Date: 09.07.2025



Copyright @ 2024 PibyThree.com All Rights Reserved

Jenkins

Contents

[1. What is Jenkins? 2](#_Toc202970603)

[2. What is CI/CD and its need 2](#_Toc202970604)

[3. SCM and Jenkins 2](#_Toc202970605)

[4. Jenkins Architecture 2](#_Toc202970606)

[5. Installing and Accessing Jenkins 2](#_Toc202970607)

[6. Job Types in Jenkins 3](#_Toc202970608)

[a. Freestyle Projects 3](#_Toc202970609)

[b. Pipeline Projects 3](#_Toc202970610)

[c. Multi-branch Pipeline 3](#_Toc202970611)

[d. Maven Projects 3](#_Toc202970612)

[e. Multi-Configuration 3](#_Toc202970613)

[f. Organization Folders 3](#_Toc202970614)

[7. Jenkins Pipeline 3](#_Toc202970615)

[a. What is Pipeline? 3](#_Toc202970616)

[b. Key concepts 4](#_Toc202970617)

[c. Syntax 4](#_Toc202970618)

[d. Benefits 4](#_Toc202970619)

[8. Resources 5](#_Toc202970620)

# 1. What is Jenkins?

Jenkins is a self-contained, open source automation server which can be used to automate all sorts of tasks related to building, testing, and delivering or deploying software.

# 2. What is CI/CD and its need

* **Continuous Integration (CI):** Developers merge code changes into a central repository as often as possible. This practice helps reduce testing costs and the number of bugs that get shipped to production.
* **Continuous Delivery or Continuous Deployment (CD):** The next step in the process, where code changes are tested and deployed fully automatically (with or without manual approval steps).

Modern software development requires frequent integration and deployment. Manual builds and testing introduce delays and human errors. Jenkins automates these steps, increasing speed and reliability.

# 3. SCM and Jenkins

The first phase in CI/CD practice is code development. For this, a SCM tool like Git is required to track code changes, collaborate, and manage other versions.

Jenkins can integrate with various popular SCM tools like GitHub, GitLab, BitBucket, etc. using the Git plugin.

# 4. Jenkins Architecture

Jenkins follows a **master-slave architecture** (also called as controller-agent). The **controller** handles UI, job scheduling, and plugin management. **Agents** run on separate machines to execute jobs in parallel.

The Jenkins architecture is designed for **distributed build environments**. It allows us to use **different environments for each build** project balancing the workload among multiple agents running jobs in parallel.

The **Jenkins controller** is the original node in the Jenkins installation. The Jenkins controller administers the Jenkins agents and orchestrates their work, including scheduling jobs on agents and monitoring agents.

**Agents** may be connected to the Jenkins controller using either local or cloud computers. The agents require a Java installation and a network connection to the Jenkins controller.

# 5. Installing and Accessing Jenkins

Jenkins can be installed on various platforms like Docker, Kubernetes, Linux, MacOS, Windows, Cloud services, etc. Refer to the official documentation for the steps: [Installing Jenkins](https://www.jenkins.io/doc/book/installing/)

Running Jenkins in a Docker container

1. docker run -p 8080:8080 -p 50000:50000 -d -v jenkins\_home:/var/jenkins\_home jenkins/jenkins:lts 🡪 run Jenkins in a container
2. docker ps 🡪 get the containerIdd
3. docker logs <containerId> 🡪 get the initialAdminPassword by checking logs  
   or   
   docker exec <containerId> cat /var/jenkins\_home/secrets/initialAdminPassword
4. Open `localhost:8080` in the browser
5. Paste the copied initialAdminPassword
6. Install the suggested plugins
7. Create the first admin user

Jenkins is now ready to be used. You may install any required plugins or configure jobs.

# 6. Job Types in Jenkins

|  |  |  |
| --- | --- | --- |
| **Project Type** | **Use Case** | **Example Workflow** |
| Freestyle Projects | Simple, custom workflows with sequential steps | Code checkout → Test → Build → Push → Deploy |
| Pipeline Projects | Complex, code-defined CI/CD pipelines | Multi-staged pipeline with conditional steps |
| Multi-branch Pipeline | Automated branch management for continuous development | Auto-detect branches and trigger corresponding pipelines |
| Maven Projects | Java projects using Maven | Execute Maven commands based on pom.xml |
| Multi-Configuration | Running builds across multiple configurations | Testing various environments and parameter combinations |
| Organization Folders | Hierarchical organization of projects | Grouping projects for clearer management in large environments |

# 7. Jenkins Pipeline

## What is Pipeline?

* Jenkins Pipeline is a suite of plugins that supports implementing and integrating continuous delivery pipelines into Jenkins.
* The definition of a Jenkins Pipeline is written into a text file (called a Jenkinsfile) which in turn can be committed to a project’s source control repository.
* This is the foundation of "Pipeline-as-code", treating the CD pipeline as a part of the application to be versioned and reviewed like any other code.

## Key concepts

|  |  |
| --- | --- |
| **Pipeline** | A user-defined model of a Continuous Delivery (CD) pipeline. |
| **Node** | A machine where Jenkins runs. |
| **Stage** | A block that defines a segment of the pipeline. |
| **Step** | A single task that is executed within a stage. |

## Syntax

A Jenkinsfile can be written using two types of syntax - Declarative and Scripted.

* Declarative Pipeline is designed to make writing and reading Pipelines easier and more structured. [ Opinionated DSL ]
* Scripted Pipeline provides more flexibility and control, allowing for more complex workflows. [ Groovy DSL ]

|  |  |
| --- | --- |
| **Declarative Pipeline** | **Scripted Pipeline** |
| pipeline {  agent any  stages {  stage('Build') {  steps {  //  }  }  stage('Test') {  steps {  //  }  }  stage('Deploy') {  steps {  //  }  }  }  } | node {  stage('Build') {  //  }  stage('Test') {  //  }  stage('Deploy') {  //  }  } |

## Benefits

* Enables a more streamlined CI/CD process
* Code-as-configuration for easy version control and sharing.
* Resilience with features like pause and resume.
* Efficient handling of complex build processes with minimal job maintenance.
* Seamless integration with numerous Jenkins plugins that extend pipeline capabilities.

# 8. Resources

[Jenkins User Documentation](https://www.jenkins.io/doc/)

[What Is CI/CD? Complete 2025 Guide |](https://octopus.com/devops/ci-cd/)

[Using Jenkins agents](https://www.jenkins.io/doc/book/using/using-agents/)

[Installing Jenkins](https://www.jenkins.io/doc/book/installing/) [Run Jenkins in Docker Container - YouTube](https://www.youtube.com/watch?v=pMO26j2OUME)

[Pipeline](https://www.jenkins.io/doc/book/pipeline/)

[Jenkins Course - KodeKloud Notes](https://notes.kodekloud.com/docs/Jenkins-For-Beginners/Introduction-and-Basics/Course-Introduction)