
DOCKER
BY PYUSH DWIVEDI

1. **What is Docker?**

Docker is an open-source platform that allows you to automate the deployment, scaling, and management of applications using containerization.

2. **What is a container?**

A container is a lightweight and isolated environment that encapsulates an application and its dependencies, allowing it to run consistently across different environments.

3. **How is Docker different from virtualization?**

Virtualization emulates an entire operating system, while Docker containers share the host system's kernel and only isolate the application processes.

4. **What is Docker image?**

A Docker image is a read-only template that contains the application and all its dependencies. It is used to create Docker containers.

5. **How do you create a Docker image?**

You can create a Docker image using a Dockerfile, which is a text file that contains a set of instructions for building the image.

6. **What is a Dockerfile?**

A Dockerfile is a text file that contains instructions for building a Docker image. It specifies the base image, application code, dependencies, and other configurations.

7. **What is Docker Compose?**

Docker Compose is a tool that allows you to define and manage multi-container Docker applications. It uses a YAML file to configure the application's services and their dependencies.

8. How do you link containers in Docker?

In Docker, you can link containers using the --link flag or by creating a user-defined network and connecting containers to the network.

9. What is Docker Swarm?

Docker Swarm is a native clustering and orchestration solution provided by Docker. It allows you to create and manage a cluster of Docker nodes to deploy and scale applications.

10. What is the difference between Docker Swarm and Kubernetes?

Docker Swarm is a simpler and less feature-rich orchestration tool compared to Kubernetes. It is suitable for small to medium-sized deployments, while Kubernetes is more scalable and suitable for complex, large-scale deployments.

11. How do you scale Docker containers?

You can scale Docker containers manually by running multiple instances of the same container or use orchestration tools like Docker Swarm or Kubernetes to automatically scale the containers based on predefined rules.

12. How does Docker ensure security?

Docker provides security through isolation by using kernel namespaces and control groups. It also supports various security features such as user namespaces, seccomp, and AppArmor profiles.

13. What is the difference between a Docker container and an image?

An image is a template used to create containers. Containers are running instances of images that have their own filesystem and can be started, stopped, and managed.

14. How can you share Docker images with others?

You can share Docker images with others by pushing them to a Docker registry, such as Docker Hub or a private registry. Others can then pull the image from the registry to use it.

15. How do you manage data persistence in Docker?

Data persistence in Docker can be achieved by using Docker volumes or mounting host directories into containers. Docker volumes provide a way to manage and share data between containers and also persist data even if the containers are removed.

16. What is the purpose of Dockerfile's ENTRYPOINT and CMD instructions?

The ENTRYPOINT instruction specifies the command that will be executed when the container starts. The CMD instruction provides default arguments to the entry point command.

17. How can you access logs from a Docker container?

You can access logs from a Docker container using the docker logs command followed by the container ID or name.

18. How can you pass environment variables to a Docker container?

You can pass environment variables to a Docker container using the -e flag followed by the variable name and value when running the docker run command.

19. What is the difference between a Docker image and a Docker container?

A Docker image is a static, read-only file that contains the application and its dependencies, while a Docker container is a running instance of an image that has its own state and can be started, stopped, and managed.

20. How do you remove Docker images and containers?

You can remove Docker images using the docker rmi command followed by the image ID or name. Containers can be removed using the docker rm command followed by the container ID or name.

21. What is Docker Registry?

Docker Registry is a service that stores and distributes Docker images. Docker Hub is the default public Docker Registry, but you can also set up your own private registry.

22. How can you update a Docker image?

To update a Docker image, you need to rebuild it with the necessary changes and then push the updated image to the Docker registry. Existing containers can be stopped and recreated with the new image.

23. What are Docker volumes?

Docker volumes are a way to persist data generated by and used by Docker containers. They provide a mechanism for managing and sharing data between containers, as well as persisting data even if the containers are removed.

24. How can you list Docker containers?

You can list Docker containers using the docker ps command. The -a flag can be used to display all containers, including the ones that are not currently running.

25. What is the purpose of Docker networks?

Docker networks provide a way to enable communication between containers running on the same host or across different hosts. They allow containers to discover and connect to each other using container names.

26. How do you update Docker containers?

To update a Docker container, you need to stop the existing container, pull the updated image from the Docker registry, and then start a new container with the updated image.

27. What is the difference between COPY and ADD instructions in a Dockerfile?

The COPY instruction in a Dockerfile copies files and directories from the build context to the image. The ADD instruction can do the same but also supports additional features like extracting tar archives and downloading files from URLs.

28. What is the role of a Dockerfile's EXPOSE instruction?

The EXPOSE instruction in a Dockerfile informs Docker that the container listens on the specified network ports at runtime. It does not actually publish the ports to the host, but it is useful for documentation purposes and when creating container links.

29. What is the difference between Docker and Kubernetes?

Docker is a containerization platform that allows you to create, package, and run applications in containers. Kubernetes, on the other hand, is a container orchestration platform that automates the deployment, scaling, and management of containerized applications.

30. How can you clean up unused Docker resources?

You can clean up unused Docker resources using the docker system prune command. This command removes unused images, containers, volumes, and networks, freeing up disk space. Use it with caution, as it permanently deletes resources.

31. What is the difference between a Docker volume and a bind mount?

- **Answer**: A Docker volume is managed by Docker and is stored in a specific directory on the host. It is more portable and easier to manage than a bind mount, which directly mounts a host directory or file into the container.

32.What is a multi-stage build in Docker?

- **Answer**: Multi-stage builds allow you to use multiple `FROM` statements in a Dockerfile to create smaller and more efficient images by copying only the necessary artifacts from intermediate stages.

33. How can you reduce the size of a Docker image?

- **Answer**: To reduce the size of a Docker image, you can use multi-stage builds, choose a smaller base image, remove unnecessary files and packages, and minimize the number of layers in the Dockerfile.

34.What are Docker tags?

- **Answer**: Docker tags are labels used to identify different versions of a Docker image. Tags are added to an image name using a colon, such as `myimage:latest` or `myimage:v1.0`.

35. What is a Docker context?

- **Answer**: A Docker context is the set of files and directories that are accessible to the Docker daemon during the build process. It is typically specified as the directory path passed to the `docker build` command.

36.How can you use secrets in Docker?

- **Answer**: Secrets in Docker can be managed using Docker Swarm or Kubernetes. In Docker Swarm, secrets can be created and used by services using the `docker secret` command. Kubernetes also has a secrets management feature that allows you to store and manage sensitive data.

37. What is the difference between a bridge network and an overlay network in Docker?

- **Answer**: A bridge network is used for communication between containers on the same host, while an overlay network allows containers to communicate across different hosts in a Docker Swarm cluster.

38.What is Docker In Docker (DinD)?

- **Answer**: Docker In Docker (DinD) refers to running Docker inside a Docker container. This can be useful for CI/CD pipelines or testing Docker-related workflows.

39.What is the purpose of the `.dockerignore` file?

- **Answer**: The `.dockerignore` file is used to specify files and directories that should be ignored by the Docker build context, similar to a `.gitignore` file. This helps reduce the build context size and improve build performance.

40.What is Docker Trusted Registry (DTR)?

- **Answer**: Docker Trusted Registry (DTR) is an enterprise-grade, on-premises Docker registry solution provided by Docker, Inc. It offers features like role-based access control, image signing, and vulnerability scanning.

41.What is a Docker entrypoint script?

- **Answer**: A Docker entrypoint script is a script specified as the `ENTRYPOINT` in a Dockerfile that initializes and starts the main application. It is useful for setting up the environment and running any necessary commands before starting the main process.

42. What is the difference between a Docker layer and a Docker image?

- **Answer**: A Docker layer is a read-only filesystem layer that is stacked to form a Docker image. Each instruction in a Dockerfile creates a new layer. A Docker image is a collection of layers stacked together to form the complete filesystem for a container.

43. How can you troubleshoot a Docker container that won't start?

- **Answer**: To troubleshoot a Docker container that won't start, you can check the container logs using `docker logs <container_id>`, inspect the container for configuration errors using `docker inspect`

`<container_id>`, and ensure there are no conflicts with ports or resources.

44.What are Docker health checks?

- **Answer**: Docker health checks are used to determine the health status of a container. They are defined in the Dockerfile using the `HEALTHCHECK` instruction and can specify a command that Docker runs to check if the container is healthy.

45.How can you roll back to a previous version of a Docker image?

- **Answer**: To roll back to a previous version of a Docker image, you can pull the desired image version using its tag with `docker pull <image>:<tag>` and then update your containers to use this version.

46.What is Docker Engine?

- **Answer**: Docker Engine is the core part of Docker that enables containerization. It includes the Docker daemon, a REST API, and the CLI for managing containers, images, volumes, and networks.

47.What is the purpose of Docker's `--no-cache` option in the `docker build` command?

- **Answer**: The `--no-cache` option in the `docker build` command forces Docker to build the image without using any cached layers, ensuring that all steps in the Dockerfile are executed.

48.What is the difference between a bind mount and a volume in Docker?

- **Answer**: Bind mounts directly mount a host directory or file into the container, allowing for direct access to host files. Volumes are managed by Docker and provide a more flexible and portable way to persist data.

49.How can you inspect the running processes inside a Docker container?

- **Answer**: You can inspect the running processes inside a Docker container using the `docker top <container_id>` command, which shows the processes running within the container.

50.What is the purpose of the `docker exec` command?

- **Answer**: The `docker exec` command allows you to run a new command in a running container. This can be used to interactively debug a container or run additional commands without stopping the container.

51. What is the purpose of Docker's `--rm` option in the `docker run` command?

- **Answer**: The `--rm` option in the `docker run` command automatically removes the container when it exits, helping to keep the system clean by not leaving stopped containers behind.

52. How can you limit the memory usage of a Docker container?

- **Answer**: You can limit the memory usage of a Docker container using the `-m` or `--memory` flag followed by the memory limit when running the `docker run` command, for example, `docker run -m 512m <image>`.

53. What is the purpose of the `docker inspect` command?

- **Answer**: The `docker inspect` command retrieves detailed information about Docker objects like containers, images, volumes, or networks in JSON format. It helps to examine configuration details and diagnose issues.

54. How can you limit the CPU usage of a Docker container?

- **Answer**: You can limit the CPU usage of a Docker container using the `--cpus` flag followed by the number of CPUs, for example, `docker run --cpus 2 <image>`.

55. What is a Docker service?

- **Answer**: A Docker service is an abstraction that allows you to define and manage a single application container across multiple Docker nodes. It is used in Docker Swarm to scale and maintain application availability.

56. What is the purpose of the `docker service create` command?

- **Answer**: The `docker service create` command is used to create a new service in Docker Swarm. It allows you to define and deploy a containerized application that can be scaled and managed across a cluster of Docker nodes.

57. How can you view the details of a Docker service?

- **Answer**: You can view the details of a Docker service using the `docker service inspect` command followed by the service name or ID. This command provides detailed information about the service configuration and status.

58. What is Docker's `docker stack` command?

- **Answer**: The `docker stack` command is used to deploy and manage a group of related services defined in a Compose file (YAML format) in a Docker Swarm cluster. It allows you to deploy multi-service applications easily.

59. How can you remove a Docker service?

- **Answer**: You can remove a Docker service using the `docker service rm` command followed by the service name or ID.

60. What is a Docker secret and how is it used?

- **Answer**: A Docker secret is a way to securely manage sensitive data, such as passwords or API keys, used by services in a Docker Swarm. Secrets can be created using the `docker secret create` command and used by services by referencing the secret in the service definition.

61. How do you create a Docker network?

- **Answer**: You can create a Docker network using the `docker network create` command followed by the network name. You can specify the network driver and other options as needed.

62. What is a Docker bridge network?

- **Answer**: A Docker bridge network is a default network driver used to connect containers on the same host. It allows containers to communicate with each other and with the host.

63. What is the purpose of the `docker network inspect` command?

- **Answer**: The `docker network inspect` command provides detailed information about a Docker network, including its configuration, connected containers, and IP address range.

64. How can you disconnect a container from a Docker network?

- **Answer**: You can disconnect a container from a Docker network using the `docker network disconnect` command followed by the network name and container ID or name.

65. What is a Docker overlay network?

- **Answer**: A Docker overlay network allows containers running on different Docker hosts to communicate securely. It is used in Docker Swarm to enable communication between services across the cluster.

66. How can you list all Docker networks?

- **Answer**: You can list all Docker networks using the `docker network ls` command.

67. What is Docker Compose's `docker-compose.yml` file?

- **Answer**: The `docker-compose.yml` file is a YAML file used to define and configure multi-container Docker applications. It specifies the services, networks, volumes, and other configurations needed for the application.

68. How can you start services defined in a Docker Compose file?

- **Answer**: You can start services defined in a Docker Compose file using the `docker-compose up` command. This command starts and runs the services as specified in the `docker-compose.yml` file.

69. How can you stop and remove services started by Docker Compose?

- **Answer**: You can stop and remove services started by Docker Compose using the `docker-compose down` command. This command stops the services and removes the containers, networks, and volumes created by `docker-compose up`.

70. What is the purpose of Docker's `--build-arg` option in the `docker build` command?

- **Answer**: The `--build-arg` option in the `docker build` command allows you to pass build-time variables to the Dockerfile. These variables can be used in the Dockerfile to customize the build process.

71. How can you tag a Docker image?

- **Answer**: You can tag a Docker image using the `docker tag` command followed by the image ID or name and the new tag name, for example, `docker tag <image_id> myimage:latest`.

72. How can you push a Docker image to a registry?

- **Answer**: You can push a Docker image to a registry using the `docker push` command followed by the image name and tag, for example, `docker push myimage:latest`.

73. What is the purpose of Docker's `--detach` or `-d` option in the `docker run` command?

- **Answer**: The `--detach` or `-d` option in the `docker run` command runs the container in the background and prints the container ID, allowing the terminal to be used for other commands.

74. How can you attach to a running Docker container?

- **Answer**: You can attach to a running Docker container using the `docker attach` command followed by the container ID or name. This allows you to interact with the container's standard input, output, and error streams.

75. What is Docker's `--volume` or `-v` option in the `docker run` command?

- **Answer**: The `--volume` or `-v` option in the `docker run` command mounts a host directory or Docker volume into the container, for example, `docker run -v /host/path:/container/path <image>`.

76. How can you run a Docker container with a specific hostname?

- **Answer**: You can run a Docker container with a specific hostname using the `--hostname` or `-h` option followed by the desired hostname, for example, `docker run -h myhostname <image>`.

77. What is the purpose of Docker's `--restart` policy?

- **Answer**: The `--restart` policy in Docker specifies how the container should be restarted in case of failure. Policies include `no`, `on-failure`, `always`, and `unless-stopped`.

78. How can you configure a Docker container to restart automatically?

- **Answer**: You can configure a Docker container to restart automatically using the `--restart` flag followed by the desired policy, for example, `docker run --restart=always <image>`.

79. What is the purpose of Docker's `--link` option?

- **Answer**: The `--link` option creates a network connection between two containers, allowing them to communicate with each other. This option is now deprecated in favor of user-defined networks.

80. How can you view the resource usage of Docker containers?

- **Answer**: You can view the resource usage of Docker containers using the `docker stats` command, which provides real-time information on CPU, memory, network, and disk I/O usage for running containers.

81. What is Docker's `--env-file` option in the `docker run` command?

- **Answer**: The `--env-file` option in the `docker run` command allows you to specify a file containing environment variables to be passed to the container, for example, `docker run --env-file myenvfile.env <image>`.

82. How can you create a Docker volume?

- **Answer**: You can create a Docker volume using the `docker volume create` command followed by the volume name, for example, `docker volume create myvolume`.

83. How can you list all Docker volumes?

- **Answer**: You can list all Docker volumes using the `docker volume ls` command.

84. What is the purpose of Docker's `docker volume rm` command?

- **Answer**: The `docker volume rm` command is used to remove one or more Docker volumes by specifying the volume names or IDs.

85. How can you back up a Docker volume?

- **Answer**: You can back up a Docker volume by running a container that mounts the volume and using a command to create a backup, for example, `docker run --rm -v myvolume:/data -v $(pwd):/backup ubuntu tar cvf /backup/backup.tar /data`.

86. How can you restore a Docker volume from a backup?

- **Answer**: You can restore a Docker volume from a backup by running a container that mounts the volume and using a command to extract the backup, for example, `docker run --rm -v myvolume:/data -v $(pwd):/backup ubuntu tar xvf /backup/backup.tar -C /data`.

87. What is Docker Machine?

- **Answer**: Docker Machine is a tool that allows you to provision and manage Docker hosts (virtual machines) on various platforms, such as local hypervisors, cloud providers, and bare-metal servers.

88. How can you create a new Docker Machine?

- **Answer**: You can create a new Docker Machine using the `docker-machine create` command followed by the driver and machine name, for example, `docker-machine create --driver virtualbox mymachine`.

89. What is the purpose of Docker Machine's `docker-machine env` command?

- **Answer**: The `docker-machine env` command displays the environment variables needed to configure the Docker CLI to communicate with a specific Docker Machine. You can use these variables with the `eval` command to set up the environment.

90. How can you list all Docker Machines?

- **Answer**: You can list all Docker Machines using the `docker-machine ls` command.

91. What is Docker Hub?

- **Answer**: Docker Hub is a cloud-based registry service provided by Docker for storing, sharing, and managing Docker images. It includes features like image repositories, automated builds, and team collaboration.

92. How can you search for Docker images on Docker Hub?

- **Answer**: You can search for Docker images on Docker Hub using the `docker search` command followed by the search term, for example, `docker search nginx`.

93. How can you log in to Docker Hub from the Docker CLI?

- **Answer**: You can log in to Docker Hub from the Docker CLI using the `docker login` command followed by your Docker Hub username and password.

94. What is the purpose of Docker's `docker logout` command?

- **Answer**: The `docker logout` command logs you out from a Docker registry, such as Docker Hub, by removing the stored credentials.

95. How can you inspect a Docker image?

- **Answer**: You can inspect a Docker image using the `docker inspect` command followed by the image name or ID. This command provides detailed information about the image, including its layers and metadata.

96. What is Docker's `docker history` command?

- **Answer**: The `docker history` command shows the history of an image, including the layers and the instructions used to create each layer.

97. How can you export a Docker container's filesystem as a tar archive?

- **Answer**: You can export a Docker container's filesystem as a tar archive using the `docker export` command followed by the container ID or name and redirecting the output to a file, for example, `docker export <container_id> > container.tar`.

98. How can you import a tar archive as a Docker image?

- **Answer**: You can import a tar archive as a Docker image using the `docker import` command followed by the tar archive file and the new image name, for example, `docker import container.tar myimage`.

99. What is the purpose of Docker's `docker save` command?

- **Answer**: The `docker save` command saves one or more Docker images and their layers as a tar archive, which can be used for backup or transfer to another Docker host.

100. How can you load a Docker image from a tar archive?

- **Answer**: You can load a Docker image from a tar archive using the `docker load` command followed by the tar archive file, for example, `docker load < image.tar`.