What is Docker and why it is used?

**Docker** is a tool designed to make it easier to create, deploy, and run applications by using containers. Containers allow a developer to package up an application with all of the parts it needs, such as libraries and other dependencies, and ship it all out as one package.

What is the benefit of containerization?

The **benefits of containerization**. **Containerization** of applications brings many **benefits**, including the following: Portability between different platforms and clouds—it's truly write once, run anywhere. Efficiency through using far fewer resources than VMs and delivering higher utilization of compute resources

Why should I use Docker?

Why **use Docker**? **Docker** enables you to rapidly deploy server environments in “containers.” ... While **Docker** utilizes the virtualization technology in the Linux kernel, it does not create virtual machines (in fact, if you **run Docker** on MacOS or Windows, you'll have to **run** it on a virtual machine).

When should we use Docker?

**Docker is a basic tool, like git or java, that you should start incorporating into your daily development and ops practices.**

1. Use Docker as version control system for your entire app's operating system.
2. Use Docker when you want to distribute/collaborate on your app's operating system with a team.

What is Docker storage driver?

**Docker storage drivers**. ... This is where **storage drivers** come in. **Docker** supports several different **storage drivers**, using a pluggable architecture. The **storage driver** controls how images and containers are stored and managed on your **Docker** host.

Where is Docker container data stored?

In layman's terms, volumes are external **storage** areas used to **store data** produced by a **Docker container**. Volumes can be located on the **docker** host or even on remote machines. **Containers** are ephemeral, a fancy way of saying that they have very short lives.

Where are Docker images stored?

**Images** are **stored** inside /var/lib/**docker** and then under applicable storage driver directory.Aug 7, 2014

What is the docker image ?

A **Docker image** is a file, comprised of multiple layers, used to execute code in a **Docker container**. ... When the **Docker** user runs an **image**, it becomes one or multiple instances of that **container**. **Docker** is an open source OS-level virtualization software platform primarily designed for Linux and Windows

How does a docker image work?

**Docker** creates a new container, as though you had run a **docker** container create command manually. **Docker** allocates a read-write filesystem to the container, as its final layer. This allows a running container to create or modify files and directories in its local filesystem.

What is Docker overlay?

The docker\_gwbridge is a virtual bridge that connects the **overlay** networks (including the ingress network) to an individual **Docker** daemon's physical network. **Docker** creates it automatically when you initialize a swarm or join a **Docker** host to a swarm, but it is not a **Docker** device.

What is Docker swarm mode?

**Docker Swarm** is a clustering and scheduling tool for **Docker** containers. With **Swarm**, IT administrators and developers can establish and manage a cluster of **Docker** nodes as a single virtual system.

What is Docker swarm and Kubernetes?

**Docker Swarm** or simply **Swarm** is an open-source container orchestration platform and is the native clustering engine for and by **Docker**. Any software, services, or tools that run with **Docker** containers run equally well in **Swarm**. Also, **Swarm** utilizes the same command line from **Docker**

How does Docker swarm work?

A **swarm** consists of multiple **Docker** hosts which run in **swarm** mode and act as managers (to manage membership and delegation) and workers (which run **swarm** services). ... **Docker works** to maintain that desired state. For instance, if a worker node becomes unavailable, **Docker** schedules that node's tasks on other nodes.

What is Docker and Docker Swarm?

**Docker Swarm** is a clustering and scheduling tool for **Docker** containers. With **Swarm**, IT administrators and developers can establish and manage a cluster of **Docker** nodes as a single virtual system.

What is Docker swarm service?

To deploy an application image when **Docker** Engine is in **swarm** mode, you create a **service**. Frequently a **service** is the image for a microservice within the context of some larger application.

What is docker stack ?

**docker stack** is a command that's embedded into the **Docker** CLI. It lets you manage a cluster of **Docker** containers through **Docker** Swarm. It just so happens both **Docker** Compose and the **docker stack** command support the same **docker**-compose.yml file with slightly different features

installing docker on cent os ?

The process of installing Docker CE is fairly straightforward, but can differ slightly depending on the environment. In this lesson, we'll explore how to install and configure Docker CE in a CentOS environment. We'll also go through the process of granting a user permission to execute docker commands. This lesson will provide the understanding of the steps necessary to install Docker on a CentOS machine.

Instuling on docker on ubuent ?

While the process of installing Docker CE on Ubuntu is similar to the installation process for CentOS, there are some differences. In addition, a Docker Certified Associate is required to be able to install Docker in a variety of environments. In this lesson, we will install Docker CE in an Ubuntu environment.

Scaling and storage drives ?

The variety of environments and use cases in which Docker can be used means that there are a variety of storage needs. Container storage can be implemented in multiple ways through the use of various storage drivers, and those provide a pluggable framework for using different kinds of container storage. In this lesson, we will discuss what storage drivers are and identify the most widely-used ones. We will also demonstrate how to explicitly set the storage driver to use for a system.

Running container ?

Executing containers is the core feature of Docker. In this lesson we will dive into the process of executing containers using docker run. We will demonstrate how to use this command, and learn some of the important options and flags that can be used with it. We will also discuss some additional commands that can allow us to manage containers on a host. After completing this lesson, we'll know how to run and manage containers with Docker.

Updating the docker engine ?

When using Docker to manage containers, it is important to be able to keep the Docker engine up-to-date. In this lesson, we will discuss the process of both downgrading and upgrading the Docker engine.

Configuring logging drive ?

Storing and accessing container logs is an essential part of managing containers. Docker logging drivers allow us to choose our own logging implementation to fit our particular needs. In this lesson, we will discuss logging drivers. We will also see how to customize the system default logging driver configuration, as well as how to override the defaults for individual containers.

What are the two types of Docker swarm services?

A **swarm** consists of one or more nodes: physical or virtual machines running **Docker** Engine 1.12 or later in **swarm** mode. There are **two types** of nodes: managers and workers.

what is the log-driver in docker ?

**Docker** includes multiple **logging** mechanisms to help you get information from running containers and services. These mechanisms are called **logging drivers**. Each **Docker** daemon has a default **logging driver**, which each container uses unless you configure it to use a different **logging driver**.

What is Docker daemon?

The **Docker daemon** is a service that runs on your host operating system. It currently only runs on Linux because it depends on a number of Linux kernel features, but there are a few ways to run **Docker** on MacOS and Windows too. The **Docker daemon** itself exposes a REST API

What is a volume in Docker?

**Volumes** are the preferred mechanism for persisting data generated by and used by **Docker** containers. While bind mounts are dependent on the directory structure of the host machine, **volumes** are completely managed by **Docker**. ... New **volumes** can have their content pre-populated by a container.

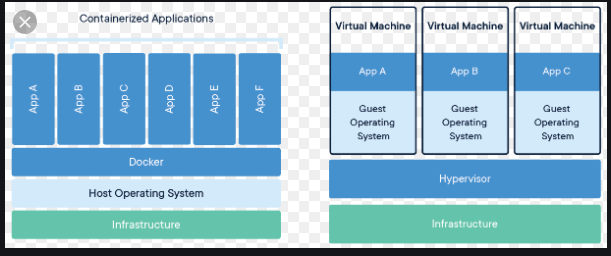
what is difference between run and cmd in dockerfile ?

**RUN and CMD** are both **Dockerfile** instructions. **RUN** lets you execute commands inside **of** your **Docker** image. These commands get executed once at build time and get written into your **Docker** image as a new layer. ... **CMD** lets you define a default **command** to **run** when your container starts

what is difference between add and copy in dockerfile ?

**COPY** and **ADD** are both **Dockerfile** instructions that serve similar purposes. They let you **copy** files from a specific location into a **Docker** image. **COPY** takes **in a** src and destination. ... A valid use case **for ADD** is when you want to extract a local tar file into a specific directory in your **Docker** image.

docker container architecture ?

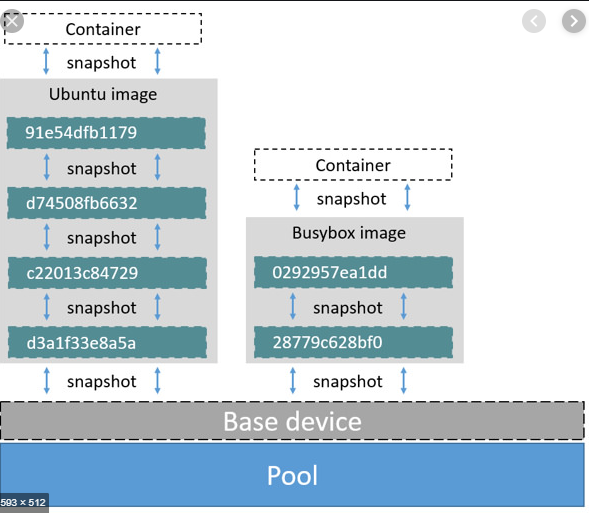


what is difference between container and vm in docker ?

**Docker** is **container** based technology and **containers** are just user space **of** the operating system. ... In **Docker**, the **containers** running share the host OS kernel. A **Virtual Machine**, on the other hand, is not based on **container** technology. They are made up **of** user space plus kernel space **of** an operating system

what is the Device Mapper in docker ?

**Device Mapper** is a kernel-based framework that underpins many advanced volume management technologies on Linux. ... The **devicemapper** driver uses block **devices** dedicated to **Docker** and operates at the block level, rather than the file level.



What is the container ?

Containers are the ready applications created from Docker Images or you can say a Docker Container is a running instance of a Docker Image and they hold the entire **package** needed to run the application.

What is the docker service ?

**Docker service** will be the image for a microservice within the context of some larger application. Examples of services might include an HTTP server, a database, or any other type of executable program that you wish to run in a distributed environment.

what are the service type in docker ?

There are two **types** of **service** deployments, replicated and global. For a replicated **service**, you specify the number of identical tasks you want to run. For example, you decide to deploy an HTTP **service** with three replicas, each serving the same content. A global **service** is a **service** that runs one task on every node

what is service in docker swarm ?

With **Docker Swarm** Mode, a **service** is a long-running **Docker** container that can be deployed to any node worker. It's something that either remote systems or other containers within the **swarm** can connect to and consume

What is the

What is the docker volumes ?

In order to be able to save (persist) data and also to share data between containers, **Docker** came up with the concept of **volumes**. Quite simply, **volumes** are directories (or files) that are outside of the default Union File System and exist as normal directories and files on the host filesystem.

what is difference between bind and volume in docker ?

The main **difference** a **bind** mount has from a **volume** is that since it can exist anywhere on the host filesystem, processes outside **of Docker** can also modify it. **Volumes**: **Volumes** are the preferred way to store persistent data **Docker** containers create or use. The host filesystem also stores **volumes**, similar to **bind** mounts.

# What is the Configure logging drivers

# multiple logging mechanisms to help you [get information from running containers and services](https://docs.docker.com/engine/admin/logging/view_container_logs/). These mechanisms are called logging drivers.

|  |  |
| --- | --- |
| [json-file](https://docs.docker.com/config/containers/logging/json-file/) | The logs are formatted as JSON. The default logging driver for Docker. |
| [syslog](https://docs.docker.com/config/containers/logging/syslog/) | Writes logging messages to the syslog facility. The syslog daemon must be running on the host machine. |

# what is difference between docker compose and docker stack ?

# Learn the differences between them. Docker Compose is an official tool that helps you manage your Docker containers by letting you define everything through a docker-compose.yml file. docker stack is a command that's embedded into the Docker CLI. It lets you manage a cluster of Docker containers through Docker Swarm

# what is service in docker?

# Docker service: Docker service will be the image for a microservice within the context of some larger application. Examples of services might include an HTTP server, a database, or any other type of executable program that you wish to run in a distributed environment.

# what is node labels in docker?

# Labels are metadata that describe the node, like its role (development, QA, production), its region (US, EU, APAC), or the kind of disk (hdd, ssd). Once you have labeled your nodes, you can add deployment constraints to your services, to ensure they are scheduled on a node with a specific label.

# 

# 