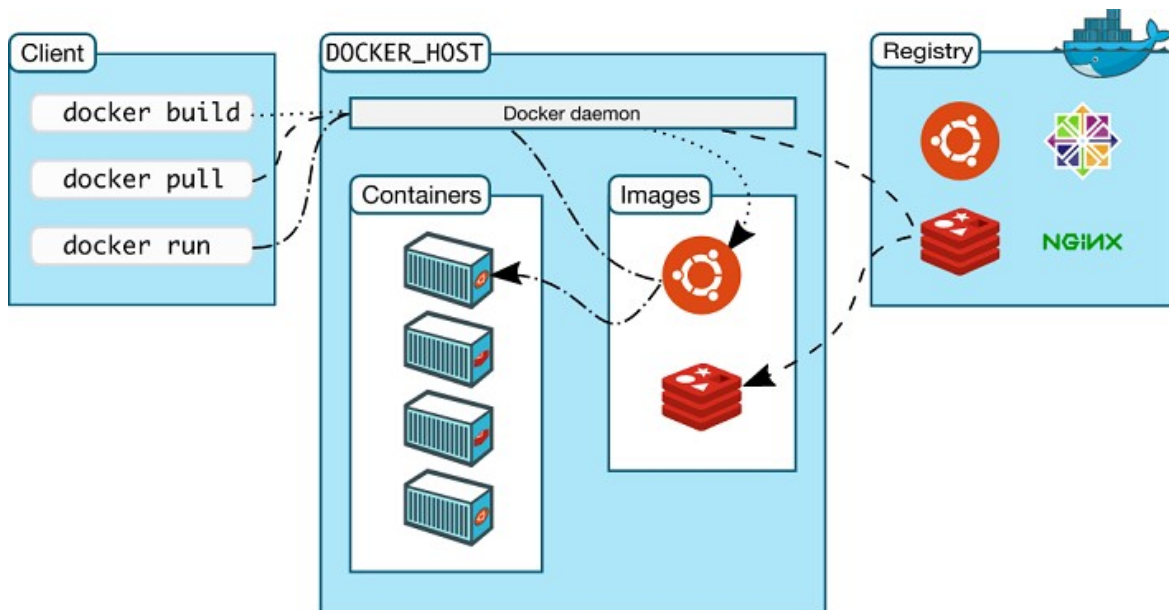


Card 17 - Docker e Containers para Aplicações (III)

Leonardo José Reis Pinto

1. Introdução

O Docker é um software open source que cria ambientes para aplicações, de forma que uma aplicação pode rodar em qualquer máquina, pois ele cria um ambiente (contêiner) com todas as dependências necessárias para rodar uma aplicação. Tópicos importantes do Docker: imagem, são modelos prontos para criar contêineres, como se fosse um molde que fizesse várias rosquinhas com as especificações do molde. Tem os contêineres, que são os ambientes criados a partir de uma imagem para determinada aplicação. Imagine uma caixa fechada com o código da aplicação, com as bibliotecas da aplicação, em que o ambiente externo não vai interferir no ambiente interno e tudo vai rodar tranquilamente. O Dockerfile é um arquivo que contém as instruções para a criação de imagens (molde) do jeito que o usuário quiser, e, por fim, o Docker Hub, um ambiente no qual tem a maioria das imagens hospedadas para qualquer usuário baixar e usar. Um exemplo real onde o Docker é utilizado é em uma aplicação de e-commerce, que pode ter contêineres separados para o serviço de autenticação, o sistema de pagamento e o banco de dados, todos girando de forma independente.



2. Estrutura Básica de Comandos

A sintaxe para todo comando no docker é assim:

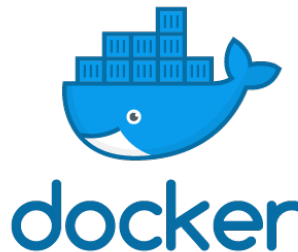
docker {objeto} {ação}

Exemplos são para listar todos os contêineres ativos:

docker container ls

Ou pedir ajuda com algum objeto:

docker {objeto} -help



Exemplos de uso

vou criar um container com o nome teste usando a imagem alpine

\$ docker container create -name teste alpine

agora para listar todos os containers criados basta usar esse comando, vai aparecer todos até os que não estão em execução.

\$ docker container ls -a

para remover o container usa esse comando:

\$ docker rm nome_container

para listar todas as imagens:

\$ docker image ls -a

para remover uma imagem específica:

\$ docker image rm alpine

para iniciar um container permitindo que você mexa com o terminal dentro do container:

\$ docker container start -it teste

para abrir um shell diretamente em modo -it

\$ docker container attach teste

ou tambem pode criar um container usando só uma linha de comando no modo interativo com o shell já aberto:

```
$ docker container run -it -name teste alpine sh
```

para renomear um container:

```
$ docker container rename B1f teste-1
```

para sair de um container é só usar Ctrl + z

para copiar uma pasta do host para o container:

```
$ docker container cp alluny teste: /
```

para visualizar os logs do container:

```
$ docker container logs teste
```

para inspecionar o container:

```
$ docker container inspect teste
```

para mapear volumes do host para o container

```
$ docker container run -v /user/bkp:/bkp -name teste -it alpine sh
```

para mapear a porta do container nginx para porta 80

```
$ docker container run -d -p 80:80 -name nginx nginx
```

para terminar a execução de um container:

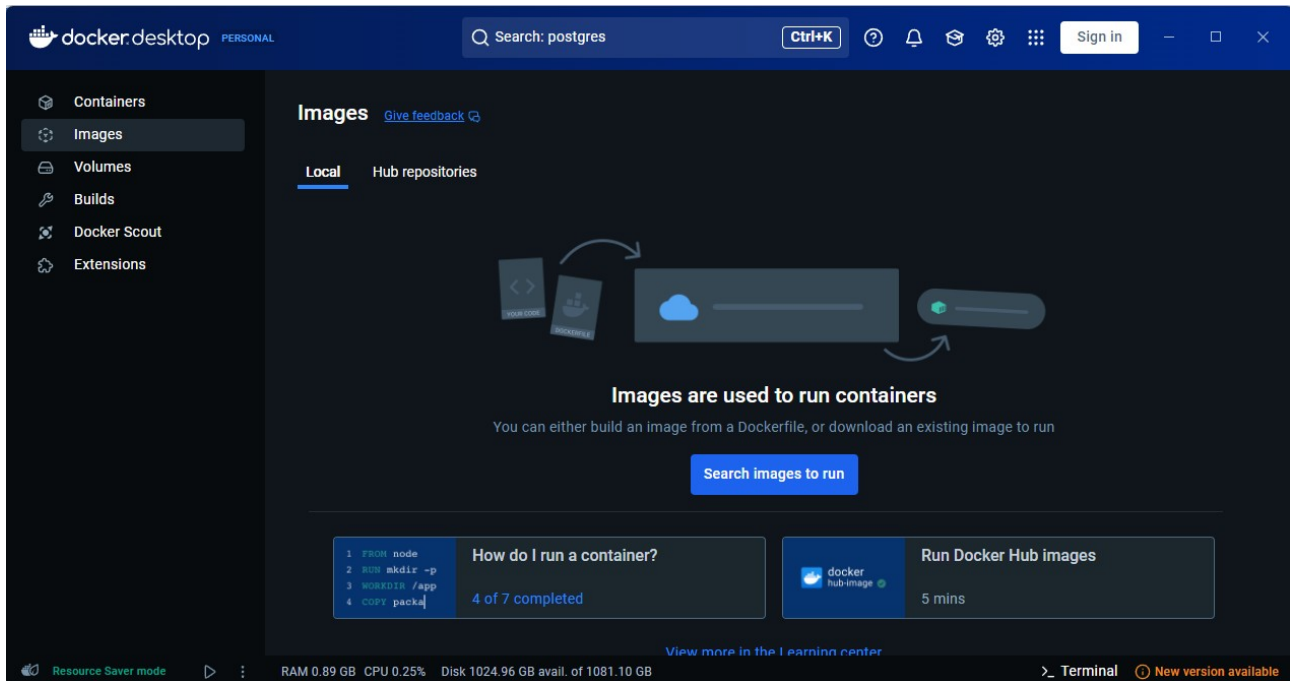
```
$ docker stop teste
```

para fazer um commit de um container para uma image

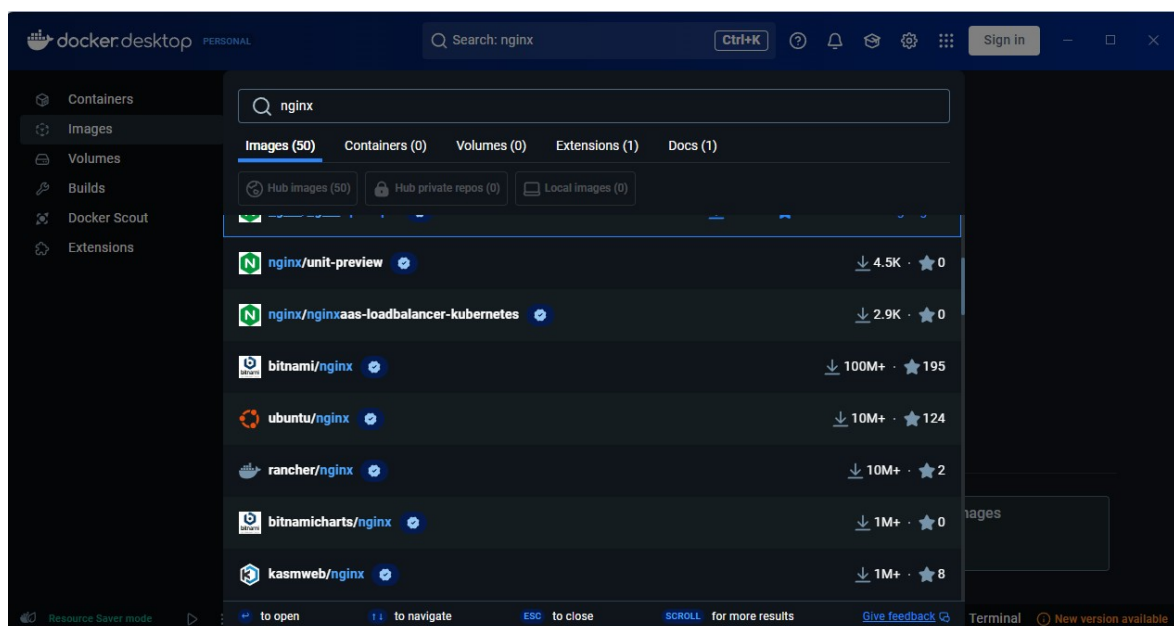
```
$ docker container commit nginx-teste nginx-teste-img
```

3. Docker Desktop

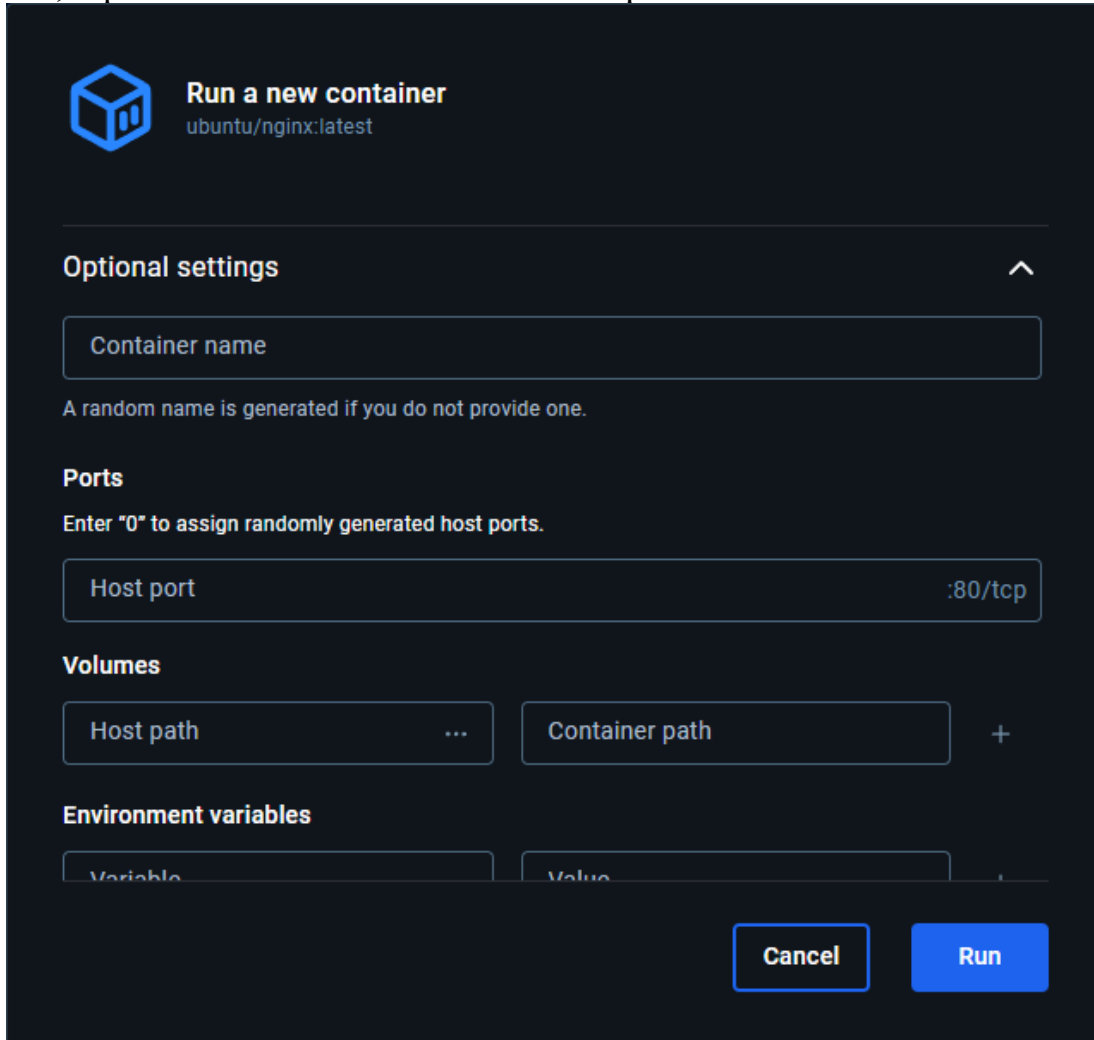
O docker desktop é um aplicativo desktop para manipular containers, image, volume, etc



Vou criar uma container bem rápido, primeiramente vai na aba images e baixa alguma image:



Feito isso, rapidamente eu crio o container sem precisar usar a comand line:



Run a new container
ubuntu/nginx:latest

Optional settings ^

Container name

A random name is generated if you do not provide one.

Ports
Enter "0" to assign randomly generated host ports.

Host port :80/tcp

Volumes

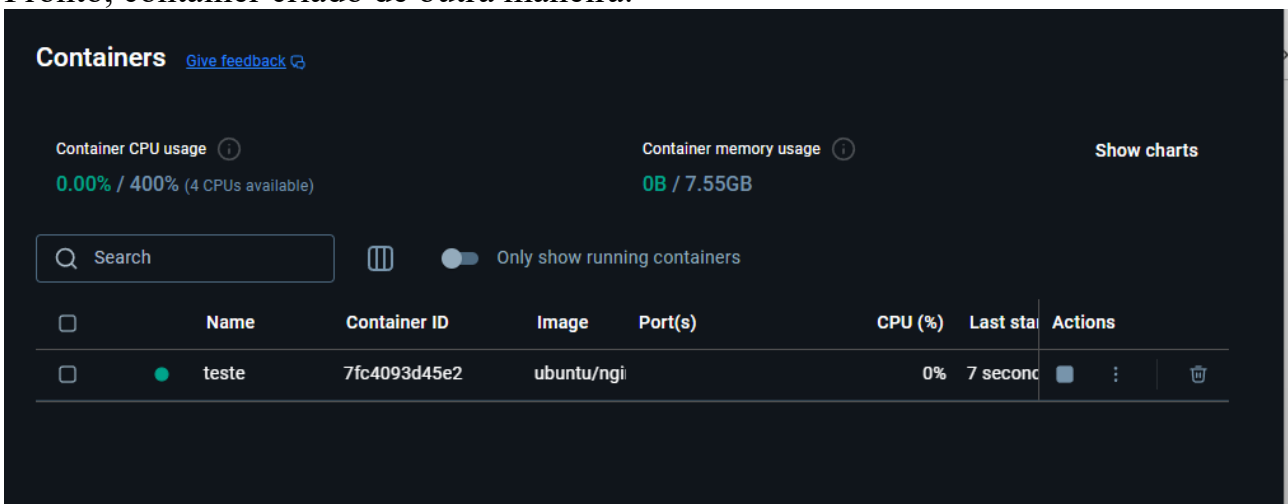
Host path ... Container path +

Environment variables

Variable Value

Cancel Run

Pronto, container criado de outra maneira.



Containers [Give feedback](#)

Container CPU usage ⓘ 0.00% / 400% (4 CPUs available)

Container memory usage ⓘ 0B / 7.55GB [Show charts](#)

Search

☐ Only show running containers

| <input type="checkbox"/> | Name | Container ID | Image | Port(s) | CPU (%) | Last state | Actions |
|--------------------------|---------|--------------|--------------|---------|---------|---------------|------------------------------|
| <input type="checkbox"/> | ● teste | 7fc4093d45e2 | ubuntu/nginx | | 0% | 7 seconds ago | <input type="checkbox"/> ⋮ 🗑 |

4. Conclusão

docker é muito utilizado para ciencias de dados e outras areas em geral, com ele é possível a criação de ambientes isolados, consistentes e portáteis. Ele evita conflitos de dependências entre projetos, facilita a colaboração por meio do compartilhamento de imagens contendo código e bibliotecas