Docker Compose

npNOG 10

November 25 - 28, 2024

Docker Compose

Docker Compose is a tool for defining and running multi-container applications.

- It is the key to unlocking a streamlined and efficient development and deployment experience.
- Compose simplifies the control of your entire application stack
  - making it easy to manage services
  - networks
  - volumes
  - in a single, comprehensible YAML configuration file
- Then, with a single command, you create and start all the services from your configuration file.
- Compose works in all environments
  - production, staging, development, testing, as well as CI workflows.
- It also has commands for managing the whole lifecycle of your application:
  - Start, stop, and rebuild services
  - View the status of running services
  - Stream the log output of running services
  - Run a one-off command on a service

Why use Compose?

Using Docker Compose offers several benefits that streamline the development, deployment, and management of containerized applications:

- **Simplified control**
  - Docker Compose allows you to define and manage multi-container applications in a single YAML file.
  - This simplifies the complex task of orchestrating and coordinating various services, making it easier to manage and replicate your application environment.
- **Efficient collaboration**
  - Docker Compose configuration files are easy to share, facilitating collaboration among developers, operations teams, and other stakeholders.
  - This collaborative approach leads to smoother workflows, faster issue resolution, and increased overall efficiency.

Why use Compose? (contd.)

- **Rapid application development**
  - Compose caches the configuration used to create a container.
  - When you restart a service that has not changed, Compose re-uses the existing containers.
  - Re-using containers means that you can make changes to your environment very quickly.
- **Portability across environments**
  - Compose supports variables in the Compose file.
  - You can use these variables to customize your composition for different environments, or different users.
- **Extensive community and support**
  - Docker Compose benefits from a vibrant and active community, which means abundant resources, tutorials, and support.
  - This community-driven ecosystem contributes to the continuous improvement of Docker Compose and helps users troubleshoot issues effectively.

How Compose works?

With Docker Compose you use a YAML configuration file, known as the Compose file, to configure your application's services, and then you create and start all the services from your configuration with the Compose CLI.

Compose works by creating a YAML file that defines the services, networks, and volumes for your application.

Docker Compose file (Sample)

```yaml
version: '3.8'

services:
  web:
    image: nginx:latest
    ports:
      - "80:80"
    volumes:
      - ./web:/usr/share/nginx/html
    depends_on:
      - redis

  redis:
    image: redis:latest
    ports:
      - "6379:6379"
```

The Compose file

The default path for a Compose file is `compose.yaml` (preferred) or `compose.yml` that is placed in the working directory. Compose also supports `docker-compose.yaml` and `docker-compose.yml` for backwards compatibility of earlier versions. If both files exist, Compose prefers the canonical `compose.yaml`.

Here is a breakdown of the Compose file:

- `version`: The version of the Compose file format.
- `services`: A list of services that make up your application.
- `networks`: A list of networks that your services use.
- `volumes`: A list of named volumes that your services use.
- `depends_on`: A list of services that your services depend on.
- `environment`: A list of environment variables that your services use.
- `ports`: A list of ports that your services use.
- `expose`: A list of ports that your services expose.
- `restart`: A restart policy that your services should use.

CLI

The Docker CLI lets you interact with your Docker Compose applications through the `docker compose` command, and its subcommands. Using the CLI, you can manage the lifecycle of your multi-container applications defined in the `compose.yaml` file. The CLI commands enable you to start, stop, and configure your applications effortlessly.

Key commands

- To start all the services defined in your compose.yaml file:

```
docker compose up
```

- To stop and remove the running services:

```
docker compose down
```

- If you want to monitor the output of your running containers and debug issues, you can view the logs with:

```
docker compose logs
```

- To lists all the services along with their current status:

```
docker compose ps
```