Docker concepts and Vocabulary

For understanding, motivation and discussion

25.1.2024

Haaga-Helia

Docker

PaaS environment to build, deliver and run software as isolated and independent containers.

- By default, containers are sand-boxed / isolated.
- You can make container services available to others by **exposing ports**, e.g. for your browser to use
- You can make containers to see others by connecting them by virtual **networks** between containers
- You can make containers to share data also by shared **volumes.** Basically a shared folder that two or more containers can access.

Docker Engine

- The engine (*dockerd* daemon/process and Docker Engine API)
- To manage and run the containers.
- Isolate and connect them based on definitions

docker (CLI command)

- The *docker* command line command.
- Client for giving commands e.g. about
  - starting or stopping containers, removing containers
  - building, publishing images, removing images
  - finding status of containers and images
- Or executing commands inside the container from outside: docker exec …

Dockerfile

- Your ‘script’ for making your own Docker images.
- Image could be built based on source code and other assets on your disk
    - and, if needed, based on / expanding on ready-made images from *Docker Hub* or other docker image registry.

.dockerignore

- Lists which files and folders won’t be packed into the Docker image.

image

- Ready-made from Docker Hub, or one you have created. Template that can be used to create container.
- E.g. some MariaDB image you want to take into use. It’s a snapshot of a running/runnable MariaDB or other DB server that starts from a certain documented state. Typically there is a root user with known password (public information, everyone knows the password!), a certain database/schema created, like ‘test’. When taking that image into use, you must then immediately:
  - secure the root user by changing the password
  - create a user with less privileges with safe password or other safe access.
  - give that user access to wanted schema etc.
  - continue possibly with table creation, etc…

Docker Image registry, e.g. Docker Hub

- We can push = publish our images for others to use.
- Or pull = download images to use ourselves.

docker compose

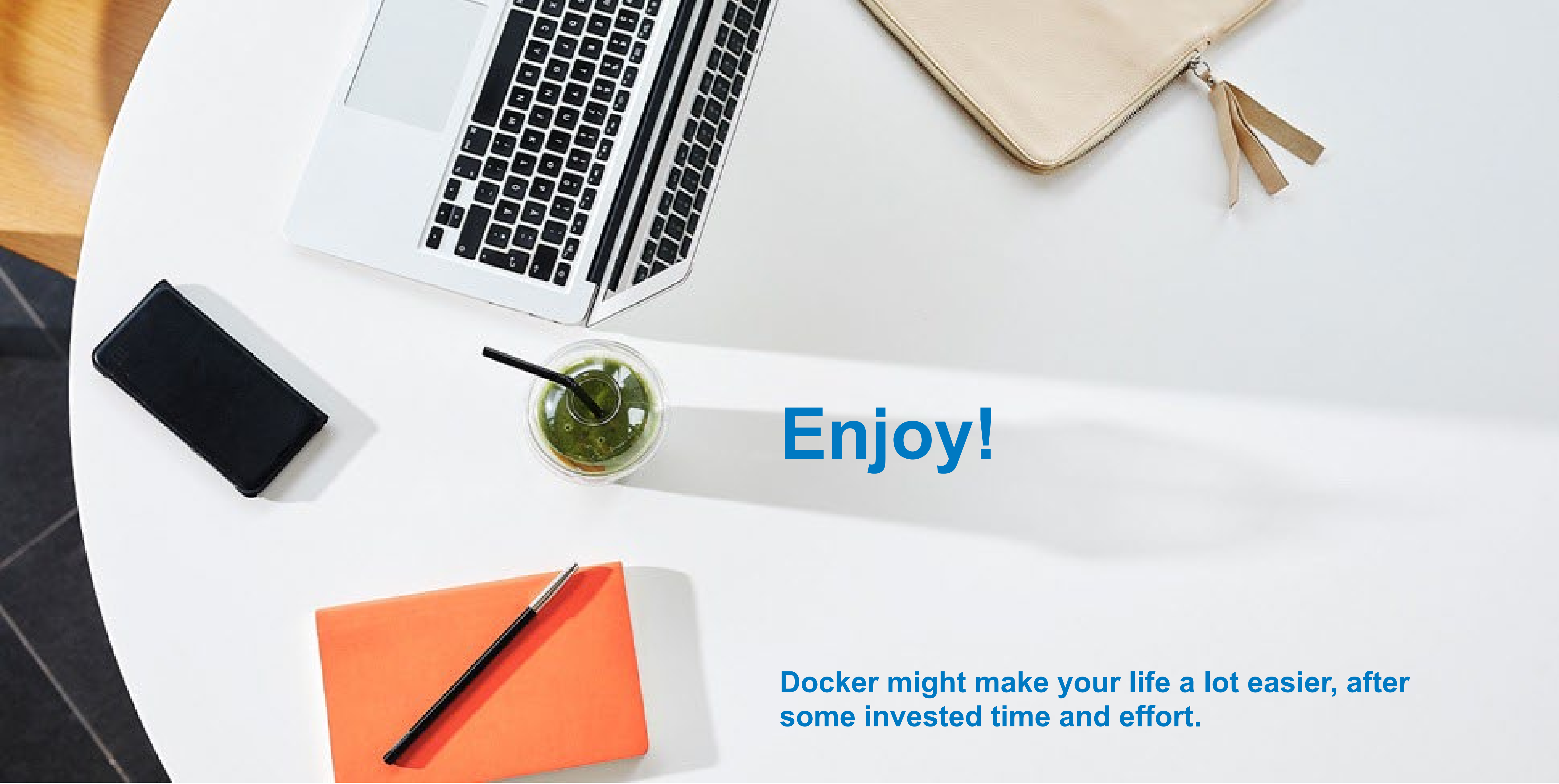
- Tool for creating and starting multiple containers that depend on / talk to each other.
- Thus, you’ll have to:
  - make some ports exposed
  - and/or define (virtual) networks shared by multiple containers
  - or share volumes.
- You can define those in a docker-compose.yml file

volume

- Persisted folders and files on disk.
- Allows sharing between containers
- Also for keeping data between container deletion and re-creation! (Container is deleted totally, but volume resides by default in the host file system)

network

- Allows (by default isolated) containers to communicate with each other, or the host computer
- There are multiple network configurations available, e.g. Bridge, Host, Overlay, none
- https://docs.docker.com/network/drivers/bridge/ **Bridge** is the normal one, letting included containers communicate with each other.
- https://docs.docker.com/network/drivers/host/ **Host** refers to the host computer where Docker engine is run. Do you want to e.g. let the containers interact with your host computer services? Sometimes absolutely not, some rare times yes.
- Also for keeping data between container deletion and re-creation!



Enjoy!

Docker might make your life a lot easier, after some invested time and effort.