

## Jenkins initial setup

1. Go to `<master_ip_address>:8080` and follow the instructions. First enter the password form given file.

Getting Started


# Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log ([not sure where to find it?](#)) and this file on the server:

```
/var/lib/jenkins/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

Administrator password



Continue

2. Then choose option 'Select plugins to install'



3. Unselect all preselected plugins. Then select **SSH Slaves** plugin which is the only plugin from this list you actually need. Install.

Getting Started

Organization and Administration

Build Features

Build Tools

Build Analysis and Reporting

Pipelines and Continuous Delivery

Source Code Management

Distributed Builds

User Management and Security

Notifications and Publishing

All | None | Suggested

Selected (1/57)

Integrates Jenkins with Rational Team Concert source control and build using the richer features of the build toolkit instead of the command line.

☐ Team Foundation Server

23

Distributed Builds (1/3)

☐ Matrix Project

8

Multi-configuration (matrix) project type.

☒ SSH Slaves

6

Allows to launch agents over SSH, using a Java implementation of the SSH protocol

☐ Windows Slaves

3

Allows you to connect to Windows machines and start slave agents on them.

User Management and Security (0/5)

☐ Matrix Authorization Strategy

2

Offers matrix-based security authorization strategies (global and per-project).

☐ PAM Authentication

8

Adds Unix Pluggable Authentication Module (PAM) support to Jenkins

☐ LDAP

5

Adds LDAP authentication to Jenkins

☐ Role-based Authorization Strategy

4

☐ Active Directory

11

Enables authentication through Active Directory

Notifications and Publishing (0/5)

☐ Email Extension

10

Jenkins 2.107.3

Back

Install

4. Create a user with password for future login.

Getting Started

## Create First Admin User

Username:

Password:

Confirm password:

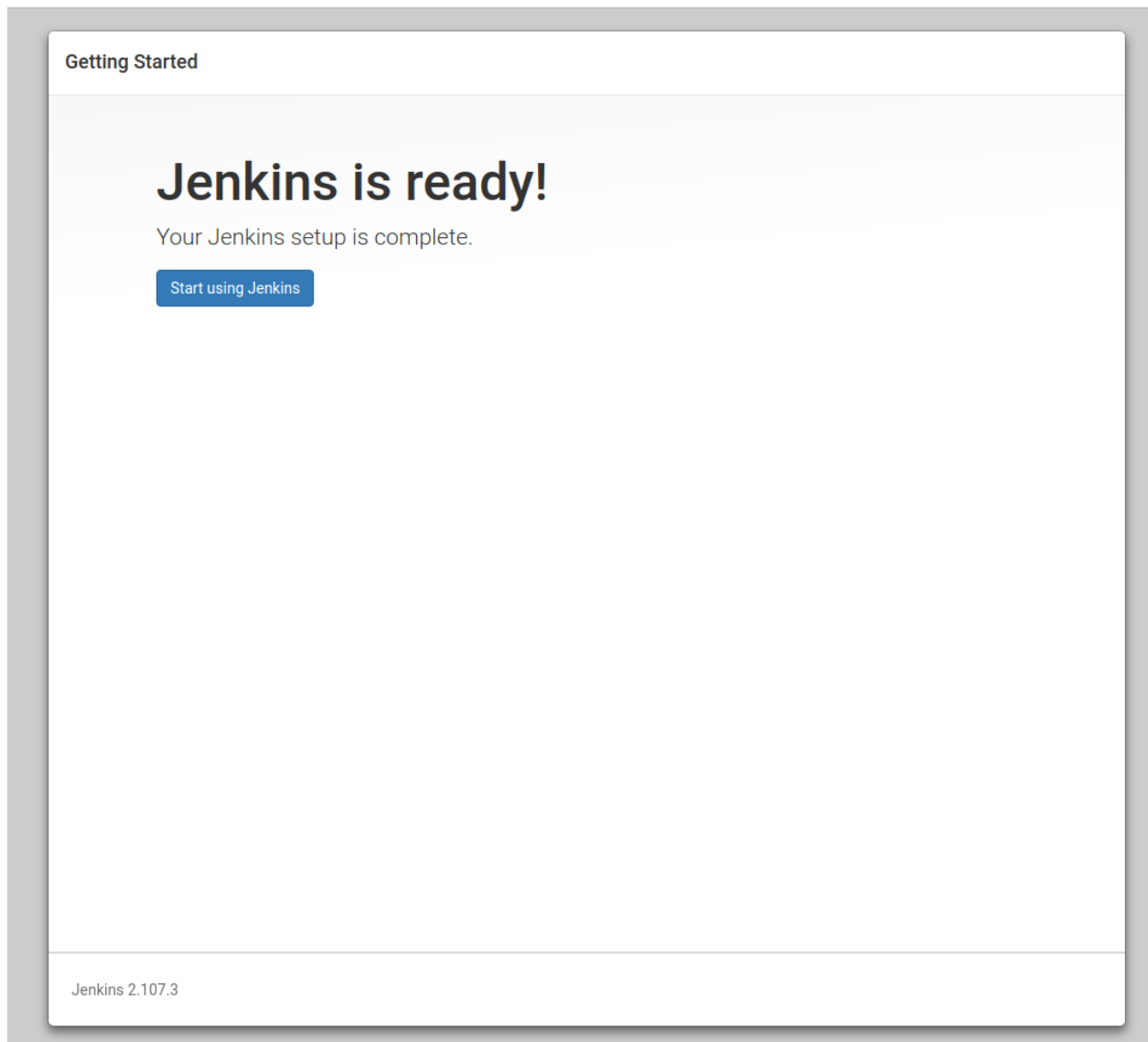
Full name:

Jenkins 2.107.3

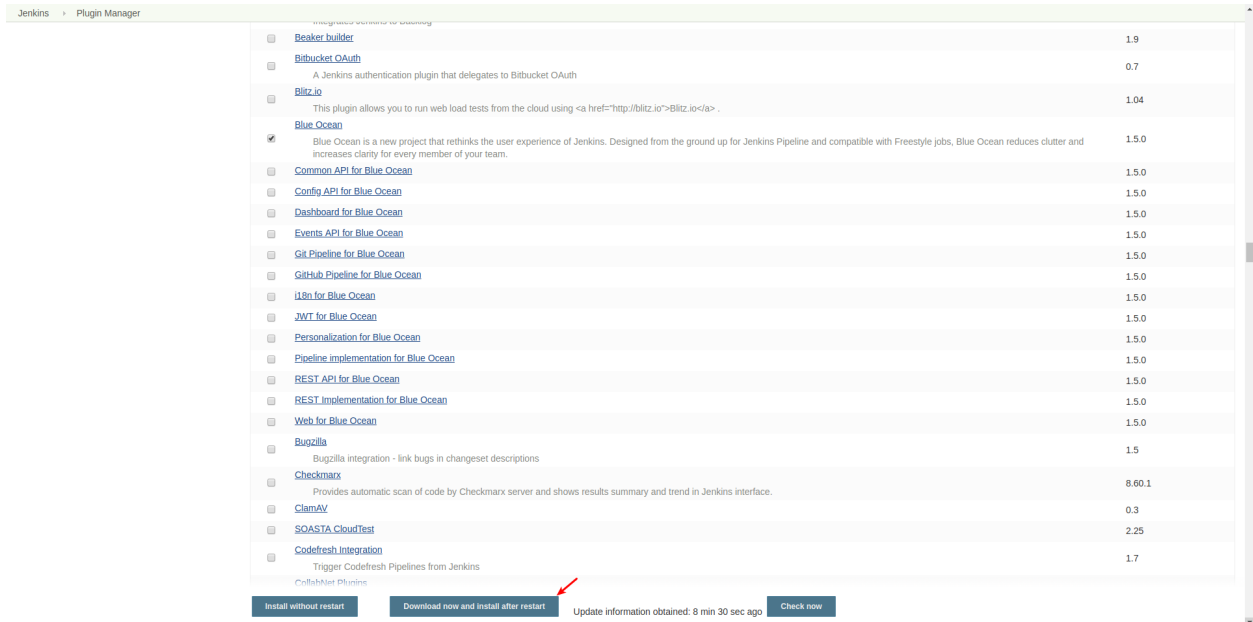
[Continue as admin](#)

Save and Finish






5. You can start using Jenkins now.



6. Now it is time to install other needed plugins which was not in the list of recommended plugins during initial settings. Go to Manage Jenkins -> Manage Plugins (<master\_ip\_address>:8080/pluginManager/) and in *Available* tab select **Blue Ocean** plugin and **Slave SetupPlugin**. Then click on 'Download and install after restart' button.



7. Don't forget to restart Jenkins when all plugins are downloaded.

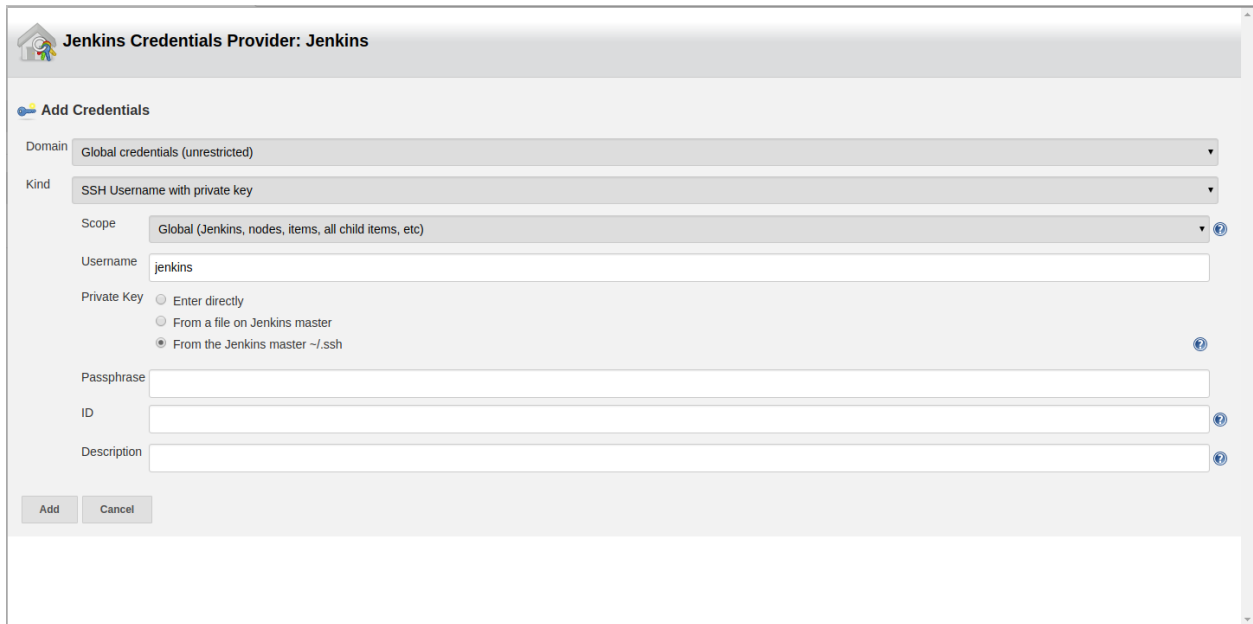
Blue Ocean Pipeline Editor	 Pending
i18n for Blue Ocean	 Pending
Autofavorite for Blue Ocean	 Pending
Blue Ocean	 Pending
Restarting Jenkins	 Pending

➡ [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

## Set credentials for ssh connection between machines

1. Go to `<master_ip_address>:8080/credentials/store/system/domain/_/newCredentials` where we add new credentials.
2. Set credentials according to screenshot and confirm.



The screenshot shows the 'Jenkins Credentials Provider: Jenkins' interface. At the top, there's a header with the Jenkins logo and the text 'Jenkins Credentials Provider: Jenkins'. Below this is a section titled 'Add Credentials' with a key icon. The form contains several fields: 'Domain' is a dropdown menu set to 'Global credentials (unrestricted)'; 'Kind' is a dropdown menu set to 'SSH Username with private key'; 'Scope' is a dropdown menu set to 'Global (Jenkins, nodes, items, all child items, etc)'; 'Username' is a text input field containing 'jenkins'; 'Private Key' has three radio button options: 'Enter directly' (selected), 'From a file on Jenkins master', and 'From the Jenkins master ~/.ssh'; 'Passphrase' is a text input field; 'ID' is a text input field; and 'Description' is a text input field. At the bottom left of the form are two buttons: 'Add' and 'Cancel'. There are also small question mark icons next to the 'Scope', 'Private Key', 'ID', and 'Description' fields.

## Slave setup

1. Go to Manage Jenkins -> Manage Nodes (<master\_ip\_address>:8080/computer).
2. Configure the slave according to screenshot. Don't forget to replace IP address and machine ID (use your machine private IP and ID).

The screenshot shows the Jenkins 'Manage Nodes' configuration page for a new slave node named 'slave01'. The configuration is as follows:

- Name:** slave01
- Description:** (empty)
- # of executors:** 1
- Remote root directory:** /data
- Labels:** (empty)
- Usage:** Use this node as much as possible
- Launch method:** Start and stop this node on-demand
  - Start script:** aws ec2 start-instances --instance-ids i-04830bf59ca621a83
  - Stop script:** aws ec2 stop-instances --instance-ids i-04830bf59ca621a83
  - Actual launch method:** Launch slave agents via SSH
    - Host:** 172.31.12.55
    - Credentials:** jenkins (with an 'Add' button)
    - Host Key Verification Strategy:** Non verifying Verification Strategy
- Availability:** Take this agent online when in demand, and offline when idle
  - In demand delay:** 0
  - Idle delay:** 1

Below the configuration fields, there are checkboxes for 'Environment variables' and 'Tool Locations', both of which are unchecked. A 'Save' button is located at the bottom left of the configuration area.

## Tips for aws console

Find machine ID

- aws ec2 describe-instances --filters 'Name=tag:Name,Values=<machine\_name>' --query 'Reservations[\*].Instances[\*].[InstanceId]'

Find machine private IP

- aws ec2 describe-instances --filters 'Name=tag:Name,Values=<machine\_name>' --query 'Reservations[\*].Instances[\*].[PrivateIpAddress]'

Start machine

- aws ec2 start-instances --instance-ids <machine\_id>

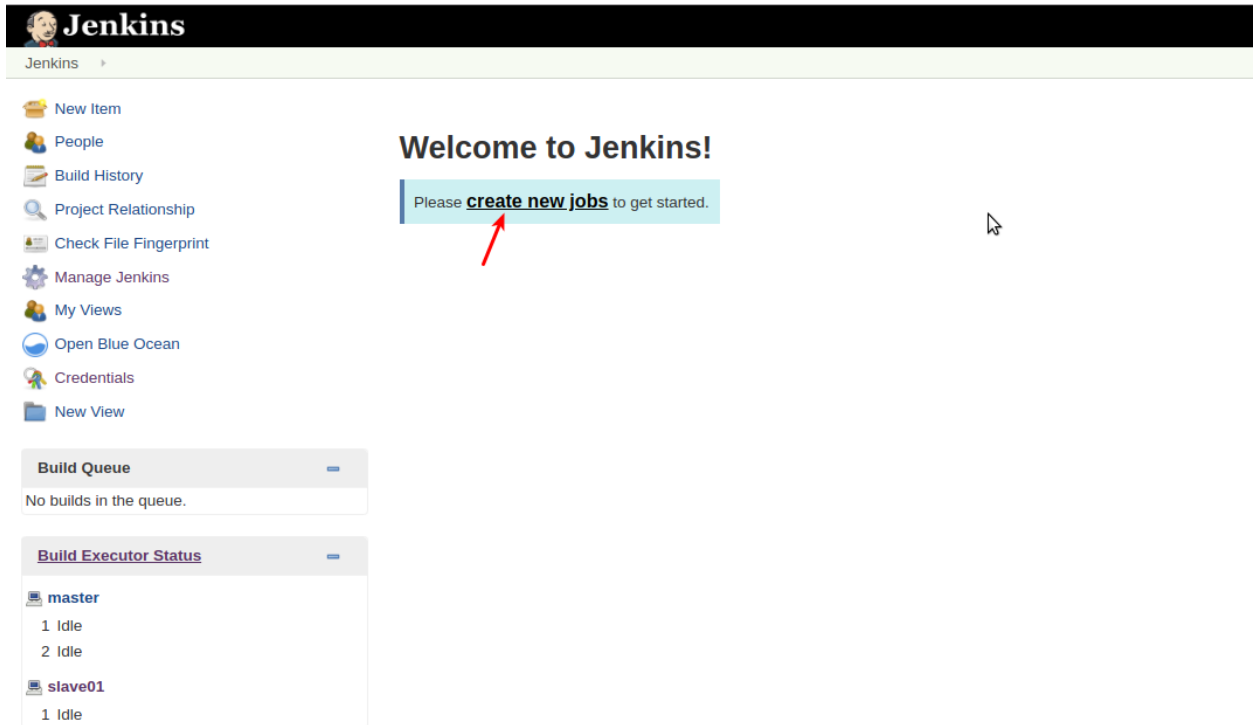
Stop machine

- aws ec2 stop-instances --instance-ids <machine\_id>



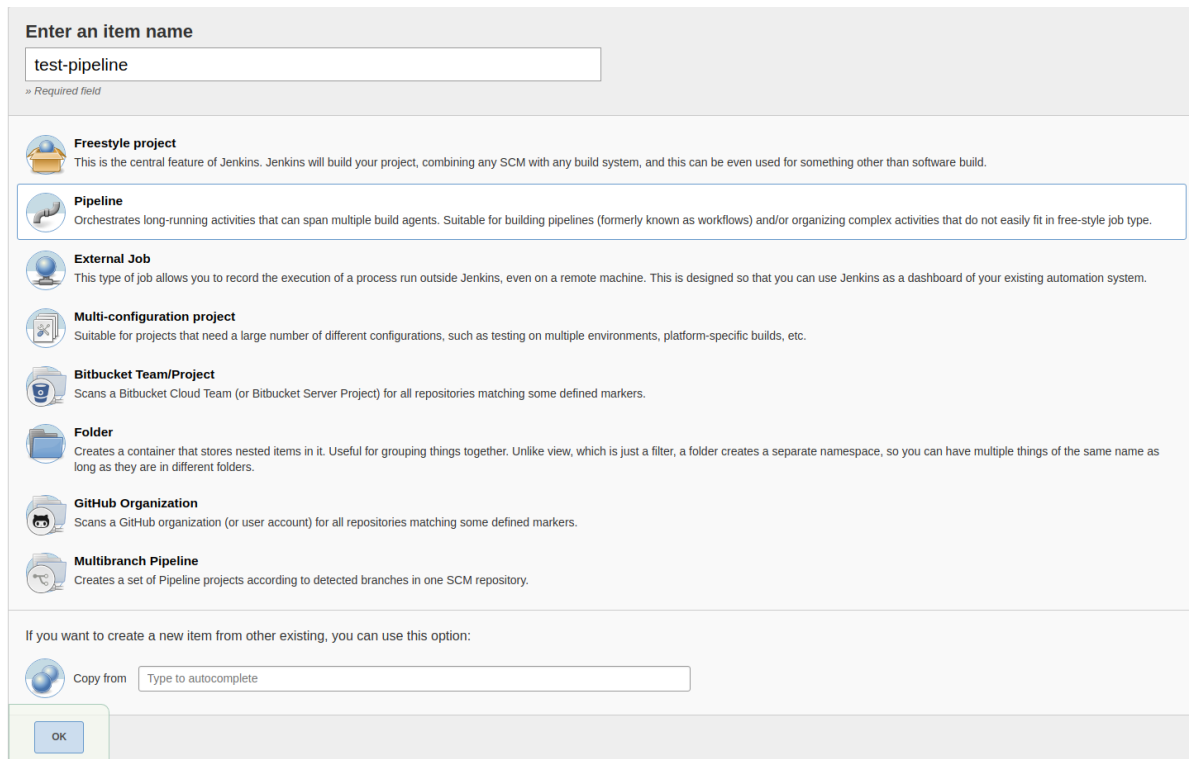
## Create new job

1. Click on create new job link.



The image shows the Jenkins dashboard. On the left is a sidebar with navigation links: New Item, People, Build History, Project Relationship, Check File Fingerprint, Manage Jenkins, My Views, Open Blue Ocean, Credentials, and New View. The main area has a 'Welcome to Jenkins!' message and a light blue box with the text 'Please create new jobs to get started.' A red arrow points to the underlined text. Below the welcome message are two sections: 'Build Queue' showing 'No builds in the queue.' and 'Build Executor Status' showing a list of executors: 'master' (1 Idle) and 'slave01' (1 Idle).

2. Enter job name, select 'Pipeline' and confirm.



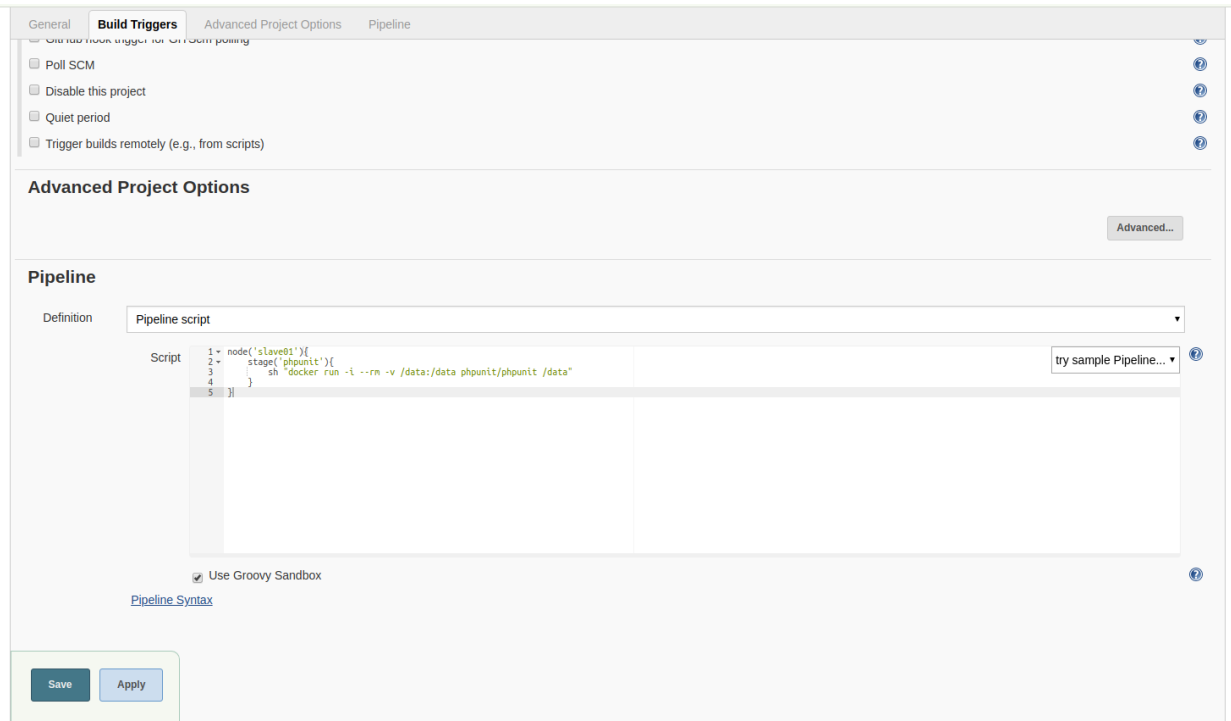
The image shows the 'Create New Item' form in Jenkins. At the top, there's a section 'Enter an item name' with a text input field containing 'test-pipeline' and a note '» Required field'. Below this is a list of item types, each with an icon and a description:

- 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.
- External Job**: This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This is designed so that you can use Jenkins as a dashboard of your existing automation system.
- Multi-configuration project**: Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
- Bitbucket Team/Project**: Scans a Bitbucket Cloud Team (or Bitbucket Server Project) for all repositories matching some defined markers.
- 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 separate namespace, so you can have multiple things of the same name as long as they are in different folders.
- GitHub Organization**: Scans a GitHub organization (or user account) for all repositories matching some defined markers.
- Multibranch Pipeline**: Creates a set of Pipeline projects according to detected branches in one SCM repository.

Below the list, there's a section 'If you want to create a new item from other existing, you can use this option:' with a 'Copy from' dropdown menu and a text input field with the placeholder 'Type to autocomplete'. At the bottom left is an 'OK' button.

3. Enter following pipeline script and save:

```
node('slave01'){
  stage('phpunit'){
    sh "docker run -i --rm -v /data:/data phpunit/phpunit /data"
  }
}
```



The screenshot shows the Jenkins configuration page for a pipeline. The tabs at the top are 'General', 'Build Triggers', 'Advanced Project Options', and 'Pipeline'. The 'Build Triggers' tab is active, showing options like 'Poll SCM', 'Disable this project', 'Quiet period', and 'Trigger builds remotely (e.g., from scripts)'. Below this is the 'Advanced Project Options' section with an 'Advanced...' button. The 'Pipeline' section is expanded, showing a 'Definition' dropdown set to 'Pipeline script'. The 'Script' field contains the following code:

```
1 node('slave01'){
2   stage('phpunit'){
3     sh "docker run -i --rm -v /data:/data phpunit/phpunit /data"
4   }
5 }
```

Below the script field, there is a checkbox for 'Use Groovy Sandbox' which is checked, and a link for 'Pipeline Syntax'. At the bottom left, there are 'Save' and 'Apply' buttons.

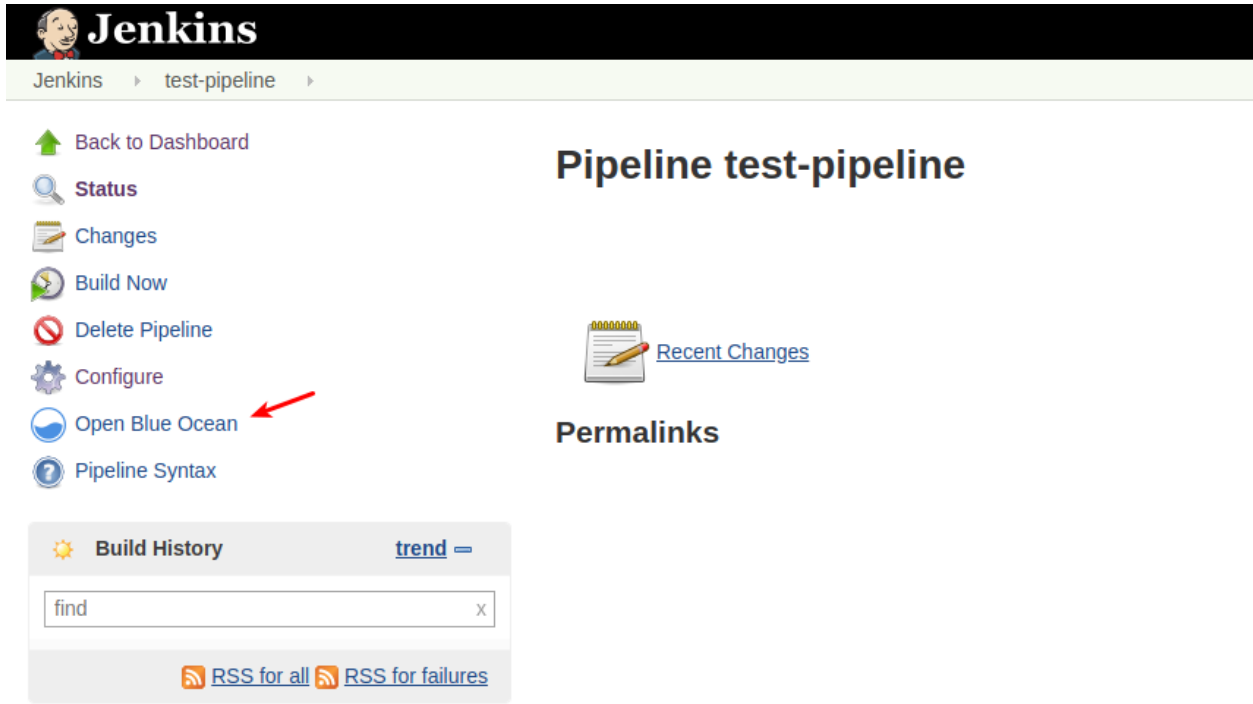
Notes:

- replace node name if you named your slave differently
- if you want to run something else that phpunit in docker, replace command 'docker run ...' with anything you want

## Run the job

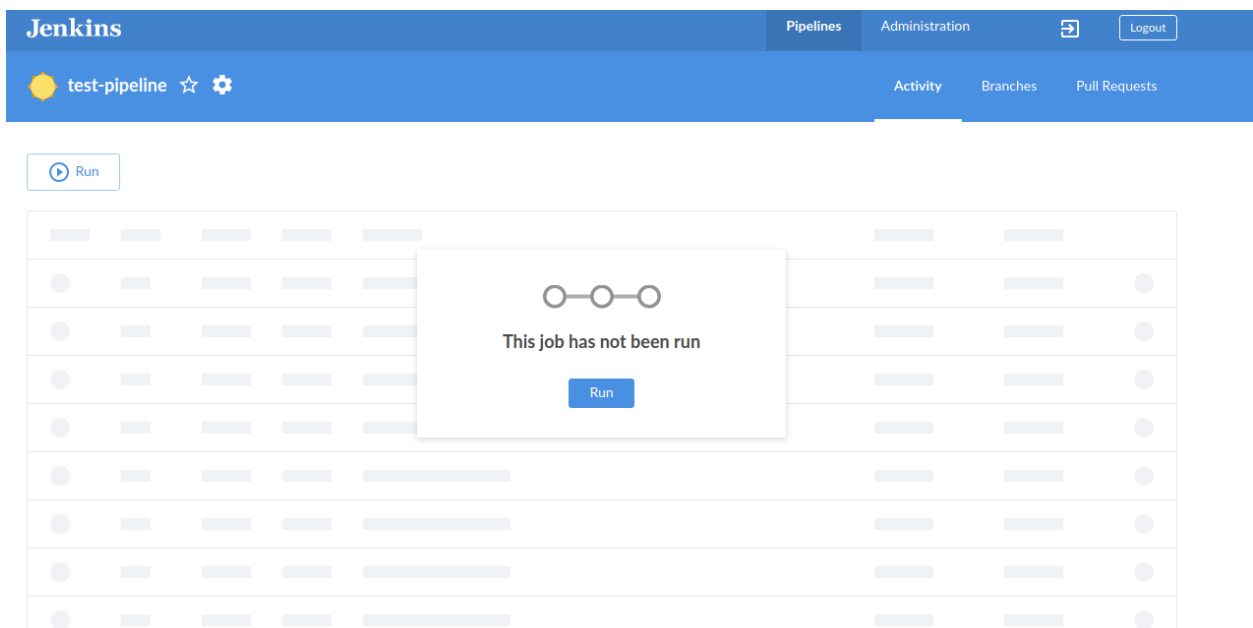
Now you can finally run the job.

1. Switch to Blue Ocean for better user friendly view.



The screenshot shows the Jenkins web interface for a pipeline named 'test-pipeline'. The top navigation bar includes 'Jenkins' and 'test-pipeline'. On the left sidebar, there are links: 'Back to Dashboard', 'Status', 'Changes', 'Build Now', 'Delete Pipeline', 'Configure', 'Open Blue Ocean' (highlighted with a red arrow), and 'Pipeline Syntax'. The main content area is titled 'Pipeline test-pipeline' and includes a 'Recent Changes' section with a notepad icon and a 'Permalinks' section. Below these is a 'Build History' section with a search bar containing 'find', a 'trend' dropdown, and RSS links for 'all' and 'failures'.

2. Run.



The screenshot shows the Jenkins 'Run' dialog for the 'test-pipeline' job. The top navigation bar includes 'Jenkins', 'Pipelines', 'Administration', and a 'Logout' button. The 'test-pipeline' job is selected, and the 'Run' button is visible. A modal dialog box is displayed in the center with the text 'This job has not been run' and a 'Run' button. The background shows a table with columns for job status, but it is partially obscured by the dialog.

### 3. If you see green color, you did it!

✓ test-pipeline 1

PipelineChangesTestsArtifacts🔄⚙️📄Logout✕

Branch: —

🕒 2m 18s

No changes

Commit: —

🕒 a few seconds ago

Started by user test@liw.com

Start

phpunit

End

phpunit - 2s

📄📄

✓

> docker run -i -rm -v /data:/data phpunit/phpunit /data — Shell Script

2s