Fundamentos de DevOps

Aula 05 - Contêineres com Docker

Prof. Esp. Guilherme Jorge Aragão da Cruz

guilherme.cruz@alumni.usp.br

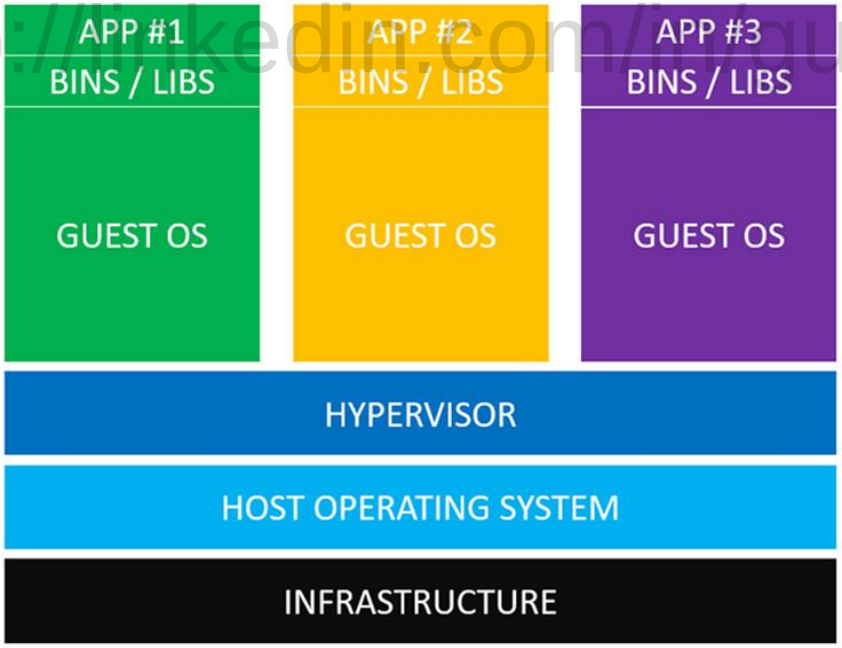
linkedin.com/in/guijac

Roteiro

- Revisitando Virtualização:
  - Máquinas Virtuais;
  - WSL (WSL (*Windows Subsystem for Linux*).
- Contêineres:
  - Definição;
  - Contexto Histórico.
- Docker x Máquina Virtual;
- Imagem Docker;
- Dockerfile;
- Principais Comandos Docker;
- Docker Engine;
- Arquitetura do Docker
- Referências Bibliográficas.

Máquinas Virtuais

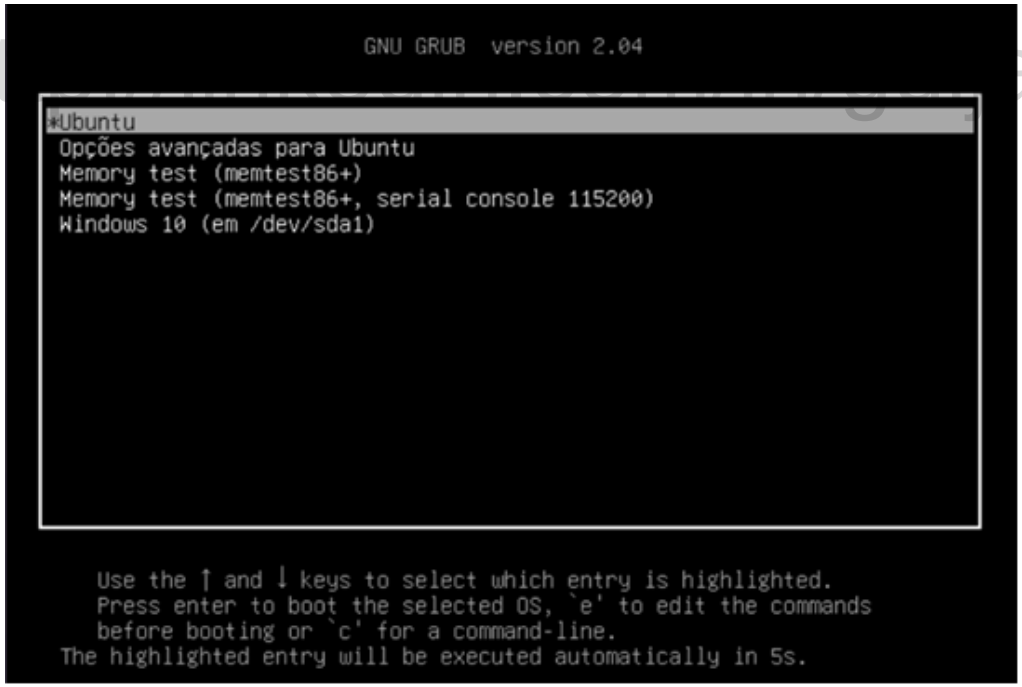
- Máquinas virtuais (virtual machines – VMs) são **softwares** que possibilitam um uso como um computador físico, através de um processo chamado de **virtualização**.



Fonte: Comparing Virtual Machines vs Docker Containers — Nick Janetakis

WSL (*Windows Subsystem for Linux*)

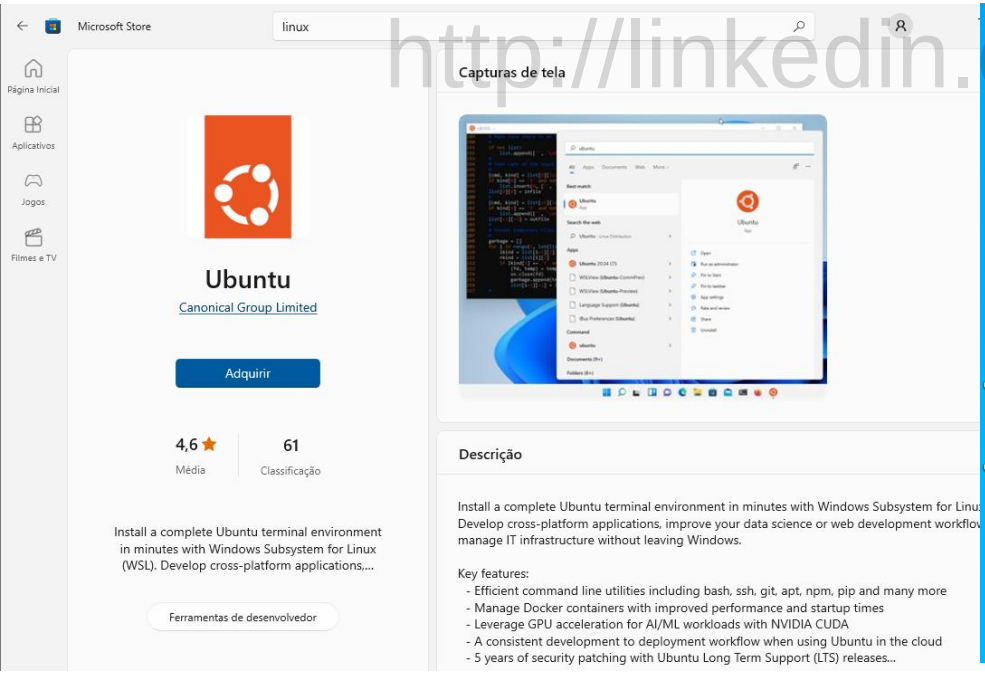
- No passado recente, uma pessoa usuária de Windows que tivesse a necessidade de utilizar Linux precisaria recorrer a técnicas de dual boot ou instalação de uma VM de terceiros.

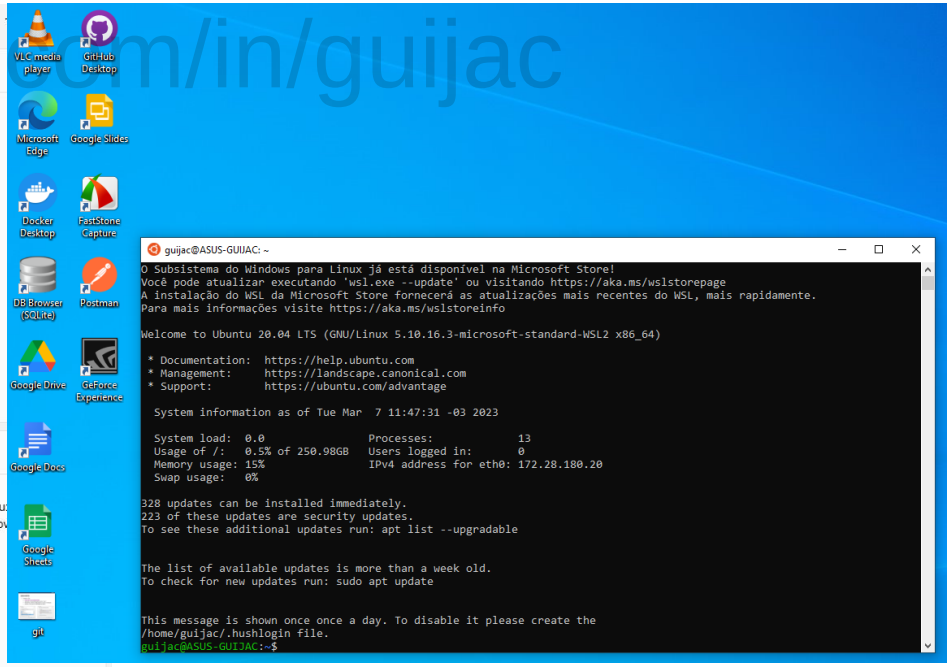


Fonte: Como fazer um dual boot com Windows e Linux – Tecnoblog

WSL (*Windows Subsystem for Linux*)

- WSL surge para simplificar este processo, possibilitando uma virtualização nativa para Linux em um ambiente Windows (Windows 11 ou no Windows 10, versão 1903, build 18362 ou superior).

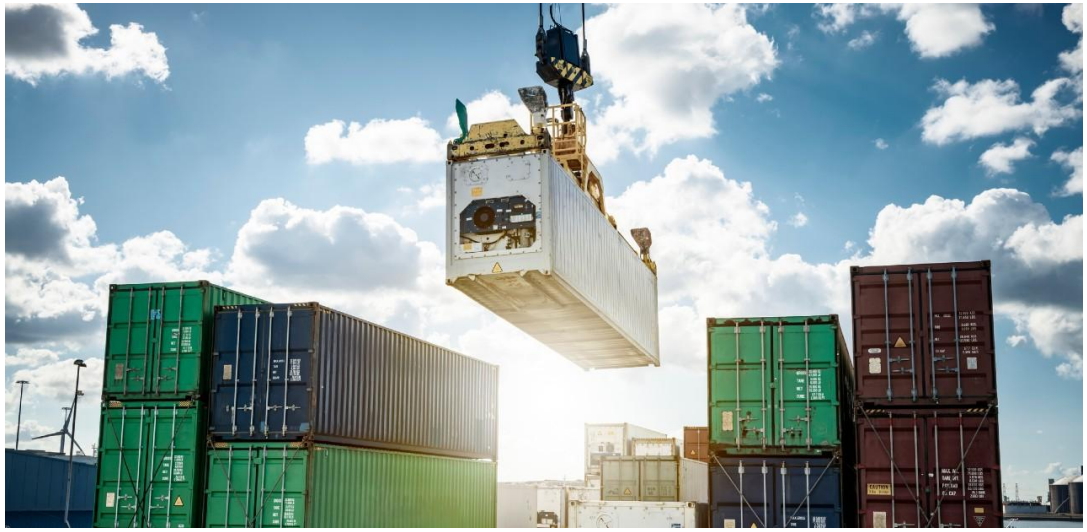




Fonte: Elaboração Própria (2023)

Contêineres?

Contêineres?



Fonte: Entenda como funciona o transporte de container - TPC Logística Inteligente

Contêineres?

> *Equipamento logístico versátil, utilizado para transportar carga;*
> *Permite uma **economia de escala** que não seria possível com o manuseio de carga fracionada padrão, trazendo também uma maior garantia de **inviolabilidade** e redução de perdas e avarias.*

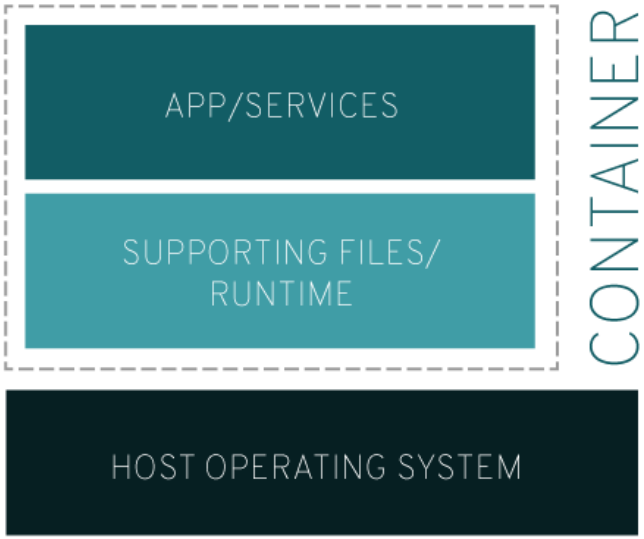
TSA CARGO (2023)

Fonte: Entenda como funciona o transporte de container - TPC Logística Inteligente

Contêineres: Definição

> *Conjunto de um ou mais **processos** organizados de **forma isolada** em um sistema;*
> *Todos os arquivos necessários para sua execução são disponibilizados por uma **imagem** individual;*
> *São **portáteis** e **consistentes**, possibilitando uma **migração** entre os **diferentes ambientes** de desenvolvimento.*

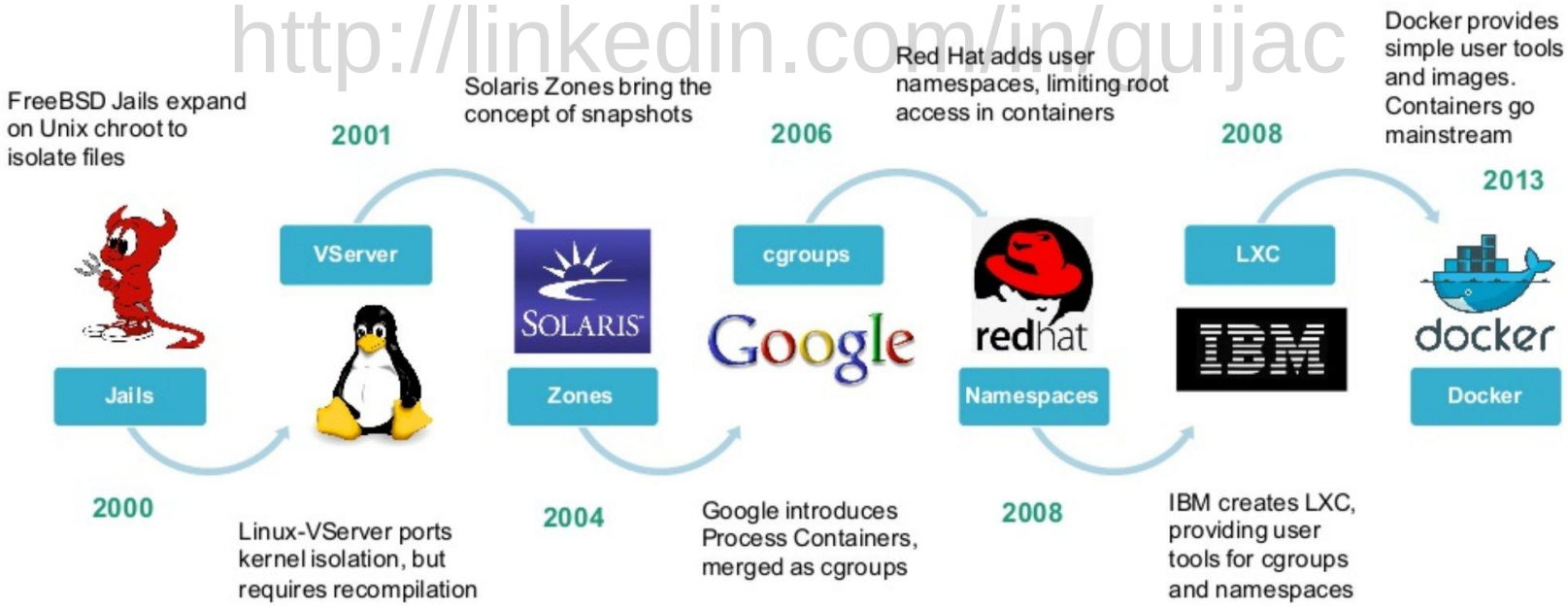
RED HAT (2025)



Fonte: O que é um container Linux? (redhat.com)

Contêineres: Contexto Histórico

- De forma ampla, contêineres surgem a partir da necessidade de melhorias das máquinas virtuais;
- Docker surge após uma série de conceitos introduzidos por outras tecnologias.



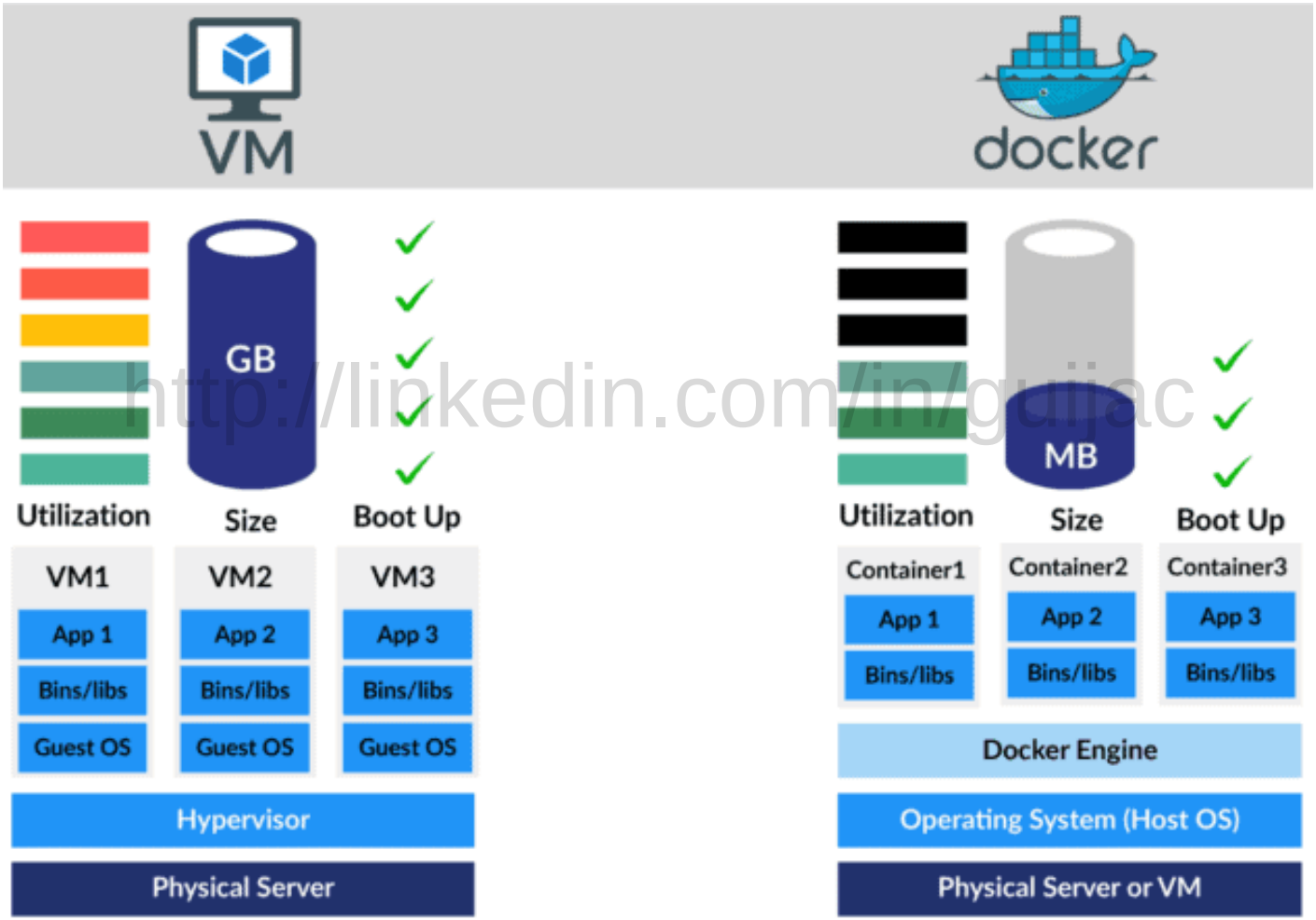
Fonte: Containerization History - Docker Handbook (gitbook.io)

Contêineres: Contexto Histórico

- De forma ampla, contêineres surgem a partir da necessidade de melhorias das máquinas virtuais;
- Docker surge após uma série de conceitos introduzidos por outras tecnologias.

Fonte: Podman, a nova opção para container engine – uma alternativa ao Docker | iMasters

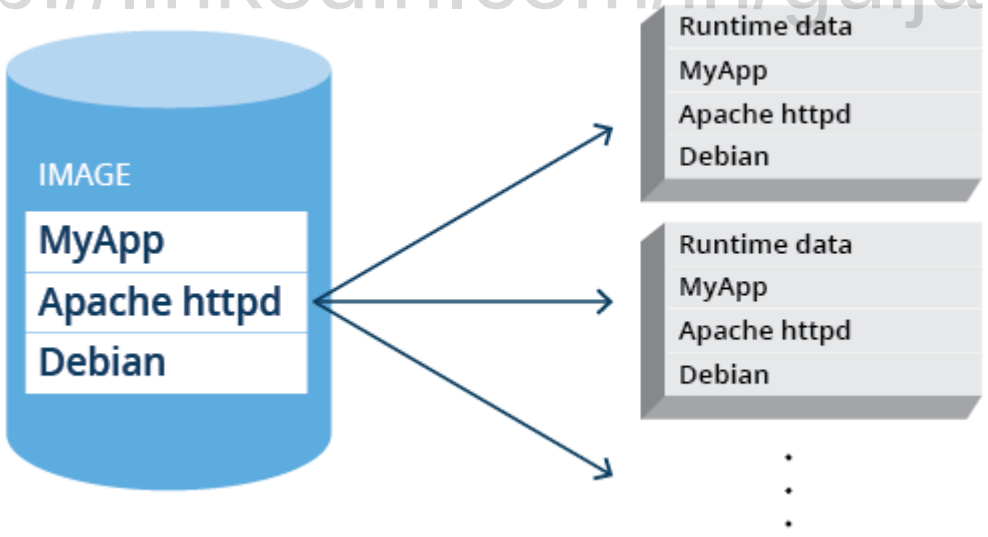
Docker x Máquina Virtual



Fonte: [Container vs VM (Virtual Machine) - Know the Difference (k21academy.com)]

Imagem Docker

- Uma imagem Docker contém todos os arquivos necessários para a execução de um contêiner;
- É uma **representação imutável** de como o contêiner será construído.



Fonte: Docker containers: What are the open source licensing considerations? (linuxfoundation.org)

Dockerfile

- Arquivo que descreve as etapas para que o Docker possa construir uma imagem, desta forma podemos criar imagens próprias, como de aplicações desenvolvidas no dia a dia.

```
# Especifica a imagem base a ser utilizada:
FROM <nome-da-imagem>:<versão-da-imagem>
# Especifica um diretório de trabalho do contêiner, para execução dos comandos seguintes:
WORKDIR </nome-do-seu-diretório-de-trabalho>
# Permite a cópia de um arquivo do diretório local para o workdir do contêiner:
COPY <nome-do-arquivo-origem.txt> <nome-do-arquivo-destino.txt>
# Permite a cópia de um diretório local para o workdir do contêiner:
COPY <nome-do-diretório-origem> <./nome-do-diretório-destino>
# Permite a execução de comandos de SO, como a geração de um JAR, através do Maven:
RUN <comando-mvn>
# Documenta a porta de exposição da aplicação (opcional, mas útil para documentar):
EXPOSE <porta-http>
# Instrução para a execução da aplicação do contêiner:
CMD ["<seu>", "<-comando>" , "<sua-aplicação.jar>"]
```

Exemplo parcial de um arquivo Dockerfile para execução de uma aplicação Java + Spring Boot

Dockerfile

- Arquivo que descreve as etapas para que o Docker possa construir uma imagem, desta forma podemos criar imagens próprias, como de aplicações desenvolvidas no dia a dia.

```
# Especifica a imagem base a ser utilizada:
FROM openjdk:21
# Especifica um diretório de trabalho do contêiner, para execução dos comandos seguintes:
WORKDIR my-app
# Permite a cópia de um arquivo do diretório local para o workdir do contêiner:
COPY pom.xml pom.xml
# Permite a cópia de um diretório local para o workdir do contêiner:
COPY src ./src
# Permite a execução de comandos de SO, como a geração de um JAR, através do Maven:
RUN mvn clean package -DskipTests
# Documenta a porta de exposição da aplicação (opcional, mas útil para documentar):
EXPOSE 8080
# Instrução para a execução da aplicação do contêiner:
CMD ["java", "-jar" , "app.jar"]
```

Exemplo parcial de um arquivo Dockerfile para execução de uma aplicação Java + Spring Boot

Principais Comandos Docker

- **docker run**
  - Cria um contêiner a partir de uma imagem especificada;
  - Mesmo que a imagem não esteja armazenada localmente, irá realizar o download da mesma.


```
docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:6e8b6f026e0b9c419ea0fd02d3905dd0952ad1feea67543f525c73a0a790fefb
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
```

Principais Comandos Docker

- **docker ps**
  - Possibilita a visualização dos contêineres criados, podendo ser combinado com outros parâmetros, como o “-l”, que exibe o último contêiner criado.


```
docker ps -l
CONTAINER ID   IMAGE         COMMAND    CREATED       STATUS                     PORTS     NAMES
01fcbb8fc7d1   hello-world   "/hello"   2 hours ago   Exited (0) 2 hours ago             angry_margulis
```

Principais Comandos Docker

- **docker stop**
  - Irá interromper a execução de um determinado contêiner, através do seu nome ou ID.


```
docker stop 8b35f6ed183f
8b35f6ed183f

docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

Principais Comandos Docker

- **docker build**
  - Responsável por criar uma imagem a partir de um arquivo Dockerfile;
  - A partir do parâmetro “—tag” define um nome opcional para o identificador da imagem.


```
docker build --tag guijac/nginx:1.0 .
[+] Building 39.8s (7/7) FINISHED
 => [internal] load build definition from Dockerfile
0.0s
 => => transferring dockerfile: 392B
0.0s
 => [internal] load .dockerignore
0.0s
 => => transferring context: 2B
0.0s
 => [internal] load metadata for docker.io/library/debian:latest
2.7s
 => [1/3] FROM
docker.io/library/debian:latest@sha256:f81bf5a8b57d6aa1824e4edb9aea6bd5ef6240bcc7d86f303f197a2eb7  12.2s
```

Principais Comandos Docker

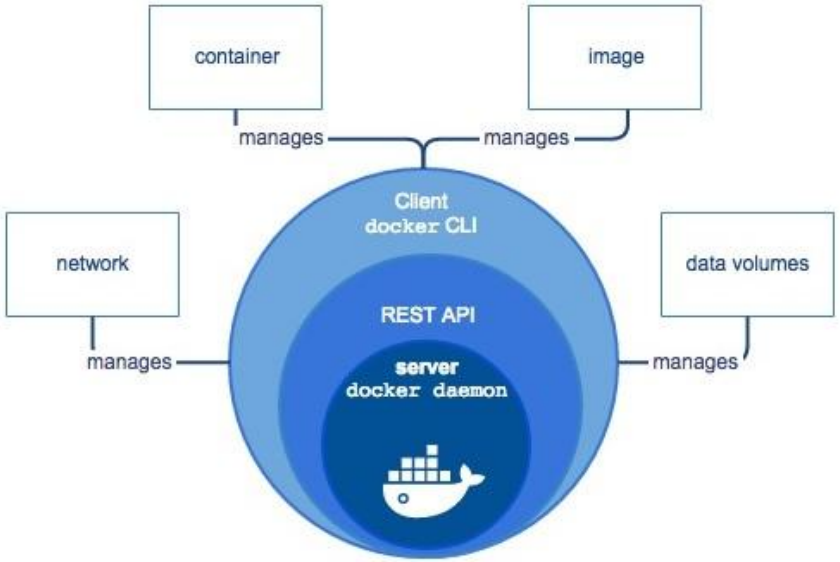
- **docker images**
  - Responsável por gerenciar as imagens locais, aceitando também parâmetros opcionais como “rm”, para remover determinada imagem.


```
docker images

REPOSITORY          TAG       IMAGE ID       CREATED              SIZE
guijac/nginx        1.0       74fe6b8a5e08   About a minute ago   211MB
docker101tutorial   latest    d0b552085444   7 weeks ago          47MB
alpine/git          latest    22d84a66cda4   3 months ago         43.6MB
hello-world         latest    feb5d9fea6a5   17 months ago        13.3kB
```

Docker Engine

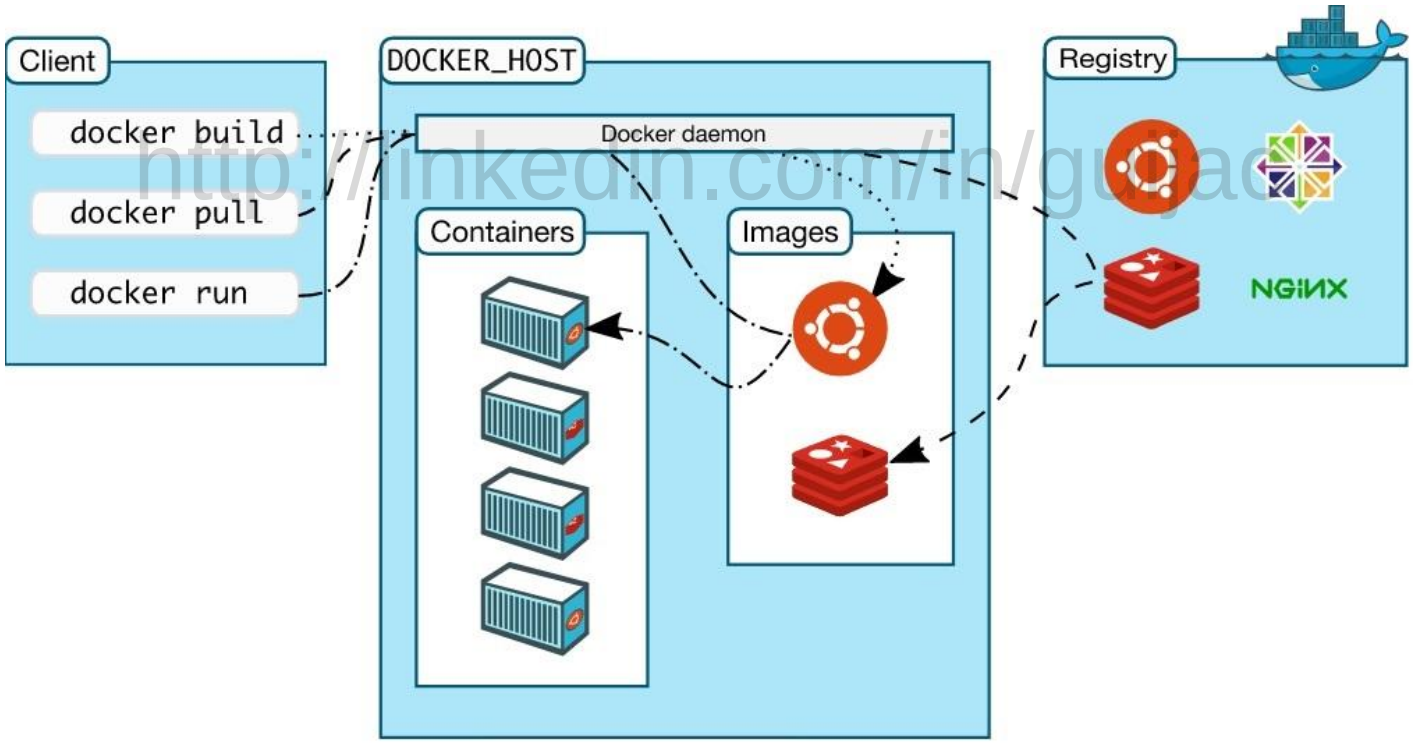
- **Docker CLI –** *command line interface*;
- **REST API –** especifica as interfaces que o daemon deve executar;
- **Server –** o servidor Docker em si, o chamado “Docker Daemon”, que cria e gerencia objetos do docker, como imagens, contêineres, redes e volumes.



Fonte: Basic Concepts - Docker Handbook (gitbook.io)

Arquitetura do Docker

- **Docker Registry –** também conhecido como Docker Hub, é aqui que suas imagens de contêiner são armazenadas e recuperadas.



Fonte: Basic Concepts - Docker Handbook (gitbook.io)

Referências Bibliográficas

BOROSAN, Payamam. **Docker Handbook**. Disponível em https://borosan.gitbook.io/docker-handbook/. Acesso em 18 jan 2025;

DANIEL, João Francisco Lino. **MAC0475 2021 - Aula05.1 - Docker + Docker-Compose**. Disponível em https://uclab.xyz/sistemas-complexos-2021-aula05-1. Acesso em 18 jan 2025;

DOCKER. **Docker Docs**. Disponível em https://docs.docker.com/. Acesso em 18 jan 2025;

HERNANDEZ, Matt. **Using Docker in WSL 2**. Disponível em https://code.visualstudio.com/blogs/2020/03/02/docker-in-wsl2. Acesso em 18 jan 2025;

MICROSOFT. **Introdução aos contêineres remotos do Docker no WSL 2**. Disponível em https://learn.microsoft.com/pt-br/windows/wsl/tutorials/wsl-containers. Acesso em 18 jan 2025;

RED HAT. **O que é um container Linux?** Disponível em https://www.redhat.com/pt-br/topics/containers/whats-a-linux-container. Acesso em 18 jan 2025;

TSA CARGO. **Transporte de container: o que você precisa saber?** Disponível em https://www.tsacargo.com.br/blog/transporte-de-container-o-que-voce-precisa-saber. Acesso em 18 jan 2025.

Por hoje (de teoria!) é só!

Fonte: Understanding and Building Docker Images (jfrog.com)

Prof. Esp. Guilherme Jorge Aragão da Cruz

guilherme.cruz@alumni.usp.br

linkedin.com/in/guijac

Licença

Prof. Esp. Guilherme Jorge Aragão da Cruz


guilherme.cruz@alumni.usp.br


linkedin.com/in/guijac