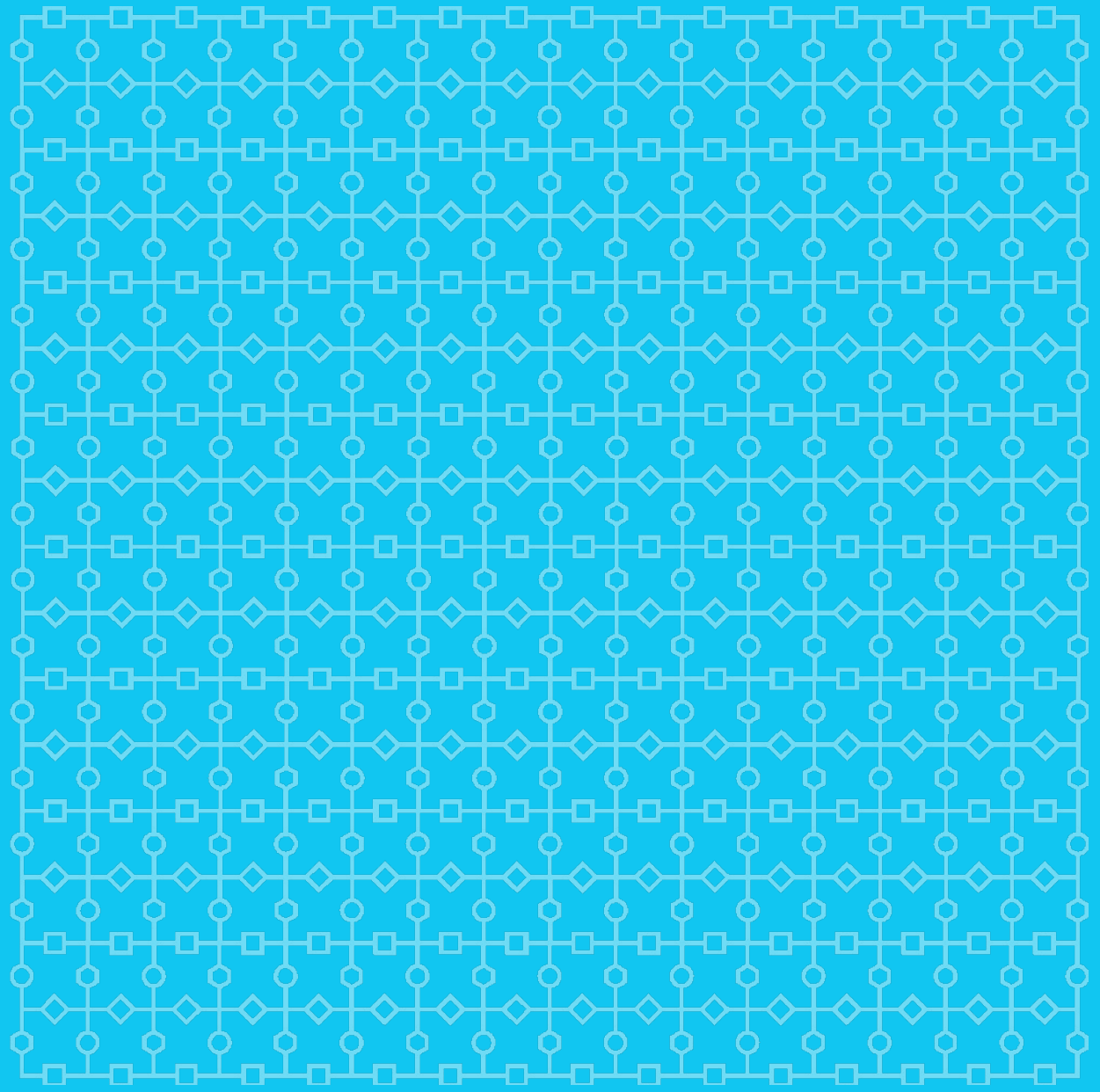
Software Engineering



TECH ACADEMY

Docker Compose

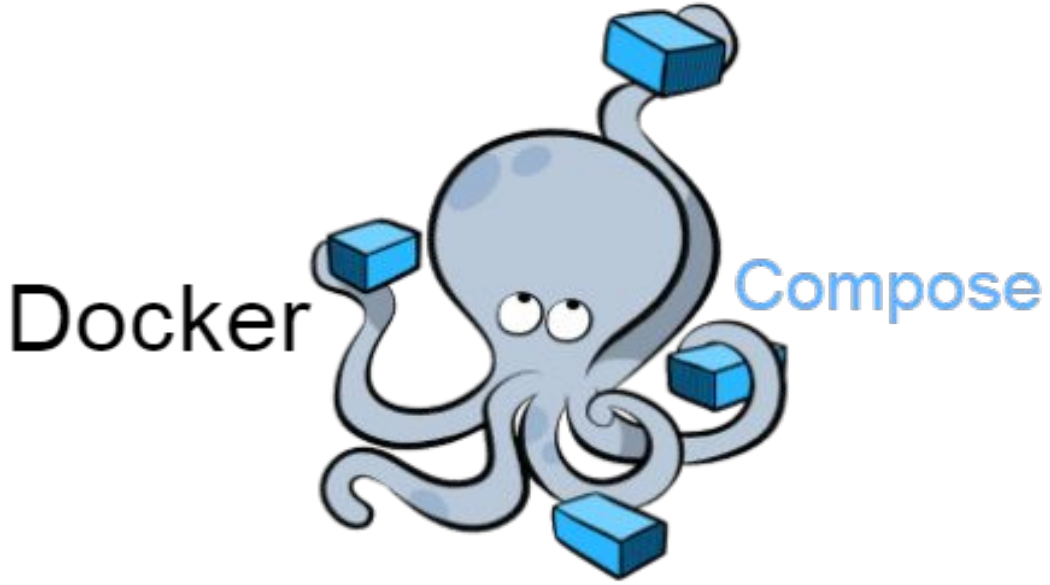
Instructor: Idris Shabanlı

TECH ACADEMY

Table of content

What is Docker Compose

Docker Compose is a Docker tool used to define and run multi-container applications. With Compose, you use a **YAML** file to configure your application's services and create all the app's services from that configuration. Think of docker-compose as an automated multi-container workflow.

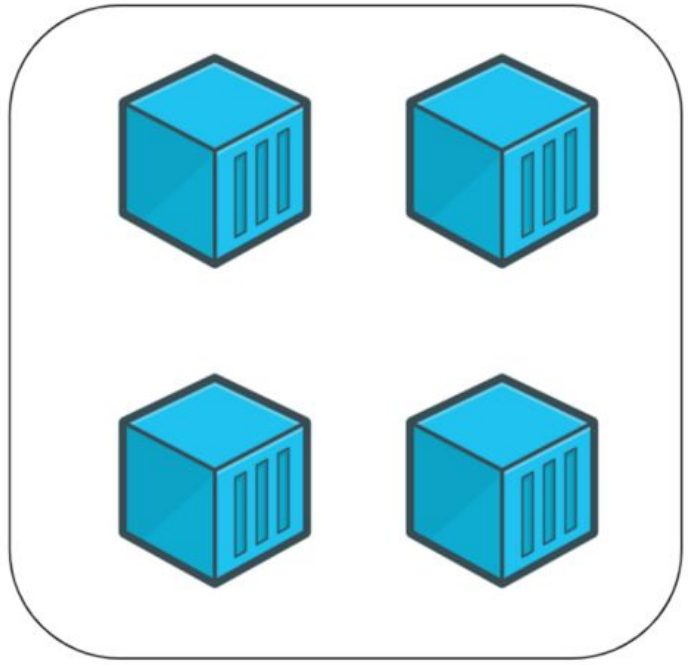


Features of Docker Compose

According to the Docker documentation, the most popular features of Docker Compose are:

- Multiple isolated environments on a single host
- Preserve volume data when containers are created
- Only recreate containers that have changed
- Variables and moving a composition between environments
- Orchestrate multiple containers that work together

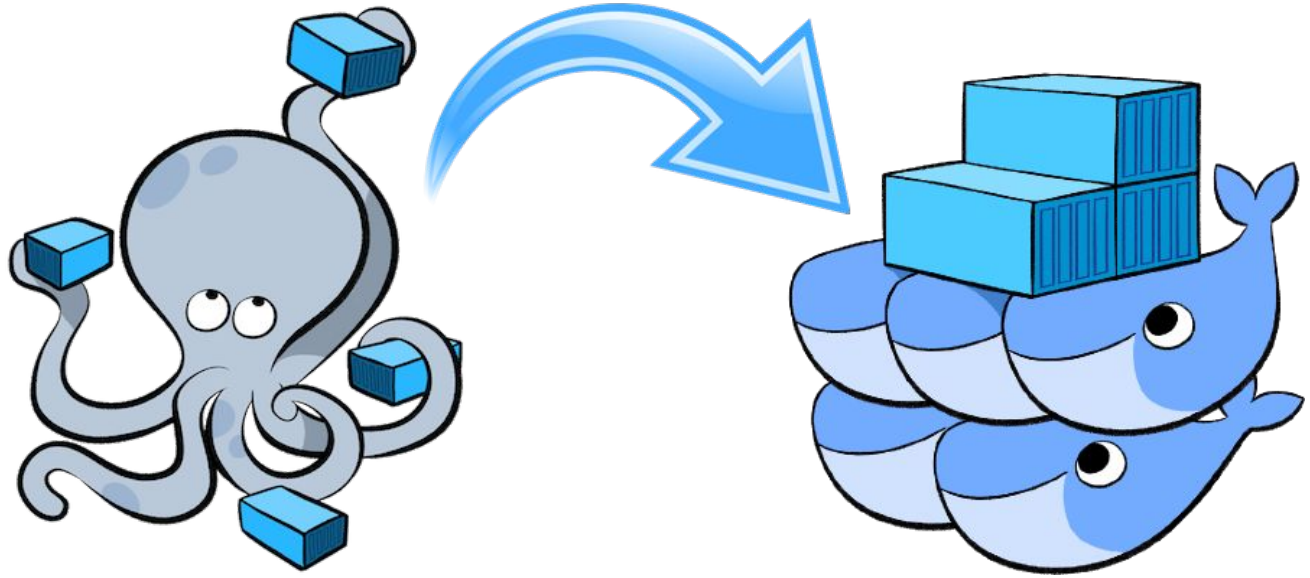
Docker



Docker-Compose

How to use and install Docker Compose

Compose uses the Docker Engine, so you'll need to have the Docker Engine installed on your device. You can run Compose on Windows, Mac, and 64-bit Linux. Installing Docker Compose is actually quite easy.



Docker Compose file structure

In short, Docker Compose files work by applying mutiple commands that are declared within a single docker-compose.yml configuration file. The basic structure of a Docker Compose YAML file looks like this:

```
version: '3'

services:
  redis:
    image: redis
  web:
    build: .
    command: python3 app.py
    volumes:
      - .:/code
    ports:
      - "8000:8000"
    depends_on:
      - redis
```

Docker compose file instructions

- **build**: Describes configurations applied at build time. It can be just a pathname, as shown here, or it can provide details such as a Dockerfile to use (instead of the default Dockerfile in the directory) or arguments to pass to the Dockerfile during the build process.
- **command**: A command to run when starting the container. This overrides the CMD statement supplied in the container's Dockerfile, if any.
- **volumes**: Any paths to volumes to be mounted for this service. You can also specify volumes as a top-level configuration option and reuse any volumes defined there across multiple containers in the docker-compose.yml file.

Docker compose file instructions

- **ports**: Port mappings for the container. You can use a simple format, as shown here, or a more detailed format that describes which protocols to use.
- **depends_on**: Describes the order of dependencies between services. Here, because web depends on redis, redis must be brought up first when Docker Compose starts the app.
- **environment**: The clause allows us to set up an environment variable in the container. This is the same as the -e argument in Docker when running a container.

Docker Compose commands

- docker-compose build
  - This command builds images in the docker-compose.yml file
- docker-compose images
  - This command will list the images you've built using the current docker-compose file
- docker-compose stop
  - This command stops the running containers of specified services
- docker-compose run
  - This is similar to the docker run command. It will create containers from images built for the services mentioned in the compose file
- docker-compose up
  - This command does the work of the docker-compose build and docker-compose run commands

Docker Compose commands

- **docker-compose ps**
  - This command list all the containers in the current docker-compose file. They can then either be running or stopped.
- **docker-compose down**
  - This command is similar to the docker system prune command. However, in Compose, it stops all the services and cleans up the containers, networks, and images.

Resources

- Docker Compose Documentation
- Introduction to Docker Compose
- Docker Compose Tutorial: advanced Docker made simple
- “What, Why, How” Docker Compose?
- 5 Common Mistakes When Writing Docker Compose
- Understanding Docker and Docker Compose

Please take notes for questions

THANK YOU!