



Neste artigo vou apresentar os conceitos básicos relativos ao Docker sob o ponto de vista de um desenvolvedor .NET.



Hoje vamos instalar o [Docker](#) e aprender a criar [contêineres](#) a partir de [imagens](#) existentes. ([artigo anterior](#)).



Instalando o Docker

O Docker é baseado na tecnologia [LXC \(Linux Containers\)](#) mas existem atualmente versões para Windows e Mac.

A seguir temos os links para cada uma das instalações :

1. [Docker para Mac](#) - Docker Community Edition for Mac
2. [Docker para Windows](#) - Docker Community Edition for Windows

Para estes ambientes basta fazer o login, verificar os pré-requisitos, baixar o pacote e instalar.

Você pode acompanhar os artigos usando o Docker no ambiente Windows ou Mac.

Nota: [Você também pode instalar o Ubuntu em uma máquina virtual no Windows](#)

Eu vou usar o Docker em uma máquina [Linux](#) física com a distribuição [Ubuntu LTS 18.04.1](#).

- [Docker para Linux\(Ubuntu\)](#) - Docker CE for Ubuntu

Existem duas versões do Docker disponíveis :

- [Docker Community Edition\(CE\)](#).
- [Docker Enterprise Edition\(EE\)](#).

As duas versões possuem acesso a toda a API (*até agora*). A versão EE possui um ambiente homologado pelo Docker com uma infraestrutura certificada. Na versão CE você tem que fazer tudo manualmente.

Eu vou instalar a versão [Docker Community Edition\(CE\)](#) que é voltada para desenvolvedores.

Seguindo a documentação para instalação, abra um terminal no Ubuntu e faça o seguinte:

1. Atualize o pacote apt

```
$ sudo apt-get update
```

2. Instale os pacotes para permitir o apt usar o repositório sobre HTTPS

```
$ sudo apt-get install \  
apt-transport-https \  
ca-certificates \  
curl \  
software-properties-common
```

3. Adicione a chave GPG do Docker

```
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

4. Configure o repositório(X86_64/ amd64)

```
$ sudo add-apt-repository \  
"deb [arch=amd64] https://download.docker.com/linux/ubuntu \  
"
```

```
$(lsb_release -cs) \
stable"
```

5. Atualize os pacotes apt

```
$ sudo apt-get update
```

6. Instale a última versão do Docker CE

```
$ sudo apt-get install docker-ce
```

Pronto ! Para saber se a instalação foi realizada com sucesso digite o comando : `docker --version`

```
root@linux: /home/macoratti
Arquivo Editar Ver Pesquisar Terminal Ajuda
root@linux:/home/macoratti#
root@linux:/home/macoratti# docker --version
Docker version 18.06.1-ce, build e68fc7a
root@linux:/home/macoratti#
```

Você deverá ver a versão do Docker Instalada que no meu caso é a versão `18.06.1-ce`.

Agora digite o comando : `docker --help`

```
root@linux: /home/macoratti
Arquivo Editar Ver Pesquisar Terminal Ajuda
logout      Log out from a Docker registry
logs        Fetch the logs of a container
pause       Pause all processes within one or more containers
port        List port mappings or a specific mapping for the container
ps          List containers
pull        Pull an image or a repository from a registry
push        Push an image or a repository to a registry
rename      Rename a container
restart     Restart one or more containers
rm          Remove one or more containers
rmi         Remove one or more images
run         Run a command in a new container
save        Save one or more images to a tar archive (streamed to STDOUT by default)
search      Search the Docker Hub for images
start       Start one or more stopped containers
stats       Display a live stream of container(s) resource usage statistics
stop        Stop one or more running containers
tag         Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE
top         Display the running processes of a container
unpause     Unpause all processes within one or more containers
update      Update configuration of one or more containers
version     Show the Docker version information
wait        Block until one or more containers stop, then print their exit codes

Run 'docker COMMAND --help' for more information on a command.
root@linux:/home/macoratti#
```

Você deverá ver a ajuda exibindo os comandos Docker.

Dessa forma a palavra-chave `docker` é o driver que executa os comandos e os argumentos usados na linha de comando.

Criando o seu primeiro Contêiner Docker : Hello World

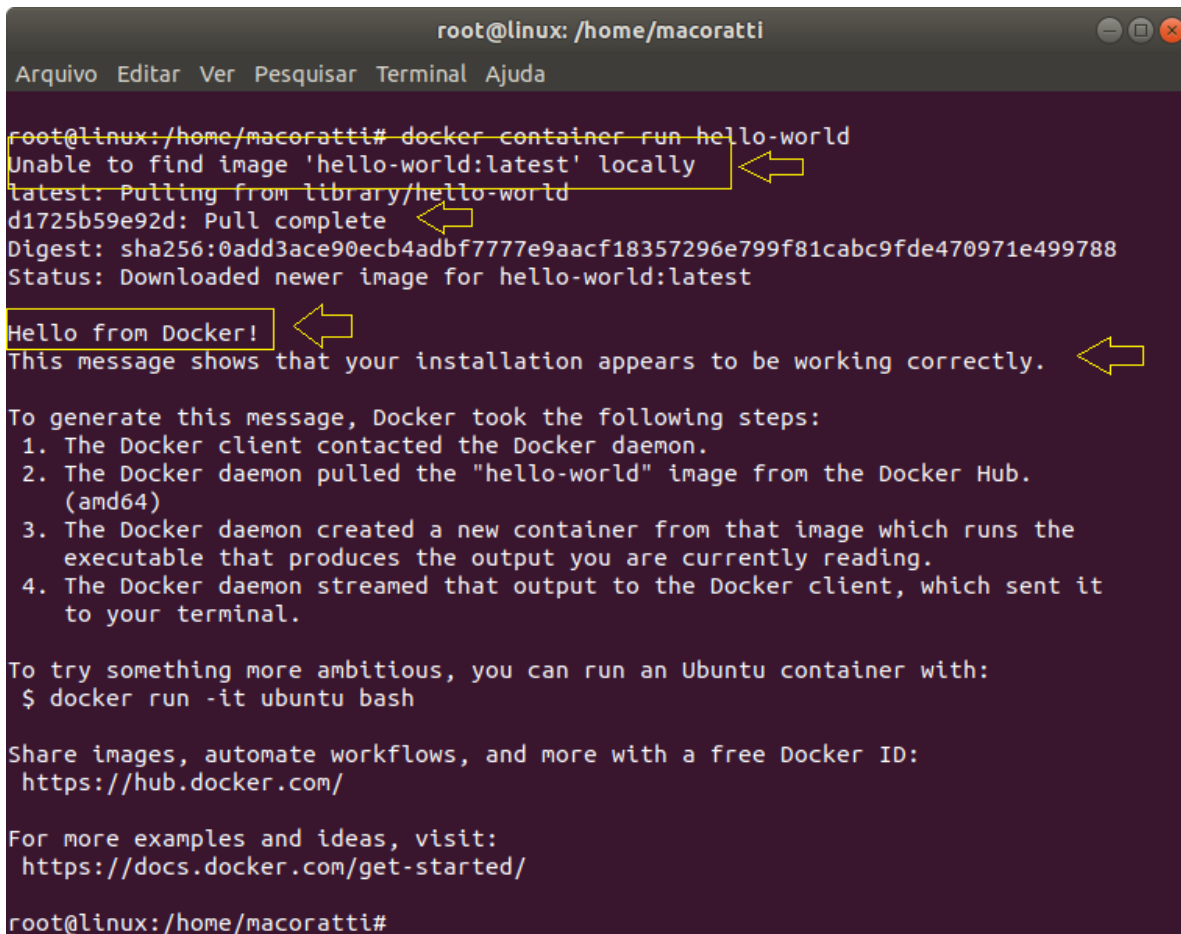
O principal objetivo do Docker é criar e gerenciar contêineres. Vamos então criar o nosso primeiro contêiner **Docker**.

Estamos usando o modo **Client** do Docker e vamos usar o **modo iterativo** do Docker como um usuário. *(Existe também o modo **daemon** no qual o contêiner fica rodando como processo em background)*

Abra o terminal e digite o comando: **docker container run hello-world**

- **docker** - é o executor do comando
- **container** - indica que o comando vai atuar em um contêiner
- **run** - é a porta de entrada no Docker e agrupa as seguintes funcionalidades básicas :
 - Download automático das imagens não encontradas localmente: **docker image pull**
 - Criação do container: **docker container create**
 - Inicialização do container: **docker container start**
 - Execução do contêiner : **docker container exec**
- **hello-world** - é a imagem existente usada para criar o contêiner

Abaixo vemos o resultado da execução do comando:

A terminal window titled 'root@linux: /home/macoratti' with a menu bar (Arquivo, Editar, Ver, Pesquisar, Terminal, Ajuda). The command 'docker container run hello-world' is entered. The output shows the Docker client pulling the 'hello-world:latest' image from the Docker Hub. It displays the image ID 'd1725b59e92d', the digest 'sha256:0add3ace90ecb4adbf7777e9aacf18357296e799f81cab9fde470971e499788', and the status 'Downloaded newer image for hello-world:latest'. The container then outputs 'Hello from Docker!' followed by a message confirming the installation is working correctly. A list of four steps explains the process: 1. Docker client contacted the daemon, 2. daemon pulled the image from Docker Hub, 3. daemon created a container running the executable, 4. daemon streamed output to the client. Finally, it suggests running 'docker run -it ubuntu bash' and provides links to Docker Hub and documentation.

```
root@linux: /home/macoratti# docker container run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
d1725b59e92d: Pull complete
Digest: sha256:0add3ace90ecb4adbf7777e9aacf18357296e799f81cab9fde470971e499788
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

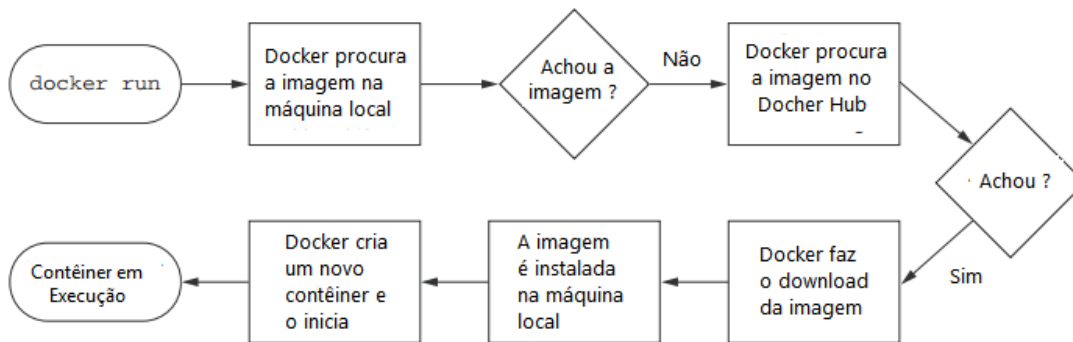
Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/

root@linux: /home/macoratti#
```

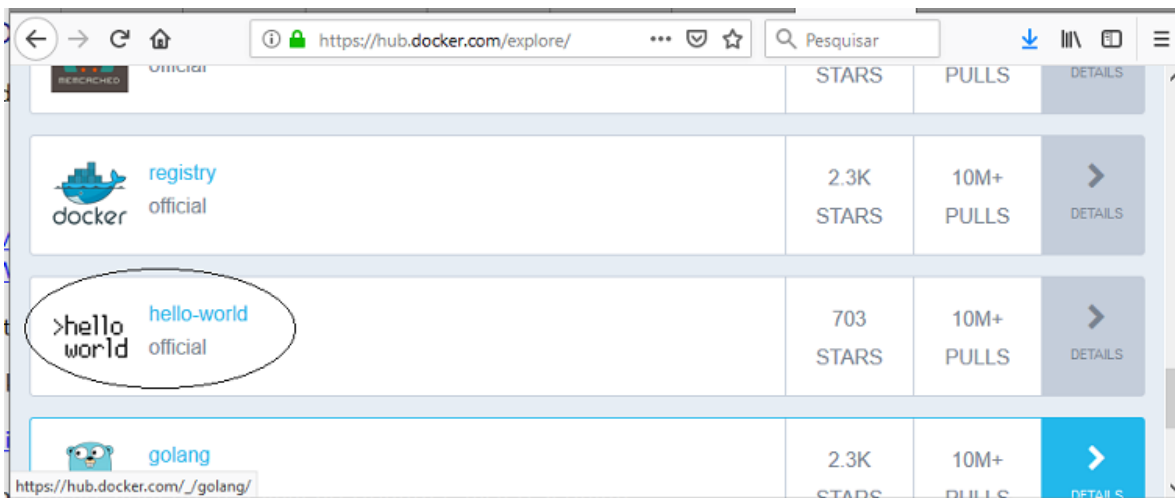
O que aconteceu aqui ???

O exemplo hello-world



O Docker tentou localizar a imagem [hello-world](#) usada na sua máquina local e não encontrou.

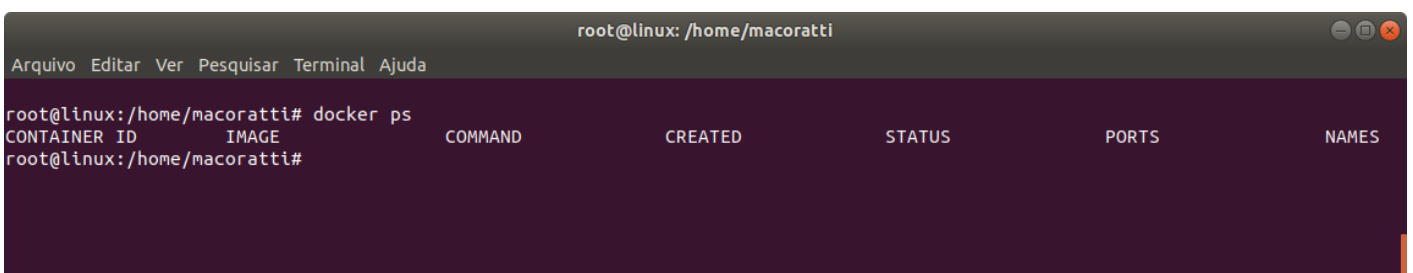
Daí ele baixou a imagem do repositório [Docker hub](#) como mostrada abaixo:



Executou o contêiner e exibiu o texto : [Hello from Docker](#)

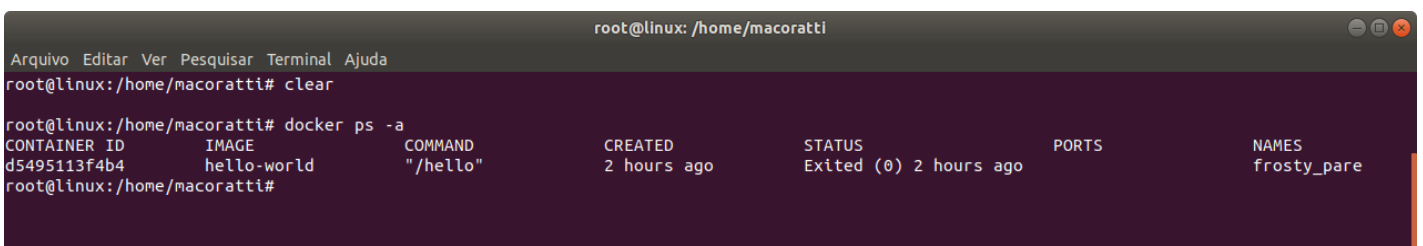
Concluindo ele informou que a instalação do Docker esta ok.

Para listar os [processos](#) dos contêineres em execução usamos o comando : **docker container ps**



Note que o [contêiner](#) que criamos não aparece, pois ele foi executado e encerrado, e não esta em execução.

Para exibir todos os contêineres em execução e os parados digite : **docker container ps -a**



Agora vemos o contêiner criado a partir da imagem [hello-world](#) exibindo o seu [container id](#) e sendo identificado pelo nome [frosty_pare](#); temos também a data de criação([created](#)) e o [status](#) do contêiner.

Nota: *Voce pode usar ou não palavra **container** nestes comandos, mas recomenda-se usar para deixar claro que o comando atua sobre contêineres.*

Pronto. Você criou o primeiro contêiner Docker usando o comando:

docker container run <nome_da_imagem>

O comando **run** faz o download da imagem, cria o contêiner, executa o contêiner e permite o uso do contêiner no modo interativo.

Na [próxima aula](#) vamos continuar explorando o **Docker** e a criação de **contêineres**.