Mostafa Wael

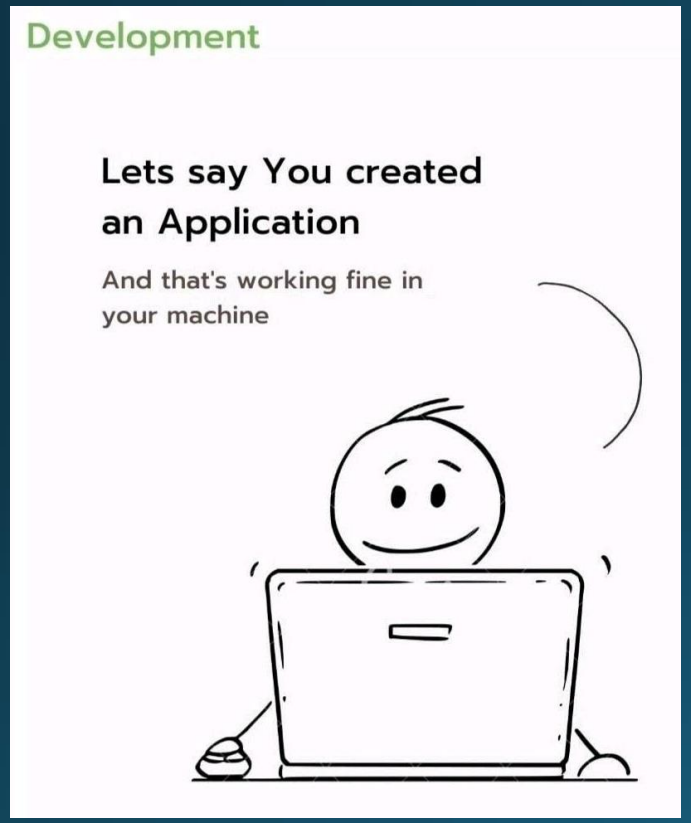
Dockers

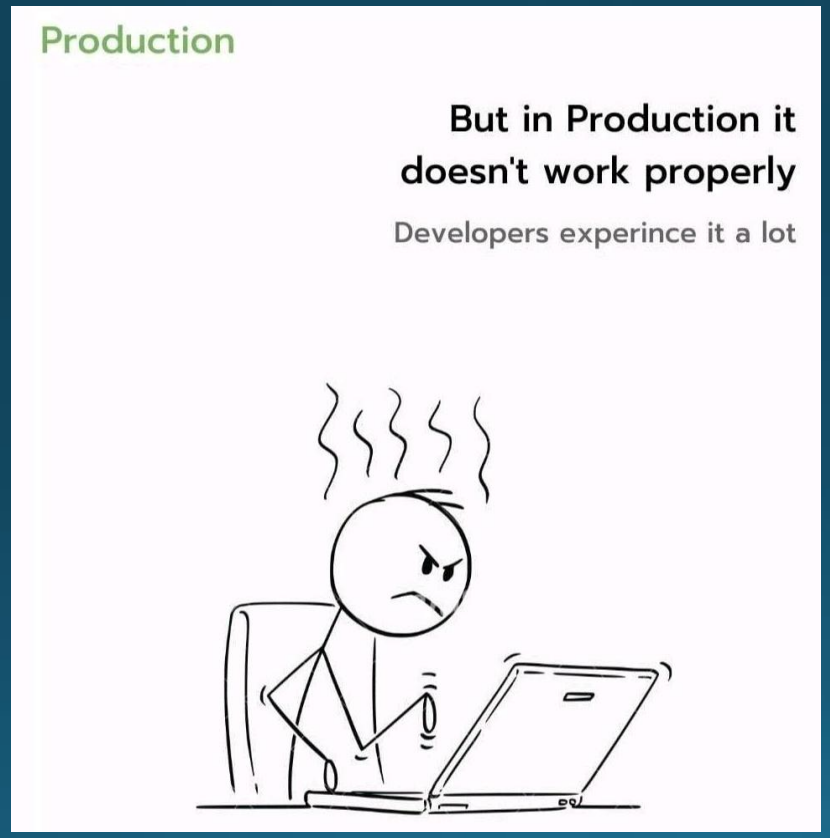
Agenda

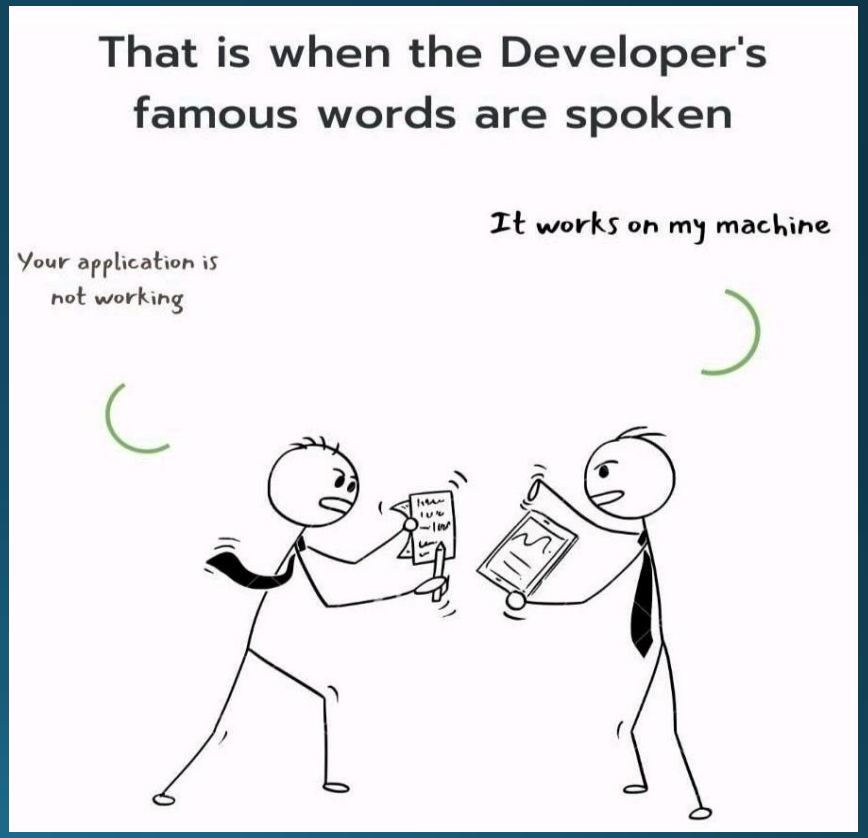
- Intro + Why?
- What is docker? How it works?
- Difference between docker, virtual machine, python env
- Docker image? Container? File?
- Docker commands
- Docker file + Layered Architecture
- Networking
- Volumes
- More?

Developers' Life!

Developers' Life!







Developers' Life!

- Why?
  - Dependencies.
  - Libraries and Frameworks.
  - OS Level Features.
  - Env variables.
  - Others…

Developers' Life!

So, what is Docker?

What is Docker?

We need a **standardized way to package the application with its dependencies** and deploy it on any environment.

Docker is a **tool designed to make it easier** to create, deploy and run applications by using containers.

It **packages an application and all its dependencies in a virtual container** that can run on any Linux server.

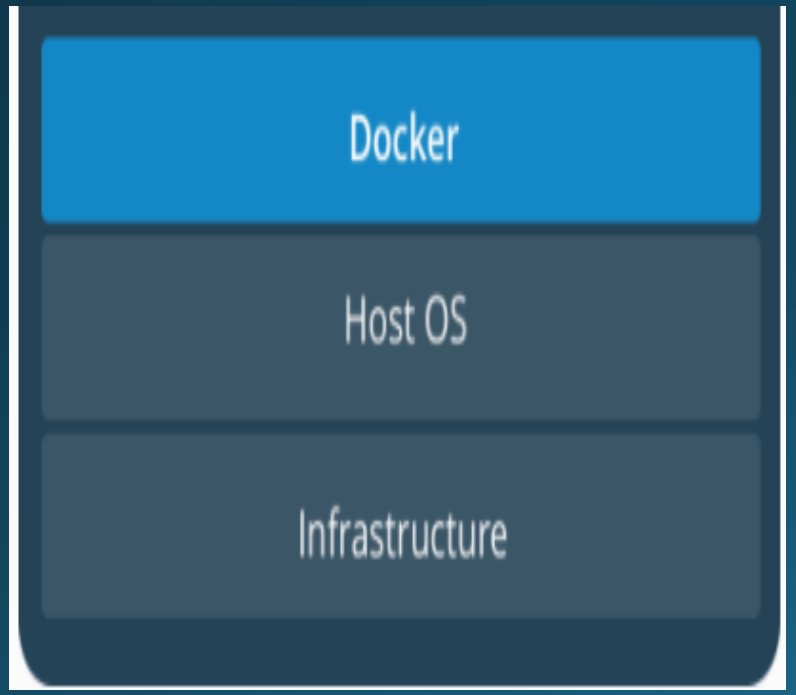
Each **container runs as an isolated process** in the user space and take up less space than regular VMs due to their layered architecture.

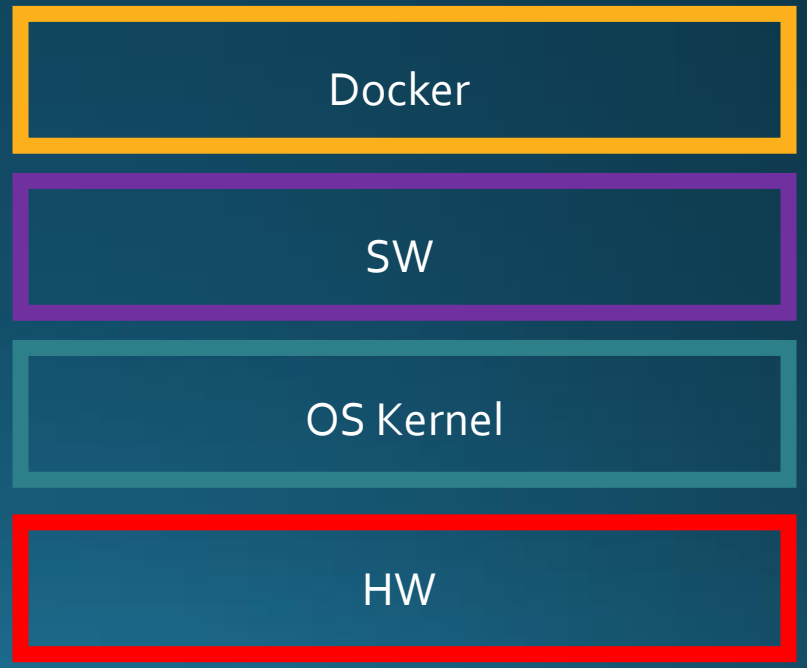
So, it **will always work the same regardless of its environment**.

More Use Cases?

What is Docker?

What is Docker?





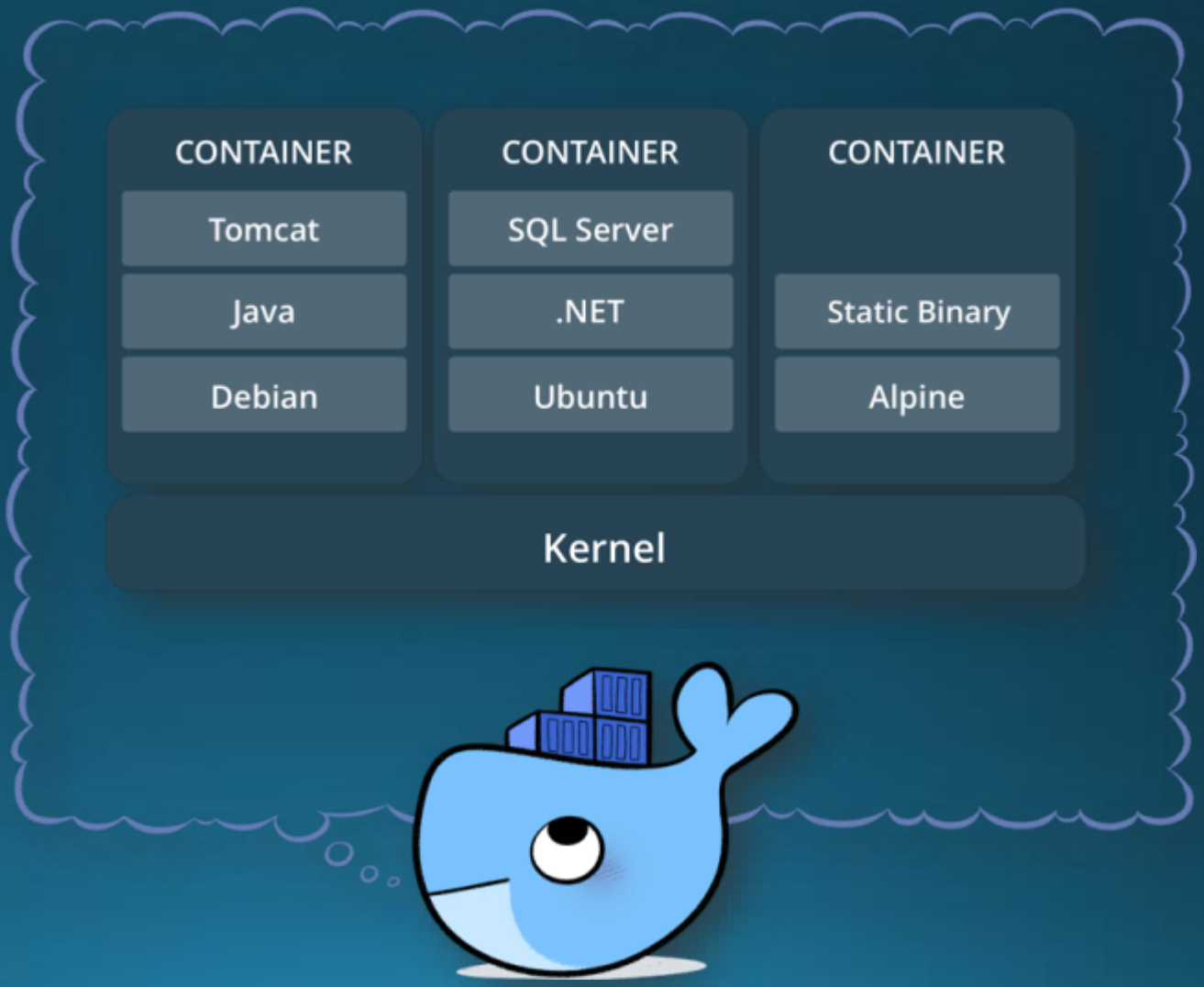
What is Docker Container?

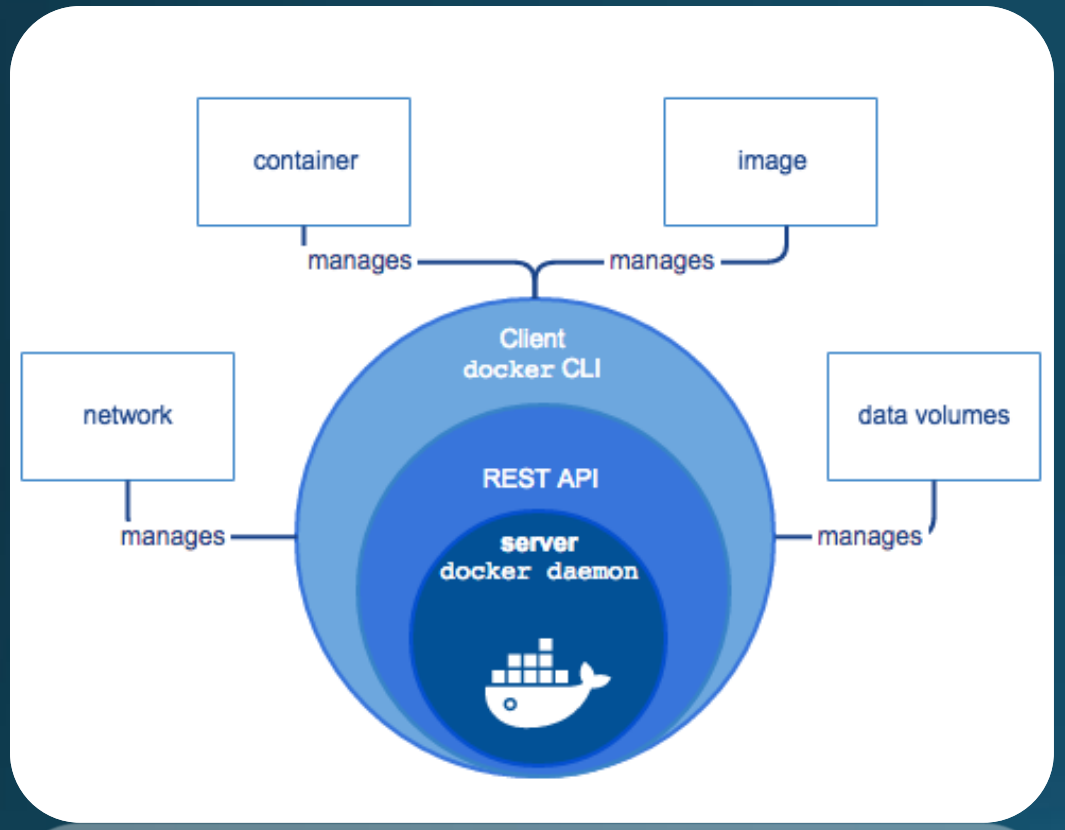
Docker

SW

OS Kernel

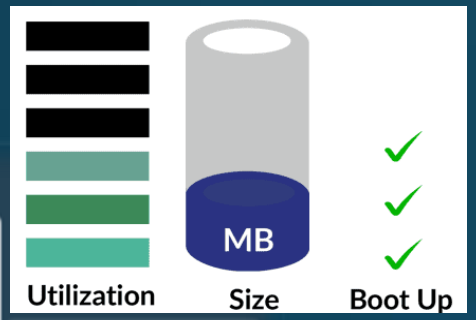
HW

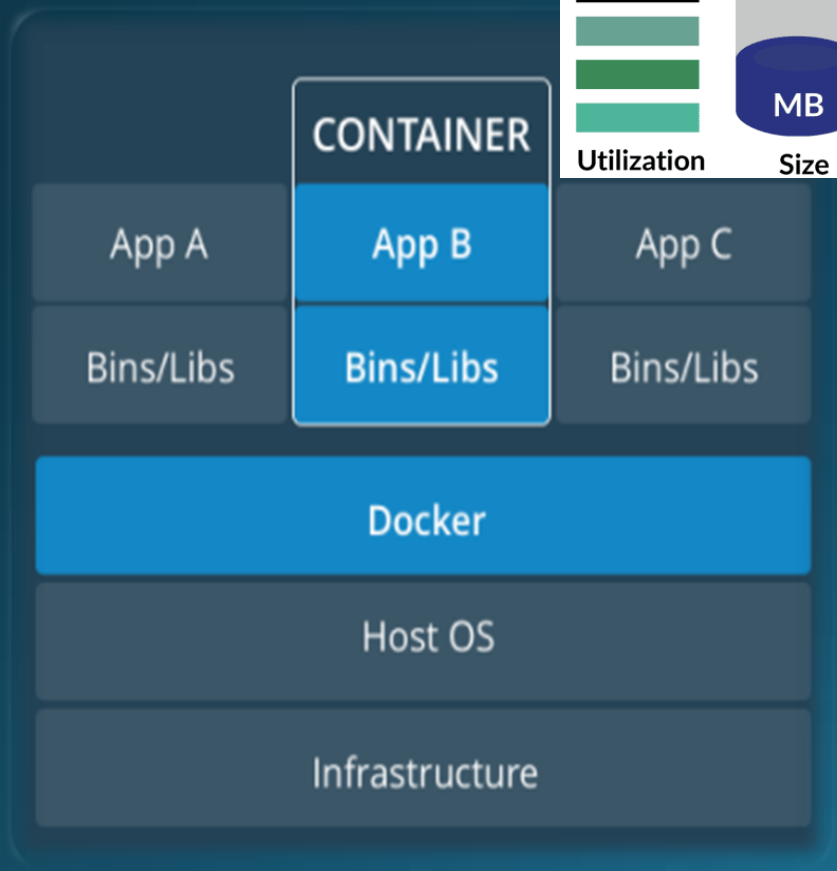


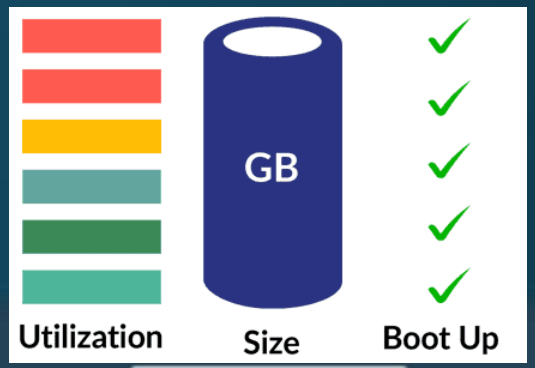


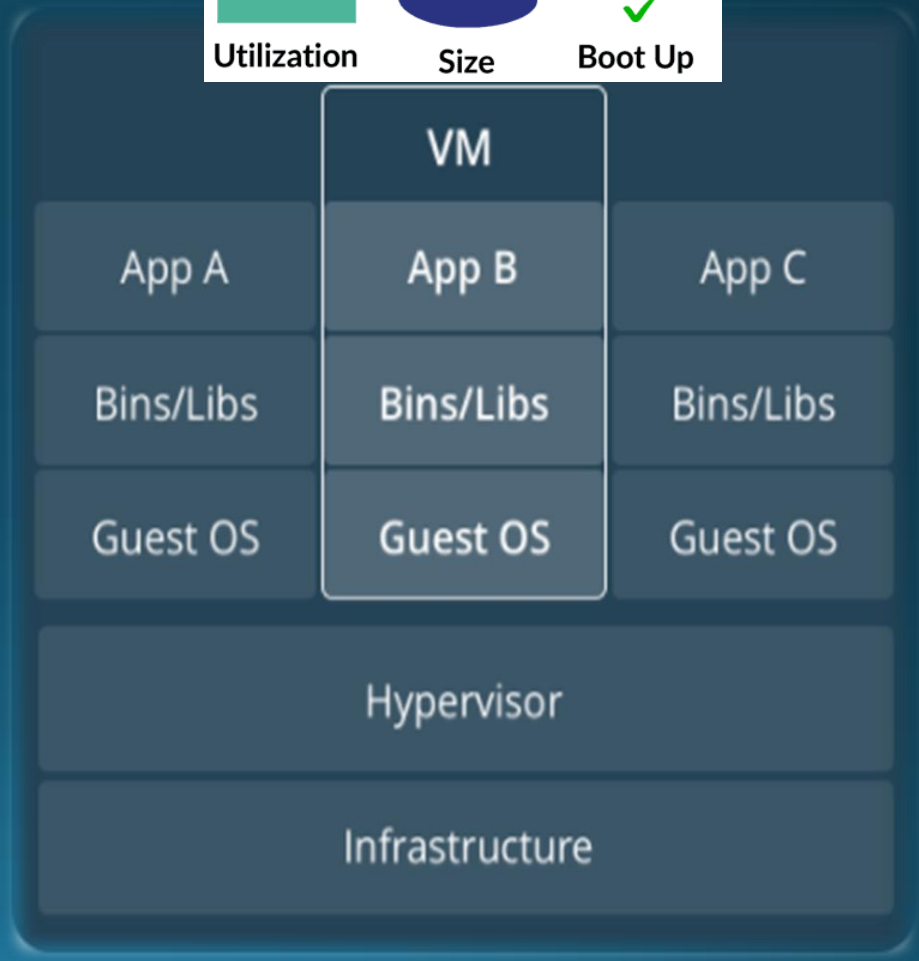
Docker Engine

IS Docker a VM?

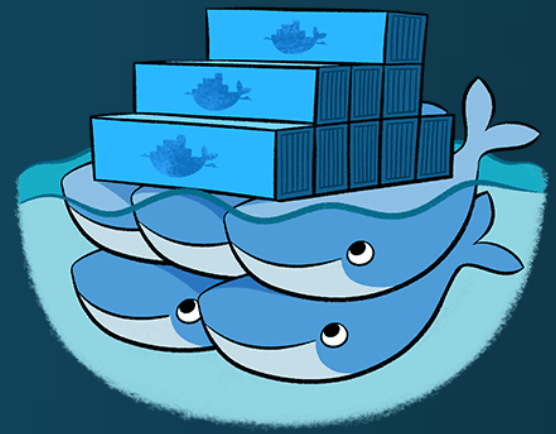


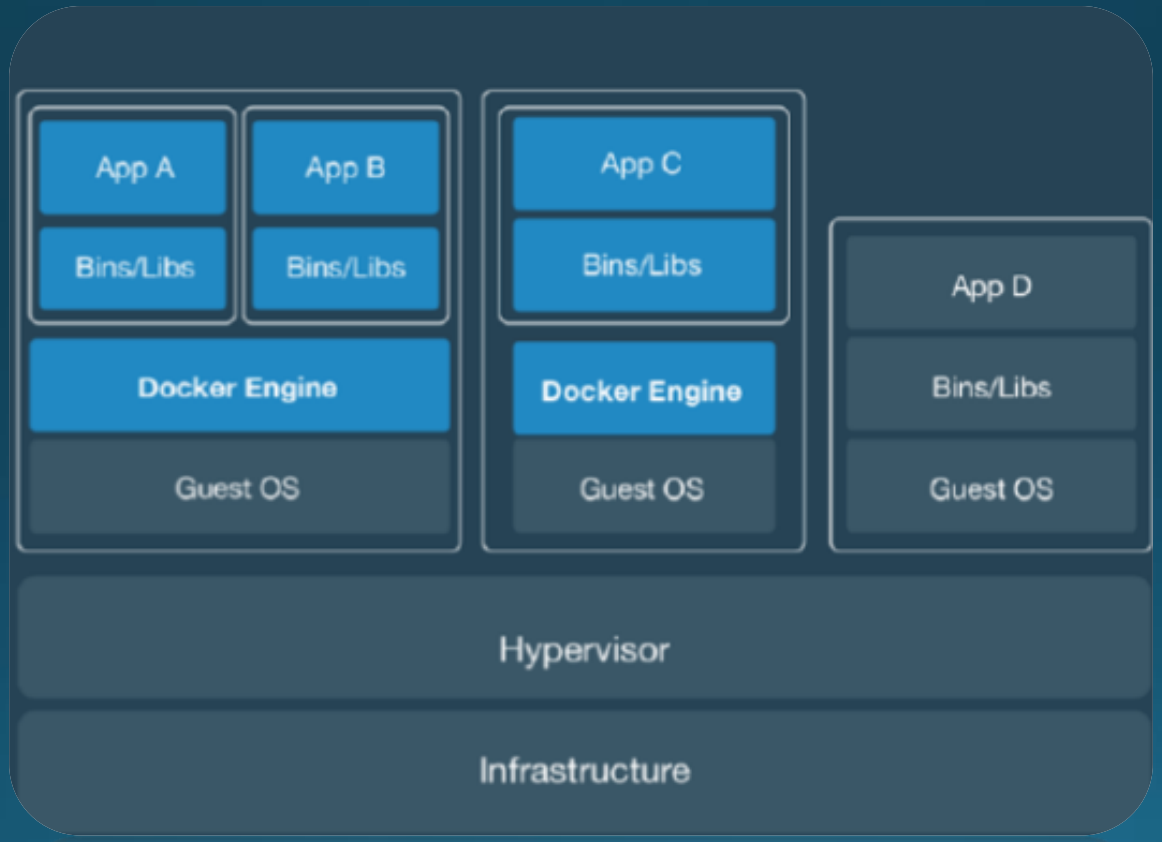


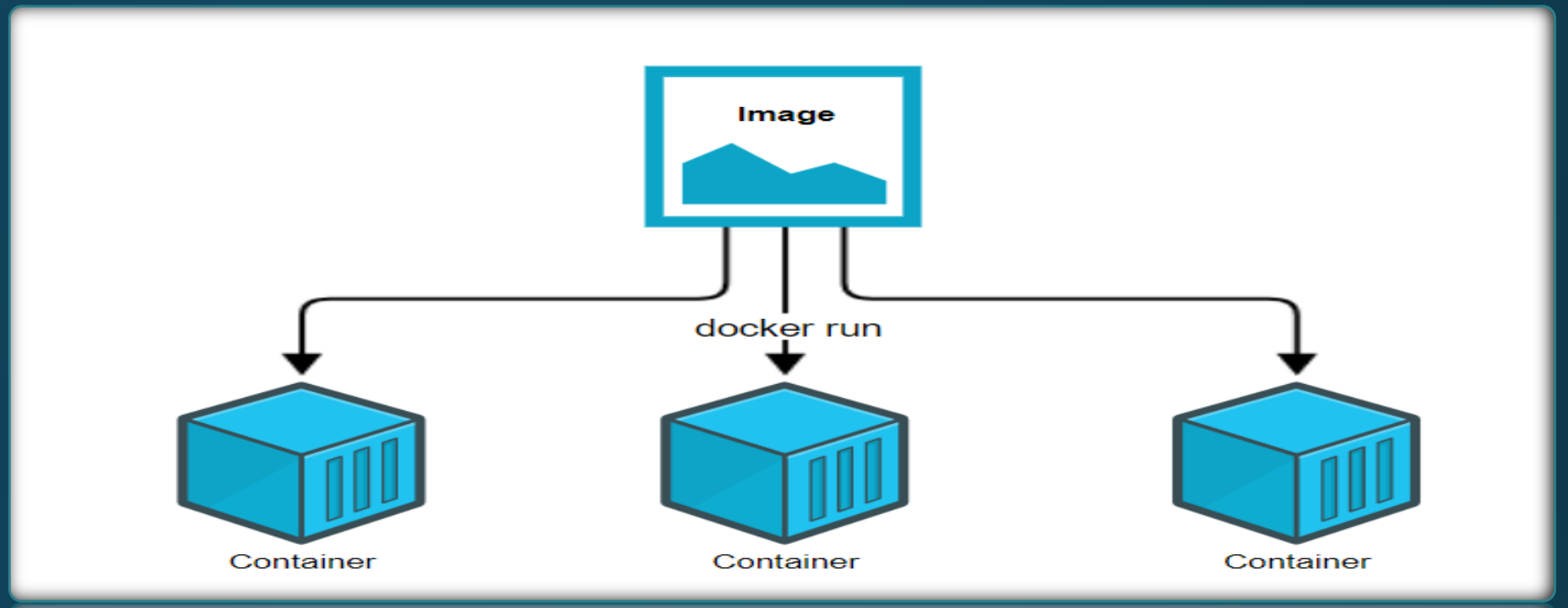




VMs can run Docker!







Images and containers?

To summarize

Bundled Dependencies: Docker images **contain all their own dependencies**, (you don't have to do any installation yourself)

Cross-platform Installation: Docker containers contain their own operating system, so they will **run on any platform** (even Windows!)

Easy Distribution - Can be distributed as a single **.tar** image file, or put on **docker hub** so it can be pulled

To summarize

Safety: Files in a container **can't access files on the host machine**, so users can trust dockerized applications

Ease-of-Use: Docker containers can always be **run using one single docker run command**

Easy Upgrades: Docker containers can be easily swapped out for **newer versions**, while all persistent **data can be retained** in a data volume

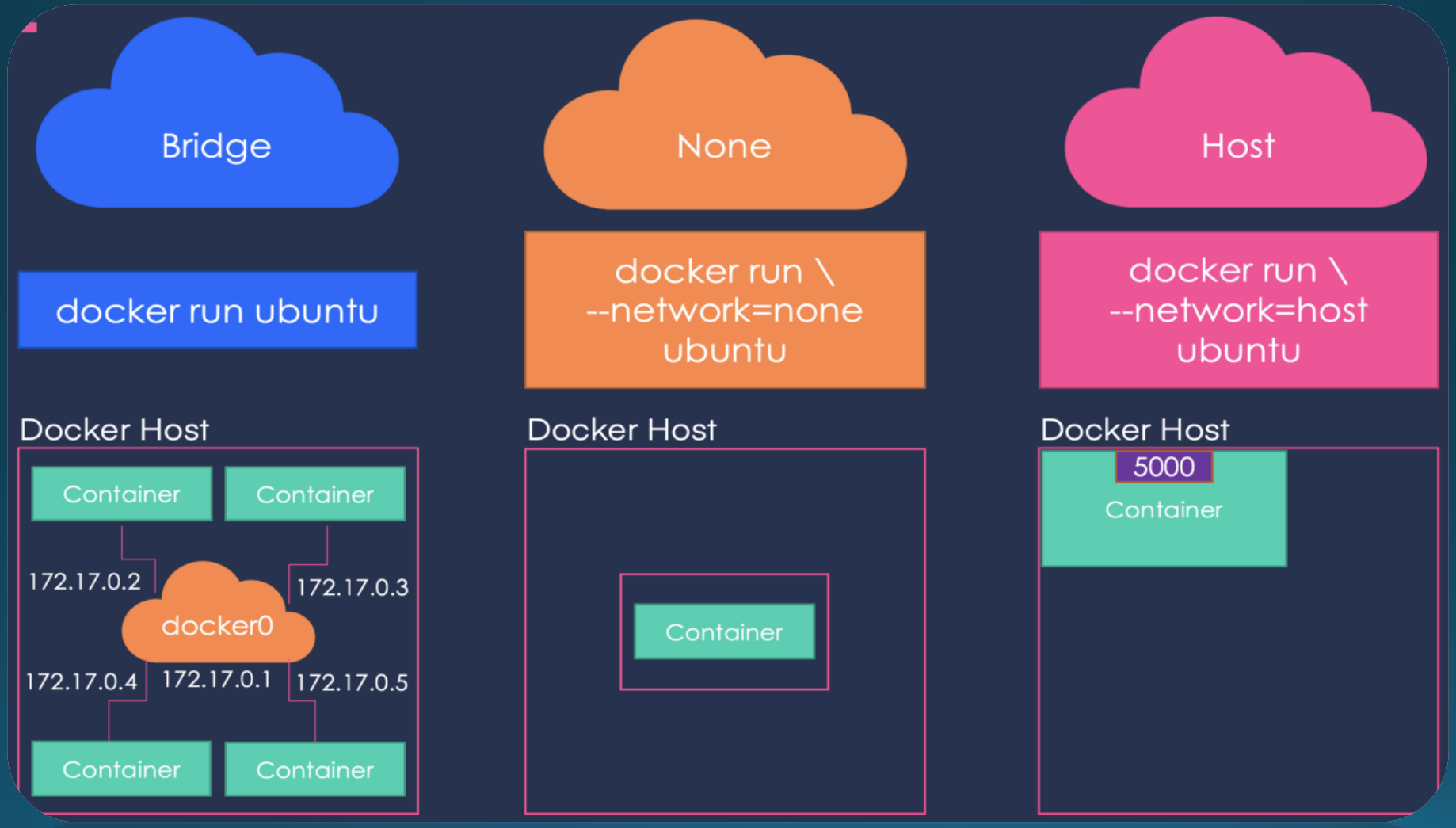
Docker Commands

```
[node1] (local) root@192.168.0.28 ~
$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:37a0b92b08d4919615c3ee023f7ddb068d12b8387475d64c622ac30f45c29c51
Status: Downloaded newer image for hello-world:latest

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

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.
```

Docker Commands Exmaples

Let's run Docker containers!

Docker Network



Persistent Data

By default, Docker containers cannot access data on the host system. This means

- You can't use host data in your containers
- All data stored in the container will be lost when the container exits

You can solve this in two ways:

- Volumes
- Binding Mounts

Persistent Data: volumes

-v volume_name:/path/in/container

This mounts a named volume into the container, which will live separately from the rest of your files.

This is preferred, unless you need to access or edit the files from the host

Persistent Data: Binding Mounts

-v /path/in/host:/path/in/container

This bind mounts a host file or directory into the container. Writes to one will affect the other.

Note that both paths must be absolute paths, so you often want to use `pwd`/some/path

Docker Files

Docker File

The core of making a Docker image is a **Dockerfile**

These files should have the exact name "**Dockerfile**" and should be in their own directory.

Docker File

- A Dockerfile is a list of commands, a lot like a shell script, that progressively builds the image:
  - **FROM** lists the image to "inherit" from
  - **RUN** executes a shell command
  - **COPY** copies some data from the host to the image
  - **ENTRYPOINT** sets the command that will be run when a container is created
  - **WORKDIR**, like cd, sets the current working directory for the build script

Docker File

```
1   FROM node:16
2
3   # Create app directory
4   WORKDIR /usr/src/app
5
6   # Install app dependencies
7   # A wildcard is used to ensure both package.json AND package-lock.json are copied
8   COPY package*.json ./
9
10  RUN npm install
11  # If you are building your code for production
12  # RUN npm ci --only=production
13
14  EXPOSE 8080
15  CMD [ "node", "server.js" ]
16
17
```

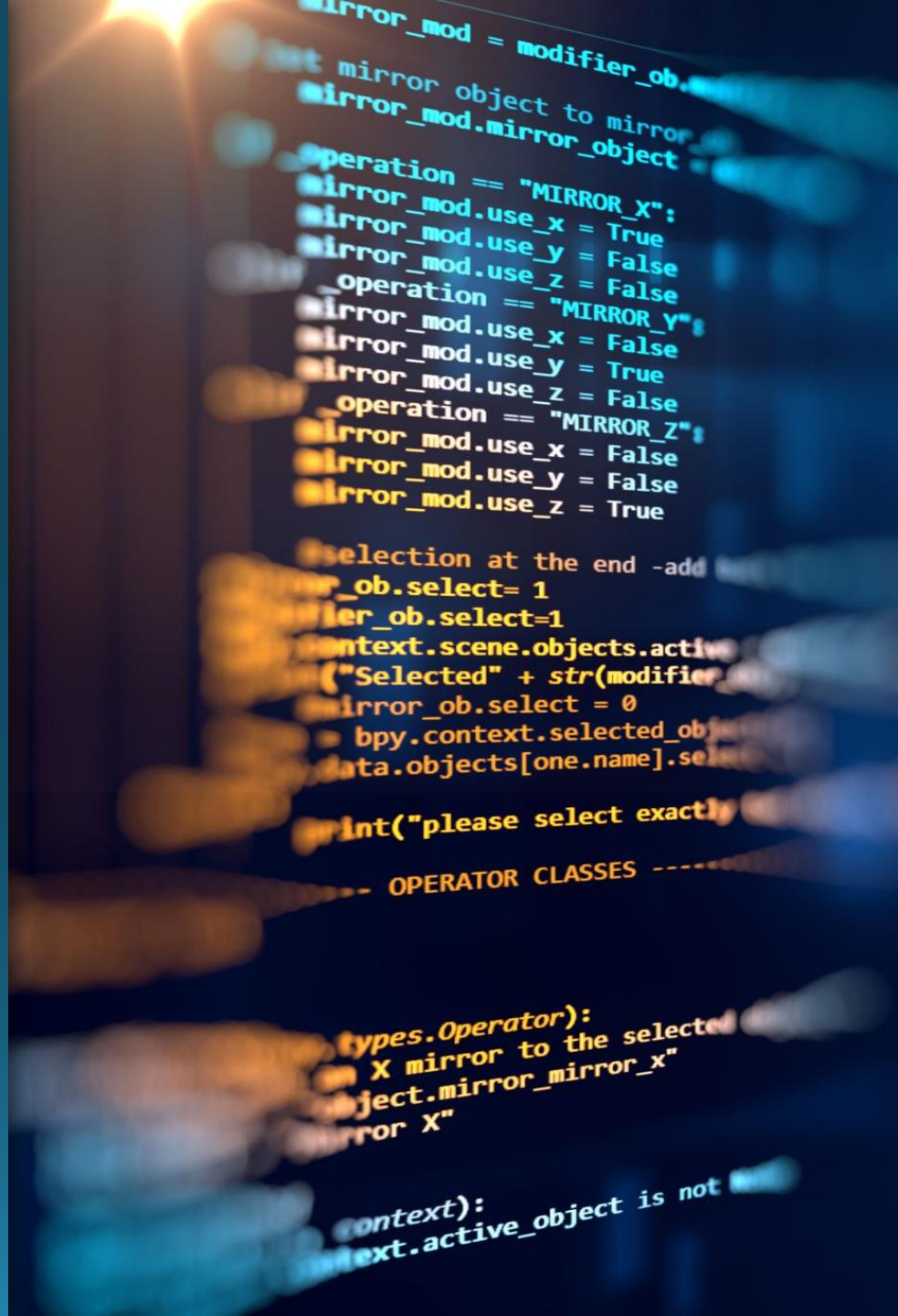
Web Apps

```
[node2] (local) root@192.168.0.27 ~/docker-hello-world-spring-boot
$ curl localhost:8080
Hello World
[node2] (local) root@192.168.0.27 ~/docker-hello-world-spring-boot
$
```

```
docker run -p 8080:8080 -it --rm hello-world-java

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::        (v2.2.1.RELEASE)

2021-10-30 01:19:52.151  INFO 1 --- [           main] c.d.hello.Application                    : Starting Application v0.1.0 on 2bcd3b9fc40c with PID 1 (/data/hello-
orld-0.1.0.jar started by root in /)
2021-10-30 01:19:52.176  INFO 1 --- [           main] c.d.hello.Application                    : No active profile set, falling back to default profiles: default
2021-10-30 01:19:56.679  INFO 1 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8080 (http)
2021-10-30 01:19:56.715  INFO 1 --- [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2021-10-30 01:19:56.722  INFO 1 --- [           main] org.apache.catalina.core.StandardEngine  : Starting Servlet engine: [Apache Tomcat/9.0.27]
2021-10-30 01:19:57.169  INFO 1 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
2021-10-30 01:19:57.170  INFO 1 --- [           main] o.s.web.context.ContextLoader            : Root WebApplicationContext: initialization completed in 4676 ms
2021-10-30 01:19:57.997  INFO 1 --- [           main] o.s.s.concurrent.ThreadPoolTaskExecutor  : Initializing ExecutorService 'applicationTaskExecutor'
2021-10-30 01:19:58.889  INFO 1 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8080 (http) with context path ''
2021-10-30 01:19:58.899  INFO 1 --- [           main] c.d.hello.Application                    : Started Application in 9.163 seconds (JVM running for 10.728)
^C2021-10-30 01:20:00.716  INFO 1 --- [extShutdownHook] o.s.s.concurrent.ThreadPoolTaskExecutor  : Shutting down ExecutorService 'applicationTaskExecutor'
```

More about the topic

- Docker compose
- Docker ignore and env file
- Docker registry
- Orchestration
- Kubernetes



Learning Resources

- **Docker Commands Exmaples**
- Docker playGround
- Docker Tutorial (NANA)
- Docker Tutorial (free code camp)