Manual Docker

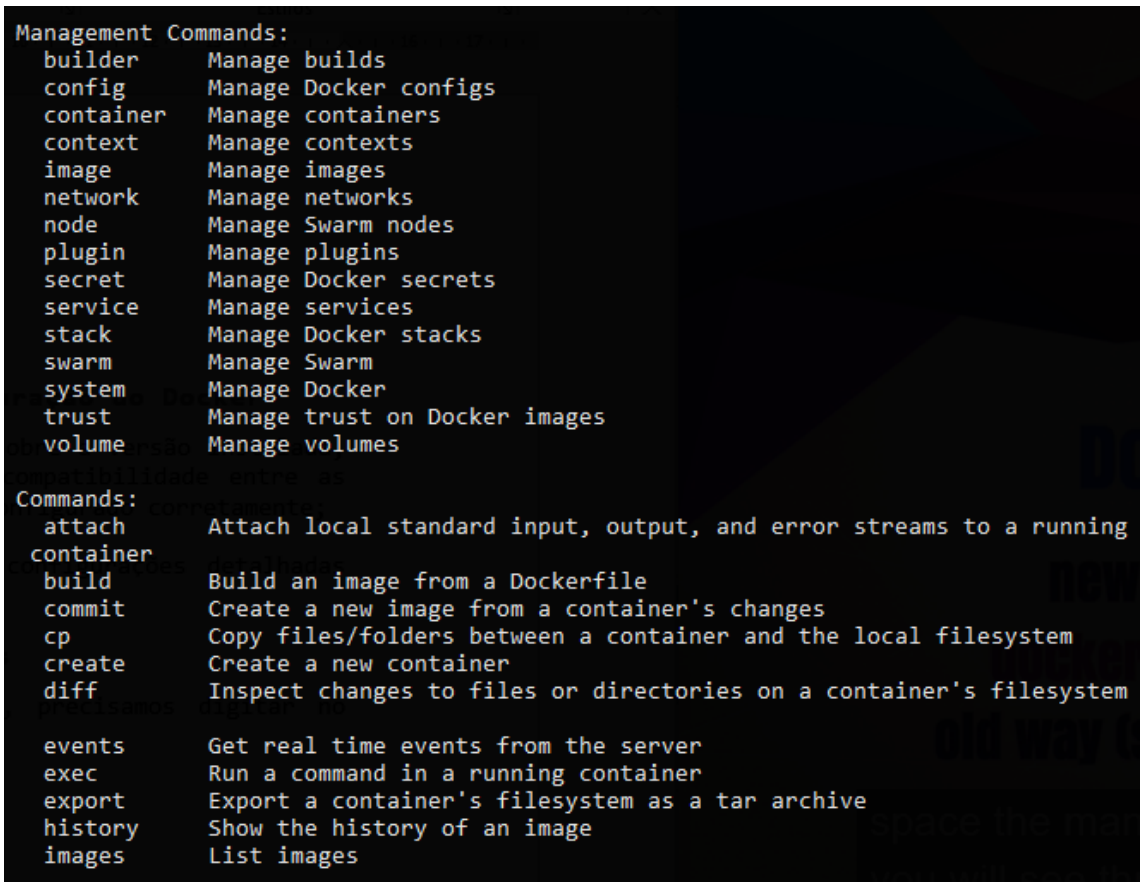
1 - Verificando a instalação e configuração do Docker

a) ***docker version***: Mostra informações sobre a versão instalada, tanto no *client* quanto no *server*. Verifica a compatibilidade entre as versões, indicando que o Docker provavelmente foi instalado e configurado corretamente;

b) ***docker info***: Mostra informações e configurações detalhadas sobre a *docker engine* instalada.

2 - Verificando os principais comandos

Para verificar os comandos, precisamos digitar no terminal o comando ***docker***.



Deste modo, serão mostrados os principais comandos de utilização do Docker.

Atualmente, na versão mais recente durante a elaboração deste manual, o padrão de utilização dos comandos do Docker segue a seguinte regra: ***docker <comando> <sub-comando> (opções)***. Por exemplo, para iniciarmos um container, devemos utilizar ***docker container run***.

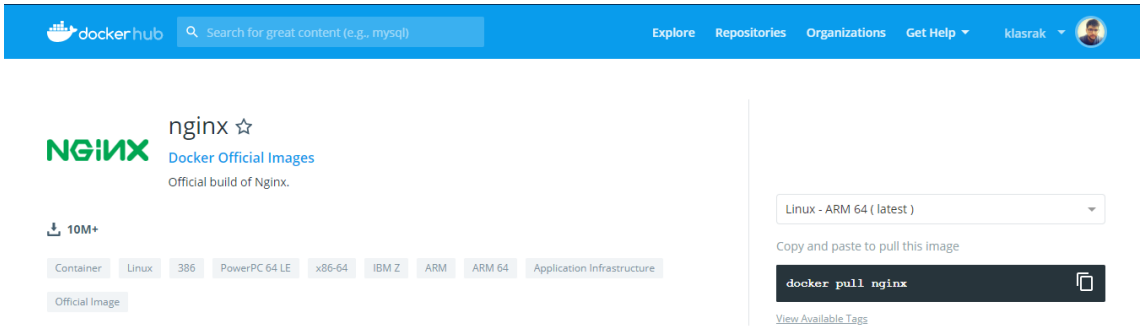
Todavia, a forma antiga de executar comandos ainda funciona, no padrão ***docker <comando> (opções)***, ou seja, utilizando o exemplo do parágrafo anterior, não teríamos problema nenhum para iniciar um container com o comando ***docker run***.

3 - IMAGE vs CONTAINER

Quando falamos em ***image***, estamos nos referindo à imagem dos binários, bibliotecas, arquivos e código fonte de nossa aplicação. Por outro lado, ***container*** é a instância de uma determinada ***image***, que pode ser executada neste ambiente isolado, analogicamente, como se de fato estivesse dentro de um contêiner.

O Docker disponibiliza um serviço chamado Docker Hub, onde podemos encontrar várias imagens, oficiais ou da comunidade, que podem ser utilizadas por qualquer pessoa. Grandes empresas como Google, Microsoft, Oracle, Canonical, Red Hat, dentre outras, mantêm versões atualizadas das imagens de seus softwares.

Nada impede que você tenha uma infinidade de ***containers*** rodando com base na mesma imagem. Por exemplo, a imagem oficial do NGINX já possui mais de 10 milhões de *downloads*, sendo utilizada por pessoas em todo o mundo.



Além disso, essa imagem sempre será "identidade" própria, ou seja, pouco importa qual ambiente o ***container*** será executado, pois temos a plena certeza de que os arquivos, binários e bibliotecas responsáveis para rodar o NGINX serão sempre os mesmos, no mesmo ambiente, isolados das demais dependências do sistema operacional utilizado.

4 - Baixando uma imagem e executando em um container

Para executar um ***container*** Docker, precisamos primeiro de uma imagem. Neste caso, vamos utilizar como exemplo a imagem oficial do NGINX. Todavia, mostrarei que não precisamos necessariamente ter a imagem em nossa máquina, pois o Docker busca automaticamente no Docker Hub e inicia o ***container*** com a imagem baixada:

```
daugusto@OOP-DSK-0204 ~/Desktop/DEV/learn/docker-mastery
λ docker container run --publish 80:80 --detach --name webhost nginx
```

Baixando a imagem e dependências, depois iniciando o ***container***:

```
daugusto@OOP-DSK-0204 ~/Desktop/DEV/learn/docker-mastery
λ docker container run --publish 80:80 --detach --name webhost nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
8ec398bc0356: Pull complete
dfb2a46f8c2c: Pull complete
b65031b6a2a5: Pull complete
Digest: sha256:8aa7f6a9585d908a63e5e418dc5d14ae7467d2e36e1ab4f0d8f9d059a3d071ce
Status: Downloaded newer image for nginx:latest
2afbcaade6b1ddf583321a71622cc6db5c893c7c067a44500e4b937e039cff0a
```

Vemos que para iniciar a execução de um ***container***, basta dizermos ao ***docker***, para executar um ***container*** com a imagem do NGINX através do comando ***run*** (***docker container run --options nginx***).

Para melhor entendimento, vamos analisar cada uma das opções passadas junto ao comando principal:

a) ***docker container run***: Comando para executar um container;

b) ***--publish*** ou ***-p***: Determina por qual porta da máquina *host* poderá ser acessada a porta do ***container***, ou seja, cria um vínculo. Neste exemplo, ficou determinado que a porta 80 do ***container*** será acessada pela porta 80 da máquina *host* (80:80).

c) --***detach*** ou ***-d***: Faz com que o ***container*** rode em segundo plano, ou seja, o terminal de comando não fica "preso" ao processo de execução;

d) ***--name***: Nomeia um ***container***. Nesse exemplo, foi dado o nome de "*webhost*".

No documentação do Docker, podemos encontrar todos os comandos que poder ser utilizados.

Podemos listar todos os ***containers*** que estão em execução, utilizamos o comando ***docker container ls***:

```
daugusto@OOP-DSK-0204 ~/Desktop/DEV/learn/docker-mastery
λ docker container ls
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                NAMES
2afbcaade6b1        nginx               "nginx -g 'daemon of…"   38 minutes ago      Up 38 minutes       0.0.0.0:80->80/tcp   webhost
```

Por padrão, o comando ***ls*** mostra apenas ***containers*** que estão em execução, porém se adicionarmos a opção ***-all*** ou ***-a***, mostrará até os ***containers*** ociosos