Lesson 1. What is Docker

What is Docker

https://en.wikipedia.org/wiki/Docker_(software)

> Docker is an open-source project that automates the deployment of applications inside software containers

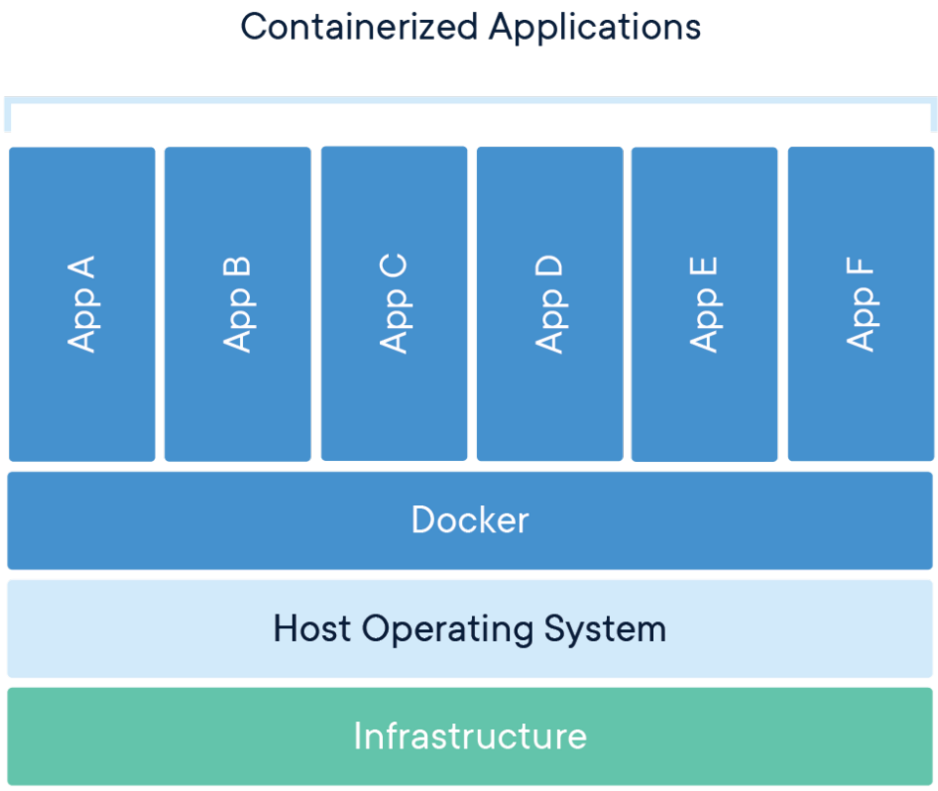
https://www.docker.com/resources/what-container

> A Docker container image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings.

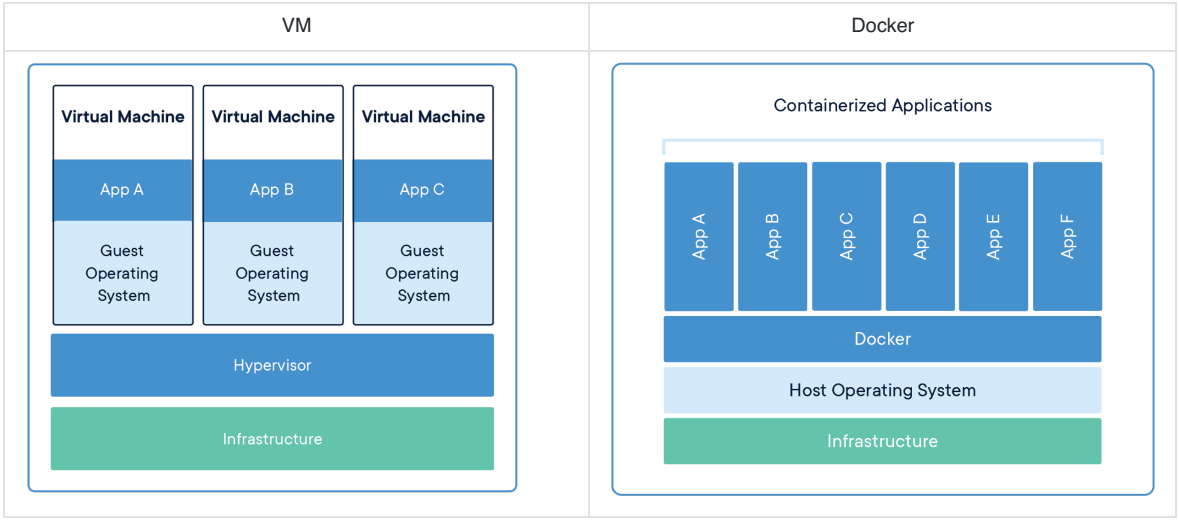
How does Docker work?

Linux containers (LXC) uses the resource isolation features of the Linux kernel to allow independent "containers" to run within a single Linux instance, avoiding the overhead of starting and maintaining virtual machines.

Docker is basically built on top of LXC and offers some high level LXC features with useful interface.



Docker vs Virtual machines



The advantages of Docker

Pros:

- Rapid application deployment
- Portability across machines
- Version control and component reuse
- Sharing
- Lightweight footprint and minimal overhead
- Simplified maintenance

Cons:

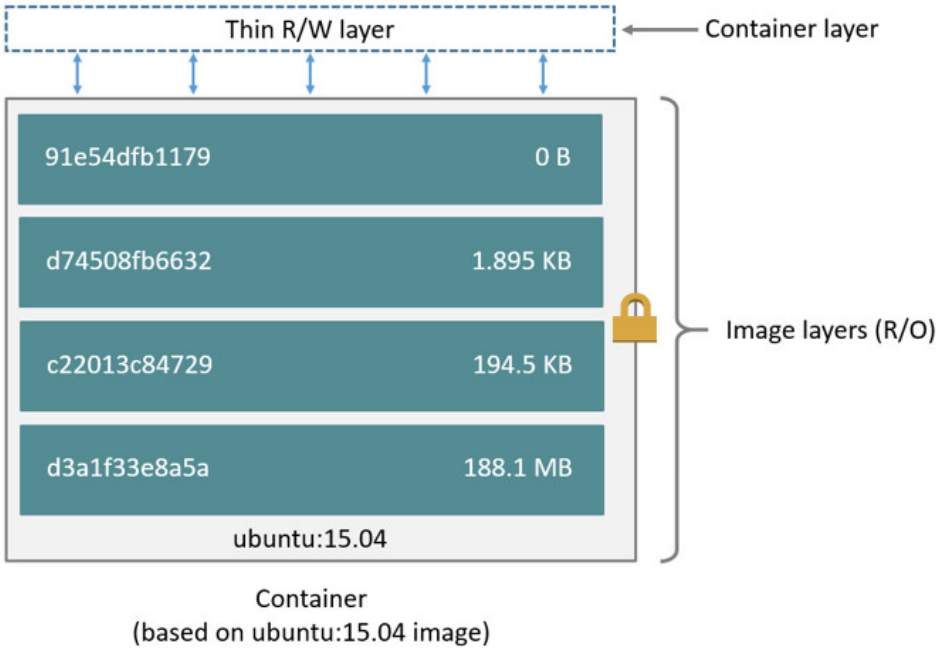
- DevOps

Understanding `Images` & `Containers`

https://docs.docker.com/v17.09/engine/userguide/storagedriver/imagesandcontainers/

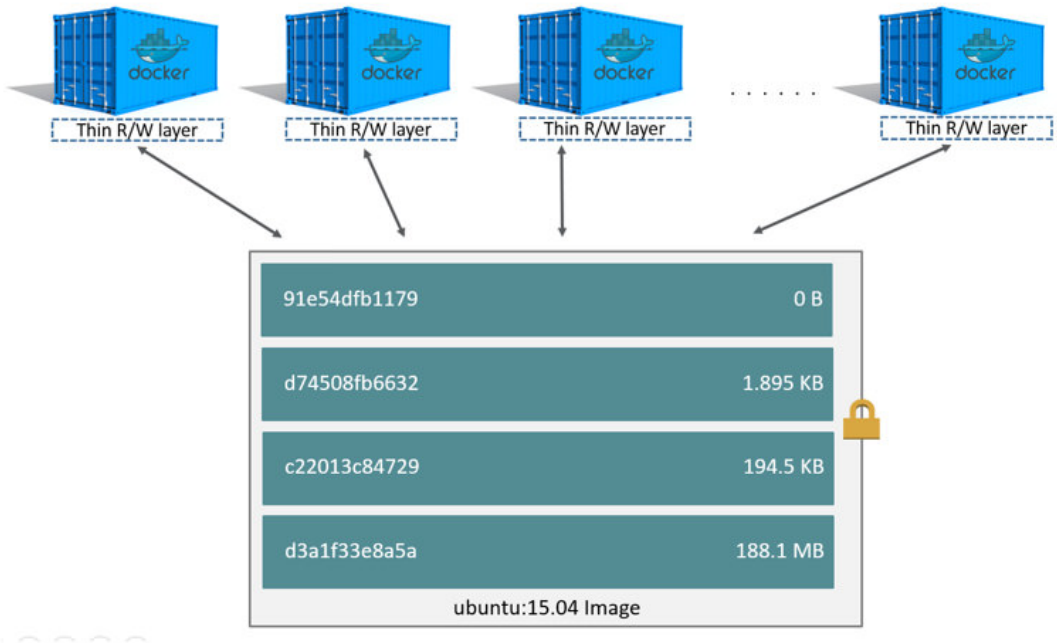
An `image` and its layers

An image is a filesystem and parameters to use at runtime. It doesn't have state and never changes.



A `container` and its layers

A container is a running instance of an image with the top writable layer.



Docker register

https://hub.docker.com is a cloud-based registry service which allows you to link to code repositories, build your images and test them, stores manually pushed images. Read more on https://docs.docker.com/docker-hub.

https://store.docker.com is the best way to discover high-quality Docker content (and bay it if needed). Read more on https://docs.docker.com/docker-store.

Commands to work with a register

```
# download
docker search hello
docker pull hello-world
# upload
docker login -h
docker push -h
```

Run a container

Container lifecycle

```
docker ps -a
docker create --name fc hello-world
docker start --attach fc
docker rm fc
```

`run` command

```
docker run --rm --name fc hello-world
docker run --help
```

Read more on https://docs.docker.com/engine/reference/run/.

Most used `run`'s options

```
-d, --detach           Run container in background and print container ID
-i, --interactive      Keep STDIN open even if not attached
-t, --tty              Allocate a pseudo-TTY
--name string          Assign a name to the container
--rm                   Automatically remove the container when it exits
```

Passing custom arguments to a container

```
docker run docker/whalesay cowsay 'Hello dear student! :)'
```

where

- `run` – load image, if absent, and run
- `docker/whalesay` – a name of the image
- `cowsay` – a command to run
- `'Hello dear student! :)'` - argument to pass in the command

Get Docker

Check out from https://store.docker.com/editions/community/docker-ce-desktop-windows or https://store.docker.com/editions/community/docker-ce-desktop-mac or https://docs.docker.com/install/linux/docker-ce/debian/.

The installation includes:

- Docker Engine
- Docker CLI client
- Docker Compose
- Docker Machine

Homeworks

Please send the results of homeworks as an email.
Please use the following template:

- *Subject:* `[Docker] Homework 1`
- *To:* `trainer's email`
- *Body:* `your homework as a plain text - NO ATTACHMENTS!!!`

Homework 1.1 (mandatory)

Create a Docker HUB account (if required) and login to the register via CLI interface. Please send a link to your HUB account.

Homework 1.2 (mandatory)

Install Docker and send the output of the following commands:

```
docker --version
docker-compose --version
```

Homework 1.3 (optional)

Please send a command to display a name of current system user for `ubuntu` image.