

Jenkins Roadmap for Java Developers

Here's the roadmap we'll follow to cover all **required and essential topics** in one go:

✅ Jenkins Complete Learning Plan:

1. What is Jenkins?
 2. Why Jenkins is used in DevOps?
 3. Jenkins Architecture (Master-Agent)
 4. Installation and Setup
 5. Jenkins Dashboard & UI Overview
 6. Jobs / Projects
 7. Build Triggers
 8. Build Pipeline (Freestyle & Pipeline Jobs)
 9. Jenkinsfile (Declarative vs Scripted)
 10. CI/CD Workflow with Jenkins
 11. Integration with Git & GitHub
 12. Integration with Maven/Gradle
 13. Build Artifacts
 14. Email Notifications
 15. Plugins
 16. Real-Life CI/CD Pipeline
 17. Jenkins Interview Questions
-

✅ 1. What is Jenkins?

Jenkins is an **open-source automation server** that helps in **continuous integration** and **continuous delivery** of software.

✅ 2. Why Use Jenkins in DevOps?

Purpose	Description
✓ Automate Build & Test	No manual code build required
✓ CI/CD Pipeline	Automate deploy on push/merge
✓ Integrate with GitHub	Automatically fetch code
✓ Schedule Jobs	Daily/nightly builds
✓ Plugins	Add support for Docker, Maven, SonarQube, etc.

3. Jenkins Architecture

Master-Agent Model:

- **Master:** Controls job scheduling & UI.
- **Agent/Slave:** Executes tasks on different machines.

 Use Case: Run different builds (e.g., Windows, Linux) simultaneously using agents.

4. Installing Jenkins

For Local Machine:

- Download from: <https://jenkins.io>
- Java required (JDK 8+)

bash

CopyEdit

```
java -jar jenkins.war
```

Then go to: <http://localhost:8080>

5. Jenkins Dashboard

- **New Item:** Create Job
- **Manage Jenkins:** Configure system & plugins
- **Build History:** Previous runs

- **Job Configuration:** Define source, trigger, build steps, etc.
-




6. Jobs (or Projects)

Types:

- **Freestyle Job:** UI-based build job (easy)
 - **Pipeline Job:** Script-based (Jenkinsfile)
-

7. Build Triggers

Trigger builds:

-  On Code Push (Webhook from GitHub)
-  Periodically (e.g., every night)
-  Manual Trigger

Example (cron-style schedule):

nginx

CopyEdit


H 0 * * * # Run at midnight every day

8. Pipeline (CI/CD) – Freestyle vs Pipeline

Type	Description
Freestyle	Easy UI-based jobs
Pipeline	Groovy-scripted, version-controlled, powerful

9. Jenkinsfile

Written in **Groovy**, defines build pipeline in code.

 Sample Declarative Pipeline:

groovy

CopyEdit

```
pipeline {
  agent any
  stages {
    stage('Build') {
      steps {
        sh 'mvn clean install'
      }
    }
    stage('Test') {
      steps {
        sh 'mvn test'
      }
    }
    stage('Deploy') {
      steps {
        sh 'scp target/app.jar user@server:/deploy'
      }
    }
  }
}
```

10. CI/CD Pipeline Flow

scss

CopyEdit

GitHub → Jenkins → Build (Maven) → Test → Deploy (Server/Docker)

11. GitHub Integration

- Configure webhook in GitHub

- Install “Git” plugin
 - Add GitHub repo URL in job
-

12. Maven or Gradle Integration

- Install Maven on Jenkins
- Use Maven goal in build step:

bash


CopyEdit

```
mvn clean install
```

13. Build Artifacts

After build, Jenkins can:

- Archive .jar, .war, .zip files
- Publish to artifact repo (Nexus, JFrog)

 Configuration:

groovy

CopyEdit

```
archiveArtifacts artifacts: '**/target/*.jar', fingerprint: true
```

14. Email Notifications

Send build success/failure emails:

- Install "Email Extension" plugin
 - Configure SMTP server
 - Add "Editable Email Notification" in post-build action
-

15. Plugins (Most Useful)

Plugin	Purpose
GitHub Plugin	Git integration
Maven Integration	Run Maven builds
Email Extension	Alerts
Docker Plugin	Build inside Docker
Build Pipeline	Visual pipeline view

16. Real-Life Jenkins CI/CD Use Case

Let's say you're working on a Spring Boot API:

1. Developer pushes code to GitHub
 2. Jenkins triggers build automatically
 3. Jenkins pulls code → runs mvn clean install
 4. Tests are executed
 5. If success → deploys .jar to server or Docker container
 6. Sends build result via email
-

17. Jenkins Interview Questions

Question	Quick Answer
What is Jenkins?	CI/CD automation tool
Difference: Freestyle vs Pipeline?	UI-based vs Script-based
What is a Jenkinsfile?	Groovy-based pipeline script
What is CI/CD?	Automating build, test, deploy
How to trigger build on Git push?	Use GitHub Webhook
Jenkins Master-Agent?	Master schedules, agent executes
Common plugins?	Git, Maven, Email, Docker

Question

Quick Answer

Can Jenkins run Docker builds? Yes, using Docker plugin






Jenkins with Docker Support

What does "Docker Support" in Jenkins mean?

It means Jenkins can:

- **Build Docker images**
 - **Run Jenkins inside a Docker container**
 - **Use Docker containers as build environments**
 - **Push Docker images to Docker Hub or other registries**
 - **Deploy containers automatically**
-

Why Use Docker in Jenkins?

Benefit	Explanation
 Consistent Environment	Same container runs on any machine (dev, test, prod)
 Fast Build/Deploy	Spin up containers quickly during builds
 Cloud Ready	Easy to scale with container orchestration
 Dependency Isolation	Avoid conflicts in different builds (Java, Node, etc.)
 Reusability	Reuse same Docker image across jobs

Jenkins + Docker Use Cases

1. Run Jenkins Inside Docker

- Jenkins itself runs as a container.
- Use the official image:

bash

CopyEdit

```
docker run -p 8080:8080 -v jenkins_home:/var/jenkins_home jenkins/jenkins:lts
```

2. Build Docker Images from Jenkins Job

- Install **Docker** on the Jenkins server.
- Add user jenkins to docker group:

bash

CopyEdit

```
sudo usermod -aG docker jenkins
```

- Use **Pipeline script** to build an image:

groovy

CopyEdit

```
pipeline {
    agent any

    stages {
        stage('Build Docker Image') {
            steps {
                sh 'docker build -t myapp:1.0 .'
            }
        }

        stage('Push to DockerHub') {
            steps {
                withCredentials([usernamePassword(credentialsId: 'docker-hub-creds',
                    usernameVariable: 'USER', passwordVariable: 'PASS')]) {
                    sh """
                        echo $PASS | docker login -u $USER --password-stdin

                        docker push myapp:1.0
                    """
                }
            }
        }
    }
}
```



```
}  
}  
}  
}
```

3. 📦 Use Docker in Jenkinsfile (Docker Agent)

groovy

CopyEdit

```
pipeline {  
  agent {  
    docker {  
      image 'maven:3.8.1-jdk-11'  
    }  
  }  
  stages {  
    stage('Build') {  
      steps {  
        sh 'mvn clean install'  
      }  
    }  
  }  
}
```

💡 This will:

- Pull the Maven Docker image
 - Run mvn clean install inside that container
-

🔧 Required Plugin

Plugin	Purpose
Docker Plugin	Interact with Docker from Jenkins
Docker Pipeline	Use docker {} block inside Jenkinsfile

Docker Credentials in Jenkins

To push images securely to Docker Hub:

1. Go to: Manage Jenkins → Credentials
 2. Add Docker Hub username/password
 3. Use it with withCredentials in the Jenkinsfile
-

Real-Life Scenario

1. Developer pushes code to GitHub
 2. Jenkins pulls code and runs pipeline
 3. Jenkins builds a Docker image from Dockerfile
 4. Tags it and pushes it to Docker Hub
 5. Triggers deployment using Docker on server/Kubernetes
-

Drawbacks / Considerations

Issue	Solution
Jenkins user may lack Docker access	Add Jenkins to Docker group
Security	Use secrets for Docker credentials
Resource Heavy	Use Docker agents only when needed

Summary

Feature	Description
Build Docker Images	Yes – docker build in pipeline

Feature	Description
Use Docker Containers as Agent	Yes – agent { docker { ... } }
Push to Docker Hub	Yes, using credentials
Run Jenkins in Docker	Yes, commonly used in DevOps
Plugin Needed?	Docker Pipeline Plugin