**Jenkins Pipeline: JenkinsFile EXAMPLE**

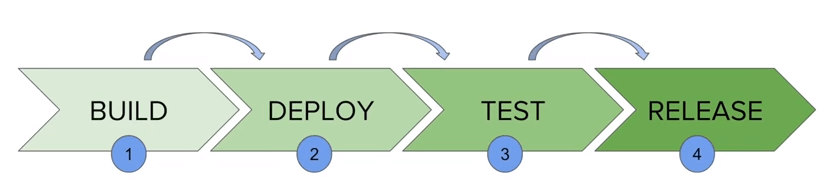
## What is Jenkins Pipeline?

In Jenkins, a pipeline is a group of events or jobs which are interlinked with one another in a sequence.

In simple words, Jenkins Pipeline is a combination of plugins that support the integration and implementation of **continuous delivery pipelines**using Jenkins. A pipeline has an extensible automation server for creating simple or complex delivery pipelines "as code," via pipeline DSL (Domain-specific Language).

### What is Continuous Delivery Pipelines? How it Works?

In a Jenkins pipeline, every job or event has some sort of dependency on at least one or more events.

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe1.png)

The picture above represents a continuous delivery pipeline in Jenkins. It contains a group of states called build, deploy, test and release. These events are interlinked with each other. Every state has its events, which work in a sequence called a continuous delivery pipeline.

A continuous delivery pipeline is an automated expression to display your process for getting software for version control. Thus, every change made in your software goes through a number of complex processes on its way to being released. It also involves developing the software in a reliable and repeatable manner, and progression of the built software through multiple stages of testing and deployment.

In this tutorial, you will learn

## What is a JenkinsFile?

Jenkins pipelines can be defined using a text file called **JenkinsFile.**You can implement pipeline as code using JenkinsFile, and this can be defined by using a domain specific language (DSL). With JenkinsFile, you can write the steps needed for running a Jenkins pipeline.

The benefits of using J**enkinsFile are**:

* You can create pipelines automatically for all branches and execute pull requests with just one **JenkinsFile.**
* You can review your code on the pipeline
* You can audit your Jenkins pipeline
* This is the singular source for your pipeline and can be modified by multiple users.

JenkinsFile can be defined by either Web UI or with a JenkinsFile.

**Declarative versus Scripted pipeline syntax:**

There are two types of syntax used for defining your JenkinsFile.

1. Declarative
2. Scripted

**Declarative:**

Declarative pipeline syntax offers an easy way to create pipelines. It contains a predefined hierarchy to create Jenkins pipelines. It gives you the ability to control all aspects of a pipeline execution in a simple, straight-forward manner.

**Scripted:**

Scripted Jenkins pipeline runs on the Jenkins master with the help of a lightweight executor. It uses very few resources to translate the pipeline into atomic commands. Both declarative and scripted syntax are different from each other and are defined totally differently.

## Why Use Jenkin's Pipeline?

Jenkins is an open continuous integration server which has the ability to support the automation of software development processes. You can create multiple automation jobs with the help of use cases, and run them as a Jenkins pipeline.

Here are the reasons why you use should use Jenkins pipeline:

* Jenkins pipeline is implemented as a code which allows multiple users to edit and execute the pipeline process.
* Pipelines are robust. So if your server undergoes an unforeseen restart, the pipeline will be automatically resumed.
* You can pause the pipeline process and make it wait to resume until there is an input from the user.
* Jenkins Pipelines support big projects. You can run multiple jobs, and even use pipelines in a loop.

## Jenkins Pipeline Concepts

|  |  |
| --- | --- |
| **Term** | **Description** |
| Pipeline | The pipeline is a set of instructions given in the form of code for continuous delivery and consists of instructions needed for the entire build process. With pipeline, you can build, test, and deliver the application. |
| Node | The machine on which Jenkins runs is called a node. A node block is mainly used in scripted pipeline syntax. |
| Stage | A stage block contains a series of steps in a pipeline. That is, the build, test, and deploy processes all come together in a stage. Generally, a stage block is used to visualize the Jenkins pipeline process. |
| Step | A step is nothing but a single task that executes a specific process at a defined time. A pipeline involves a series of steps. |

Creating a Jenkinsfile

A Jenkinsfile is a text file that contains the definition of a Jenkins Pipeline and is checked into source control. Consider the following Pipeline which implements a basic three-stage continuous delivery pipeline.

Jenkinsfile (Declarative Pipeline)

pipeline {

agent any

stages {

stage('Build') {

steps {

echo 'Building..'

}

}

stage('Test') {

steps {

echo 'Testing..'

}

}

stage('Deploy') {

steps {

echo 'Deploying....'

}

}

}

}

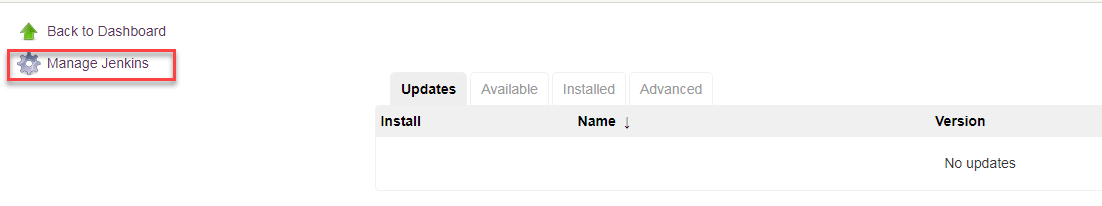
Not all Pipelines will have these same three stages, but it is a good starting point to define them for most projects. The sections below will demonstrate the creation and execution of a simple Pipeline in a test installation of Jenkins.

## Install Build Pipeline Plugin in Jenkins

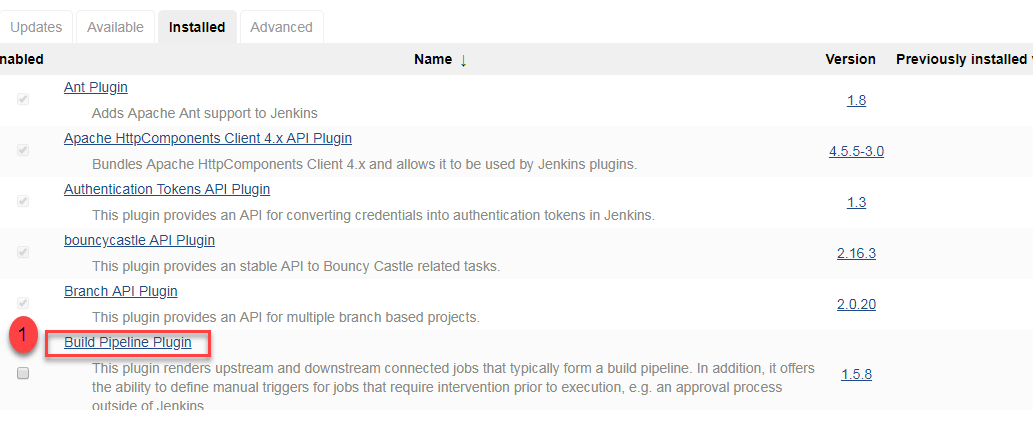
With the **build pipeline** plugin, you can create a pipeline view of incoming and outgoing jobs, and create triggers which require manual intervention.

Here is how you can install the **build pipeline**plugin in your Jenkins:

**Step 1**) The settings for the plugin can be found under **Manage Jenkins > Manage Plugins.**

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe2.png)

If you have already installed the plugin, it is shown under the installed tab.

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe3.png)

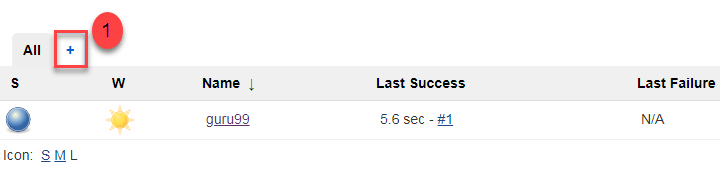
**Step 2**) If you do not have the plugin previously installed, it shows up under the **Available**tab.

Once you have successfully installed the **build pipeline** plugin in your Jenkins, follow these steps to create your Jenkins pipeline:

## How to Create Jenkins Pipeline

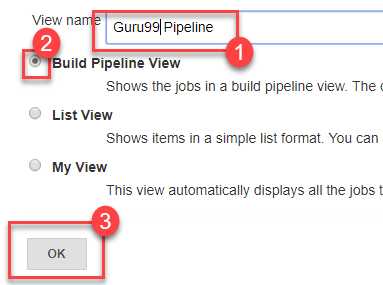
Once you are logged in to your Jenkins dashboard:

**Step 1**) Click on the **"+"**button on the left-hand side of your Jenkins dashboard to create a pipeline.

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe4.png)

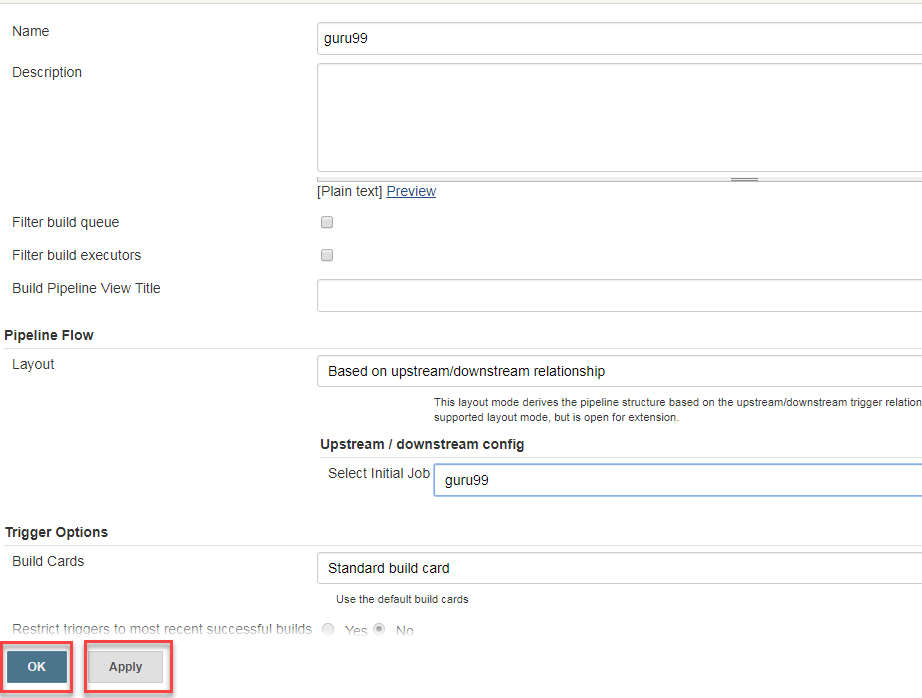
**Step 2**)

1. You will be asked to give a name to the pipeline view. We shall call it "**Guru99 Pipeline**" for the duration of this demo.
2. Select **Build a pipeline view**under **options**
3. Click **ok**

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe5.png)

**Step 4**) In the next page, you will be asked for some more details to configure your Jenkins pipeline. Just accept the default settings, and make sure you choose the first job under the settings.

Click on **Apply**and then **OK**.

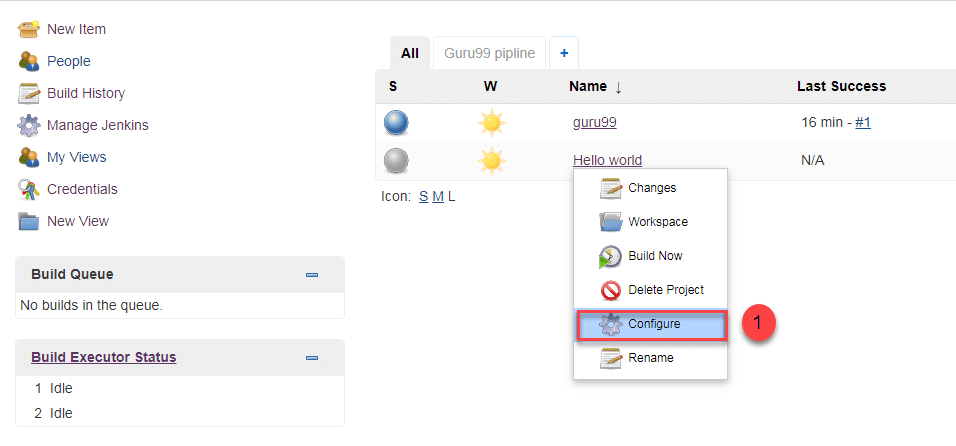
[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe6.png)

This will show you the sample pipeline view of your item, as given below:

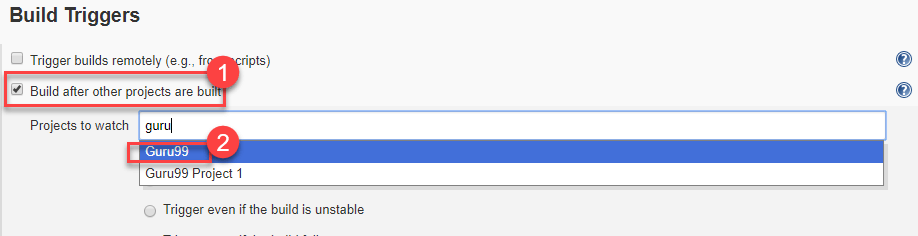
[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe7.png)

## Running a pipeline build

**Step 1**) For running a pipeline build, you need to chain your jobs first. For this, go to your first job and click on configure.

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe8.png)

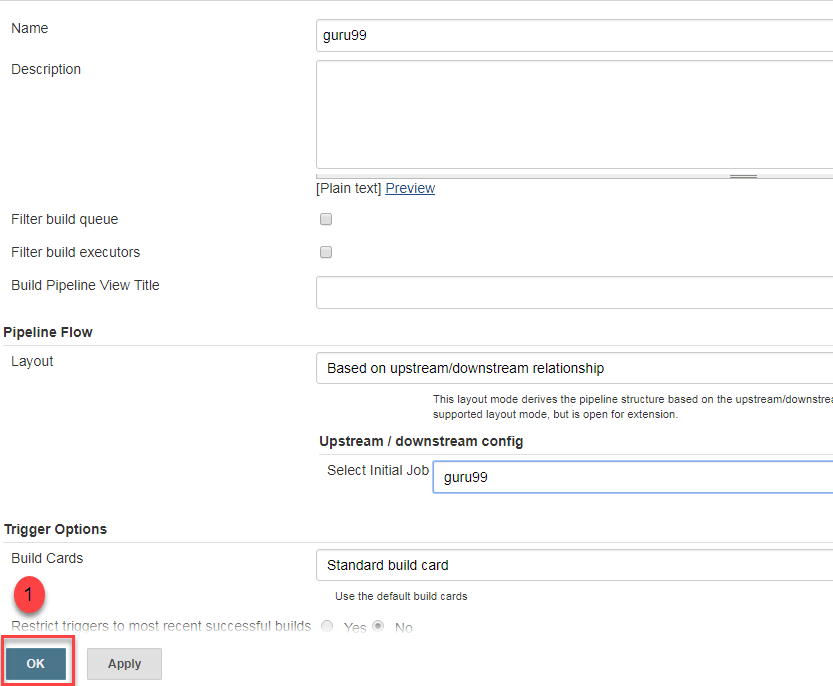
**Step 2**) Now, under **Build Triggers**, check the **Build after other projects are built** option.

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe9.png)

Thus, a chain for all your jobs has been created.

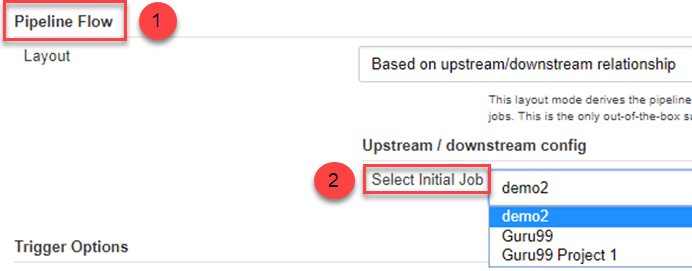
**Step 3**) Install the **Build Pipeline view** plugin if you don't have it installed already.

**Step 4**) Go to your Jenkins dashboard and create a view by clicking on the "**+**" button. Select the **Build Pipeline View** option and click **OK**.

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe10.png)

**Step 5**) Under **Pipeline view configuration**, locate **Pipeline Flow**.

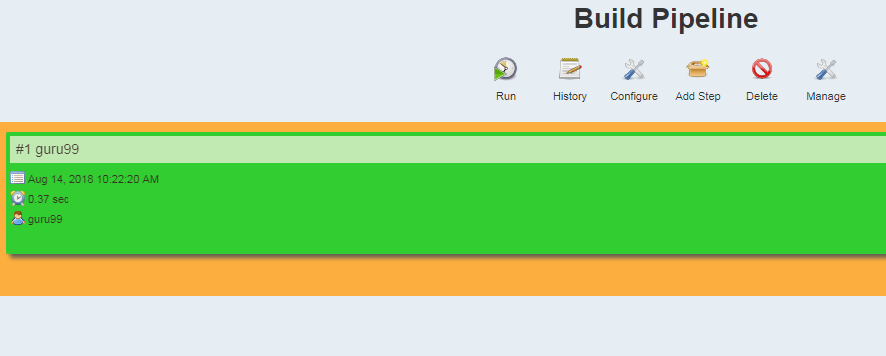
Under **Pipeline flow**, select the initial job to run. Now choose the job which has chains to other jobs, as shown in **Step 1** and **Step 2**.

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe11.png)

Here we have selected **Guru99 Project 1** as the initial job, chained to other jobs. So, one by one, the jobs will run in the pipeline.

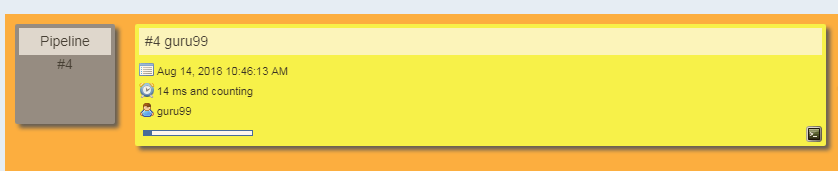
When the Jenkins pipeline is running, you can check its status with the help of Red and Green status symbols. Red means the pipeline has failed, while green indicates success.

In this example, we see that the button is green. Hence, the pipeline is successful.

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe12.png)

## Running Jenkins pipeline

Click on **Run**to run the Jenkins pipeline. It will look something like this:

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe13.png)

In the example above, we are demonstrating a simple "helloworld.java" program. But in real time projects, you will be responsible for creating and building complex pipelines in Jenkins. See below for a sample pipeline view.

[](https://www.guru99.com/images/1/091318_0510_JenkinsPipe14.png)