Teamwork

Docker basics

Thomas Domingues

Summary

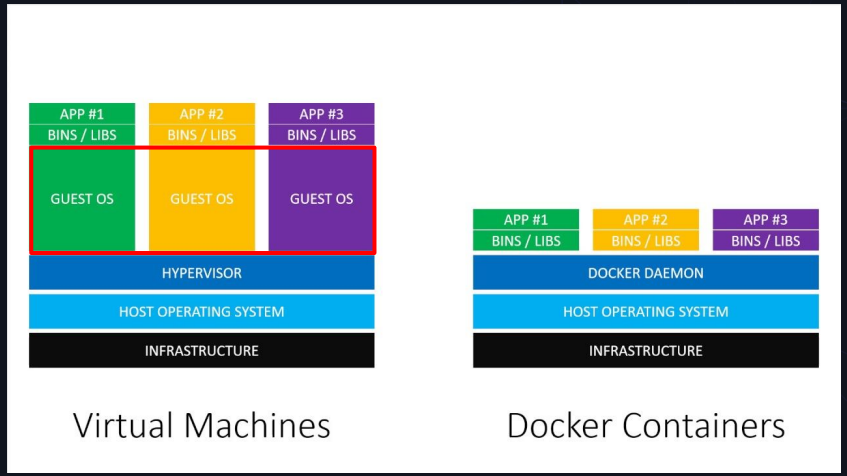
- What is Docker?
- Why Docker?
- How Docker works?
- Docker commands
  - pull / run
  - push / login
  - ps / rm
  - images / rmi
  - prune
- Tutorials

What is Docker?

- Created by Solomon Hykes, french developer in 2010
- Open-sourced in 2013
- Today runs on more than 20% of hosts and used by (almost) all big structures (Netflix, Facebook, Google, ...)

What is Docker?

- Docker is a tool aiming to help create, deploy and execute pre-built applications thanks to a feature in the Linux kernel
- Docker IS NOT virtualization, it's containerization



Why Docker?

Pros and cons over virtualization

Pros

- More, more, more lightweight
- Also means better performances
- Consistency for collaborative work
- Community is HUGE and awesome
- Work on specific versions of a tool / framework / program is easier by A LOT
- Faster and easier Continuous Deployment
- More flexible Continuous Integration

Cons

- Performance issues on non-native environments
- Migration over virtualization can be difficult
- ...That's all?

How Docker works?

- Unlike virtualization that run a different small operating system inside yours, containerization encapsulates an application as a single executable package of software that bundles application code together with all of the related configuration files, libraries, and dependencies required for it to run
- The result of the build, called **image**, is stored and can be runned locally. Its configuration is defined in a file called **Dockerfile** which has its own syntax
- This build is saved as **layers** (named **steps**), avoiding full rebuild if the end of the Dockerfile config is changed

How Docker works?

- At the run, the Docker Engine runs the latest command of the Dockerfile (which is usually a runnable script) and stop the container once the script is done.
- In the Web Development world, the last command is usually watching a continuously running daemon to prevent the Web Server / Application to stop right after

Docker commands - Pull

Download images from Distant Repository (default: Docker Hub) to create a local copy

Command syntax: `docker pull [repository]/[image]:[tag]`

- `repository/`: (facultative) for private repositories, pulling from Docker Hub by default
- `image` : image / bundled application name (example: nginx)
- `tag` : version of the image defined at build (example: 1.19) (default: latest) - You can find all tags available directly in Docker Hub: https://hub.docker.com/_/nginx

Docker commands - Pull

```
user@/git-project$ docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
0e03bdcc26d7: Pull complete
Digest:
sha256:31b9c7d48790f0d8c50ab433d9c3b7e17666d6993084c002c
2ff1ca09b96391d
Status: Downloaded newer image for hello-world:latest
docker.io/library/hello-world:latest
```

Command
Guessing tag: latest (default)
...
Layer 1 (the only one)
File verification
...
...
Status: downloaded from Docker Hub (default)
...

https://docs.docker.com/engine/reference/commandline/pull

Docker commands - Run

Run specific image and configured behavior in a container

- Command syntax: `docker run [repository]/[image]:[tag]`
- By default: run `docker pull` if the image / tag combination is not available locally

```
user@/git-project$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be
working correctly

[... explanation of how Docker works ...]
```

Command
Result of the script runned by hello-world container
...
...
...
After that, the container is stopped because the script is completed

https://docs.docker.com/engine/reference/commandline/run

Docker commands - Push

Upload local copy of an image to a specific repository

- Command syntax: `docker push [repository]/[image]:[tag]`

```
user@/git-project$ docker push hello-world
Using default tag: latest
The push refers to repository
[docker.io/library/hello-world]
9c27e219663c: Layer already exists
errors:
denied: requested access to the resource is denied
unauthorized: authentication required
```

Command
Guessing tag: latest (default)
Guessing repository (default: Docker Hub)
...
Layer 1
Error: not authorized
...
...

https://docs.docker.com/engine/reference/commandline/push

Docker commands - Login

Log in to a Docker registry

```
user@/git-project$ docker login
Login with your Docker ID to push and pull images from
Docker Hub. If you don't have a Docker ID, head over to
https://hub.docker.com to create one.
Username: tdomingues
Password:
WARNING! Your password will be stored unencrypted in
/home/thomas/.docker/config.json.
Configure a credential helper [...]

Login Succeeded
```

Command
Precise it's expecting your Docker Hub credentials
...
Username
Password
See official Docker documentation to configure a credential helper
...

Login confirmation

https://docs.docker.com/engine/reference/commandline/login

Docker commands - Push

Now let’s retry to push hello-world…

…same error?

Answer: you only can push to your namespace (like you can’t create a project at the root of GitHub / GitLab). To do so, you’ll also need to add a new tag to the image:

```
user@/git-project$ docker image tag hello-world:latest tdomingues/hello-world:latest
user@/git-project$ docker push tdomingues/hello-world
Using default tag: latest
The push refers to repository [docker.io/tdomingues/hello-world]
9c27e219663c: Mounted from library/hello-world
latest: digest:
sha256:90659bf80b44ce6be8234e6ff90a1ac34acbeb826903b02cfa0da11c82cbc042 size: 525
```

Docker commands - ps

List of all Docker containers

```
user@/git-project$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

Empty because our previous run was completed and ps displays only running containers by default. Add -a to also list stopped containers.

```
user@/git-project$ docker ps -a
CONTAINER ID   IMAGE         COMMAND    CREATED        STATUS                    PORTS     NAMES
7fb7d5acbbb0   hello-world   "/hello"   18 hours ago   Exited (0) 18 hours ago             dazzling_boyd
```

https://docs.docker.com/engine/reference/commandline/ps

Docker commands - rm

Remove specific stopped container (by container id or name)

- Can remove running containers with `--force` parameter

```
user@/git-project$ docker rm dazzling_boyd
dazzling_boyd
user@/git-project$ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

https://docs.docker.com/engine/reference/commandline/rm

Docker commands - rmi

Remove specific image (by image id)

- Remove by name can cause issues
- If you specify only the beginning of the ID, Docker will try to auto-complete

```
user@/git-project$ docker images
REPOSITORY    TAG       IMAGE ID       CREATED         SIZE
hello-world   latest    bf756fb1ae65   18 hours ago    13.3kB
user@/git-project$ docker rmi hello-world
Untagged: hello-world:latest
Untagged: hello-world@sha256:31b9c7d48790f0d8c50ab433d9c3b7e17666d6993084c002c2ff1ca09b96391d
user@/git-project$ docker rmi bf7
Untagged: tdomingues/hello-world:latest
Untagged: tdomingues/hello-world@sha256:90659bf80b44ce6be8234e6ff90a1ac34acbeb826903b02cfa0da11c82cbc042
Deleted: sha256:bf756fb1ae65adf866bd8c456593cd24beb6a0a061dedf42b26a993176745f6b
Deleted: sha256:9c27e219663c25e0f28493790cc0b88bc973ba3b1686355f221c38a36978ac63
```

https://docs.docker.com/engine/reference/commandline/rmi

Docker commands - Purge

Purge temporary files:

- all stopped containers
- all networks not used by at least one running container
- all volumes not used by at least one running container
- all images without at least one running container associated to them
- all build cache

Command: `docker system prune -a (--volumes)` <-- To also clean volumes

Tutorials

- Install Docker
- Create a Docker Hub account
- Thanks to the previous slides, push the hello-world image onto your Docker Hub namespace