

# How to deploy the Spring Boot application with the help of Jenkins CI/CD. - Clouds Baba



# Spring Boot®

The **Spring Framework** is an application framework and inversion of control container for the Java platform. The framework's core features can be used by any Java application, but there are extensions for building web applications on top of the Java EE (Enterprise Edition) platform. Although the framework does not impose any specific programming model, it has become popular in the Java community as an addition to the Enterprise JavaBeans (EJB) model. The Spring Framework is open-source.

Here we are going to discuss the way of generation .jar file out of the spring boot project with Jenkins from a remote repository and copy the .jar file into another remote location by Jenkins and deploying it.

## Pre-requisites

Install Jenkins and Java

Install Apache-Maven on the Jenkins server and setting up the path

Source Code repository from Github/Bitbucket

## Step 1:- Install Jenkins and Java

For Jenkins and Java installation please refer to this post:-

[Jenkins Installation And Configuration On AWS EC2-instance](#)

## Step 2:- Install apache maven on the Jenkins server and setting up the path

for installing Apache-Maven make sure that java is installed on the server

Go to the [official Apache Maven download](#) page and download the latest version of it

## Files

Maven is distributed in several formats for your convenience. Simply pick a ready-made binary distribution archive and follow the [installation instructions](#). Use a source archive if you intend to build Maven yourself.

In order to guard against corrupted downloads/installations, it is highly recommended to [verify the signature](#) of the release bundles against the public [KEYS](#) used by the Apache Maven developers.

	Link	Checksums	Signature
Binary tar.gz archive	<a href="#">apache-maven-3.6.3-bin.tar.gz</a>	<a href="#">apache-maven-3.6.3-bin.tar.gz.sha512</a>	<a href="#">apache-maven-3.6.3-bin.tar.gz.asc</a>

Binary zip archive	<a href="#">apache-maven-3.6.3-bin.zip</a>	<a href="#">apache-maven-3.6.3-bin.zip.sha512</a>	<a href="#">apache-maven-3.6.3-bin.zip.asc</a>
Source tar.gz archive	<a href="#">apache-maven-3.6.3-src.tar.gz</a>	<a href="#">apache-maven-3.6.3-src.tar.gz.sha512</a>	<a href="#">apache-maven-3.6.3-src.tar.gz.asc</a>
Source zip archive	<a href="#">apache-maven-3.6.3-src.zip</a>	<a href="#">apache-maven-3.6.3-src.zip.sha512</a>	<a href="#">apache-maven-3.6.3-src.zip.asc</a>

login to the Jenkins server and go to the directory where you want to install Apache-Maven

```
cd /u01
curlhttps://mirrors.estointernet.in/apache/maven/maven3/3.6.3/binaries/apache-maven-3.6.3-
bin.tar.gz
```

Extract the download file

```
sudo tar -xf apache-maven-3.6.3-bin.tar.gz
```

```
drwxr-xr-x  6 root  root  4096 Aug 30 06:14 apache-maven-3.6.3/
```

Setting up the path

```
sudo vi /etc/profile.d/maven.sh
```

```
export M2_HOME=/u01apache-maven-3.6.3
export PATH=${M2_HOME}/bin:${PATH}
```

reload the file

```
source /etc/profile.d/maven.sh
```

check if the path is set or not

```
echo $M2_HOME
```

```
jenkins01:~$ echo $M2_HOME
/u01/apache-maven-3.6.3
```

### Step 3:- Setting up the deployment of spring boot with Jenkins

Login to Jenkins Server

Installed the **Maven**, **Git**, **Bitbucket** plugins from the plugin manager

After installing go to Manage Jenkins>Global Tool Configuration

Set the Apache-Maven path

#### Maven

Maven installations

Add Maven

Maven

Name

M2\_HOME

MAVEN\_HOME

/u01/apache-maven-3.6.3

☐ Install automatically



Delete Maven

Add Maven

List of Maven installations on this system

save it


Create New Item

Click on New Item




# Jenkins

Jenkins ▶

 New Item

 People

 Build History

 Manage Jenkins

 My Views

 Lockable Resources

 New View

### Build Queue

No builds in the queue.

### Build Executor Status

1 Idle

2 Idle

54.81.237.145:8080/computer/new

Select Maven Project and type the name of the project

## Enter an item name

» Required field



### Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.



### Maven project

Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.



### 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

Builds projects that need a large number of different configurations, such as testing on multiple environments, platform-specific

OK

click ok

it will open a configuration window for you

In the Source Code Management tab add the git repository URL

## Source Code Management

☐ None  
☒ Git

Repositories

Repository URL

Credentials

**Note-** If your repository is private then you need to add credentials for authenticating it to your repository

Navigate to Build Environment add the **pre** and **post-build** scripts like below:

### Build Environment

☐ Provide Configuration files  
☐ Send files or execute commands over SSH before the build starts  
☐ Send files or execute commands over SSH after the build runs  
☒ Execute shell script on remote host using ssh

SSH site

Pre build script

```

TIMESTAMP=$(date +"%F-%T")
BACKUP_DIR="/home/centos/deployment/target/backup/$TIMESTAMP"
sudo mkdir -p $BACKUP_DIR
tomcat_pid() {
echo `ps -ef | grep java | grep -v grep | awk '{ print $2 }'`
}

```

Post build script

```

cd /home/centos/deployment/target/
sudo nohup java -jar testing_app_0_1.jar > service.out 2> errors.txt < /dev/null &

```

Pre-build script:-

```

TIMESTAMP=$(date +"%F-%T")
BACKUP_DIR="/home/centos/deployment/target/backup/$TIMESTAMP"
sudo mkdir -p $BACKUP_DIR
tomcat_pid() {
echo `ps -ef | grep java | grep -v grep | awk '{ print $2 }'`
}
pid=$(tomcat_pid)
sudo kill -9 $pid
sudo mv /home/centos/deployment/target/*.jar $BACKUP_DIR

```

Post-build script:-

```

cd /home/centos/deployment/target/
sudo nohup java -jar testing_app_0_1.jar > service.out 2> errors.txt < /dev/null &

```

Now add **clean install package -DskipTests=true** in Goals and options

### Build

Post-Build

Root POM pom.xml

Goals and options clean install package -DskipTests=true

Advanced...

Now add Post Steps

### Post Steps

☒ Run only if build succeeds ☐ Run only if build succeeds or is unstable ☐ Run regardless of build result

Should the post-build steps run only for successful builds, etc.

Add post-build step ^

- Execute NodeJS script
- Execute Windows batch command
- Execute shell
- Execute shell script on remote host using ssh
- Invoke top-level Maven targets
- Provide Configuration files
- Send files or execute commands over SSH

Select “Send files or execute commands over SSH”

Add the remote server where you want to deploy the .jar

### SSH Publishers

#### SSH Server

Name testing-node

Advanced...

#### Transfers

##### Transfer Set

Source files target/\*.jar

Remove prefix

Remote directory deployment

Exec command

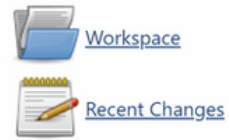
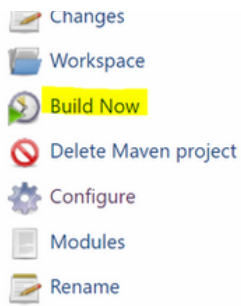
**Either Source files, Exec command or both must be supplied**

All of the transfer fields (except for Exec timeout) support substitution of [Jenkins environment variables](#)

Save Apply

Click Apply and Save

Click on *Build now*



## Permalinks

check the console output

If it is showing success at the last then your configuration is correct. If its showing failure then you need to check the configuration again

Congratulations.... You follow all the steps and deployed spring boot application to the remote server

Reference:-

[https://www.tutorialspoint.com/spring\\_boot/](https://www.tutorialspoint.com/spring_boot/)

[spring\\_boot\\_introduction.htm#:~:text=Spring%20Boot%20is%20an%20open,and%20production%20ready%20spring%20applications.](https://www.tutorialspoint.com/spring_boot/introduction.htm#:~:text=Spring%20Boot%20is%20an%20open,and%20production%20ready%20spring%20applications.)

<https://www.jenkins.io/>

<https://maven.apache.org/>