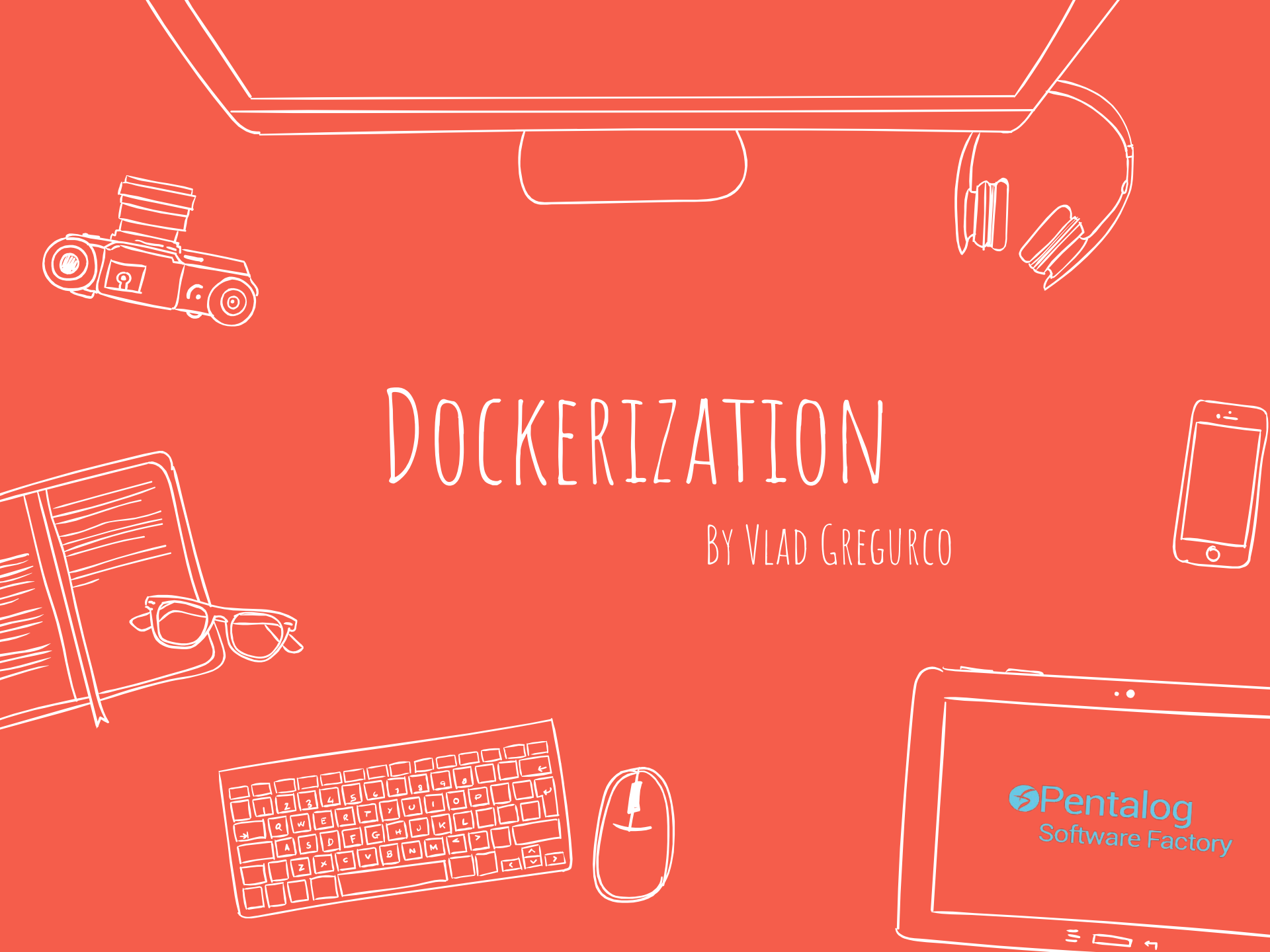
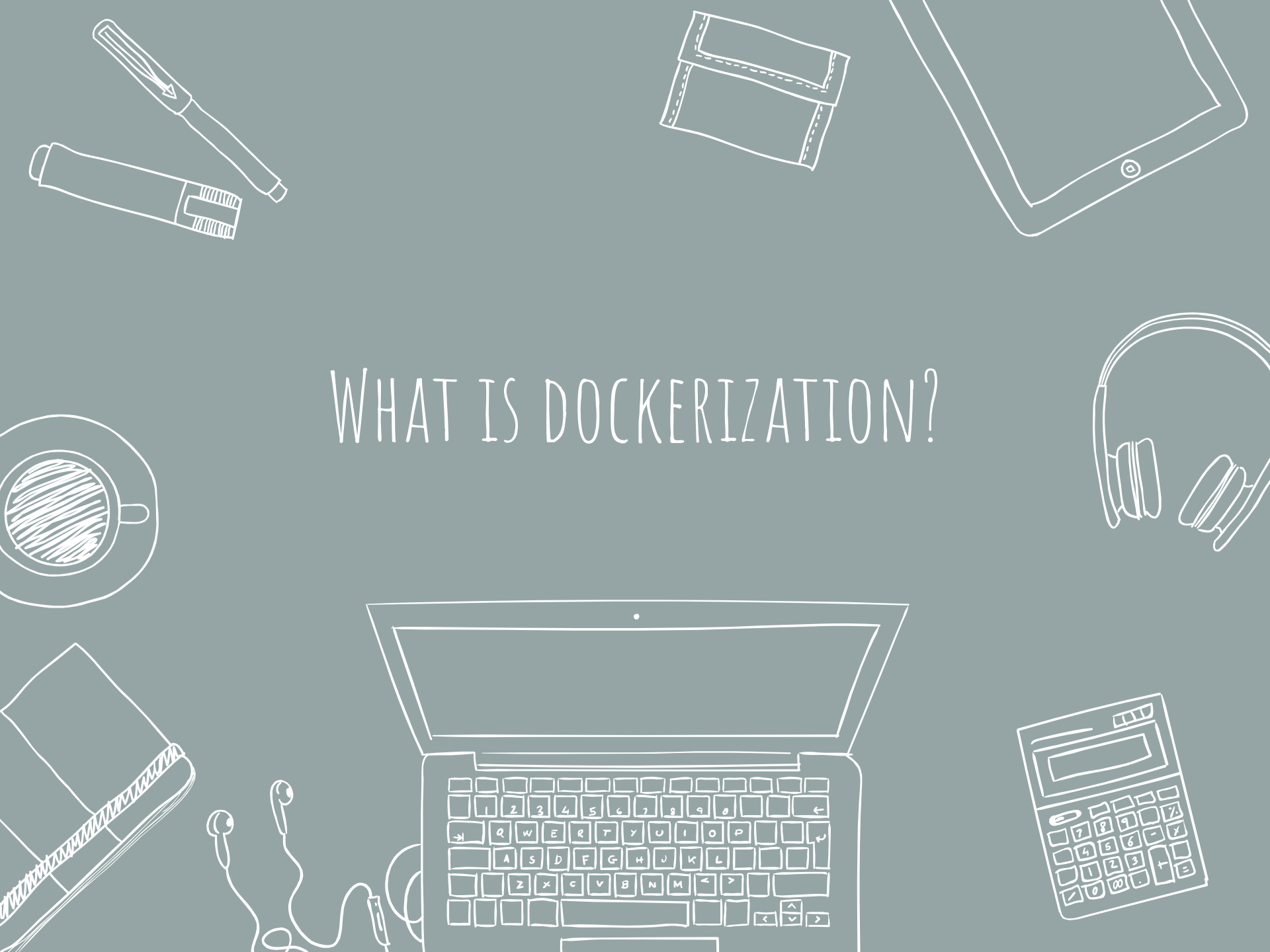
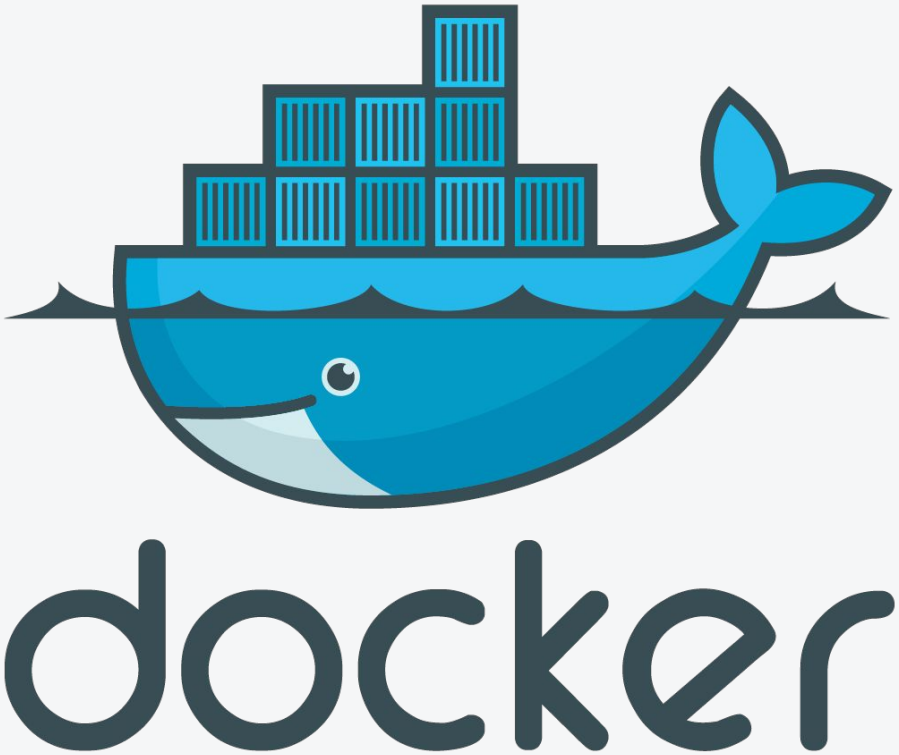
Dockerization

By Vlad Gregurco

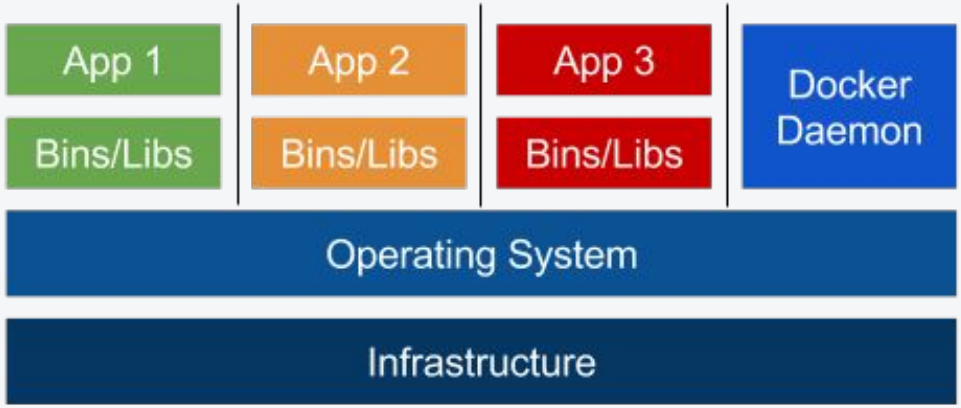


What is dockerization?



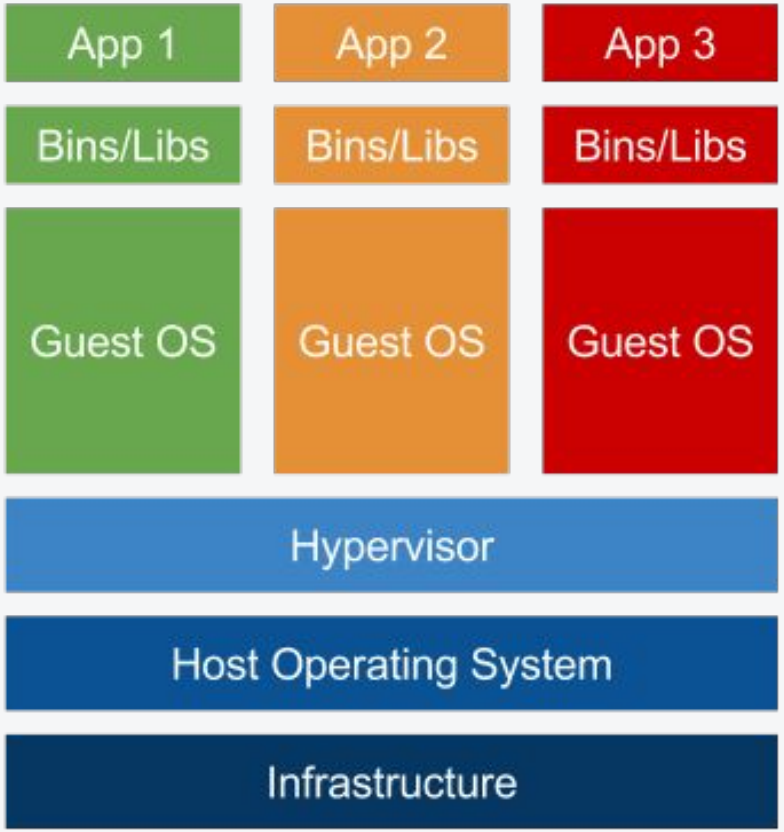


Docker logo



CONTAINER-BASED VIRTUALIZATION SCHEMA

WHAT WAS BEFORE?



Hypervisor-based Virtualization schema

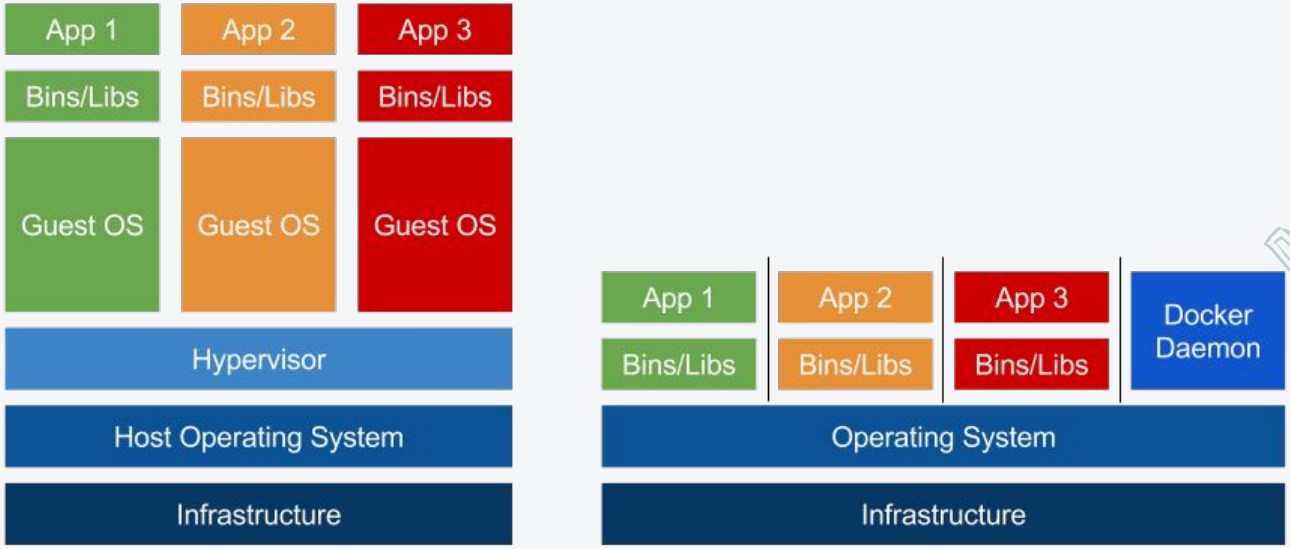
Parallels™

vmware

ORACLE VirtualBox

QEMU

Hypervisor-based Virtualization


App 1
App 2
App 3
Bins/Libs
Bins/Libs
Bins/Libs
Guest OS
Guest OS
Guest OS
Hypervisor
Host Operating System
Infrastructure
App 1
App 2
App 3
Docker Daemon
Bins/Libs
Bins/Libs
Bins/Libs
Operating System
Infrastructure

Comparison


SOUNDS GOOD BUT
DO WE NEED IT?
imgflip.com

YES

Advantages

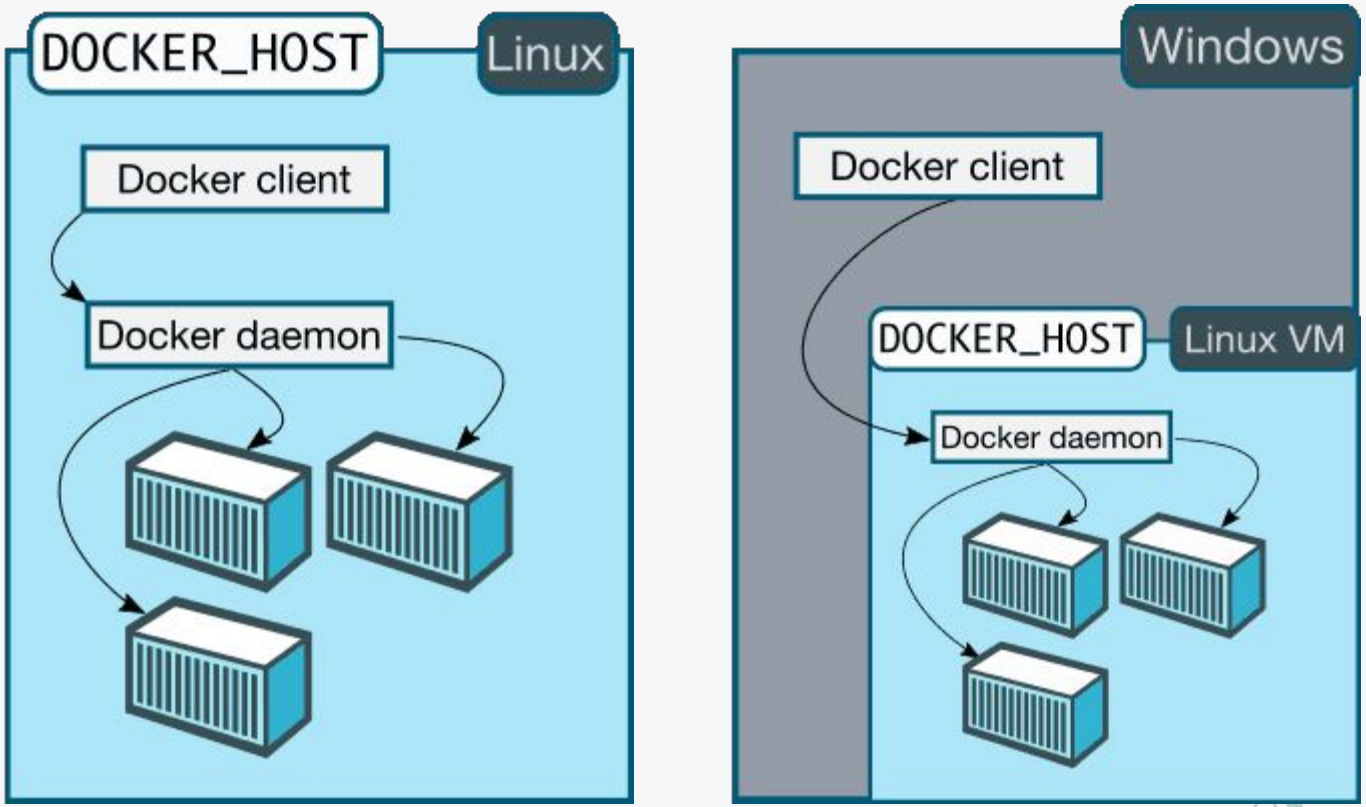
- Simple setup
- Possibility to develop different application with different dependencies
- Prepared official/community images
- Implements *“Infrastructure as Code”* principle
- Common environment for devs
- Common setup for dev/test/prod
- Security
- Isolation


SETUP

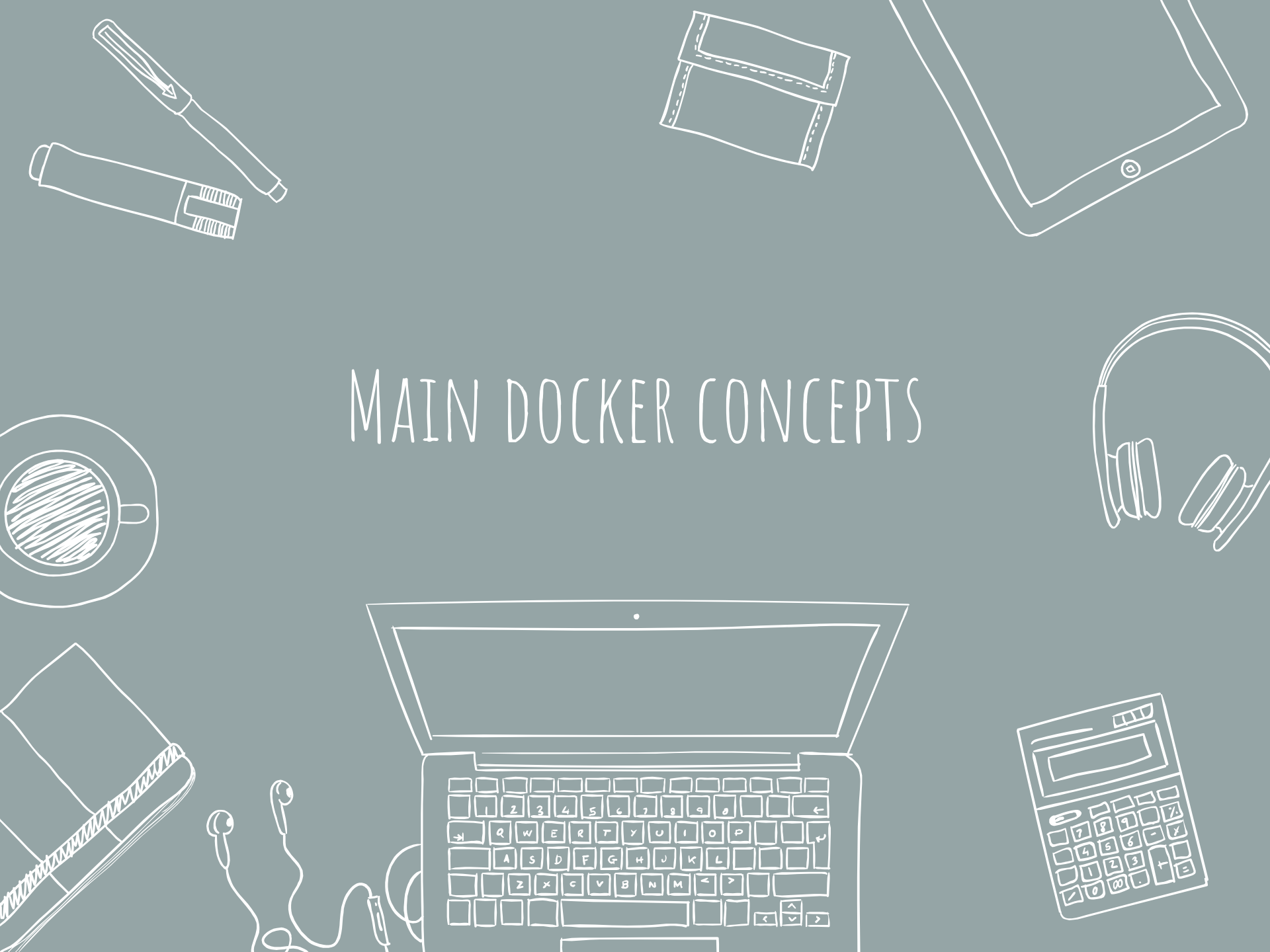
Setup

- Install docker
  - Ubuntu: https://docs.docker.com/install/linux/docker-ce/ubuntu/
  - Windows: https://docs.docker.com/docker-for-windows/install/
- Install docker-compose
  - https://docs.docker.com/compose/install/
- Search for images:
  - https://hub.docker.com/

Docker on Linux and Windows



Main docker concepts



Images

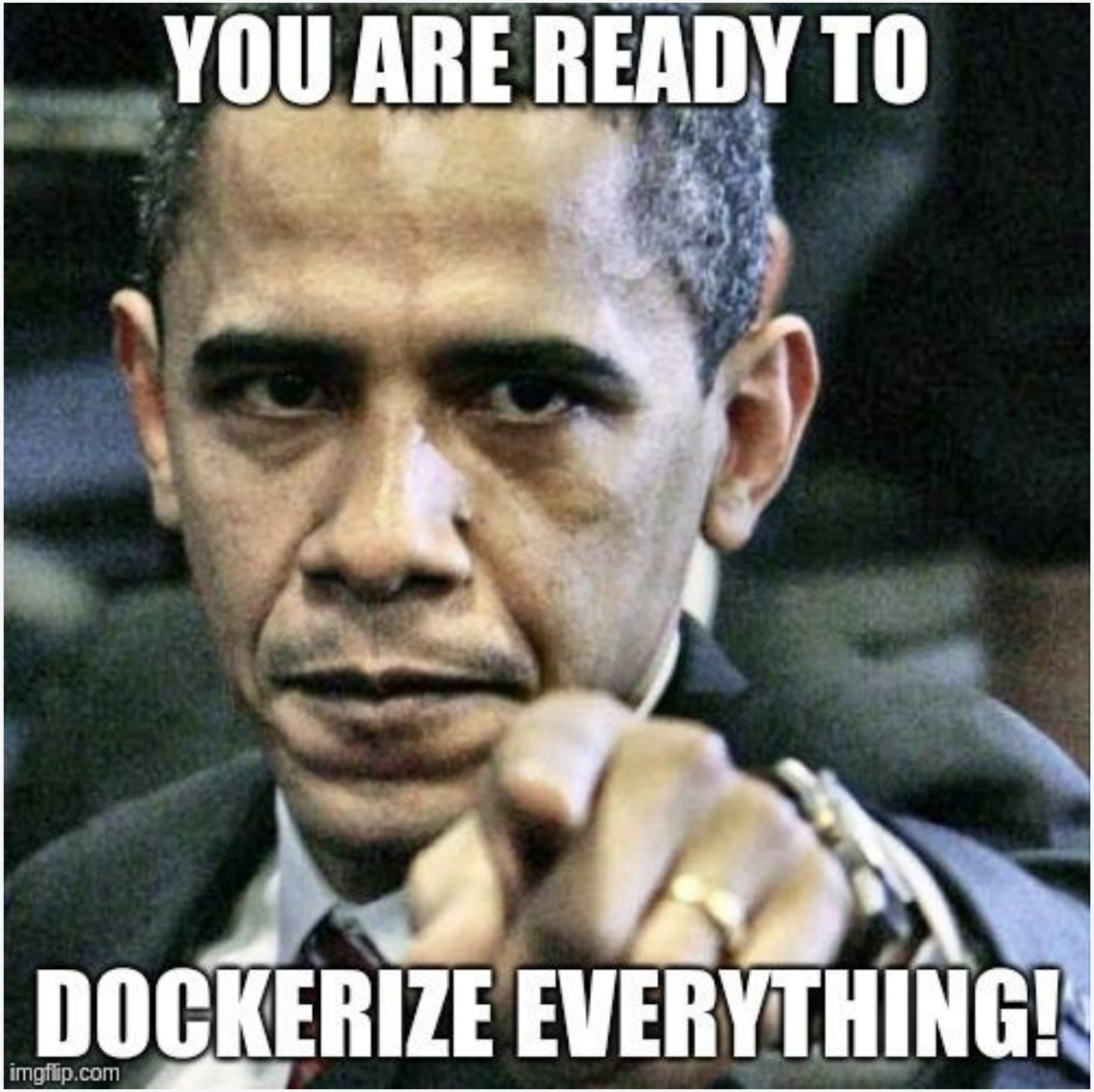
- Images are read only templates used to create containers.
- Images are created with the *docker build* command.
- Images are composed of layers of other images.
- Images are stored in a Docker registry.

Containers

- If an image is a class, then a container is an instance of a class - a runtime object.
- Containers are created from images. Inside a container, it has all the binaries and dependencies needed to run the application.
- Containers are lightweight and portable encapsulations of an environment in which to run applications.

Registries

- A registry is where we store our *images*.
- You can host your own registry, or you can use Docker's public registry which is called DockerHub.


YOU ARE READY TO
DOCKERIZE EVERYTHING!
imgflip.com

Run first container

Find BusyBox image

Search - Docker Hu

1

Secure https://hub.docker.com/search/?isAutomated=0&isOfficial=0&page=1&pullCount=0&q=busybox&starCount=0

Docker Store is the new place to discover public Docker content. Check it out →

busybox

2

Explore Help Sign up Sign in

Repositories (1882)

All

3

busybox official	1.2K STARS	10M+ PULLS	DETAILS
arm64v8/busybox public	0 STARS	500K+ PULLS	DETAILS
s390x/busybox public	2 STARS	100K+ PULLS	DETAILS
radial/busyboxplus public | automated build	19 STARS	500K+ PULLS	DETAILS
hypriot/rpi-busybox-httpd	40	1M+	

Find last BusyBox version

library/busybox - D

Secure | https://hub.docker.com/_/busybox/

busybox is now available in the Docker Store, the new place to discover public Docker content. Check it out →

busybox

Explore Help Sign up Sign in

OFFICIAL REPOSITORY

busybox

Last pushed: 10 days ago

Repo Info | Tags (2)

Short Description

Busybox base image.

Docker Pull Command

```
docker pull busybox
```

Full Description

Supported tags and respective `Dockerfile` links

- `1.28.3-uclibc`, `1.28-uclibc`, `1-uclibc`, `uclibc` (*uclibc/Dockerfile*)
- `1.28.3-glibc`, `1.28-glibc`, `1-glibc`, `glibc` (*glibc/Dockerfile*)
- `1.28.3-musl`, `1.28-musl`, `1-musl`, `musl` (*musl/Dockerfile*)
- (1) `1.28.3`, `1.28`, `1`, `latest` (*uclibc/Dockerfile*)

Quick reference

- **Where to get help**:

Run BusyBox container

```
[node1] (local) root@192.168.0.23 ~
$ docker run busybox:1.28 echo "Hello World"
Unable to find image 'busybox:1.28' locally
1.28: Pulling from library/busybox
f70adabe43c0: Pull complete
Digest: sha256:58ac43b2cc92c687a32c8be6278e50a063579655fe3090125dcb2af0ff9e1a64
Status: Downloaded newer image for busybox:1.28
Hello World
```

Docker command parts:

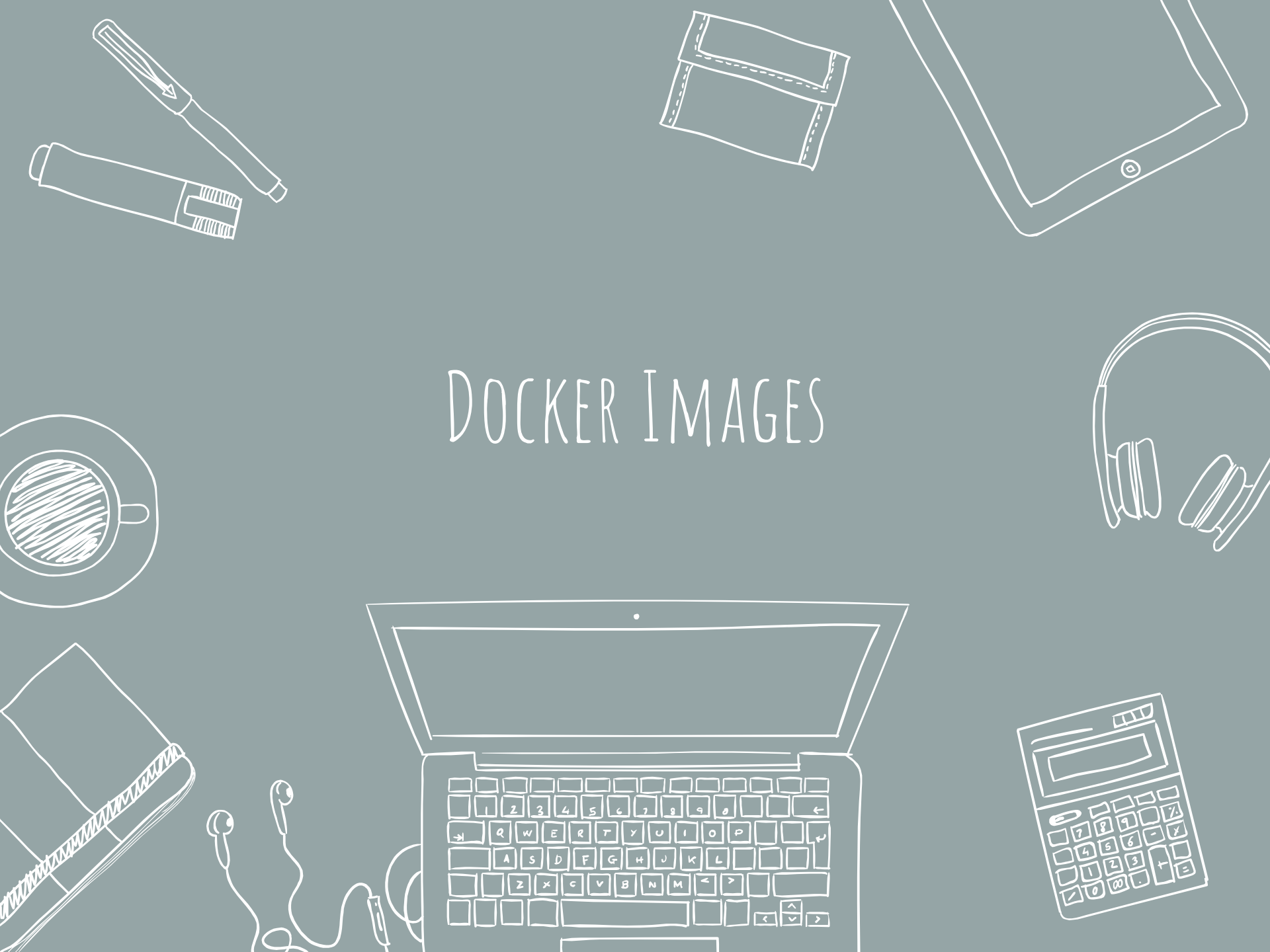
- **run** - run a command in a new container
- **busybox** - image name
- **1.28** - image version
- **Echo "hello World"** - command to execute in container

** To run commands it's possible to use https://labs.play-with-docker.com/ , **but** better is to do it locally*

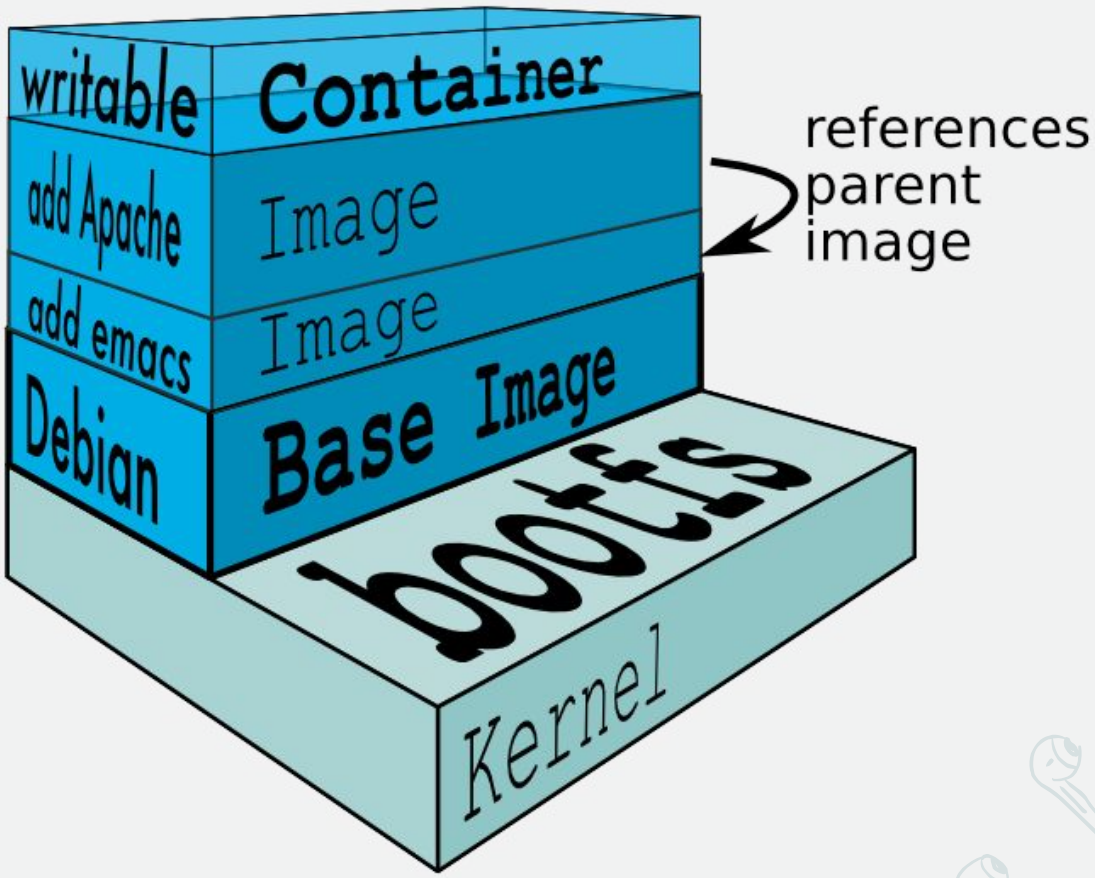
Practical Part

- Run: docker run busybox:1.28 **ls**
- Run:
  - docker run **-it** busybox:1.28
  - Create inside the container an empty file
  - Exit from container
  - Run first command once again
  - Check if file exists
- Run:
  - docker run **-d** busybox:1.28 sleep 1000
  - docker **ps**
  - docker ps **-a**
  - Try docker run command with **--rm** and **--name** options
  - docker **inspect** container_id
- Run:
  - docker run -d **-p 8888:80** nginx:1.13
  - Open a browser with address localhost:8888
  - docker **logs** container_id
  - docker logs **-f** container_id

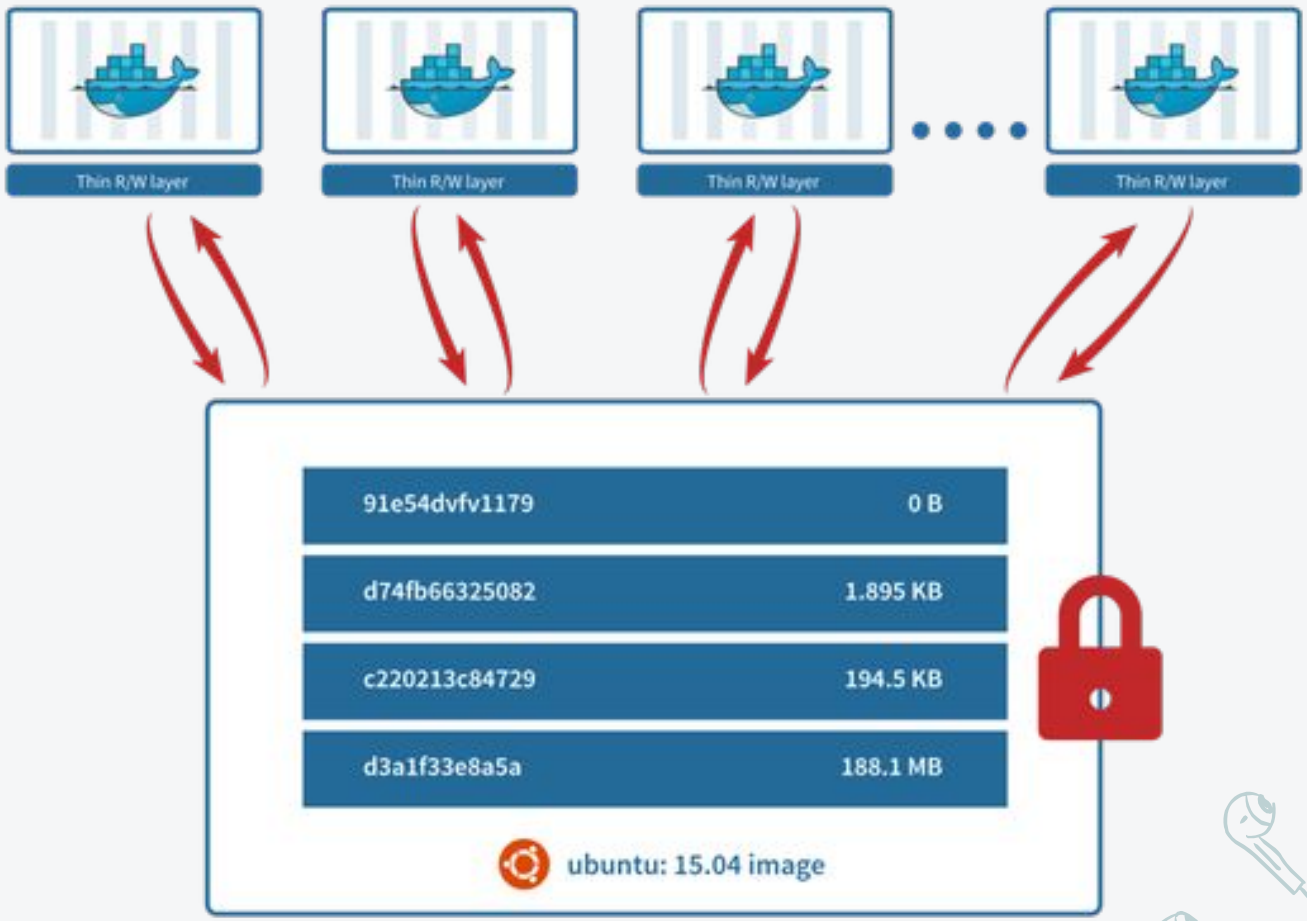
Docker Images



Docker Image Layers



Docker Image Layers



Build docker image (commit command)

- Run: *docker run -it debian:jessie*
- Check that git is not installed
- Run: *apt-get update && apt-get install git*
- Check that git is installed and exit from container
- Run:
  - *docker ps -a* and find there the container id of debian
  - *docker commit container_id your_login/debian:1.0*
  - *docker images*
- Check that new image appeared
- Run: *docker history your_login/debian:1.0.0*

Note: Now you can push this image to registry or run containers based on this image.

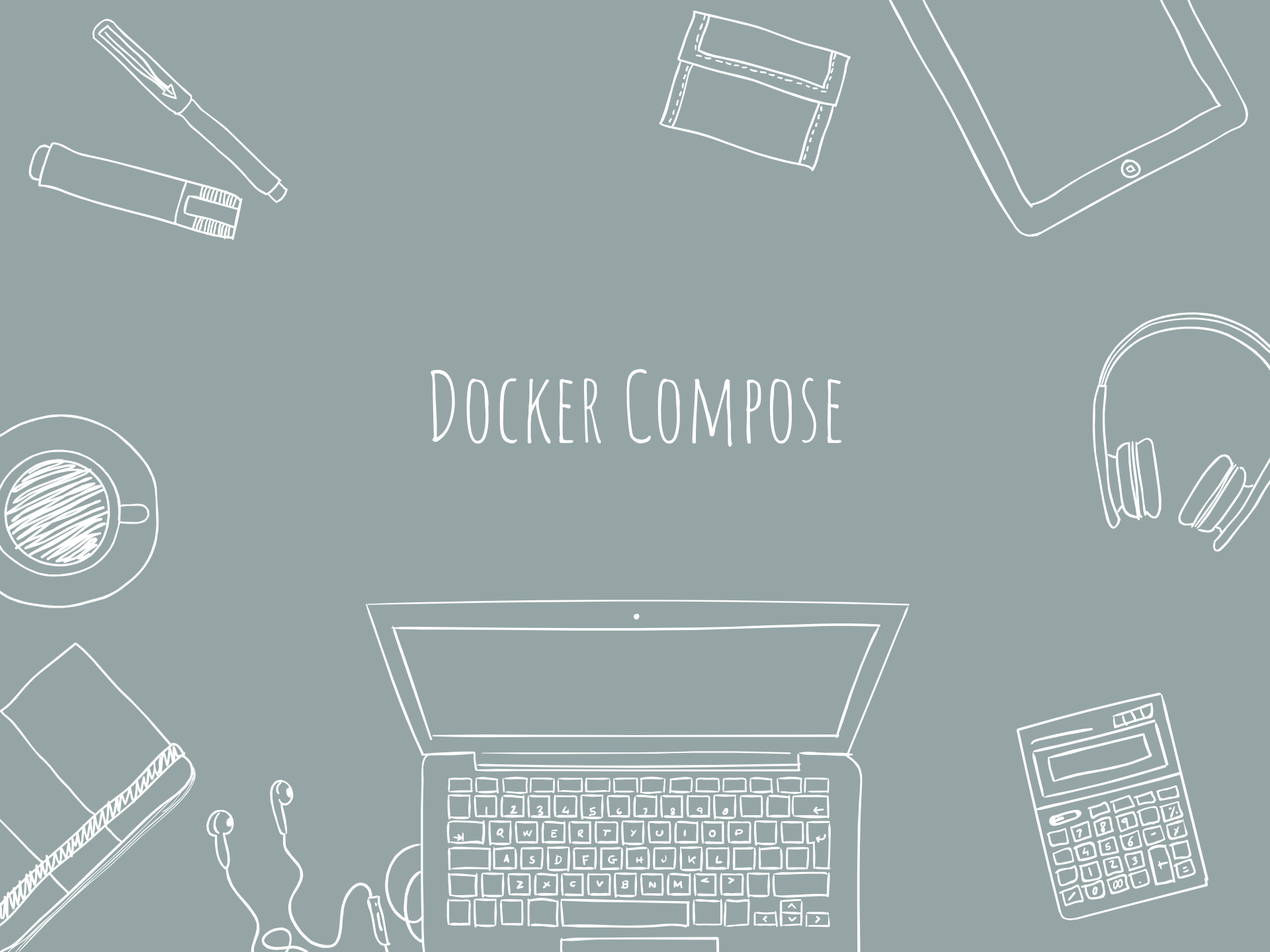
Build docker image (dockerfile)

- Create new file with name *Dockerfile* and next content:

```
FROM debian:jessie

RUN apt-get update && apt-get install -y git
```

- Run: *docker build -t your_login/debian_from_dockerfile .*
- Run: *docker images*
- Check that image is created
- Try to do the same build but split **run** in two commands. See the history of the image.

Docker Compose



Docker Compose

- Docker compose is a very handy tool to quickly get docker environment up and running.
- Docker compose uses yaml files to store the configuration of all the containers, which removes the burden to maintain our scripts for docker orchestration.

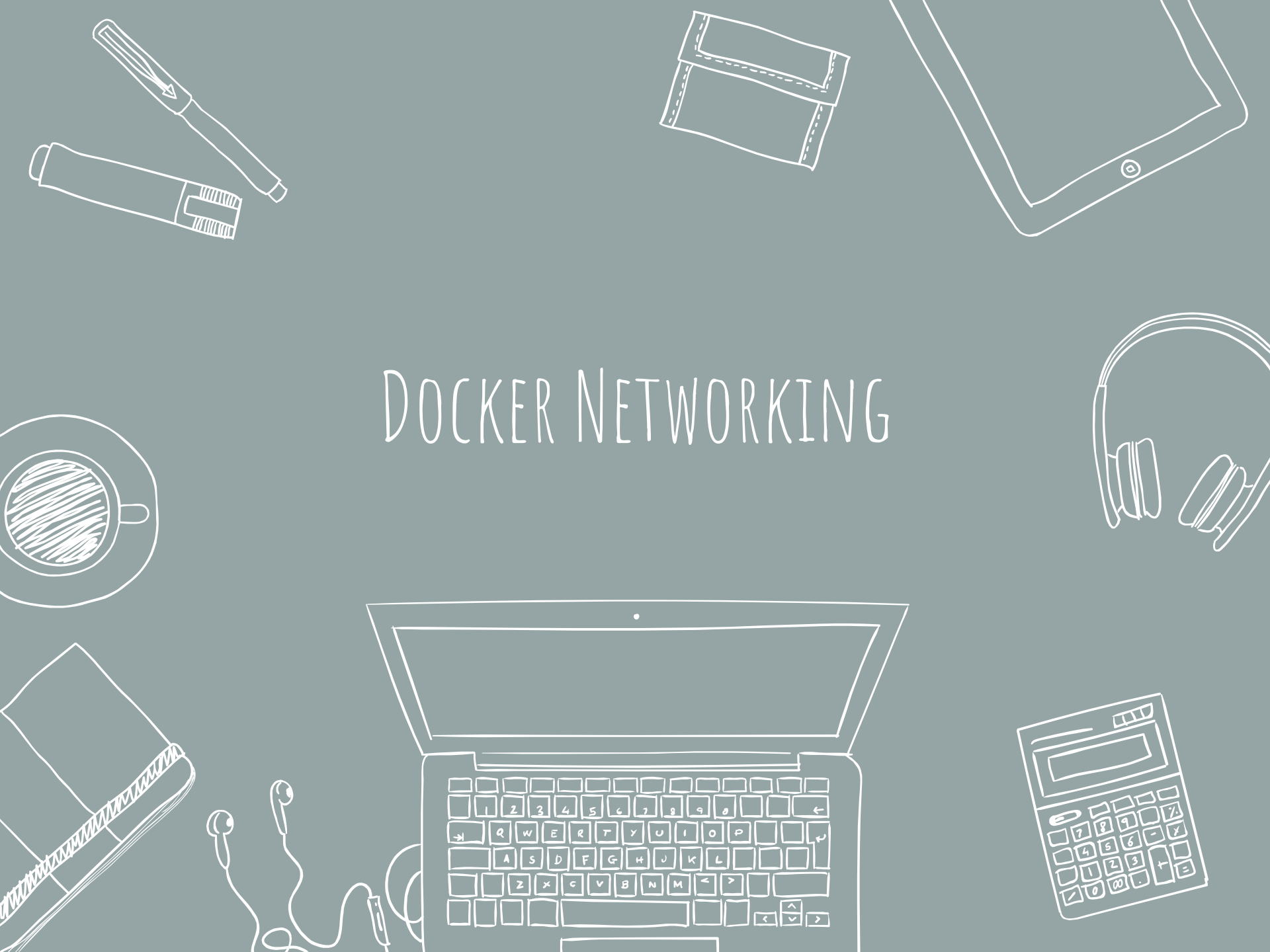
Docker Compose Config example

```yaml
version: "3.1"
services:
    redis:
      image: redis:alpine
      container_name: project-redis

    mysql:
      image: mysql:5.7
      container_name: project-mysql
      working_dir: /application
      volumes:
        - .:/application
      environment:
        - MYSQL_ROOT_PASSWORD=password
        - MYSQL_DATABASE=project_db
        - MYSQL_USER=user
        - MYSQL_PASSWORD=password
      ports:
        - '3306:3306'

    webserver:
      image: nginx:alpine
      container_name: project-webserver
      working_dir: /application
      volumes:
          - .:/application
          - ./phpdocker/nginx/nginx.conf:/etc/nginx/conf.d/default.conf
      ports:
       - "8080:80"

    php-fpm:
      build: phpdocker/php-fpm
      container_name: project-php-fpm
      working_dir: /application
      volumes:
        - .:/application
```

Docker Compose Commands

- **docker compose up** - starts up all the containers.
- **docker compose ps** - checks the status of the containers managed by docker compose.
- **docker compose logs** - outputs colored and aggregated logs for the compose-managed containers.
- **docker compose logs -f** - outputs appended log when the log grows.
- **docker compose logs *container_name*** - outputs the logs of a specific container.
- **docker compose stop** - stops all the running containers without removing them.
- **docker compose rm** - removes all the containers.
- **docker compose build** - rebuilds all the images.

Docker Networking

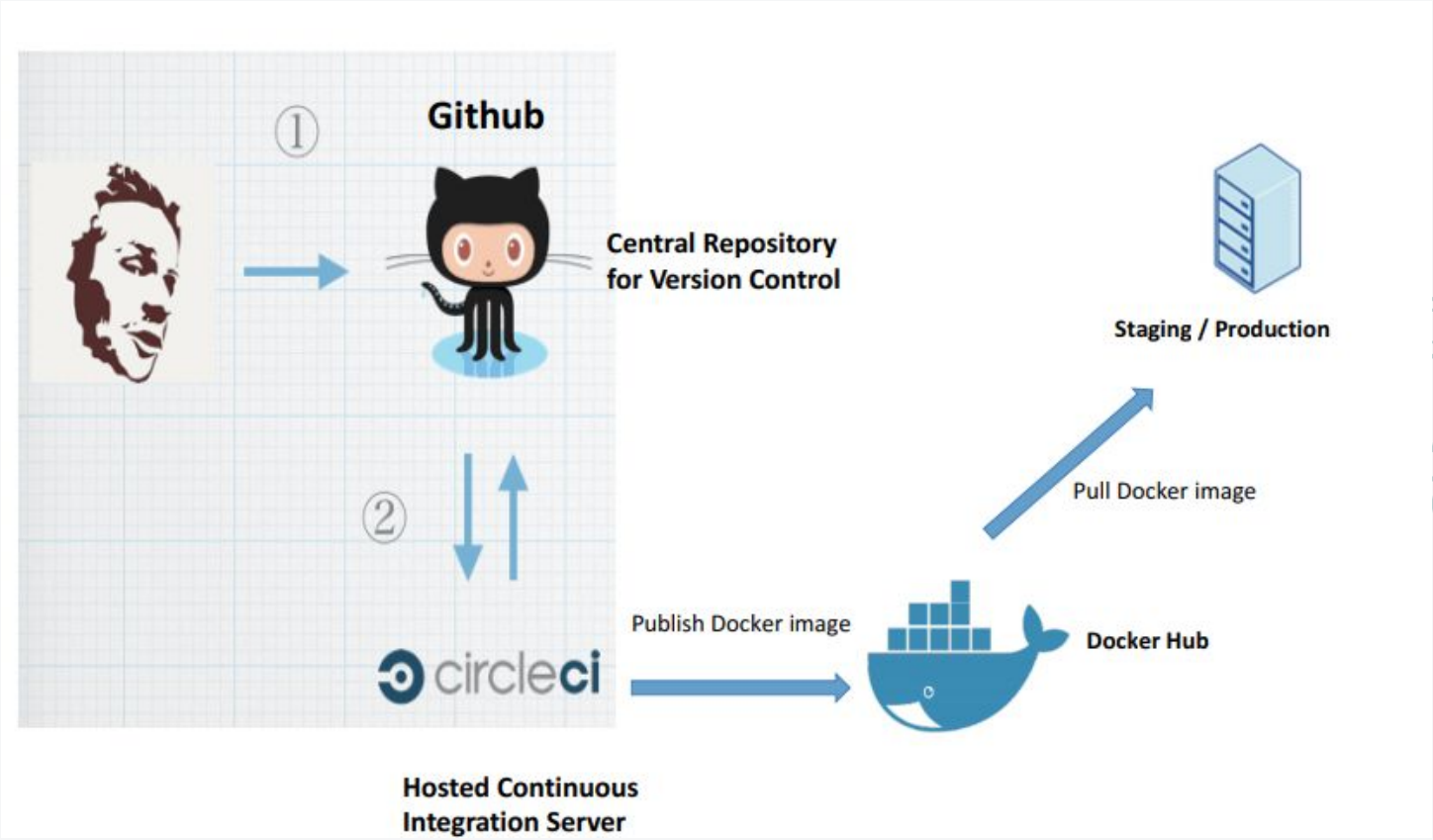


Docker Network Types

- **Closed Network / None Network** - disable all networking.
- **Bridge Network** - the default network driver which allows containers connected to the same bridge network to communicate.
- **Host Network** - adds a container on the host's network stack.

Complete CI Workflow

Complete CI Workflow



Final Tips And Tricks

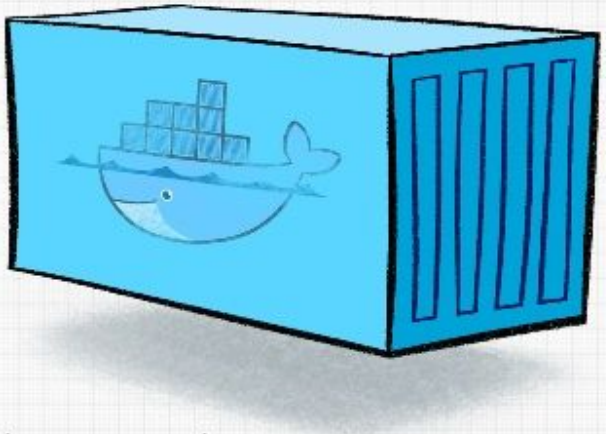
- Use Docker
- Use Docker Compose in case running multi-containers
- Whenever possible, use current Official Repositories as the basis for your image
- Use minimal base image as it's possible
- Be always up to date with last changes

More information

- Docker Documentation: https://docs.docker.com
- Symfony Devs Slack / Docker channel: https://symfony-devs.slack.com/messages/C6Y94Q3CZ
- Docker online sandbox: https://labs.play-with-docker.com
- DockerCon 2017: https://www.youtube.com/playlist?list=PLkA60AVN3hh_nihZ1mh6cO3n-uMdF7UlV
- Docker YouTube Channel: https://www.youtube.com/channel/UC76AVf2JkrwjxNKMuPpscHQ
- Generator of docker compose config: https://phpdocker.io/generator

Quotation

The real value of Docker is not technology



It's getting people to agree on something

Questions?