# **Docker: criando o seu primeiro container**

[](https://medium.com/@lauradamaceno?source=post_page-----61ac022ceeef--------------------------------)

[](https://medium.com/devs-javagirl?source=post_page-----61ac022ceeef--------------------------------)

[Laura Damaceno](https://medium.com/@lauradamaceno?source=post_page-----61ac022ceeef--------------------------------)

·

[Follow](https://medium.com/m/signin?actionUrl=https%3A%2F%2Fmedium.com%2F_%2Fsubscribe%2Fuser%2F23dc6c5711de&operation=register&redirect=https%3A%2F%2Fmedium.com%2Fdevs-javagirl%2Fdocker-criando-o-seu-primeiro-contêiner-61ac022ceeef&user=Laura+Damaceno&userId=23dc6c5711de&source=post_page-23dc6c5711de----61ac022ceeef---------------------post_header-----------)

Published in

[Devs JavaGirl](https://medium.com/devs-javagirl?source=post_page-----61ac022ceeef--------------------------------)

·

5 min read

·

Jan 23, 2019

2

*Este é meu primeiro post no medium e será bem especial! Quer aprender a construir o seu primeiro container sem dor de cabeça? Vem comigo…*



Fala, empoderadas(os)!

Vamos para uma breve introdução do Docker! Docker para quem não sabe, é uma ferramenta que facilita a criação e alterações de containers.

# Mas o que são containers?

Containers é como se fosse uma casa que abrigará a minha aplicação. Ou seja, funcionando junto com o SO, rodando de forma isolada e independente.

Essa ferramenta está em alta atualmente por conta da disseminação da cultura DevOps.

DevOps é a integração do time de desenvolvimento com o time de operações, surgindo assim a sua sigla.

Tratando-se de um conjunto de práticas que enfatizam a colaboração e comunicação entre as áreas de TI, podendo ser um recurso e uma cultura corporativa para a entrega contínua de software e serviços com mais rapidez e confiabilidade para o cliente, a fim de obter um feedback mais valioso do mesmo.



# VM X Container

Mas porque eu usaria ele, sendo que uma Máquina Virtual também pode armazenar minha aplicação?

1. Leveza: VM’s são bem mais pesadas que containers, pois em cada VM precisamos instalar um SO e as dependências da aplicação, já os containers é executado junto com kernel da máquina.
2. Agilidade: na hora de criar e remover aplicações.
3. Facilidade: na hora de trabalhar com diferentes versões de linguagens e bibliotecas, pois o Docker encapsula todas as dependências de software associadas à execução de um programa.
4. Menos custo operacional/manutenção: pois manter vários sistemas operacionais requer muito tempo de trabalho.
5. Melhor utilização dos recursos.
6. Oferece uma quantidade maior de ecossistemas e ferramentas.

Agora que você já sabe os pontos positivos de se usar containers, vamos aprender como criar nosso primeiro container, neste post irei falar especificamente como construí-los, utilizando o Docker for windows.

# Preparando a máquina

Para instalar, você irá precisar ver se o seu computador contém os seguintes pré-requisitos:

* Windows 10 versão pro, enterprise ou education(para conferir isso, basta ir na barra de pesquisa do menu iniciar e procurar por “sistema”).
* X64.
* Virtualização habilitada (vá no seu gerenciador de tarefas, clique na aba “desempenho”, por padrão o windows já deixa habilitado).

Caso você não tenha nenhum desses requisitos acima, pode instalar o Docker toolbox, que faz a mesma coisa que o “Docker for windows” e já vem com as dependências necessárias para a sua execução.

# Mas e se eu não utilizo o Windows, o que eu faço?

Calma gente, dá para você aprender os comandos que irei ensinar online mesmo, chamada “Play with docker”(<https://labs.play-with-docker.com/>).

A diferença entre o “Docker for Windows” e o “Docker toolbox” é que, no primeiro caso, utilizamos o terminal nativo do Windows. Já no segundo, ele instalará o Docker machine que deverá ser utilizado no lugar do terminal nativo.

Para instalar, é necessário criar uma conta no docker hub e seguir os passos para a instalação, autorize o instalador. Depois de fazer o login, habilite o Hyper-v clicando em “Ok” e, com isso, o computador será reiniciado.

Após a instalação vamos confirmar se está mesmo instalado, abra seu terminal do windows (powershell ou cmd) e digite:

>> *docker - -version*



Antes de criarmos nosso primeiro container, temos que ter em mente que o Docker utiliza uma “receita de bolo”, sendo chamada de “imagem”, ou seja, uma série de instruções que o Docker faz para criar um container.

Elas estão disponíveis em um repositório open source chamada Docker hub.

Então se quiser criar um container que tenha o Ubuntu, lá vem a frase secreta…..

Coloque:

*>> docker run ubuntu*

Sempre que formos criar um container utilizaremos a sintaxe:

*>>docker run imagem(imagem que queremos que ele utilize)*

Exemplo, se eu quiser criar um container com node:

*>> docker run node*

Para verificarmos se foi criado:

*>> docker ps -a*

Existem 2 tipos de comandos para listar os containers:

* *docker ps: exibe todos os containers ativos*
* *docker ps -a: exibe todos os containers criados*

# Parabéns! Você acabou de criar seu primeiro container.

Ai Laura, mas como eu vou saber que ele instalou mesmo o Ubuntu no meu container? Vamos lá hehehe

Vamos criar um novo container só que dessa vez Vamos colocar a flag “-it”.

*>> docker run -it Ubuntu*

Depois que ele criar meu container, ele vai abrir um cmd interativo para podermos manipula-lo. Se digitar qualquer comando para ubuntu, como por exemplo:

>>echo “Ola mundo”, ele irá imprimir.

Para sair do terminal ubuntu basta dar um CRTL+C ou escrever exit.

Para ativar um container já criado, basta utilizar o comando:

*>> docker attach idcontainer*

# E como remover um container?

*>> docker rm idContainer*

Esse comando só remove containers inativos, então sempre verifique se o container está inativo antes de removê-lo, através do comando “Docker ps” se ele estiver ativo, pare a execução dele através do comando:

*>> docker stop idContainer*

Para buscar o id do container, podemos listar eles:

*>> docker ps -a*



Após o comando de remoção, se listarmos novamente, o container não estará mais lá.

E como remover uma image do docker

docker rmi image   
 Ex: docker rmi mysql