

Instituto Federal de Educação Ciência e Tecnologia do Ceará
Departamento de Telemática - Sistemas Operacionais de Rede

Emanoel Silva de Sousa
Edesson da Silva Mendes

Avaliação 09 - Docker Compose

O Compose é uma ferramenta para definir e executar aplicativos Docker de vários contêineres. Através do compose, você usa o arquivo YAML para configurar os serviços de seu aplicativo, com isso com um único comando, você cria e inicia todos os serviços de sua configuração.

O Docker Compose é utilizado justamente para facilitar o provisionamento e gerenciamento de multi-contêineres principalmente em ambientes de desenvolvimento, testes automatizados ou cenários de execução em um único host.

Ele também ainda realiza o isolamento do ambiente de um conjunto de contêineres separando-os por nome de projeto, que geralmente é dado o nome do local onde ele está sendo executado.

Recursos

Os principais recursos que torna o Docker Compose eficaz:

Vários ambientes isolados em um único host.

Preserva os dados do volume quando os contêineres são criados.

Recrie apenas contêineres que foram alterados.

Suportam variáveis e movem uma composição entre ambientes.

Serviços

O Docker Compose trata todos os contêineres que desejamos executar como serviços e é dessa forma que devemos referenciá-los no arquivo de configuração. Tratando os contêineres dessa forma, o Docker Compose consegue atribuir o mesmo conjunto de configurações para todos os contêineres que fizerem parte deste serviço.

Estrutura do arquivo

Para executar nossos contêineres com o Compose é necessário possuir um arquivo YAML que contenha todas as informações e parâmetros que desejamos passar para a execução deles. Por padrão, os comandos docker-compose procuram um arquivo no diretório corrente nomeado como docker-compose.yml, porém é possível indicar um outro nome de arquivo e também em um outro local passando o parâmetro -f.

version: '3.3': versão utilizada do Docker Compose, muda de acordo com a versão.

services: inicia um serviço e tudo abaixo dessa linha faz parte deste serviço.

image: imagem do container.

volumes: volume que é utilizado no container.

environment: variáveis de ambiente utilizadas para acessar o container.

Docker Compose - MySql e Adminer

Para a simulação foi utilizado o sistema Ubuntu. O arquivo **docker-compose.yml** será criado e pode ser salvo em uma pasta qualquer. O comando **docker-compose up -d** cuidará da criação da network e dos containers esperados.

A seguir está o conteúdo do arquivo **docker-compose.yml**:

```
edesson@edesson: ~/Desenvolvimento/AmbienteMySQL
GNU nano 4.8                                docker-compose.yml
Version: '3.3'

services:
  mysqlsrv:
    image: mysql:5.7
    environment:
      MYSQL_ROOT_PASSWORD: "MySql2019!"
      MYSQL_DATABASE: "testedb"
    ports:
      - "3306:3306"
    volumes:
      - /home/edesson/Desenvolvimento/Docker/Volumes/MySQL:/var/lib/mysql
    networks:
      - mysql-compose-network

  adminer:
    image: adminer
    ports:
      - 8080:8080
    networks:
      - mysql-compose-network

networks:
  mysql-compose-network:
    driver: bridge
```

O serviço **mysqrsrv** é a instância do MySQL 5.7 que vai ser criada para acesso na porta 3306. O serviço **adminer** é o container que vai permitir a execução do ADMINER na porta 8080.

As imagens que foram referenciadas no arquivo, serão baixadas caso não existam.

Foi especificado um volume para o **mysqrsrv**, onde serão salvos os arquivos de dados no Ubuntu: (/home/edesson/Desenvolvimento/Docker/Volumes/MySQL).

Por meio da network **mysql-compose-network** acontecerá a comunicação entre os containers **mysqrsrv** e **adminer**.

A seguir o resultado do comando **sudo docker-compose up -d**:

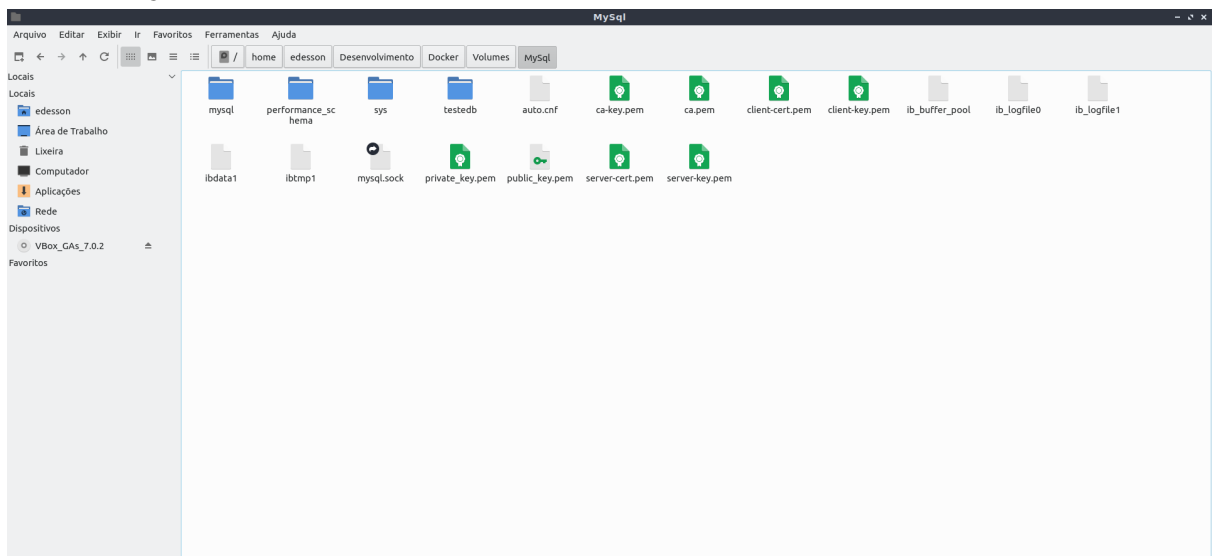
```
edesson@edesson: ~/Desenvolvimento/AmbienteMySQL
edesson@edesson:~/Desenvolvimento/AmbienteMySQL$ sudo docker-compose up -d
[sudo] senha para edesson:
Creating network "ambientesql_mysql-compose-network" with driver "bridge"
Pulling mysqrsrv (mysql:5.7)...
5.7: Pulling from library/mysql
d96bccd7291f: Pull complete
feae47af51d5: Pull complete
a64d3809fa31: Pull complete
da689aaabebf: Pull complete
6a2a532732de: Pull complete
9063a4855045: Pull complete
625a84b6ad66: Pull complete
7084e7269f30: Pull complete
161795475ff3: Pull complete
6b2f7a84c8fc: Pull complete
cf120096e55e: Pull complete
Digest: sha256:4149a92977a5dd27cbd6f81cca3817e6278a844d566b45f9ff1908bb2714b1ca
Status: Downloaded newer image for mysql:5.7
Pulling adminer (adminer)...
latest: Pulling from library/adminer
ca7dd9ec2225: Pull complete
1078b4fe0ca1: Pull complete
9d6840f2a28f: Pull complete
0e2e66b89284: Pull complete
3b1be5f02bec: Pull complete
96243f515dda: Pull complete
4006f70ca99d: Pull complete
3c76b37a5dd7: Pull complete
6491a80416fc: Pull complete
ff1b30214647: Pull complete
e447fcd9b764: Pull complete
3814f2022577: Pull complete
e2bc4f3bd070: Pull complete
be31139c5821: Pull complete
d183076dd611: Pull complete
Digest: sha256:3b4e25b39404729b27bbb2895fa0a8fe9ec19cdadife85c85500f4c080c7f4
Status: Downloaded newer image for adminer:latest
Creating ambientesql_mysqlsrv_1 ... done
Creating ambientesql_adminer_1 ... done
edesson@edesson:~/Desenvolvimento/AmbienteMySQL$
```

O comando **docker-compose ps** ps mostrará que os containers do MySQL (porta 3306) e do Adminer (porta 8080) foram gerados corretamente e se encontram em

execução. O comando **docker network ls** mostra que a rede mysql-compose-network foi criada com sucesso (como ambientemysql_mysql-compose-network, resultado da concatenação com o nome do diretório em que se encontra o arquivo docker-compose.yml).

```
edesson@edesson: ~/Desenvolvimento/AmbienteMySQL$ docker-compose ps
-----
Name                                Command                                State      Ports
-----
ambientemysql_adminer_1             entrypoint.sh docker-php-e ...       Up         0.0.0.0:8080->8080/tcp,:::8080->8080/tcp
ambientemysql_mysqlsrv_1             docker-entrypoint.sh mysqld          Up         0.0.0.0:3306->3306/tcp,:::3306->3306/tcp, 33060/tcp
edesson@edesson:~/Desenvolvimento/AmbienteMySQL$ docker network ls
-----
NETWORK ID    NAME                                DRIVER    SCOPE
5fcea046b0c09 ambientemysql_mysql-compose-network bridge    local
39d45ac98c0c  bridge                             bridge    local
70f0f742258c  host                               host      local
a855a5bc3788  null                               null      local
edesson@edesson:~/Desenvolvimento/AmbienteMySQL$
```

A seguir arquivos e diretórios que foram criados para o volume definidos no arquivo **docker-compose.yml**:



Testando o ambiente:

Testando o Adminer via localhost. Informar o nome do container do MySQL como servidor e a senha indicada no MYSQL_ROOT_PASSWORD do arquivo docker-compose.yml.

Entrar - Adminer — Mozilla Firefox

Entrar - Adminer

localhost:8080

Idioma: Português (Brazil) ▾

Adminer 4.8.1

Entrar

Sistema	MySQL ▾
Servidor	mysqlsrv
Usuário	root
Senha	●●●●●●●●
Base de dados	

Entrar ☐ Login permanente

Concluído o processo de login, aparecerá o banco de dados testedb (também especificado em docker-compose.yml, através da variável de ambiente MYSQL_DATABASE) como uma das opções disponíveis:

Selecionar Base de dados - mysqlsrv - Adminer — Mozilla Firefox

Selecionar Base de dados

Idioma: Português (Brazil) ▾ MySQL » mysqlsrv Sair

Adminer 4.8.1

DB: ▾

[Comando SQL](#) [Importar](#) [Exportar](#)

[Criar Base de dados](#) [Privilégios](#) [Lista de processos](#) [Variáveis](#) [Estado](#)

Versão MySQL: 5.7.40 através da extensão PHP MySQLi

Logado como: root@172.20.0.2

	Base de dados - Atualizar	Colaço	Tabelas	Size - Compute
<input type="checkbox"/>	information_schema	utf8_general_ci	?	?
<input type="checkbox"/>	mysql	latin1_swedish_ci	?	?
<input type="checkbox"/>	performance_schema	utf8_general_ci	?	?
<input type="checkbox"/>	sys	utf8_general_ci	?	?
<input type="checkbox"/>	testedb	latin1_swedish_ci	?	?

Selected (0)

Apagar