

# Docker - Limpando contêineres, imagens e volumes



Neste artigo vou apresentar os principais comandos **Docker** para limpar **contêineres, imagens e volumes**.



Se você não conhece o **Docker** sugiro que acompanhe a série de artigos : [Docker uma introdução básica](#)

A facilidade em criar contêineres, imagens e volumes pode poluir o seu ambiente, e, em um momento você vai precisar fazer um limpeza removendo os recursos que não serão mais utilizados.



**Curso Docker Essencial  
para a plataforma .NET**

Vejamos quais os principais comandos que podemos usar para *limpar o nosso ambiente do Docker*.

## Exibindo os recursos do ambiente

A primeira coisa a fazer é identificar os recursos, **contêineres, imagens, volumes e redes** existentes no seu ambiente Docker.

Para isso podemos usar os seguintes comandos :

<b>docker container ls</b>	Lista os contêineres que estão em execução. ( <b>docker ps</b> )
<b>docker container ls -a</b>	Lista todos os contêineres. ( <b>docker ps -a</b> )
<b>docker image ls</b>	Lista as imagens ( <b>docker images</b> )
<b>docker volume ls</b>	Lista os volumes
<b>docker network ls</b>	Lista as redes
<b>docker info</b>	Lista a quantidade de contêineres e imagens e informações do ambiente

Dessa forma temos **contêineres** que pode estar em execução e os que estão parados, e, temos também **imagens não usadas**, imagens que foram usadas para criar contêineres e **imagens pendentes** que não possuem relacionamento com nenhuma imagem taguada.(as **dangling images**).

Uma imagem não usada significa que ela não foi atribuída ou usada em um contêiner. Por outro lado, uma imagem pendente significa que você criou um novo **build** da imagem, mas não recebeu um novo nome.

Então as velhas imagens que você tem se tornam a "*imagem pendente*". Essas imagens antigas são aquelas sem tag e exibem "**<none>**" em seu nome quando você executa o comando : **docker images**

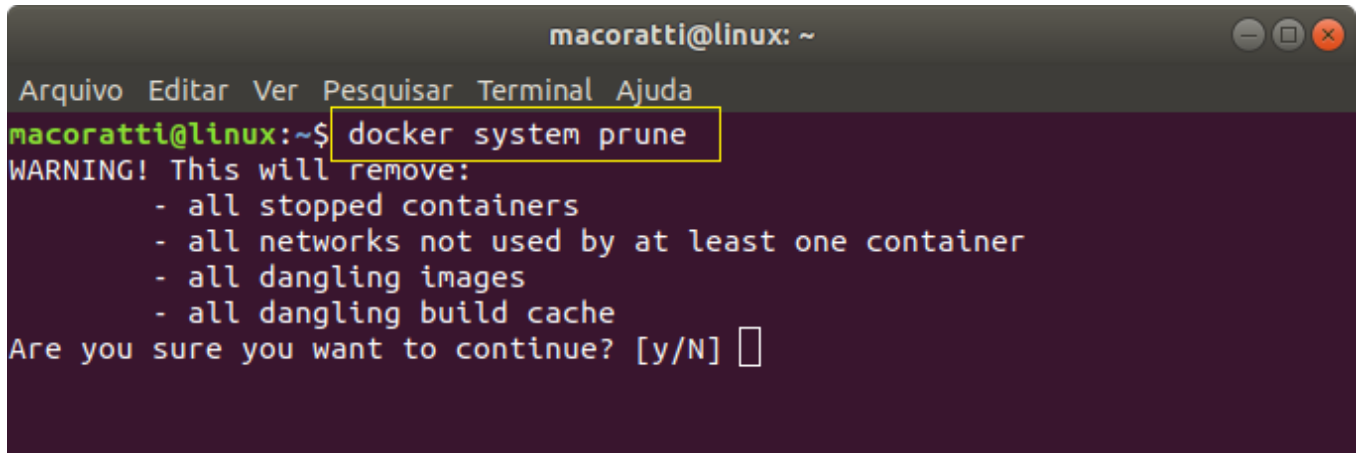
## Removendo todos os recursos

O Docker oferece um comando conveniente para remover **contêineres** , **networks** e **imagens** não usadas :

### **docker system prune**

Este comando remove :

- Todos os contêineres parados
- Todas as redes não usadas pelo menos por um contêiner
- Todas as imagens pendentes (*dangling images*)
- Todo o cache build pendente



```
macoratti@linux: ~  
Arquivo Editar Ver Pesquisar Terminal Ajuda  
macoratti@linux:~$ docker system prune  
WARNING! This will remove:  
- all stopped containers  
- all networks not used by at least one container  
- all dangling images  
- all dangling build cache  
Are you sure you want to continue? [y/N] ☐
```

Note que é solicitada a confirmação para continuar o processamento [y/N].

Para remover também os volumes e imagens não utilizadas e sobrescrever o prompt de confirmação podemos usar o comando:

### **docker system prune --all --force --volumes**

Este comando remove:

- Todos os contêineres parados
- Todas as redes não usadas pelo menos por um contêiner
- Todos os volumes não usados por pelo menos um contêiner
- Todas as imagens sem nenhum contêiner associado
- Todo o cache build pendente

E não solicita confirmação. Portanto cuidado ao usar o comando. 😊

Também podemos aplicar o comando **prune** a cada recurso individualmente.

Assim:

- **docker container prune** - remove todos os contêineres não usados;
- **docker image prune** - remove todas as imagens não usadas;
- **docker volume prune** - remove todos os volumes não usados;
- **docker network prune** - remove todas as redes não utilizadas;

## Removendo recursos individualmente

Outra opção é remover cada recurso individualmente. Para isso podemos usar os seguintes comandos:

- **docker container rm <id> ou docker rm <id>**
- **docker image rm <id> ou docker rmi <id>**
- **docker volume <id>**
- **docker network <id>**

Remove o recurso identificado pelo **<id>** ou **nome especificado**.

Para **containeres, imagens e volumes** temos ainda a opção **-f** que força a remoção do recurso.

- **docker container rm <id> -f**
- **docker image rm <id> -f**
- **docker volume <id> -f**

Essa abordagem é mais segura, mas dependendo da quantidade de recursos, terá que ser repetida várias vezes.

Podemos ainda usar uma combinação de comandos para remover mais de um recurso de uma vez.

- Contêineres: **docker container rm \$(docker container ls -a -q)**
- Imagens : **docker image rm \$(docker image ls -a -q)**
- Volumes : **docker volume rm \$(docker volume ls -q)**
- Networks : **docker network rm \$(docker network ls -q)**

Onde:

- a** : significa todos os recursos
- q** : significa o **ID numérico** do recurso

Assim, para parar todos os contêineres podemos usar o comando:

**docker container stop \$(docker container ls -a -q)**

E podemos também vincular dois comandos para limpar todo o ambiente:

**docker container stop \$(docker container ls -a -q) && docker system prune -a -f --volumes**

Dessa forma, você tem essas opções para limpar o seu ambiente, podendo também criar scripts com combinação de comandos para serem executados automaticamente.