# DOCKER CHEAT SHEET

## WHAT IS DOCKER AND WHY DO WE NEED CONTAINERS?

**Docker** is an open-source project for building containers and container-based applications. It was introduced in 2013 to solve the challenge of running multiple applications on one server, while keeping the components of each application separate.

**How does it work?**
Docker small, lightweight execution environments make shared use of the operating system kernel, while keeping them running in isolation from one another. Making it possible to simplify the development cycle, for example when testing your application and setting up different database versions quickly. Docker allows you to easily manage different containers and versions.

## HOW TO INSTALL DOCKER

Update the apt package index and install packages to allow apt to use a repository over HTTPS:

```
$ sudo apt-get update

$ sudo apt-get install \
    apt-transport-https \
    ca-certificates \
    curl \
    gnupg-agent \
    software-properties-common
```

Add Docker's official GPG key:
Use the following command to set up the **stable** repository.

```
$ curl -fsSL https://download.docker.com/linux/debian/gpg | sudo apt-key add -
$ sudo add-apt-repository \
   "deb [arch=amd64] https://download.docker.com/linux/debian \
   $(lsb_release -cs) \
   stable"
```

Update the apt package index, and install the latest version of Docker Engine and containerd, or go to the next step to install a specific version

```
$ sudo apt-get update
$ sudo apt-get install docker-ce docker-ce-cli containerd.io
```

## GENERAL DEFINITIONS

**General Definitions**

**Image:** Static snapshot of a container's configuration.

**Container:** Application sandbox, where each container is based on an image.

**Layer:** Image is composed of a read-only file system layer.

**Docker registry:** Remote server for sorting Docker images. For example, the JFrog Container Registry.

**Docker hub:** Central official repository registry service provided by Docker, see the integration below.

**Dockerfile:** Configuration file with build instructions for a Docker image.

**Volume:** Directory shared between the host and the container.

**Docker host:** Server that runs docker-engine.

**Docker engine:** Docker platform installation running on a given host.

A beginner's guide to understanding and building Docker images >

## 10 USEFUL DOCKER COMMANDS

**10 useful Docker commands**

**Informational**

`$ docker ps` List all running containers. Add the -a flag to get a list of the stopped containers.

`$ docker logs` <Container_Name> List the logs from a running container.

`$ docker image ls` List all locally stored images.

**Container Management**

`$ docker create <Image_Name>` Create a container without starting it

`$ docker run <Image_Name>` Create and start a container

`$ docker stop <Container_Name>` Stop a running container.

`$ docker rm <Container_Name>` Delete a container that isn't running. Use the --force flag to delete a running container.

`$ docker rmi <Image_Name>` Remove an image.

`$ docker pull <Image_Name>` Pull an image from a registry.

`$ docker push <Image_Name>` Push an image to a registry.

## WHAT IS DOCKER COMPOSE?

Docker Compose is a tool for defining and running multi-container Docker applications. Compose uses a YAML file to configure your application services. Then, with a single command, it allows you to create and start all services from your configuration.

**3 step process to start using docker-compose:**

1. Define your app's environment with a Dockerfile.
2. Define the services that make up your app in docker-compose.yml, enabling them to run together in an isolated environment.
3. Run the `docker-compose up` command and Compose will start and run your entire app.

```
FROM python:3.7-alpine
WORKDIR /code
ENV FLASK_APP=app.py
ENV FLASK_RUN_HOST=0.0.0.0
RUN apk add --no-cache gcc musl-dev linux-headers
COPY requirements.txt requirements.txt
RUN pip install -r requirements.txt
EXPOSE 5000
COPY . .
CMD ["flask", "run"]
```

Read more about getting started with Docker compose >

## HOW TO INSTALL DOCKER-COMPOSE

**How to install docker-compose**

Run the following command to download the current stable release of Docker Compose:

```
$ sudo curl -L "https://github.com/docker/compose/releases/download/1.28.2/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
```

Apply executable permissions to the binary:

```
$ sudo chmod +x /usr/local/bin/docker-compose
```

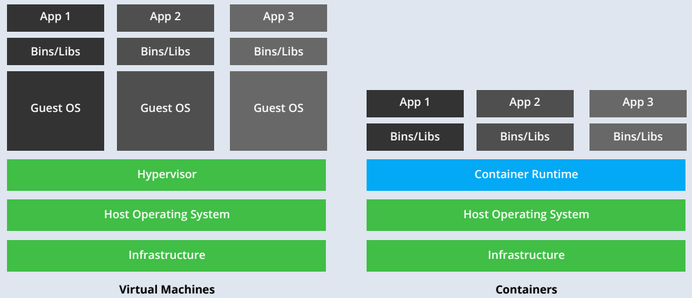
You can also create a symbolic link to /usr/bin or any other directory in your path, such as:

```
$ sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
```

## DOCKER CONTAINERS VS VIRTUAL MACHINES

A virtual machine (VM) is a compute resource that uses software instead of a physical computer. It provides the ability to run what appears to be many separate computers on hardware that is actually one computer. The operating systems (OS) and their applications share hardware resources from a single host server or from a pool of host servers. Each VM requires its own underlying OS and the hardware is virtualized.

A Docker container is a bit different. The underlying computer is like a virtual machine, but the OS service is virtualized. Each container shares the host OS kernel and usually also the binaries and libraries as well. Shared components are read-only. Sharing OS resources, such as libraries, significantly reduces the need to reproduce the operating system code. This means that a server can run multiple workloads with a single operating system installation.




## DOCKER & JFROG

The JFrog Container Registry is a secure private Docker registry that lets you manage all your Docker images.

JFrog supports unlimited pulls from Docker hub out of the box without any additional configurations. Users are automatically excluded from Docker hub's image rate limit and enjoy endless access to Docker Hub.