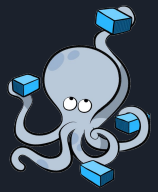
Introduction to Docker in Robotics

John Alejandro Duarte Carrasco

IN / johnaduarte

Goals

Docker



Docker for Development (ROS + Gazebo)

Github Actions + Docker

Goals

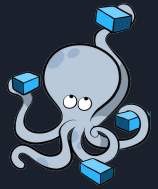
Docker

- Problem 💣.
- History of global transportation system🚢.
- Docker as a game changer 🚀 .
- Play with Containers 🎂.
- Images 📜 and Registries 💾!!
- Access the host ⇒ Volumes and networks 🔗.

Docker for Development (ROS)

Github Actions + Docker

Goals



Docker

Docker for Development (ROS)

- Let’s create a Image (Dockerfiles 📋) Git is that you?
- So many commands 😖! ⇒ Compose.
- Let’s group this 💼: Build, volumes, networks, command.
- Let’s quick off 🛫: Envs, resources, depends.
- Can I turn it on? 👀 Manipulate multiple services

Github Actions + Docker

Goals

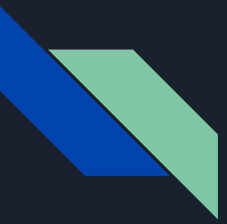
Docker

Docker for Development (ROS)

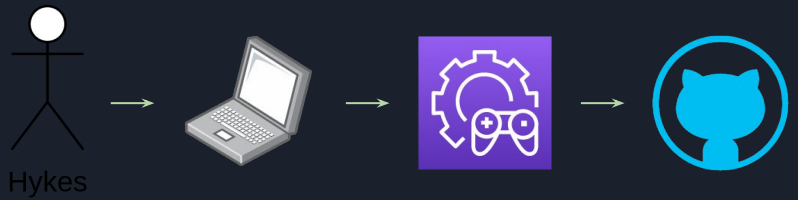
Github Actions + Docker

- Let’s breath… What is anyway CI?
- GA: Workflows, triggers and jobs.
- Check your code! Automatic test.

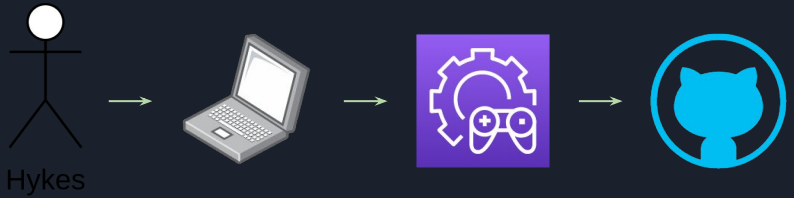
Ready?



Problem: Development



Problem: Development



Clone the repo
and compile it.

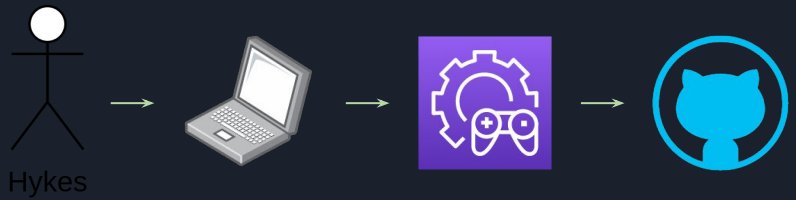
You need:
- Ubuntu 20.04
- ROS2 galactic.
- Gazebo.
- Python3 - pip3
- Tensorflow
- Nav2
- Third Repos...
- Tools and configs…
- ...

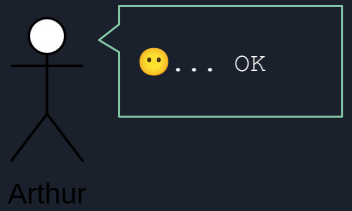
Arthur

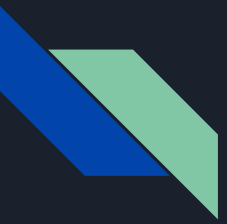
Can I be
part of?

How can I
compile it?

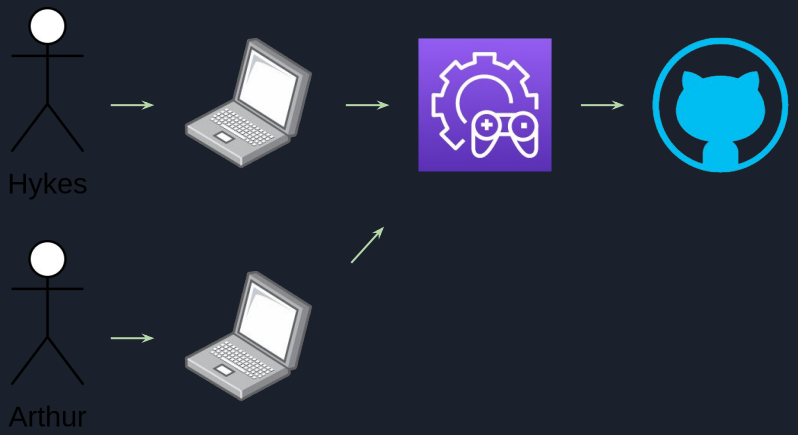
Problem: Development



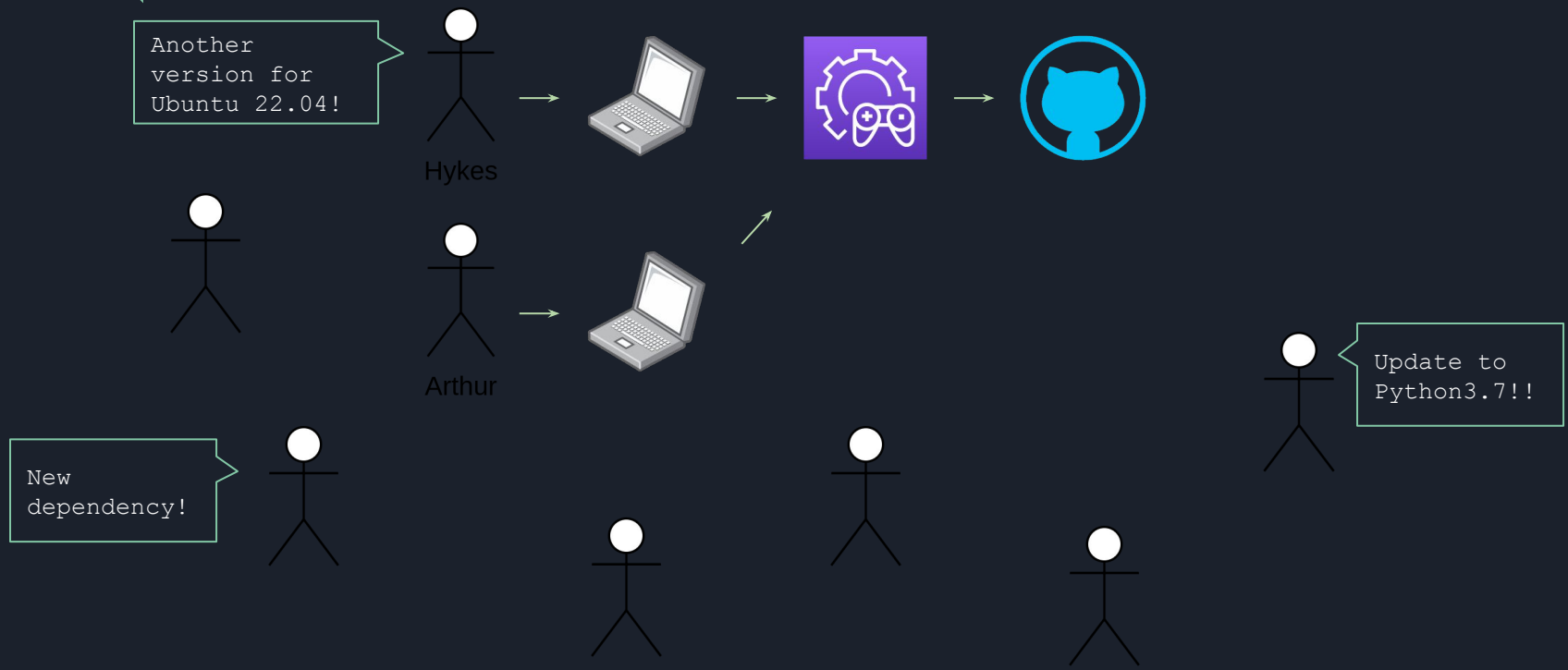




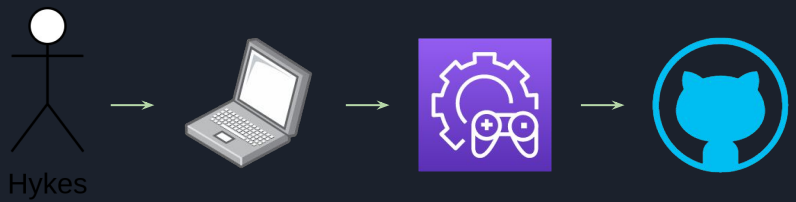
Problem: Development



Problem: Development



Problem: Development



There is a problem of **development environments.**

Similar, maintainable, scalable environment for the developers?

Problem: Testing

Problem: Testing

Problem: Testing

App V1

App V2

App Exp

Use all the time?

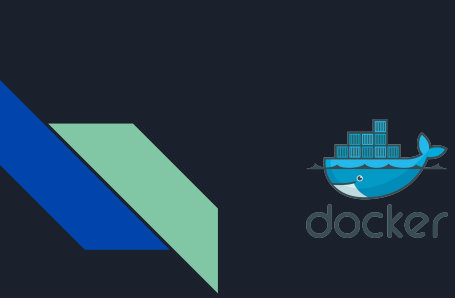
New server for versions?

How much it cost?

Problem: Testing

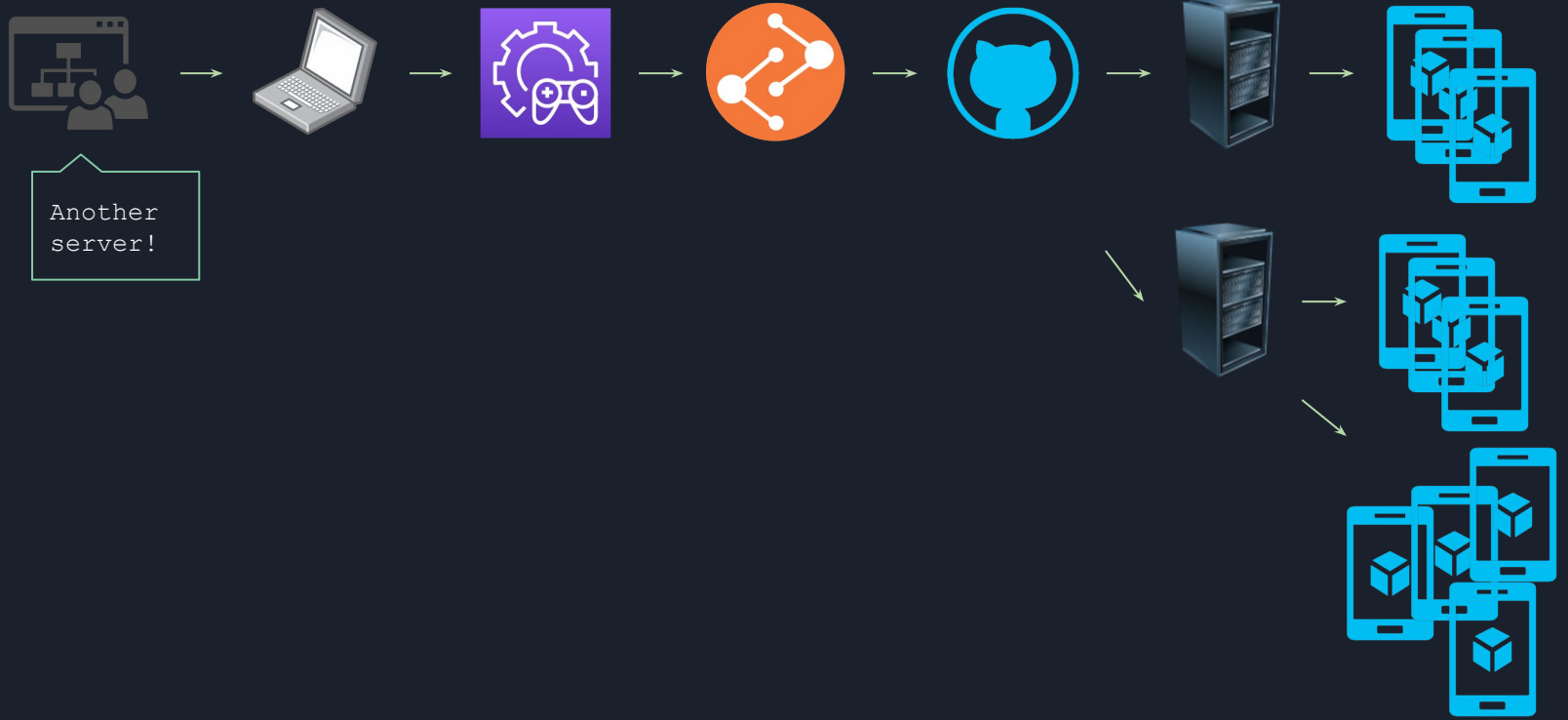
There is a problem of **testing environments.**

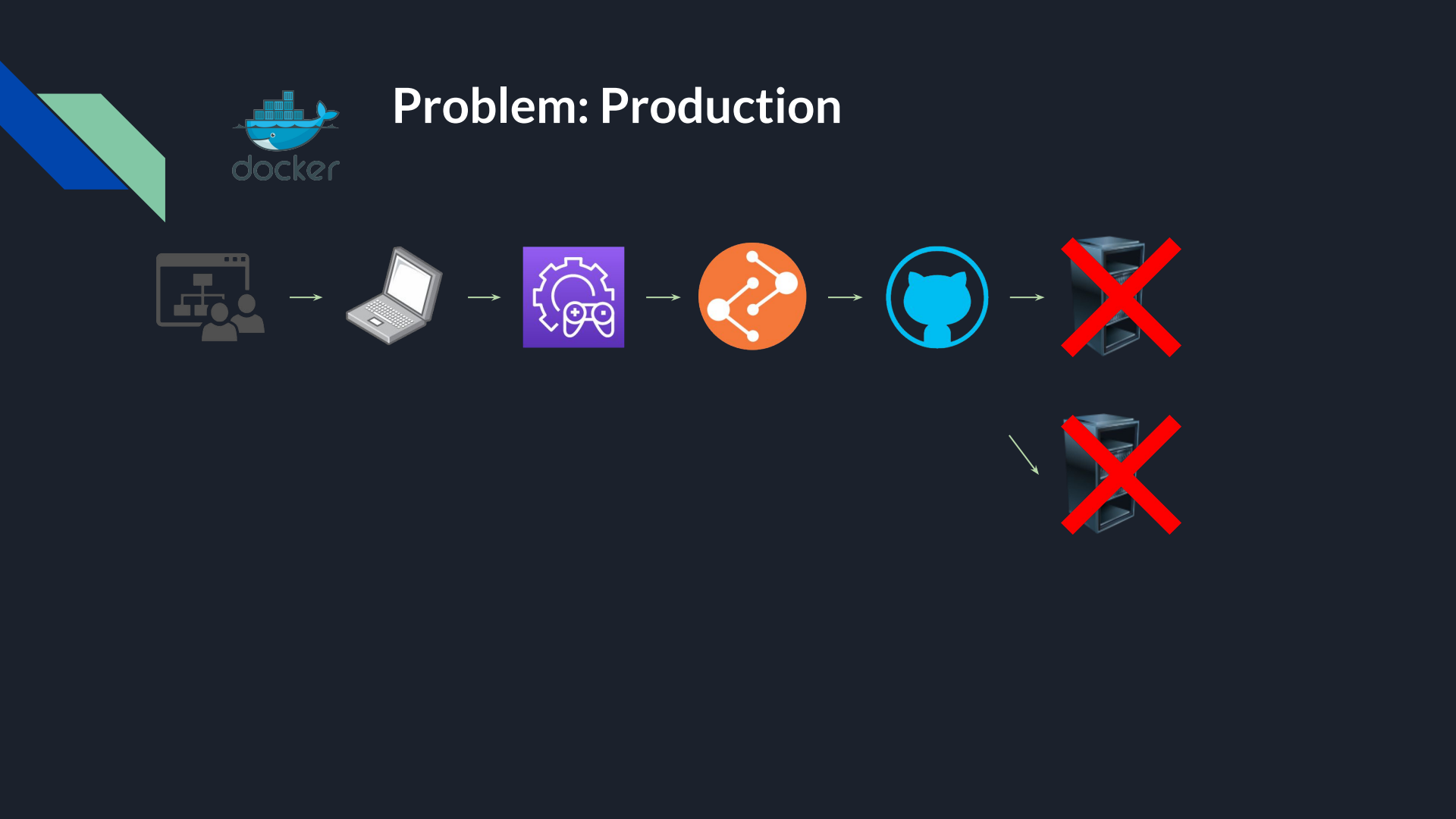
Configurable, maintainable and scalable environment for testing?



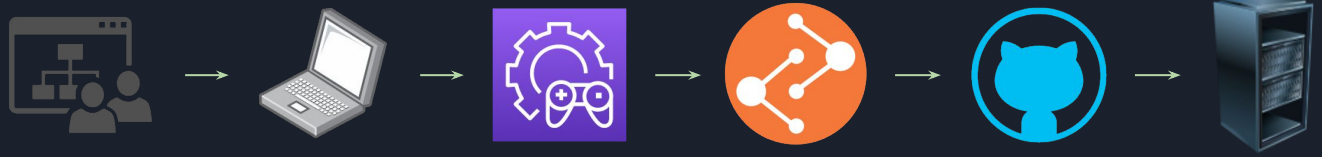
Problem: Production

Problem: Production



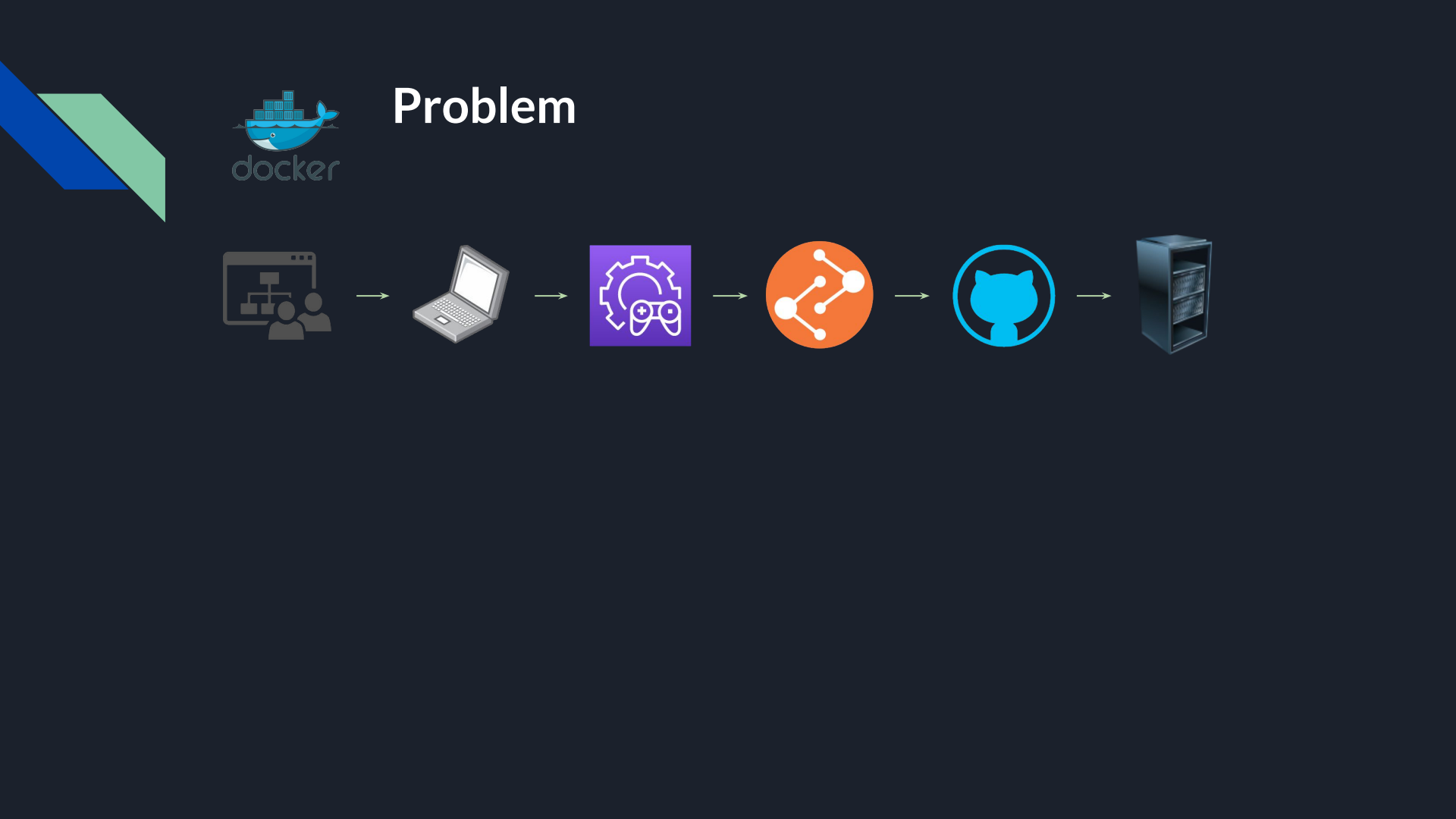

docker
Problem: Production

Problem: Production

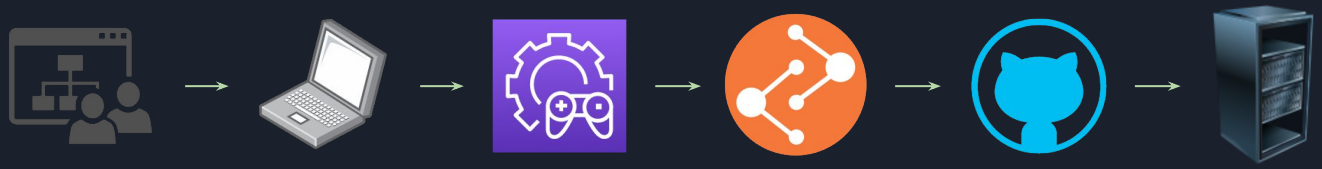


There is a problem of **production environments.**

Automatic, scalable, measurable environments for production?


docker
Problem

Problem



There is a problem of **development environments.**

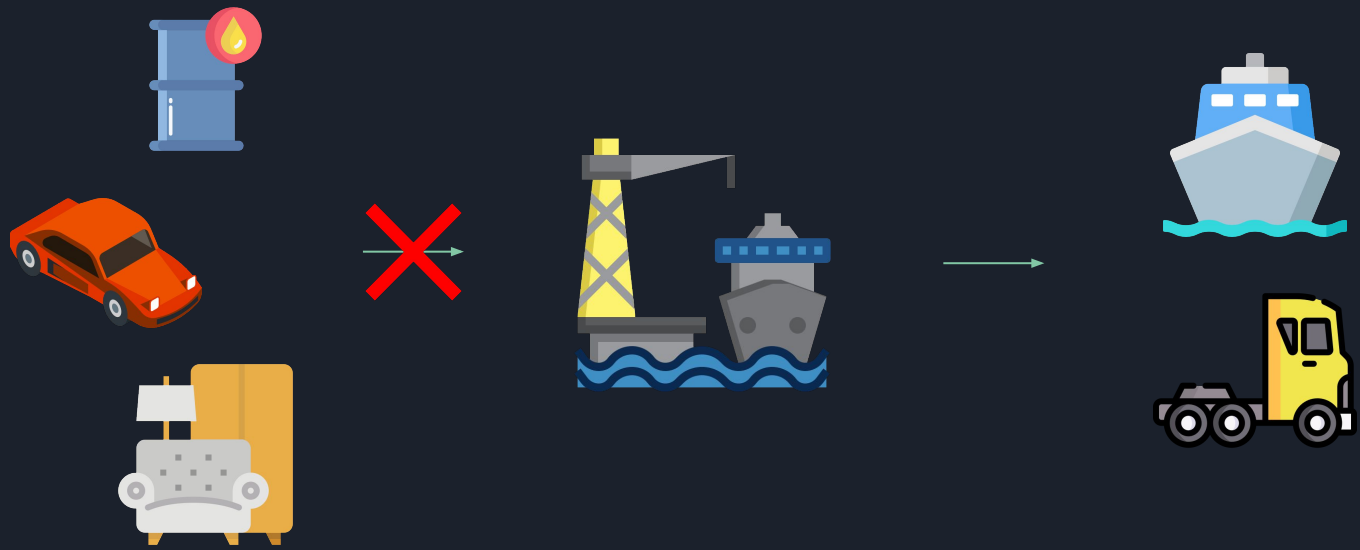
There is a problem of **testing environments.**

There is a problem of **production environments.**

History of global transportation system 🚢

From: https://science.howstuffworks.com/environmental/earth/geophysics/map.htm

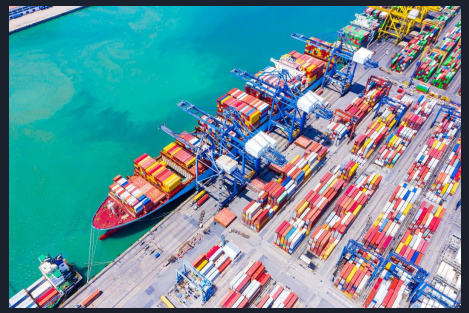
History of global transportation system



History of global transportation system 🚢

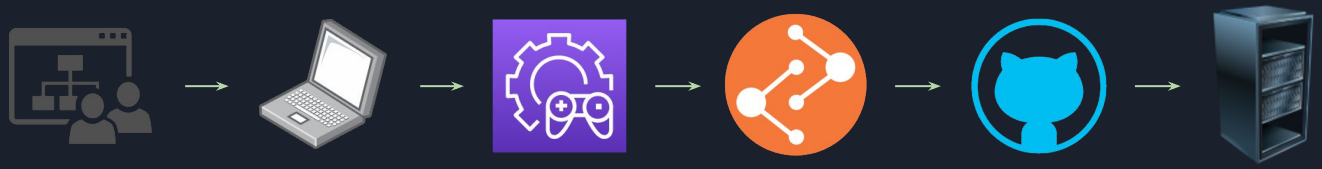
In 1795, Englishman Benjamin Outram invented the first ”container”

Malcolm McLean
1950s standarde size



Today

Problem

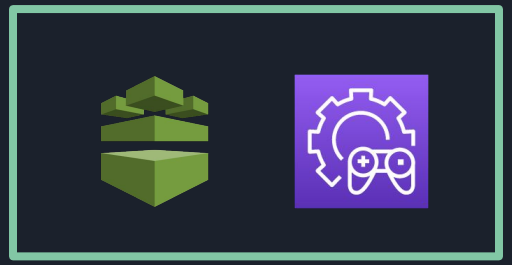


There is a problem of **development environments.**

There is a problem of **testing environments.**

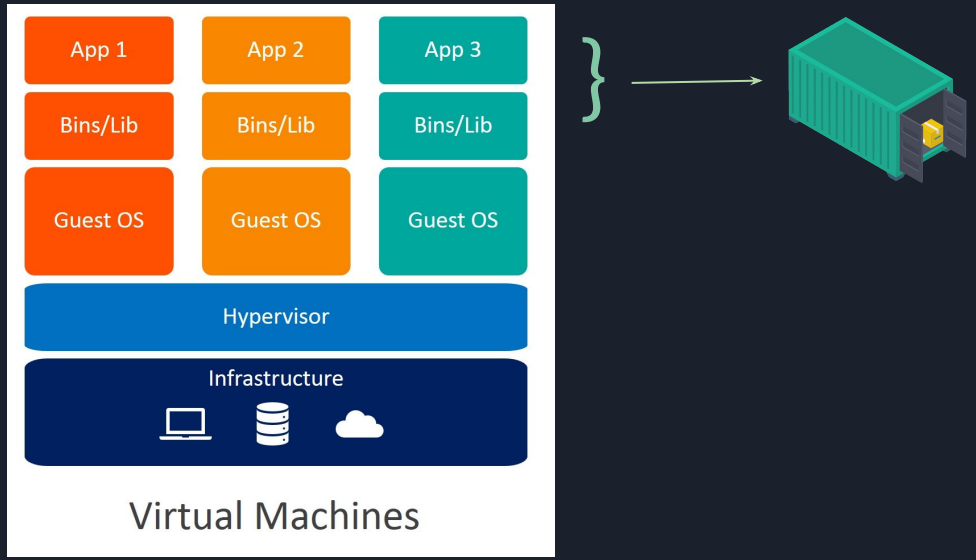
There is a problem of **production environments.**

Problem: Solution?

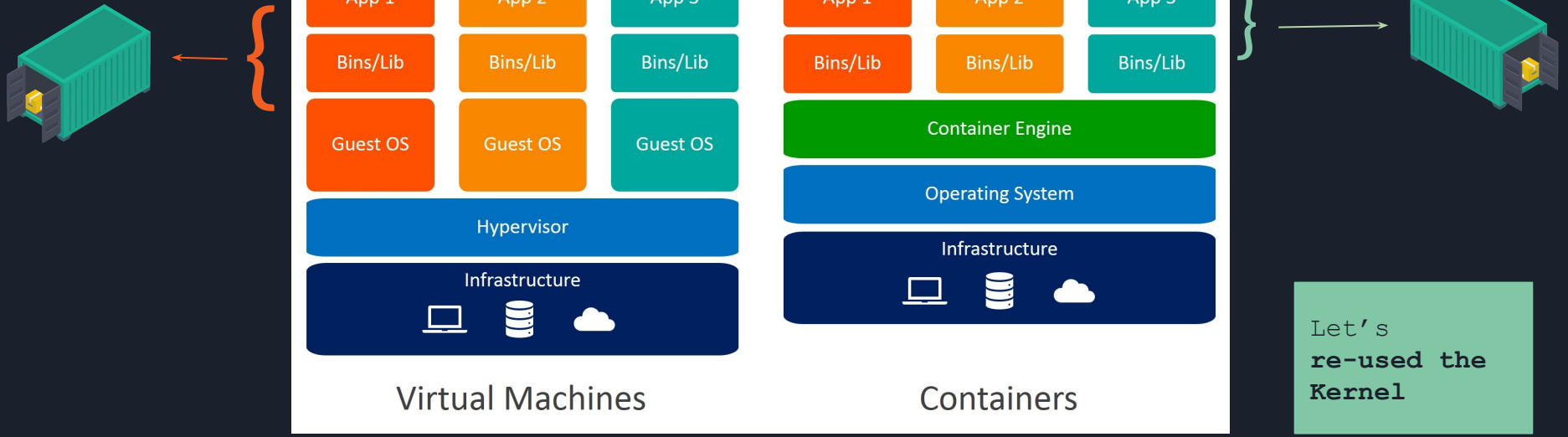


Virtual Machines!

Docker as a game changer 🚀

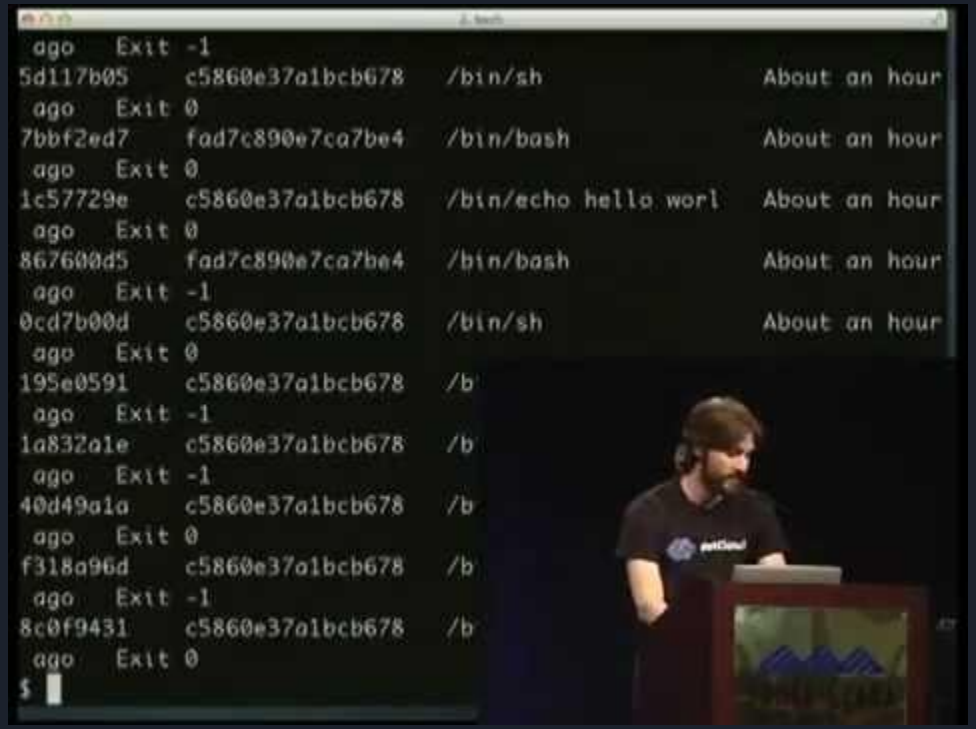


Docker as a game changer 🚀



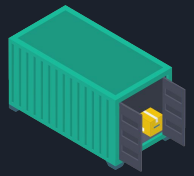
Let's **re-used the Kernel**

Docker as a game changer 🚀

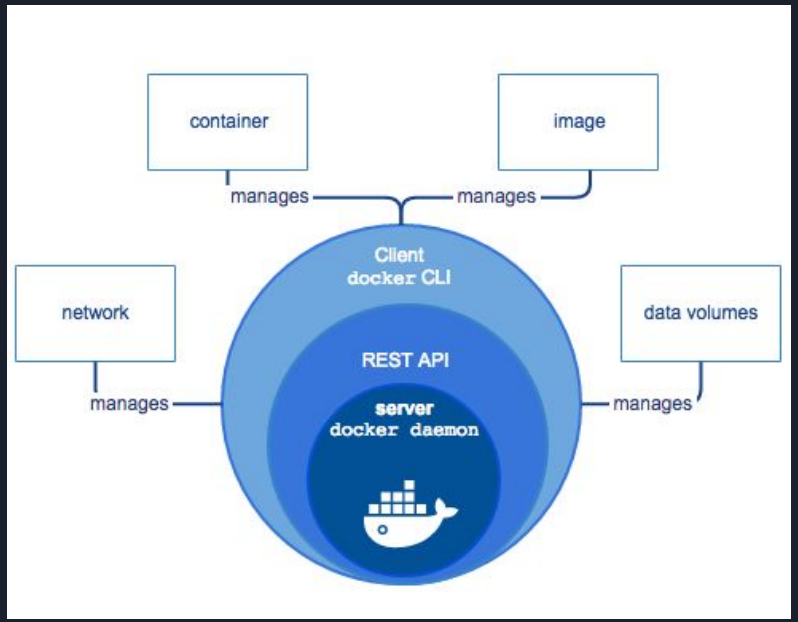


From: https://www.youtube.com/watch?v=wW9CAH9nSLs&t=1s

Docker as a game changer 🚀



Containers:
- Self contained units of software.
- Delivered on "any" OS machine.
- Isolated at the process level.
- Unique file system and network.



Docker:
- Docker is an open platform for developing, shipping, and running applications.

From: https://www.codementor.io/blog/docker-technology-5x1kilcbow

Play with containers 🎂

```
docker run hello-world
```

```
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:62af9efd515a25f84961b70f973a798d2eca956b1b2b026d0a4a63a3b0b6a3f2
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.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

Play with containers 🎂

Run a new container.

```
docker run ubuntu:18.04
```

Nothing happens?

Its process (*bash*) stops immediately.

Interactive:
Send STDIN commands (*-i*) and simulate TTY (*-t*).

```
docker run -it ubuntu:18.04
```

Stop bash process:
exit

```
docker container run -it ubuntu:18.04
```

Play with containers 🎂

List containers

`docker ps`

There is nothing?

Let's look all!

All:
Show already stopped containers (*-a*)

`docker ps -a`

`docker container ps [ls / list]`

Play with containers 🎂

```
docker run -it --name <name> ubuntu:18.04
```

Starts a stopped container.

```
docker start -i <name>
```

```
docker container start -i <name>
```

Play with containers 🎂

Remove a container

```
docker rm <name>
```

Is running?

Force:
Stop it and remove it (*-f*).

```
docker rm -f <name>
```

```
docker container rm -f <name>
```

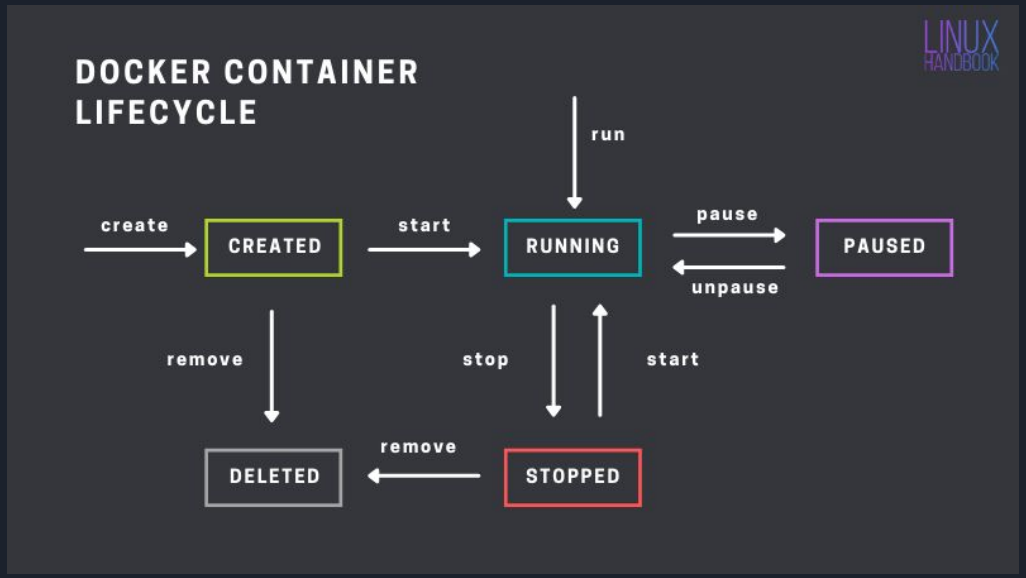
Play with containers 🎂

```
docker container --help
```

```
Manage containers

Commands:
  attach      Attach local standard input, output, and error streams to a running container
  commit      Create a new image from a container's changes
  cp          Copy files/folders between a container and the local filesystem
  create      Create a new container
  diff        Inspect changes to files or directories on a container's filesystem
  exec        Run a command in a running container
  export      Export a container's filesystem as a tar archive
  inspect     Display detailed information on one or more containers
  kill        Kill one or more running containers
  logs        Fetch the logs of a container
  ls          List containers
  pause       Pause all processes within one or more containers
  port        List port mappings or a specific mapping for the container
  prune       Remove all stopped containers
  rename      Rename a container
  restart     Restart one or more containers
  rm          Remove one or more containers
  run         Run a command in a new container
  start       Start one or more stopped containers
  stats       Display a live stream of container(s) resource usage statistics
  stop        Stop one or more running containers
  top         Display the running processes of a container
  unpause     Unpause all processes within one or more containers
  update      Update configuration of one or more containers
  wait        Block until one or more containers stop, then print their exit codes
```

Play with containers 🎂



https://linuxhandbook.com/container-lifecycle-docker-commands/

This is all about **processes.**

The container lifecycle is tied to its **main process.**

Play with containers 🎂

```
docker run --name example -d ubuntu:18.04 sleep infinity
```

Detached the container (background).

Override **command.**

Executes new process to a container.

```
docker container exec -it /bin/bash
```

Look the processes:
ps -aux

The container lifecycle is tied to its **main process.**

PID: 1
On the container

Look the processes from outside:
ps -aux | grep sleep

Kill the process:
sudo kill -9 PID

Images 📜 and Registries 💾

Where does the containers come from?

Recipes for creating containers.
📜 ⇒ 🎂

OOP: Class ⇒ Instance

List images.

```
docker image ls
```

```
REPOSITORY    TAG       IMAGE ID       CREATED          SIZE
ubuntu        18.04     35b3f4f76a24   3 weeks ago      63.1MB
hello-world   latest    feb5d9fea6a5   12 months ago    13.3kB
```

Images 📜 and Registries 💾

```
docker image ls
```

```
REPOSITORY    TAG       IMAGE ID       CREATED          SIZE
ubuntu        18.04     35b3f4f76a24   3 weeks ago      63.1MB
hello-world   latest    feb5d9fea6a5   12 months ago    13.3kB
```

Repository:
Project

Tag:
Version

Images 📜 and Registries 💾

Go to hub.docker.com/

Search any image that you want!

Registries!

The Registry is a stateless, highly scalable server side application that stores and lets you distribute Docker images

Like GitHub or GitLab.

- Docker Hub
- AWS ECR
- Azure Container Registry

Images 📜 and Registries 💾


```
docker pull osrf/ros:rolling-desktop-full
```

Play with it!

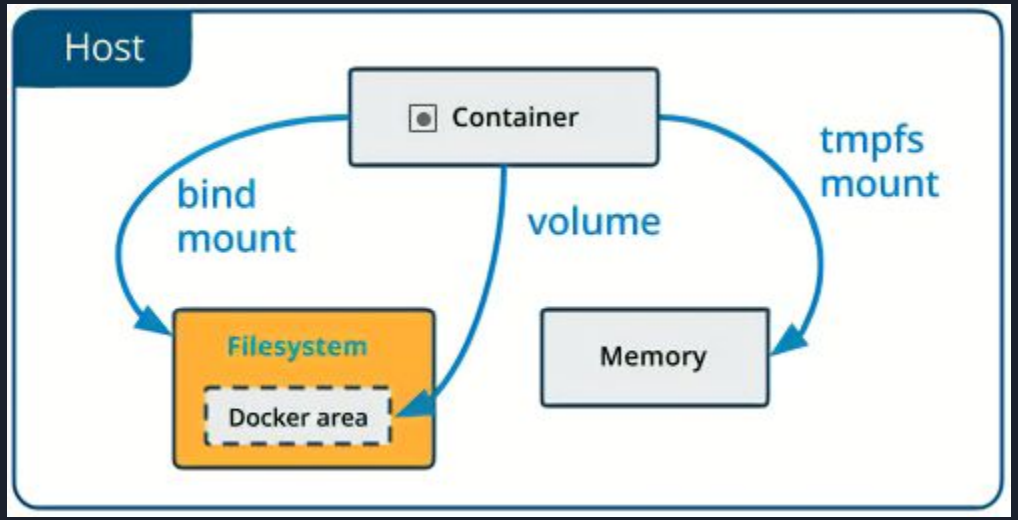
Pull an image.

(Default Docker hub)

Pull the images that you want and work on them.

Access the host ⇒ Volumes and networks 🔗

How can I persist data?



Access the host ⇒ Volumes and networks 🔗

Bind mounts

File or directory on the host machine is mounted into a container.

Create a bind mount.

Absolute path!

```
docker run -it -v /path:/c_path container
```

```
docker run -it
--mount type=bind,source=/path,target=/c_path
container
```

Access the host ⇒ Volumes and networks 🔗

Bind mounts

File or directory on the host machine is mounted into a container.

Create a bind mount.

Absolute path!

```
docker run -it -v /path:/c_path container
```

```
docker run -it
--mount type=bind,source=/path,target=/c_path
container
```

Access the host ⇒ Volumes and networks 🔗

Bind mounts

File or directory on the host machine is mounted into a container.

Create a bind mount.

Absolute path!

```
docker run -it -v /path:/c_path container
```

```
docker run -it
--mount type=bind,source=/path,target=/c_path
container
```

Access the host ⇒ Volumes and networks 🔗

Bind mounts

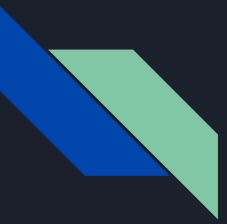
File or directory on the host machine is mounted into a container.

Create a bind mount.

Absolute path!

```
docker run -it -v /path:/c_path container
```

```
docker run -it
--mount type=bind,source=/path,target=/c_path
container
```

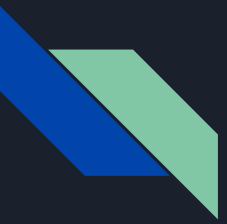


Access the host ⇒ Volumes and networks 🔗

Bind mounts

File or directory on the host machine is mounted into a container.

Files inside the container are created under container user (default root).



Access the host ⇒ Volumes and networks 🔗

Volumes

File or directory on the host machine, completely managed by Docker.

Create, delete, configure.

Set limits!

Access the host ⇒ Volumes and networks 🔗

Create a volume.

```
docker volume create data
```

```
docker volume ls
```

```
docker volume -v data:/c_path container
```

```
docker run
--mount source=data,target=/c_path
container
```

Access the host ⇒ Volumes and networks 🔗

Create a volume.

```
docker volume create data
```

```
docker volume ls
```

```
docker volume -v data:/c_path container
```

```
docker run
--mount source=data,target=/c_path
container
```

Access the host ⇒ Volumes and networks 🔗

Create a volume.

```
docker volume create data
```

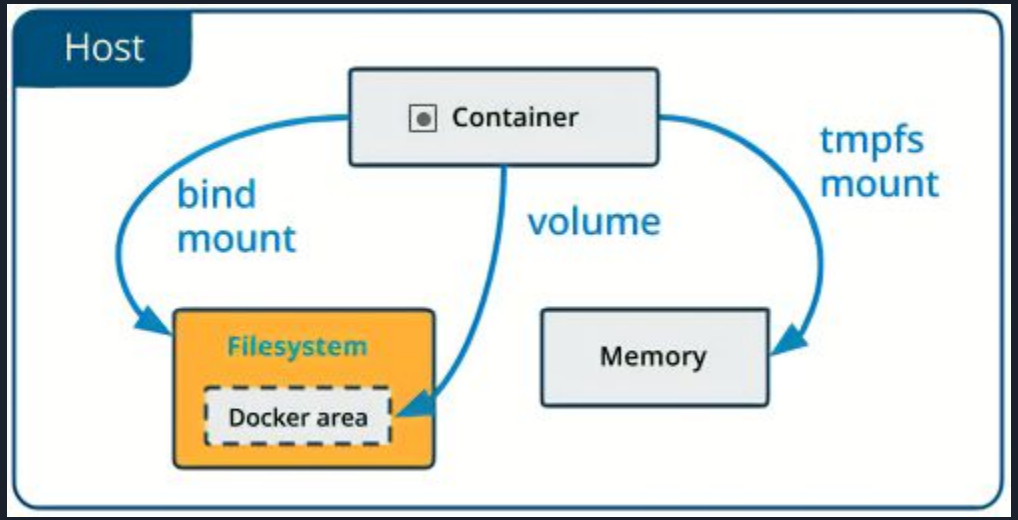
```
docker volume ls
```

```
docker volume -v data:/c_path container
```

```
docker run
--mount source=data,target=/c_path
container
```

Access the host ⇒ Volumes and networks 🔗

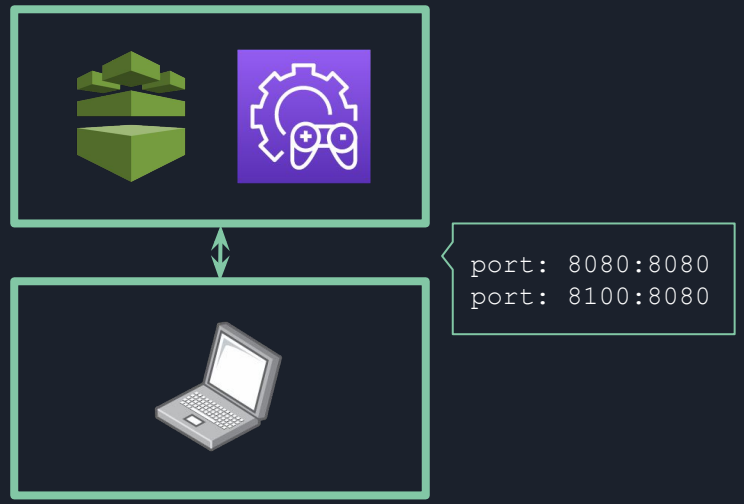
How can I persist data?



Access the host ⇒ Volumes and networks 🔗

How can use communication protocols

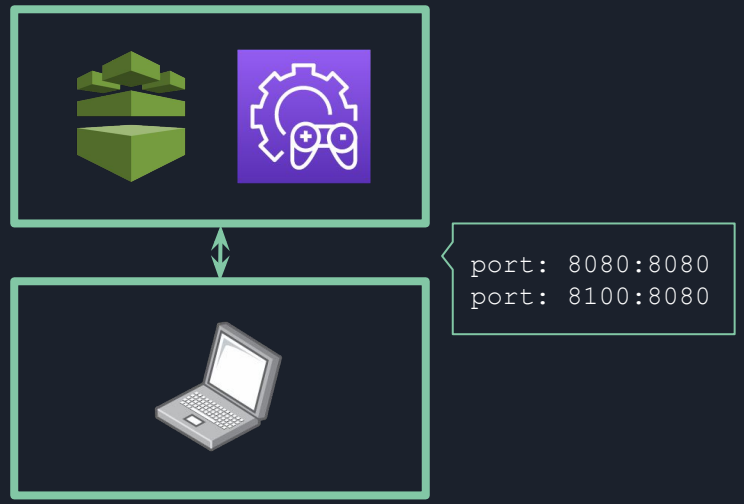
Networks



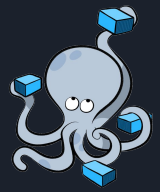
Access the host ⇒ Volumes and networks 🔗

How can use communication protocols

Networks



Let’s take a breath

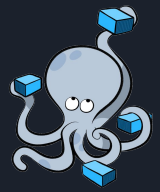


Let’s create a Image (Dockerfiles 📋) Git is that you?

How we can create our own image?

Dockerfiles!

Docker can build images automatically by reading the instructions from a Dockerfile.



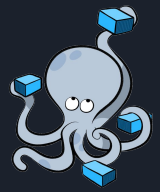
Let’s create a Image (Dockerfiles 📋) Git is that you?

How we can create our own image?

Dockerfiles!

Docker can build **image**s automatically by reading the **instruction**s from a Docker**file**.

Create any environment that you want.



Let’s create a Image (Dockerfiles 📋) Git is that you?

INSTRUCTIONS

FROM - Initial base image

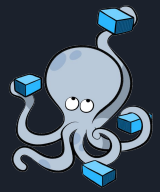
RUN - Execute a given command

ENV - Set an build-time environment

WORKDIR - Set the container working dir

USER - Set the container user

Dockerfile references
https://docs.docker.com/engine/reference/builder/#env



Let’s create a Image (Dockerfiles 📋) Git is that you?

EXERCISE

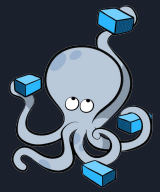
Let’s complete the simulation.Dockerfile

Docker image layers:
Think as a commit

Docker image layers:

- Only contain changes
- Compile once (if not changed)
- Data saved in a layer can’t be removed
- Many options to make it optimal!

- Docker cache
- Multi-Stage
- BuildKit
- Etc



Let's create a Image (Dockerfiles 📋) Git is that you?

Build a Image

```
docker build -f <Dockerfile> -t gazebo_tb3 .
```

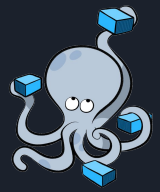
File (--file)
Dockerfile path

Tag (--tag)
Repository:tag

Context
Indicate Dockerfile where it is.

```
docker run -it gazebo_tb3 /bin/bash
```

Command
We can change it with: COMMAND



So many commands 😖! ⇒ Compose.

Define a yaml file with all the docker instruction.

docker-compose.yaml

docker-compose.override.yaml

Let’s take a look at it

Thanks