

**Practice for Lesson 7:  
Jenkins Integration with  
Docker**

## Practices for Lesson 7

---

### Overview

In these practices, you will learn how to setup the environment for Docker container job creation on Jenkins instance. Further, create the Pipeline Job for Docker on Jenkins instance using a sample example and then deploying the Docker Container with the Jenkins Pipeline using the GitHub repository

## Practice 7-2: Create a Pipeline Job for Docker on Jenkins

---

### Overview

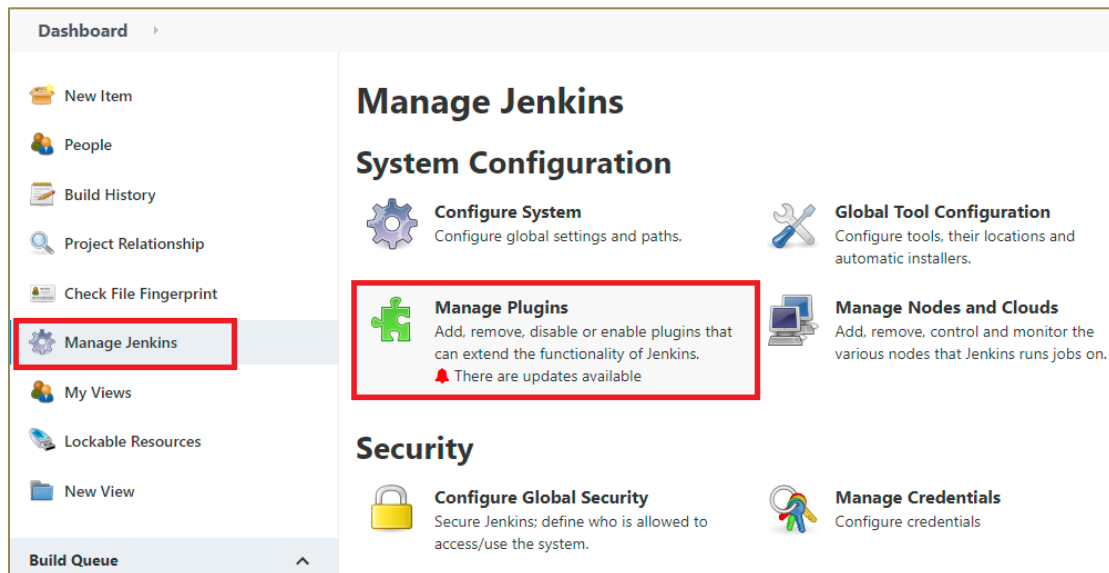
In this practice, you will learn how to create the Pipeline Job for Docker on Jenkins instance using a sample example.

### Assumptions

You should have completed the Practice of Lesson 7-1.

### Tasks

1. Install **Docker Pipeline** plugin on the **Linux** instance.
  - a. Navigate to Dashboard, select **Manage Jenkins** and click **Manage Plugins** as shown below.



- b. Select **Available** and search for **Docker Pipeline** plugin. Select the check box of the **Docker Pipeline** plugin and click **Install without restart**.

Dashboard > Plugin Manager

Back to Dashboard  
Manage Jenkins

Search:

Updates Available Installed Advanced

Install ↑	Name	Version	Released
<input type="checkbox"/>	<b>Docker</b> Cloud Providers Cluster Management and Distributed Build docker This plugin integrates Jenkins with Docker	1.2.2	5 mo 13 days ago
<input type="checkbox"/>	<b>Docker Commons</b> api-plugin docker Library plugins (for use by other plugins) Provides the common shared functionality for various Docker-related plugins.	1.17	1 yr 0 mo ago
<input checked="" type="checkbox"/>	<b>Docker Pipeline</b> Deployment DevOps docker pipeline Build and use Docker containers from pipelines.	1.26	4 mo 13 days ago
<input type="checkbox"/>	<b>Docker API</b> api-plugin docker This plugin provides docker-java API for other plugins. This plugin is up for adoption! We are looking for new maintainers. Visit our Adopt a Plugin initiative.	3.1.5.2	1 yr 3 mo ago

Update information obtained: 20 min

- c. The installation process will proceed and Success message will be displayed as shown below.

Jenkins

Dashboard > Update Center

Back to Dashboard  
Manage Jenkins  
Manage Plugins

## Installing Plugins/Upgrades

Preparation

- Checking internet connectivity
- Checking update center connectivity
- Success

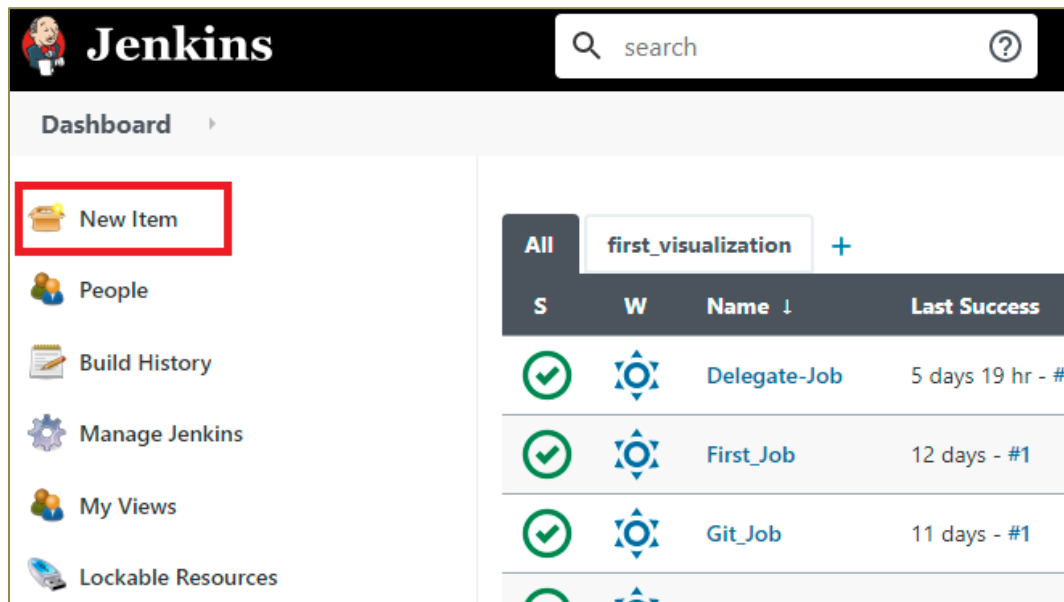
Docker Commons ☒ Success  
 Docker Pipeline ☒ Success  
 Loading plugin extensions ☒ Success

☒ [Go back to the top page](#)  
 (you can start using the installed plugins right away)

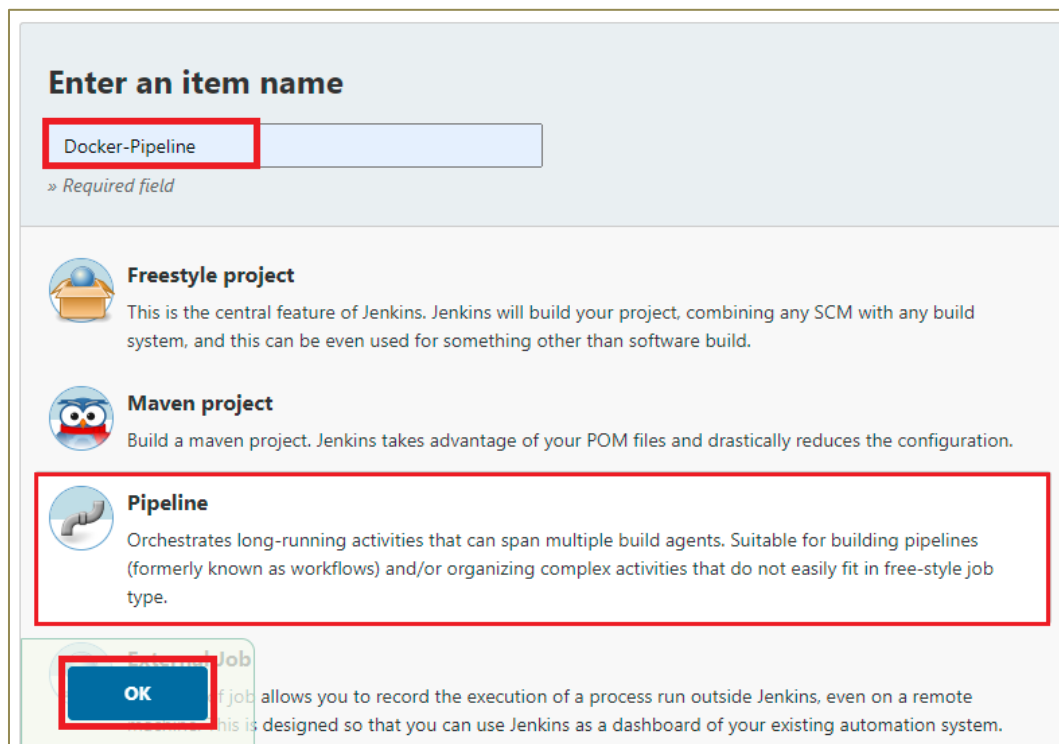
☐ Restart Jenkins when installation is complete and no jobs are running

2. Create a Docker Pipeline job in the Jenkins instance.

- a. In the Jenkins Dashboard, navigate to main menu and select **New Item** to create a **Docker** pipeline Job as shown below.



- b. Provide the name for Job, select **pipeline** and click **OK**.



- c. Navigate to **Pipeline** and select **Pipeline Script** under **Definition**. Copy the groovy script given below and paste it in the Jenkins Pipeline **Script** block. Click **Apply** and **Save**.

```

pipeline {
  agent {
    docker { image 'node:14-alpine' }
  }
  stages {
    stage('Test') {
      steps {
        sh 'node --version'
      }
    }
  }
}

```

General Build Triggers Advanced Project Options **Pipeline**

### Pipeline

Definition

Pipeline script

Script

```

1 pipeline {
2   agent {
3     docker { image 'node:14-alpine' }
4   }
5   stages {
6     stage('Test') {
7       steps {
8         sh 'node --version'
9       }
10    }
11  }
12 }

```

try sample Pipeline...

**ERROR**

☒ Use Groovy Sandbox

Pipeline Syntax

**Save** Apply

- d. Job is created successfully, click **Build Now** to execute the pipeline. Click on build created under **Build History** as shown below

- e. In the **Build** page, click **Console Output** to view the output of the job. The given Docker image is pulled successfully and tested.

```

Started by user oralabuser
Running in Durability level: MAX_SURVIVABILITY
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/Docker-Pipeline
[Pipeline] {
[Pipeline] isUnix
[Pipeline] sh
+ docker inspect -f . node:14-alpine

Error: No such object: node:14-alpine
[Pipeline] isUnix
[Pipeline] sh
+ docker pull node:14-alpine
14-alpine: Pulling from library/node
ddad3d7c1e96: Pulling fs layer
59857143b378: Pulling fs layer
9f27d30bcf3a: Pulling fs layer
be8cf85a075d: Pulling fs layer
be8cf85a075d: Waiting
ddad3d7c1e96: Verifying Checksum
ddad3d7c1e96: Download complete
9f27d30bcf3a: Verifying Checksum
9f27d30bcf3a: Download complete
59857143b378: Verifying Checksum
59857143b378: Download complete
ddad3d7c1e96: Pull complete
be8cf85a075d: Verifying Checksum
be8cf85a075d: Download complete

```

- f. The Docker image **version** is also displayed and in the end **SUCCESS** message is displayed as shown below.

```

[Pipeline] withDockerContainer
Jenkins does not seem to be running inside a container
$ docker run -t -d -u 995:993 -w /var/lib/jenkins/workspace/Docker-Pipeline -v /var/lib/jenkins/workspace/Docker-
Pipeline:/var/lib/jenkins/workspace/Docker-Pipeline:rw,z -v /var/lib/jenkins/workspace/Docker-
Pipeline@tmp:/var/lib/jenkins/workspace/Docker-Pipeline@tmp:rw,z -e ***** -e ***** -e ***** -e *****
-e ***** -e ***** -e ***** -e ***** -e ***** -e ***** -e ***** -e ***** -e ***** -e
***** -e ***** -e ***** -e ***** -e ***** -e ***** -e ***** -e ***** -e ***** -e
***** -e ***** -e ***** -e ***** -e ***** node:14-alpine cat
$ docker top 99a83a08243e8a381af9da4065472a826d0a3c4fe0ffba016f275340918b9a59 -eo pid,comm
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Test)
[Pipeline] sh
+ node --version
v14.17.3
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
$ docker stop --time=1 99a83a08243e8a381af9da4065472a826d0a3c4fe0ffba016f275340918b9a59
$ docker rm -f 99a83a08243e8a381af9da4065472a826d0a3c4fe0ffba016f275340918b9a59
[Pipeline] // withDockerContainer
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS

```

3. Keep the Jenkins Dashboard, terminal and the AWS Management Console open for the next practice.