DOCKER

COMPLETE GUIDE TO DOCKER FOR BEGINNERS AND INTERMEDIATES

Questions and Answers

DAY 3

MASTER DEVOPS WITH VINAY

21: How can you remove unused Docker volumes?

Answer:

To remove Docker volumes that are no longer in use, you can use the docker volume prune command.

Command:

```
docker volume prune
```

Explanation:

- This command deletes all volumes that are not currently associated with any container.
- To remove a specific volume instead of all unused ones, use:
  `docker volume rm <volume_name>`

22. What is Docker Compose?

Answer:

Docker Compose is a tool that simplifies the definition and management of multi-container Docker applications using a YAML file (docker-compose.yml).

Key Features:

1. Multi-Container Management – Define and run multiple containers as services in a single file.
2. Dependency Handling – Ensures proper startup order using the depends_on directive.
3. Automatic Networking – Creates a dedicated network for the defined services.

Basic Commands:

- Start all services:
  `docker-compose up`
- Stop and remove containers, networks, and volumes:
  `docker-compose down`

Example docker-compose.yml:

```
version: '3.8'
services:
 app:
   image: my-app
   ports:
     - "8080:80"

 db:
   image: mysql
   environment:
     MYSQL_ROOT_PASSWORD: example
```

This setup allows you to manage your entire application stack with a single file and command.

23. How can you list all Docker volumes?

Answer:

To view all Docker volumes on your system, use the docker volume ls command.

Command:

```
docker volume ls
```

Output:

- Shows a list of all volumes along with their associated drivers.
- Useful for identifying existing volumes and their usage.

24. How do you scale services with Docker Compose?

Answer:

Docker Compose enables you to scale a service by running multiple instances (replicas) of it.

Command: `docker-compose up --scale <service-name>=<number-of-replicas>`

Example: `docker-compose up --scale web=3`

- This command launches 3 replicas of the web service.

Note:

- Scaling is not supported for services that use the depends_on directive.

25. How can you check the status of services in Docker Compose?

Answer:

To view the current status of all services managed by Docker Compose, use the docker-compose ps command.

Command:

```
docker-compose ps
```

Output:

- Shows details such as service name, container ID, status (e.g., running, exited), and port mappings.

26. Can you restart all services in a Docker Compose application? If yes, how?

Answer:

Yes, you can restart all services defined in a Docker Compose application using the docker-compose restart command.

Command:

```
docker-compose restart
```

Explanation:

- Restarts all currently running services specified in the docker-compose.yml file.
- To restart a specific service only, use:
  `docker-compose restart <service-name>`

27 .How can you pass environment variables to Docker Compose services?

Answer:

You can provide environment variables to Docker Compose services in multiple ways:

1. Define Directly in docker-compose.yml:

```
services:
 app:
   image: my-app
   environment:
     - ENV_VAR_NAME=value
```

2. Use an .env File:

Create a file named .env:

```
DB_USER=root
DB_PASS=password
```

Reference it in your docker-compose.yml:

```
services:
 db:
   image: mysql
   environment:
     - MYSQL_USER=${DB_USER}
     - MYSQL_PASSWORD=${DB_PASS}
```

3. Pass Variables via Command Line:

```
DB_USER=root DB_PASS=password docker-compose up
```

Each method offers flexibility depending on your use case and environment.

28. What is the docker prune command, and when should you use it?

Answer:

The docker prune command is used to remove unused Docker resources such as stopped containers, dangling images, unused networks, and volumes, helping to reclaim disk space.

1. Common Variants:
   Remove all unused containers, networks, images, and build cache:
   ```
   docker system prune
   ```
2. Remove unused volumes only:
   ```
   docker volume prune
   ```
3. Remove dangling (unused) images:
   ```
   docker image prune
   ```

When to Use:

- Use docker prune commands when you want to clean up unnecessary Docker artifacts and free up system storage.

29. What is the purpose of the .dockerignore file?

Answer:

The .dockerignore file is used to specify files and directories that should be excluded from the Docker build context. This helps streamline the image build process by avoiding the inclusion of unnecessary or sensitive files.

Example .dockerignore file:

```
node_modules
*.log
.env
```

Use Cases:

- Preventing sensitive files (e.g., .env) from being added to the image.
- Reducing the build context size, leading to faster builds and smaller images.

30. How do you manage and use private Docker registries?

Answer:

Private Docker registries allow organizations to securely store, manage, and share Docker images within internal or restricted environments.

Steps to Set Up and Use a Private Registry:

1. Start a Private Registry:
   ```
   docker run -d -p 5000:5000 --name registry registry:2
   ```
2. Tag and Push an Image to the Registry:
   ```
   docker tag my-app localhost:5000/my-app
   docker push localhost:5000/my-app
   ```
3. Pull an Image from the Private Registry:
   ```
   docker pull localhost:5000/my-app
   ```
4. Use Authentication for Security:
   - Secure access by logging in to the registry:
     ```
     docker login <registry-url>
     ```
   - This stores your Docker credentials and enables authenticated image push/pull operations.

Note:

For production environments, it's recommended to use a fully managed private registry service (like AWS ECR, Azure ACR, or Docker Hub Private Repos) and enable TLS for secure communication.