Docker 101

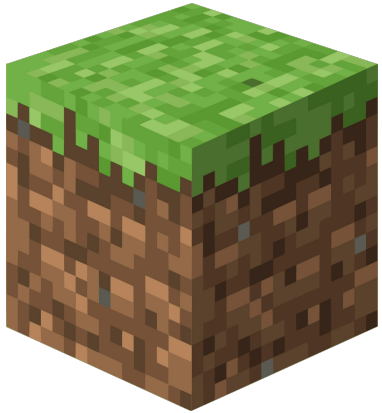
workshop

Miguel Matos

clone the repo

git clone git@github.com:mankings/workshop-docker.git

a story about a minecraft server

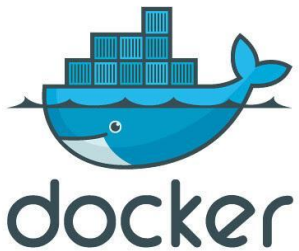


Containers

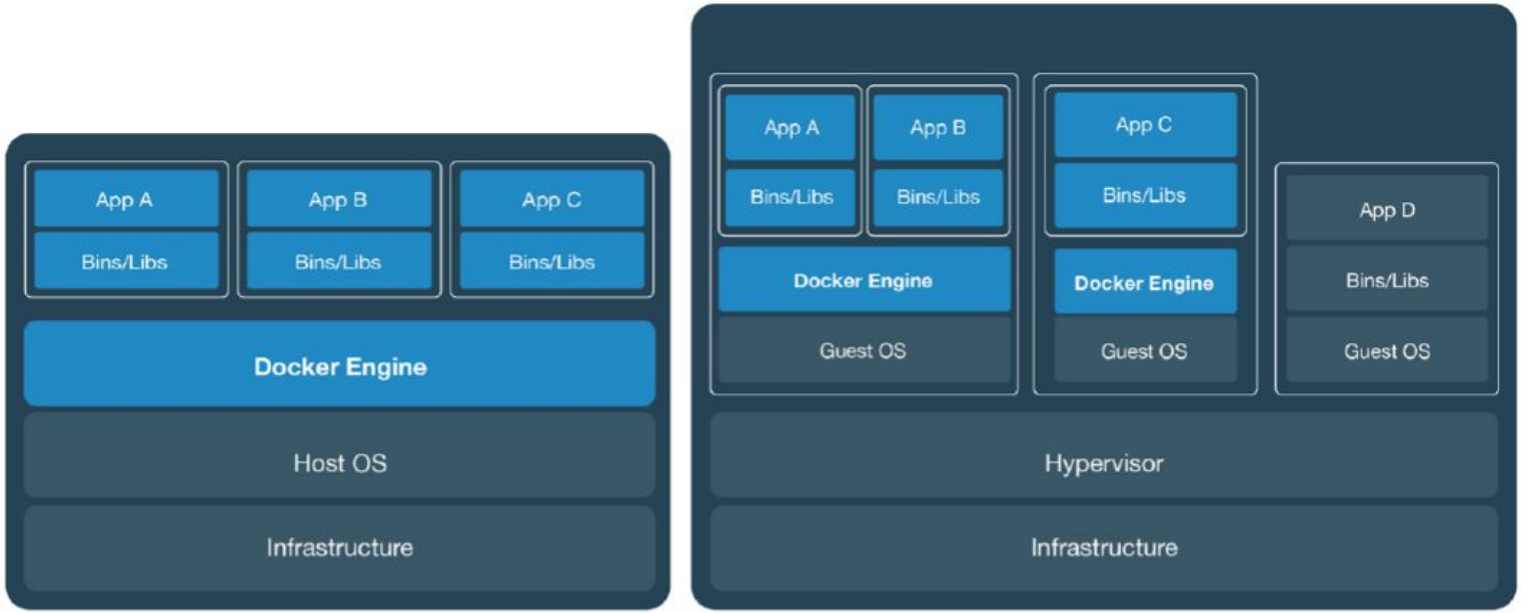
Container

- Lightweight, executable units of software that contain everything needed to run an application.

Docker is a platform designed to make it easier to create, deploy, and run applications by using containers.



Containers vs VMs



Containers

Virtual Machines

Why containers

Consistency

- "Works on my machine" issue resolved.

Isolation

- Applications run in separate containers without affecting each other.

Portability

- Run anywhere: on-premises, cloud, or hybrid environments.

Efficiency

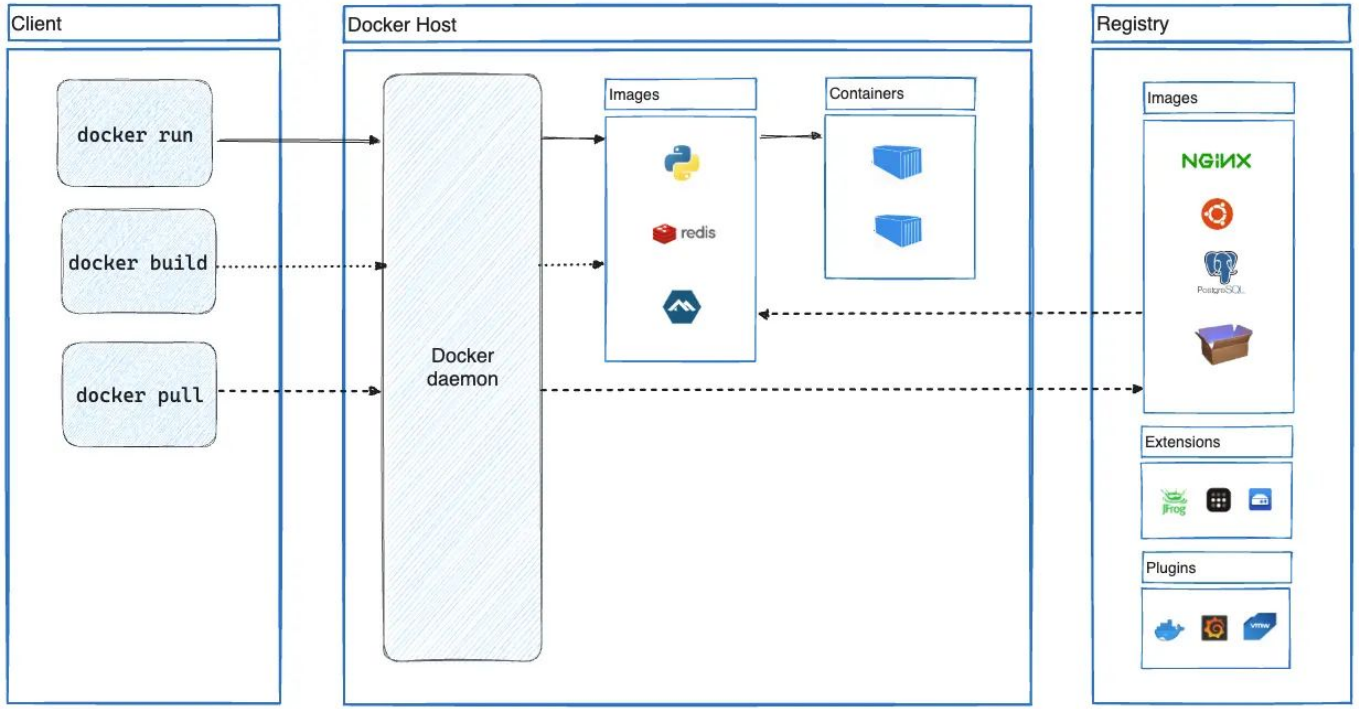
- Lightweight compared to traditional VMs, uses fewer resources.

Docker real-world uses

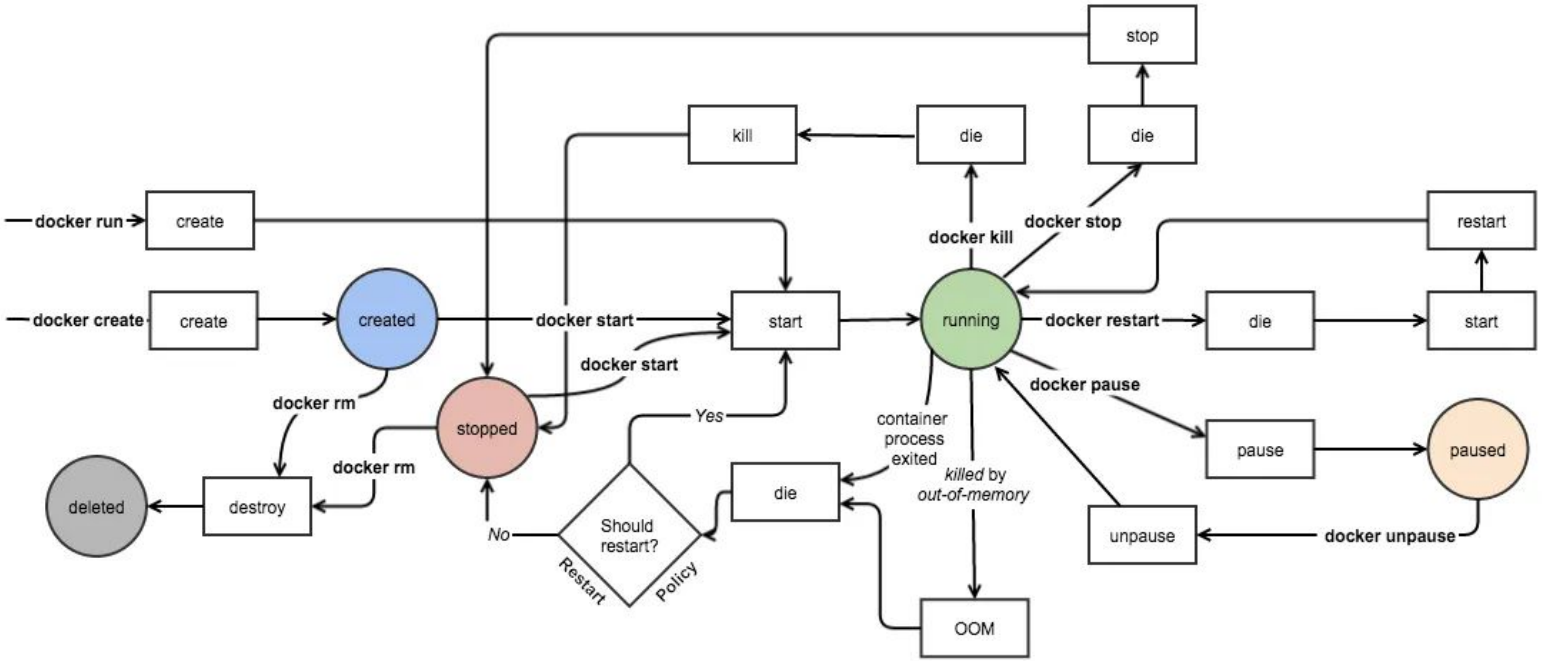
- Netflix
- Spotify
- Pinterest
- The New York Times
- …

- IES
- TQS
- CBD
- SIO
- TPW
- …

Docker Architecture



Container Lifecycle



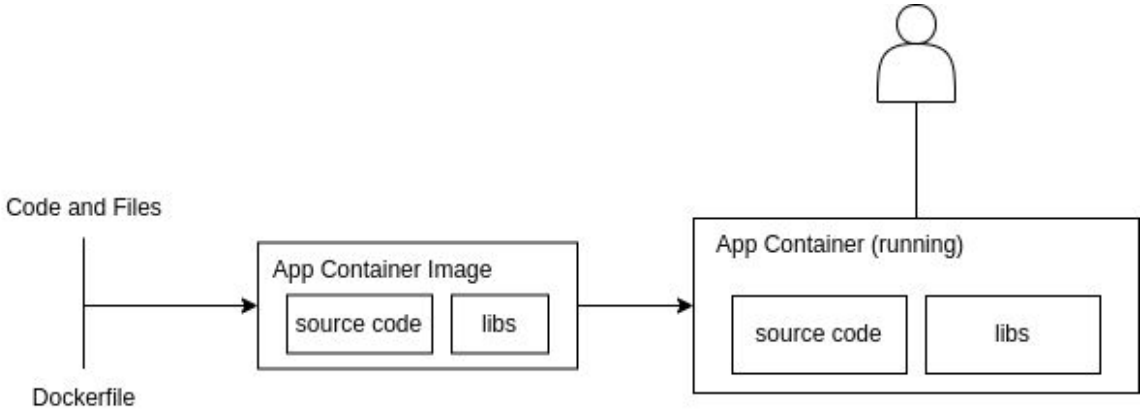
Docker Basics

Basic Commands

- docker pull - pull an image from a registry
- docker run - run a container
- docker ps - list running containers
- docker stop - stop a container
- docker rm - remove/delete a container
- docker build - build an image
- docker rmi - remove an image
- docker exec - execute a command inside a container
- docker logs - check or follow the logs of a container

Example 1 - Simple Application

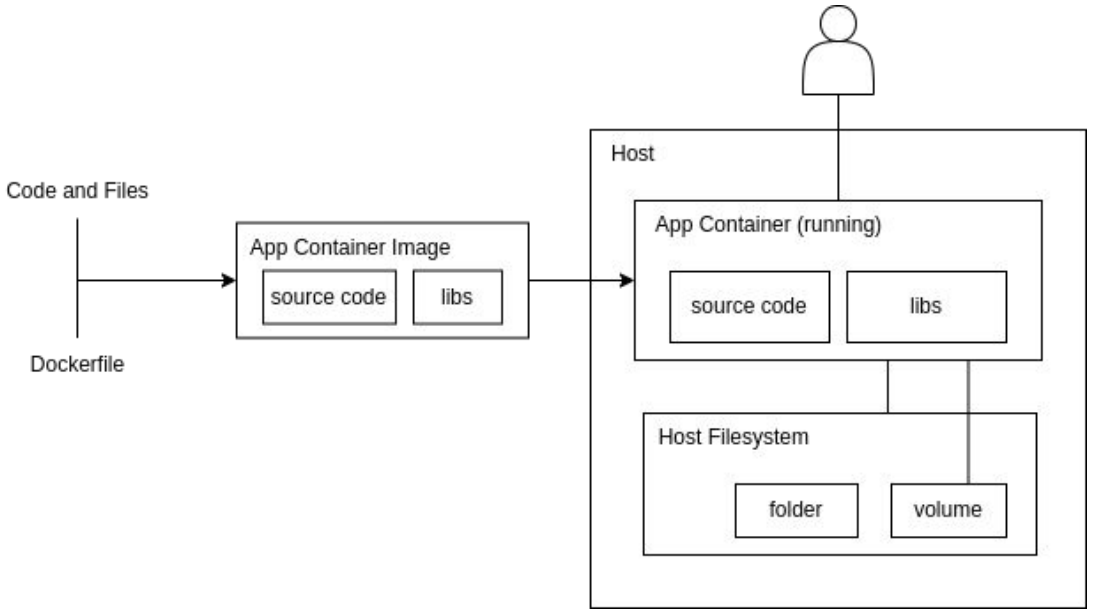
- *app.py* - application source code
- *requirements.txt* - required libraries/packages
- *Dockerfile* - instructions to build the container image



Example 2 - Persistence

Containers are ephemeral - data within them is lost upon stop or destruction

- Volumes
- Bind mounts



Docker Compose

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

Uses .yaml configuration files, defining the following:

- Services
- Networking
- Volumes

Docker Compose basics

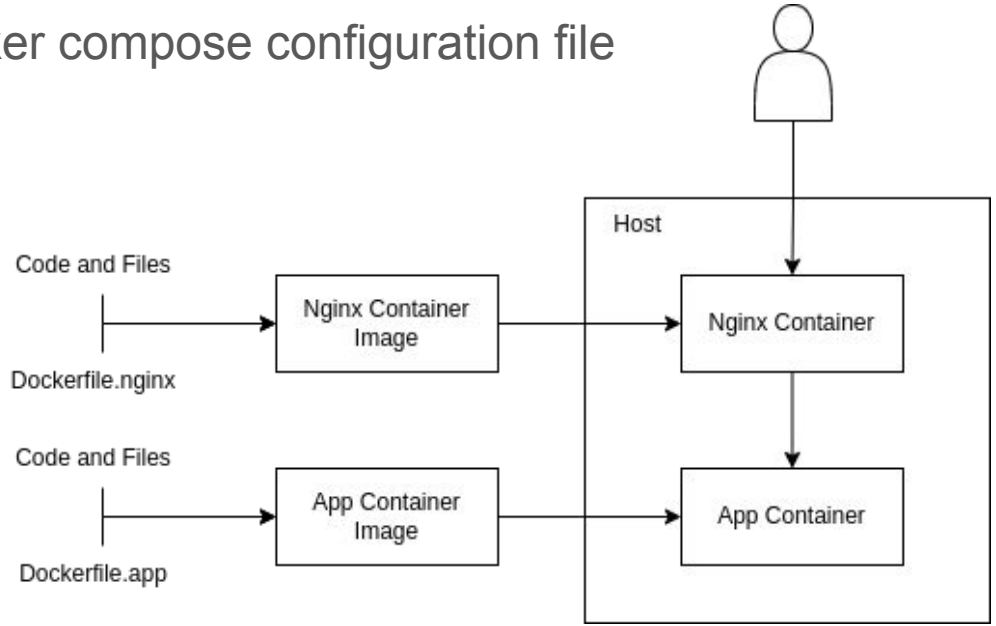
Basic Commands

- docker compose up [-d] - launch services
- docker compose down - stop services

- docker compose ps - list services
- docker compose pull - pull the image of a service
- docker compose build - build service images
- …

Example 3 - Multi-container Application

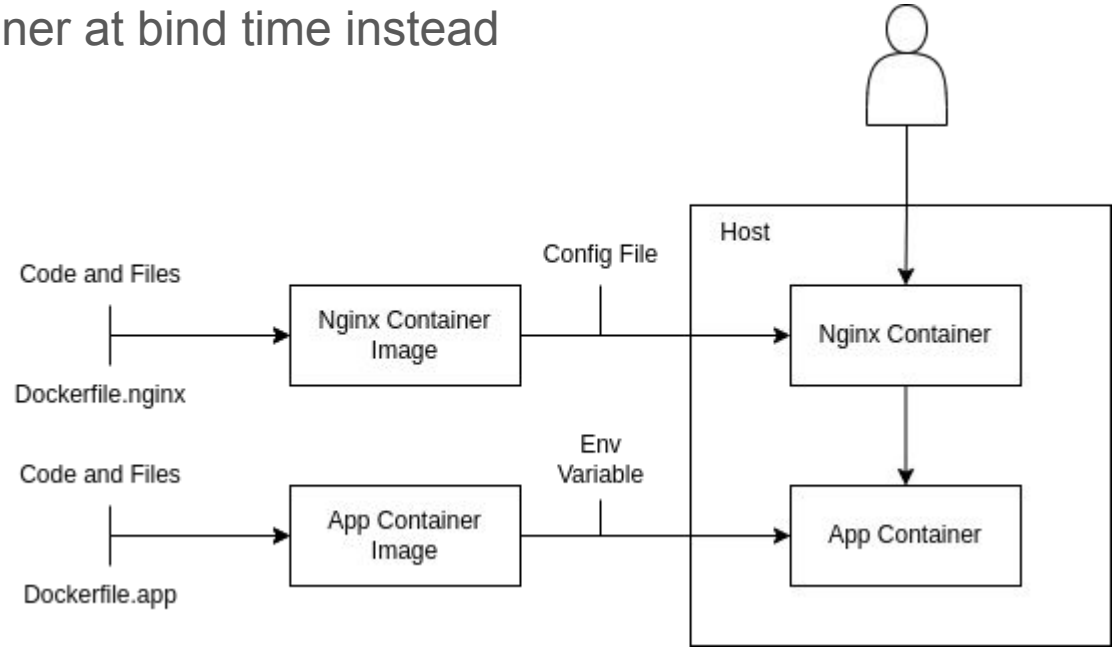
- *Dockerfile.app* - main application Dockerfile
- *Dockerfile.nginx* - Nginx service Dockerfile
- *docker-compose.yml* - docker compose configuration file



Example 4 - Configs

Used to manage data required by the application at runtime.

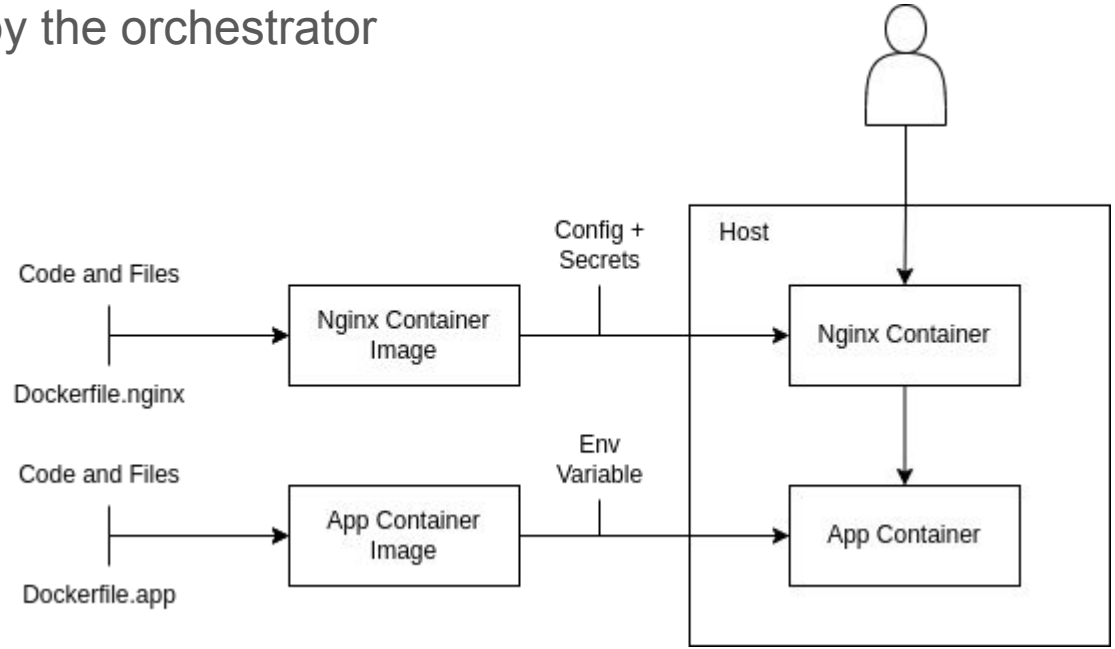
Add configuration to container at bind time instead of build time.



Example 5 - Secrets

Used to manage sensitive data required by the application at runtime.

Controlled and encrypted by the orchestrator



Thanks!