

### JENKINS

Jenkins is an open source automation tool written in Java that allows continuous integration and continuous delivery of projects.

#### JENKINS INSTALLATION

##### ON WINDOWS:

1. Install Java Development Kit (JDK)
2. Set the Path for the Environmental Variable for JDK.
3. Download and Install Jenkins.
4. Run Jenkins on Localhost 8080.

##### ON UBUNTU:

1. Install Java (\$sudo apt install openjdk-8-jdk or 11-jdk).
2. Add the Jenkins repository.
3. Install Jenkins (\$sudo apt update, \$sudo apt install Jenkins).
4. Set up Jenkins <http://localhost:8080>

#### JOB TYPES

1. **Freestyle build jobs** are general-purpose build jobs, which provides maximum flexibility. It can be used for any type of project.
2. **Pipeline** runs the entire software development workflow as code.
3. **Multiconfiguration project** allows us to run the same build job on different environments.
4. **Folder** allows users to create folders to organize and categorize similar jobs.
5. **GitHub Organization** scans your entire GitHub organization and creates Pipeline jobs for each repository containing a Jenkinsfile.
6. **Multibranch Pipeline** lets us implement different Jenkinsfiles for different branches of the same project.
7. **Maven** builds a maven project. Jenkins takes advantage of our POM files for packaging.

#### JENKINS PIPELINE

**Jenkins pipeline** is a single platform that runs the entire pipeline as code instead of building several jobs for each phase and it puts it in a Jenkinsfile. Jenkinsfile is a text file that stores the pipeline as code. It is written using the Groovy DSL. It can be written based on two syntaxes:

**1. Scripted pipeline:** Code is written on the Jenkins UI instance and is enclosed within the node block.

```
node {
  scripted pipeline code}
```

**2. Declarative pipeline:** Code is written locally in a file and is checked into a SCM and is enclosed within the pipeline block.

```
node {
  declarative pipeline code}
```

#### BUILD PIPELINE

Build Pipeline can be used to chain several jobs together and run them in a sequence.

##### BUILD PIPELINE EXAMPLE:

1. Create 3 freestyle Jobs.
2. Chain the 3 jobs together.

*Job1 ->configure ->Post Build ->Build other projects ->Job2*

*Job2 ->configure ->Post Build ->Build other projects ->Job3*

3. Create a build pipeline view.

*Jenkins Dashboard ->Add view ->Enter a name ->Build pipeline view ->ok ->configure ->Pipeline flow ->Select Initial job ->Job1 ->ok*

4. Run the Build Pipeline.

#### JENKINS PLUGINS

Jenkins comes with over 2000 plugins and each plugin has a unique functionality. But when it comes to software development most developers use a set of plugins, such as, **Maven, Git, Ant, Docker, Copy artefact**, etc. Follow the below step to install the above plugins or any other Jenkins plugin.

**Jenkins Dashboard-> Manage Jenkins-> Manage Plugins-> Available**

#### MAVEN PROJECT

1. Build Step where we can write goals and options. We can write different maven commands here such as **CLEAN, INSTALL, COMPILE, TEST, PACKAGE**, etc. that affect the **POM.xml** file present on our system or GitHub Repository.
2. We can also deploy **war/ear** to a container using post-build actions and archive the same.
3. We can execute batch or shell commands as a part of pre-build steps.

#### AUTOMATED DEPLOYMENT

1. Head to the manage plugins and install the respective plugins
2. It takes the war/ear file and deploys that to the running remote application build.
3. Go to build and configure and click on 'deploy to war/ear to container'
4. In the war container section save details about the destination server and click save.

##### EXAMPLE:

Open a freestyle project. Paste the URL of GitHub repository where POM.xml file is stored in SCM. Add a maven command (**MVN CLEAN INSTALL**). This will give us a war/jar file which we can deploy onto TOMCAT.



### MANAGE USERS & ROLES

Jenkins allows us to give different permissions to a user across different jobs using "**Project Based Matrix Authorization strategy**". To use Project-based Matrix Authorization Strategy, First login with Admin user go to *Manage Jenkins -> Configure Global Security*. In "Authorization", Select "**Project-based Matrix Authorization Strategy**". Then add "Admin" user and check all the checkbox to grant all permission to admin user.



By **azka123**  
[cheatography.com/azka123/](https://cheatography.com/azka123/)

Not published yet.  
Last updated 4th March, 2022.  
Page 3 of 2.

Sponsored by **Readable.com**  
Measure your website readability!  
<https://readable.com>