Galvanize Deep Learning Workshop

Objectives

- Docker motivation

- What is Docker?

- How to make a Docker container

- Docker components

- Docker volumes

- Why we are using Docker in this course

- Docker command reference

- References

Docker motivation

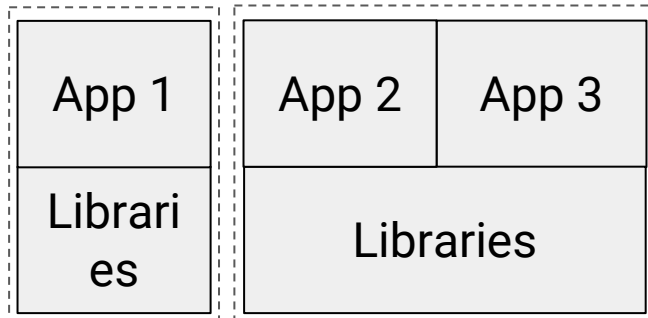From founder and CTO of Docker Soloman Hykes, as motivation for Docker:

"Shipping code to the server should not be hard."

The developer can specify what's inside the container, and IT just needs to handle the container.

Container 1	Container 2	
App 1	App 2	App 3
Libraries	Libraries	

Docker Engine

Host operating system

| Server (your computer) |
|---|

No matter where the container goes, what's inside will work the same way.

What is Docker

- Docker is an open-source project based on Linux containers.

- Docker is a software *containerization* platform.

- A docker *container* is a stand-alone piece of software that includes everything needed to run it.

- Containers are isolated from each other, but can share libraries where possible.

- Containers are created with Linux, but share a *kernel* with almost any type of OS.

A kernel is the central part of the OS.

Container 1	Container 2	
App 1	App 2	App 3
Libraries	Libraries	
Docker Engine		
Host operating system		
Server (your computer)		

4
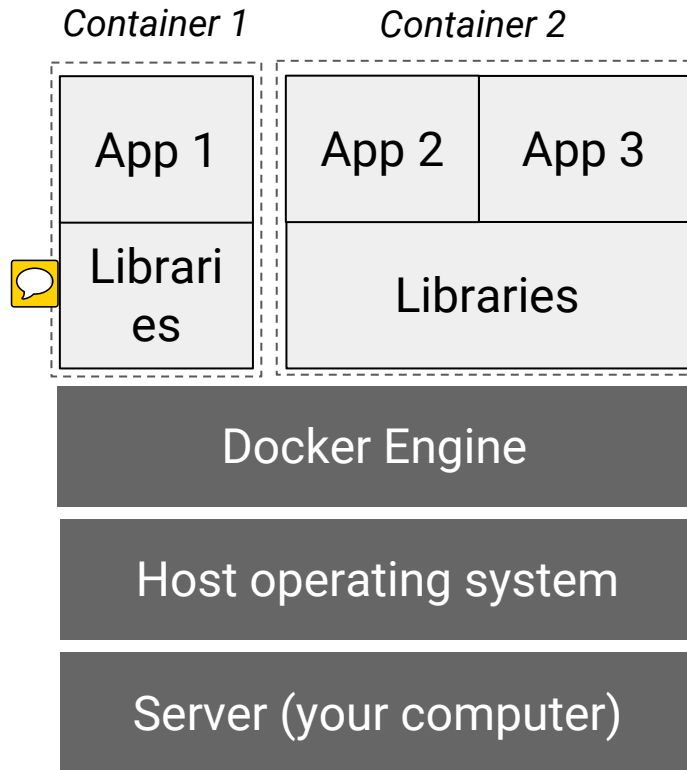
Docker Timeline and Popularity

Docker was made open source in 2013 by Soloman Hykes at dotCloud.

Main contributors in 2016: Docker team, Cisco, Google, Huawei, IBM, Microsoft.

Has been downloaded more than 13 billion times as of 2017.

Main reasons for popularity:

Ease of use: Build and test it on your laptop and it will run anywhere.

Speed: Containers are sandboxed environments running on the kernel that take up fewer resources and are faster to start-up than virtual machines.

Docker Hub: the free app-store for Docker images

Modularity and Scalability: An application can be divided into multiple containers but those containers can communicate.

Making a Docker container from scratch (1/3)

A text *Dockerfile* defines what's in your container:

```
# Use an official Python runtime as a parent image
FROM python:2.7-slim

# Set the working directory to /app
WORKDIR /app

# Copy the current directory contents into the container at /app
ADD . /app

# Install any needed packages specified in requirements.txt
RUN pip install --trusted-host pypi.python.org -r requirements.txt

# Make port 80 available to the world outside this container
EXPOSE 80

# Define environment variable
ENV NAME World

# Run app.py when the container launches
CMD ["python", "app.py"]
```

Source: Docker Docs, Getting Started   6

Making a Docker container from scratch (2/3)

Put the *Dockerfile* and other files that define your application in the same directory and then build them into a *Docker Image* using the *Docker Client*:

```
$ ls
Dockerfile              app.py                  requirements.txt


docker build -t friendlyhello .
```

View the *Docker Image* "friendlyhello" that you've created:

```
$ docker images


REPOSITORY              TAG                     IMAGE ID
friendlyhello           latest                  326387cea398
```

Making a Docker container from scratch (3/3)

Use the *Docker Client* and your *Docker Image* to build your container:
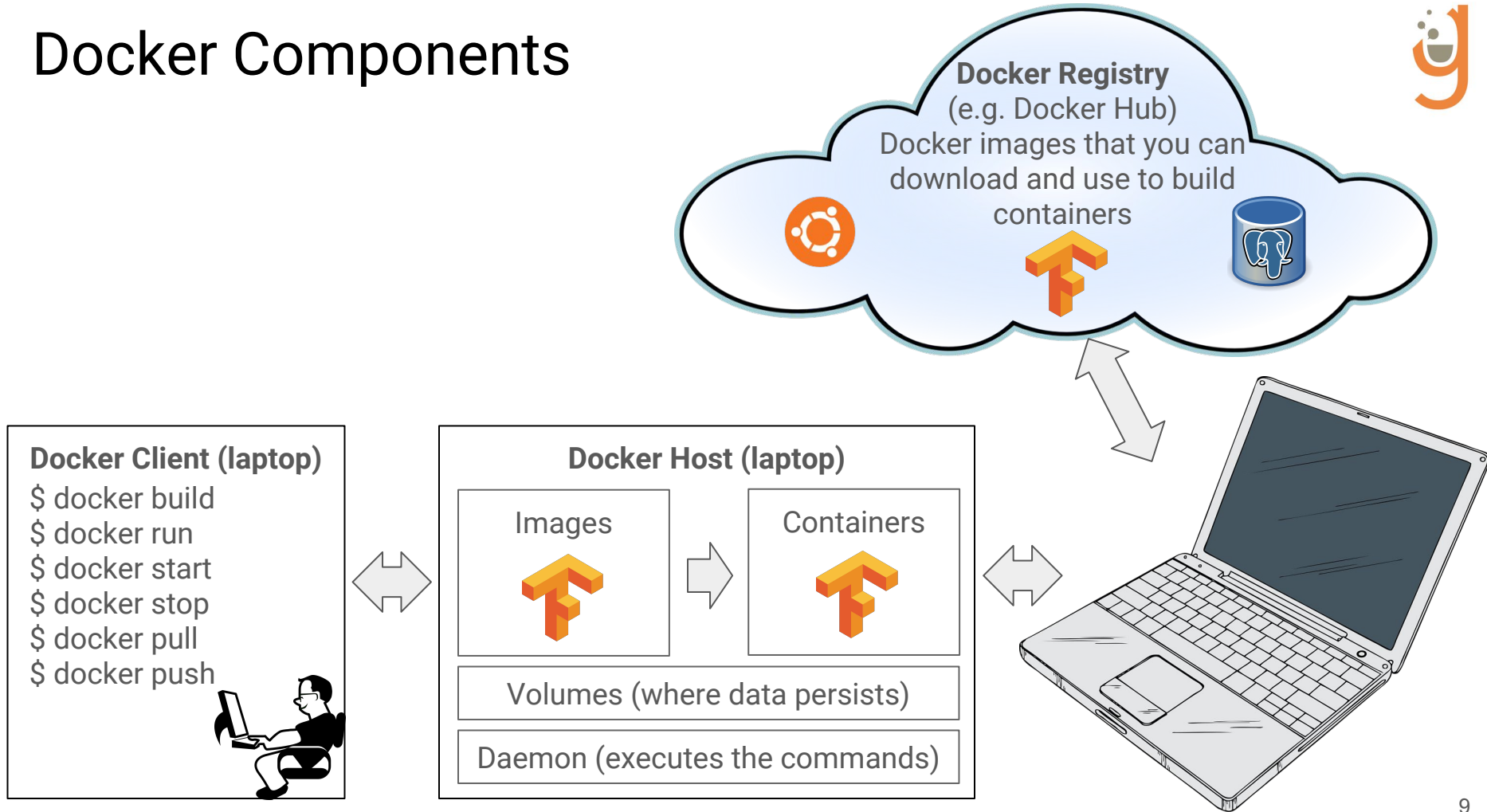
```
docker run -p 4000:80 friendlyhello
```

You can see what containers are presently running:

```
$ docker container ls
CONTAINER ID          IMAGE                 COMMAND
1fa4ab2cf395          friendlyhello         "python app.py"
```

Note: we won't be doing all these steps. We'll be downloading and building a tensorflow image from *Docker Hub*.

Docker Components

Docker Registry
(e.g. Docker Hub)
Docker images that you can download and use to build containers

Docker Client (laptop)

$ docker build
$ docker run
$ docker start
$ docker stop
$ docker pull
$ docker push

Docker Host (laptop)

Images

Containers

Volumes (where data persists)

Daemon (executes the commands)

Docker Volumes

- Docker volumes are the preferred way of persisting data needed and created by containers.

- Containers by themselves have a writable layer (you can store data in a container without using a volume), but a volume is better because:
    - it doesn't increase the size of the container using it
    - the volume exists outside of the lifecycle of the container
    - the volume can be shared with other containers

- You can create a volume when a container is made.

- You can move data back and forth between the host file system and the docker volume.

```
docker cp /tmp/my_data/. my_container:/my_data
```

Docker volumes

Why we use Docker for this workshop

- You all are on different operating systems, but we'd like to make sure that Tensorflow works the same way for all of you during this part-time course.

  - Don't have much time for debugging! If we didn't use Docker, we'd be debugging in Linux, Windows, and Mac operating systems.

- Perfect application for Docker!
- The CE (Community Edition) is free.

- [Install Docker Ubuntu Linux](#)
- [Install Docker Mac](#)
- Install [Docker Toolbox](#) for Windows and older versions for Mac

Docker command reference (1/2)

Basic Docker commands

Command	Description
`docker run <image>`	Make and start a container from an image
`docker start <name | id>`	Start a pre-existing container
`docker stop <name | id>`	Stop a container
`docker ps`	Show running containers
`docker ps -a`	Show running and stopped containers
`docker rm <name | id>`	Removes a container

Docker command reference (2/2)

Docker reference

Command	Description
`docker`	Show available docker commands
`docker COMMAND --help`	Gets help on COMMAND
`docker logs <id>`	Return log of container (including browser link and token)

References

- [Docker Documentation](#)

- [Docker on Github](#)

- [Docker Hub](#)

- [Solomon Hykes talk at Twitter, 2 Oct. 2013](#)

- [Preethi Kasireddy - Beginner friendly docker concepts](#)