Docker Interview Question & Answers

https://digitalksystems.com

We offer a wide range of comprehensive middleware courses on Udemy, including WebLogic Administration, Oracle OHS Administration, JBoss Administration, IBM WebSphere Administration, SOA Suite Administration, SSL Keys and Keystore Administration, OID Administration, Docker Administration, Kubernetes Administration DevOps and more. These courses are meticulously designed to meet real-world professional requirements, ensuring that the content aligns with what is needed to work in a live production environment. Each course is curated with practical concepts, hands-on labs, and detailed explanations to equip learners with the skills they need to manage complex middleware environments confidently. By taking these courses, learners can avoid paying high fees for training programs that often lack practical application or outcomes. Our courses are aimed at making professionals job-ready in a short time, ensuring that they gain the expertise necessary to excel in middleware technologies without unnecessary expenses or wasted time.

DigiTalk

https://digitalksystems.com/

Reach us at digitalk.fmw@gmail.com
DigiTalk Channel: https://www.youtube.com/channel/UCCGTnI9vvF_ETMhGUXGdFWw
Playlists: https://www.youtube.com/@digitalk.middleware/playlists

DigiTalk Udemy Courses and Coupon Code

Docker Administration

https://www.udemy.com/course/mastering-docker-administration-from-fundamentals-to-expert/?referralCode=12387A8B195F7B90381D

Coupon Code: 46F6E27C19D06CA56B9D

Kubernetes Administration

https://www.udemy.com/course/mastering-kubernetes-administration-from-intro-to-advanced/?referralCode=7E0250BEC21F46192834

Coupon Code: C48311E6276C7678BE45

WebSphere Administration

https://www.udemy.com/course/mastering-ibm-websphere-9x-administration/?referralCode=F372B3EA78D7BEA92416

Coupon Code: D7BB41A951A764F212D2

Mastering SSL (TLS), Keys and Certificates Management

https://www.udemy.com/course/mastering-ssl-tls-keys-and-certificates-management/?referralCode=DB9F1429CB01921969F6

Coupon Code: EDB22E66F664B4920777

SOA Suite Administration

https://www.udemy.com/course/mastering-oracle-soa-suite-12c-administration/?referralCode=2BCF3895067AA7BC22E0

Coupon Code: F0DA9B3FCDED9C6770EF

JBoss 8 Administration

https://www.udemy.com/course/mastering-jboss-eap-8-administration-from-intro-to-advanced/?referralCode=D095CE8B1EE9D065C822

Coupon Code: C5AF753A6DF55F9CE0F9

OHS Administration

https://www.udemy.com/course/mastering-oracle-ohs-http-12c-web-server-administration/?referralCode=165F6A3A8B662CEA1489

Coupon Code: 49830BDBEF3C99624D43

WebLogic Server Administration

https://www.udemy.com/course/oracle-weblogic-server-12c-and-14c-administration/?referralCode=340FBB94DBFEE1A8CC09

Coupon Code: C3B616B82D57F495E4DA

Mastering Oracle Internet Direct (OID) Administration

https://www.udemy.com/course/mastering-oracle-internet-direct-oid-administration/?referralCode=906E9AC0059A29A3CD0D

Coupon Code: 515B41C98AA8F3075ACE

For any query or concerns please write us on digitalk.fmw@gmail.com

1. What is Docker?
Docker is an open-source platform that automates the deployment, scaling, and management of applications by packaging them into lightweight, portable containers. It uses containerization technology to ensure that applications run consistently across different environments.

2. What is a Docker Container?
A Docker container is a lightweight, standalone, executable package that includes everything needed to run a piece of software, including the code, runtime, system tools, libraries, and settings.

3. What is a Docker Image?
A Docker image is a read-only template used to create containers. It contains the application code, libraries, dependencies, and the runtime environment.

4. What is the difference between a Docker Container and a Docker Image?

- **Docker Image**: A static template used to create containers.
- **Docker Container**: A running instance of a Docker image.

5. What is the purpose of Dockerfile?
A Dockerfile is a script that contains a set of instructions to build a Docker image. It specifies the base image, dependencies, environment variables, and commands to execute inside the container.

6. How do you create a Docker container?
You can create a Docker container using the `docker run` command, which starts a container from an image. For example:

```
docker run -it ubuntu
```

7. What is Docker Hub?
Docker Hub is a cloud-based registry service where you can publish and share Docker images. It provides a repository of public and private images for use in Docker.

8. What is the purpose of the `docker build` command?
The `docker build` command is used to build a Docker image from a Dockerfile. The command processes the Dockerfile and produces an image.

9. What is the purpose of the `docker pull` command?
The `docker pull` command downloads a Docker image from a registry (e.g., Docker Hub) to your local system.

10. What is the purpose of the `docker push` command?
The `docker push` command uploads a local Docker image to a registry (e.g., Docker Hub).

11. How can you see the running containers in Docker?
You can list all running containers using the command:

```
docker ps
```

12. How do you stop a Docker container?

You can stop a Docker container using the `docker stop` command followed by the container ID or name:

```
docker stop <container_id>
```

13. What is the `docker exec` command?

The `docker exec` command allows you to run commands inside a running container. For example:

```
docker exec -it <container_id> bash
```

14. How do you remove a Docker container?

You can remove a Docker container using the `docker rm` command:

```
docker rm <container_id>
```

15. How do you remove a Docker image?

You can remove a Docker image using the `docker rmi` command:

```
docker rmi <image_id>
```

16. What is Docker Compose?

Docker Compose is a tool for defining and running multi-container Docker applications. It uses a YAML file to configure the services, networks, and volumes for a containerized application.

17. What is the `docker-compose up` command?

The `docker-compose up` command is used to start up all the services defined in a `docker-compose.yml` file. It creates containers, networks, and volumes as necessary.

18. What is the `docker-compose down` command?

The `docker-compose down` command is used to stop and remove all the containers, networks, and volumes defined in a `docker-compose.yml` file.

19. What are Docker Volumes?

Docker Volumes are used to persist data in containers. Volumes allow data to exist outside the container's lifecycle, so the data is not lost when the container is stopped or removed.

20. How do you mount a volume in Docker?

You can mount a volume in Docker using the `-v` flag during the `docker run` command:

```
docker run -v /path/to/host/dir:/path/to/container/dir <image_name>
```

21. What is Docker Networking?
Docker Networking allows containers to communicate with each other and the outside world. Docker provides several network drivers such as bridge, host, overlay, and none.

22. What is the default Docker network?
The default network driver in Docker is the `bridge` network. When no network is specified, containers are connected to the bridge network by default.

23. What is a Docker Bridge Network?
A Docker Bridge Network is a default network driver that creates an isolated network on the host, allowing containers to communicate with each other and the host machine.

24. What is a Docker Host Network?
The Docker Host Network driver connects containers directly to the host network. Containers use the host's IP address and share the network stack with the host.

25. What is the purpose of the `docker network ls` command?
The `docker network ls` command lists all available networks in Docker.

26. What is a Docker Overlay Network?
An Overlay Network allows containers on different Docker hosts to communicate with each other, typically used in Docker Swarm and Kubernetes for multi-host networking.

27. What is the `docker info` command used for?
The `docker info` command provides detailed information about the Docker installation, including system information, number of containers, number of images, and storage driver.

28. What is the `docker logs` command used for?
The `docker logs` command is used to fetch the logs of a running or stopped container:

```
docker logs <container_id>
```

29. What is Docker Swarm?
Docker Swarm is Docker's native clustering and orchestration tool. It allows you to manage a cluster of Docker nodes (machines) and deploy applications across them.

30. What is the purpose of the `docker swarm init` command?
The `docker swarm init` command is used to initialize a Docker Swarm cluster on the current host, making it the manager node of the Swarm.

31. What is the purpose of the `docker service` command in Docker Swarm?
The `docker service` command is used to create and manage services in a Docker Swarm. A service is a set of tasks that run the same container image across the cluster.

32. What is the purpose of the `docker stack` command in Docker Swarm?

The `docker stack` command is used to deploy and manage multi-container applications using a `docker-compose.yml` file in a Docker Swarm.

33. What is the `docker attach` command used for?

The `docker attach` command is used to attach your terminal to a running container. It allows you to interact with the container's input and output.

34. How can you inspect a running container in Docker?

You can inspect a running container using the `docker inspect` command:

```
docker inspect <container_id>
```

35. What is the difference between `docker run` and `docker exec`?

- **docker run**: Creates and starts a new container.
- **docker exec**: Executes a command inside an already running container.

36. How do you build a Docker image from a Dockerfile?

You can build a Docker image using the `docker build` command:

```
docker build -t <image_name> .
```

37. What is the Dockerfile command `FROM` used for?

The `FROM` command specifies the base image for the Docker image. For example, `FROM ubuntu:20.04` means the image is based on the Ubuntu 20.04 image.

38. What is the Dockerfile command `RUN` used for?

The `RUN` command executes commands during the build process of the Docker image. For example, `RUN apt-get update` installs dependencies.

39. What is the Dockerfile command `CMD` used for?

The `CMD` command provides the default command that is executed when a container starts. It can be overridden at runtime.

40. What is the Dockerfile command `ENTRYPOINT` used for?

The `ENTRYPOINT` command specifies the executable to run when a container starts. It cannot be overridden by default but can be combined with `CMD`.

41. What is the difference between `CMD` and `ENTRYPOINT` in a Dockerfile?

- **CMD**: Specifies the default command and arguments that can be overridden at runtime.
- **ENTRYPOINT**: Specifies the command to be executed when the container starts and cannot be overridden.

42. What is Docker's layered filesystem?
Docker images are built in layers, where each layer represents a change or command in the Dockerfile. Layers are cached and reused to optimize build times.

43. What is Docker's Union File System (UFS)?
The Union File System (UFS) allows Docker to combine multiple layers into a single, unified filesystem, providing benefits like shared layers and efficient storage.

44. How does Docker differ from virtual machines (VMs)?
Docker containers share the host OS kernel, making them lightweight and faster to start compared to VMs, which run their own operating systems.

45. What is a Docker Registry?
A Docker registry is a place where Docker images are stored and distributed. Docker Hub is the default public registry.

46. How do you create a Docker network?
You can create a Docker network using the `docker network create` command:

```
docker network create my_network
```

47. How do you check the status of Docker services?
You can check the status of Docker services using the `docker info` or `docker service ls` command.

48. What is the `docker stats` command used for?
The `docker stats` command provides real-time resource usage statistics for running containers.

49. What is Docker's `docker cp` command?
The `docker cp` command allows you to copy files between a Docker container and the host machine.

50. What is Docker's `docker save` command?
The `docker save` command is used to save a Docker image to a tarball archive.

51. What is Docker's `docker load` command?
The `docker load` command is used to load a Docker image from a tarball archive.

52. How can you secure Docker containers?
You can secure Docker containers by limiting container privileges, using user namespaces, ensuring proper networking settings, and keeping containers up-to-date with security patches.

53. How does Docker handle resource limits for containers?
Docker allows you to set resource limits for containers, such as CPU, memory, and block IO, using flags like `--memory`, `--cpu-shares`, and `--blkio-weight`.

54. What is Docker's `docker update` command?
The `docker update` command is used to update the configuration of a running container, such as resource limits.

55. What is Docker's `docker volume` command?
The `docker volume` command allows you to manage Docker volumes, which are used for persistent storage across containers.

56. What is the difference between `docker volume` and `docker bind mount`?

- **Docker Volume**: Managed by Docker, used for persistent storage.
- **Docker Bind Mount**: Mounts a file or directory from the host filesystem directly into a container.

57. What is a multi-stage build in Docker?
A multi-stage build allows you to use multiple `FROM` statements in a Dockerfile, enabling you to build an image in stages and reduce the final image size.

58. How can you update a Docker image?
To update a Docker image, you can pull the latest version from a registry or build a new image from an updated Dockerfile.

59. How can you tag Docker images?
You can tag Docker images using the `docker tag` command:

```
docker tag <image_id> <new_image_name>:<tag>
```

60. How do you create a Docker container from a Docker image?
You can create and start a Docker container using the `docker run` command:

```
docker run <image_name>
```

61. What is Docker's "Dockerfile Cache"?
Docker uses caching to speed up the build process. If a layer has been built previously, Docker will reuse it instead of rebuilding it.

62. What are some of the Docker image formats?
Docker images are stored in a format called the Docker Image Format, which includes layers and metadata.

63. How do you handle Docker container logs?
You can view Docker container logs using the `docker logs` command or configure centralized logging solutions like ELK (Elasticsearch, Logstash, and Kibana).

64. How do you use Docker with Kubernetes?
Docker is the default container runtime for Kubernetes, and Kubernetes orchestrates Docker containers for automated deployment, scaling, and management.

65. How do you configure Docker to start on boot?
You can configure Docker to start on boot using system services like `systemd`. For example:

```
sudo systemctl enable docker
```

66. How do you run a Docker container in detached mode?
You can run a Docker container in detached mode by using the `-d` flag:

```
docker run -d <image_name>
```

67. What are the limitations of Docker?
Some limitations of Docker include resource isolation limits, security concerns, and challenges with stateful applications.

68. What is the purpose of `docker-compose.yml`?
The `docker-compose.yml` file is used to define and configure multiple containers in a multi-container Docker application, specifying the services, networks, and volumes.

69. What is the `docker-compose logs` command used for?
The `docker-compose logs` command is used to view logs for all services defined in a `docker-compose.yml` file.

70. What is the `docker-compose build` command used for?
The `docker-compose build` command is used to build images for services defined in the `docker-compose.yml` file.

71. How do you scale a Docker service?
You can scale a Docker service using the `docker service scale` command:

```
docker service scale <service_name>=<replica_count>
```

72. How do you specify the environment variables in Docker?
You can specify environment variables in Docker using the `-e` flag during the `docker run` command:

```
docker run -e VAR_NAME=value <image_name>
```

73. What is a Docker Registry Mirror?
A Docker Registry Mirror is a proxy for Docker Hub, allowing faster image pulls by caching frequently used images.

74. How do you use a Docker container with a specific port mapping?
You can map a container port to a host port using the `-p` flag:

```
docker run -p <host_port>:<container_port> <image_name>
```

75. What is Docker's `docker volume` command used for?
The `docker volume` command is used to manage Docker volumes, which allow containers to persist data outside their lifecycle.

76. What is the purpose of `docker swarm`?
Docker Swarm is used to manage a cluster of Docker nodes, ensuring high availability, scalability, and load balancing of services across the cluster.

77. How do you deploy a service in Docker Swarm?
You can deploy a service in Docker Swarm using the `docker service create` command:

```
docker service create --name <service_name> <image_name>
```

78. How do you monitor Docker containers?
You can monitor Docker containers using commands like `docker stats` or third-party monitoring tools like Prometheus and Grafana.

79. What is Docker's `docker-compose up` command?
The `docker-compose up` command is used to start up all services defined in a `docker-compose.yml` file. It creates containers, networks, and volumes as necessary.

80. How do you list Docker images?
You can list all Docker images on your system using the `docker images` command.

81. How do you assign a name to a Docker container?
You can assign a name to a Docker container using the `--name` flag during the `docker run` command:

```
docker run --name <container_name> <image_name>
```

82. How do you get the IP address of a running Docker container?
You can get the IP address of a running Docker container using the `docker inspect` command:

```
docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}'
<container_name>
```

83. What is the `docker info` command used for?
The `docker info` command provides detailed information about the Docker installation, including the number of containers, images, and storage driver used.

84. How do you configure logging for Docker containers?
Docker supports various logging drivers, including `json-file`, `syslog`, `fluentd`, and more, which can be specified during container creation.

85. How do you run a Docker container with a specific environment variable?
You can set environment variables during container creation using the `-e` flag:

```
docker run -e ENV_VAR=value <image_name>
```

86. What is the purpose of Docker Volumes?
Docker volumes are used to persist data across container restarts. They can be shared between containers and provide a way to manage data outside the container lifecycle.

87. How do you handle Docker container logs?
Docker container logs can be viewed using the `docker logs` command. You can also configure centralized logging solutions for better management.

88. What is the `docker logs` command?
The `docker logs` command is used to fetch logs from a running or stopped container.

89. How do you interact with a Docker container?
You can interact with a Docker container using the `docker exec` command:

```
docker exec -it <container_id> bash
```

90. How do you create a Docker container from a Docker image?
You can create a container using the `docker run` command. Example:

```
docker run -it ubuntu bash
```

91. How do you configure Docker container storage?
Docker storage is managed through volumes and bind mounts. Volumes are stored in a Docker-managed location, while bind mounts reference specific locations on the host filesystem.

92. How do you check the status of a Docker container?
You can check the status of a Docker container using `docker ps`. This shows all running containers, their status, and other details.

93. What is Docker Swarm Mode?
Swarm mode is Docker's native clustering and orchestration tool, enabling you to manage a multi-host environment and deploy services with high availability.

94. How do you secure Docker containers?
You can secure Docker containers by limiting privileges, using Docker's security features like namespaces, and using security tools like Docker Content Trust.

95. What is Docker's "build cache"?
Docker uses a build cache to speed up the image build process by reusing layers that have not changed since the last build.

96. How can you push a Docker image to a registry?
You can push a Docker image to a registry using the `docker push` command:

```
docker push <image_name>
```

97. How do you list Docker networks?

You can list Docker networks using the `docker network ls` command.

98. How do you create a custom Docker network?

You can create a custom Docker network using the `docker network create` command:

```
docker network create my_network
```

99. What is a Docker Overlay Network?

An Overlay network is used to connect Docker containers across multiple hosts in a Docker Swarm or Kubernetes cluster.

100. How do you restart a Docker container?

You can restart a Docker container using the `docker restart` command:

```
docker restart <container_id>
```

DISCLAIMER AND CONSENT

This document is being provided by DigiTalk as part of its effort to assist users in understanding and working with Middleware and DevOps. While every effort has been made to ensure the accuracy and reliability of the information presented in this document, there is a possibility of typographical errors or inaccuracies. DigiTalk does not guarantee the correctness or completeness of the content provided in this document.

Users of this document are encouraged to cross-reference the information presented here with official documentation available on their website or other authoritative sources. Any discrepancies or inaccuracies found in this document should be reported to us at digitalk.fmw@gmail.com.

By using this document, you acknowledge and consent to the following:

This document is not officially endorsed or verified by any other third party organization.

We do not claim or guarantees about the accuracy or suitability of the information contained in this document.

Users are responsible for verifying and validating any information presented here for their specific use case.

DigiTalk disclaims any liability for any errors, omissions, or damages that may result from the use of this document.

If you discover any inaccuracies or errors in this document, please report them to digitalk.fmw@gmail.com, and the Company will endeavor to correct them as necessary.

This consent statement is provided to ensure transparency and understanding of the limitations of the information contained in this document. By using this document, you agree to abide by the terms and conditions outlined herein.