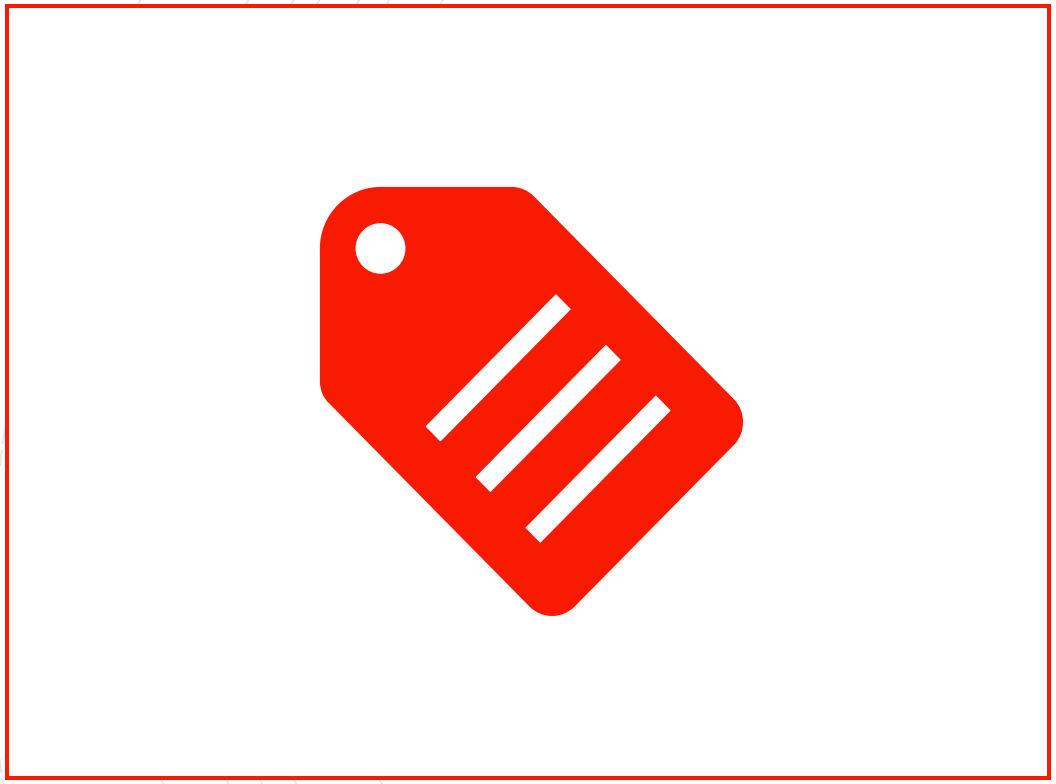
Docker

Ahmed Abdelnabi

Agenda

- Docker Images
- Docker File
- Docker Networks
- Docker Compose
- Lab

Custom Images



- To create our own image, we can do one of the following:
  - Create a container from your desired image, add and edit
    - whatever you want then commit all these changes to an image.
    - docker commit CONTAINER_ID "new_image_name"
  - Use Dockerfile
    - Dockerfile is a set of instructions.
    - Each instruction used to build an image.
    - Layers are stacked.

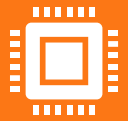
Dockerfile instructions

- FROM: Initializes a new build stage and sets the Base Image.
- RUN: Will execute any OS commands in a new layer to build the image.
- CMD: Provides a default for an executing container.
- ENTRYPOINT: Allows for configuring a container that will run as an executable.
- EXPOSE: Informs Docker that the container listens on the specified network ports at runtime
- ENV: Sets the environment variable <key> to the value <value>
- ADD: Copies new files, directories or remote file URLs from <src> and adds them to the filesystem of the image at the path <dest>.
- COPY: Copies new files or directories from <src> and adds them to the filesystem of the container at the path <dest>.

Dockerfile instructions

- LABEL: Adds metadata to an image
- VOLUME: Creates a mount point with the specified name and marks it as holding externally mounted volumes from native host or other containers
- USER: Sets the user name (or UID) and optionally the user group (or GID) to use when running the image and for any RUN, CMD,
- WORKDIR: Sets the working directory for any RUN, CMD, ENTRYPOINT, COPY, and ADD instructions that follow it in the Dockerfile
- ARG: Defines a variable that users can pass at build-time to the builder with the docker build command, using the --build-arg <varname>=<value> flag

RUN VS Cmd VS Entreypoint



RUN. Mainly used to build images and install applications and packages. Builds a new layer over an existing image by committing the results.

CMD. Sets default parameters that can be overridden from the Docker Command Line Interface (CLI) when a container is running.

ENTRYPOINT. Default parameters that cannot be overridden when Docker Containers run with CLI parameters.

ADD VS COPY

COPY takes in a src and destination. It only lets you copy in a local or directory from your host (the machine-building the Docker image) into the Docker image itself.

ADD lets you do that too, but it also supports 2 other sources. First, you can use a URL instead of a local file/directory. Secondly, you can extract tar from the source directory into the destination.

Docker Networks

Docker Networks

- Bridge:
  - Private internal network created by docker on the host.
  - Default network the container attach to.
  - Each created container get internal IP address usually in range 172.17.x.x.
  - Containers can access each. Others by this IP
  - To Access any of these containers "PORT MAPPING"
- Host:
  - No type of isolation between docker container and docker host
  - No need to "PORT MAPPING"
  - Only one docker container can be access

Docker Networks

- None:
  - Docker container not attached to any network
  - It will not be accessible outside even from other containers

- Overlay:
  - Create new private network that allow all containers to access each other.
  - Docker Swarm can create it.
  - Docker network create --driver overlay --subnet 10.0.9.0/24 name.
  - Then attach the container or services to this network during creation.

Lab

- P1: What do you know about Docker Images Tags and what is the difference between them?
- P2: Create docker image for Python app
- P3: Create your own nginx docker image "NEVER USE FROM nginx"
  - Install nginx
  - Copy Two html files one as file and another as .tar "/usr/share/www/html"
  - Expose
  - Start
  - Port mapping
- P4: Create your bridge network, two containers from ubuntu image with different names and try to ping each other using NAME.

Docker Compose

Docker Compose

- Docker Compose is used to run multiple containers as a single service.
- Container named Service
- All services are to be defined in a YAML format.
- compose file name MUST be “docker-compose.yml”

Docker Compose Commands

- docker-compose up => build services
- docker-compose kill => Kill the containers
- docker-compose logs => Show the logs of the containers
- docker-compose down => Stop and remove containers, volumes and networks
- docker-compose rm => Remove stopped containers

Docker Compose File Syntax

```
version: '3' # if no version is specified then v1 is assumed.
services: # containers. same as docker run
  service_name1: # container name. this is also DNS name inside network
    image: # name of the image
    command: # Optional, replace the default CMD specified by the image
    environment: # same as -e in docker run
    ports: # same as –p in docker run
    volumes: # same as -v in docker run
  service_name2:
volumes: # Optional, same as docker volume create
networks: # Optional, same as docker network create
```

Thank You