

Argparse, Makefile, Docker



ENS
ÉCOLE NORMALE
SUPÉRIEURE
DE LYON

Ali Farnudi, Fall 2021

Argparse

Hands on - arguments

ArgumentParser

name

- **Optional** argument, identified by `'-'`
 - `parser.add_argument('-f', '--foo')`
- **Positional** arguments, anything without a prefix
 - `parser.add_argument('bar')`

ArgumentParser.add_argument()

- **name or flags** - foo or -f, --foo.
- **action** - Basic action taken when t encountered
- **help** - Brief description
- **type** - Type to be converted to
- **default** - The value produced if the argument is absent
- **choices** - Allowable values
- **required** - If option can be omitted (optionals only)
- **nargs** - Number of arguments consumed
- **metavar** - A name for the argument in usage messages.
- **const** - A constant value required by some action and nargs selections
- **dest** - Name of attribute returned by parse_args()

Hands on - argparse

Hands on - mutually exclusive group

Take the Fibonacci model

Add an argument to **print** all fibonacci numbers up to the provided input

Makefile

Project Maintenance

- A makefile is a **file** (script) containing:
 - The project **structure** (files, dependencies).
 - Instructions for **files creation**.
- The **make** command reads a makefile, understands the project structure and makes up the **Target(s)**.

filename: makefile

Target01: Dependencies

←tab→ rules

Target02: Dependencies

←tab→ rules

```
>cd 'dir' of the makefile  
>make
```

- Type **make**
- finds a file called **makefile**
- run the commands from the **first target**
- Check dependancies

Hands on - Makefile

**Hands on - Look at the sphinx Makefile
and execute the command**

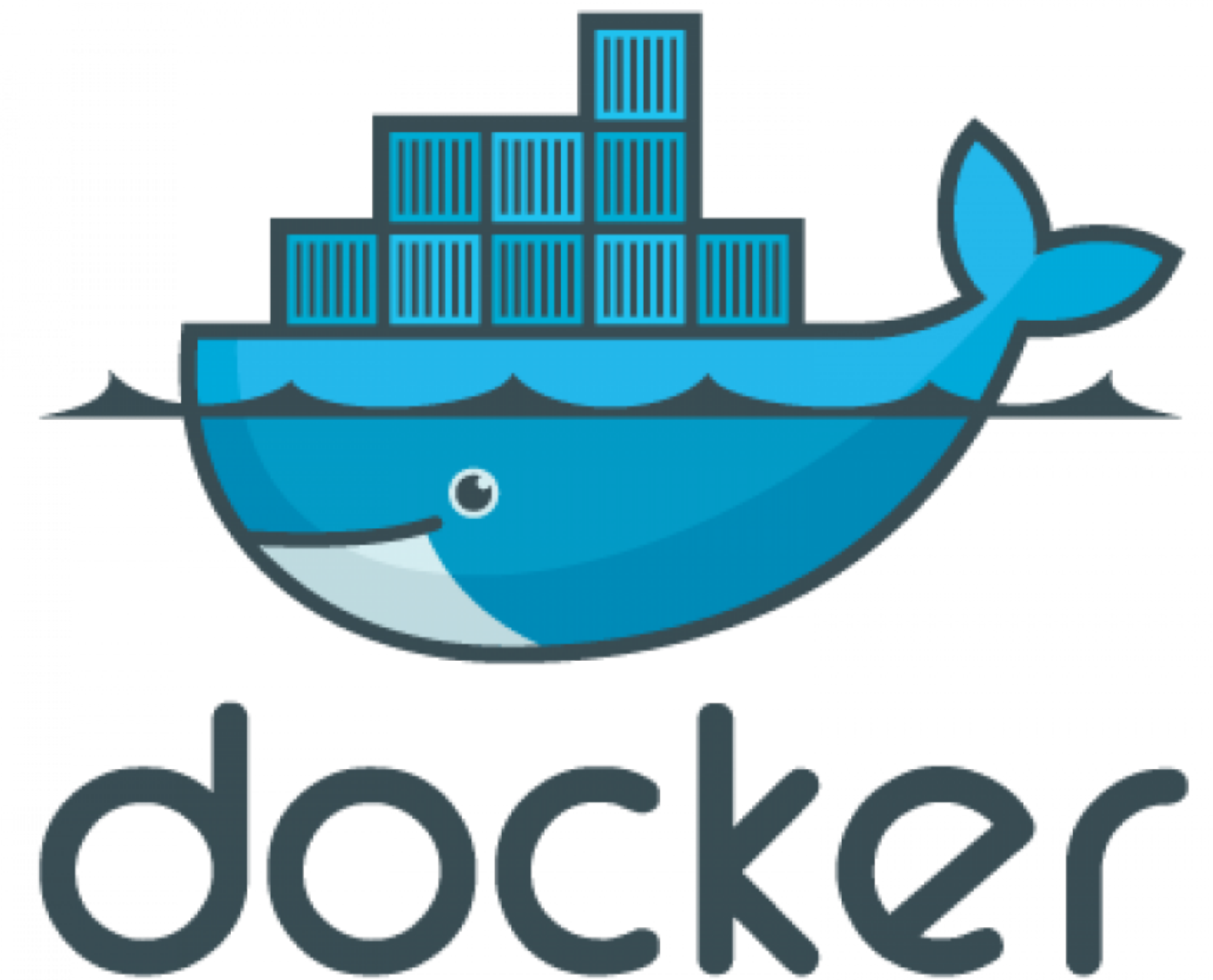
Hands on - Makefile - excersises

Docker

Containers

A platform for building, running, and shipping applications

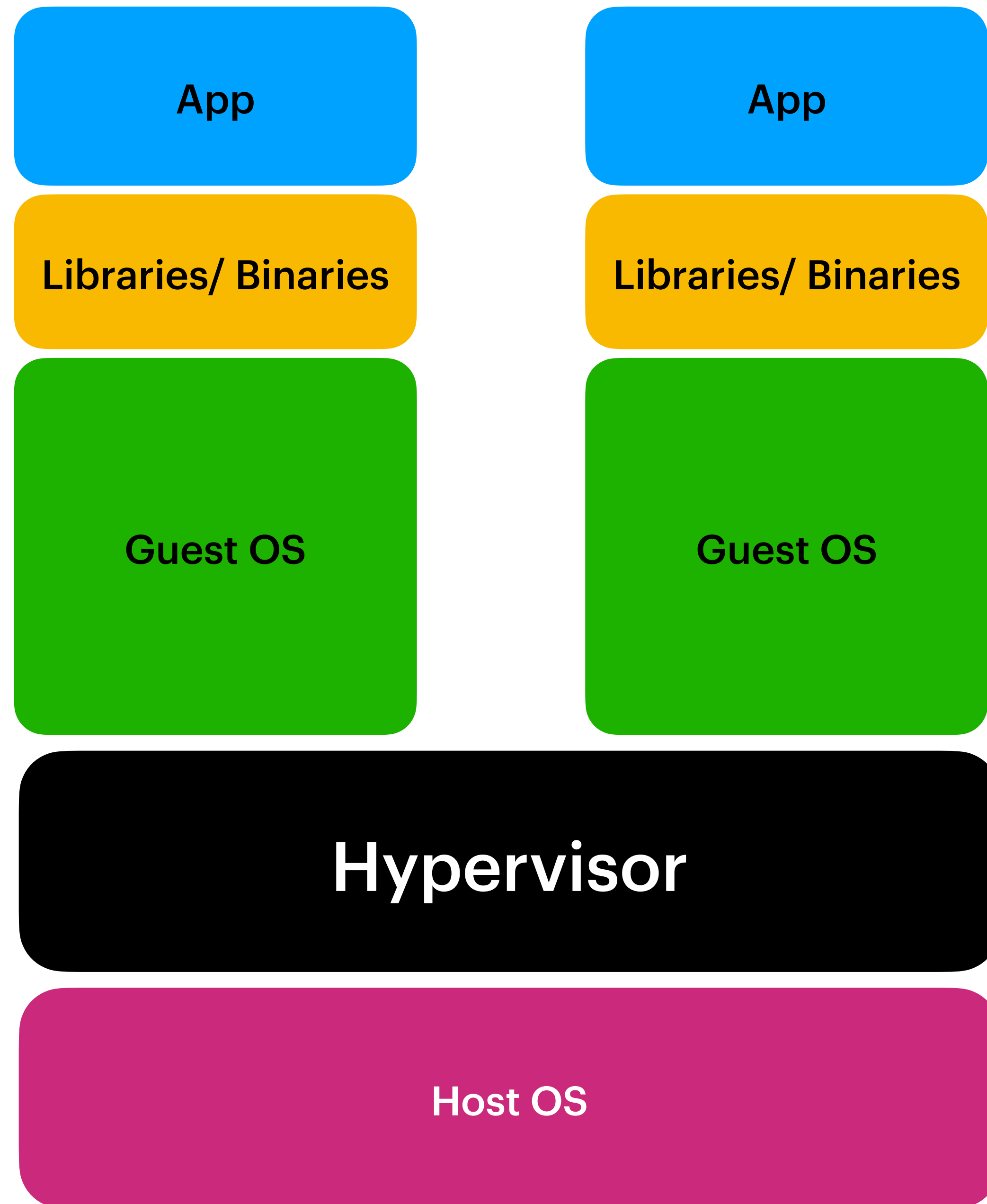
- Code works on my machine
- It doesn't work on another machine
- WHY?
 - Some files are missing
 - Software version mismatch
 - Different configuration settings
- What is the solution?



**Package and
run anywhere**

New user wants to use your code

- Read install instructions and requirements
- Spend half a day setting up the computer
- What if ..
 - Run multiple applications on same machine with different dependencies?



Virtual Machines

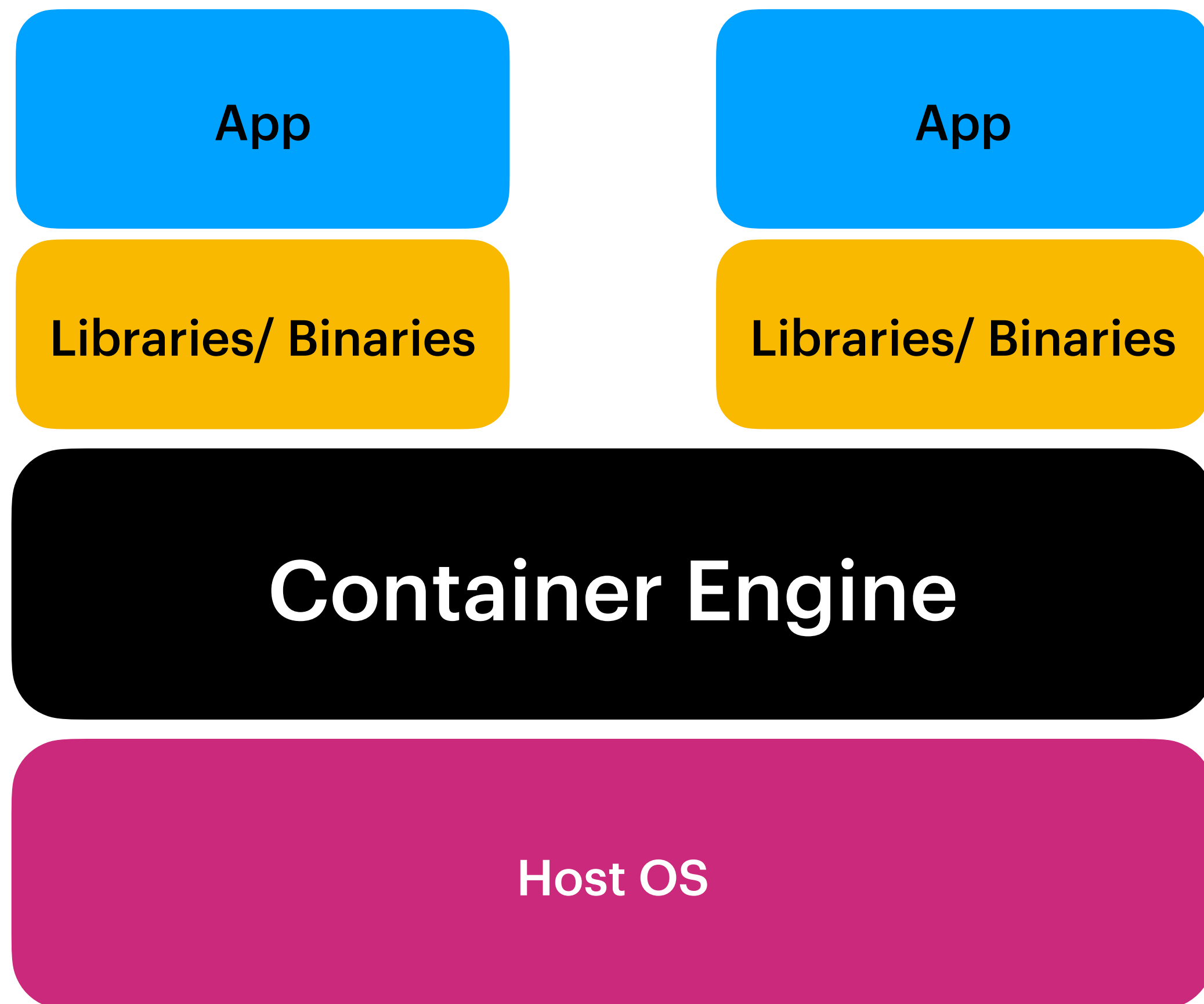
Run multiple apps in isolation

- Each VM needs a full copy of a OS
- Slow
- Resource intensive (limited number of VMs to run)

Containers

Run multiple apps in isolation

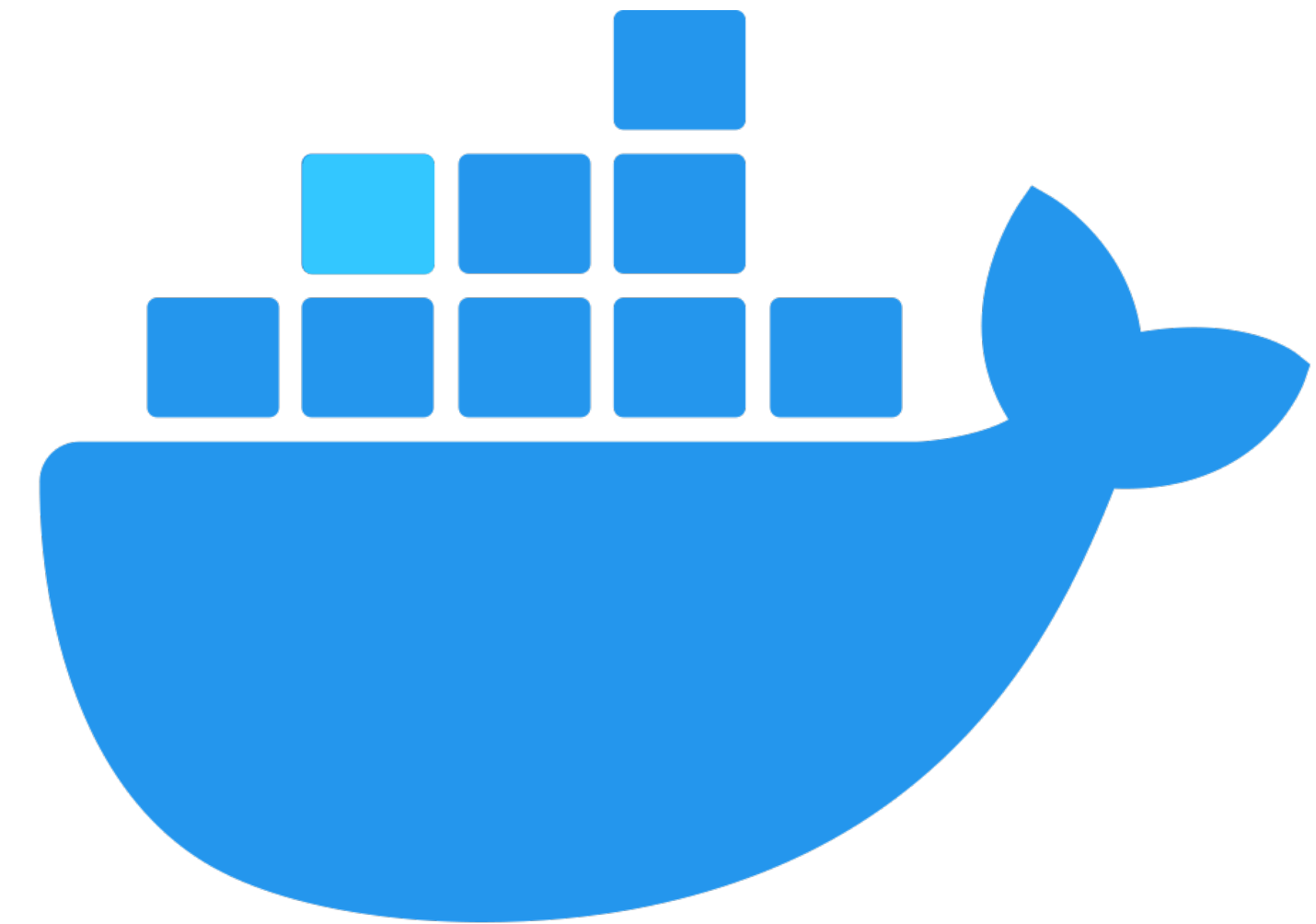
- Light weight
- Uses Host OS (Kernel)
 - Windows, linux, and Mac can run **linux apps**.
- Start quickly
- Needs less hardware resources (10-100s of containers)



Install Docker

- Get Docker
- Check system requirements (specially Windows users)
- Check if docker is running:
 - \$ docker version
 - Client and Server

- Docker Image
 - Application
 - Dependancies (files, libraries)
 - Dockerfile
 - A cut-down OS
- Push Image to DockerHub
- Pull on machine with Docker



No need to follow complicated
install instructions.

Just pull

Hands on - Docker - Hello-Docker

- Dockerfile
 - Create and OS
 - Copy Script
 - Run script
- Go to DockerHub
- Find a Bash Container
- Write the Dockerfile
- `$ docker build -t hello-docker .`
- `$ docker images / docker image ls`
- `$ docker run hello-docker`

Hands on - Docker - Python-Ubuntu

Hands on -docker -Interactive ubuntu

Hands on -docker -Setup ubuntu pdflatex

Useful docker commands

- `$ docker build ./path/to/dockerfile` #Build a docker image
- `$ docker build -t image-name ./path/to/dockerfile` #Build image and name it
- `$ docker images / docker image ls` #List of docker images
- `$ docker ps` # list of active containers
- `$ docker --help`
- `$ docker cp ContainerID:/path/to/file ./local/path` #copy file from container to the local machine
- `$ docker cp ./path/to/file ContainerID:/path` #copy file from local machine to container
- `$ docker stop ContainerID` #Stop container
- `$ docker rmi --force ImageID` #remove Image Note: Use “docker stop” to stop the container first

When updating a Dockerfile

- Stop any container running the image (check previous slide for `docker ps` and `docker`)
- Delete the image (`docker rmi --force`)
- Build a new image with the dockerfile (`docker build`)

- You can interactively run an image without the need to write a dockerfile with:
 - `$ docker run -it iamge-name`
- Many docker images are available at [dockerhub](https://hub.docker.com/).
- Note: Docker closes a container once the last command (CMD) in the Dockerfile is finished.
- If you wish to use the image to tryout stuff, comment out the execution of your programme from the Dockerfile. Depending on your project you may want to include the running of your programme as a part of the image with RUN.
 - Example:
 - CMD ["make"] vs RUN make

Write tests for the “Station” module

Find test coverage

Licensing

Who has used Open Source software?

- Softwares usually come with a license
- It will determine what a user can and cannot do with the software
- Some licenses are free, some you have to purchase, some might have different requirements.

- Licenses are written by lawyers and not by developers.
- But developers must know the basics.
- Facebook's ReactNative vs Apache

General License Types

- Permissive
 - Users have more freedom
- Copyleft
 - Makes sure the software stays on the same license

Permissive

- Fewer restrictions on what users can do with the software
- Allows companies to use the software without contributing back
- Apache, MIT, VSD, creative commons attribution
 - Sharing material within specific parameters.
 - Crediting authors for their work
 - Allows free leagle copying/distribution/sharing of content
 - Like what Samsung does with Android.
 - Most of Android in on Apache license but Samsung freely modifies it and does not release the changes. And sell it

Copyleft

Want to keep it on the license

- If you use it you have to give back to the community
- Prevents companies from using the software without contributing back.
- Can have potential licensing conflicts.
- How linux Kernel is setup underneath Android
- GPL, CC attribute share.., IBM, Mozilla
-

Which license should I choose?

- Depends on your project.
 - If you want to write a library so that anyone can take it and put it in their software and use it how they want, better off with permissive (Apache)

Dual licensing

- Usually it means the user can pick one license
 - Your code must be compatible with both licenses
- Occasionally projects require users to abide by both licenses.

Contributor License Agreement (CLA)

- Gives project leader the ability to relicense the codebase.
- Good: Allows projects to adapt as the licensing situation changes.
- Bad: Allows projects to be dramatically shifted away from the contributors wishes.
- Bad/Good: Companies can take the code and relicense and sell it to other companies without contributing (Canonical, Eclipse, Cyanogen, etc)