Docker

# Topics

* Container
* Image
* Running Containers in Interactive Environment
* Push at Dockerhub
* Docker Volume
* Mount Binds
* Working with API
* Container with Local database
* Docker Network
* Docker Compose(<https://www.youtube.com/watch?v=SXwC9fSwct8>)

# Basic Docker Interview Questions for Beginners

1. What is Docker?

[Docker](https://www.simplilearn.com/tutorials/docker-tutorial/getting-started-with-docker) is an open-source containerization platform. It is used to automate the deployment of any application, using lightweight, portable containers.

2. What are Docker’s most notable features?

Docker’s most essential features include:

* Application agility
* Developer productivity
* Easy modeling
* Operational efficiencies
* Placement and affinity
* Version control

3. Why should anyone use Docker? What does it offer?

Docker gives users many incentives for adoption, such as:

* An efficient and easy initial set up experience
* The means to describe an application lifecycle in detail
* Simple configuration and smooth interaction with [Docker Compose](https://www.simplilearn.com/tutorials/docker-tutorial/docker-compose)
* Complete and well-detailed documentation
* Ability to run on a PC or enterprise IT system with equal ease

4. What about the opposite? Does Docker have any downsides?

Docker isn’t perfect. It comes with its share of drawbacks, including:

* Lacks a storage option
* Monitoring options are less than ideal
* You can’t automatically reschedule inactive nodes
* Automatic horizontal scaling set up is complicated

5. Name and explain the various Docker components.

The three main [Docker components](https://www.simplilearn.com/tutorials/docker-tutorial/what-is-docker) are:

1. Docker Client. Performs Docker build pull and run operations to open up communication with the Docker Host. The Docker command then employs Docker API to call any queries to run.
2. Docker Host. Contains Docker daemon, containers, and associated images. The Docker daemon establishes a connection with the Registry. The stored images are the type of metadata dedicated to containerized applications.
3. Registry. This is where [Docker images](https://www.simplilearn.com/tutorials/docker-tutorial/docker-images) are stored. There are two of them, a public registry and a private one. [Docker Hub](https://www.simplilearn.com/tutorials/docker-tutorial/docker-hub) and Docker Cloud are two public registries available for use by anyone.

6. What is a container?

Containers are deployed applications bundled with all necessary dependencies and configuration files. All of the elements share the same OS kernel. Since the container isn’t tied to any one IT infrastructure, it can run on a different system or the cloud.

7. Explain virtualization.

[Virtualization](https://www.simplilearn.com/virtualization-in-cloud-computing-article) is the means of employing software (such as Hypervisor) to create a virtual version of a resource such as a server, [data storage](https://www.simplilearn.com/big-data-era-data-storage-rules-article), or application. Virtualization lets you divide a system into a series of separate sections, each one acting as a distinct individual system. The virtual environment is called a virtual machine.

Prepare Yourself to Answer All Questions!

Automation Testing Masters Program[Explore Program](https://www.simplilearn.com/automation-testing-masters-program-certification-training-course?source=GhPreviewCTABanner)



8. What’s the difference between virtualization and containerization?

Virtualization is an abstract version of a physical machine, while containerization is the abstract version of an application.

9. Last simple question…Describe a Docker container’s lifecycle.

Although there are several different ways of describing the steps in a Docker container’s lifecycle, the following is the most common:

1. Create container
2. Run container
3. Pause container
4. Unpause container
5. Start container
6. Stop container
7. Restart container
8. Kill container
9. Destroy container

# Exclusive Intermediate Interview Questions on Docker

10. Name the essential Docker commands and what they do.

The most critical [Docker commands](https://www.simplilearn.com/tutorials/docker-tutorial/docker-commands) are:

* Build. Builds a Docker image file
* Commit. Creates a new image from container changes
* Create. Creates a new container
* Dockerd. Launches Docker daemon
* Kill. Kills a container

11. What are Docker object labels?

Labels are the mechanism for applying metadata to Docker objects such as containers, images, local daemons, networks, volumes, and nodes.

12. How do you find stored Docker volumes?

Use the command: /var/lib/docker/volumes

Want a Top Software Development Job? Start Here!

Full Stack Developer - MERN Stack[Explore Program](https://www.simplilearn.com/full-stack-developer-course-mern-certification-training?source=GhPreviewCTABanner)



13. How do you check the versions of Docker Client and Server?

This command gives you all the information you need: $ docker version

14. Show how you would create a container from an image.

To create a container, you pull an image from the Docker repository and run it using the following command: $ docker run -it -d <image\_name>

15. How about a command to stop the container?

Use the following command: $ sudo docker stop container name

16. How would you list all of the containers currently running?

Use the command: $ docker ps

17. What’s involved in scaling a Docker container?

Docker containers have the potential to be scaled to any level needed. Thanks to the platform’s flexibility, you can have anything from a few hundred to a few thousand, to millions of containers, providing they all have continual, unconstrained access to the required memory and OS.

18. What do you know about the Docker system prune?

It’s a command used to remove all stopped containers, unused networks, build caches, and dangling images. Prune is one of the most useful commands in Docker. The syntax is:  $ docker system prune

We will next look into the advanced level docker interview questions and answers.

Advanced Docker Interview Questions for Experienced Professionals

19. List some of the more advanced Docker commands and what they do.

Some advanced commands include:

* Docker info. Displays system-wide information regarding the Docker installation
* Docker pull. Downloads an image
* Docker stats. Provides you with container information
* Docker images. Lists downloaded images

20. Can you lose data stored in a container?

Any data stored in a container remains there unless you delete the container.

21. What platforms can you run Docker on?

The Linux platforms are:

* ArchLinux
* CentOS 6+
* CRUX 3.0+
* Fedora 19/20+
* Gentoo
* openSUSE 12.3+
* RHEL 6.5+
* Ubuntu 12.04, 13.04 et al

Docker can also run on the following cloud-based platforms:

* Amazon EC2
* Amazon ECS
* Google Compute Engine
* Microsoft Azure
* Rackspace

22. Which is the best method for removing a container: the command “stop container” followed by the command “remove the container,” the rm command by itself?

Stop the container first, then remove it. Here’s how:

* $ docker stop <coontainer\_id>
* $ docker rm -f <container\_id>

23. Can a container restart on its own?

Since the default flag -reset is set to false, a container cannot restart by itself.

24. How do Docker daemon and the Docker client communicate with each other?

You use a combination of Rest API, socket.IO, and TCP to facilitate communication.

25. Can you implement continuous development (CD) and continuous integration (CI) in Docker?

Yes, you can. You can run [Jenkins](https://www.simplilearn.com/tutorials/jenkins-tutorial/what-is-jenkins) on Docker and use Docker Compose to run integration tests.

26. Finally, how do you create a Docker swarm?

Use the following command: docker swarm init –advertise-addr <manager IP>

# 25 Popular Docker Interview Questions You Should Prepare For

**Q1. Define Docker!**

Docker is one of the world’s leading containerization technologies. It is a lightweight open-source solution popular in the app packaging and cloud domains. Docker helps businesses automate app deployment by dividing it into portable and lightweight containers.

**Q2. What are the greatest advantages of Docker Containers?**

**There are several benefits of Docker for enterprises. These include:**

- A simple and efficient initial infrastructure set-up

- Enables businesses to describe in detail their app lifecycle

- Simple configuration

- Docker Compose interaction functionality

- Shares all necessary information through its documentation

**Q3. What are Docker’s Most Critical Features?**

**The fundamental features of Docker are:**

- Version control

- Easy modelling

- Application agility

- Placement or affinity

- Operational efficiencies

- Developer productivity

**Q4. What do you know about Docker Image?**

Docker image helps in creating Docker containers. Users can create a Docker image using the ‘Build’ command. This will make a container which starts as soon as it runs. The Docker registry stores every Docker image made.

**Q5. What do you know about Docker Engine?**

Docker engine or daemon represents the Docker server. Clients and the daemon must always run on a remote host or the same host that communicates through a full REST API and client’s command-line binary.

**Q6. What is a Docker Registry?**

**Docker registries can be categorised into two types:**

- Private Registry

- Public Registry

The public Docker registry is known as Docker Hub and enables businesses to store Docker images privately. Docker Hub can generally store images well into the millions.

**Q7. What is the command for viewing all the running containers in Docker?**

This command is -> $ docker ps

**Q8. What is the command that stops a container that is running?**

**This command is ->** $ sudo docker stop container name

**Q9. Which command helps a Docker image run as a container?**

**This command is ->** $ sudo docker -i -t alpine/bin/bash

**Q10. What are the most common instructions given in Dockerfile?**

FROM, RUN, CMD and LABEL are the Dockerfile common instructions.

**Q11. What is a memory-swap flag?**

This refers to a modified flag with meaning only under the condition that the memory is set too. Swap helps the container write memory needs to a disk when containers exhaust all the available RAM in the local system.

**Q12. Explain Docker Swarm very briefly.**

Docker Swarm is a native Docker gathering designed to help businesses connect to several Docker Hosts into one virtual host. Docker Swarm provides businesses with the standard Docker app interface.

**Q13. How can you monitor Docker in a production environment?**

Businesses use Docker events and Docker states to monitor Docker within production environments.

**Q14. What are the different states a Docker container goes through?**

A Docker container’s important states are running, restarting, pausing and exiting.

**Q15. What is Docker Hub?**

As mentioned before, Docker Hub refers to a public Docker registry based on the cloud that helps enterprises connect with code repositories. Hub enables businesses to build, store and test Docker images in the cloud. Additionally, they can deploy images to their hosts using Docker Hub.

**Q16. What is virtualization? Explain!**

Virtualization is a technique of dividing mainframes logically, allowing several apps to run at the same time. However, virtualization has seen a change ever since businesses as well as open-source communities started providing a method to handle privileged instructions. This allows several operating systems to run on singular x86 systems simultaneously.

**Q17. What can you tell me about Hypervisor?**

Hypervisors enable users to create virtual environments in which guest VMs can operate. A hypervisor controls guest systems while ensuring guests get allocated all the necessary resources.

**Q18. What are Docker Object Labels?**

Think of Docker Object Labels as a process used to apply metadata to images, volumes, containers, services, swam nodes, networks and other Docker objects.

**Q19. Give an example of a Docker file used for creating and copying a directory. Build it with Python modules.**

**An example of this situation is**

FROM python:2.7-slim

WORKDIR /app

COPY ./app

docker build -tag

**Q20. Will cloud overtake containerization in the coming years?**

Docker containers have become increasingly popular today but are almost paralleled by cloud services in terms of growth. However, there’s enough reason to believe that cloud services cannot replace Docker. Instead, combining cloud services using containerization will take the capabilities of both technologies to a whole new level. In situations where either of the two technologies can be used, businesses must choose after looking at their specific requirements instead of following existing trends. Most organisations today use Docker with cloud integration. This gives them the best of both technologies and enables improved service offerings and business growth.

**Q21. How many containers per host can be run?**

A single host can run as many containers as they wish or require. Docker has not placed any restrictions on the number of containers. However, every container requires storage space, which is represented by the CPU and other memory hardware required for support. Businesses should also consider the size of the application. Docker containers are usually lightweight, but that depends heavily on the hosting operating system.

**Q22. Is it generally considered a good practice to use Docker to run stateful apps?**

Stateful applications are built on the concept of storing data on their local file systems. If users move the app to a different machine, data retrieval is a cumbersome process. Therefore, experts recommend that Docker should not be used to run stateful applications regularly.

**Q23. Consider a situation where you have an app with several dependent services. Does Docker Compose wait for your present container before it is ready to move ahead and run the next service? Explain.**

The short answer is yes. Docker Compose will always run in the order of dependency. Each dependency is a specification such as links, depends\_on and volumes\_from.

**Q24. Which changes should you make in a personal Docker Compose file before it moves to production?**

Some changes are essential in a Docker Compose file before migrating the application to a production environment. These are:

- Removing volume bindings to ensure the code remains within the container will mean no one can change it from outside.

- Binding to separate localhost ports

- Adding log aggregator and other extra services

- Specifying a restart policy

**Q25. What do you know about load balancing done across hosts and containers? Can you explain how it works?**

When you use Docker with several containers across multiple hosts, you find that load balancing all incoming traffic is a necessity. HAProxy and load balancing are generally used to balance this traffic across various healthy or available containers. If any container crashes for any reason, a separate container should start running automatically, and the traffic must get rerouted to it. Both HAProxy and Load Balancing operate on this concept.