Jenkins

Jenkins is an open-source server that is written entirely in Java. It lets you execute a series of actions to achieve the continuous integration process, that too in an automated fashion.

This CI server runs in servlet containers such as Apache Tomcat. Jenkins facilitates continuous integration and continuous delivery in software projects by automating parts related to build, test, and deployment. This makes it easy for developers to continuously work on the betterment of the product by integrating changes to the project.

Jenkins automates the software builds in a continuous manner and lets the developers know about the errors at an early stage. A strong Jenkins community is one of the prime reasons for its popularity. Jenkins is not only extensible but also has a thriving plugin ecosystem.

CI :: Continuous Integration

CD: Continuous Deployment Delivery

Jenkins helps the team to automat various activities related to AJILE model project development to deployment which says development, build, test and deployment operations should happen smoothly

Jenkins also useful in the maintainance of the Project towards fixing the bugs and resolving the issues by enabling this continuous Integration and Continuos Deployment Process.

we can intergrate muliple supporting softwares or plugins in Jenkins eco System as needed.

(like maven/gradle, SonarQube, selenium, junit, Mockito and etc...)

Some of the possible steps that can be performed using Jenkins are:

• Software build using build systems such as Gradle, Maven, and more.

• Automation testing using test frameworks such as Nose2, PyTest, Robot, Selenium, and

more.

• Execute test scripts (using Windows terminal, Linux shell, etc.

• Achieve test results and perform post actions such as printing test reports, and more.

• Execute test scenarios against different input combinations for obtaining improved test coverage.

• Continuous Integration (CI) where the artifacts are automatically created and tested. This aids in identification of issues in the product at an early stage of development. close to 1500+ plugins contributed by the

community. Plugins help in customizing the experience with Jenkins, along with providing support for accelerating activities related to building, deploying, and automating and etc.. Salient Features Of Jenkins

Jenkins is more functionality-driven rather than UI-driven hence, there is a learning curve involved in getting to know what is Jenkins. Here are the powerful developer-centric features offered by Jenkins:

1. Easy Installation & Configuration

Jenkins is a self-contained Java program that is agnostic of the platform on which it is installed. It is available for almost all the popular operating systems such as Windows, different flavors of Unix, and Mac OS.

It is available as a normal installer, as well as a .war file. Once installed, it is easy to configure using its web interface.

2. Open-Source

As it is open-source, it is free for use. There is a strong involvement of the community which makes it a powerful CI/CD tool. You can take support from the Jenkins community, whether it is for extensibility, support, documentation, or any other feature related to Jenkins.

3. Thriving Plugin Ecosystem

The backbone of Jenkins is the community and the community members have been instrumental in the development (and testing) of close to 1500+ plugins available in the Update Center.

4. Easy Distribution

Jenkins is designed in such a manner that makes it relatively easy to distribute work across multiple machines and platforms for the accelerated build, testing, and deployment.

note:: The Jenkin agent for "dev","test" env.. deploys the build code in Local server of the company that runs in windows OS (or) in cloud note:: The Jenkin agent for "uat", "prod" env.. deploys the build code in Producttion server of the Client Organization that runs in Linux OS

(or) in cloud

note:: Keeping the code ready for execution is called Project/Application Build Process

=>learning curve is high means learning time is high =>learning curve is small means learning time is less

How does the open source companies are surveying?

Ans) Open Source software companies release their sofwares at free of cost to the market but they try to generate revenue for following services and bi-products

a) by offering certification exams

b) by offering corporate trainings

c) By plublishing books

d) by releasing bi products like IDEs, drivers and etc..

e) by offering services to s/w development companies

if they struckup in the middle of the Project

and etc..

How Does Jenkins Work?

we look at the internal functioning of Jenkins

i.e. what happens once the developer commits changes to the repository and how CI/CD

is realized in Jenkins. We also look at the Master-Agent architecture in Jenkins.

Architecture Of Jenkins

Before we dive into how does Jenkins works, we must understand the architecture of Jenkins. These are the series of steps that outlines the interaction between different elements in Jenkins:

• Developers do the necessary modifications in the source code and commit the changes

to the repnew version of that file will be created in the version control system that is used for maintaining the repository of source code.

(Like in GIT)

• The repository is continuously checked by the Jenkins CI server for any changes (either

in the form of code or libraries) and changes are pulled by the server.

• In the next step, we ensure that the build with the 'pulled changes' is going through or

jar/warfleif

not. The Build server performs a build with the code and an executable jar war file)if

the build process is successful. In case of a build failure, an automated email with a link to build logs and other build artifacts is sent to the developer.

Development

and UnitTestng activities

and also committing the code

Jenkins Pulls committed chanages

Jenkite Project

Builds

(jar/war file)

• In case of a successful build, the built application (or executable) is deployed to the test server. This step helps in realizing continuous testing where the newly built executable goes through a series of automated tests. Developers are alerted in case the changes have caused any breakage in functionality.

Testing the Project (Selenium Automated testing)

• If there are no build, integration, and testing issues with the checked-in code, the changes and tested application are automatically deployed to the Prod/Production (Continues deployment activitity)

server.

Here is the diagrammatic representation of the Jenkins architecture:

COMMIT CHANGES

TO THE SOURCE CODE

Deployment of the Project

Development-UnitTesting-

Static Testing

I

Buid Process

(GIT)

Testing

Live Deployment

Code Repository

CI:: Continuos Integration

CD: Continous Delivery

JENKINS SERVER

SELENIUM

PRODUCTION

SERVER

JENKINS CHECK THE SHARED

REPOSITORY AT PERIODIC INTERVALS

JENKINS DEPLOYS THE BUILD APPLICATION ON THE TEST SERVER

THE BUILD APPLICATION IS

THEN DEPLOYED TO THE PRODUCTION

AND EVERY CHECK-IN IS PULLED AND

SERVER

CI

THEN BUILD

CD

developers

A single Jenkins server might not be sufficient to realize the following requirements:

• Testing needs to be performed on different environments (i.e. code written using different languages e.g. Java, Python, C, etc. are committed to the version control system), where a single server might not suffice the requirement.

• A single Jenkins server might not be sufficient to handle the load that comes with large-scale software projects. (Especially Micro Service Architecture Projects)

In such scenarios, the distributed (or Master-Agent) architecture of Jenkins is used for continuous integration and testing. Diving deeper into how does Jenkins works, we take a look at the architecture of Jenkins.

Master- Agent Architecture In Jenkins

(Also called as Master-Slave Architecture)

The master-agent (or distributed) architecture in Jenkins is used for managing distributed

builds. The Master and Agent(s) communicate through the TCP/IP protocol.

These are the roles and responsibilities of the Jenkins Master and Agent(s):

Jenkins Master

The main server in Jenkins is the Master. Here are the jobs handled by Jenkins Master:

• Schedule build jobs

• Choosing the appropriate agent in the master-agent ecosystem for dispatching the builds.

• Monitor agents and take them online/offline as and when required.

• Presenting the build results (and reports) to the developer.

The Jenkins master can also execute the jobs directly but it is always recommended to select the appropriate agent(s) for build and execution-related tasks.

Jenkins Agent(s)

A agent is a remote machine that is connected to the Master. Depending on the project

and build requirements, you could opt for 'N' number of agents. agents can run on different operating systems and depending on the 'type of build request, the appropriate Agent is chosen by the Master for build execution and testing.

Here are the jobs handled by the Jenkins Agent(s):

The Jenkins Master and the Jenkins Agents are the instances

of the Jenkins server /Jenkins software

• Listen to commands from the Jenkins Master.

• Execute build jobs that are dispatched by the Master.

• Developers have the flexibility to run the build and execute tests on a particular agent

or a particular type of Agent. The default option is Jenkins Master selecting the best- suited Agent for the job.

Here is a simple diagrammatic representation of how does Jenkins work, with multiple Jenkins Agents connected to the Jenkins Master:

Jenkins master will distribute its workload to the agent

Jenkins Agent

(java)

Jenkins Master

Jenkins Agent (.net)

Jenkins Agent python)

Jenkins agent are generally required to provide the desired environment. It works on the basis of requests received from Jenkins Master.

How Does Jenkins Work In Master-Agent

Architecture?

In the Jenkins Master-Agent architecture shown below, there are three Agents, each

running on a different operating system (i.e. Windows 10, Linux, and Mac OS).

REMOTE SOURCE

(Like GIT)

CODE REPOSITORY

JENKINS MASTER PULLS THE CODE EVERYTIME THERE IS A COMMIT

(java)

JENKINS AGENT

WINDOWS

مله

JENKINS SERVER

MASTER

(.net)

(python)

JENKINS AGENT

JENKINS AGENT

A LINUX

MACOS

In "dev" "test" env.. we want to

execute the build in the windows OS env..

where is "uat" and "prod" envs.we want to execute the build in Linux OS

> JENKINS MASTER DISTRIBUTES ITS WORKLOAD TO ALL THE AGENT

> ON REQUEST FROM JENKINS MASTER, THE AGENT CARRY OUT BUILDS AND TESTS AND PRODUCE TEST REPORTS

• Developers check-in their respective code changes in 'The Remote Source Code Repository' that is depicted on the left-hand side. (Like GITHUB)

• Only the Jenkins master is connected to the repository and it checks for code-changes (in the repository) at periodic intervals. All the Jenkins Agents are connected to the Jenkins Master.

• Jenkins master dispatches the request (for build and test) to the appropriate Jenkins Agent depending on the environment required for performing the build. This lets you perform builds and execute tests in different environments across the entire architecture.

• The Agent performs the testing, generates test reports, and sends the same to the Jenkins Master for

monitoring

As developers keep pushing code, Jenkins Agents can run different builds versions of the code for different

platforms. Jenkins Master (or Master Node) controls how the respective builds should operate.

Q) Jenkins falls under which category tool?

Ans) Build and deployment automation tool or CI and CD Tool

Q) What are alternate tools for Jenkins?

Ans) git hub (some extend), git lab (some extend), autorabit, gearset, ansible automation platform, bitraise, buddy and etc...

Q) Is Jenkins devOps Tool or DevSecOpsTool?

ans) yes, it is main tool for DevOps

Q) Who needs

Jenkins knowledge?

ans) Build and Deployment team (DevOps team) (or)

Any Domain,Full stack developers