01

Salvando dados com volumes

Transcrição

Nesta aula, começaremos a falar sobre os **volumes**, mas antes vamos relembrar o que vimos na aula anterior.

Recapitulando...

Na aula anterior, vimos que os *containers* nada mais são do que uma pequena camada de leitura e escrita, que funcionam em cima das imagens, que não podem ser modificadas, pois são somente para leitura.

Quando removemos um *container* (comando `docker rm`), a camada de leitura e escrita também é removida, o que faz com que os nossos dados também sejam removidos, o que é muito ruim, já que esses dados podem ser importantes, como por exemplo um banco de dados, então toda vez que o *container* for removido, tudo o que escrevemos nele será jogado fora? Não é isso que queremos, então temos que ver um jeito de **persistir esses dados**, mas também trabalhando com *containers*.

É da natureza dos *containers* a volatilidade, isto é, eles são criados e removidos rapidamente e facilmente, mas devemos ter um lugar para salvar os dados, e esse lugar são os **volumes**.

O que são os volumes?

Quando escrevemos em um *container*, assim que ele for removido, os dados também serão. Mas podemos criar um local especial dentro dele, e

especificamos que esse local será o nosso **volume de dados**.

Quando criamos um volume de dados, o que estamos fazendo é apontá-lo para uma pequena pasta no **Docker Host**. Então, quando criamos um volume, criamos uma pasta dentro do *container*, e o que escrevermos dentro dessa pasta na verdade estaremos escrevendo do Docker Host.

Isso faz com que não percamos os nossos dados, pois o *container* até pode ser removido, mas a pasta no **Docker Host** ficará intacta.

Trabalhando com volumes

Sabendo disso, vamos ver como trabalhar com o **Docker Host**. No Terminal ou PowerShell (ou Docker Quickstart Terminal), criamos um *container* com o `docker run`, mas dessa vez utilizando a *flag* `-v` para criar um volume, seguido do nome do mesmo:

```
docker run -v "/var/www" ubuntu
```

COPIAR CÓDIGO

No exemplo acima, criamos o volume **/var/www**, mas a que pasta no **Docker Host** ele faz referência? Para descobrir, podemos inspecionar o *container*, executando o comando `docker inspect`, passando o seu **id** para o mesmo:

```
docker inspect 8cf7b40ce226
```

COPIAR CÓDIGO

Temos uma saída com diversas informações, mas a que nos interessa é o **"Mounts"**:

```
"Mounts": [
    {
```

```
        "Type": "volume",
        "Name":
"5e1cbfd48d07284680552e56087c9d5196659600ccd6874bfa3831b5

        "Source":
"/var/lib/docker/volumes/5e1cbfd48d07284680552e56087c9d51

        "Destination": "/var/www",
        "Driver": "local",
        "Mode": "",
        "RW": true,
        "Propagation": ""
    }
]
```

COPIAR CÓDIGO

Nele, podemos ver que o **/var/www** será escrito na nossa máquina no diretório **/var/lib/docker/volumes/5e1cbfd48d07284680552e56087c9d5196659600ccd6874bfa383** endereço que foi gerado automaticamente pelo Docker. Ou seja, tudo que escrevermos na pasta **/var/www** do *container*, na verdade estaremos escrevendo na pasta **/var/lib/docker/volumes/5e1cbfd48d07284680552e56087c9d5196659600ccd6874bfa383** da nossa máquina.

E ao remover o *container*, a pasta continuará na nossa máquina. Essa pasta gerada pelo Docker pode ser configurada, podemos dizer a pasta que será referenciada pela pasta **/var/www** do *container*. Por exemplo, se quisermos escrever dentro do Desktop da nossa máquina, devemos passá-lo antes do volume, separando-os com dois pontos. Além disso, vamos executar o *container* no modo interativo:

```
docker run -it -v "C:\Users\Alura\Desktop:/var/www"
ubuntu
root@abd0286c0083:/#
```

COPIAR CÓDIGO

Ou seja, quando escrevermos na pasta **/var/www** do *container*, estaremos escrevendo no Desktop da nossa máquina. Para provar isso, na pasta **/var/www**, vamos criar um arquivo e escrever nele uma mensagem:

```
root@abd0286c0083:/# cd /var/www/
root@abd0286c0083:/var/www# touch novo-arquivo.txt
root@abd0286c0083:/var/www# echo "Este arquivo foi
criado dentro de um volume" > novo-arquivo.txt
```

COPIAR CÓDIGO

Ao acessarmos o nosso Desktop, o arquivo estará lá, também com a mensagem escrita. E ao remover o *container*, a sua camada de escrita é removida, mas os arquivos continuam no nosso Desktop.

Então, o uso de volumes é importante para salvarmos os nossos dados fora do *container*, e esses volumes sempre estarão atrelados ao **Docker Host**. No caso acima, atrelamos o volume com o Desktop, mas podemos atrelar com um lugar mais seguro, salvando os dados do banco de dados nele, logs, e até mesmo o código fonte, coisa que faremos no próximo vídeo.