

Documento para rodar tudo em apenas um container ou rodar cada servidor e cliente em containers diferentes. Na apresentação utilizamos apenas um container, pois para usar outros containers teríamos que arrumar uma forma de compartilhar o nosso arquivo servers_online.txt. A nossa imagem do Docker foi criada com o projeto padrão, com esse arquivo em branco como se nenhum server ainda estivesse de pé. Para esse teste podemos criar muitos containers e em cada um deles abrir manualmente o arquivo e preencher os servidores já de pé. Fizemos assim para simular:


- cria um container e executa o servidor dele. A porta 7000 será inserida no servers_online.txt
- cria o segundo container, abre o servers_online.txt e preenche com a 7000:0 para dizer que já temos um aberto nessa porta. Executa o server que abrirá na porta 7001 preenchendo o arquivo.
- Fazer isso até a quantidade de servidores desejados.
- O(s) clientes rodam normalmente sem precisar alterar algo manualmente.

Chegamos a usar o DockerHub para armazenar nossa imagem, mas depois da entrega 3 começamos a ter problemas de timeout ao enviar a imagem para lá:

<https://hub.docker.com/repository/docker/trocadilho/trocadilhosistemasdistribuido>

Nesse mesmo documento da entrega 2 que está na master usávamos o DockerHub, dando um pull (sudo docker pull trocadilho/trocadilhosistemasdistribuido:firsttry) na imagem e depois bastava subir o(s) container(s). Sem ele precisamos baixar o projeto e buildar com o DockerFile.

 trocadilho / trocadilhosistemasdistribuido

This repository does not have a description 

 Last pushed: an hour ago

Docker commands

[Public View](#)

To push a new tag to this repository,

```
docker push
trocadilho/trocadilhosistemasdistribuido:tagname
```

FAZER TUDO RODAR EM UM SÓ CONTAINER

Baixar o projeto da branch entrega 3:

```
git clone -b entrega3 https://github.com/Fziliotti/TrocadilhoSistemasDistribuido
```

Entrar no diretório:

```
cd TrocadilhoSistemasDistribuido/
```

Criar a imagem:

```
sudo docker build -t trocadilho/server:1.0 .
```

Ver se deu tudo certo:

```
sudo docker images
```

```
eduardo@NB-SBDEV-ELDOC ~/documento-docker/TrocadilhoSistemasDistribuido $ sudo docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
trocadilho/server   1.0                f50e0ab1b909       6 minutes ago      303MB
```

Criar um container:

```
sudo docker run -it IMAGE-ID /bin/bash
```

OBS: IMAGE-ID é o IMAGE ID quando você executa o comando “sudo docker images”.

```
eduardo@NB-SBDEV-ELDOC ~/documento-docker/TrocadilhoSistemasDistribuido $ sudo docker run -it f50e0ab1b909 /bin/bash
bash-4.4# ls -lh
total 38156
-rw-r--r-- 1 root root 1.7K Dec 23 01:45 Dockerfile
-rw-r--r-- 1 root root 1.6K Dec 23 01:45 README.md
-rw-r--r-- 1 root root 18.2M Dec 23 01:48 client.jar
-rw-r--r-- 1 root root 72 Dec 23 01:45 constants.txt
-rw-r--r-- 1 root root 33.3K Dec 23 01:45 log-structured-merge-trees.pdf
-rw-r--r-- 1 root root 6.3K Dec 23 01:45 pom.xml
-rw-r--r-- 1 root root 627.4K Dec 23 01:45 projeto-com-docker.pdf
-rw-r--r-- 1 root root 903 Dec 23 01:45 run.py
-rw-r--r-- 1 root root 18.2M Dec 23 01:48 server.jar
-rw-r--r-- 1 root root 0 Dec 23 01:48 servers_online.txt
drwxr-xr-x 3 root root 4.0K Dec 23 01:45 src
drwxr-xr-x 8 root root 4.0K Dec 23 01:48 target
-rw-r--r-- 1 root root 265 Dec 23 01:45 test-client.sh
-rw-r--r-- 1 root root 262 Dec 23 01:45 test-server.sh
-rw-r--r-- 1 root root 122.9K Dec 23 01:48 trocadilhos-1.0-SNAPSHOT.jar
bash-4.4#
```

Já dentro do terminal do container posso rodar o servidor. O Jar dele é o “server.jar”.
java -jar server.jar

```
bash-4.4# java -jar server.jar
```

```
Starting Atomix server [0] on port 7000
Snapshot created! id:1
Starting grpcServer...
Server [0] started on port 7000
```

O servidor subiu.

Abrindo outro terminal posso ver a lista dos containers ativos, o que acabei de subi. Comando:
sudo docker ps

```
eduardo@NB-SBDEV-ELDOC ~ $ sudo docker ps
[sudo] password for eduardo:
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS
072403f0673e   f50e0ab1b909  "/bin/bash"             2 minutes ago   Up 2 minutes   8000/tcp
```

Agora posso entrar no container criado e subir quantos servidores forem necessários de acordo com o SERVERS_QUANTITY e CLUSTER_SIZE do arquivo contants.txt localizado na raiz do projeto.

```
1  SERVERS_QUANTITY=3
2  CLUSTER_SIZE=3
3  BASE_PORT=7000
4  INTERVAL_TO_SNAPSHOT=30
```

sudo docker exec -i -t CONTAINER_ID /bin/bash

OBS: CONTAINER_ID é o id do container que estou rodando, pode ser verificado com o comando “sudo docker ps”.

```

eduardo@NB-SBDEV-ELDOC ~ $ sudo docker ps
[sudo] password for eduardo:
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS
072403f0673e       f50e0ab1b909      "/bin/bash"        2 minutes ago      Up 2 minutes       8000/tcp
eduardo@NB-SBDEV-ELDOC ~ $ sudo docker exec -i -t 072403f0673e /bin/bash
bash-4.4#

```

Abrir um novo servidor:

```
java -jar server.jar
```

```

Starting Atomix server [1] on port 7001
Snapshot created! id:1
Starting grpcServer...
Server [1] started on port 7001

```

Aberto agora na porta 7001.

Com esse mesmo procedimentos podemos abrir quantos clientes quisermos. Abrir outro terminar e executar:

```
sudo docker exec -i -t CONTAINER_ID /bin/bash
```

Rodar o cliente:

```
java -jar client.jar
```

```

-----Bem vindo ao Rei dos Trocadilhos.-----
1(listar) - Buscar um trocadilho pelo código
2(criar)  - Criar um trocadilho
3(editar) - Editar um trocadilho
4(deletar) - Deletar um trocadilho
9(sair)   - Sair

Digite o número ou escreva a opção desejada:

```

RODAR SERVIDOR E CLIENTES EM CONTAINERS SEPARADOS

Baixar o projeto da branch entrega 3:

```
git clone -b entrega3 https://github.com/Fziliotti/TrocadilhoSistemasDistribuido
```

Entrar no diretório:

```
cd TrocadilhoSistemasDistribuido/
```

Criar a imagem:

```
sudo docker build -t trocadilho/server:1.0 .
```

Ver se deu tudo certo:

```
sudo docker images
```

```

eduardo@NB-SBDEV-ELDOC ~/documento-docker/TrocadilhoSistemasDistribuido $ sudo docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
trocadilho/server    1.0                f50e0ab1b909       6 minutes ago      303MB

```

Criar um container:

```
sudo docker run -it --network="host" IMAGE_ID /bin/bash
```

Tivemos que usar o “network=“host” para criar todos os containers na rede da máquina local que estão sendo executados, compartilhando a mesma rede.

```

eduardo@NB-SBDEV-ELDOC ~ $ sudo docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS
072403f0673e       f50e0ab1b909       "/bin/bash"        13 minutes ago     Up 13 minutes      8000/tcp
eduardo@NB-SBDEV-ELDOC ~ $ sudo docker run -it --network="host" f50e0ab1b909 /bin/bash
bash-4.4#

```

Subir o servidor:

```
java -jar server.jar
```

```

Starting Atomix server [0] on port 7000
Snapshot created! id:1
Starting grpcServer...
Server [0] started on port 7000

```

Para criar outros containers e subir os servidores neles é necessário alterar o arquivo `servers_online.txt` manualmente, conforme informei no início desse documento. O primeiro server subiu automaticamente na porta 7000.

Criar outro container compartilhando a rede local:

```
sudo docker run -it --network="host" IMAGE_ID /bin/bash
```

Editar o `servers_online.txt` informando em quais portas já temos servidores de pé. Nesse exemplo só subimos o primeiro na 7000, portanto:

```

Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda
7000:0
~

```

Salvar e subir o server:

```
java -jar server.jar
```

```

Starting Atomix server [1] on port 7001
Snapshot created! id:1
Starting grpcServer...
Server [1] started on port 7001

```

Subiu automaticamente na porta 7001. No próximo container devemos alterar o arquivo colocando:
7000:0
7001:0

Ele subirá na 7002:0

Agora podemos subir quantos servidores forem necessários de acordo com o `SERVERS_QUANTITY` e `CLUSTER_SIZE` do arquivo `contants.txt` localizado na raiz do projeto.

```
1  SERVERS_QUANTITY=3
2  CLUSTER_SIZE=3
3  BASE_PORT=7000
4  INTERVAL_TO_SNAPSHOT=30
```

Criar outro container para rodar o(s) cliente(s):
Abrir outro terminal.

Rodar o cliente:

`sudo docker run -it --network="host" IMAGE_ID /bin/bash`

Não esquecer de colocar o `--network="host"` para ser criado na mesma rede dos servers.

```
eduardo@NB-SBDEV-ELDOC ~ $ sudo docker run -it --network="host" f50e0ab1b909 /bin/bash
bash-4.4# java -jar client.jar
```

```
-----Bem vindo ao Rei dos Trocadilhos.-----
1(listar) - Buscar um trocadilho pelo código
2(criar) - Criar um trocadilho
3(editar) - Editar um trocadilho
4(deletar) - Deletar um trocadilho
9(sair) - Sair
Localizar
Digite o número ou escreva a opção desejada:
```

Posso criar e rodar quantos containers clientes quiser, abrindo outros terminais e criando os containers. Se conectarão aos servidores que sobem na localhost. Como o comando `"docker ps"` podemos ver os containers que estão rodando:

```
eduardo@NB-SBDEV-ELDOC ~ $ sudo docker ps
[sudo] password for eduardo:
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS
bclfbfe2c210   f50e0ab1b909  "/bin/bash"             26 seconds ago Up 25 seconds
f84648c5a900   f50e0ab1b909  "/bin/bash"             6 minutes ago  Up 6 minutes
c8796258845e   f50e0ab1b909  "/bin/bash"             10 minutes ago Up 10 minutes
1fb9afa66f6d   f50e0ab1b909  "/bin/bash"             15 minutes ago Up 15 minutes
072403f0673e   f50e0ab1b909  "/bin/bash"             29 minutes ago Up 29 minutes
eduardo@NB-SBDEV-ELDOC ~ $
```

Cinco containers foram criados, um se comunicando com o outro.