Docker Basics Continued & Docker Swarm

starts at 9:05 pm

Agenda

1. Logging Drivers
2. Creating Docker Images
    1. Committing from an existing container
    2. Dockerfile
3. Docker Compose
4. Multi-Stage Builds
5. Docker inspect
6. Docker Registry
    1. Docker Hub
7. Docker Swarm

Logging Drivers

stdout ✓

stderr ✓

stdinp

different types of logging drivers.

→ syslog

/var/log/syslog

```
docker info | grep "Logging Driver"
```

```
docker run --log-driver=syslog ubuntu
```

→ journald

→ stored in binary log format

/var/log/journal

→ awslogs

sends logs to Amazon Cloudwatch.

→ change default logging driver for all containers

/etc/docker/daemon.json

```
{
  "log-driver": "awslogs",
```

```
  "log-opts": {
    "awslogs-region": "us-west-2",
    "awslogs-group": "my-log-group"
    "awslogs-stream": "my-log-stream"
  }
}
```

```
{
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "10m",
    "max-file": "3"
  }
}
```

docker logs <container-id>

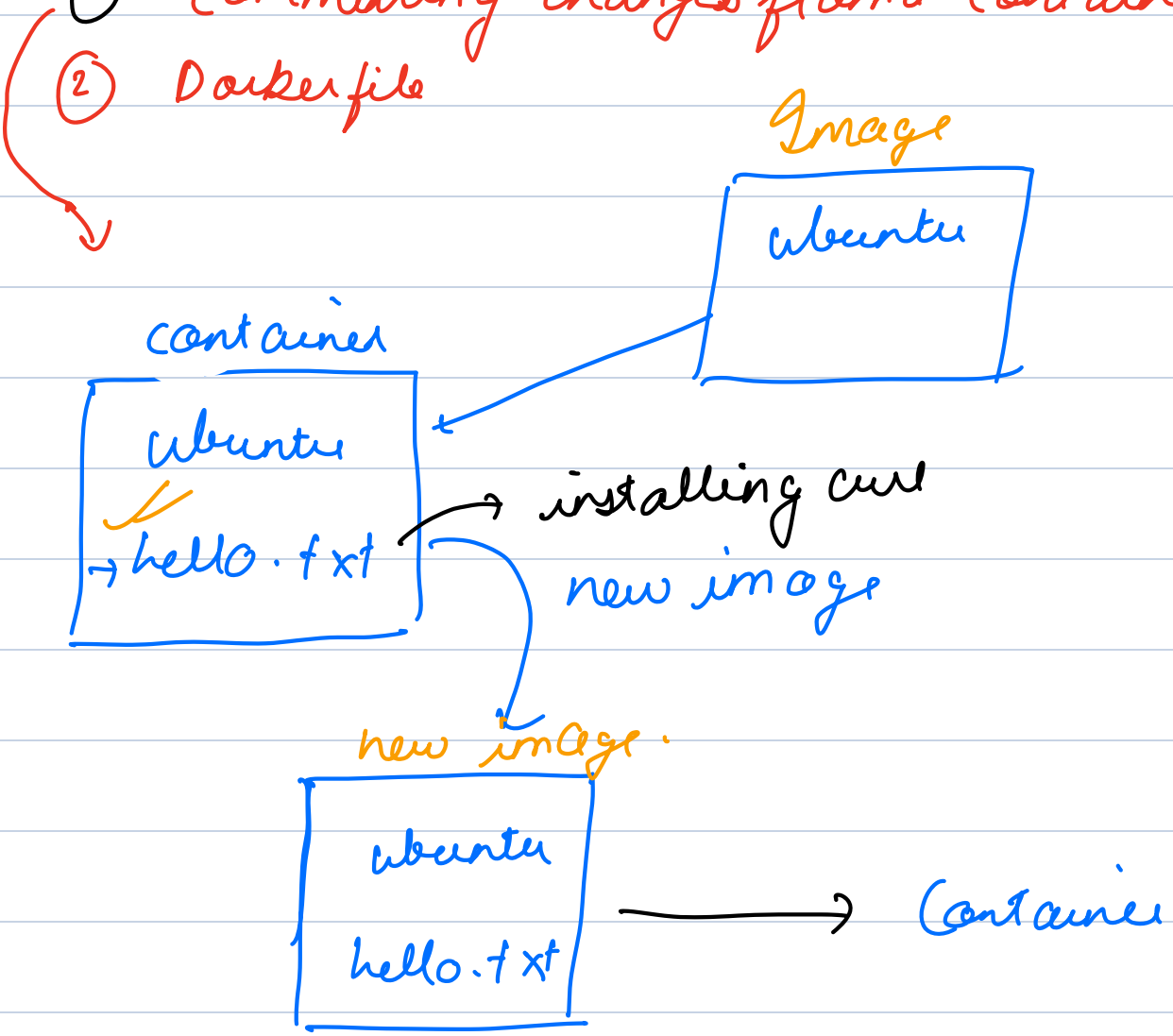
docker logs -f <container-id>

docker logs --tail 10 <container>

→ Creating Docker Images

① committing changes from a Container

② Dockerfile



docker run -it ubuntu

Make changes

install curl

create hello.txt

exit

docker commit id my-custom-ubuntu

↳ container-id

Removing containers

docker rm $(docker ps -aq)

Removing Images

docker rmi image-id

→ Dockerfile

Dockerfile —Build→ Image

FROM base image.

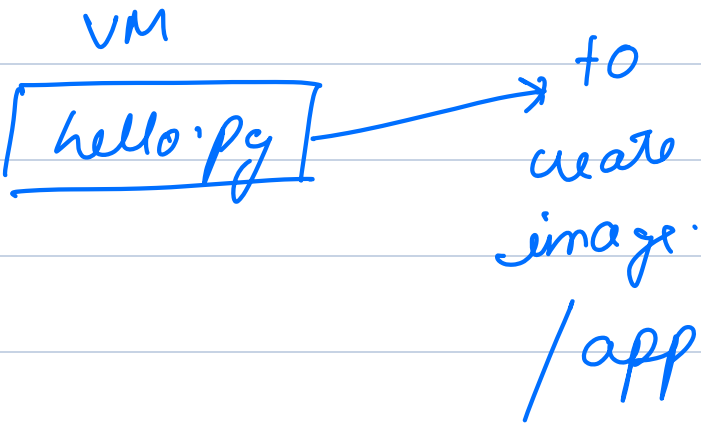
FROM python:3.9-slim ——→ FROM ubuntu

WORKDIR /app RUN apt-get install python

COPY . .

RUN echo "Running hello.py"

CMD ["python", "hello.py"]



python hello.py

docker build -t python-app .

↓

image got created

↓

docker run python-app

Break → 10:20pm

ADD hello.py /app/

COPY hello.py /app/

COPY http://som-wesite /app/

ADD https://www.w3.org/TR/PNG/iso_8859-1.txt /app/iso_8859-1.txt

CMD and ENTRYPOINT

define behaviour of container when it starts

CMD vs ENTRYPOINT

Feature	CMD	ENTRYPOINT
Purpose	Default command/arguments for the container.	Configures the container as an executable.
Overridable	Yes, completely overridable with `docker run`.	No, unless `--entrypoint` is used.
Command Arguments	Can pass arguments when using `docker run`.	Appends arguments to the specified command.
Flexibility	Better for default behaviors.	Better for containers with fixed behavior.

hello.py

```
import sys

print("Hello, ", " ".join(sys.argv[1:]))
```

Dockerfile

FROM python:3.9-slim

WORKDIR /app

COPY hello.py .

ENTRYPOINT ["python", "hello.py"]

CMD ["World!"]

LABEL version = "1.0"

docker build -t cmd-vs-entrypoint-demo .

docker run cmd-vs-entrypoint-demo Alice

docker run --entrypoint echo cmd-vs-entrypoint-demo "Hello from new entrypoint!"

Hello from new entrypoint! → output

FROM ubuntu:latest

CMD echo "This is the first CMD"

CMD echo "This is the second CMD"

CMD echo "This is the last CMD"

#First ENTRYPOINT (This will be ignored)

ENTRYPOINT ["echo", "This is ENTRYPOINT 1"]

#Second ENTRYPOINT (This will be used)

ENTRYPOINT ["echo", "This is ENTRYPOINT 2"]

ENV

ENV APP_ENV = Production.

EXPOSE EXPOSE 80

VOLUME VOLUME ["/data"]

LABEL LABEL version = "1.0"

/data (host) → container

Docker Compose

docker-compose.yml.

docker-compose up

docker-compose down.

```
version: '3'

services:
  web:
    image: nginx
    container_name: webserver
    ports:
      - "8080:80"

  db:
    image: mysql:5.7
    container_name: mysql_db
    environment:
```

MYSQL_ROOT_PASSWORD: example

ports:

- "3306:3306"

docker-compose up -d

docker-compose down

docker-compose logs

docker run -it --cpus="0.5" ubuntu

100ms. 200 ms.

0.5 1 sec.

--cpus="0.1"

```
FROM scratch

COPY python3 /usr/local/bin/python

COPY app.py /app/app.py

WORKDIR /app

CMD ["/usr/local/bin/python", "app.py"]
```