

Pipeline

Install Jenkins pipeline plugin

To install the Jenkins Pipeline plugin, follow these steps:


1. Open your Jenkins dashboard in a web browser and log in with administrative privileges.
2. Click on "Manage Jenkins" in the left-hand menu.
3. On the "Manage Jenkins" page, click on "Manage Plugins".
4. In the "Available" tab, search for "Pipeline" in the filter box.
5. **Look for the "Pipeline" plugin in the search results. It should be called something like "Pipeline: Groovy", "pipeline", or "Pipeline: Stage View".**
6. Tick the checkbox next to the "Pipeline" plugin.
7. Once you have selected the desired plugins, click on the **"Install without restart"** button at the bottom of the page.
8. Jenkins will download and install the selected plugins. The installation progress will be displayed on the "Plugin Manager" page.
9. Once the installation is complete, Jenkins will return to the "Plugin Manager" page, and you should see the installed plugins listed under the "Installed" tab.

The Pipeline plugin is now installed on your Jenkins server, and you can start creating and executing Jenkins Pipeline jobs using the Pipeline syntax.


Create pipeline

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.

**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.

**Folder**

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

Press it OK

Got to section:

Pipeline

Definition

Pipeline script ▼

Script ?

1

try sample Pipeline... ▼

Copy and paste below Declarative pipeline syntax

```
pipeline {  
  agent any  
  stages {  
    stage('Hello') {  
      steps {  
        sh 'echo "Hello World"'  
      }  
    }  
  }  
}
```

And press build now, check the console output

Perform same step and again create a new pipeline

```
node {  
    stage('Hello') {  
        sh 'echo "Hello World"'  
    }  
}
```

Started by user [admin](#)

[Pipeline] Start of Pipeline

[Pipeline] node

Running on [jenkins](#) in

/var/lib/jenkins/workspace/testpipe

[Pipeline] {

[Pipeline] stage

[Pipeline] { (Hello) ([hide](#))

[Pipeline] sh

+ echo Hello World

Hello World

[Pipeline] }

[Pipeline] // stage

[Pipeline] }

[Pipeline] // node

[Pipeline] End of Pipeline

Finished: SUCCESS

Started by user [admin](#)

[Pipeline] Start of Pipeline

[Pipeline] node

Running on [jenkins](#) in

/var/lib/jenkins/workspace/testing pipeline

[Pipeline] {

[Pipeline] stage

[Pipeline] { (Hello)

[Pipeline] sh

+ echo Hello World

Hello World

[Pipeline] }

[Pipeline] // stage

[Pipeline] }

[Pipeline] // node

[Pipeline] End of Pipeline

Finished: SUCCESS

Other than syntax no other observation

Multi stage scripted

```
node {
```

```
    stage('Hello') {
```

```
        sh 'echo "Hello World"'
```

```
}  
stage('stage 2') {  
  sh 'echo "Hello World 2"'  
}  
stage('stage 3') {  
  sh 'echo "Hello World 3"'  
}  
}
```

Multi stage declarative pipeline

```
pipeline {  
  agent any  
  stages {  
    stage('Hello') {  
      steps {  
        sh 'echo "Hello World"'  
      }  
    }  
    stage('stage 2') {  
      steps {  
        sh 'echo "Hello World 2"'  
      }  
    }  
  }  
}
```

```
    }  
  }  
  stage('stage 3') {  
    steps {  
      sh 'echo "Hello World 3"'  
    }  
  }  
}  
}
```

Check the console output