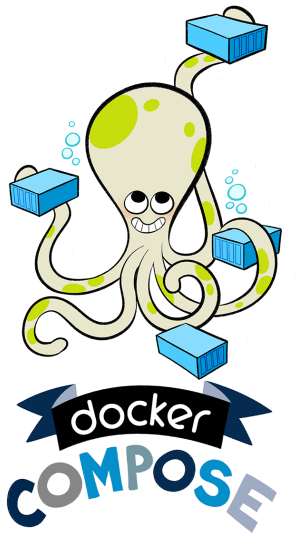
Docker Compose - Hands On



We are going to run a Python-based web application that will be accessing a Redis NoSQL Database.

What this means is that we will be having a multi-container application with two services, one for the web application and the other for Redis. The web service will be dependent on the Redis service.

Let's get started.

Docker Images

First up, let's talk about the Docker images for the two containers:

Redis Docker Image

The Redis Image will be the standard redis image from the Docker Hub. You can pull that down anytime using the $ docker pull redis command.

Web Docker Image

Our Web Application Docker image is going to be comprised of a Python Flask application.

Before we build the image, we will need to get some files in place. Follow the steps given below:

1. Create a folder named **compose-app** in your machine.
2. Copy the app.py, requirements.txt and Dockerfile as provided.

Now build out the web image via the Docker build command as given below:

```
$ docker build -t web .
```

This will build out the **web** image and one the process is complete, you can view that via the `docker images` command.

Docker Compose file

Let us take a look at the docker-compose.yml file that has been provided.

```
version: '2'
services:
      web:
        build: .
        ports:
          - "5000:5000"
        depends_on:
          - redis
      redis:
        image: redis
        ports:
          - "6379:6379"
```

Understand the compose file and then go ahead to run the composite application via the docker-compose tool.

Running the Multi-container application via docker-compose

Assuming that you had installed the Docker Toolbox, you can go to the folder that has the docker-compose.yml file and simply give the following command:

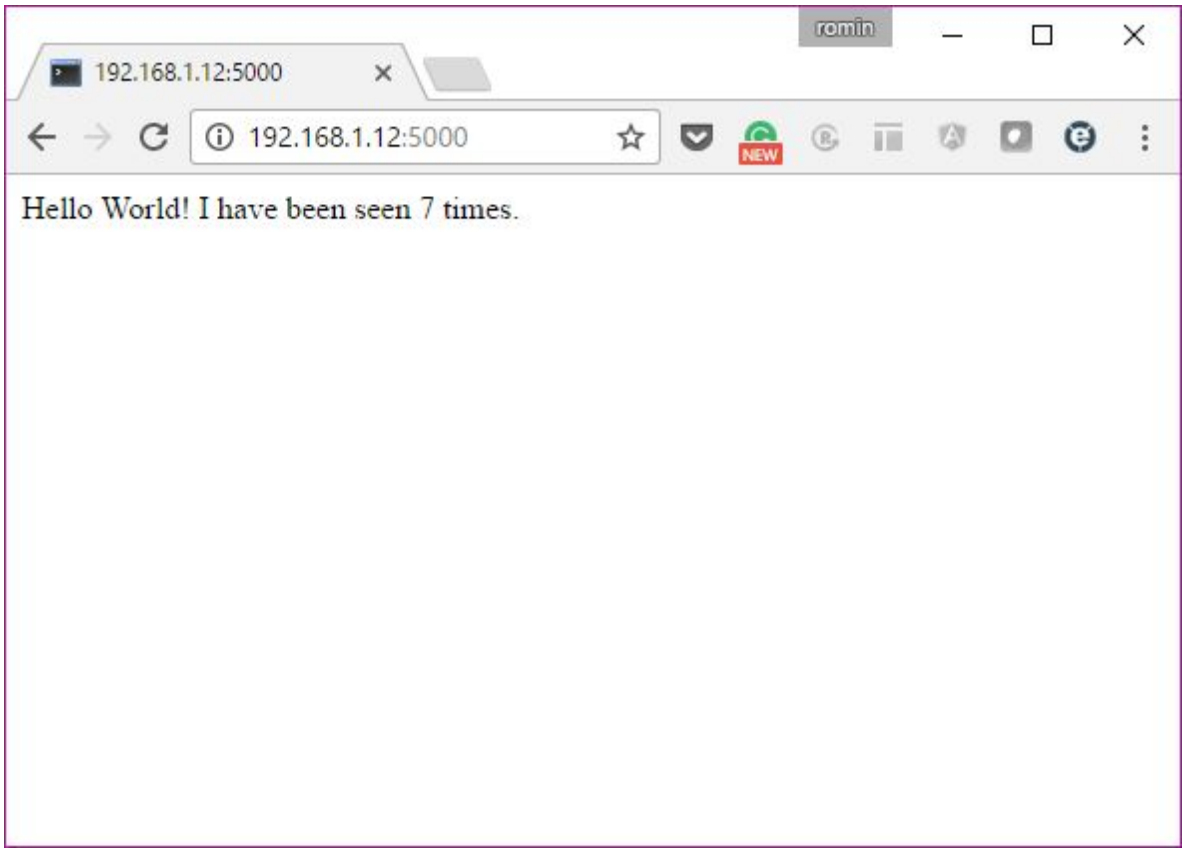
```
$ docker-compose up
```

This will build the web image, download the redis image if it is not present locally and then launch the two services. A partial output is shown below that indicates that the two services have started.

```
Creating compose_redis_1
Creating compose_web_1
Attaching to compose_redis_1, compose_web_1
redis_1  | 1:C 08 Dec 11:07:21.367 # Warning: no config file specified, using the
default config. In order to specify a config file use redis-server /path/to/redis.conf
redis_1  |                 _.-
```

```
web_1    |  * Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
redis_1  |           _.-``__ ''-._
redis_1  |      _.-``    `.  `_.  ''-._           Redis 3.2.6 (00000000/0) 64 bit
redis_1  |  .-``  .-```.  ```\/    _.,_ ''-._
redis_1  | (    '      ,       .-`  | `,    )     Running in standalone mode
redis_1  | |`-._`-...-` __...-.``-._|'` _.-'|     Port: 6379
redis_1  | |    `-._   `._    /     _.-'    |     PID: 1
redis_1  |  `-._    `-._  `-./  _.-'    _.-'
redis_1  | |`-._`-._    `-.__.-'    _.-'_.-'|
redis_1  | |    `-._`-._        _.-'_.-'    |           http://redis.io
redis_1  |  `-._    `-._`-.__.-'_.-'    _.-'
redis_1  | |`-._`-._    `-.__.-'    _.-'_.-'|
redis_1  | |    `-._`-._        _.-'_.-'    |
redis_1  |  `-._    `-._`-.__.-'_.-'    _.-'
redis_1  |      `-._    `-.__.-'    _.-'
redis_1  |          `-._        _.-'
redis_1  |              `-.__.-'
redis_1  |
redis_1  | 1:M 08 Dec 11:07:21.369 # WARNING: The TCP backlog setting of 511 cannot be
enforced because /proc/sys/net/core/somaxconn is set to the lower value of 128.
redis_1  | 1:M 08 Dec 11:07:21.369 # Server started, Redis version 3.2.6
redis_1  | 1:M 08 Dec 11:07:21.369 # WARNING overcommit_memory is set to 0! Background
save may fail under low memory condition. To fix this issue add 'vm.overcommit_memory =
1' to /etc/sysctl.conf and then reboot or run the command 'sysctl
vm.overcommit_memory=1' for this to take effect.
web_1    |  * Restarting with stat
web_1    |  * Debugger is active!
web_1    |  * Debugger pin code: 717-199-826
```

Now, if you know the IP address of the docker machine, you can simply hit that web page in the browser on port 5000 as shown below:



If you see the `docker-compose up` console window, you will see the HTTP Trace. A sample output from the run is shown below:

```
web_1    | 192.168.1.3 - - [08/Dec/2016 11:27:14] "GET / HTTP/1.1" 200
-
web_1    | 192.168.1.3 - - [08/Dec/2016 11:28:15] "GET / HTTP/1.1" 200
-
```

References

If you would like to learn more about Docker Compose, refer to the following link:

- Official Docker Compose page
- Docker Compose file reference
- Docker Compose command-line reference