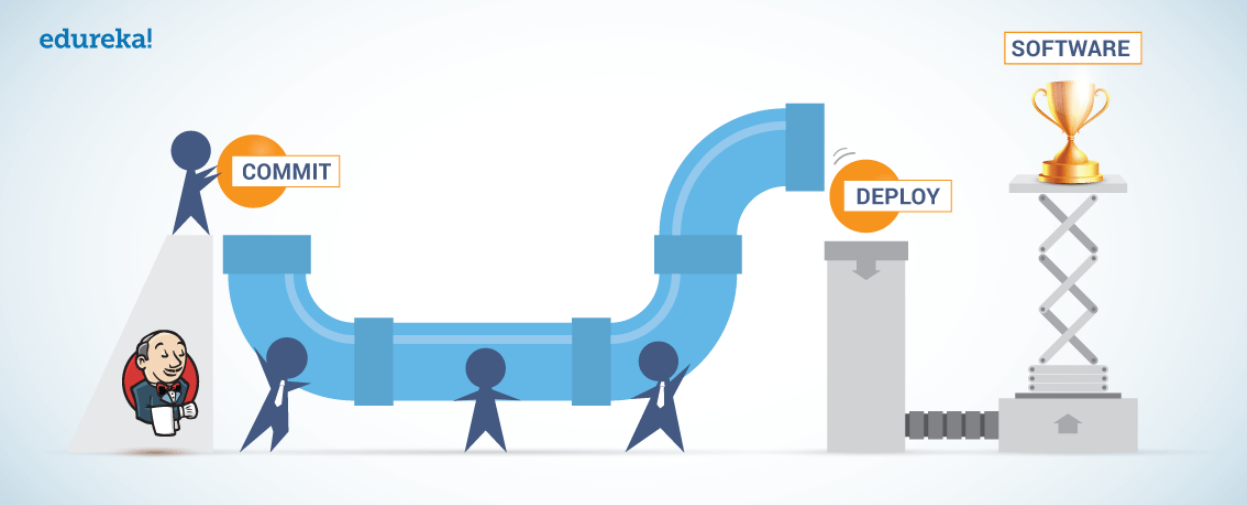
Jenkins Master and Slave Architecture

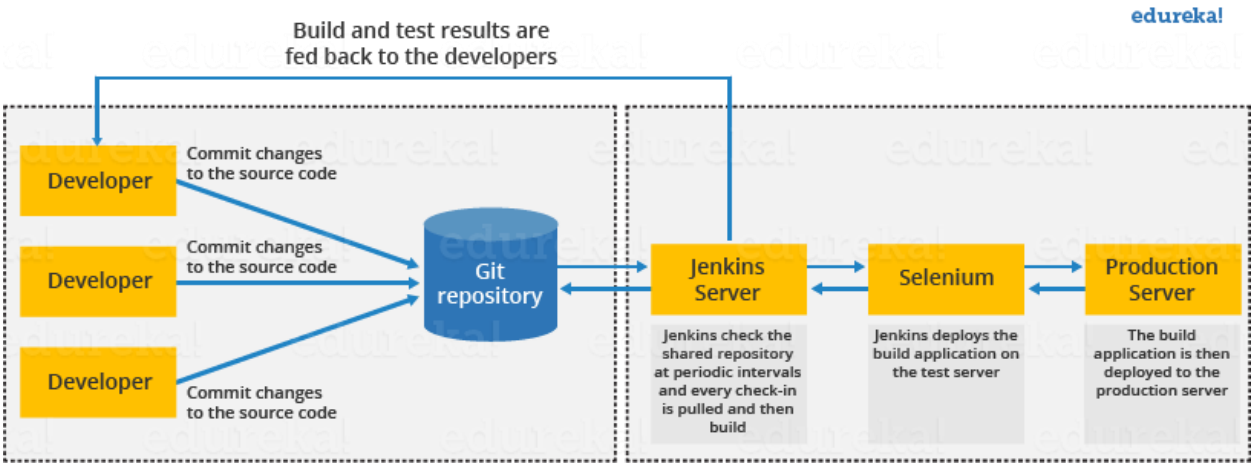
Jenkins Architecture

Let us have a look at the Jenkins Architecture below diagram depicts the same.



Advantages of Jenkins include:

- It is an open-source tool with great community support.
- Too easy to install.
- It has 1000+ plugins to ease your work. If a plugin does not exist, you can code it and share it with the community.
- It is free of cost.
- It is built with Java and hence, it is portable to all the major platforms.



This single Jenkins server was not enough to meet certain requirements like:

- Sometimes you might need several different environments to test your builds. This cannot be done by a single Jenkins server.
- If larger and heavier projects get built on a regular basis then a single Jenkins server cannot simply handle the entire load.

To address the above-stated needs, Jenkins distributed architecture came into the picture.

Jenkins Distributed Architecture

Jenkins uses a Master-Slave architecture to manage distributed builds. In this architecture, Master and Slave communicate through TCP/IP protocol.

Jenkins Master

Your main Jenkins server is the Master. The Master's job is to handle:

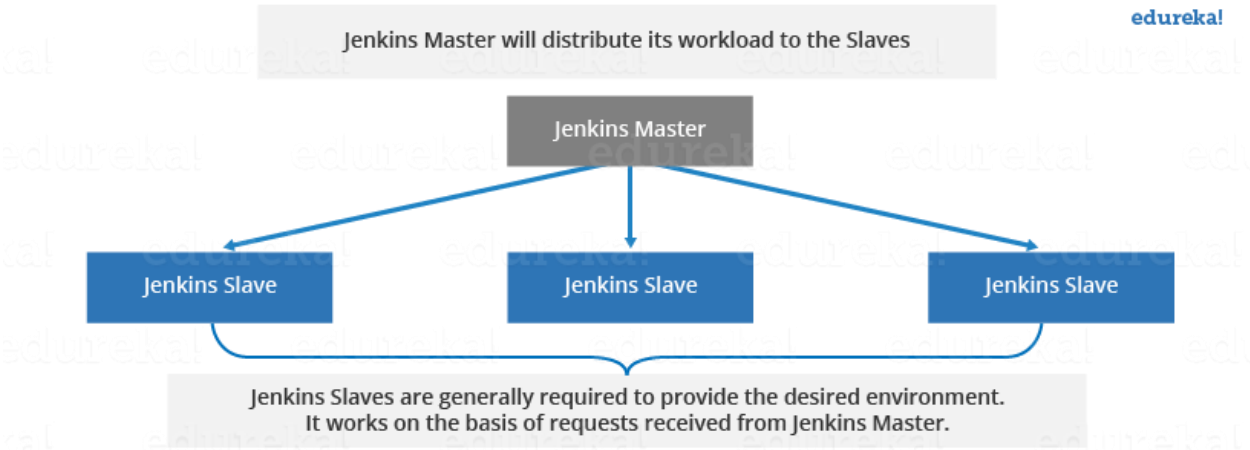
- Scheduling build jobs.
- Dispatching builds to the slaves for the actual execution.
- Monitor the slaves (possibly taking them online and offline as required).
- Recording and presenting the build results.
- A Master instance of Jenkins can also execute build jobs directly.

Jenkins Slave

A Slave is a Java executable that runs on a remote machine. Following are the characteristics of Jenkins Slaves:

- It hears requests from the Jenkins Master instance.
- Slaves can run on a variety of operating systems.
- The job of a Slave is to do as they are told to, which involves executing build jobs dispatched by the Master.
- You can configure a project to always run on a particular Slave machine or a particular type of Slave machine, or simply let Jenkins pick the next available Slave.

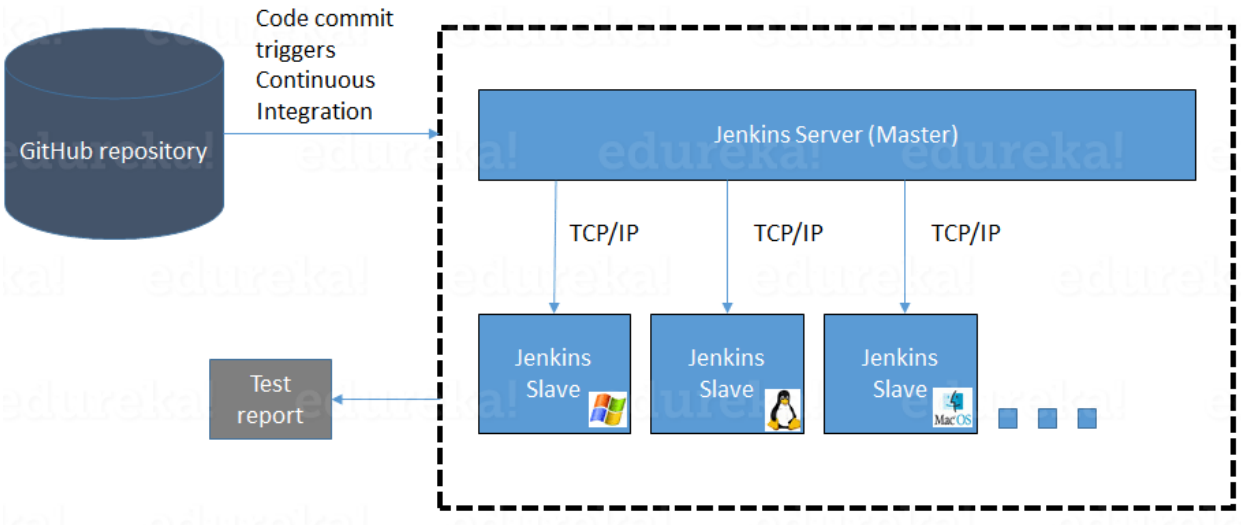
The diagram below is self-explanatory. It consists of a Jenkins Master which is managing three Jenkins Slave.



How Jenkins Master and Slave Architecture works?

Now let us look at an example in which we use Jenkins for testing in different environments like Ubuntu, MAC, Windows, etc.

The diagram below represents the same:

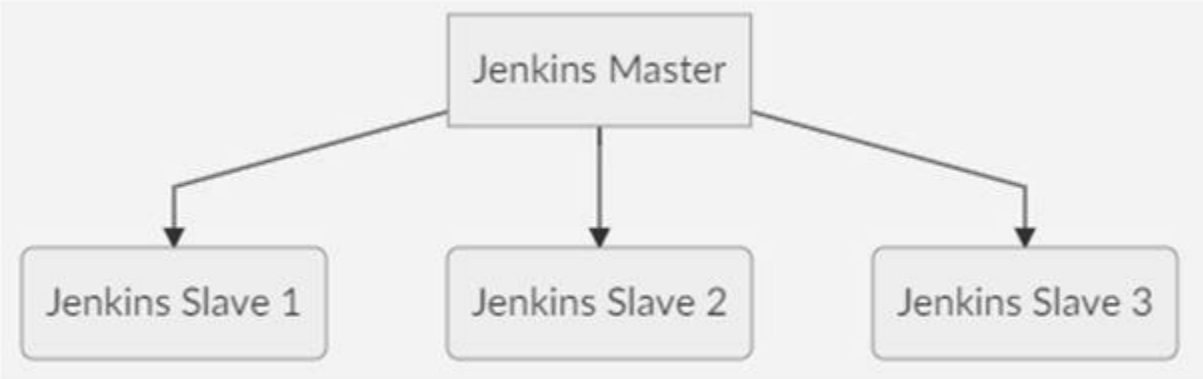


The above image represents the following functions:

- Jenkins checks the Git repository at periodic intervals for any changes made in the source code.

- Each builds requires a different testing environment which is not possible for a single Jenkins server. In order to perform testing in different environments, Jenkins uses various Slaves as shown in the diagram.
- Jenkins Master requests these Slaves to perform testing and to generate test reports.
- A Jenkins master comes with the basic installation of Jenkins, and in this configuration, the master handles all the tasks for your build system.

- If you are working on multiple projects, you may run multiple jobs on each project. Some projects need to run on some nodes, and in this process, we need to configure slaves. Jenkins slaves connect to the Jenkins master using the Java Network Launch Protocol.
- Jenkins Master and Slave Architecture



- The Jenkins master acts to schedule the jobs, assign slaves, and send builds to slaves to execute the jobs.
- It will also monitor the slave state (offline or online) and get back the build result responses from slaves and the display build results on the console output. The workload of building jobs is delegated to multiple **slaves**.