

Docker compose

Part - 1

1. Create the following files inside a directory in your local machine
 - a. app.py - this is for the Flask App that you want to run
 - b. requirements.txt - this contains the requirements, in this case flask and redis
 - c. Dockerfile - to setup the flask environment
 - d. docker-compose.yml - creating two services namely web for flask application and another redis for redis image to store the counter
2. Run 'docker-compose build' to build the containers specified in the docker-compose file
3. Run 'docker-compose up' to start the container
4. Navigate to <http://localhost:5000> to check the webserver

```
krutik@Quantiphi-912: ~/Desktop/Quantiphi Training/Containerization/Q2_Flask_Redis_Counter
File Edit View Search Terminal Help
krutik@Quantiphi-912:~/Desktop/Quantiphi Training/Containerization/Q2_Flask_Redis_Counter$ sudo docker-compose up
Starting q2_flask_redis_counter_redis_1 ... done
Starting q2_flask_redis_counter_web_1 ... done
Attaching to q2_flask_redis_counter_redis_1, q2_flask_redis_counter_web_1
redis_1 | 1:c 31 Jul 2019 07:16:41.263 # 000000000000 Redis is starting 000000000000
redis_1 | 1:c 31 Jul 2019 07:16:41.263 # Redis version=5.0.5, bits=64, commit=00000000, modified=0, pid=1, just started
redis_1 | 1:c 31 Jul 2019 07:16:41.263 # Warning: no config file specified, using the default config. In order to specify a config file use r
redis-server /path/to/redis.conf
redis_1 | 1:M 31 Jul 2019 07:16:41.267 * Running mode=standalone, port=6379.
redis_1 | 1:M 31 Jul 2019 07:16:41.267 # WARNING: The TCP backlog setting of 511 cannot be enforced because /proc/sys/net/core/somaxconn is s
et to the lower value of 128.
redis_1 | 1:M 31 Jul 2019 07:16:41.267 # Server initialized
redis_1 | 1:M 31 Jul 2019 07:16:41.267 # WARNING overcommit_memory is set to 0! Background save may fail under low memory condition. To fix t
his 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 ta
ke effect.
redis_1 | 1:M 31 Jul 2019 07:16:41.267 # WARNING you have Transparent Huge Pages (THP) support enabled in your kernel. This will create laten
cy and memory usage issues with Redis. To fix this issue run the command 'echo never > /sys/kernel/mm/transparent_hugepage/enabled' as root, a
nd add it to your /etc/rc.local in order to retain the setting after a reboot. Redis must be restarted after THP is disabled.
web_1 | * Serving Flask app "app" (lazy loading)
web_1 | * Environment: production
web_1 | WARNING: This is a development server. Do not use it in a production deployment.
redis_1 | 1:M 31 Jul 2019 07:16:41.268 * DB loaded from disk: 0.000 seconds
redis_1 | 1:M 31 Jul 2019 07:16:41.268 * Ready to accept connections
web_1 | Use a production WSGI server instead.
web_1 | * Debug mode: on
web_1 | * Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
web_1 | * Restarting with stat
web_1 | * Debugger is active!
web_1 | * Debugger PIN: 173-209-527
web_1 | 172.18.0.1 - - [31/Jul/2019 07:16:57] "GET / HTTP/1.1" 200 -
web_1 | 172.18.0.1 - - [31/Jul/2019 07:16:57] "GET / HTTP/1.1" 200 -
web_1 | 172.18.0.1 - - [31/Jul/2019 07:16:57] "GET / HTTP/1.1" 200 -
web_1 | 172.18.0.1 - - [31/Jul/2019 07:17:07] "GET / HTTP/1.1" 200 -
web_1 | 172.18.0.1 - - [31/Jul/2019 07:17:07] "GET / HTTP/1.1" 200 -
web_1 | 172.18.0.1 - - [31/Jul/2019 07:17:08] "GET / HTTP/1.1" 200 -
web_1 | 172.18.0.1 - - [31/Jul/2019 07:17:08] "GET / HTTP/1.1" 200 -
web_1 | 172.18.0.1 - - [31/Jul/2019 07:17:08] "GET / HTTP/1.1" 200 -
web_1 | 172.18.0.1 - - [31/Jul/2019 07:17:08] "GET / HTTP/1.1" 200 -
```

Terminal Running Containers

```
← → ↺ ① localhost:5000 ☆ 🚩 📄 👤 ⋮
Hello World! I have been seen b'196' times.
```

Web Server with hit counts

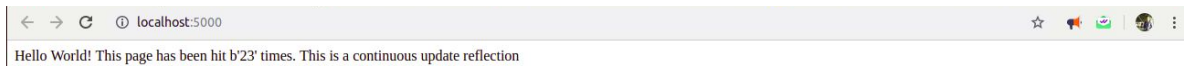
As there is no volume attached to the containers, any changes made won't reflect unless you rebuild the docker image.

Part - 2

1. To attach a volume to the container, add volumes in the docker-compose.yml file with the mapping of directories
2. Now run 'docker-compose build' to rebuild the images with the added volumes
3. To start the containers in daemon (to later edit files in same terminal) run 'docker-compose up -d' or run 'docker-compose up' to run the containers in normal mode
4. Now edit the file in the container using 'docker exec -it [container-id] bash' and then using any in terminal text editor as nano or vim
5. Or you can edit in the files that are locally stored in your machine as the volumes are attached and would reflect the changes even from the outside



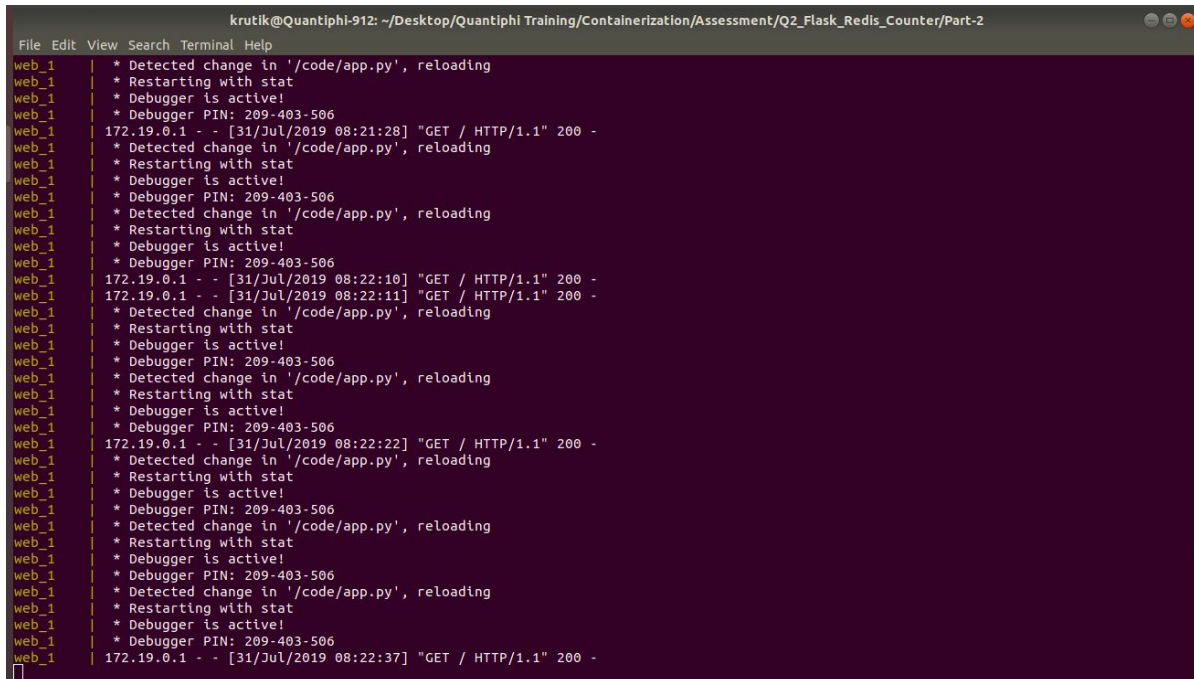
Initial build page



Updated page without rebuilding image



Another update



Terminal showing changes detected in working code as volume is attached

This is a host volume type of volume attached to the container to reflect changes from local source to inside of the container.

Another method to do the same is by attaching an external volume. For the same you need to first create a volume and then attach the same.

You can do this by adding this volumes specification in your required service -

volumes:

- [name-of-volume]:/your/local/path

And then add this at the end of the docker-compose file to mention the source of the mounted volume

volumes:

- [name-of-volume]:
 - external: false

external true if you provide your volume from an external source, not directly from the docker-compose spec