Lab Manual- Setup and Manage Docker Compose for Multicontainer

Prepared for:

Date: 18th Nov 2018

Prepared by:

Document Name: Lab Manual

Document Number DevOpsLab401

Contributor:

Table of Contents

1 OBJECTIVE

Docker composes

- Tool For Defining & Running Multi-Container Docker Applications
- Use Yaml Files To Configure Application Services (Docker-Compose.Yml)
- Can Start All Services With A Single Command : Docker Compose Up
- Can Stop All Services With A Single Command : Docker Compose Down
- Can Scale Up Selected Services When Required.

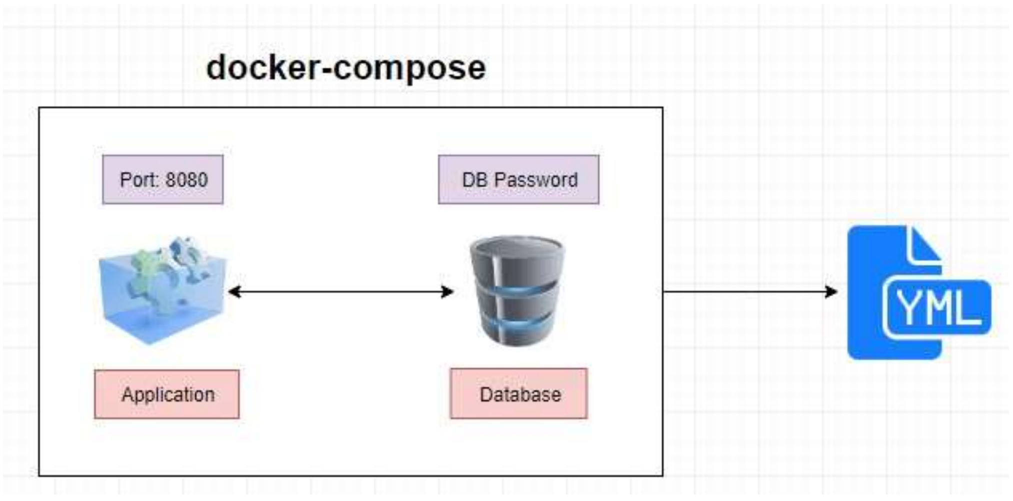
In This Lab will cover the basics of configuring docker compose with web and database services

2 PRE-REQUISISTE

- Prior knowledge of Linux
- Accounts in Docker-Hub
- A local Computer with 4 CPU, 16 GB RAM, 200 GB disk space

3 How Docker Compose Work

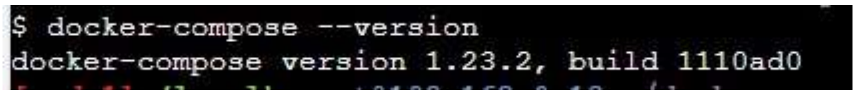
- **Docker Compose** is a tool for defining and running multi-container docker applications. With Compose, we use a YAML file to configure our application's services. And then we create and start all the services from the configuration with a single command. Here is a simple graphical illustration that shows how Docker compose works



4 Setup Up Docker Compose

Steps 1: Check the docker compose version

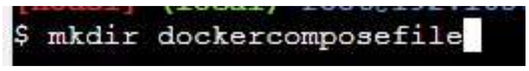
$ docker-compose --version



```
$ docker-compose --version
docker-compose version 1.23.2, build 1110ad0
```

Step 2: create a directory

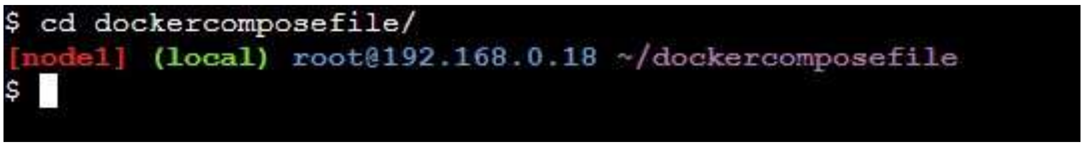
$ mkdir dockercomposefile



```
$ mkdir dockercomposefile
```

Step 3: go inside the directory

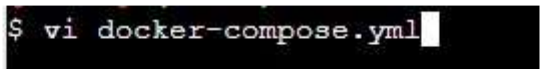
$ cd docekercomposefile/



```
$ cd dockercomposefile/
[node1] (local) root@192.168.0.18 ~/dockercomposefile
$
```

Steps 4: We are going to create simple docker compose file as an example. Here is the contents of my compose file.

vi docker-compose.yml



```
$ vi docker-compose.yml
```

Steps 5: Type below steps for docker compose

Version: ‘3’

services:

 web:

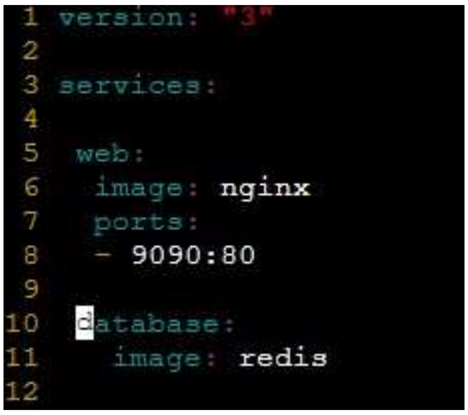
 image: nginx

ports:

- 9090:80

database:

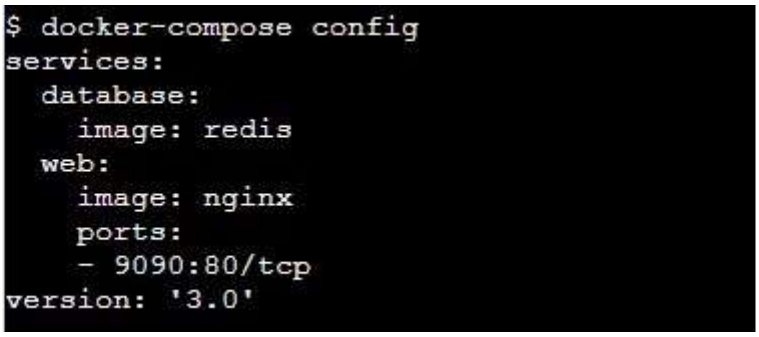
image: redis



```
version: "3"

services:

  web:
   image: nginx
   ports:
    - 9090:80

  database:
    image: redis

```

Steps 6: Type below command to check everything in docker compose is correct

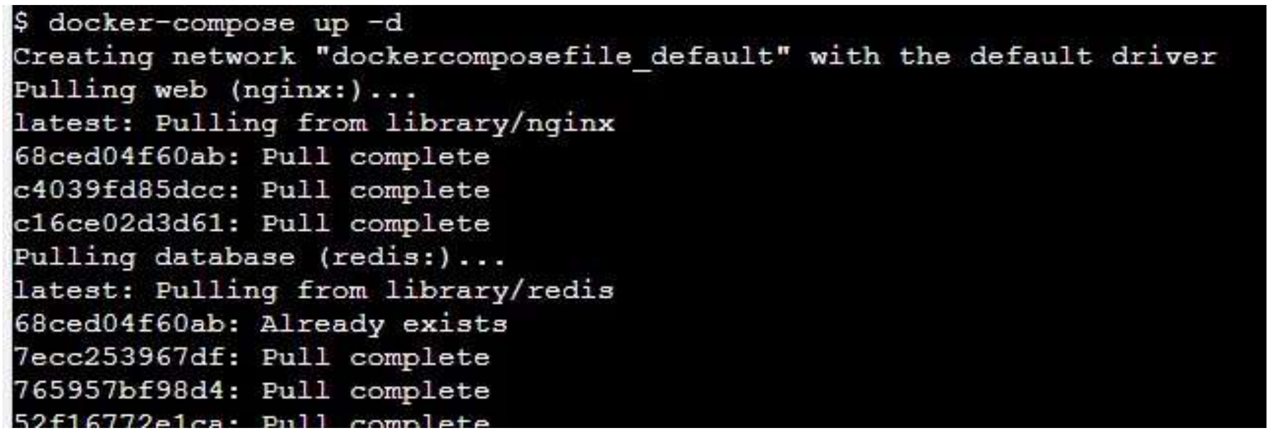
docker-compose config



```
$ docker-compose config
services:
  database:
    image: redis
  web:
    image: nginx
    ports:
    - 9090:80/tcp
version: '3.0'
```

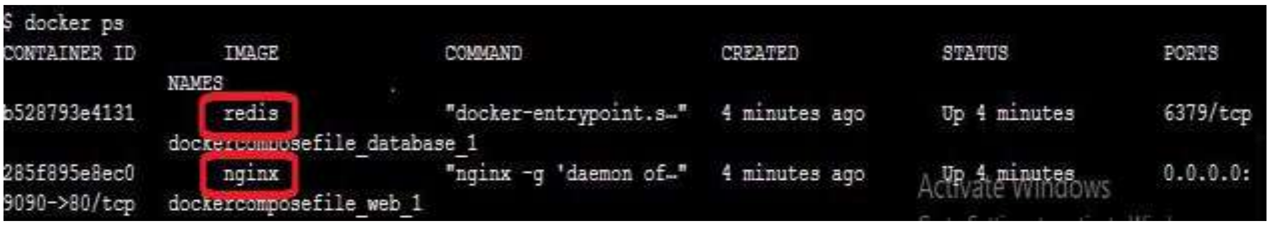
Steps 7: Now let’s run the docker compose (-d is for background)

docker-compose up -d

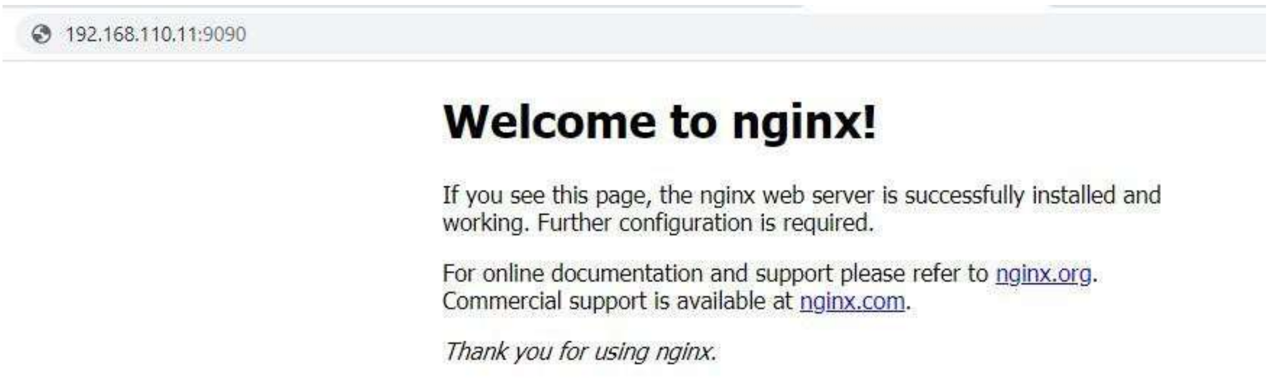

```
$ docker-compose up -d
Creating network "dockercomposefile_default" with the default driver
Pulling web (nginx:)...
latest: Pulling from library/nginx
68ced04f60ab: Pull complete
c4039fd85dcc: Pull complete
c16ce02d3d61: Pull complete
Pulling database (redis:)...
latest: Pulling from library/redis
68ced04f60ab: Already exists
7ecc253967df: Pull complete
765957bf98d4: Pull complete
52f16772e1ca: Pull complete
```

Steps 8: Now let's check the Docker ps

docker ps


```
$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS
                NAMES
b528793e4131        redis               "docker-entrypoint.s…"   4 minutes ago       Up 4 minutes        6379/tcp
                dockercomposefile_database_1
285f895e8ec0        nginx               "nginx -g 'daemon of…"   4 minutes ago       Up 4 minutes        0.0.0.0:
9090->80/tcp    dockercomposefile_web_1
```

Steps 9: Now open the browser and type the IP address with port 8080


192.168.110.11:9090

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

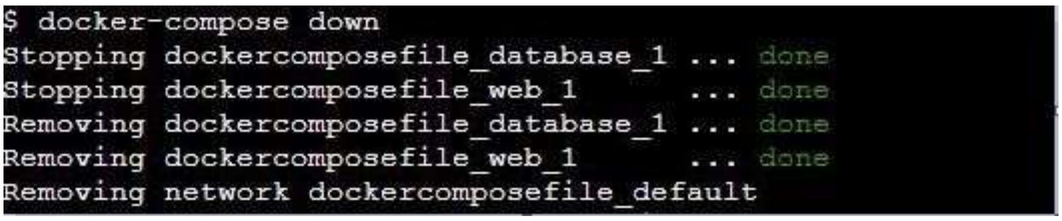
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

4.1 Manage Compose Conatiner

Steps 1: Shut down the container

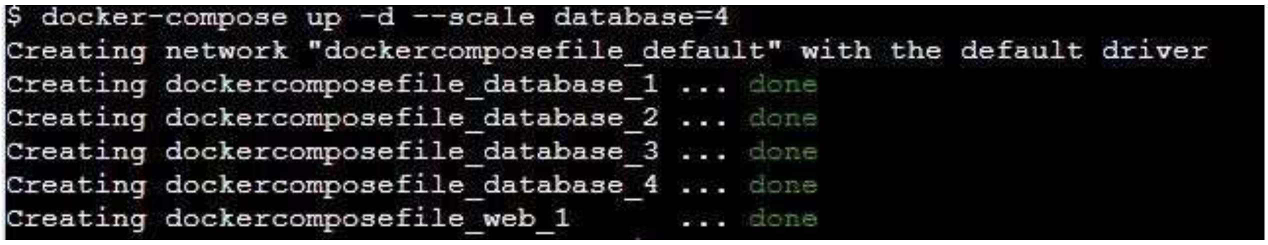
docker-compose down



```
$ docker-compose down
Stopping dockercomposefile_database_1 ... done
Stopping dockercomposefile_web_1      ... done
Removing dockercomposefile_database_1 ... done
Removing dockercomposefile_web_1      ... done
Removing network dockercomposefile_default
```

Steps 2: scale the service like database

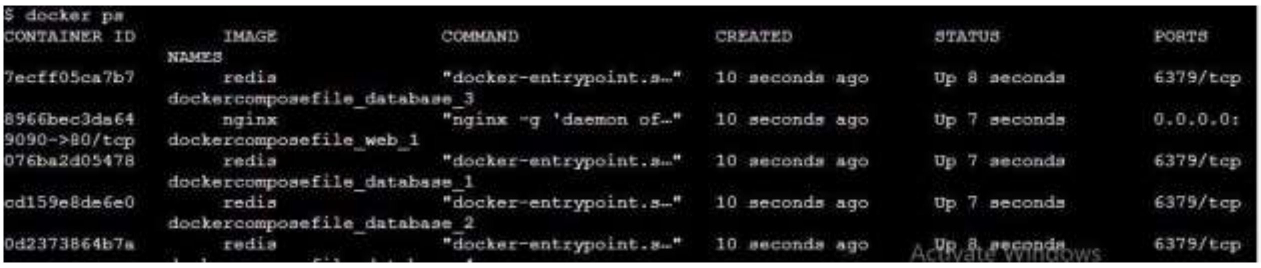
docker-compose up -d –scale database=4



```
$ docker-compose up -d --scale database=4
Creating network "dockercomposefile_default" with the default driver
Creating dockercomposefile_database_1 ... done
Creating dockercomposefile_database_2 ... done
Creating dockercomposefile_database_3 ... done
Creating dockercomposefile_database_4 ... done
Creating dockercomposefile_web_1      ... done
```

Steps 3: check the no of running container

docker ps



```
$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS
                NAMES
7ecff05ca7b7        redis               "docker-entrypoint.s…"   10 seconds ago      Up 8 seconds        6379/tcp
                dockercomposefile_database_3
8966bec3da64        nginx               "nginx -g 'daemon of…"   10 seconds ago      Up 7 seconds        0.0.0.0:
9090->80/tcp    dockercomposefile_web_1
076ba2d05478        redis               "docker-entrypoint.s…"   10 seconds ago      Up 7 seconds        6379/tcp
                dockercomposefile_database_1
cd159e8de6e0        redis               "docker-entrypoint.s…"   10 seconds ago      Up 7 seconds        6379/tcp
                dockercomposefile_database_2
0d2373864b7a        redis               "docker-entrypoint.s…"   10 seconds ago      Up 8 seconds        6379/tcp
```

Steps 4: down the container

Docker-compose down