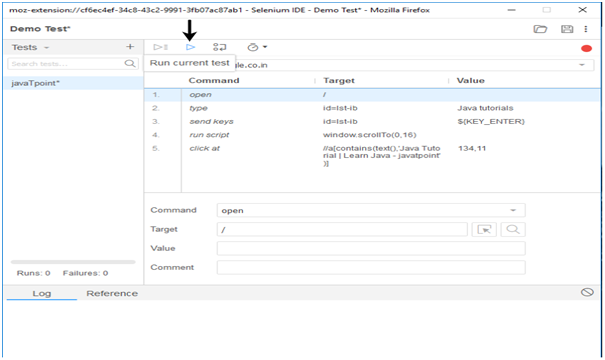


Now, we will proceed to the next step which includes executing the recorded script.

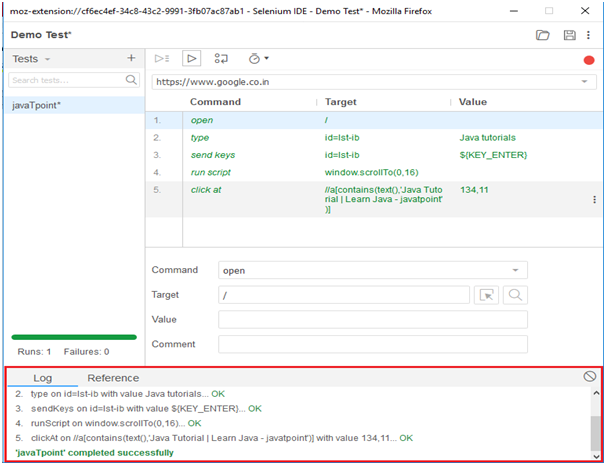
Play Videox

## 2. Playing Back

* Click on the "Run Current Test" button present on the tool bar menu of the IDE.  
  It will execute all of your interactions with the browser and gives you an overall summary of the executed test script.

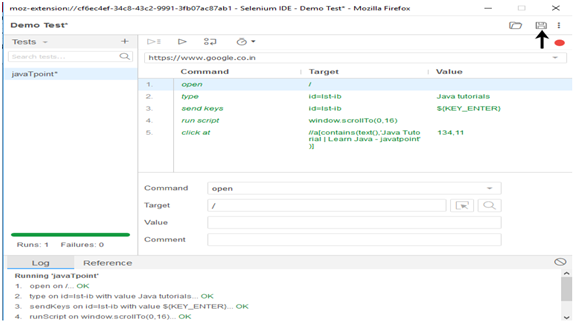


* The Log pane displays the overall summary of the executed test scripts.

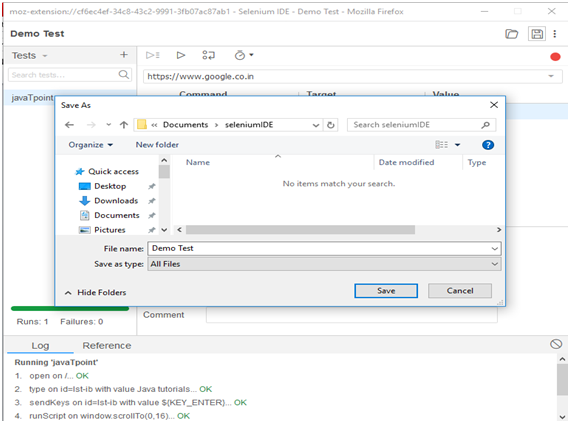


## 3. Saving the test suite

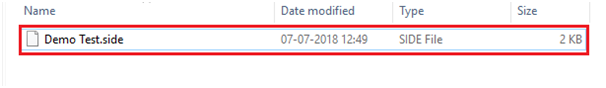
* Click on the save button present on the extreme right corner of the menu bar.



* Save the entire test suite as "Demo Test".



* The test suite can be found at the location provided in the above steps. Notice that the test script is saved in .side format.



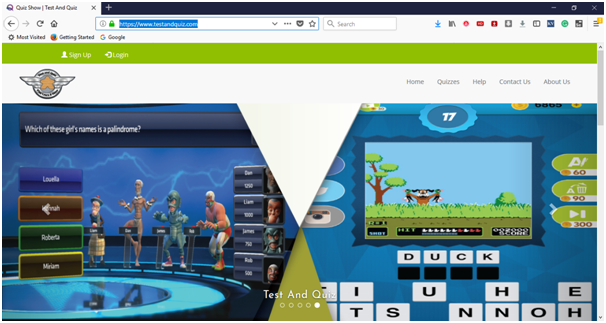
# Selenium IDE- Login Test

In this section, you will learn how to create a Login Test case in Selenium IDE.

For our test purpose, we will be testing the login page provided by the Test and Quiz website, present under the URL:[https://www.testandquiz.com/](https://www.testandquiz.com/" \t "_blank)

#### Note:you can sign-up and create a login test on any publically available website.

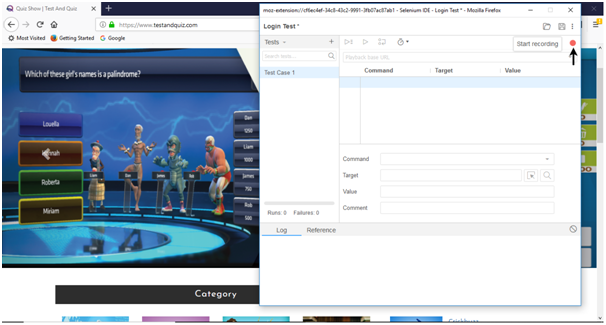
The following image shows the snapshot of the home page that will appear when we hit the above mentioned URL.



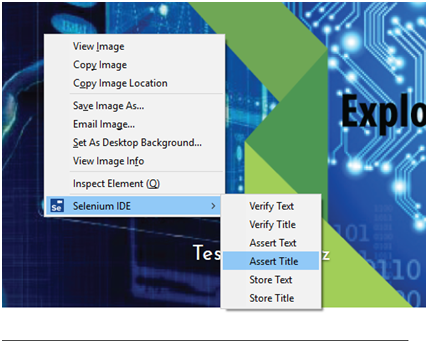
First you have sign-up to get the login credentials. For this test, we have already generated our login credentials. Now, we will generate a test script to create a Login Test in Selenium IDE.

Play Videox[](https://campaign.adpushup.com/get-started/?utm_source=banner&utm_campaign=growth_hack)

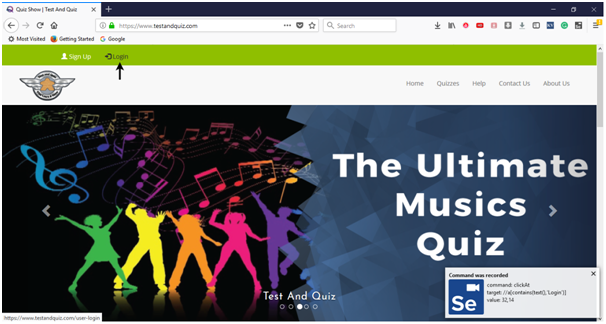
* Launch Firefox browser.
* Click on the Selenium icon present on the top right corner on your browser.
* It will launch the default interface of Selenium IDE.
* Go to your Firefox browser and open URL:<https://www.testandquiz.com/>
* Enter the project name as "Login Test".
* Enter the test case name as "Test Case 1".
* Click on the "Start Recording" button to start the recording of the test case.



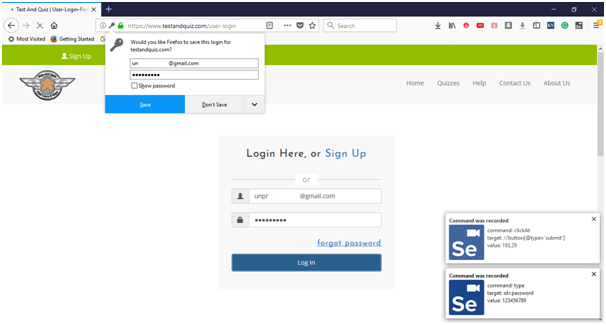
* Go to your Firefox browser and right click on any blank space within the page and select the Selenium IDE option.
* Click on Selenium IDE > Assert Title. The Assert Title command makes sure that the page title is correct.



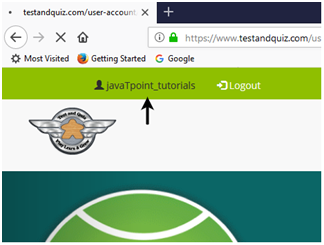
* Now, click on the login button located at the top corner of the website.



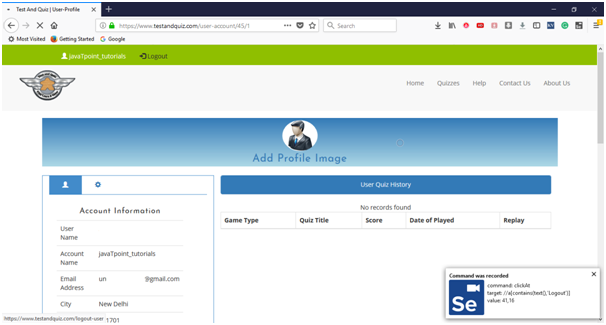
* Fill-in the login credentials and click on the login button. Meanwhile you will get the notifications of the actions performed by IDE at the extreme right corner of your browser.



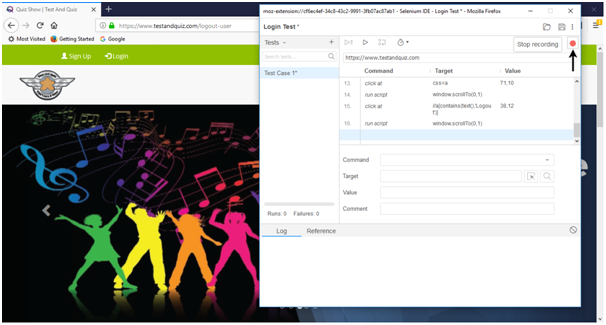
* Once you get logged-in, click on the user name section to view your account details.



* It will redirect you to your account settings page, where you can edit your personal details.

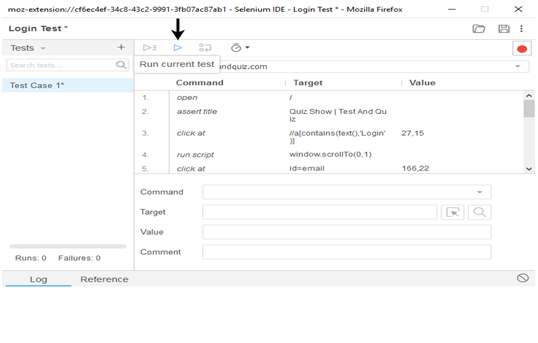


* Click on the Logout button.
* Now, go to the IDE and click on Stop Recording button to stop recording the test case.

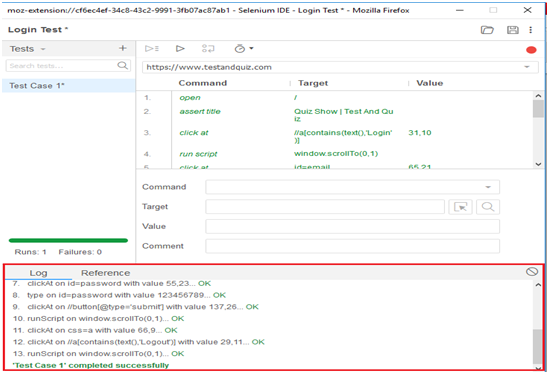


## 2. Playing Back

* Click on the "Run Current Test" button present on the tool bar menu of the IDE. It will execute all of your interactions with the browser and gives you an overall summary of the executed test script.

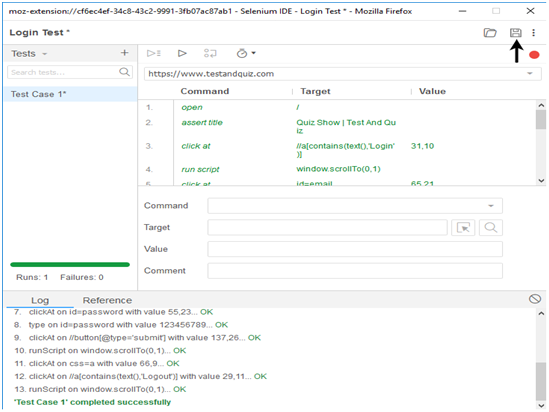


* The Log pane displays the overall summary of the executed test scripts.

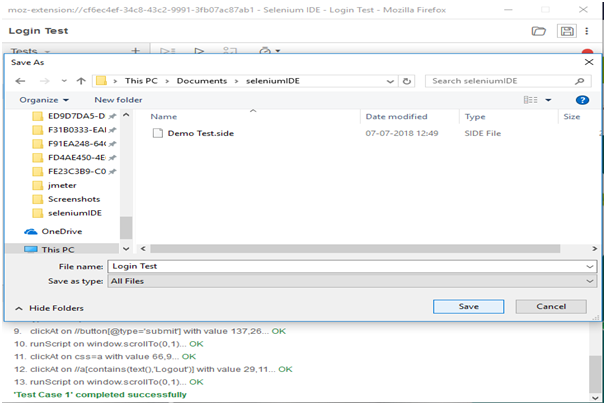


## 3. Saving the test suite

* Click on the save button present on the extreme right corner of the menu bar.



* Save the entire test suite as "Login Test".

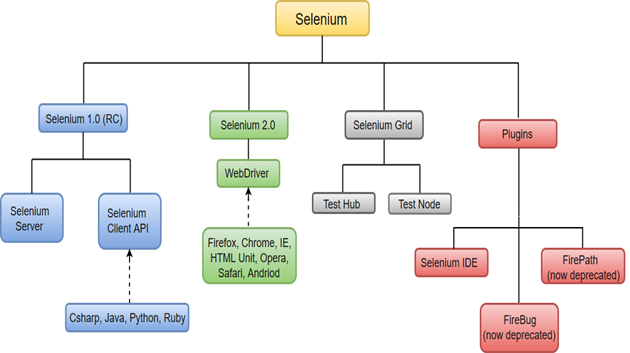


* The test suite can be found at the location provided in the above steps.

# Selenium WebDriver

Selenium WebDriver is the most important component of Selenium Tool's Suite. The latest release "Selenium 2.0" is integrated with WebDriver API which provides a simpler and more concise programming interface.

The following image will give you a fair understanding of Selenium components and the Test Automation Tools.



Selenium WebDriver was first introduced as a part of Selenium v2.0. The initial version of Selenium i.e Selenium v1 consisted of only IDE, RC and Grid. However, with the release of Selenium v3, RC has been deprecated and moved to legacy package.

In WebDriver, test scripts can be developed using any of the supported programming languages and can be run directly in most modern web browsers. Languages supported by WebDriver include C#, Java, Perl, PHP, Python and Ruby.

Play Videox[](https://campaign.adpushup.com/get-started/?utm_source=banner&utm_campaign=growth_hack)

Before learning the concepts of Selenium WebDriver, you should be well versed with any of the supported programming languages. Currently, Selenium Web driver is most popular with Java and C#. For this tutorial, we are using Selenium with java. You can refer to the links given below to learn basic as well as advance concepts of Java and C#:

Java Tutorial: <https://www.javatpoint.com/java-tutorial>

C# Tutorial: <https://www.javatpoint.com/c-sharp-tutorial>

Selenium WebDriver performs much faster as compared to Selenium RC because it makes direct calls to the web browsers. RC on the other hand needs an RC server to interact with the browser.

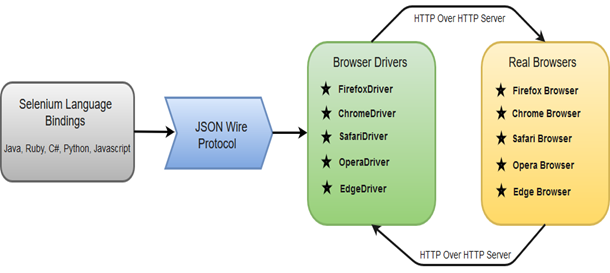
WebDriver has a built-in implementation of Firefox driver (Gecko Driver). For other browsers, you need to plug-in their browser specific drivers to communicate and run the test. Most commonly used WebDriver's include:

* Google Chrome Driver
* Internet Explorer Driver
* Opera Driver
* Safari Driver
* HTML Unit Driver (a special headless driver)

## Selenium WebDriver- Architecture

Selenium WebDriver API provides communication facility between languages and browsers.

The following image shows the architectural representation of Selenium WebDriver.



There are four basic components of WebDriver Architecture:

* Selenium Language Bindings
* JSON Wire Protocol
* Browser Drivers
* Real Browsers

### Selenium Language Bindings / Selenium Client Libraries

Selenium developers have built language bindings/Selenium Client Libraries in order to support multiple languages. For instance, if you want to use the browser driver in java, use the java bindings. All the supported language bindings can be downloaded from the official website [(https://www.seleniumhq.org/download/#client-drivers)](https://www.seleniumhq.org/download/#client-drivers) of Selenium.

### JSON Wire Protocol

JSON (JavaScript Object Notation) is an open standard for exchanging data on web. It supports data structures like object and array. So, it is easy to write and read data from JSON. To learn more about JSON, visit <https://www.javatpoint.com/json-tutorial>

JSON Wire Protocol provides a transport mechanism to transfer data between a server and a client. JSON Wire Protocol serves as an industry standard for various REST web services. To learn more about Web Services, visit <https://www.javatpoint.com/web-services-tutorial>

### Browser Drivers

Selenium uses drivers, specific to each browser in order to establish a secure connection with the browser without revealing the internal logic of browser's functionality. The browser driver is also specific to the language used for automation such as Java, C#, etc.

When we execute a test script using WebDriver, the following operations are performed internally.

* HTTP request is generated and sent to the browser driver for each Selenium command.
* The driver receives the HTTP request through HTTP server.
* HTTP Server decides all the steps to perform instructions which are executed on browser.
* Execution status is sent back to HTTP Server which is subsequently sent back to automation script.

### Browsers

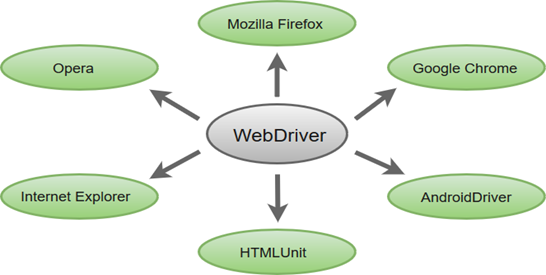
Browsers supported by Selenium WebDriver:

* Internet Explorer
* Mozilla Firefox
* Google Chrome
* Safari

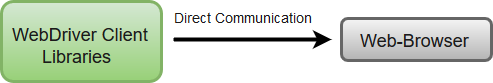
## Selenium WebDriver- Features

Some of the most important features of Selenium WebDriver are:

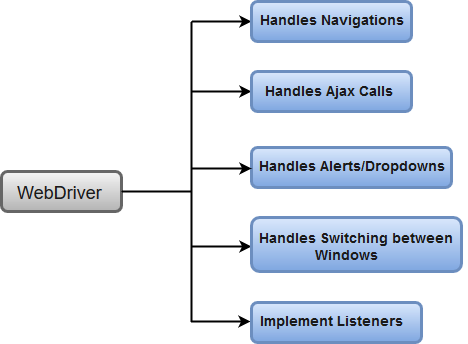
* **Multiple Browser Support**: Selenium WebDriver supports a diverse range of web browsers such as Firefox, Chrome, Internet Explorer, Opera and many more. It also supports some of the non-conventional or rare browsers like HTMLUnit.



* **Multiple Languages Support**: WebDriver also supports most of the commonly used programming languages like Java, C#, JavaScript, PHP, Ruby, Pearl and Python. Thus, the user can choose any one of the supported programming language based on his/her competency and start building the test scripts.
* **Speed**: WebDriver performs faster as compared to other tools of Selenium Suite. Unlike RC, it doesn't require any intermediate server to communicate with the browser; rather the tool directly communicates with the browser.



* **Simple Commands**: Most of the commands used in Selenium WebDriver are easy to implement. For instance, to launch a browser in WebDriver following commands are used:  
  **WebDriver driver** = **new FirefoxDriver();** (Firefox browser )  
  **WebDriver driver** = **new ChromeDriver();** (Chrome browser)  
  **WebDriver driver** = **new InternetExplorerDriver();** (Internet Explorer browser)
* **WebDriver- Methods and Classes**: WebDriver provides multiple solutions to cope with some potential challenges in automation testing.  
  WebDriver also allows testers to deal with complex types of web elements such as checkboxes, dropdowns and alerts through dynamic finders.



# Selenium WebDriver- Installation

Selenium WebDriver installation process is completed in four basic steps:

1. Download and Install Java 8 or higher version.
2. Download and configure Eclipse or any Java IDE of your choice.
3. Download Selenium WebDriver Java Client
4. Configure Selenium WebDriver

## 1. Download and Install Java

We assume that you have already installed Java 8 or above on your machine and successfully configured the environment variables required to run and compile java programs.

#### Note: you'll need to have Java 8 installed to use Selenium 3.

However, you can download the latest version of Java Development Kit (JDK) from the link given below.

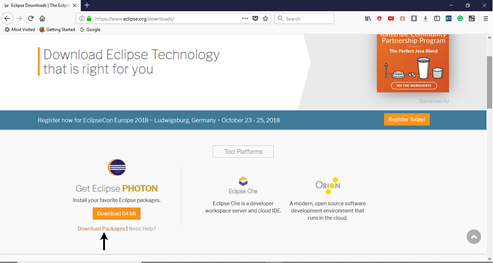
<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

Once you have downloaded and installed the latest version of Java, you need to set path or configure the environment variables in your system. Refer the link given below to understand how we can set path and configure environment variables in Java.

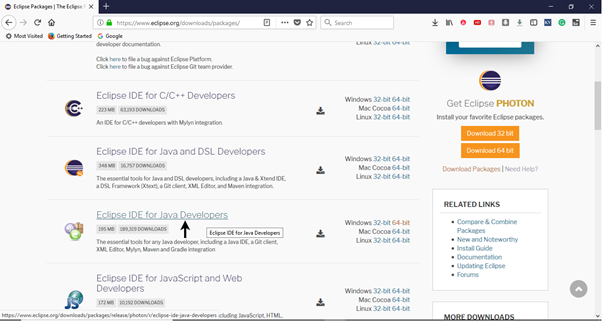
<https://www.javatpoint.com/how-to-set-path-in-java>

## 2. Download and Configure Eclipse IDE

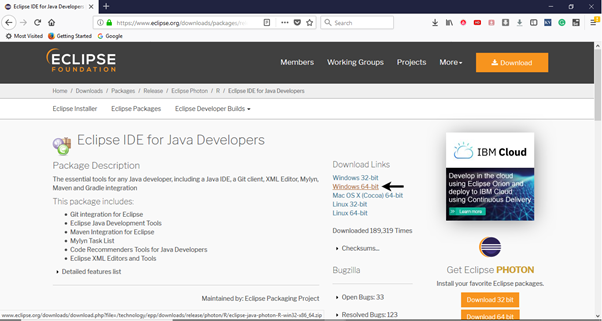
* Open URL: [https://www.eclipse.org/downloads/ .](https://www.eclipse.org/downloads/)
* Click on the "Download Packages" link (you can also download the IDE directly from the "downloads page" of Eclipse official website, but we will recommend you to navigate through the download packages section and get "Eclipse IDE for Java Developers").



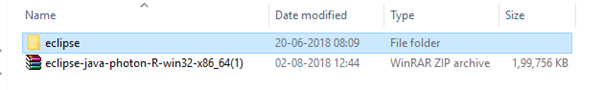
* It will redirect you to the "Download Packages" section. Scroll down through the webpage and click on "Eclipse IDE for Java Developers".



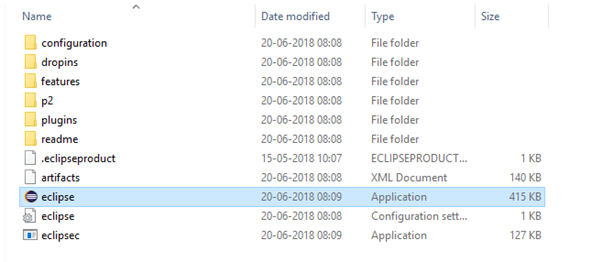
* Go to the Download Links section and click on "Windows 64-bit". You can also select other options to download based on the operating system you are currently working on.



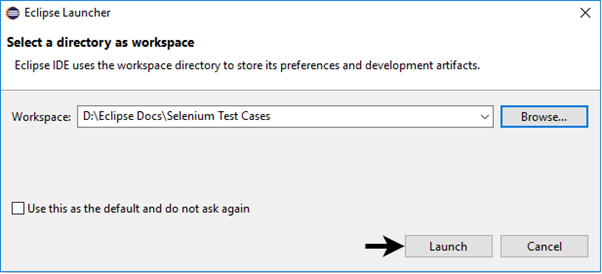
* The downloaded file would be in zipped format. Unpack the contents in a convenient directory.



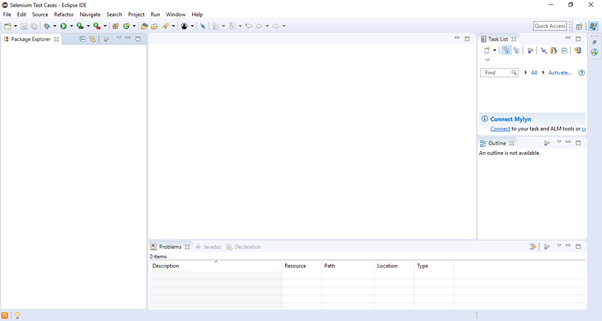
* Double click on "eclipse" (.exe file).



* To configure the workspace, select a convenient directory where you want to keep all of your Selenium trails and click on Launch button.

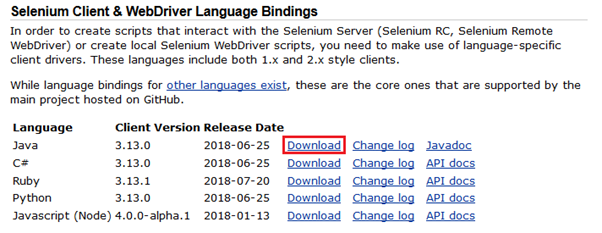


* It will launch the default interface of Eclipse IDE.

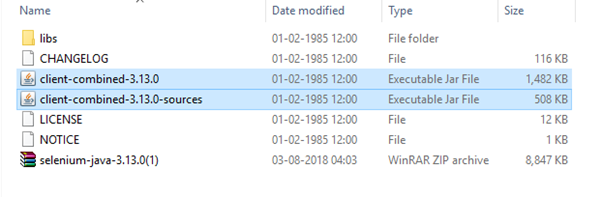


## 3. Download Selenium WebDriver Java Client

* Open URL: <https://docs.seleniumhq.org/download/>  
  It will redirect you to the "downloads page" of Selenium official website.
* Scroll down through the web page and locate **Selenium Client & WebDriver Language Bindings**.
* Click on the "Download" link of Java Client Driver as shown in the image given below.



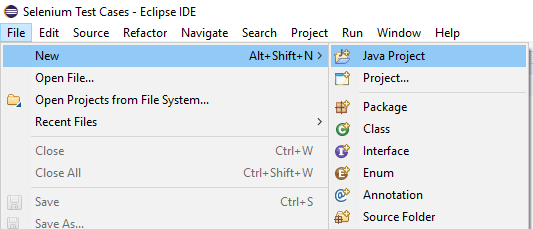
* The downloaded file would be in zipped format. Unpack the contents in a convenient directory. It contains the essential jar files required to configure Selenium WebDriver in Eclipse IDE.



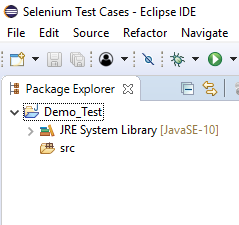
## 4. Configure Selenium WebDriver

Now we will configure our Eclipse IDE with Selenium WebDriver. In simple terms, we will create a new Java Project in Eclipse and load all the essential jar files in order to create Selenium Test Scripts.

* Launch Eclipse IDE
* Create a new Java Project from **File > New > Java Project**.

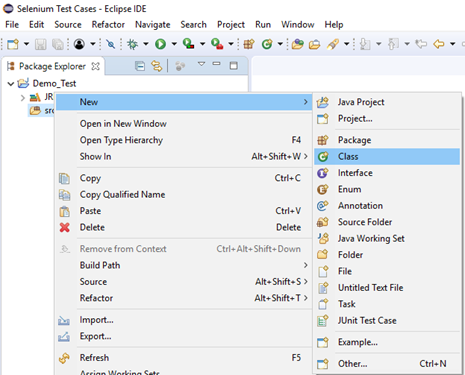


* Give your Project name as "Demo\_Test", leave the other fields unaltered and click on "Finish" button.
* It will create a new Java project with the following directories.

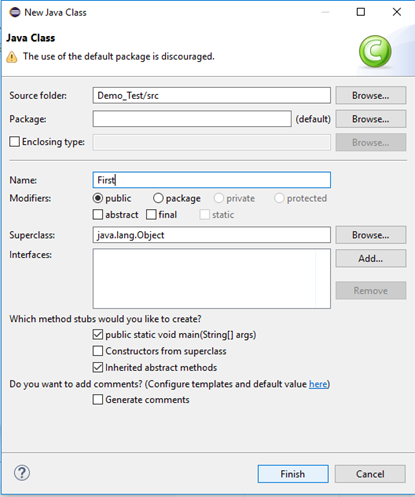


#### Note: Selenium Test Scripts are always written in ".class" file in Java. Here the project "Demo\_Test" act as a Test Suite that may contain one or more Selenium test cases/test scripts.

* Right click on the "src" folder and create a new Class File from **New > Class**.

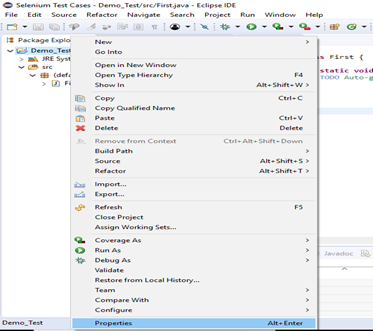


* Give your Class name as "First" and click on "Finish" button.

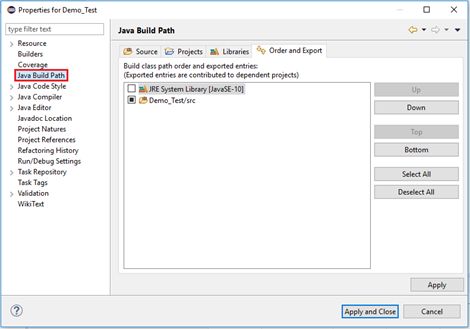


Now, we will add the Selenium jar files in our Test Suite (Demo\_Test).

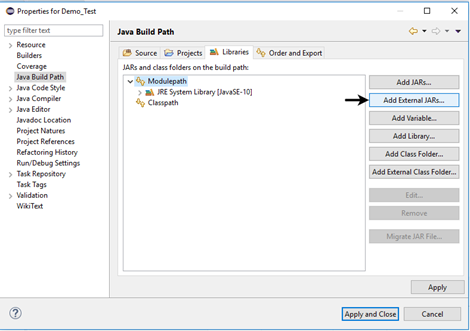
* Right click on "Demo\_Test" folder and select Properties.



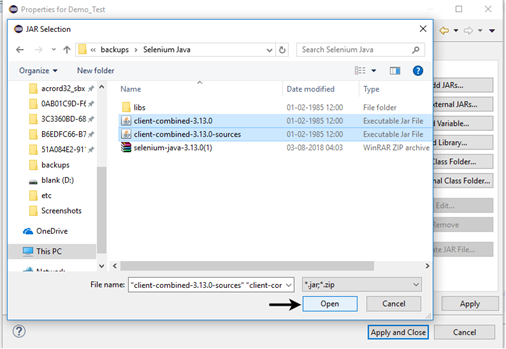
* It will launch the Properties window for our "Demo\_Test" Test Suite.
* Click on "Java Build Path" option from the left hand side panel.



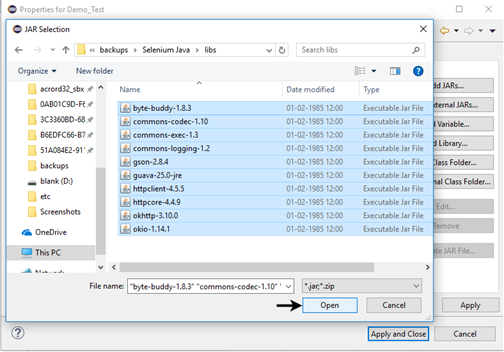
* Switch to Libraries tab and click on "Add External JARs" button.



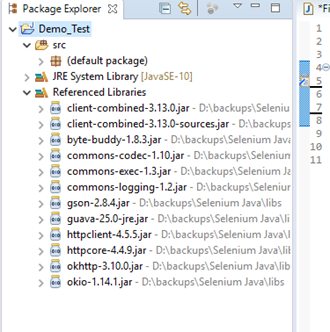
* Locate the directory where you have downloaded the Selenium jar files, select the respective jars and click on "Open" button.



* Repeat the same steps for the jars which are present under the "libs" folder.
* Open "libs" folder, select all of the respective jar files and click on "Open" button.



* Once you get all the Selenium jar files in your Libraries tab, click on Apply and Close button.
* The following image shows the directory structure of our "Demo\_Test" test suite after adding Selenium jars.



Hence, we have successfully configured Selenium WebDriver with Eclipse IDE. Now, we are ready to write our test scripts in Eclipse and run it in WebDriver.

# Selenium WebDriver- First Test Case

In this section, you will learn how to create your First Selenium Automation Test Script.

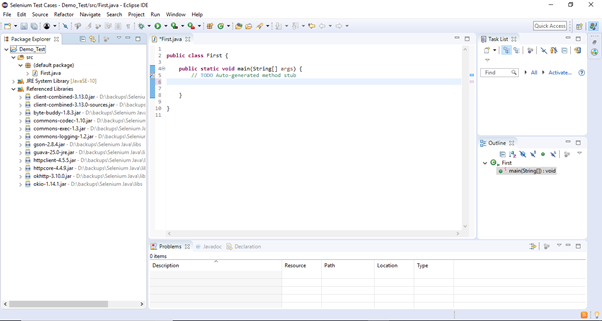
Under this test, we will automate the following scenarios:

* Invoke Google Chrome browser.
* Open URL: [www.google.com](https://www.google.com/" \t "_blank)
* Click on the Google Search text box.
* Type the value "javatpoint tutorials"
* Click on the Search button.

We will create our test case step by step to give you a complete understanding of each component in detail.

**Step1**. Launch Eclipse IDE and open project "Demo\_Test" which we have created in the previous section (Configure Selenium WebDriver) of this Tutorial. We will write our first Selenium test script in the "First.class" file under the "Demo\_Test" test suite.

Play Videox[[](https://campaign.adpushup.com/get-started/?utm_source=banner&utm_campaign=growth_hack)](https://campaign.adpushup.com/get-started/?utm_source=banner&utm_campaign=growth_hack" \t "_blank)

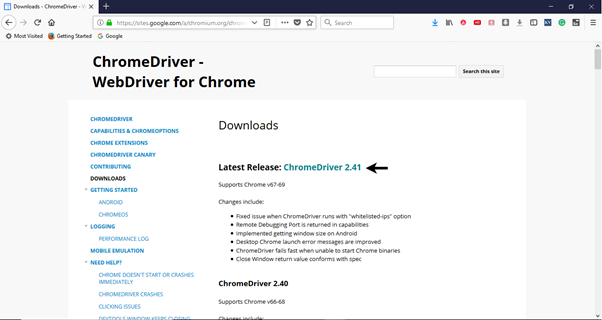


#### Note: To invoke a browser in Selenium, we have to download an executable file specific to that browser. For example, Chrome browser implements the WebDriver protocol using an executable called ChromeDriver.exe. These executable files start a server on your system which in turn is responsible for running your test scripts in Selenium.

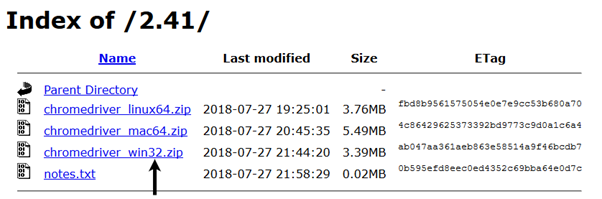
Step2. Open URL: [https://sites.google.com/a/chromium.org/chromedriver/downloads](https://sites.google.com/a/chromium.org/chromedriver/downloads" \t "_blank)

in your browser.

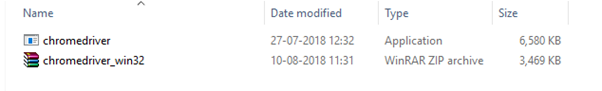
**Step3.** Click on the "ChromeDriver 2.41" link. It will redirect you to the directory of ChromeDriver executables files. Download as per the operating system you are currently on.



For windows, click on the "chromedriver\_win32.zip" download.



The downloaded file would be in zipped format. Unpack the contents in a convenient directory.



#### Note: Selenium developers have defined properties for each browser that needs the location of the respective executable files to be parsed in order to invoke a browser. For example, the property defined for Chrome browser - webdriver.chrome.driver, needs the path of its executable file - D:\ChromeDriver\chromedriver.exe in order to launch chrome browser.

Selenium WebDriver First Test Case

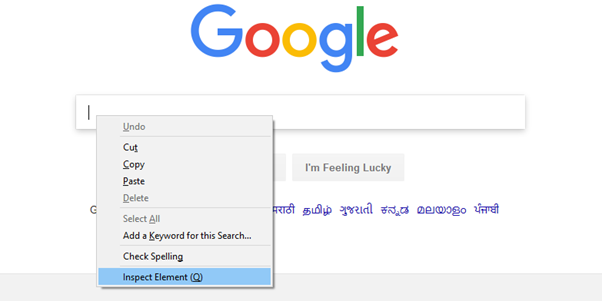
**Step4**. We would need a unique identification for the web elements like Google Search text box and Search button in order to automate them through our test script. These unique identifications are configured along with some Commands/Syntax to form Locators. Locators help us to locate and identify a particular web element in context of a web application.

The method for finding a unique identification element involves inspection of HTML codes.

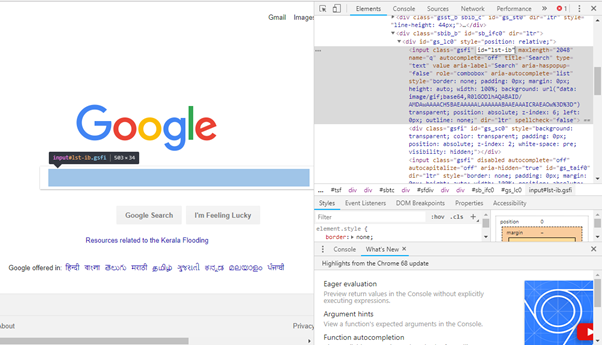
* Open URL: [https://www.google.com](https://www.google.com/" \t "_blank)

in your Chrome browser.

* Right click on the Google search text box and select Inspect Element.



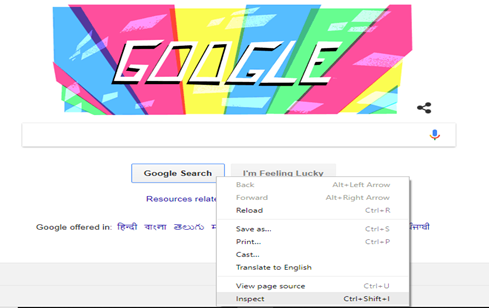
* It will launch a window containing all the specific codes involved in the development of the test box.



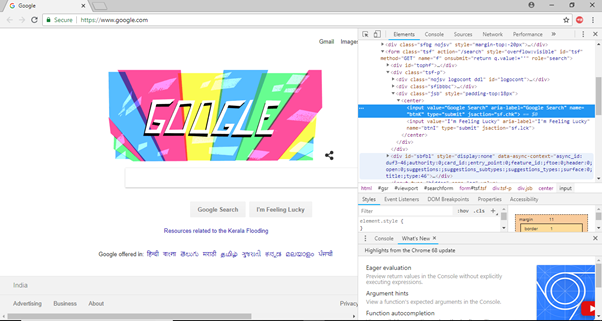
* Pick the value of id element i.e. "lst-ib".

Selenium WebDriver First Test Case

* Given below is the Java syntax for locating elements through "id" in Selenium WebDriver.
  1. driver.findElement(By.id (<element ID>))
* Here is the complete code for locating Google Search text box in our test script.
  1. driver.findElement(By.id ("lst-ib"))
* Now, right click on the Google Search button and select Inspect Element.



* It will launch a window containing all the specific codes involved in the development of the Google Search button.



* Pick the value of **name** element i.e. "btnK".

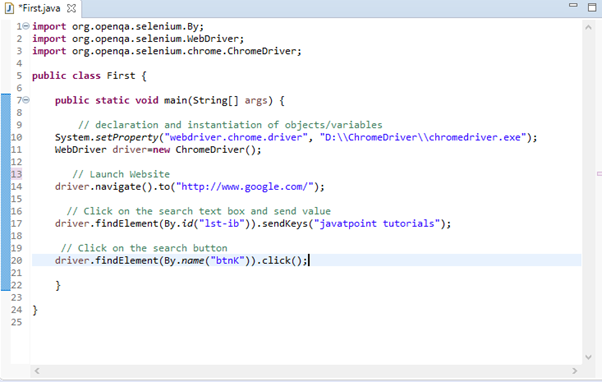
Selenium WebDriver First Test Case

* Given below is the Java syntax for locating elements through "name" in Selenium WebDriver.
  1. driver.findElement(By.name (<element name>))
* Here is the complete code for locating Google Search button in our test script.
  1. driver.findElement(By.name ("btnK"))

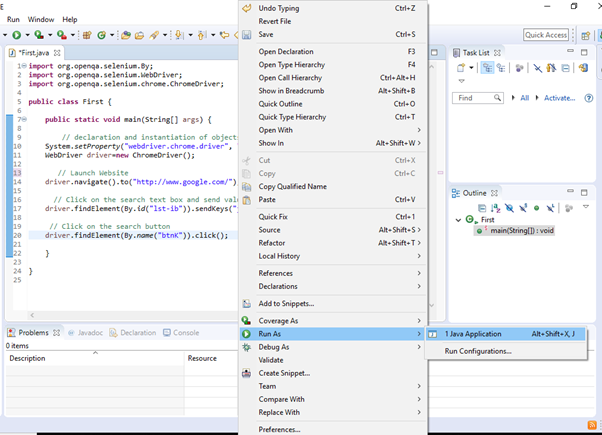
**Step5**. Now it is time to code. We have embedded comments for each block of code to explain the steps clearly.

1. **import** org.openqa.selenium.By;
2. **import** org.openqa.selenium.WebDriver;
3. **import** org.openqa.selenium.chrome.ChromeDriver;
5. **public** **class** First {
7. **public** **static** **void** main(String[] args) {
9. // declaration and instantiation of objects/variables
10. System.setProperty("webdriver.chrome.driver", "D:\\ChromeDriver\\chromedriver.exe");
11. WebDriver driver=**new** ChromeDriver();
13. // Launch website
14. driver.navigate().to("http://www.google.com/");
16. // Click on the search text box and send value
17. driver.findElement(By.id("lst-ib")).sendKeys("javatpoint tutorials");
19. // Click on the search button
20. driver.findElement(By.name("btnK")).click();
22. }
24. }

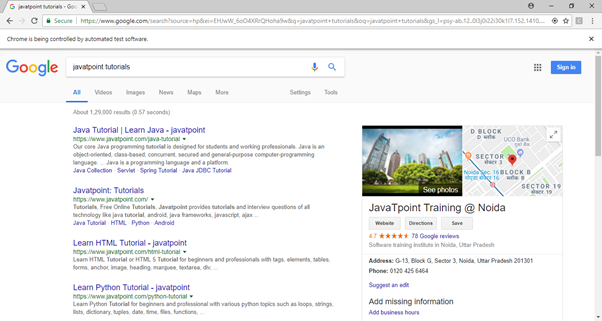
The Eclipse code window will look like this:



**Step6**. Right click on the Eclipse code and select **Run As > Java Application**.



**Step7**. The output of above test script would be displayed in Google Chrome browser.



## Explanation of the Code

### Import Packages/Statements

In java, import statements are used to import the classes present in another packages. In simple words, import keyword is used to import built-in and user-defined packages into your java source file.

1. **org.openqa.selenium.WebDriver** - References the WebDriver interface which is required to instantiate a new web browser.
2. **org.openqa.selenium.chrome.ChromeDriver** - References the ChromeDriver class that is required to instantiate a Chrome-specific driver onto the browser instantiated by the WebDriver class.

### Instantiating objects and variables

A driver object is instantiated through:

1. WebDriver driver=**new** ChromeDriver();

### Launch Website

To launch a new website, we use navigate().to() method in WebDriver.

1. driver.navigate().to("http://www.google.com/");

### Click on an element

In WebDriver, user interactions are performed through the use of Locators which we would discuss in later sessions of this tutorial. For now, following instance of code is used to locate and parse values in a specific web element.

1. driver.findElement(By.id("lst-ib")).sendKeys("javatpoint tutorials");