Selenium is a suite of tools. It consists of Selenium IDE, Selenium RC, Selenium Webdriver, and Selenium Grid

****Selenium IDE:****

Selenium IDE (Integrated Development Environment) is a Firefox plugin. It is the simplest framework in the Selenium Suite. It allows us to record and playback the scripts. Even though we can create scripts using Selenium IDE, we need to use Selenium RC or [Selenium WebDriver](https://www.softwaretestingmaterial.com/selenium-tutorial/) to write more advanced and robust [test cases](https://www.softwaretestingmaterial.com/test-case-template-with-explanation/).

****Selenium RC:****

Selenium RC AKA Selenium 1. Selenium RC was the main Selenium project for a long time before the WebDriver merge brought up Selenium 2. Selenium 1 is still actively supported (in maintenance mode). It relies on JavaScript for automation. It supports Java, Javascript, Ruby, PHP, Python, Perl and C#. It supports almost every browser out there.

Note: Selenium RC is officially deprecated.

****Selenium WebDriver:****

Selenium WebDriver is a browser automation framework that accepts commands and sends them to a browser. It is implemented through a browser-specific driver. It controls the browser by directly communicating with it. Selenium WebDriver supports Java, C#, PHP, Python, Perl, Ruby.

****Operation System Support**** – Windows, Mac OS, Linux, Solaris  
****Browser Support**** – Mozilla Firefox, Internet Explorer, Google Chrome 12.0.712.0 and above, Safari, Opera 11.5 and above, Android, iOS, HtmlUnit 2.9 and above

****Selenium Grid:****

Selenium Grid is a tool used together with Selenium RC to run tests on different machines against different browsers in parallel. That is, running multiple tests at the same time against different machines running different browsers and operating systems.

Selenium WebDriver is a well designed object oriented API supports many languages such as Java, C#, Python etc. Let’s see what is API and then we move on to know the complete picture of Selenium WebDriver Architecture.

****API:****

Application Programming Interface (API) works as an interface between various software components.

***[Learn API Testing in 10 mins](https://www.softwaretestingmaterial.com/api-testing/)***

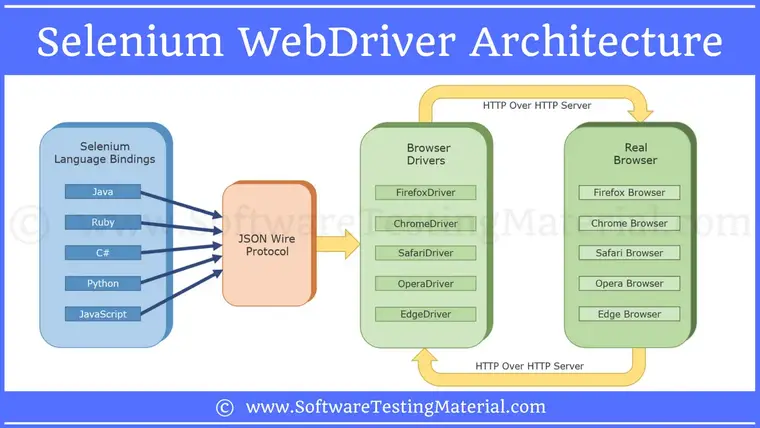
****Selenium WebDriver API:****

Selenium Webdriver API helps in communication between languages and browsers. Selenium supports many programming languages such as Java, C#, Python etc., and also it supports multiple browsers such as Google Chrome, Firefox, Internet Explorer etc.,

***[Real Time Selenium WebDriver Interview Questions](https://www.softwaretestingmaterial.com/selenium-interview-questions/)***

Every browser has different logic of performing actions like loading a page, closing the browser etc.

Here is the Selenium WebDriver Framework Architecture Diagram

[](https://www.softwaretestingmaterial.com/wp-content/uploads/2017/12/Selenium-WebDriver-Architecture.png)

Selenium WebDriver Architecture

There are four components of Selenium Architecture:

1. Selenium Client Library
2. JSON Wire Protocol over HTTP
3. Browser Drivers
4. Browsers

***Must Read: [Top TestNG Interview Questions](https://www.softwaretestingmaterial.com/testng-interview-questions/)***

****Selenium Client Libraries/Language Bindings:****

Selenium supports multiple libraries such as Java, Ruby, Python, etc., Selenium Developers have developed language bindings to allow Selenium to support multiple languages. Check out Selenium Libraries in the [official site](http://www.seleniumhq.org/download/" \l "client-drivers" \t "https://www.softwaretestingmaterial.com/selenium-webdriver-architecture/_blank).

****JSON WIRE PROTOCOL Over HTTP Client:****

JSON stands for JavaScript Object Notation. It is used to transfer data between a server and a client on the web. JSON Wire Protocol is a REST API that transfers the information between HTTP server. Each BrowserDriver (such as FirefoxDriver, ChromeDriver etc.,)  has its own HTTP server.

****Browser Drivers:****

Each browser contains separate browser driver. Browser drivers communicate with respective browser without revealing the internal logic of browser’s functionality. When a browser driver is  received any command then that command will be executed on the respective browser and the response will go back in the form of HTTP response..

****Browsers:****

Selenium supports multipe browsers such as Firefox, Chrome, IE, Safari etc.,

Let’s see how Selenium WebDriver works internally

In real time, you write a code in your UI (say Eclipse IDE) using any one of the supported Selenium client libraries (say Java).

Example:

WebDriver driver  = new FirefoxDriver();

driver.get(https://www.softwaretestingmaterial.com)

Once you are ready with your script, you will click on Run to execute the program. Based on the above statements, Firefox browser will be launched and it will navigates to *softwartestingmaterial* website.

Here we see what will happen internally after you click on Run till the launch of Firefox browser.

Once you click on Run, every statement in your script will be converted as a URL with the help of JSON Wire Protocol over HTTP. The URL’s will be passed to the Browser Drivers. (In the above code, we took FirefoxDriver). Here in our case the client library (java) will convert the statements of the script to JSON format and communicates with the FirefoxDriver. URL looks as shown below.

[http://localhost:8080/{"url":"https://www.softwaretestingmaterial.com"}](http://localhost:8080/{\"url\":\"https:/www.softwaretestingmaterial.com\"})

Every Browser Driver uses a HTTP server to receive HTTP requests.  Once the URL reaches the Browser Driver, then the Browser Driver will pass that request to the real browser over HTTP. Then the commands in your selenium script will be executed on the browser.

If the request is *POST* request then there will be an action on browser

If the request is a *GET* request then the corresponding response will be generated at the browser end and it will be sent over HTTP to the browser driver and the Browser Driver over JSON Wire Protocol and sends it to the UI (Eclipse IDE).

This is all about Selenium WebDriver Architecture.