Build, Ship, and Run

Build, Ship, Run, Any App Anywhere

From Dev

To Ops

Any App

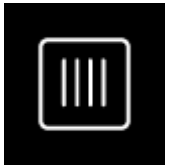
Any OS

Anywhere

Some Docker vocabulary

Docker Image

The basis of a Docker container. Represents a full application



Docker Container

The standard unit in which the application service resides and executes

Docker Engine

Creates, ships and runs Docker containers deployable on a physical or virtual, host locally, in a datacenter or cloud service provider

Registry Service (Docker Hub or Docker Trusted Registry)

Cloud or server based storage and distribution service for your images

Basic Docker Commands

```
$ docker pull mikegcoleman/catweb:latest

$ docker images

$ docker run -d -p 5000:5000 --name catweb mikegcoleman/catweb:latest

$ docker ps

$ docker stop catweb (or <container id>)

$ docker rm catweb (or <container id>)

$ docker rmi mikegcoleman/catweb:latest (or <image id>)
```

Dockerfile – Linux Example

```
# our base image
FROM alpine:latest

# Install python and pip
RUN apk add --update py-pip

# upgrade pip
RUN pip install --upgrade pip

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

# 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

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

- Instructions on how to build a Docker image
- Looks very similar to “native” commands
- Important to optimize your Dockerfile

Image Layers

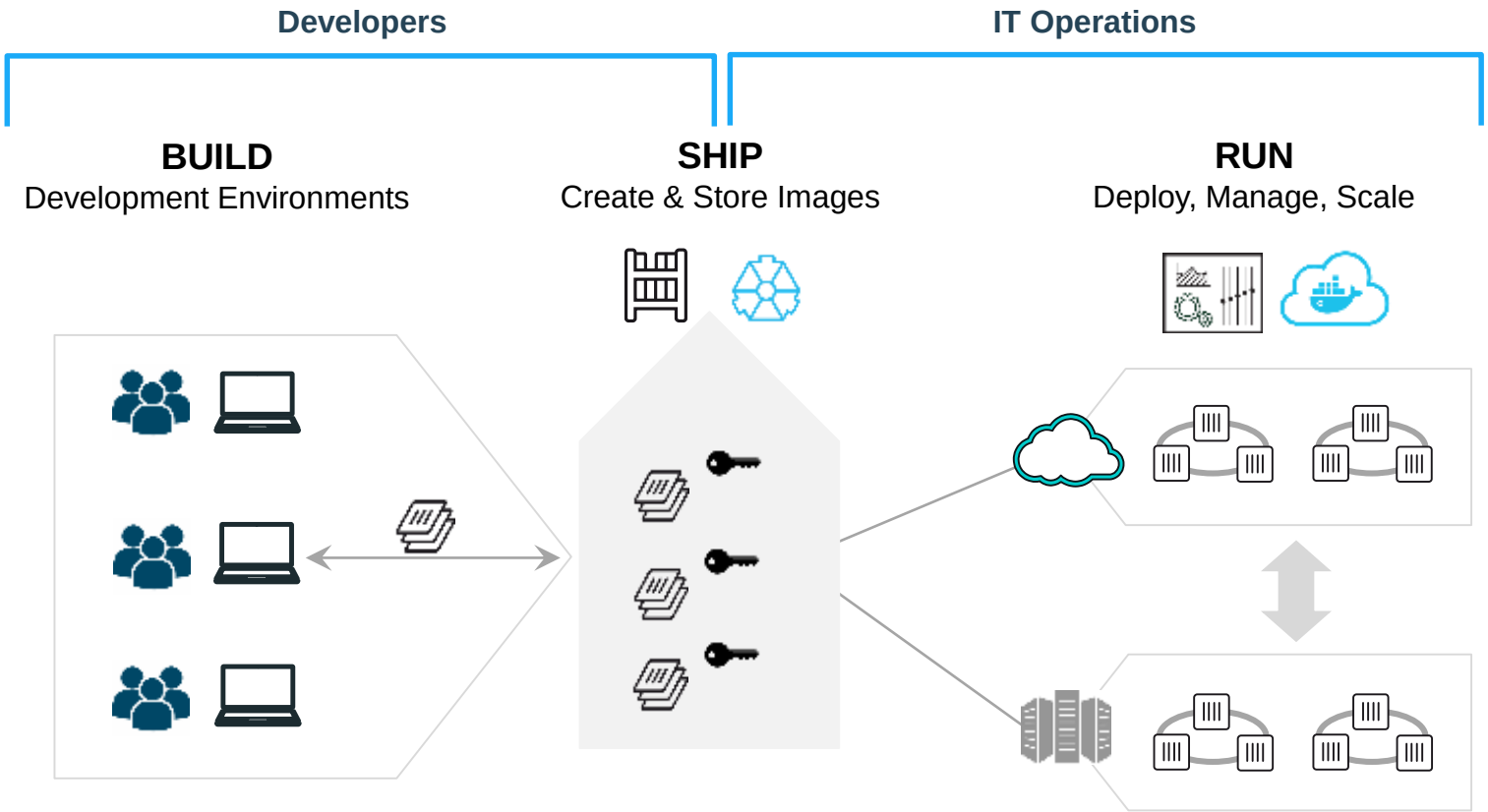
Basic Docker Commands

```
$ docker build -t mikegcoleman/catweb:2.0 .
$ docker push mikegcoleman/catweb:2.0
```

```
1  our base image
2  FROM alpine:latest
3
4  # Install python and pip
5  RUN apk add --update py-pip
6
7  # upgrade pip
8  RUN pip install --upgrade pip
9
10 # install Python modules needed by the Python app
11 COPY requirements.txt /usr/src/app/
12 RUN pip install --no-cache-dir -r /usr/src/app/requirements.txt
13
14 # copy files required for the app to run
15 COPY app.py /usr/src/app/
16 COPY templates/index.html /usr/src/app/templates/
17
18 # tell the port number the container should expose
19 EXPOSE 5000
20
21 # run the application
22 CMD ["python", "/usr/src/app/app.py"]
```

docker

Put it all together: Build, Ship, Run Workflow



docker

What about data persistence?

- Volumes allow you to specify a directory in the container that exists outside of the docker file system structure
- Can be used to share (and persist) data between containers
- Directory persists after the container is deleted
  - Unless you explicitly delete it
- Can be created in a Dockerfile or via CLI

WHAT IS DOCKER

- Allows you ship code along with all its dependencies in a self-contained manner

- Dockerfile like a manifest allows you to describe these dependencies and steps to set it up

- Spin up many instances of this image as you want (container)

- Cloud ready

WHY USE IT

- So many many libraries, so many many versions
- Dependency Install nightmare, be shielded from inadvertent upgrades
- Simplify and speed up focus on actual ML problem not supporting infrastructure

STEP 1

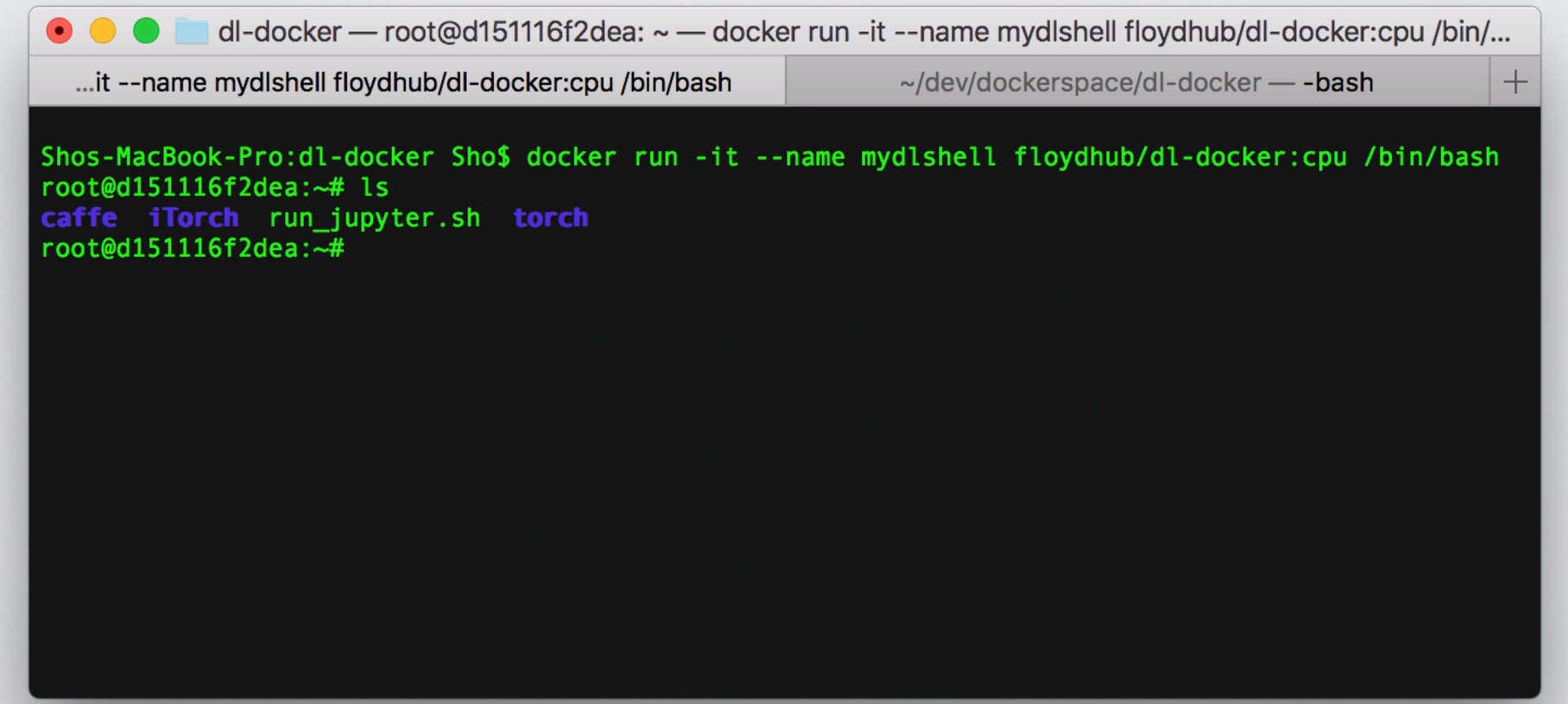
Download the image of choice from Docker Hub

```
$ docker pull floydhub/dl-docker:cpu
```

STEP 2

Start container with that image

$: docker run -it --name mydlshell floydhub/dl-docker:cpu /bin/bash



STEP 2B

Another Way to Start Container … Using Assigned Label

$: docker start -ia mydlshell

STEP 3

Interact with the container to perform various tasks

Approach 1: Copy files into Container

$: docker cp ~/dev/dockerspace/census_keras.py dl-docker/ mydlshell:/root/test/census_keras.py

STEP 3B

Or Share a Volume (my preferred method)

$: docker run -it -v ~/dev/dockerspace/dl-docker:/projects/dl-docker --name mydlspace floydhub/dl-docker:cpu

$: docker start mydlspace

$: docker exec -it mydlspace python /projects/dl-docker/census_keras.py

“HOW CAN IT BE THIS EASY ?”