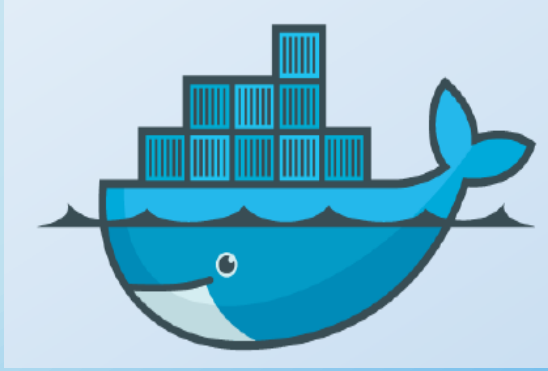
Building Custom Docker Images



Agenda

1. Docker Images
2. Dockerfile
3. Instructions in Dockerfile
4. Creating Custom Images

Docker Image

An image is a read-only template which doesn’t have any state.

A Docker image is a collection of all the files that make up an executable software application.

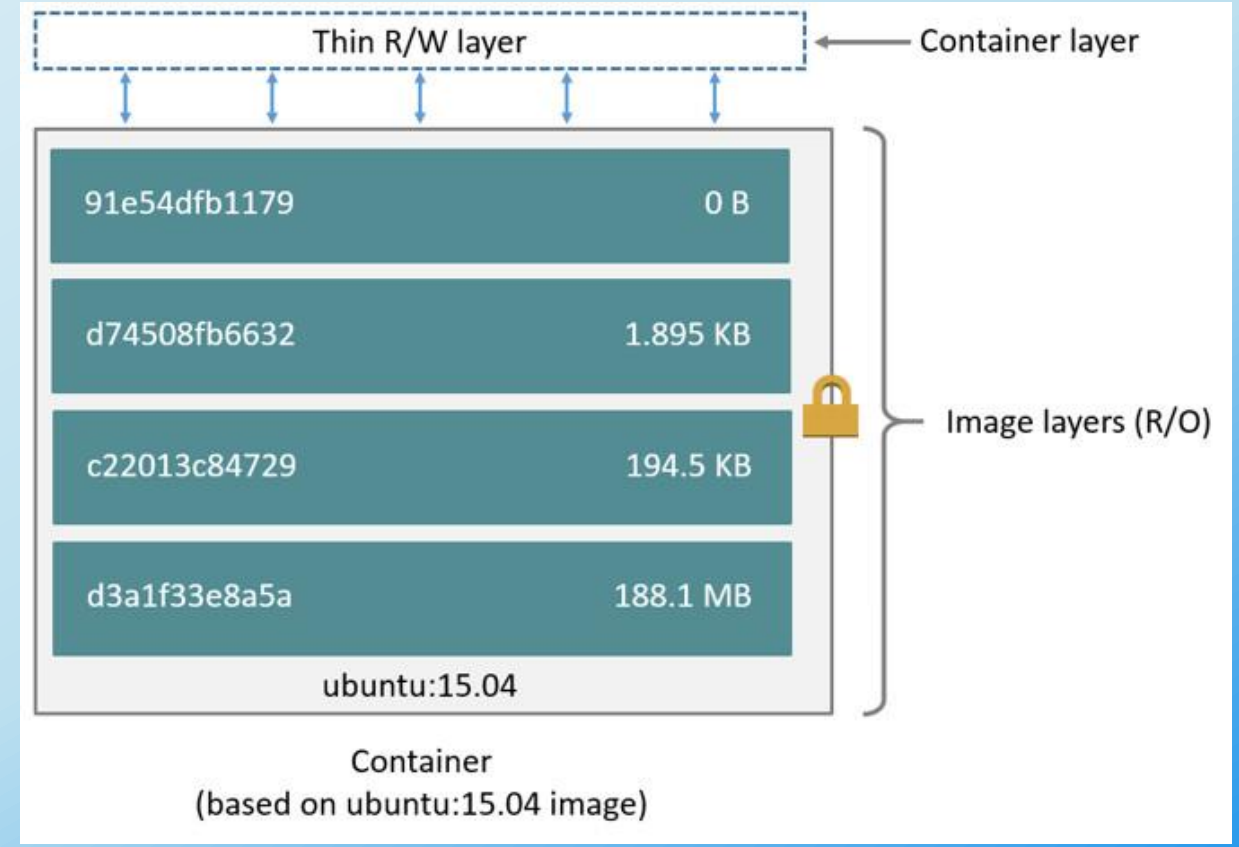
This collection includes the application plus all the libraries, binaries, and other dependencies

Docker image also contains the instructions for creating a Docker container.

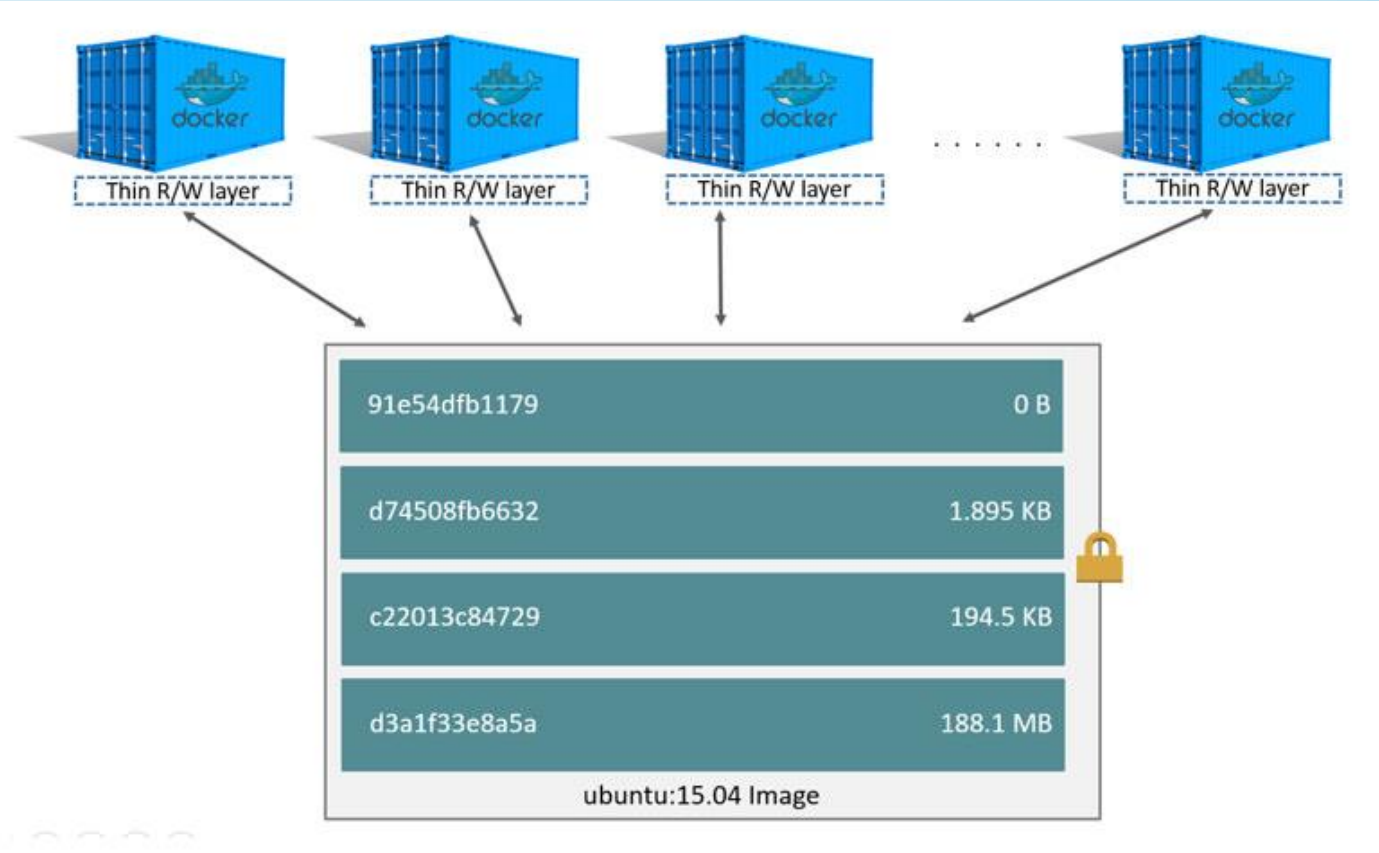
Docker Image

1. Images are built up from series of layers
2. Each layer represents changes from the prior
3. Layers are immutable (read-only) and referenced by hashes and optional tags.
4. Containers run with a file system based on an image with a thin read/write layer added.
5. Plus any volume mounts specified.

Docker Image



Docker Containers from Image



Docker Build

It describes how Docker images are built using Dockerfile, which is the standard way for building highly re-usable Docker images.

We don't need any additional tools installed, because it a simple command built into the docker engine.

Dockerfile

Dockerfile is a text-based build script that contains special instructions in a sequence for building the correct and relevant images from the base images.

The Docker Engine tightly integrates this build process with the help of the docker build subcommand.

The filename of this text based script name is recommended to be Dockerfile

Dockerfile

The sequential instructions inside Dockerfile can include

1. selecting the base image
2. installing the required application
3. adding the configuration and the data files
4. exposing those services to the external world
5. configuring the commands to start services

Syntax for Dockerfile

A Dockerfile is made up of instructions, comments, parser directives, and empty lines, as shown here:.

The INSTRUCTION can be written in any case, in other words, it is case-insensitive.

However, the standard practice or the convention is to use uppercase in order to differentiate it from the arguments.

Dockerfile Instructions

The Dockerfile instructions are

1. FROM
2. LABEL
3. MAINTAINER
4. COPY
5. ADD
6. EXPOSE
7. ENV
8. ARG
9. USER
10. WORKDIR
11. CMD

FROM Instructions

- The FROM instruction is the most important one and is the first valid instruction of a Dockerfile.
- It sets the base image for the build process.
- Subsequent instructions will use this base image and build on top of it.

```
FROM centos
FROM ubuntu:16.04
FROM
ubuntu@sha256:8e2324f2288c26e1393b63e680ee7844202391
414dbd48497e9a4fd997cd3cbf
```

Maintainer Instruction

The MAINTAINER instruction is an informational instruction of a Dockerfile.
This instruction allows us to provide the details of who wrote the Dockerfile.

MAINTAINER Goutham Kumar <example@gmail.com>

LABEL Instruction

The LABEL instruction enables you to add key-value pairs as metadata to your Docker images. These metadata can be further leveraged to provide meaningful Docker image management and orchestration

```
LABEL appName="Offers App"

LABEL appName="customercare" \
        AppVersion="1.2.3"
```

Workdir Instructions

The WORKDIR instruction will change the directory inside the docker Image file system.

It is same as CD command, but executed inside the Image.

WORKDIR /usr/src/app

Default directory while creating docker image is **"/"**

Copying Instructions

The COPY instruction enables you to copy the files from the Docker host to the filesystem of the new image.
The following is the syntax of the COPY instruction:
COPY <src> ... <dst>

The ADD instruction is similar to the COPY instruction. However, in addition to the functionality supported by the COPY instruction, the ADD instruction can handle the TAR files and remote URLs.

ADD /opt/abc.zip /opt (first copy + unzip)
ADD http://putty.org/putty.rpm /opt

ENV Instruction

The ENV instruction sets an environment variable in the new image.
An environment variable is a key-value pair, which can be accessed by any script or application.

ENV JAVA_HOME /opt/java1.8/bin/java

ENV NAME Hello_World

EXPOSE Instructions

The EXPOSE instruction opens up a container network port for communicating between the container and the external world.

The syntax of the EXPOSE instruction is as follows:

EXPOSE <port>[/<proto>]

EXPOSE 8080

EXPOSE 7373/udp

RUN Instructions

The RUN instruction is the real workhorse during the build, and it can run any command.

The general recommendation is to execute the multiple commands using one RUN instruction.

This reduces the layers in the resulting Docker image

RUN yum install –y wget

RUN mkdir fluent_logs

CMD Instructions

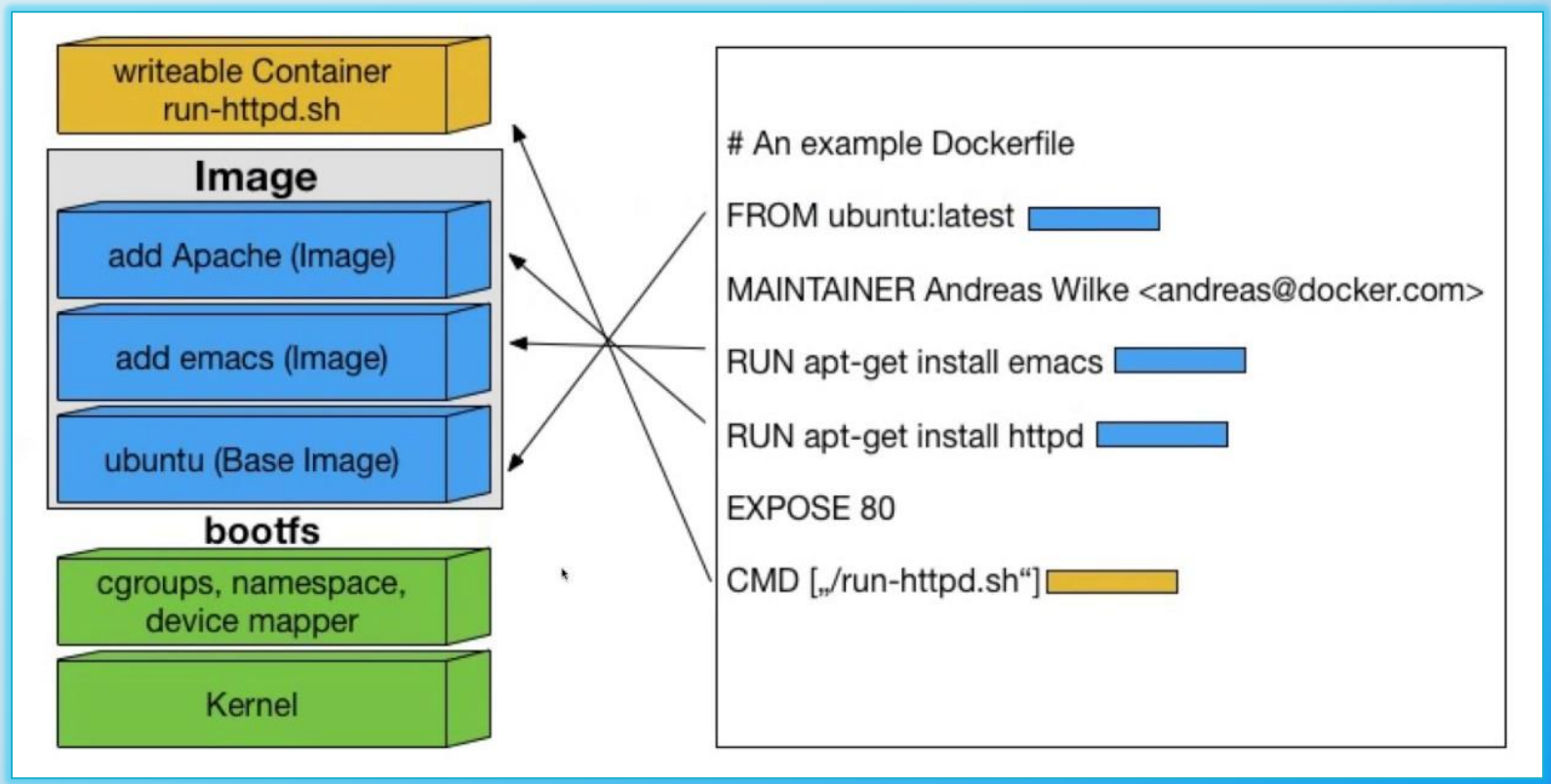
The CMD instruction can run any command (or application), which is similar to the RUN instruction. However, the major difference between these two is the time of execution.

The command supplied through the RUN instruction is executed during the build time, whereas the command specified by the CMD instruction is executed when the container is launched from the newly created image.

Thus, the CMD instruction provides a default execution for this container.

CMD ["echo", "Welcome to Docker"]

Sample custom docker image



Q & A