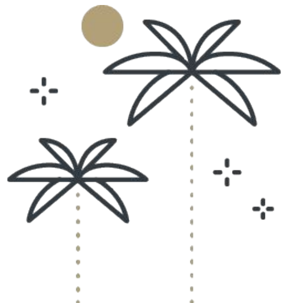
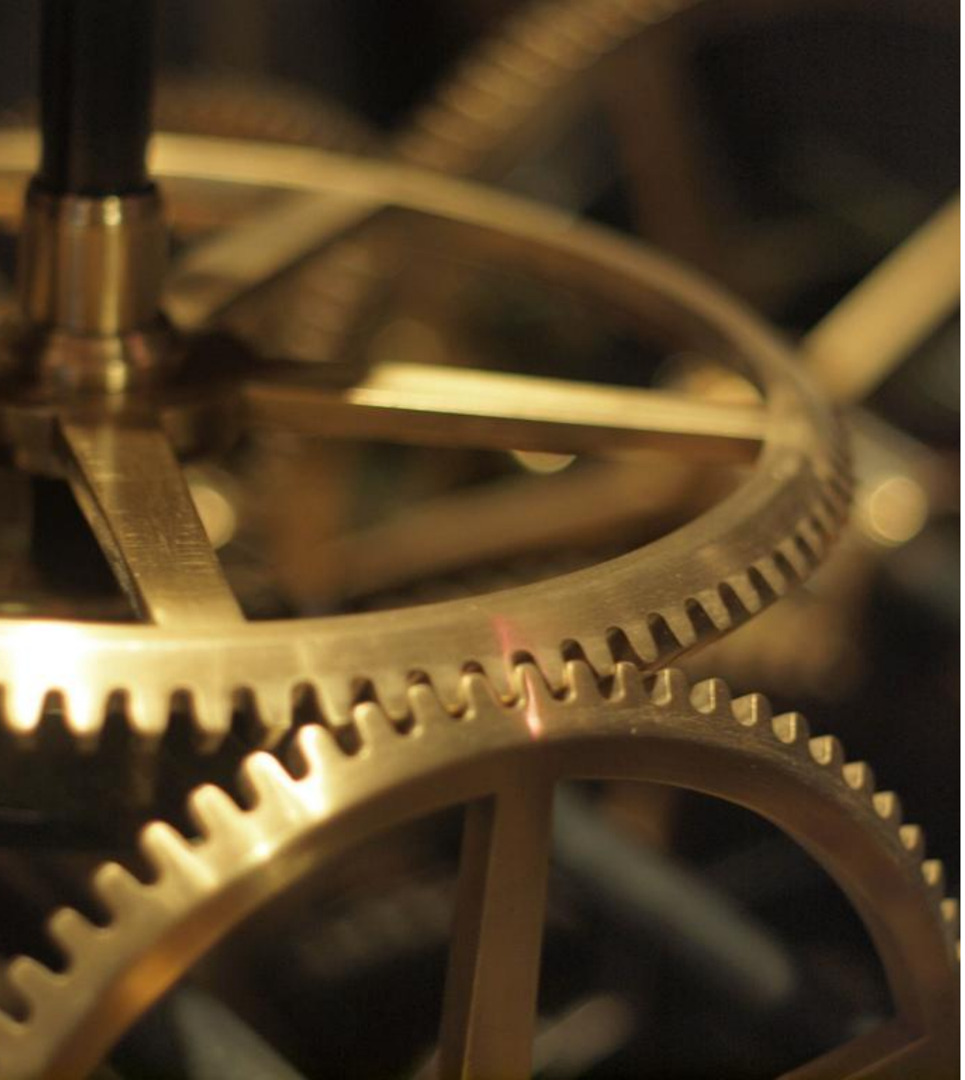
Docker

Thibaut BAYER - Tech sharing - Février 2019

Summary

- How it works
  - Virtualization VS Containerization
  - Advantages
- Let's play
  - Create Swift/Go containers
    - Hello world in web server
- Docker-compose
  - Same things using docker-compose
- CheatSheet
  - Keywords
  - Tools





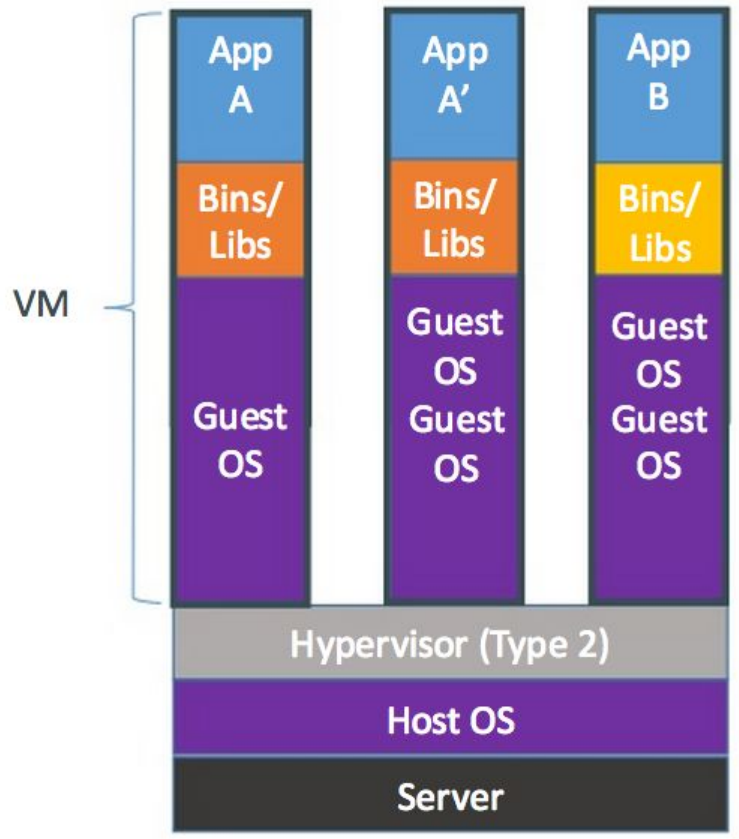
How it works

#1 How it works – Summary

“**Docker** is a computer program that performs operating-system-level virtualization, also known as containerization”

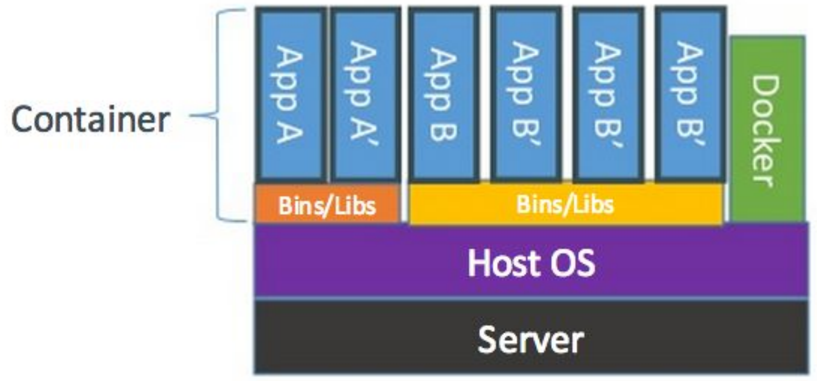
✓ Containers are isolated.
✓ Containers should run one process, not a full stack.
✓ Docker is not a common virtualization system, it’s containerization.
✘ Containers are not permanent, it mean, you should not store data.

#1 How it works - Virtualization VS Containerization



Containers are isolated, but share OS and, where appropriate, bins/libraries

...result is significantly faster deployment, much less overhead, easier migration, faster restart



#1 How it works - Advantages

- Performance (low-level virtualization)
- Security (isolation)
- Available in all operating system (windows included with WSL)
- Allows developers to test in the same environment
- Easy to test with multiple applications versions

Let’s play

#1 Let's play - Hello World

- docker run hello-world
- docker run --privileged -it --rm --name swiftfun swiftdocker/swift:latest swift
  - run: Pull & Build and run the image
  - --privilege: Ask super user privileges
  - -it: Run in interactive mode
  - --rm: Remove filesystem to have a clean container
  - --name: The name of the container
  - swiftdocker/swift: Name of the image we want to pull
  - :latest : Version of the image
  - swift: Command to execute

#1 Let's play - Swift Web server

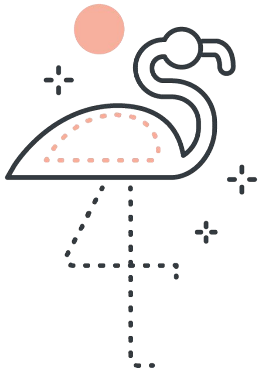
- docker build . -t btor/swift-webserver
  - build: Build the image
  - . : Location of the image
  - -t btor/swift-webserver: Name the image (with tag)
- docker run -p=8181:8181 -it btor/swift-webserver
  - run: pull & build & run the image
  - -p:8181:8181 : Ask docker to open internal port 8181 to host port 8181

And with GO ?

- docker build . -t btor/go-webserver
- docker run -p=8181:80 -it btor/go-webserver

When using Docker “vanilla” you must know all commands and arguments.

Docker-compose



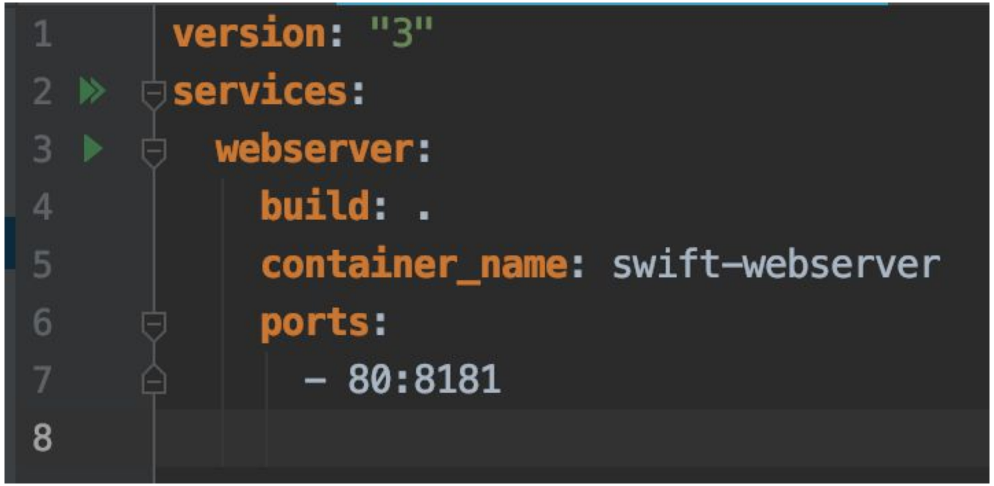
#1 Docker-compose

Docker-compose is a tool to help you to run and manage docker containers.

Without docker-compose we must do:

- docker build . -t btor/swift-webserver
- docker run -p=80:8181 -it btor/swift-webserver

With docker-compose, we can do:



```yaml
version: "3"
services:
  webserver:
    build: .
    container_name: swift-webserver
    ports:
      - 80:8181

```

version: Version of docker-compose syntax
services: Where you define all your containers
webserver: Name of the container for docker-compose
build: Path to your Dockerfile, you can replace 'build' by 'image' of you want to use a docker image
container_name: Name of the container for docker.
ports: Same as docker

#1 Docker-compose

To run a docker-compose stack:

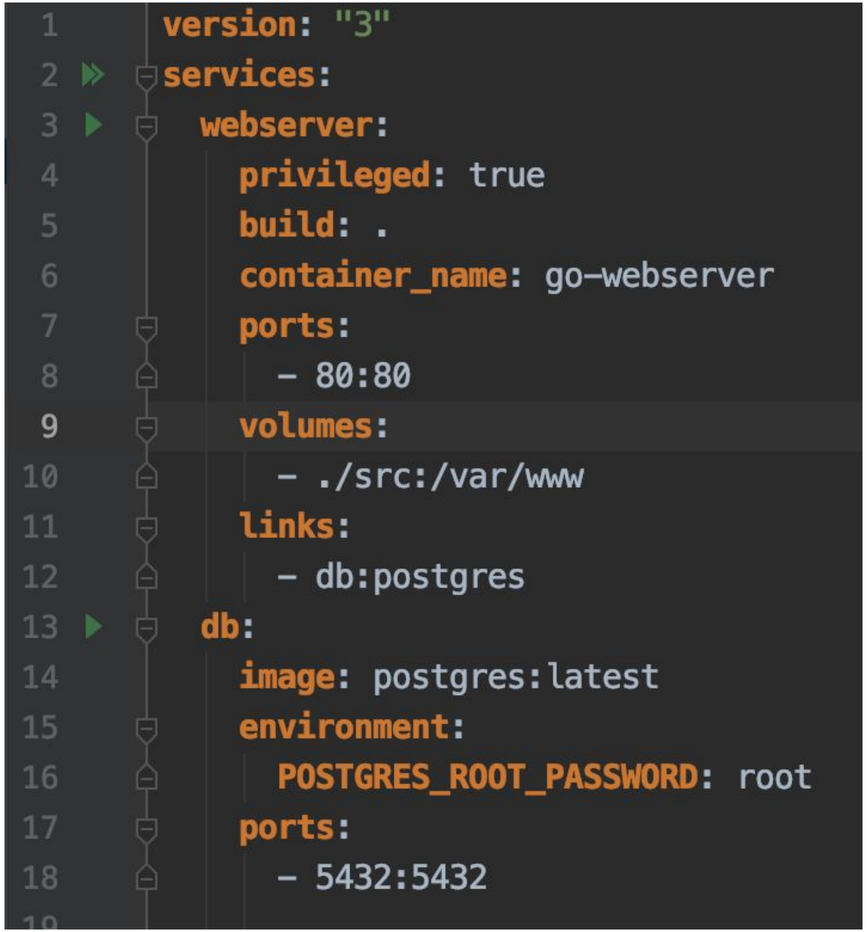
- docker-compose build
- docker-compose up

or both ! docker-compose up --build

To stop a docker-compose stack, you can send:

- docker-compose down

#1 Docker-compose



```yaml
version: "3"
services:
  webserver:
    privileged: true
    build: .
    container_name: go-webserver
    ports:
      - 80:80
    volumes:
      - ./src:/var/www
    links:
      - db:postgres
  db:
    image: postgres:latest
    environment:
      POSTGRES_ROOT_PASSWORD: root
    ports:
      - 5432:5432
```

More complex docker-compose file

volumes: Mount a directory inside the container

links: Allow the current container (webserver) to reach container “db” with an alias (here postgres)

environment: Set environments variables, most of them or explained in the docker hub readme page.

#1 Docker-compose

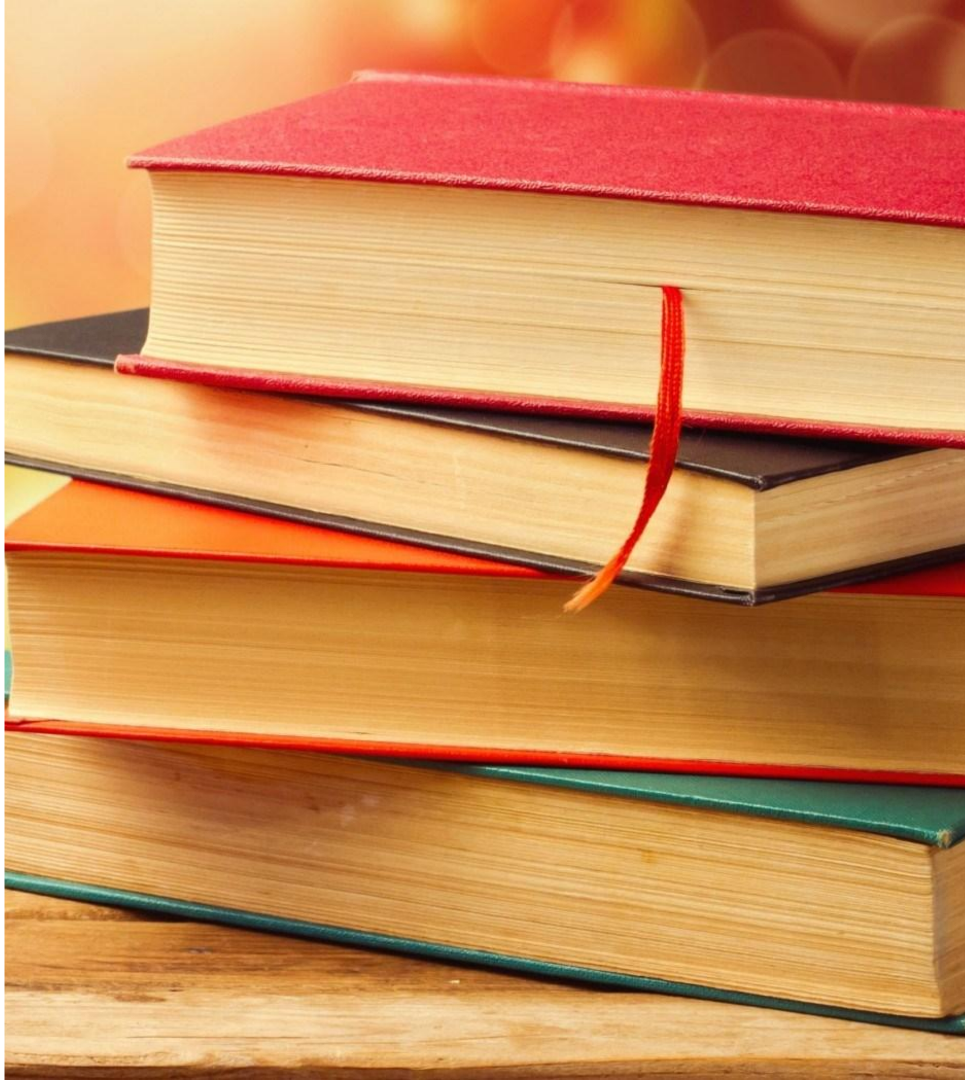
✓ Simplify the versionning of docker stacks.

✓ Useful for linking containers

✓ Easy to use than docker vanilla

✘ Add nothing more than docker, it’s just an abstraction.

CheatSheet



#1 Docker keywords

- **FROM**: The image we want to use to start (we can create our own)
- **RUN**: Run a command
- **WORKDIR**: Change de “home” directory
- **EXPOSE**: Open a specific port of the container to the others
- **CMD**: Run this command when container is up
- **COPY**: Copy files from the host to the container

#1 Tools

- **Docker hub**: Like Github but for Docker 🎉
- **docker-compose**: Tool to simplify the run/management of containers
- **Kubernetes / Docker swarm**: Orchestrator for containerized applications, manage the health/status of containers.

@LateNightSeth
THANK YOU!