

# Containers for Beginners

## Intro to Docker

Francesco Pontiggia

Senior Research Computing Facilitator

Faculty of Arts & Sciences Research Computing (FASRC)

Harvard University, Faculty of Arts & Sciences

April 16<sup>th</sup>, 2021

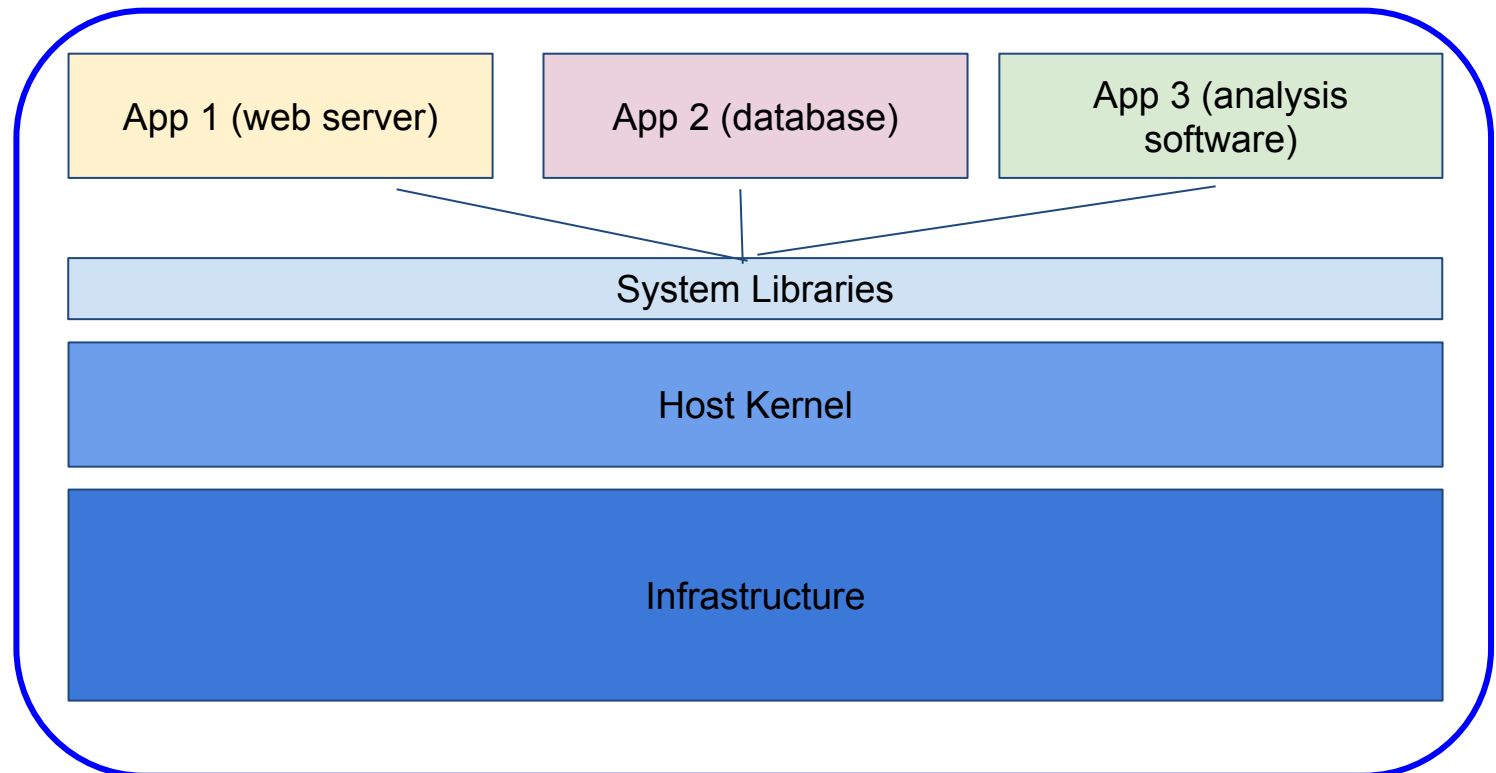
# Objectives

- What are containers? - Which problems are we trying to solve
- What is Docker and how it can help us
- A Simple Docker application
  - how to run a container
  - how to build a container
  - inspecting images
- Docker registry: How to publish containers
- Building applications with multiple components: docker-compose.

## What problems are we trying to solve?

Developing, Building, Deploying and Maintaining complex applications involves facing a number of issues including :

- managing a complex software stack
- compatibility between applications
- portability
- reproducibility



## What problems are we trying to solve?

- **Complex software stack :**

Building applications often requires a complex set of dependencies, specific versions of libraries, not always easy to manage on all operating systems.

- **Compatibility :**

Different applications might have conflicting requirements so setting up a system that can host multiple applications is not always easy.

- **Portability :**

Replicating a software setup on a different host, or between a development and production environment requires the environments to match perfectly, which is often hard to achieve.

- **Reproducibility :**

Applications behavior might be affected in unexpected way by changes to the host environment.

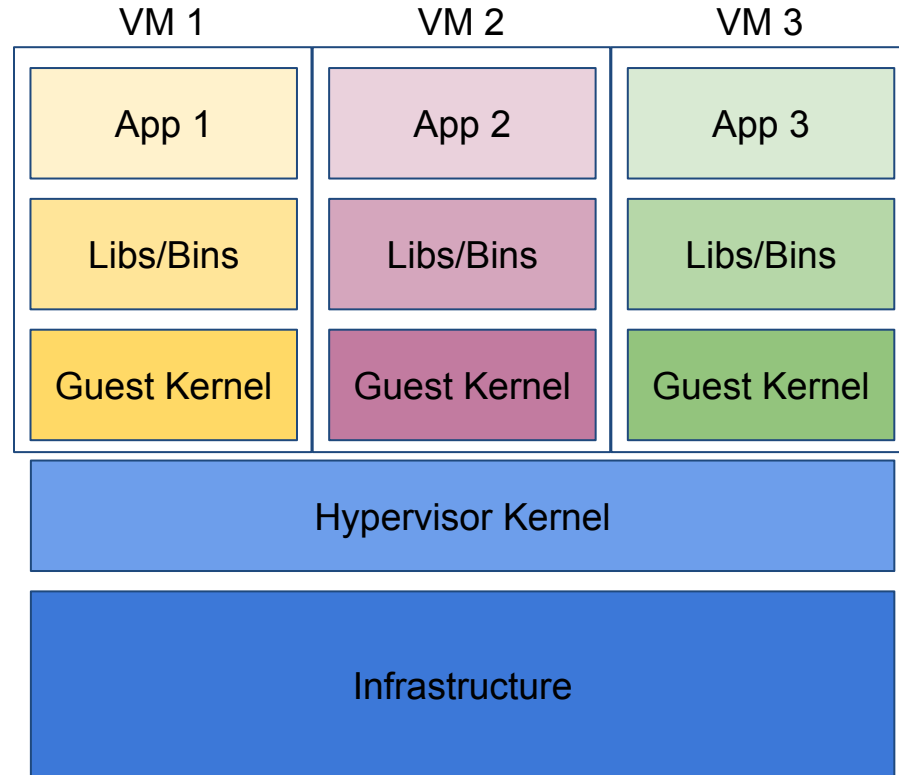
# Containers

**Containers provide a great solution to our problems :**

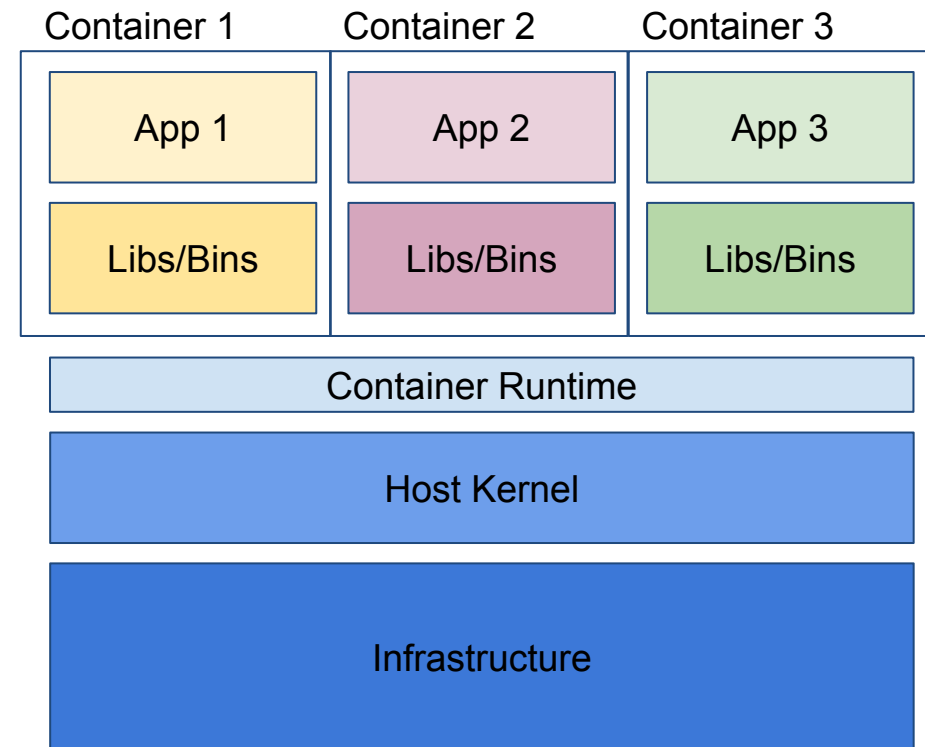
A **container** is a set of processes (typically 1) which is isolated from the rest of the operating system and can only see the specific libraries necessary to run them. This allows

- Easier software deployment:  
Users can leverage on installation tools that do not need to be available natively on the runtime host (e.g. package managers of various linux distributions).
- Software can be built on a platform different from the execution hosts.
- All necessary dependencies are packaged in one single object.
- Easy to publish and sign
- This ensure portability and reproducibility of the application stack on different production hosts, or between development and production environments.

# VMs vs Containers

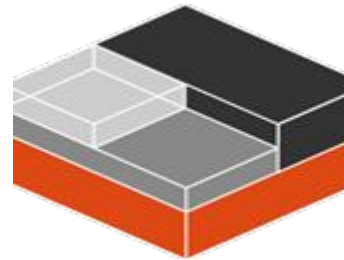


**VMs:**  
hardware virtualization + OS

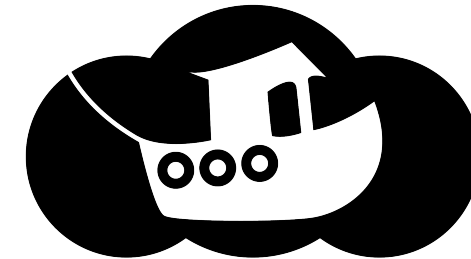


**Containers:**  
User defined software stack

## Linux container systems



Linux LXC



Charliecloud



podman



rkt



# Docker Basics

Docker packages software environments into **containers**, making it easy to share code without worrying about dependency installations, OS compatibility, or versioning issues.

## Key Terms

### • Container

- Package of libraries and dependencies
- Isolated from host
- “**docker run**” command creates a container

### • Image

- Read-only snapshot of a container
- Instantiate an image with one or many containers
- Share images on Docker Registry (eg, Docker Hub)
- “**docker build**” command creates an image

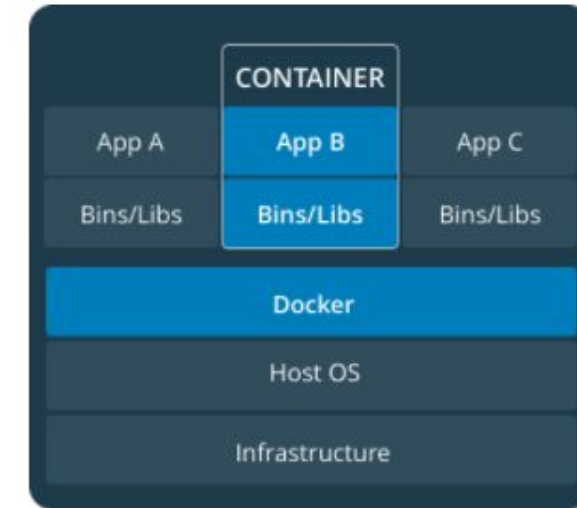
Developer



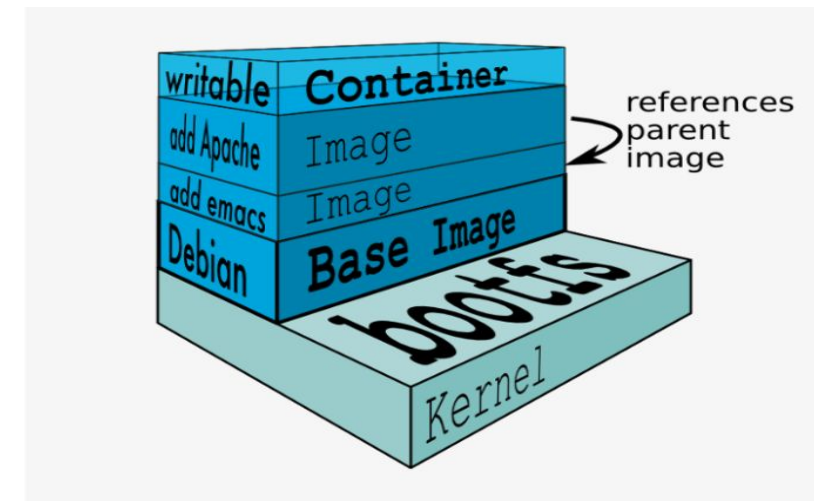
pull/push  
remote images



Docker Registry



<https://docs.docker.com/get-started/>



<https://medium.com/docker-captain/docker-basics-f1a06fde18fb>



# Install Docker

## Install Docker Desktop

– <https://docs.docker.com/get-docker/>

Docker overview

Get Docker

Get started

- Part 1: Getting started
- Part 2: Sample application
- Part 3: Update the application
- Part 4: Share the application
- Part 5: Persist the DB
- Part 6: Use bind mounts
- Part 7: Multi-container apps
- Part 8: Use Docker Compose
- Part 9: Image-building best practices
- Part 10: What next?

Language-specific guides (New)

Develop with Docker

Set up CI/CD

### Get Docker

Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications. By taking advantage of Docker's methodologies for shipping, testing, and deploying code quickly, you can significantly reduce the delay between writing code and running it in production.

You can download and install Docker on multiple platforms. Refer to the following section and choose the best installation path for you.

#### Docker Desktop for Mac

A native application using the macOS sandbox security model which delivers all Docker tools to your Mac.

#### Docker Desktop for Windows

A native Windows application which delivers all Docker tools to your Windows computer.

#### Docker for Linux

Install Docker on a computer which already has a Linux distribution installed.

Docker, download, documentation, manual

## Verify Installation

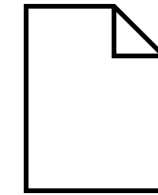
- **docker --version**  
Prints out docker version.
- **docker run hello-world**  
Output should include “your installation appears to be working correctly”.

# Simple Example of Docker project

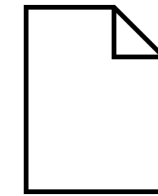
As an example, we will make a **simple web app**.

We will use three files:

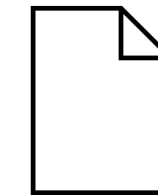
1. app.py – server
2. template/index.html – html
3. requirements.txt – dependencies



app.py



index.html



requirements.txt

<https://github.com/pontiggi/simple-docker> , adapted from  
<https://github.com/docker/labs/blob/master/beginner/chapters/webapps.md>

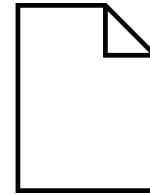
# Running the app without Docker

Deal with OS compatibility and versioning issues!

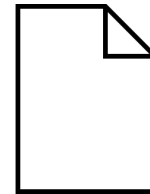
1. Install/update software (here, Python/pip)

2. Install other dependencies

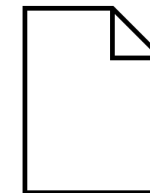
3. Run code (“python app.py”)



app.py



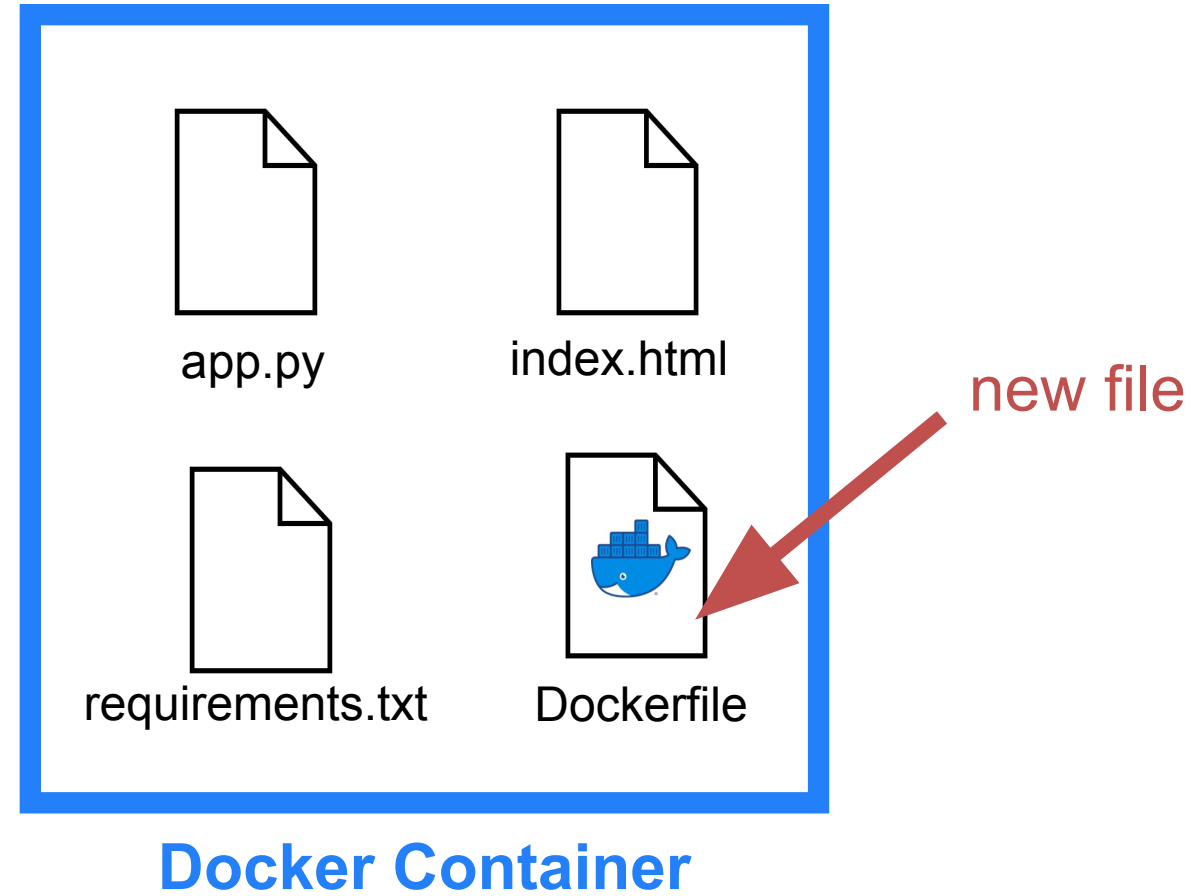
index.html



requirements.txt

# Running the app with Docker

1. “docker build ...”
2. “docker run ....”



# Example of Dockerfile

1. Install/update software (here, Python/pip)

```
>$ cat Dockerfile
# our base image
FROM alpine:latest
# Install python and pip
RUN apk add --update py3-pip
```

2. Install other dependencies

```
# install Python modules needed by the Python app
COPY requirements.txt /usr/src/app/
RUN pip3 install --no-cache-dir -r /usr/src/app/requirements.txt
```

3. Run code ("python app.py")

```
# copy files required for the app to run
COPY app.py /usr/src/app/
COPY templates/index.html /usr/src/app/templates/

# tell the port number the container should expose
EXPOSE 5000
ENV BGCOLOR 'black'

# run the application
CMD ["python3", "/usr/src/app/app.py"]
```

Dockerfile Documentation

<https://docs.docker.com/engine/reference/builder/>

# Docker Exercise - step 1 of 7

Verify Docker installation and run some basic docker commands

- **docker --version**

```
$> docker --version  
Docker version 20.10.5, build 55c4c88
```

- **docker run hello-world**

Output should include “your installation appears to be working correctly”.

- **docker run docker/whalesay cowsay Go Crimson**

(<https://hub.docker.com/r/docker/whalesay/>)

# Docker Exercise - step 2 of 7

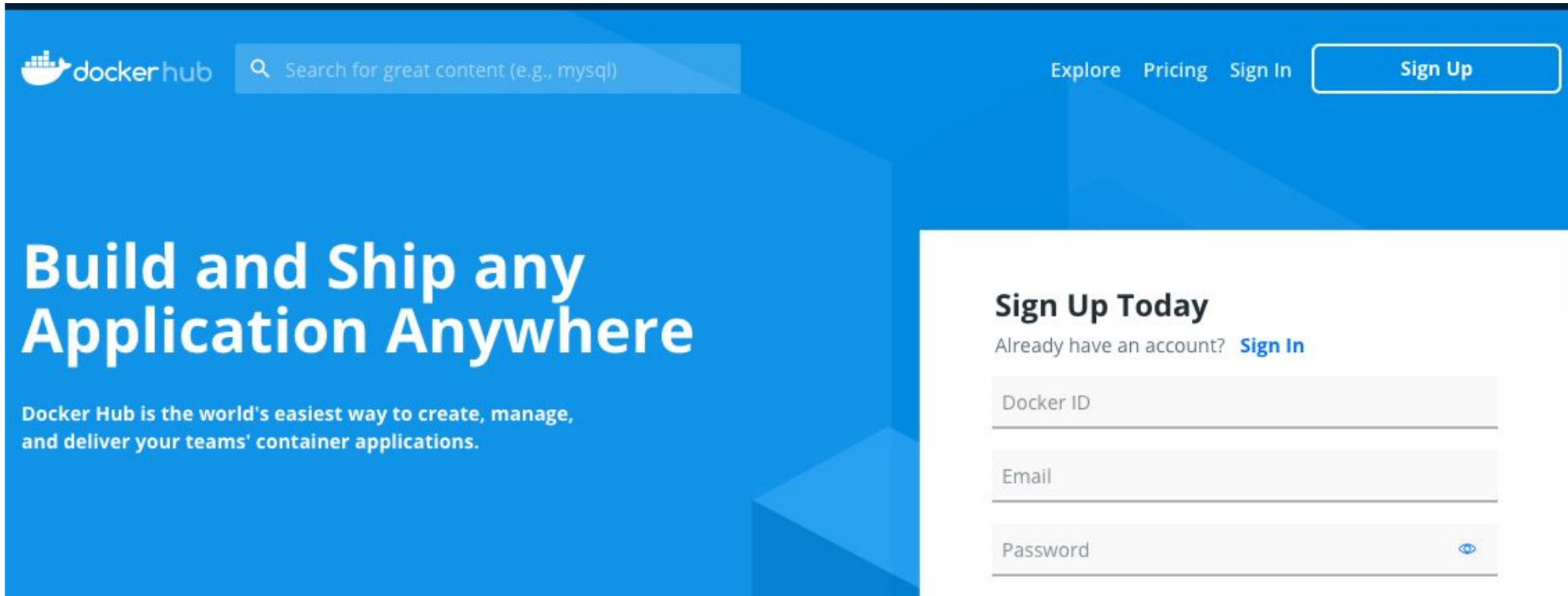
Experiment with Docker commands, such as:

- **docker ps**
  - List running containers
- **docker images**
  - List images
- **docker inspect docker/whalesay**
  - View data about the container
- **docker run -it docker/whalesay /bin/bash**
  - run bash to container
  - check the file /etc/os-release

# Docker Exercise - step 3 of 7

Sign up for Docker Hub

<https://hub.docker.com/>



The screenshot shows the Docker Hub website's sign-up interface. The header is blue with the Docker Hub logo, a search bar, and navigation links for Explore, Pricing, Sign In, and a Sign Up button. The main content area has a blue background with the text 'Build and Ship any Application Anywhere' and a description of Docker Hub. On the right, a white sign-up form is overlaid, titled 'Sign Up Today'. It includes a link for users who already have an account, followed by input fields for Docker ID, Email, and Password (with a toggle for visibility).

docker hub Search for great content (e.g., mysql) Explore Pricing Sign In Sign Up

## Build and Ship any Application Anywhere


Docker Hub is the world's easiest way to create, manage, and deliver your teams' container applications.

### Sign Up Today

Already have an account? [Sign In](#)

Docker ID

Email

Password 



# Docker Exercise - step 4 of 7

Build a Docker image and run a container.

Clone the simple-docker app repo

- git clone <https://github.com/pontiggi/simple-docker>  
(Exercise adapted from: <https://github.com/docker/labs/blob/master/beginner/chapters/webapps.md>)

In the same directory as “Dockerfile”, execute these commands:

- **docker build -t <DOCKER\_HUB\_USERNAME>/myfirstapp .**
  - Builds the image
  - The ‘-t’ flag tags the image with the name that follows.
  - The final ‘.’ means current directory.
- **docker run -p 8888:5000 --name myfirstapp <DOCKER\_HUB\_USERNAME>/myfirstapp**
  - Runs a container
  - The ‘-p’ flag binds the host port 8888 to the container’s port 5000.
  - The ‘--name’ (two dashes!) flag names the container.
  - <DOCKER\_HUB\_USERNAME>/myfirstapp is the name of the image to run.

# Docker Exercise - step 5 of 7

Publish your image on Docker Hub

- **docker login**  
(Log into Docker Hub)
- **docker push <DOCKER\_HUB\_USERNAME>/myfirstapp**  
(Push your image to Docker Hub.)

# Docker Exercise - step 6 of 7

Execute commands in a running container

- **`docker exec -ti myfirstapp /bin/sh`**

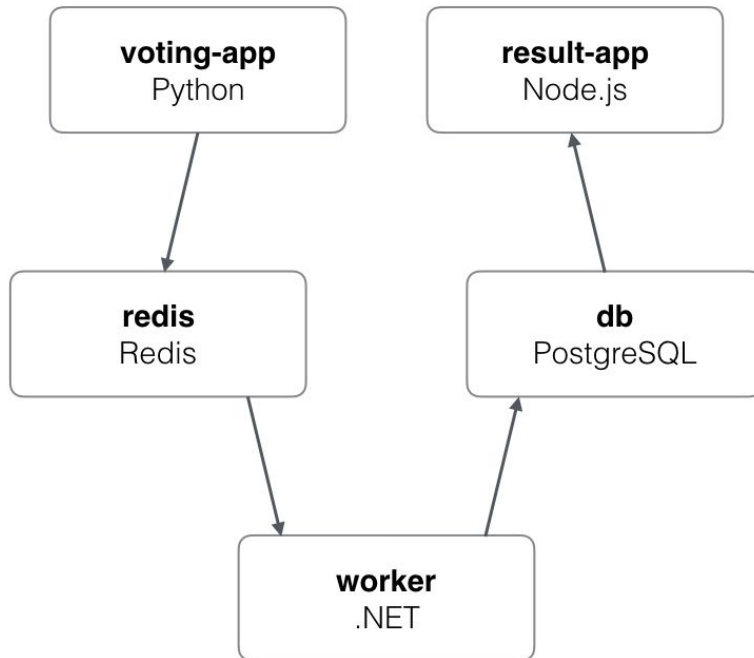
Stop and remove the container.

- **`docker stop myfirstapp`**
- **`docker rm myfirstapp`**

# Docker Exercise - step 7 of 7

Explore a more complex project, where several components, one in each container, contribute to compose the whole application.

<https://github.com/dockersamples/example-voting-app>



Explore the file docker-compose.yml

```
$> git clone https://github.com/dockersamples/example-voting-app
```

```
$> cd example-voting-app
```

```
$> docker compose up
```

<http://localhost:5000>

<http://localhost:5001>

# Three takeaways

1. Container vs. image (writable package versus read-only snapshot)
2. Dockerfile works like a list of commands to make your environment.
3. Docker Documentation <https://docs.docker.com/>

# Useful Resources

- [Docker Documentation](#)
- [Dockerfile Best Practices](#)
- [Example Medium Article](#) – “Intro to Docker for Front End Devs”
- [Example Docker Command Cheat Sheet](#)
- [Push Your Project to a New GitHub Repo](#)