**Automation Testing**

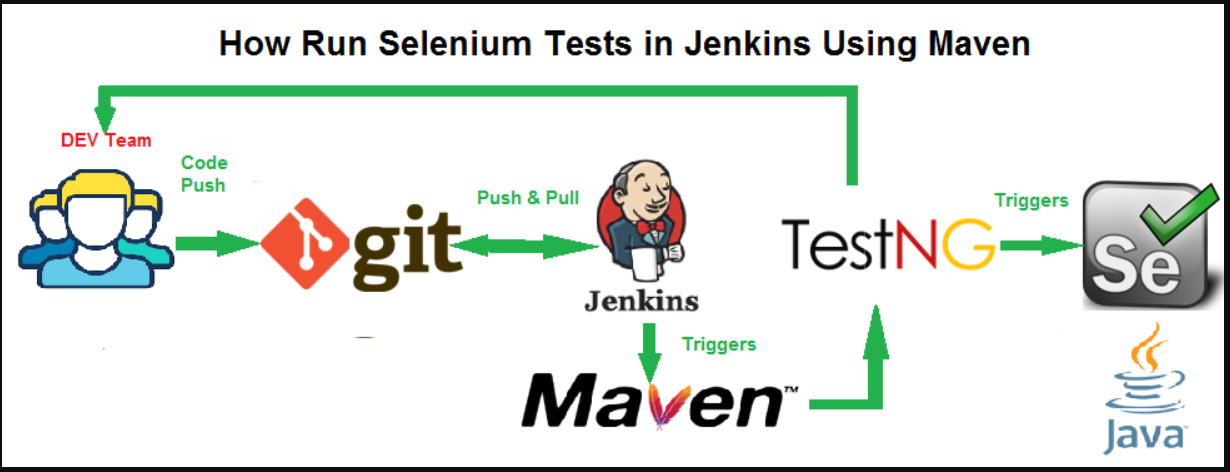
Automation testing is a Software testing technique to test and compare the actual outcome with the expected outcome. This can be achieved by writing test scripts or using any automation testing tool. Test automation is used to automate repetitive tasks and other testing tasks which are difficult to perform manually.

**System Requirement:**

1. 1 VM Machine for Github/Gitlab (For maintaining repository of automated test scripts).
2. 1 VM Machine for Eclipse, Selenium, Jenkins.

**Software Requirements:**

1. Eclipse (Open Source)
2. JAVA – Programming Language
3. Maven (Open Source)
4. Selenium API (Open Source)
5. TestNG (Open Source) – Report Generation
6. Jenkins (Open Source)
7. Github/Gitlab

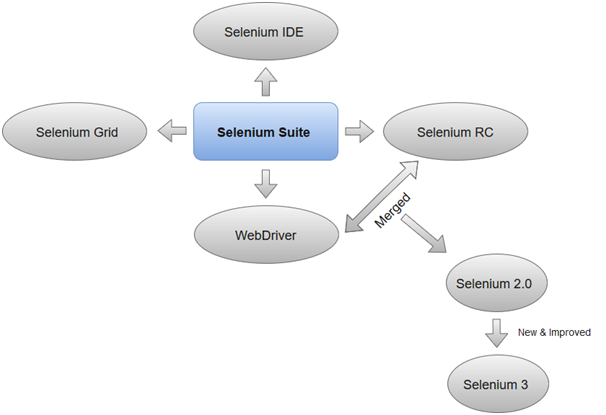


**Automation Tool Detail**

# 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. Selenium Remote Control (Now Deprecated)
3. WebDriver
4. 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.

### 2. Selenium Remote Control

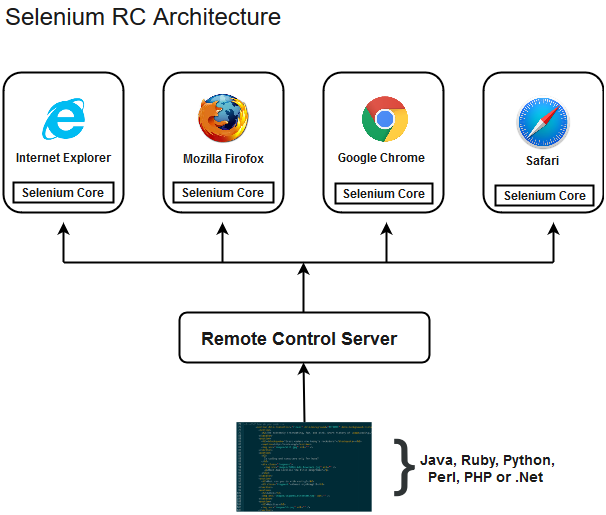
Selenium RC (officially deprecated by selenium) allows testers to write automated web application UI test in any of the supported programming languages. It also involves an HTTP proxy server which enables the browser to believe that the web application being tested comes from the domain provided by proxy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

Selenium RC comes with two components.

* Selenium RC Server (acts as a HTTP proxy for web requests).
* Selenium RC Client (library containing your programming language code).

Selenium RC had been considered quite effective for testing complex AJAX-based web user interfaces under a Continuous Integration System.

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

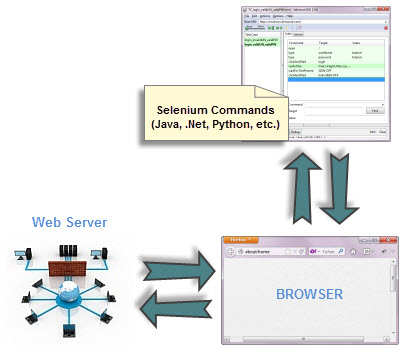


**3. Selenium WebDriver**

Selenium WebDriver (Selenium 2) is the successor to Selenium RC and is by far the most important component of Selenium Suite. Selenium WebDriver provides a programming interface to create and execute test cases. Test scripts are written in order to identify web elements on web pages and then desired actions are performed on those elements. WebDriver is a web automation framework that allows you to execute your tests against different browsers, not just Firefox, Chrome (unlike Selenium IDE). WebDriver also enables you to use a programming language in creating your test scripts (not possible in Selenium IDE).

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 web browser. Since, WebDriver directly calls the methods of different browsers hence we have separate driver for each browser.

**WebDriver Architecture:**



### 4. Selenium Grid

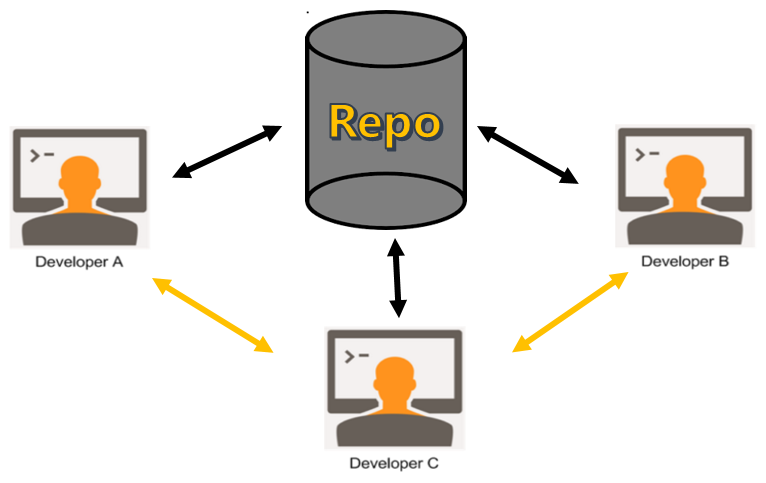
Selenium Grid is also an important component of Selenium Suite which allows us to run our tests on different machines against different browsers in parallel. In simple words, we can run our tests simultaneously on different machines running different browsers and operating systems.

Selenium Grid follows the **Hub-Node Architecture** to achieve parallel execution of test scripts. The Hub is considered as master of the network and the other will be the nodes. Hub controls the execution of test scripts on various nodes of the network.

**Test Script Repository: Github**

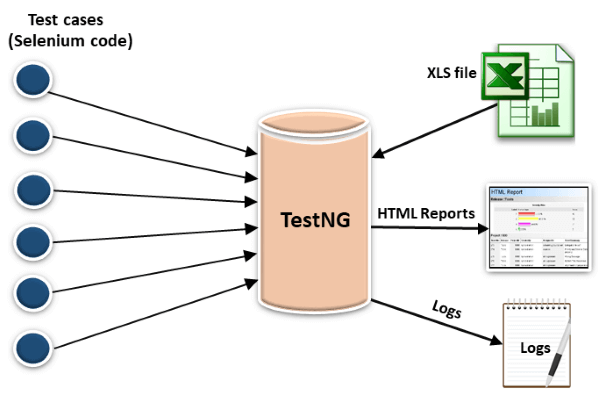
GitHub is a Git repository hosting service, but it adds many of its own features. While Git is a command line tool, GitHub provides a Web-based graphical interface. It also provides access control and several collaboration features, such as a wikis and basic task management tools for every project.

The flagship functionality of GitHub is “forking” – copying a repository from one user’s account to another. This enables you to take a project that you don’t have write access to and modify it under your own account. If you make changes you’d like to share, you can send a notification called a “pull request” to the original owner. That user can then, with a click of a button, merge the changes found in your repo with the original repo.



**Framework Design: TestNG**

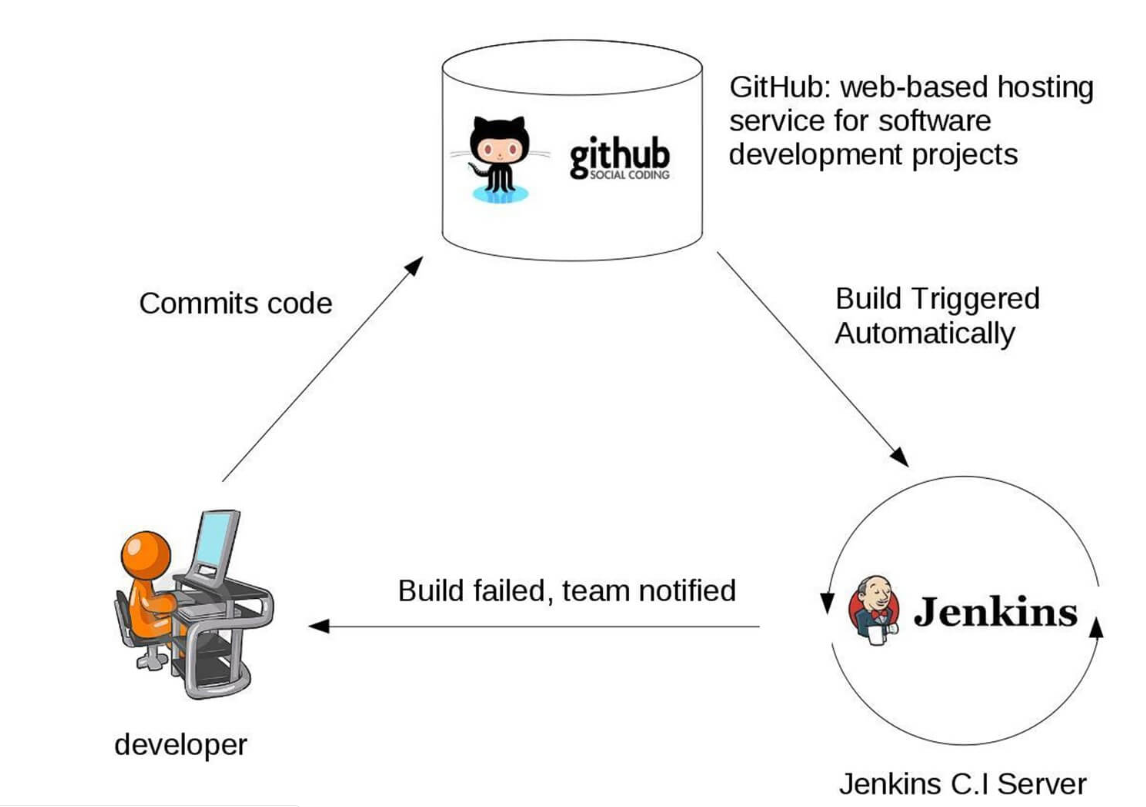
TestNG is a testing framework developed in the lines of JUnit and NUnit, however it introduces some new functionalities that make it more powerful and easier to use. TestNG is designed to cover all categories of tests: unit, functional, end-to-end, integration, etc., and it requires JDK 5 or higher. This tutorial provides a good understanding on TestNG framework needed to test an enterprise-level application to deliver it with robustness and reliability.



**Pipelining Tool: Jenkins**

Jenkins is an open source Java software system designed to aid in the continuous software integration process. It is a server-based system that runs in servlet containers such as Apache Tomcat. Jenkins supports version control tools (for example – Git) and can execute your projects based on Apache Ant, Apache Maven, and sbt as well as arbitrary shell/Windows batch scripts and commands. During the development of large systems, there is a need for a continuous process of deployment and testing new versions of your product. To solve this problem, we will use the Jenkins Pipeline.

 Jenkins Pipeline is a suite of plugins which support continuous delivery integration into Jenkins. In other words, it is a script that has several jobs in it and can be executed each time a certain condition is met. Let’s have a look at an example of working with Pipeline.



**Area Covered under Automation Testing**

1. **Sanity Testing:** Sanity Testing is a subset of regression testing. Sanity testing is performed to ensure that the code changes that are made are working as properly. Sanity testing is a stoppage to check whether testing for the build can proceed or not. The focus of the team during sanity testing process is to validate the functionality of the application and not detailed testing. Sanity testing is generally performed on build where the production deployment is required immediately like a critical bug fix.

**Features of Sanity Testing:**

**1. Subset of Regression Testing:**Sanity testing is a subset of regression testing and focuses on the smaller section of the application.

**2. Unscripted:**Most of the times sanity testing is not scripted.

**3. Not documented:**Usually sanity testing is undocumented.

**4. Narrow and deep:**Sanity testing is narrow and deep approach of testing where limited functionalities are covered deeply.

**5. Performed by testers:**   
Sanity testing is normally performed by testers.

1. **Regression Testing:** It is the process of testing the modified parts of the code and the parts that might get affected due to the modifications to ensure that no new errors have been introduced in the software after the modifications have been made. Regression means return of something and in the software field, it refers to the return of a bug.

**Advantages of Regression Testing:**

1. It ensures that no new bugs have been introduced after adding new functionalities to the system.

2. As most of the test cases used in Regression Testing are selected from the existing test suite and we already know their expected outputs. Hence, it can be easily automated by the automated tools.

3. It helps to maintain the quality of the source code.

**Development of Test Scripts**

**Programming Language: JAVA**

[Selenium](https://www.edureka.co/testing-with-selenium-webdriver) is the most prominent tool in the field of automation testing, whereas [Java](https://www.edureka.co/java-j2ee-training-course), on the other hand, is the most heavily used programming language in today’s market. Both of these technologies together make a perfect combination for automation Testing.

**Integrated Development Environment: Eclipse**

Eclipse is an integrated development environment (IDE) for Java and other programming languages like C, C++, PHP, and Ruby etc. Development environment provided by Eclipse includes the Eclipse Java development tools (JDT) for Java, Eclipse CDT for C/C++, and Eclipse PDT for PHP, among others.

The Eclipse platform which provides the foundation for the Eclipse IDE is composed of plug-ins and is designed to be extensible using additional plug-ins. Developed using Java, the Eclipse platform can be used to develop rich client applications, integrated development environments and other tools. Eclipse can be used as an IDE for any programming language for which a plug-in is available.

**Process Workflow:**

Manual Testcases > Converted into Testscripts (Using JAVA) < All the testcases will be committed over testcase repository (Github/Gitlab) < Test Suite will be executed < Test reports will be generated via TestNG reporting framework

**Manual Testcases**

**Test Scripts**

**Test Scripts committed over Github/Gitlab**

**Test Reports (TestNG)**

**Test Execution**

**Pre Requisite for Automation Testing:**

1. **Separate Automation Testing Environment.**
2. **OTP, Captcha should be disabled.**
3. **Window based popup should be disabled.**
4. **To check parallel execution of testcases over different browser single login security should be disabled.**