**Docker Compose**

Docker Compose is another best tool for docker to setup multi-container environments. Using this create a single compose file with defining all the containers with there environments. You can easily use single command to build images and run all the containers.

There is the three-step process to work with Docker Compose.

1. Define application environment with Dockerfile for all services.  
2. Create a docker-compose.yml file defining with all services under application.  
3. Run **docker-compose up** to run all services under applications.

You must have Docker Engine installed on your system. If you don’t have already installed, Visit our Docker installation section of this tutorial.

**Install Docker Compose**

Visit the Docker compose official Github page and download the latest version of Docker compose tool. You can also install Docker compose 1.16.1 using the following command. Before installing the specific version, You must check the compatibility on releases page with your docker version.

$ curl -L https://github.com/docker/compose/releases/download/1.16.1/docker-compose-`uname -s`-`uname -m` > /usr/local/bin/docker-compose

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

**Docker Compose Example File**

A docker-compose.yml file required to create to start working with the docker compose. Below is a sample configuration file of version 3. This file has only one service added in it named web.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | version: '3'  services:    db:       image: mysql       container\_name: mysql\_db       restart: always       environment:          - MYSQL\_ROOT\_PASSWORD="secret"    web:      image: apache      build: .      container\_name: apache\_web      restart: always      ports:        - "8080:80" |

**Docker Compose CLI Reference**

The docker-compose command provides a number of subcommands to manage Docker containers with docker-compose. Please find below details of the subcommands. Before reading below commands remember that you passed service name as an argument (not container name)

**build –**

The build option is used to build images for services for which build is defined.

$ docker-compose build ## Build all services

$ docker-compose build web ## Build single service

**up –**

Use to create docker containers with available services in docker-compose.yml file in current directory. Use -d switch to launch containers in daemon mode.

$ docker-compose up -d ## Create all containers

$ docker-compose up -d web ## Create single container

**down –**

This will stop and delete all containers, network and associated images for the services defined in a config file

$ docker-compose down ## Restart all containers

$ docker-compose down web ## Restart single container

**ps –**

This will list all containers created for the services defined in a config file with there status, port bindings and command.

$ docker-compose ps

**exec –**

This will execute a command to the running container. For example list files in container associated with **web** service.

$ docker-compose exec web ls -l

**start –**

This will start stopped containers of the services defined in config file

$ docker-compose start ## Start all containers

$ docker-compose start web ## Start single container

**stop –**

This will stop running containers for the services defined in config file

$ docker-compose stop ## Stop all containers

$ docker-compose stop web ## Stop single container

**restart –**

This will restart containers of the services defined in config file

$ docker-compose restart ## Restart all containers

$ docker-compose restart web ## Restart single container

**pause –**

This will pause running containers for the services defined in config file.

$ docker-compose pause ## Pause all containers

$ docker-compose pause web ## Pause single container

**unpause –**

This will start paused containers for the services defined in config file.

$ docker-compose pause ## Start all paused containers

$ docker-compose pause web ## Start single paused container

**rm –**

This will remove stopped containers for the services defined in config file.

$ docker-compose rm ## Start all paused containers

$ docker-compose pause web ## Start single paused container