

Jenkins Pipeline

--> It is a way to define CI CD process as a code

--> CI CD workflow will be defined as a code in Jenkins pipeline and whenever we are dealing with complex CI/CD process it is highly recommended to go with Jenkins pipeline concept

Pipeline contains stages to perform CI CD

--> Clone Git Repo

--> Build that project with Maven

--> Review the code with tools (SonarQube)

--> Artifact upload using Nexus

--> Build Docker image

--> Add into the Docker hub

--> Push Docker image into registry

--> Deploy App in K8s

We are going to automate the entire CI/CD pipeline using two approaches:

1. Declarative pipeline

```
pipeline {
    agents any ---> where we want to execute our job
    tools {
        maven "maven.3.9"
    }

    stages {
        // 3 operations I am performing
        // cloning the code
        // building the project
        // reviewing the code
        // uploading using Nexus etc
        stage ('Git Clone'){
            steps {
                echo 'cloning git repo...'
            }
        }

        stage ('Maven Build'){
            steps {
                echo 'project build with Maven'
            }
        }

        stage ('Deploy'){
            steps {
                echo 'deploying application with Tomcat'
            }
        }
    }
}
```

Declarative pipeline with Jenkins + Git + Maven

```
pipeline {
    agent any

    tools {
        maven "maven-3.9.10"
    }
}
```

```

stages {
  stage('git clone') {
    steps {
      git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'
    }
  }
  stage('maven build') {
    steps {
      sh 'mvn clean compile test package'
    }
  }
  stage('deploy') {
    steps {
      echo 'Deploying App with Tomcat'
    }
  }
}
}

```

Declarative pipeline with Jenkins + Git + Maven + Tomcat

---> SSH Agent Configuration

It is used to establish remote SSH connection from one server (Linux VM) to another server (Linux VM)

Ex: Jenkins server is getting connected to Tomcat server to copy WAR file

Install SSH Agent Plugin --> Manage Jenkins --> Plugins --> Available Plugins --> SSH Agent --> Install

```

pipeline {
  agent any

  tools {
    maven "maven-3.9.10"
  }

  stages {
    stage('git clone') {
      steps {
        git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'
      }
    }
    stage('maven build') {
      steps {
        sh 'mvn clean compile test package'
      }
    }
    stage('App deployment') {
      steps {
        sshagent(['Tomcat-Server-Credentials-Pipeline']) {
          sh 'scp -o StrictHostKeyChecking=no target/FirstSpringWebApp-0.0.1-SNAPSHOT.war ec2-
user@15.156.94.232:/home/ec2-user/apache-tomcat-11.0.8/webapps'
        }
      }
    }
  }
}

```

Parallel stages:

Some stages I want to execute at the same time. Few stages I want to execute parallelly

```

pipeline {
  agent any

  tools {
    maven "maven-3.9.10"
  }

  stages {
    stage('git clone') {
      steps {
        git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'
      }
    }
    stage('maven build') {
      steps {
        sh 'mvn clean compile test package'
      }
    }
    stage('parallel stage') {
      parallel {
        stage('code-review'){
          steps {
            echo 'code review'
          }
        }

        stage('nexus-upload'){
          steps {
            echo 'nexus upload'
          }
        }
      }
    }
    stage('app deployed') {
      steps {
        echo 'Deploying App with Tomcat'
      }
    }
  }
}

```

Shared libraries in Jenkins

Say we want to deploy a microservice application, there are 5 microservices, 5 RestAPIs, maven command will be the same

The code review code will be the same

Instead of writing the same logic again and again, we will reuse the same code

--> Whenever there are multiple pipelines of a project, there might be some common logics in all pipelines. Ex: Artifact upload using Nexus, Code review, Maven build

Instead of we writing this same logics in all our pipelines, we can write this logic at one place and reuse it whenever we need it

--> To achieve this pipeline logic re-usability we can go with the concept of Shared libraries

--> We use Groovy scripting to create shared libraries

```
def call()
{
    sh "mvn clean package"
}
https://github.com/Haider7214/shared-lib
https://github.com/Haider7214/shared-lib.git
```

Jenkins Pipeline with Shared library

```
pipeline {
    agent any

    tools {
        maven "maven-3.9.10"
    }

    stages {
        stage('git clone') {
            steps {
                git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'
            }
        }
        stage('maven build') {
            steps {
                mavenBuild()
            }
        }
        stage('parallel stage') {
            parallel {
                stage('code-review'){
                    steps {
                        echo 'code review'
                    }
                }

                stage('nexus-upload'){
                    steps {
                        echo 'nexus upload'
                    }
                }
            }
        }
        stage('app deployed') {
            steps {
                echo 'Deploying App with Tomcat'
            }
        }
    }
}
```

```
@Library('demo_shared_lib')_
pipeline {
    agent any
```

```

tools {
    maven "maven-3.9.10"
}

stages {
    stage('git clone') {
        steps {
            git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'
        }
    }
    stage('demo message') {
        steps {
            telusko()
        }
    }
    stage('maven build') {
        steps {
            mavenBuild()
        }
    }
    stage('parallel stage') {
        parallel {
            stage('code-review'){
                steps {
                    echo 'code review'
                }
            }

            stage('nexus-upload'){
                steps {
                    echo 'nexus upload'
                }
            }
        }
    }
    stage('app deployed') {
        steps {
            echo 'Deploying App with Tomcat'
        }
    }
}
}

```

mavenBuild.groovy

```

def call()
{
    sh "mvn clean package"
}

```

2. Scripted pipeline (Groovy)

Multi branch pipeline in Jenkins

SonarQube
Nexus
CI/CD pipeline projects

22.00

Declarative pipeline

Start JenkinsServer and Slave

<input type="checkbox"/>	JenkinsSlave	i-01f4ee18204bd61fd	Running	t2.micro	–
<input type="checkbox"/>	EKS-host	i-01289fc5ca918b25f	Stopped	t2.micro	–
<input checked="" type="checkbox"/>	Jenkins-server	i-0abe9191466188c59	Running	t2.medium	Initializing

Open

<http://99.79.39.225:8080/manage/configure>

What we have here is different from public IP

→ ↻ 🏠 ⚠ Not secure 99.79.39.225:8080/manage/configure

🌐 New Tab 🌐 🌐 Move or copy cells... 📧 Email Finder hunter.io 🛡 Heroicons 👤 loghmanb/daily-co...

Dashboard > Manage Jenkins > System >

5

SCM checkout retry count

0

☐ Restrict project naming

Jenkins Location

Jenkins URL ?

http://35.183.77.174:8080/

System Admin e-mail address ?

address not configured yet <nobody@nowhere>

I update Jenkins URL

<http://99.79.39.225:8080/>

99.79.39.225:8080/manage/configure

Manage Jenkins > System >

5

SCM checkout retry count

0

☐ Restrict project naming

Jenkins Location

Jenkins URL ?

http://99.79.39.225:8080/

System Admin e-mail address ?

address not configured yet <nobody@nowhere>

Apply and Save

Go to Nodes

Nodes >

Nodes

S	Name ↓	Architecture	Clock Difference
	Built-In Node	Linux (amd64)	In sync
	jenkins-slave1		1 min
Data obtained		21 min	21 min

0/2
(launched...)

Icon: S M L

Configure jenkins-slave1

Launch agents via SSH

Host ?

ec2-99-79-127-107.ca-central-1.compute.amazonaws.com

Credentials ?

ec2-user (jenkins-slave-credentials)

+ Add

Update Host
Apply and Save
New Item +

New Item

New Item

Enter an item name

first-pipeline

Select an item type



Freestyle project

Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.



Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.



Multi-configuration project

Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Select Pipeline , click Ok
Pipeline script, Hello World

Definition

Pipeline script


Script ?

```
1 pipeline {  
2   agent any  
3  
4   stages {  
5     stage('Hello') {  
6       steps {  
7         echo 'Hello World'  
8       }  
9     }  
10  }  
11 }  
12
```

Hello World


☒ Use Groovy Sandbox ?


Apply and Save

 Status

first-pipeline

Demo First Pipeline

 Changes

 Build Now

 Configure

 Delete Pipeline

 Stages

 Rename

 Pipeline Syntax

Permalinks

Click Build Now

Console Output

Only one stage is there (Hello World)



Console Output

```
Started by user demo
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/first-pipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Hello)
[Pipeline] echo
Hello World
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

New Item

Enter an item name

Select an item type



Freestyle project

Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.



Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.



Multi-configuration project

Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.



Folder

Creates a container that stores related items in it. Useful for grouping things together. Unlike view, which is just a filter on

```
pipeline {
  agent any

  stages {
    stage('git clone') {
      steps {
        echo 'Cloning Git repo'
      }
    }
    stage('maven build') {
      steps {
        echo 'Project build with Maven'
      }
    }
    stage('deploy') {
      steps {
        echo 'Deploying App with Tomcat'
      }
    }
  }
}
```

Apply and Save
Build Now

✓ Console Output

```
Started by user demo
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/second-pipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (git clone)
[Pipeline] echo
Cloning Git repo
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (maven build)
[Pipeline] echo
Project build with Maven
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (deploy)
[Pipeline] echo
Deploying App with Tomcat
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Pipeline Overview

✓ #1

⚙ Manually run by demo ⌚ Started 7 min 53 sec ago 📦 Queued 3 ms ⌚ Took 1 sec

Graph



🔍 Search

✓ git clone 0.13s

✓ maven build 81ms

✓ deploy 49ms

✓ deploy 49ms ⌚

✓ Deploying App with Tomcat

⊞ Deploying App with Tomcat

Go back to Configure

```

13         }
14     }
15     stage('deploy') {
16         steps {
17             echo 'Deploying App with Tomcat'
18         }
19     }
20 }
21 }
22

```

☒ Use Groovy Sandbox ?

[Pipeline Syntax](#)

There is something called as Pipeline Syntax

.....

Steps

Sample Step

archiveArtifacts: Archive the artifacts

archiveArtifacts ?

Files to archive ?

Advanced ▼

Generate Pipeline Script

Steps

Sample Step

git: Git

git ?

Repository URL ?

`https://github.com/Haider7214/SpringApp.git`

Branch ?

`main`

Credentials ?

- none -

+ Add

☒ Include in polling? ?

☒ Include in changelog? ?

Generate Pipeline Script

Click Generate Pipeline Script

☒ Include in changelog: ?

Generate Pipeline Script

`git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'`

`git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'`

We need not know the syntax

Go back to Pipeline script

Instead of echo add the generated script

Pipeline script

Script ?

```

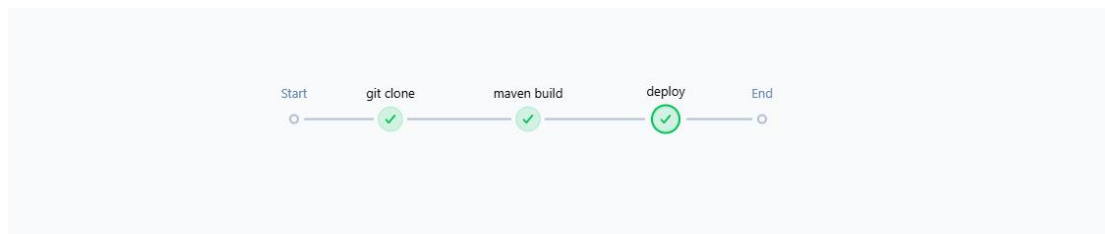
1 pipeline {
2   agent any
3
4   stages {
5     stage('git clone') {
6       steps {
7         git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'
8       }
9     }
10    stage('maven build') {
11      steps {
12        echo 'Project build with Maven'
13      }
14    }
15    stage('deploy') {

```

☒ Use Groovy Sandbox ?

[Pipeline Syntax](#)

Apply and Save
Build Now



It is able to clone the entire project

```

[Pipeline] stage
[Pipeline] { (git clone)
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/Haider7214/SpringApp.git
> git init /var/lib/jenkins/workspace/second-pipeline # timeout=10
Fetching upstream changes from https://github.com/Haider7214/SpringApp.git
> git --version # timeout=10
> git --version # 'git version 2.43.0'
> git fetch --tags --force --progress -- https://github.com/Haider7214/SpringApp.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git config remote.origin.url https://github.com/Haider7214/SpringApp.git # timeout=10
> git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision 7b1e402d1b8f6981c88b2aa6832ce94bfc21d974 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f 7b1e402d1b8f6981c88b2aa6832ce94bfc21d974 # timeout=10
> git branch --set-upstream-to=refs/remotes/origin/main # timeout=10

```

Shell script --> Generate Pipeline Script

Steps

Sample Step

sh: Shell Script

sh

Shell Script ?

mvn clean compile test package

Advanced ▾

Generate Pipeline Script

sh 'mvn clean compile test package'

sh 'mvn clean compile test package'

Update Pipeline script

Script ?

```
1 pipeline {  
2   agent any  
3  
4   stages {  
5     stage('git clone') {  
6       steps {  
7         git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'  
8       }  
9     }  
10    stage('maven build') {  
11      steps {  
12        sh 'mvn clean compile test package'  
13      }  
14    }  
15    stage('deploy') {
```

☒ Use Groovy Sandbox ?

[Pipeline Syntax](#)

Apply and Save

Build Now

It failed because of mvn: not found

```
[Pipeline] { (maven build)
[Pipeline] sh
+ mvn clean compile test package
/var/lib/jenkins/workspace/second-pipeline@tmp/durable-b10d60b6/script.sh.copy: 1: mvn: not found
[Pipeline] }
[Pipeline] // stage
```

```
tools {
  maven "maven.3.9"
}
```

script :

```
1 pipeline {
2   agent any
3
4   tools {
5     maven "maven.3.9"
6   }
7
8   stages {
9     stage('git clone') {
10      steps {
11        git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'
12      }
13    }
14    stage('maven build') {
15      steps {
```

Apply and Save

Manage Jenkins --> Tools

Use the same name here (maven-3.9.10)

Maven installations ^

Edited

Add Maven

≡ **Maven**

Name

maven-3.9.10

☒ Install automatically ?

≡ **Install from Apache**

Version

3.9.10

Add Installer ▾

Add Maven

Save

Apply

Again second-pipeline, Configure

Definition

Pipeline script

Script ?

```
1 pipeline {
2     agent any
3
4     tools {
5         maven "maven-3.9.10"
6     }
7
8     stages {
9         stage('git clone') {
10             steps {
11                 git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'
12             }
13         }
14         stage('maven build') {
15             steps {
```

☒ Use Groovy Sandbox ?

[Pipeline Syntax](#)

Advanced

```
pipeline {
    agent any

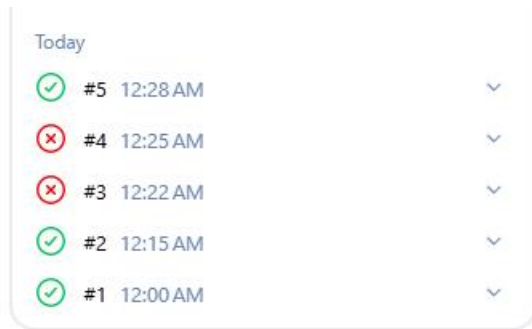
    tools {
        maven "maven-3.9.10"
    }

    stages {
        stage('git clone') {
            steps {
                git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'
            }
        }
        stage('maven build') {
            steps {
                sh 'mvn clean compile test package'
            }
        }
        stage('deploy') {
            steps {
                echo 'Deploying App with Tomcat'
            }
        }
    }
}
```

Apply and Save

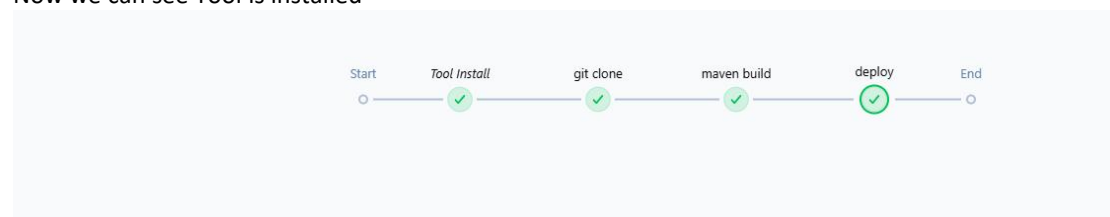
Build Now

Now Build is successful



```
[INFO] The original artifact has been renamed to /var/lib/jenkins/workspace/second-pipeline/target/firstSpringwe
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 12.128 s
[INFO] Finished at: 2025-07-02T00:29:08Z
[INFO] -----
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (deploy)
[Pipeline] tool
[Pipeline] envVarsForTool
[Pipeline] withEnv
[Pipeline] {
[Pipeline] echo
Deploying App with Tomcat
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
```

Now we can see Tool is installed



If one stage is not successful, it will not go to the next stage

Entire build and deployment process, the code works interlinked with each other

56:00

Restarted Jenkins VM

<http://16.52.71.170:8080/manage/configure>

Update slave VM configuration

Launch method ?

Launch agents via SSH

Host ?

ec2-35-182-253-224.ca-central-1.compute.amazonaws.com

Credentials ?

ec2-user (jenkins-slave-credentials)

+ Add

Declarative pipeline with Jenkins + Git + Maven + Tomcat

Jenkins Dashboard New Item

> New Item

New Item

Enter an item name

» This field cannot be empty, please enter a valid name

Select an item type



Freestyle project

Classic, general-purpose job type that checks out from up to one SCM, executes build steps like archiving artifacts and sending email notifications.



Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building workflows) and/or organizing complex activities that do not easily fit in free-style job type.



Multi-configuration project

Suitable for projects that need a large number of different configurations, such as testing or platform-specific builds, etc.



Folder

Creates a container that stores nested items in it. Useful for grouping things together. Unlike folder creates a separate namespace, so you can have multiple things of the same name as folders.

New Item

Enter an item name

git-maven-tomcat-pipeline

Select an item type



Freestyle project

Classic, general-purpose job type that checks out from up to one SCM, executes build steps like archiving artifacts and sending email notifications.



Pipeline

Orchestrates long-running activities that can span multiple build agents, jobs (or workflows) and/or organizing complex activities that do not easily fit in freestyle projects.



Multi-configuration project

Suitable for projects that need a large number of different configurations (e.g. platform-specific builds, etc.).



Multi-configuration project

Manage Jenkins --> Plugins

ssh agent			Install
Install	Name ↓		Released
<input checked="" type="checkbox"/>	SSH Agent 386.v36cc0c758210 This plugin allows you to provide SSH credentials to builds via a ssh-agent in Jenkins.		1 mo 20 days ago

Download progress

15

Preparation

- Checking internet connectivity
- Checking update center connectivity
- Success

SSH Agent

✓ Success

Loading plugin extensions

✓ Success

→ [Go back to the top page](#)

(you can start using the installed plugins right away)

→ ☐ Restart Jenkins when installation is complete and no jobs are running

Definition

Pipeline script

Script ?

```
11      git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'
12    }
13  }
14  stage('maven build') {
15    steps {
16      sh 'mvn clean compile test package'
17    }
18  }
19  stage('App deployment') {
20    steps {
21      echo 'Deploying App with Tomcat'
22    }
23  }
24 }
25 }
```

try sample Pipeline...

☒ Use Groovy Sandbox ?

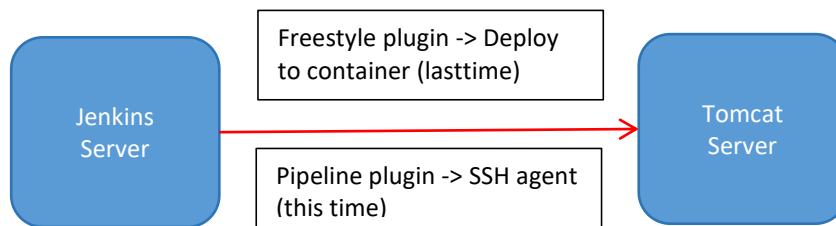
[Pipeline Syntax](#)

Pipeline Syntax

Turn ON Tomcat server, which I have installed in DevOpsCoursePractice

Find Instance by attribute or tag (case-sensitive)

	Name	Instance ID	Instance state	Instance
<input checked="" type="checkbox"/>	DevOpsCoursePractice	i-0f87d8181852ad376	Pending	t2.micro
<input type="checkbox"/>	TerraformEC2	i-0f3b562c215e434b7	Stopped	t2.micro
<input type="checkbox"/>	JenkinsSlave	i-01f4ee18204bd61fd	Running	t2.micro
<input type="checkbox"/>	EKS-host	i-01289fc5ca918b25f	Stopped	t2.micro
<input type="checkbox"/>	Jenkins-server	i-0abe9191466188c59	Running	t2.mediu



```

-rw-r--r--. 1 ec2-user ec2-user 363 Mar 2 00:54 sys-info.sh
[ec2-user@ip-172-31-13-80 ~]$ cd apache-tomcat-11.0.8/
[ec2-user@ip-172-31-13-80 apache-tomcat-11.0.8]$
[ec2-user@ip-172-31-13-80 apache-tomcat-11.0.8]$
[ec2-user@ip-172-31-13-80 apache-tomcat-11.0.8]$ ls -l
total 172
-rw-r-----. 1 ec2-user ec2-user 24262 Jun 5 17:49 BUILDING.txt
-rw-r-----. 1 ec2-user ec2-user 6166 Jun 5 17:49 CONTRIBUTING.md
-rw-r-----. 1 ec2-user ec2-user 60517 Jun 5 17:49 LICENSE
-rw-r-----. 1 ec2-user ec2-user 2333 Jun 5 17:49 NOTICE
-rw-r-----. 1 ec2-user ec2-user 3291 Jun 5 17:49 README.md
-rw-r-----. 1 ec2-user ec2-user 6469 Jun 5 17:49 RELEASE-NOTES
-rw-r-----. 1 ec2-user ec2-user 16109 Jun 5 17:49 RUNNING.txt
drwxr-x---. 2 ec2-user ec2-user 16384 Jun 22 18:30 bin
drwx-----. 3 ec2-user ec2-user 16384 Jun 22 20:20 conf
drwxr-x---. 2 ec2-user ec2-user 16384 Jun 22 18:30 lib
drwxr-x---. 2 ec2-user ec2-user 162 Jun 22 19:57 logs
drwxr-x---. 2 ec2-user ec2-user 30 Jun 22 18:30 temp
drwxr-x---. 8 ec2-user ec2-user 173 Jun 22 19:57 webapps
drwxr-x---. 3 ec2-user ec2-user 22 Jun 22 18:50 work
[ec2-user@ip-172-31-13-80 apache-tomcat-11.0.8]$
  
```

[ec2-user@ip-172-31-13-80 bin]\$ sh startup.sh

```

-rw-r-----. 1 ec2-user ec2-user 2026 Jun 5 17:49 version.bat
-rwxr-x---. 1 ec2-user ec2-user 1908 Jun 5 17:49 version.sh
[ec2-user@ip-172-31-13-80 bin]$ sh startup.sh
Using CATALINA_BASE: /home/ec2-user/apache-tomcat-11.0.8
Using CATALINA_HOME: /home/ec2-user/apache-tomcat-11.0.8
Using CATALINA_TMPDIR: /home/ec2-user/apache-tomcat-11.0.8/temp
Using JRE_HOME: /usr
Using CLASSPATH: /home/ec2-user/apache-tomcat-11.0.8/bin/bootstrap.jar:/home/ec2-user/apache-tomcat-11.0.8/bin/tomcat-jar.jar
Using CATALINA_OPTS:
Tomcat started.
[ec2-user@ip-172-31-13-80 bin]$
  
```

<http://15.156.94.232:8080/>


Not secure 15.156.94.232:8080

Move or copy cells... Email Finder hunter.io Heroicons loghmanb/daily-co... CloudSkew Excalidraw HackerR

Home Documentation Configuration Examples Wiki Mailing

Apache Tomcat/11.0.8

If you're seeing this, you've successfully installed Apache Tomcat.



Recommended Reading:

- [Security Considerations How-To](#)
- [Manager Application How-To](#)
- [Clustering/Session Replication How-To](#)

Jenkins Credentials Provider: Jenkins

Add Credentials

Domain

Global credentials (unrestricted)

Kind

SSH Username with private key

Scope ?

Global (Jenkins, nodes, items, all child items, etc)

ID ?

Tomcat-Server-Credentials-Pipeline

Description ?

Tomcat credentials

Username

ec2-user

☐ Treat username as secret ?

Copy Paste the Private Key

Click Add

Steps

Sample Step

sshagent: SSH Agent

sshagent ?

ec2-user (Tomcat credentials) ▼

?

+ Add

☐ Ignore missing credentials ?

Generate Pipeline Script

Click Generate Pipeline Script

```
sshagent(['Tomcat-Server-Credentials-Pipeline']) {  
  // some block  
}
```

Actions Projects Security Insights

SpringApp / src / main / java / com / telusko / FirstSpringWebApp /

Haider7214 first commit

Name	
..	
controller	f
model	f
FirstSpringWebAppApplication.java	f
ServletInitializer.java	f

<https://github.com/Haider7214/SpringApp/tree/main/src/main/java/com/telusko/FirstSpringWebApp>

```
sh 'scp -o StrictHostKeyChecking=no target/FirstSpringWebApp-0.0.1-SNAPSHOT.war'
```

```
[ec2-user@ip-172-31-13-80 bin]$ cd ..
[ec2-user@ip-172-31-13-80 apache-tomcat-11.0.8]$ cd webapps/
[ec2-user@ip-172-31-13-80 webapps]$
[ec2-user@ip-172-31-13-80 webapps]$ pwd
/home/ec2-user/apache-tomcat-11.0.8/webapps
[ec2-user@ip-172-31-13-80 webapps]$ █
```

Copy this path

/home/ec2-user/apache-tomcat-11.0.8/webapps

sh 'scp -o StrictHostKeyChecking=no target/FirstSpringWebApp-0.0.1-SNAPSHOT.war ec2-user@15.156.94.232:/home/ec2-user/apache-tomcat-11.0.8/webapps'

```
pipeline {
  agent any

  tools {
    maven "maven-3.9.10"
  }

  stages {
    stage('git clone') {
      steps {
        git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'
      }
    }
    stage('maven build') {
      steps {
        sh 'mvn clean compile test package'
      }
    }
    stage('App deployment') {
      steps {
        sshagent(['Tomcat-Server-Credentials-Pipeline']) {
          sh 'scp -o StrictHostKeyChecking=no target/FirstSpringWebApp-0.0.1-SNAPSHOT.war ec2-user@15.156.94.232:/home/ec2-user/apache-tomcat-11.0.8/webapps'
        }
      }
    }
  }
}
```

Apply and Save

Build Now

Status

</> Changes

Build Now

Configure

Delete Pipeline

Stages

Rename

?

Pipeline Syntax

git-maven-tomcat-pipeline

pipeline to deploy Application deployment with Tomcat

Permalinks

Builds

No builds

Today

✖

#1

2:27 AM

```
SSH_AGENT_PID=/03/
Running ssh-add (command line suppressed)
Identity added: /home/ec2-user/slavenode/workspace/git-maven-tomcat-pipeline@tmp/private_key_7406386655683991356.key (/home/ec2-user/slavenode/workspace/git-maven-tomcat-pipeline@tmp/private_key_7406386655683991356.key)
[ssh-agent] Started.
[Pipeline] {
[Pipeline] sh
+ scp -o StrictHostKeyChecking=no target/FirstSpringWebApp-0.0.1-SNAPSHOT.war ec2-user@15.156.94.232:/home/ec2-user/apache-tomcat-11.0.8/webapps
dest open("/home/ec2-user/apache-tomcat-11.0.8/webapps/FirstSpringWebApp-0.0.1-SNAPSHOT.war"): Failure
failed to upload file target/FirstSpringWebApp-0.0.1-SNAPSHOT.war to /home/ec2-user/apache-tomcat-11.0.8/webapps/FirstSpringWebApp-0.0.1-SNAPSHOT.war
[Pipeline] }
$ ssh-agent -k
```

It is there in the slavenode


```


drwxr-xr-x. 6 ec2-user ec2-user 80 Jul 3 02:28 slavenode
[ec2-user@ip-172-31-30-34 ~]$ cd
.cache/ .config/ .m2/ .ssh/ slavenode/
[ec2-user@ip-172-31-30-34 ~]$ cd slavenode/
[ec2-user@ip-172-31-30-34 slavenode]$ ls -l
total 1368
drwxrwxr-x. 3 ec2-user ec2-user 26 Jul 3 02:28 caches
drwxrwxr-x. 4 ec2-user ec2-user 34 Jun 27 02:58 remoting
-rw-rw-r--. 1 ec2-user ec2-user 1396936 Jun 27 02:58 remoting.jar
drwxrwxr-x. 3 ec2-user ec2-user 50 Jul 3 02:28 tools
drwxrwxr-x. 5 ec2-user ec2-user 96 Jul 3 02:28 workspace
[ec2-user@ip-172-31-30-34 slavenode]$ pwd
/home/ec2-user/slavenode
[ec2-user@ip-172-31-30-34 slavenode]$ cd workspace/
[ec2-user@ip-172-31-30-34 workspace]$ ls -l
total 0
drwxrwxr-x. 2 ec2-user ec2-user 23 Jun 27 03:23 DemoFirstJob
drwxrwxr-x. 6 ec2-user ec2-user 138 Jul 3 02:54 git-maven-tomcat-pipeline
drwxrwxr-x. 2 ec2-user ec2-user 6 Jul 3 02:54 git-maven-tomcat-pipeline@tmp
[ec2-user@ip-172-31-30-34 workspace]$ cd git-maven-tomcat-pipeline
[ec2-user@ip-172-31-30-34 git-maven-tomcat-pipeline]$ ls -l
total 40
-rw-rw-r--. 1 ec2-user ec2-user 10665 Jul 3 02:28 mvnw
-rw-rw-r--. 1 ec2-user ec2-user 7061 Jul 3 02:28 mvnw.cmd
-rw-rw-r--. 1 ec2-user ec2-user 1890 Jul 3 02:28 pom.xml
drwxrwxr-x. 4 ec2-user ec2-user 30 Jul 3 02:28 src
drwxrwxr-x. 10 ec2-user ec2-user 16384 Jul 3 02:54 target
[ec2-user@ip-172-31-30-34 git-maven-tomcat-pipeline]$ cd target/
[ec2-user@ip-172-31-30-34 target]$ ls -l
total 51884
drwxrwxr-x. 5 ec2-user ec2-user 49 Jul 3 02:54 FirstSpringWebApp-0.0.1-SNAPSHOT
-rw-rw-r--. 1 ec2-user ec2-user 28070167 Jul 3 02:54 FirstSpringWebApp-0.0.1-SNAPSHOT.war
-rw-rw-r--. 1 ec2-user ec2-user 25051845 Jul 3 02:54 FirstSpringWebApp-0.0.1-SNAPSHOT.war.original
drwxrwxr-x. 3 ec2-user ec2-user 47 Jul 3 02:54 classes
drwxrwxr-x. 3 ec2-user ec2-user 25 Jul 3 02:54 generated-sources
drwxrwxr-x. 3 ec2-user ec2-user 30 Jul 3 02:54 generated-test-sources
drwxrwxr-x. 2 ec2-user ec2-user 28 Jul 3 02:54 maven-archiver
drwxrwxr-x. 3 ec2-user ec2-user 35 Jul 3 02:54 maven-status
drwxrwxr-x. 2 ec2-user ec2-user 161 Jul 3 02:54 surefire-reports
drwxrwxr-x. 3 ec2-user ec2-user 17 Jul 3 02:54 test-classes
[ec2-user@ip-172-31-30-34 target]$


```

don't know why it is not copying into Tomcat-server

I deleted existing FirstSpringWebApp-0.0.1-SNAPSHOT.war


 Stages


 Rename


 Pipeline Syntax


Builds


Today

 #10 3:12 AM

 #9 3:09 AM

 #8 3:01 AM

 #7 2:54 AM

 #6 2:52 AM

```

pipeline {
    agent any

```

```

    tools {

```

```

    maven "maven-3.9.10"
  }

  stages {
    stage('git clone') {
      steps {
        git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'
      }
    }
    stage('maven build') {
      steps {
        sh 'mvn clean compile test package'
      }
    }
    stage('App deployment') {
      steps {
        sshagent(['Tomcat-Server-Credentials-Pipeline1']) {
          sh 'scp -o StrictHostKeyChecking=no target/FirstSpringWebApp-0.0.1-SNAPSHOT.war ec2-user@15.156.94.232:/home/ec2-user/apache-tomcat-11.0.8/webapps'
        }
      }
    }
  }
}

```

```

[Pipeline] sshagent
[ssh-agent] Using credentials ec2-user
$ ssh-agent
SSH_AUTH_SOCK=/tmp/ssh-XXXXXXQf7Pk7/agent.8289
SSH_AGENT_PID=8291
Running ssh-add (command line suppressed)
Identity added: /home/ec2-user/slavenode/workspace/git-maven-tomcat-pipeline@tmp/private_key_9013734709230507359.key (/home/ec2-user/s
pipeline@tmp/private_key_9013734709230507359.key)
[ssh-agent] Started.
[Pipeline] {
[Pipeline] sh
+ scp -o StrictHostKeyChecking=no target/FirstSpringWebApp-0.0.1-SNAPSHOT.war ec2-user@15.156.94.232:/home/ec2-user/apache-tomcat-11.0.
[Pipeline] }
$ ssh-agent -k
unset SSH_AUTH_SOCK;
unset SSH_AGENT_PID;
echo Agent pid 8291 killed;
[ssh-agent] Stopped.
[Pipeline] // sshagent
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS

```



<http://15.156.94.232:8080/manager/html>

Go to Tomcat ---> Manage Apps

<http://15.156.94.232:8080/FirstSpringWebApp-0.0.1-SNAPSHOT/register>

Not secure 15.156.94.232:8080/FirstSpringWebApp-0.0.1-SNAPSHOT/register

Move or copy cells... Email Finder hunter.io Heroicons loghmanb/daily-co... CloudSkew Excalidraw HackerRank/Java/St... Maven Repository...

Registration Application

Employee ID	<input type="text"/>
Employee Name	<input type="text"/>
Employee City	<input type="text" value="Bengaluru"/>
Employee Salary	<input type="text"/>

1:30

```
-rw-r----- 1 ubuntu ubuntu 2026 Jun 5 17:49 version.bat
-rwxr-x--- 1 ubuntu ubuntu 1908 Jun 5 17:49 version.sh
ubuntu@ip-172-31-11-116:~/apache-tomcat-11.0.8/bin$ sh startup.sh
Using CATALINA_BASE:   /home/ubuntu/apache-tomcat-11.0.8
Using CATALINA_HOME:   /home/ubuntu/apache-tomcat-11.0.8
Using CATALINA_TMPDIR: /home/ubuntu/apache-tomcat-11.0.8/temp
Using JRE_HOME:        /usr
Using CLASSPATH:       /home/ubuntu/apache-tomcat-11.0.8/bin/bootstrap.jar:/home
Using CATALINA_OPTS:
Tomcat started.
ubuntu@ip-172-31-11-116:~/apache-tomcat-11.0.8/bin$
```

Parallel Stages:

Create a new pipeline

New Item

Enter an item name

parallel-pipeline

Select an item type



Freestyle project

Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.



Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.



Multi-configuration project

Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Just get any of the script to add into this pipeline

```
pipeline {
  agent any
```

```
  tools {
```

```

    maven "maven-3.9.10"
  }

  stages {
    stage('git clone') {
      steps {
        git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'
      }
    }
    stage('maven build') {
      steps {
        sh 'mvn clean compile test package'
      }
    }
    stage('deploy') {
      steps {
        echo 'Deploying App with Tomcat'
      }
    }
  }
}

```

Modified script

```

pipeline {
  agent any

  tools {
    maven "maven-3.9.10"
  }

  stages {
    stage('git clone') {
      steps {
        git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'
      }
    }
    stage('maven build') {
      steps {
        sh 'mvn clean compile test package'
      }
    }
    stage('parallel stage') {
      parallel {
        stage('code-review'){
          steps {
            echo 'code review'
          }
        }

        stage('nexus-upload'){
          steps {
            echo 'nexus upload'
          }
        }
      }
    }
  }
}

```



```

stage('app deployed') {
    steps {
        echo 'Deploying App with Tomcat'
    }
}
}
}
}

```

Apply and Save

Build Now



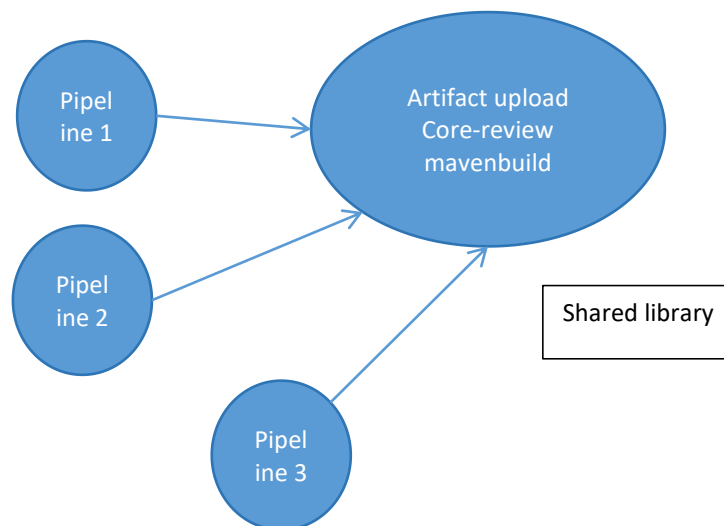
```

[INFO] Skipping execution of surefire because it has already been run for this configuration
[INFO] --- war:3.4.0:war (default-war) @ FirstSpringWebApp ---
[INFO] Packaging webapp
[INFO] Assembling webapp [FirstSpringWebApp] in [/var/lib/jenkins/workspace/parallel-pipeline/target/FirstSpringWebApp-0.0.1-SNAPSHOT]
[INFO] Processing war project
[INFO] Copying webapp resources [/var/lib/jenkins/workspace/parallel-pipeline/src/main/webapp]
[INFO] Building war: /var/lib/jenkins/workspace/parallel-pipeline/target/FirstSpringWebApp-0.0.1-SNAPSHOT.war
[INFO] --- spring-boot:3.3.5:repackage (repackage) @ FirstSpringWebApp ---
[INFO] Replacing main artifact /var/lib/jenkins/workspace/parallel-pipeline/target/FirstSpringWebApp-0.0.1-SNAPSHOT.war with repackaged archive, adding nested dependencies in BOOT-INF/.
[INFO] The original artifact has been renamed to /var/lib/jenkins/workspace/parallel-pipeline/target/FirstSpringWebApp-0.0.1-SNAPSHOT.war.original
[INFO] BUILD SUCCESS
[INFO]

```

Code review and nexus-upload execute parallelly

Shared library



How do we define Groovy syntax in Jenkins?

Lets say code-review code is the same, nexus-upload code is the same, why to write the same code again and again

Manage Jenkins --> System

New Item

Build History

Project Relationship

Check File Fingerprint

Manage Jenkins

My Views

Build Queue

No builds in the queue.

Build Executor Status

Built-In Node

0/2

jenkins-slave1

0/2

Manage Jenkins


New version of Jenkins (2.504.3) is available for [download](#) ([changelog](#)).


Building on the built-in node can be a security issue. You should set the number of executors on the built-in node to 0.

Jenkins URL is empty but is required for the proper operation of many Jenkins features like email notifications, etc. Please provide an accurate value in [Jenkins configuration](#).

The *Restrict project naming* configuration is not set to the *Role-based Strategy*. This can lead to problems as it allows the creation of projects with arbitrary names.

System Configuration

 **System**
Configure global settings and paths.

 **Tools**
Configure tools, their locations and automatic installers.

Global Trusted Pipeline Libraries

GitHub Enterprise Servers

Add

Global Trusted Pipeline Libraries

Sharable libraries available to any Pipeline jobs running on this system. These libraries will be trusted, meaning they run without "sandbox" restrictions and may use @Grab.

Add


Global Untrusted Pipeline Libraries

Sharable libraries available to any Pipeline jobs running on this system. These libraries will be untrusted, meaning they run with "sandbox" restrictions and cannot use @Grab.

Add

Github groovy scripts are there

shared-lib / vars /

 **Haider7214** first commit

Name

..

mavenBuild.groovy

telusko.groovy

Global Trusted Pipeline Libraries

Sharable libraries available to any Pipeline jobs running on this system. These libraries will be trusted, mean

≡

Library

Name ?

demo_shared_lib

Default version ?

main

Cannot validate default version until after saving and reconfiguring.

☐ Load implicitly ?

☒ Allow default version to be overridden ?

☒ Include @Library changes in job recent changes ?

☐ Cache fetched versions on controller for quick retrieval ?

Retrieval method

Modern SCM

Credentials are not needed because it is a Public repo

Retrieval method

Modern SCM

Loads a library from an SCM plugin using newer interfaces optimized for this purpose. The recommended option when

Source Code Management

Git

Project Repository ?

https://github.com/Haider7214/shared-lib.git

Credentials ?

- none -

+ Add

Behaviors

Discover branches

?

Add ▾

☐ Fresh clone per build ?

Apply and Save

New Item

Enter an item name

shared-lib-pipeline

Select an item type



Freestyle project

Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.



Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.



Multi-configuration project

Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.



Folder

```
@Library('demo_shared_lib')
pipeline {
    agent any
```

```

tools {
    maven "maven-3.9.10"
}

stages {
    stage('git clone') {
        steps {
            git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'
        }
    }
    stage('demo message') {
        steps {
            telusko()
        }
    }
    stage('maven build') {
        steps {
            mavenBuild()
        }
    }
    stage('parallel stage') {
        parallel {
            stage('code-review'){
                steps {
                    echo 'code review'
                }
            }

            stage('nexus-upload'){
                steps {
                    echo 'nexus upload'
                }
            }
        }
    }
    stage('app deployed') {
        steps {
            echo 'Deploying App with Tomcat'
        }
    }
}
}

```

Apply and Save

Build Now

It failed

Console Output

```
Started by user demo
org.codehaus.groovy.control.MultipleCompilationErrorsException: startup failed:
WorkflowScript: 2: unexpected token: pipeline @ line 2, column 1.
    pipeline {
    ^

1 error

at org.codehaus.groovy.control.ErrorCollector.failIfErrors(ErrorCollector.java:32)
at org.codehaus.groovy.control.ErrorCollector.addFatalError(ErrorCollector.java:174)
at org.codehaus.groovy.control.ErrorCollector.addError(ErrorCollector.java:144)
```

Added an _

@Library('demo_shared_lib')_

```
pipeline {
    agent any

    tools {
        maven "maven-3.9.10"
    }

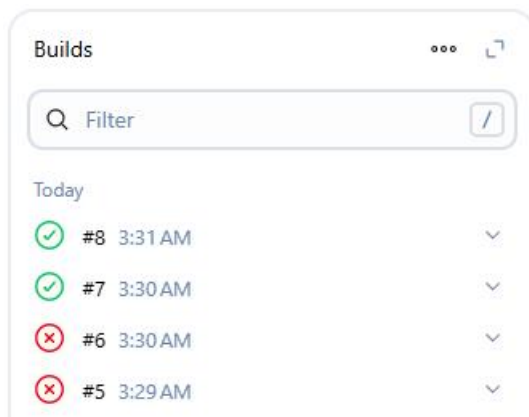
    stages {
        stage('git clone') {
            steps {
                git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'
            }
        }
        stage('demo message') {
            steps {
                telusko()
            }
        }
        stage('maven build') {
            steps {
                mavenBuild()
            }
        }
        stage('parallel stage') {
            parallel {
                stage('code-review'){
                    steps {
                        echo 'code review'
                    }
                }

                stage('nexus-upload'){
                    steps {
                        echo 'nexus upload'
                    }
                }
            }
        }
    }
}
```

```

    }
    stage('app deployed') {
        steps {
            echo 'Deploying App with Tomcat'
        }
    }
}
}
}

```



Function runs fine

```

[Pipeline] withEnv
[Pipeline] {
[Pipeline] echo
Welcome to Shared LIBRARY concept
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (maven build)

```

Pipeline overview

