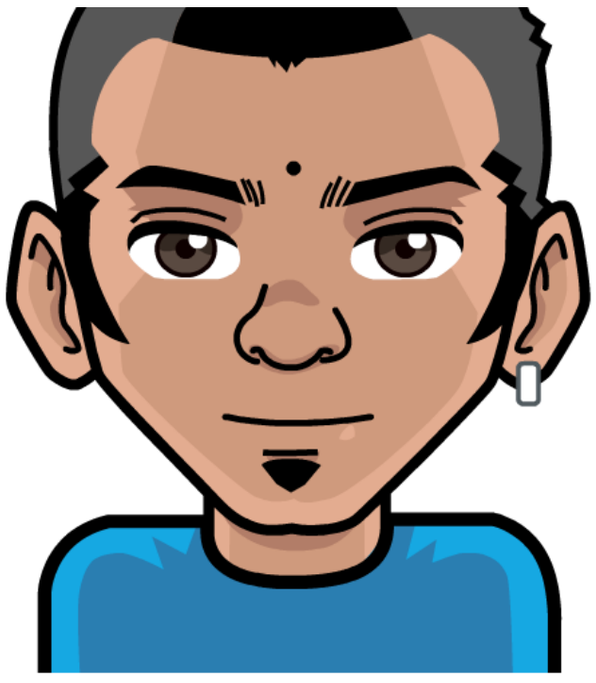
Raju Gandhi

DOCKER WORKSHOP



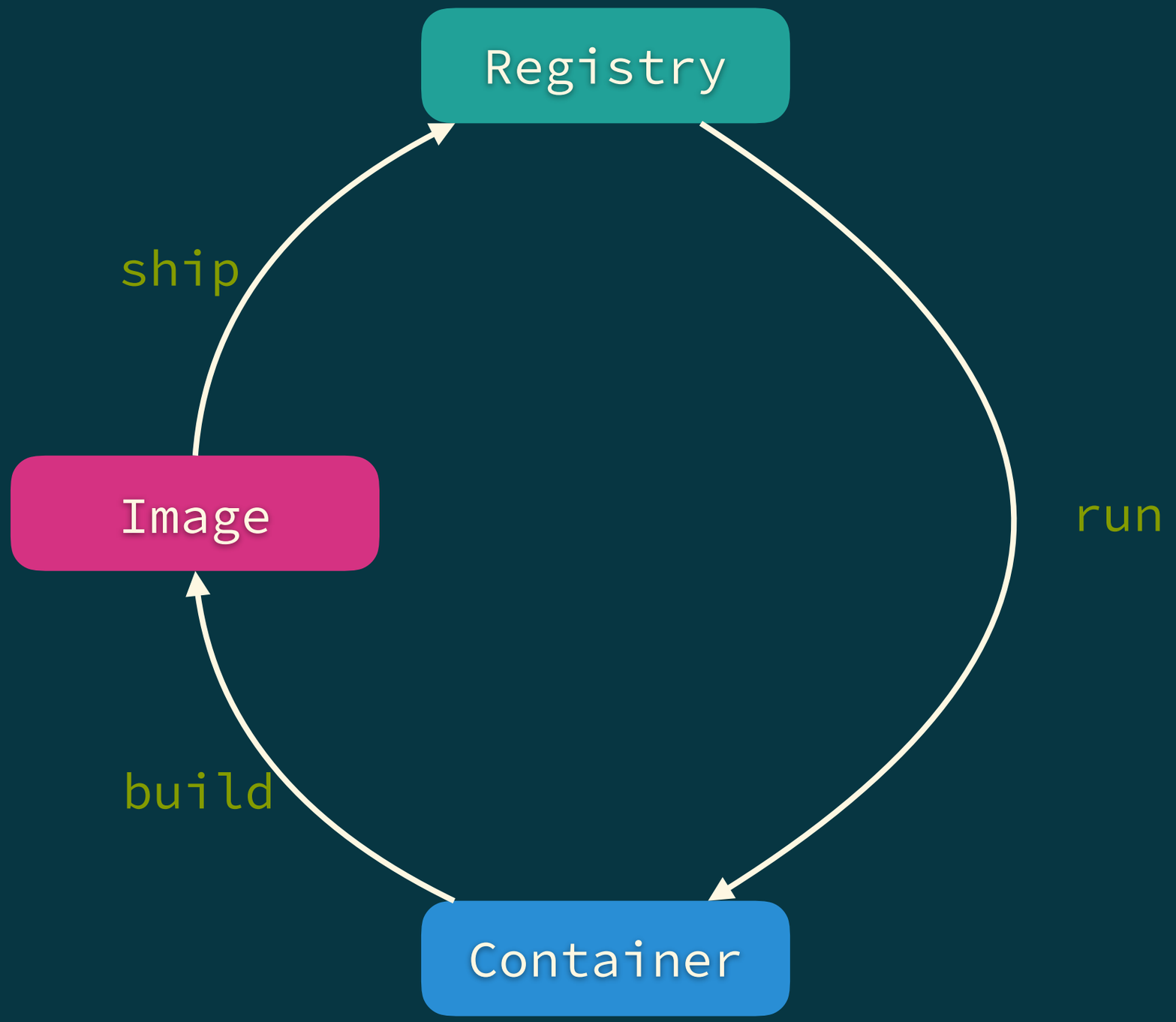
RAJU GANDHI

@LOOSELYTYPED

CTO - INTEGRALLIS SOFTWARE

WHY?

BUILD ONCE, RUN ANYWHERE


Registry
ship
Image
run
build
Container

WHY?

- Local application development and testing
- Team (and OSS) collaboration
- Ci/Cd

HELLO WORLD!

@EXERCISE

EXERCISE

- Run your first ubuntu:16.10 container and make it echo a message

INTERACTIVE CONTAINERS

@EXERCISE

EXERCISE

- Start an interactive `ubuntu:16.10` container
- Explore the following
  - Who are you logged in as?
  - What directory are you in?
  - What does the file system look like?
- Can you `ping www.google.com`?
- See if you can install a utility, like `iputils-ping`
  - Now can you ping google?

HINTS

- whoami;
- pwd; # print working directory
- ls -al; # listing files
- apt-get update && apt-get install -y iputils-ping; # installing items

THE RUNTIME

@EXERCISE

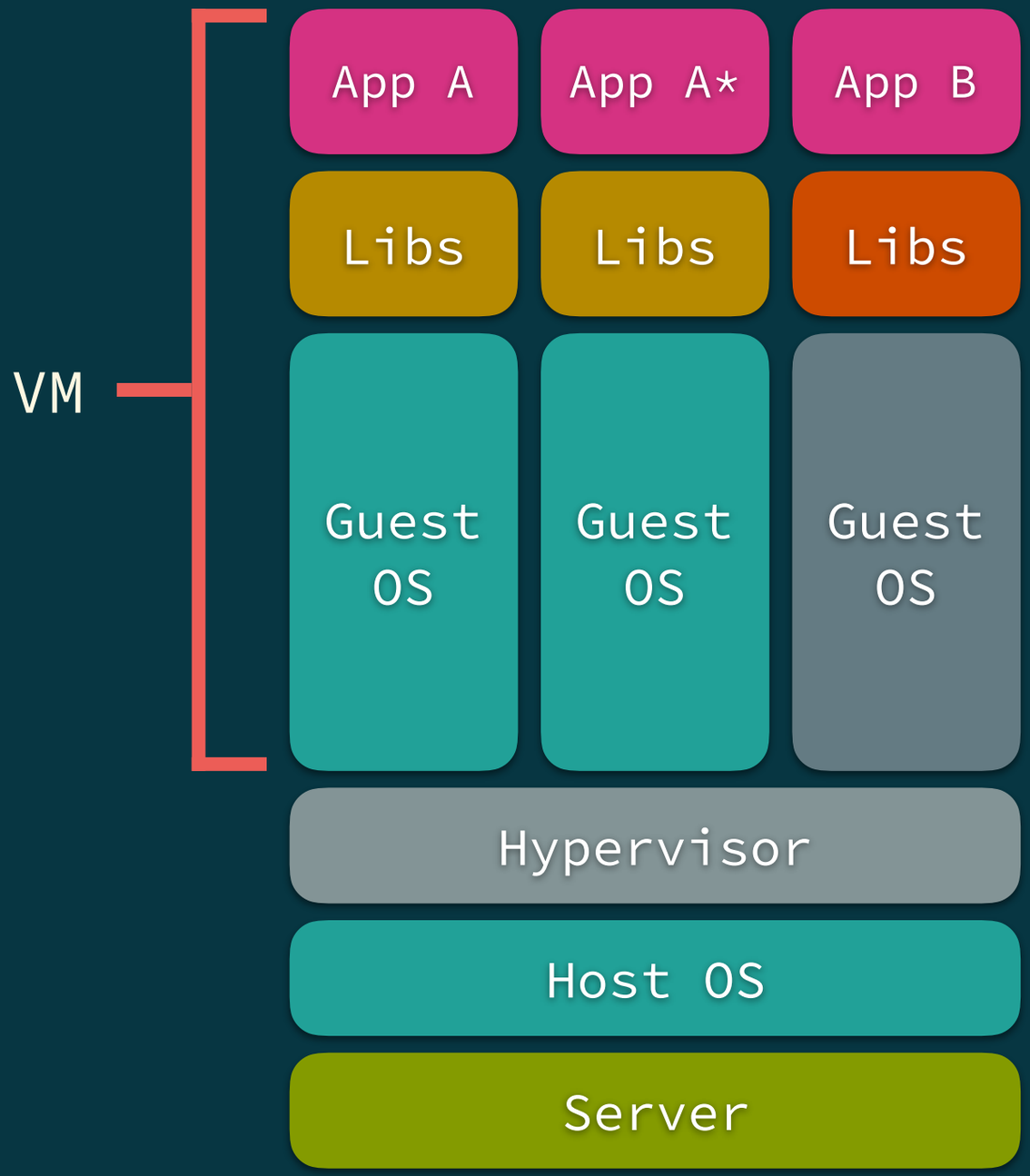
EXERCISE

- Figure out what containers are running on your machine
- Figure out what was run, but is no longer running
- Remove the containers that are no longer needed

HINTS

- `docker ps;`
- `docker help ps; # ask for help!`
- `docker help rm; # more help`

VM? CONTAINERS?


App A
App A*
App B
Libs
Libs
Libs
VM
Guest OS
Guest OS
Guest OS
Hypervisor
Host OS
Server

VM

App A | App A* | App B

Libs | Libs | Libs

Guest OS | Guest OS | Guest OS

Hypervisor

Host OS

Server

Container

App A | App A* | App A | App A* | App A | App A*

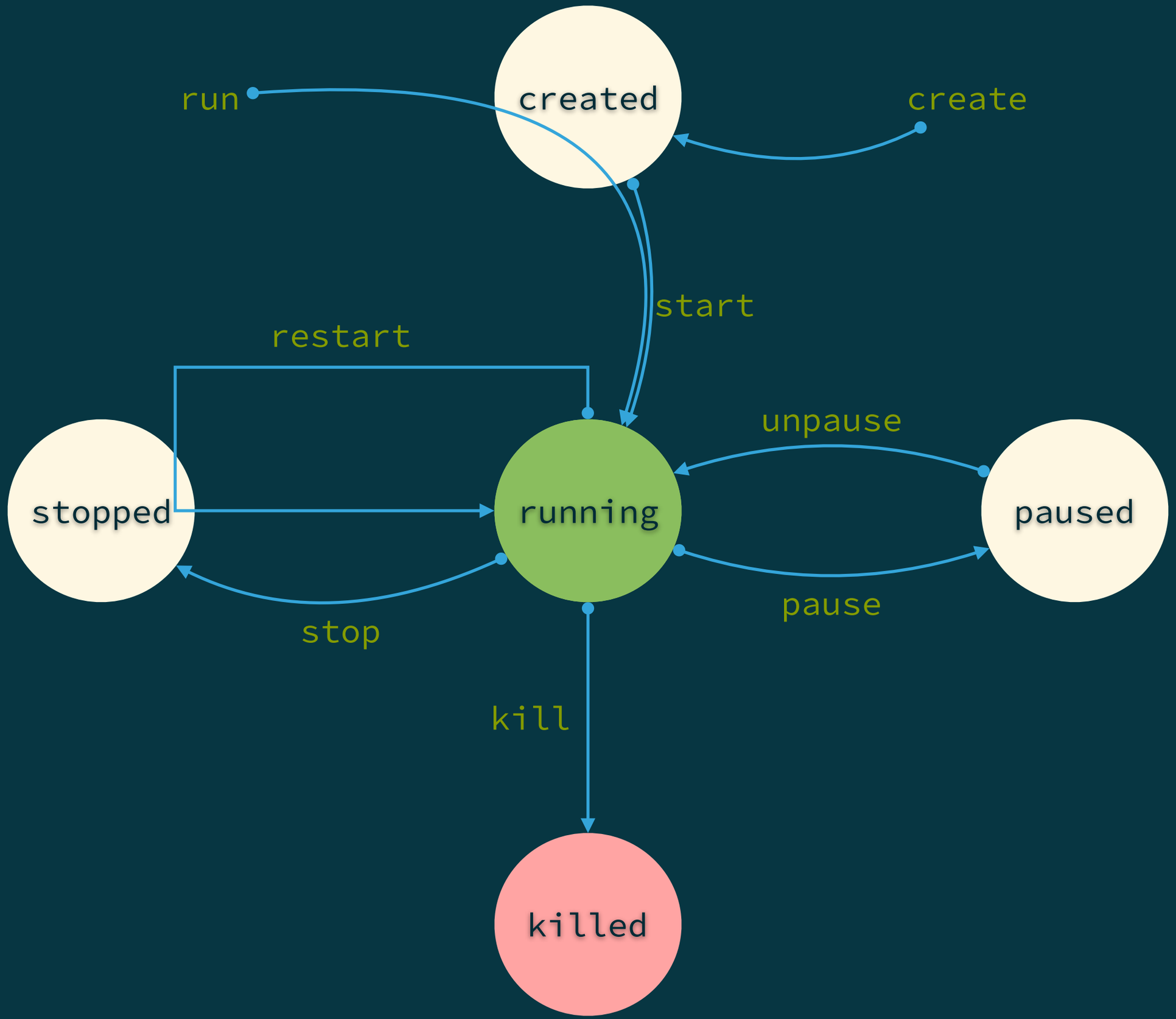
Libs | Libs

Docker Engine

Host OS

Server

DOCKER LIFECYCLE


run
created
create
start
restart
unpause
stopped
running
paused
stop
pause
kill
killed

@EXERCISE

EXERCISE

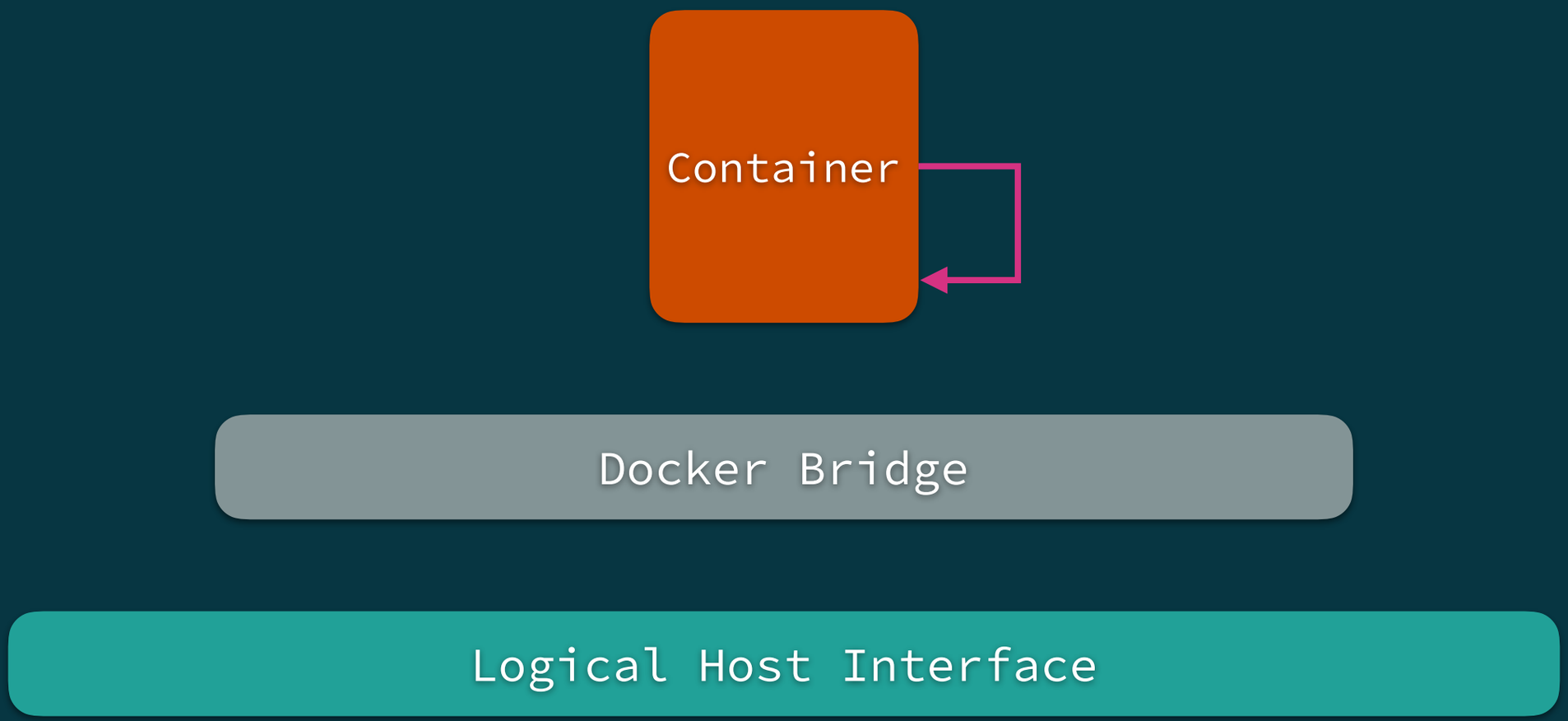
- Use create + start to run jenkins:2.32.3 (Be sure to name it!)
- Trace the logs
- Be sure to `stop` it, and then remove it

HINTS

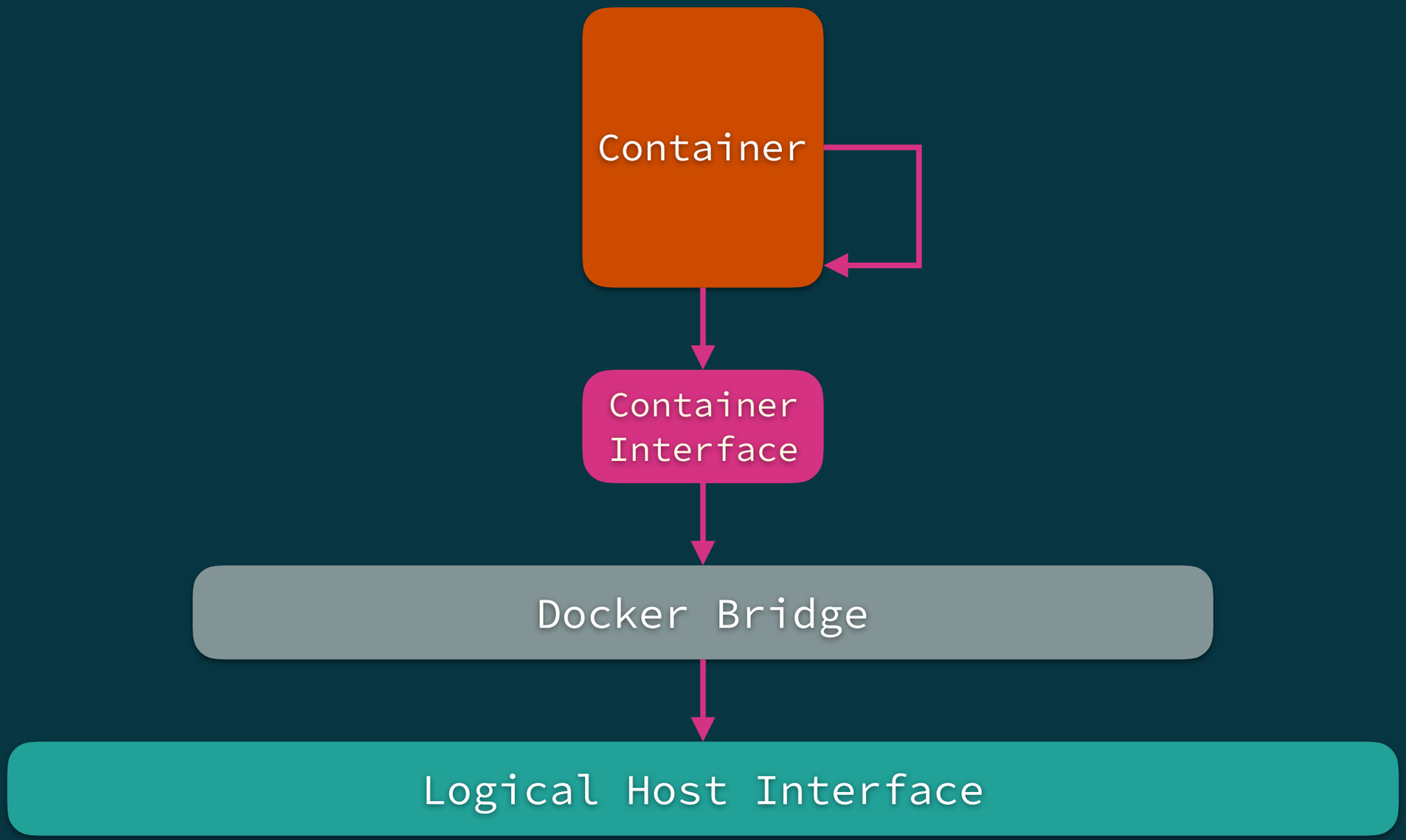
- create (use the --name or -n flag)
- start
- logs (use the --follow or -f flag)
- stop
- rm

NETWORK

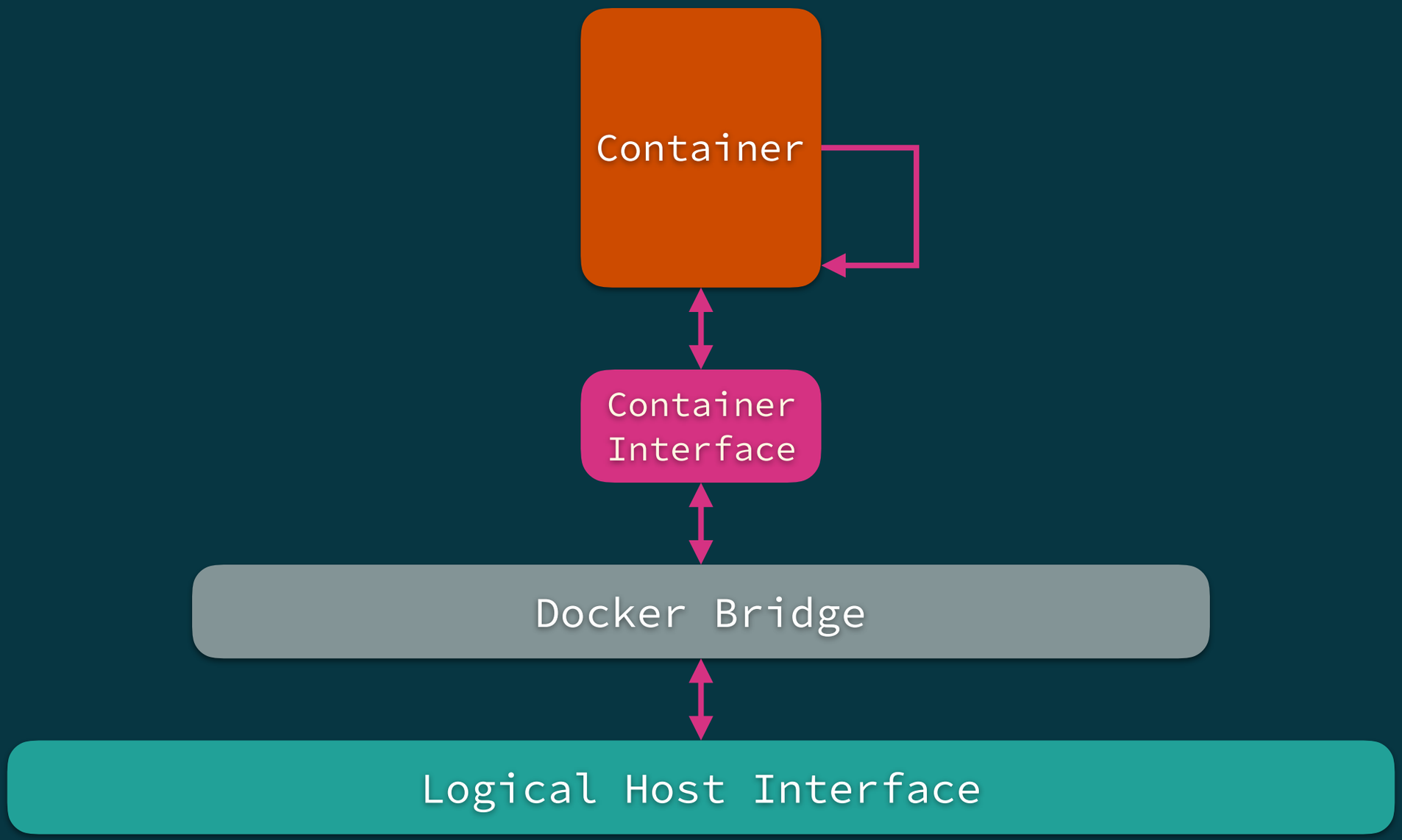
```
docker run -it --net none --rm alpine /bin/sh
```



```
docker run -it --rm alpine /bin/sh
```



```
docker run -it --rm -p 8080:8080 alpine /bin/sh
```



@EXERCISE

EXERCISE

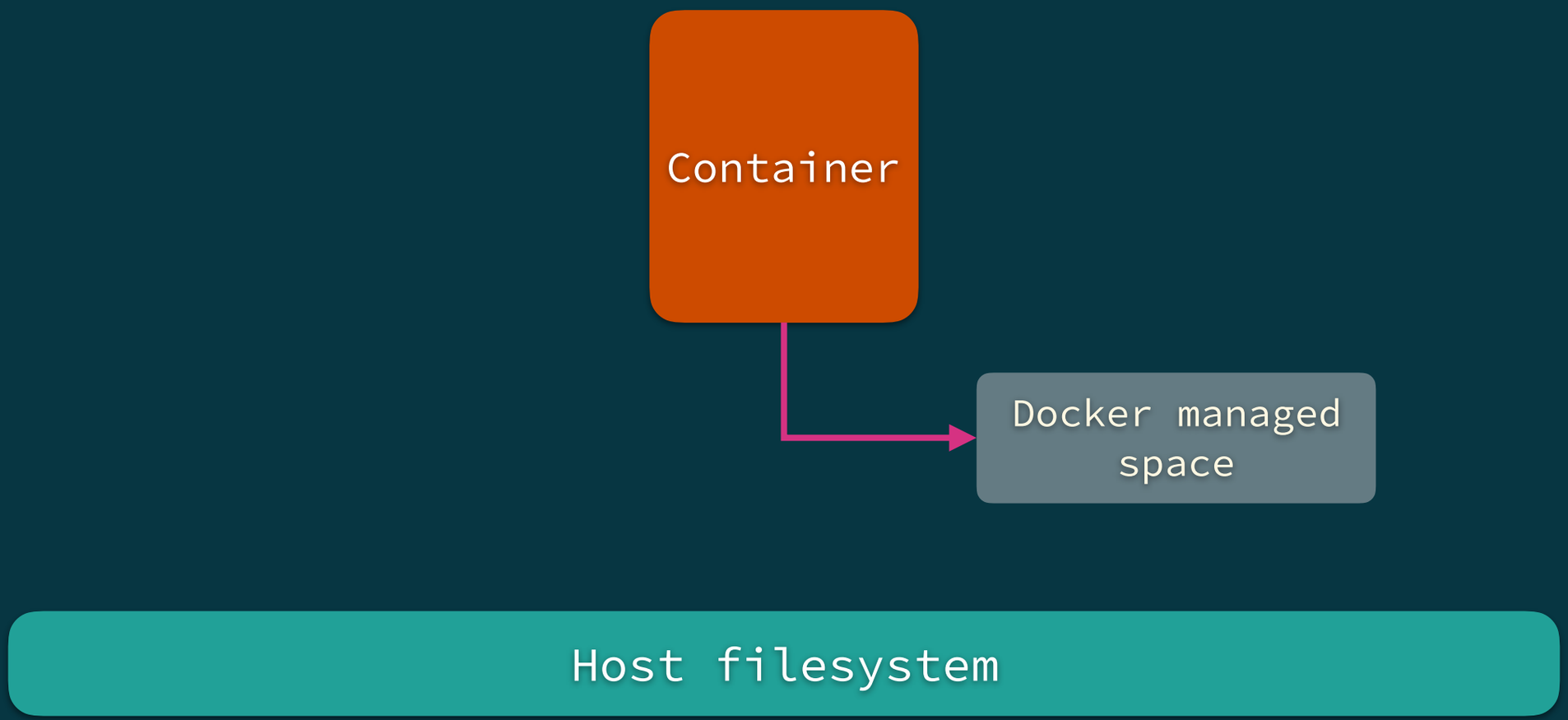
- Use run to start a jenkins:2.32.2 container
    - Name it!
    - Expose port 8080
- Stop and remove that container, start another one on another port, and see if you can get to it

HINTS

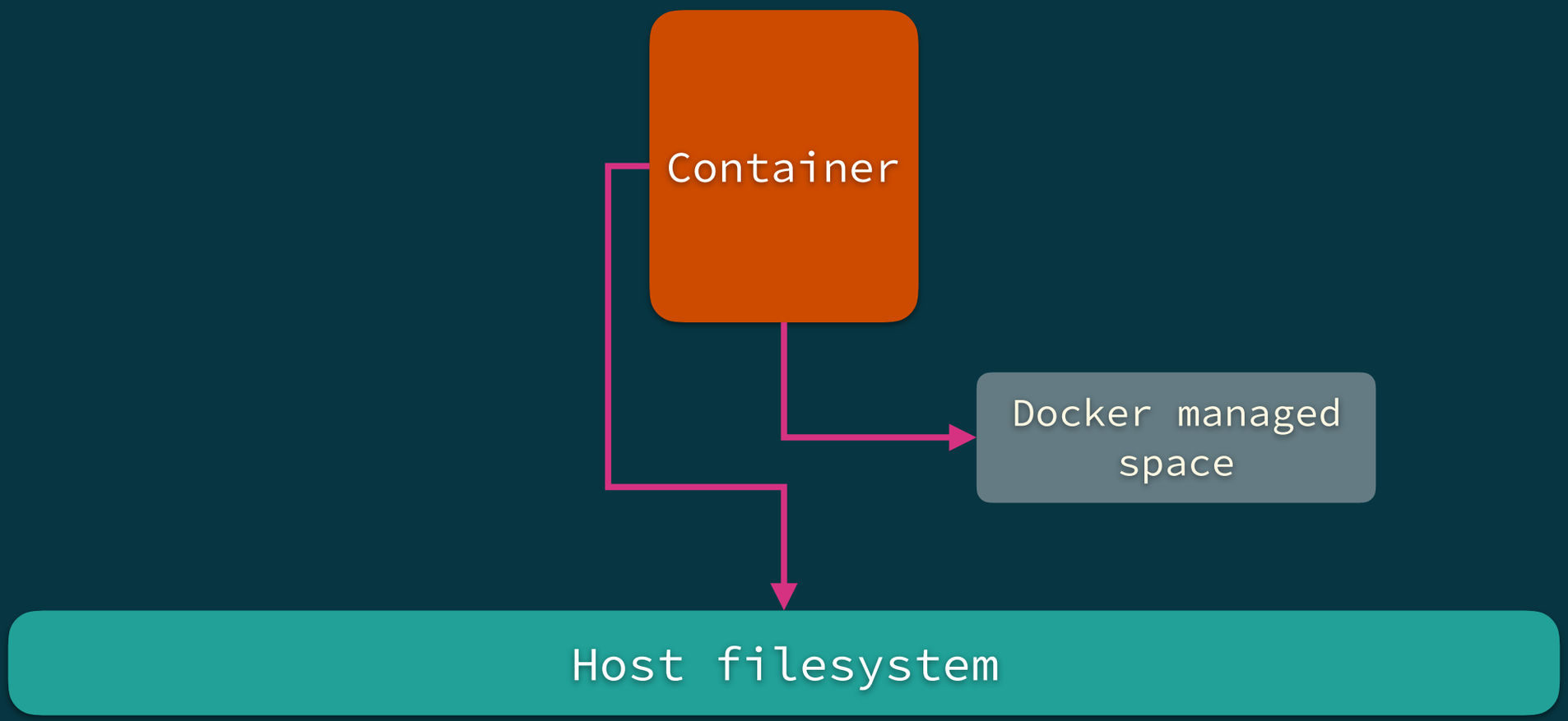
- run (use the --port or -p flag) # create a new container

VOLUMES

```
docker run -it --rm ubuntu /bin/bash
```



```
docker run -it -v /host/path:/tmp ubuntu /bin/bash
```



@EXERCISE

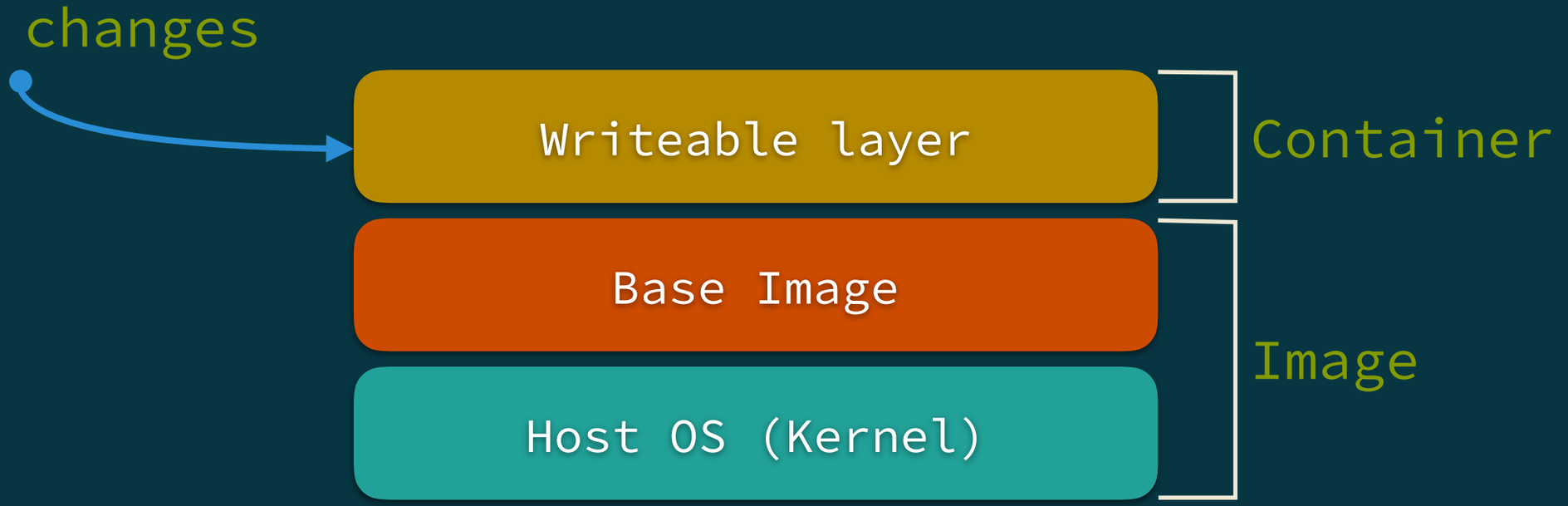
EXERCISE

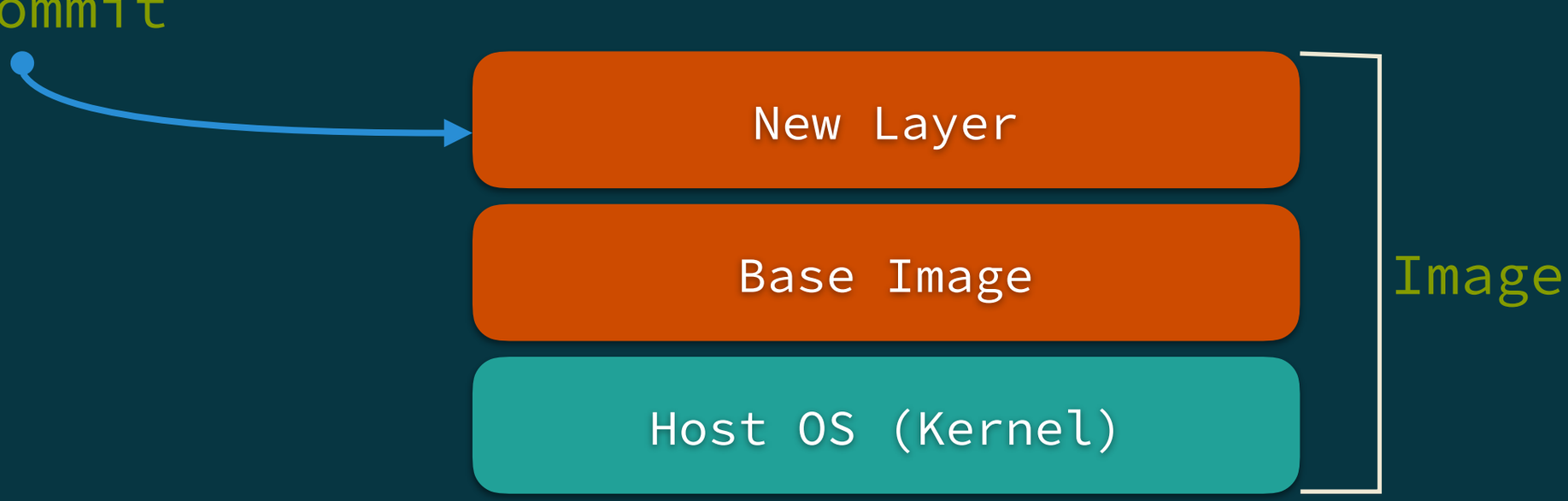
- Use run to start a jenkins:2.32.2 container
    - Name it!
    - Expose port 8080
    - Mount a volume from a scratch directory *into* the container
- Inspect the scratch directory and ensure you are seeing Jenkins logs

HINTS

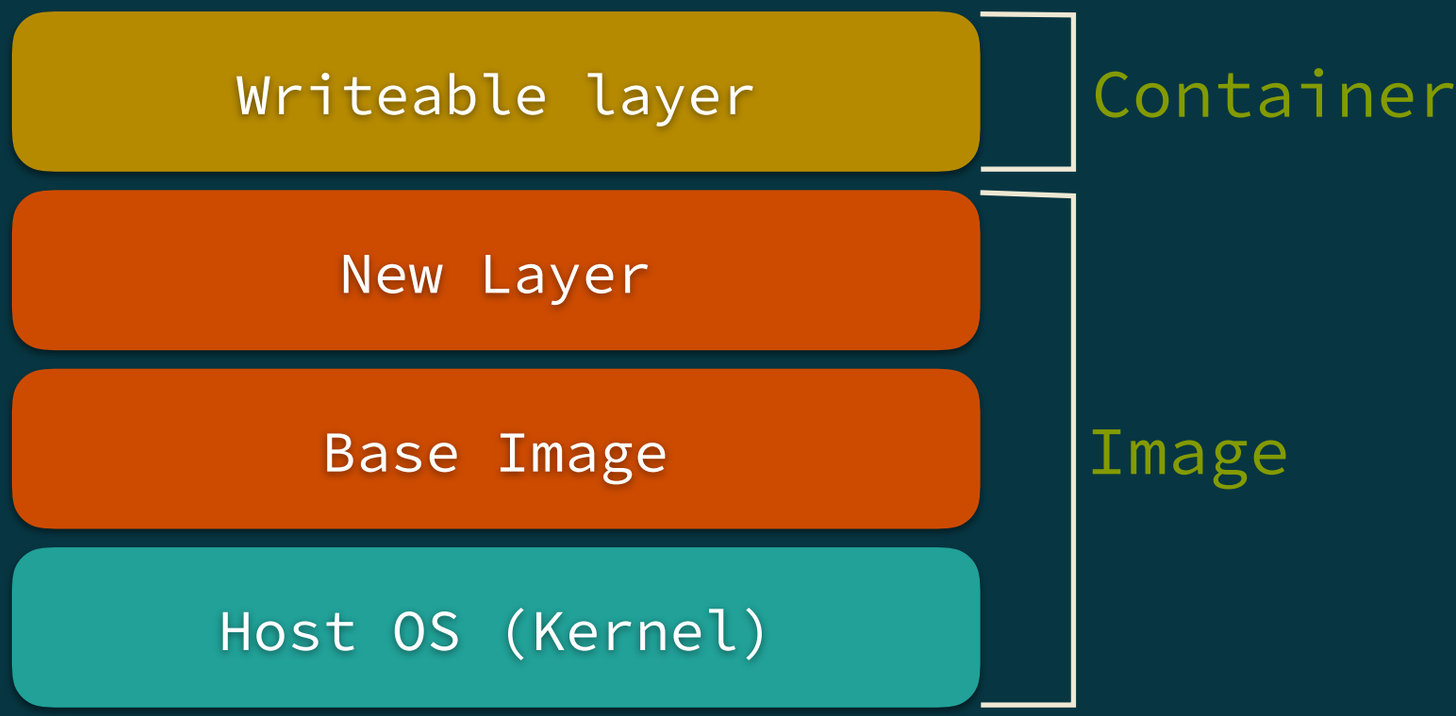
- run with the --port or -p flag AND the --volume or -v flag

WHAT IS A CONTAINER?


your changes
Writeable layer
Container
Base Image
Image
Host OS (Kernel)


commit
New Layer
Base Image
Host OS (Kernel)
Image

```
run <new-image>
```



@EXERCISE

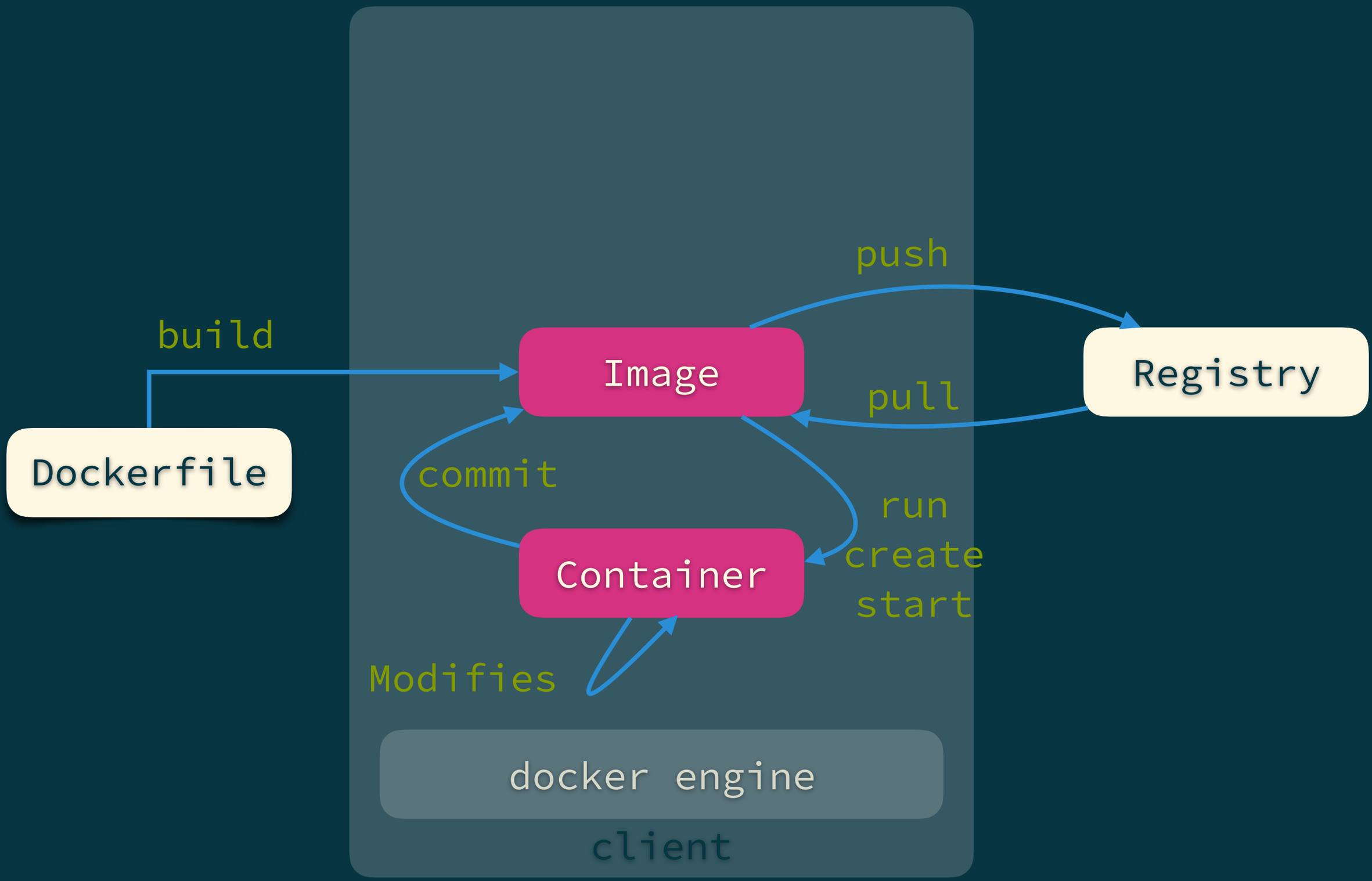
EXERCISE

- Use run to start a ubuntu:16.10 container (Be sure to name it!)
- touch a couple of new files
- Exit, then commit to create a new image named “ubuntu-addons”
- Create a new interactive container using the image “ubuntu-addons”
- See if your files are there

HINTS

- run (use the --name or -n flag) # create a new container
- echo “My OWN image” >> myFile.txt # create a new file with contents
- commit <contain-name> <new-image-name> # create a new image
- cat someFile # displays the contents of a file

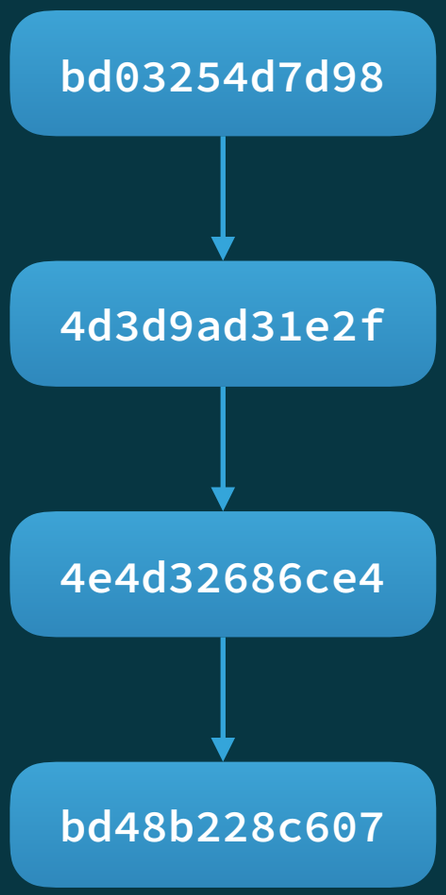
WORKFLOW


push
build
Image
Registry
pull
Dockerfile
commit
run
create
start
Container
Modifies
docker engine
client

DOCKERFILE

DOCKERFILES

- A set of instructions to build a Docker image
- Plain text, version controlled
- Provides insight into the image needs/capabilities/ intents

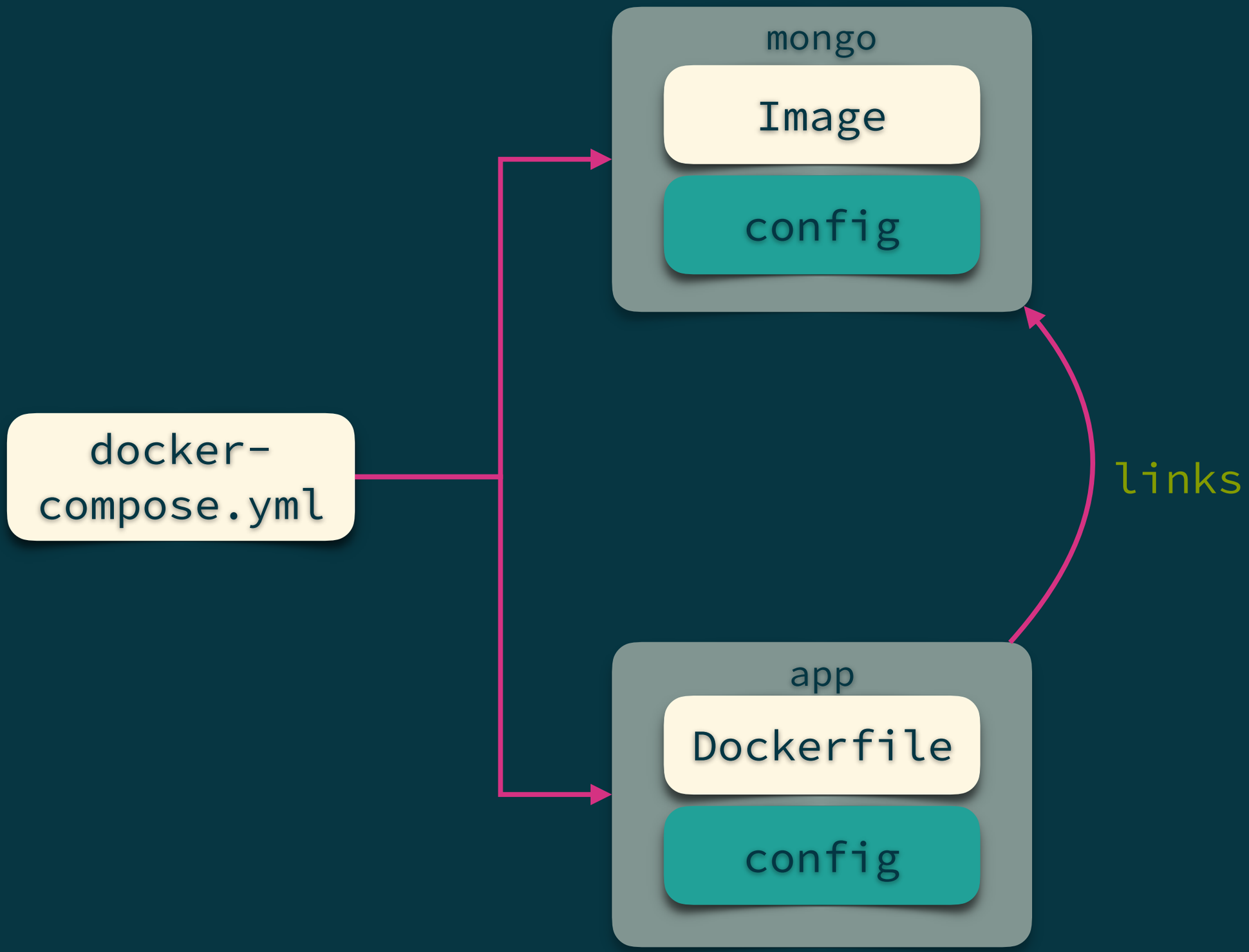



```
FROM openjdk:8u131-jre

RUN apt-get update && apt-get install -y netcat

COPY build/libs/app-fat.jar /var/app.jar

CMD ["java", "-jar", "/var/app.jar"]
```

DOCKER COMPOSE

DOCKER COMPOSE

- Define multi-container applications in a single file
- Supports scaling, healing
- Single host


mongo
Image
config
docker-compose.yml
links
app
Dockerfile
config

THANKS!!

RESOURCES

- Docker in action
- Docker in practice