



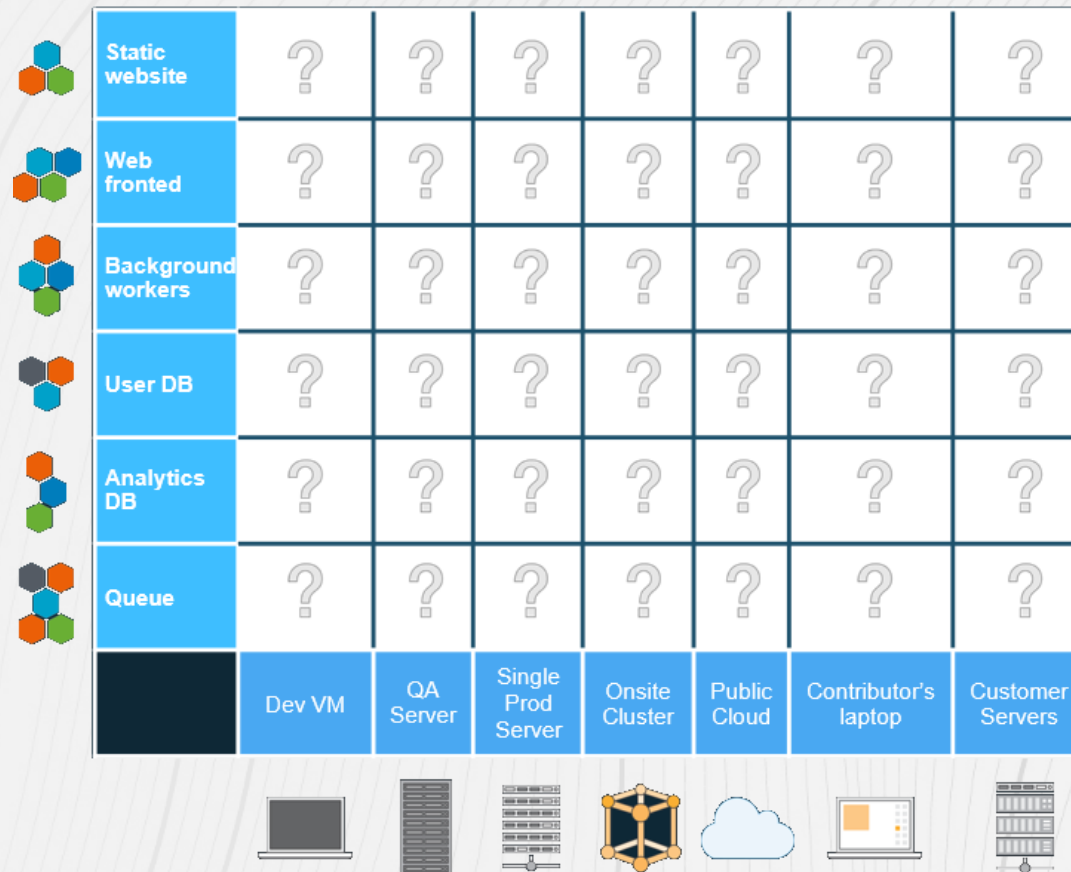
# Introdução a Containers

Vinícius Lima

[vinicius@infomach.com.br](mailto:vinicius@infomach.com.br)

## ○ Problema

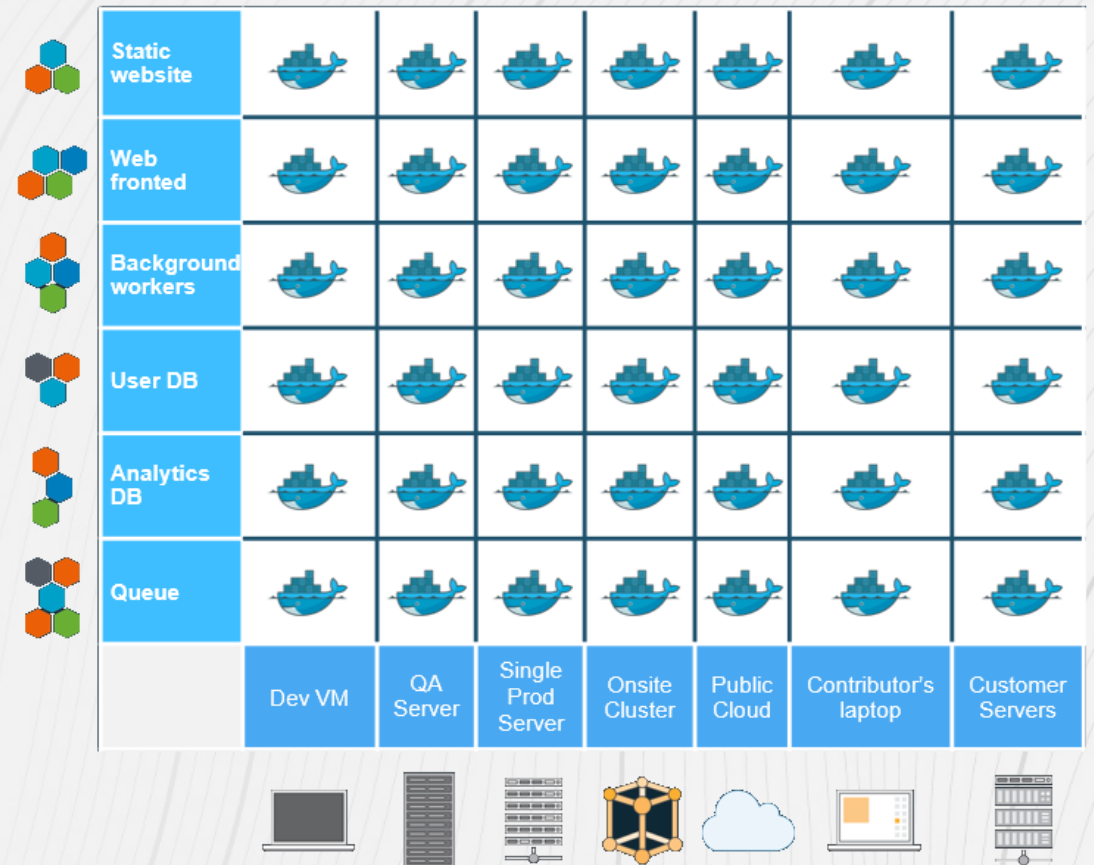
- Diferentes camadas de aplicação.
- Diferentes ambientes para deploy (hardware).
- Como executar todas as aplicações em diferentes ambientes?
- Como migrar de forma fácil de um ambiente para outro?



Static website	?	?	?	?	?	?	?
Web fronted	?	?	?	?	?	?	?
Background workers	?	?	?	?	?	?	?
User DB	?	?	?	?	?	?	?
Analytics DB	?	?	?	?	?	?	?
Queue	?	?	?	?	?	?	?
	Dev VM	QA Server	Single Prod Server	Onsite Cluster	Public Cloud	Contributor's laptop	Customer Servers

# A Solução

- Unidade na entrega de software.
- Leve, portátil e consistente.
- Implanta e executa em qualquer lugar.
- Implanta e executa qualquer coisa.



# O que é um container?

---

- Um pacote executável leve e independente de software que inclui todas as dependências: código, runtime, ferramentas de sistema, bibliotecas de sistema, configurações.
- Containers isolam o software das suas remediações.
- Ajuda a reduzir conflitos entre times que executam diferentes aplicações em uma mesma infraestrutura.
- Docker simplificou a criação, gerencia e operação de containers.

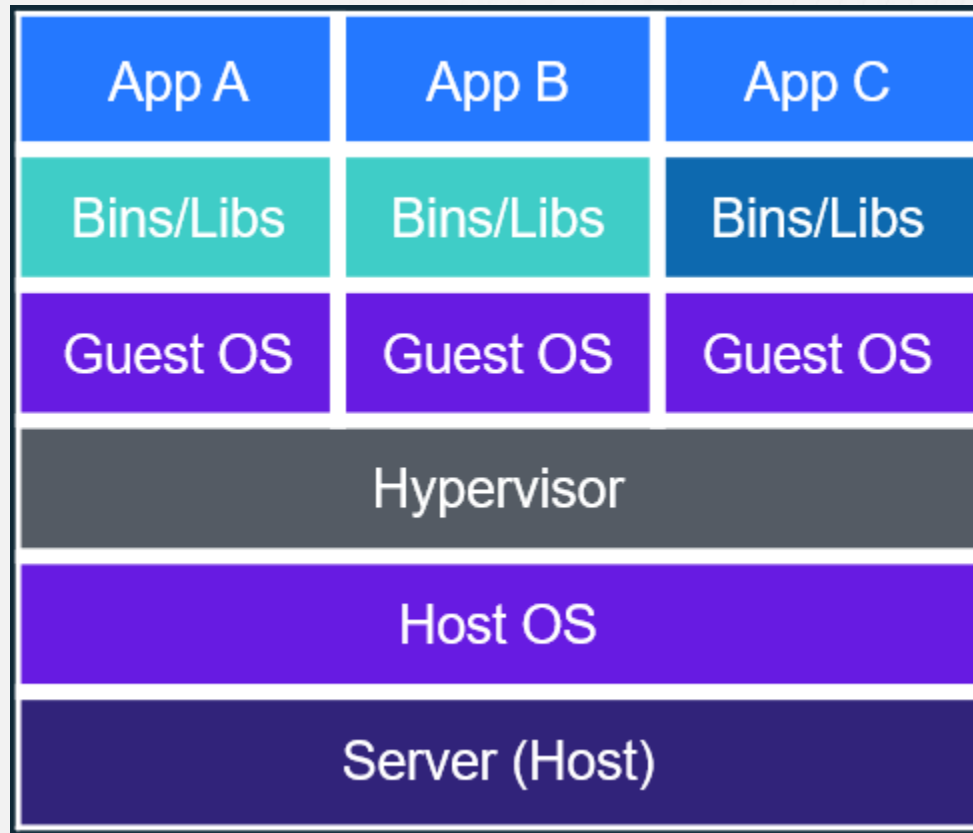


# O que é um container?

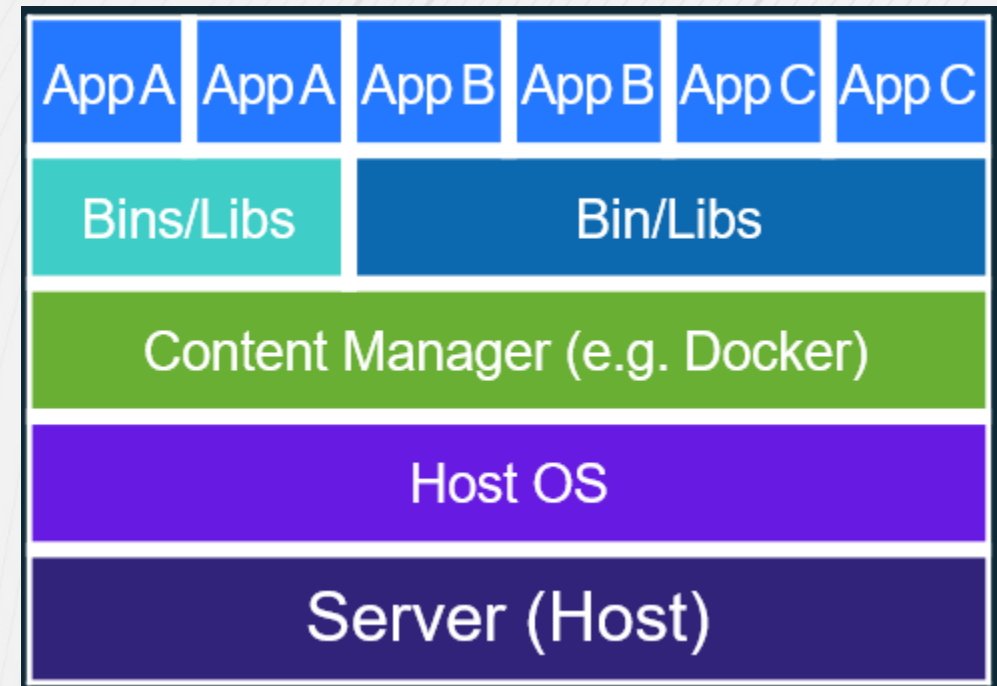
---

- Containers compartilham o kernel do SO da máquina host.
- Iniciam instantaneamente e consomem menos memória RAM.
- Imagens são construídas a partir de camadas de sistemas de arquivos, compartilhando arquivos em comum. Isso diminui o consumo de disco o download de imagens é mais rápido.

# VMs vs Containers



VMs



Containers

# Casos de uso

---

- Ambientes consistentes entre Desenvolvimento & Produção.
- Continuous Integration and Deployment (CI/CD).
- Arquiteturas Orientadas a Serviços (SOA) / Microserviços.
- Tarefas com ciclo de vida curto.
- Modernização de aplicações.

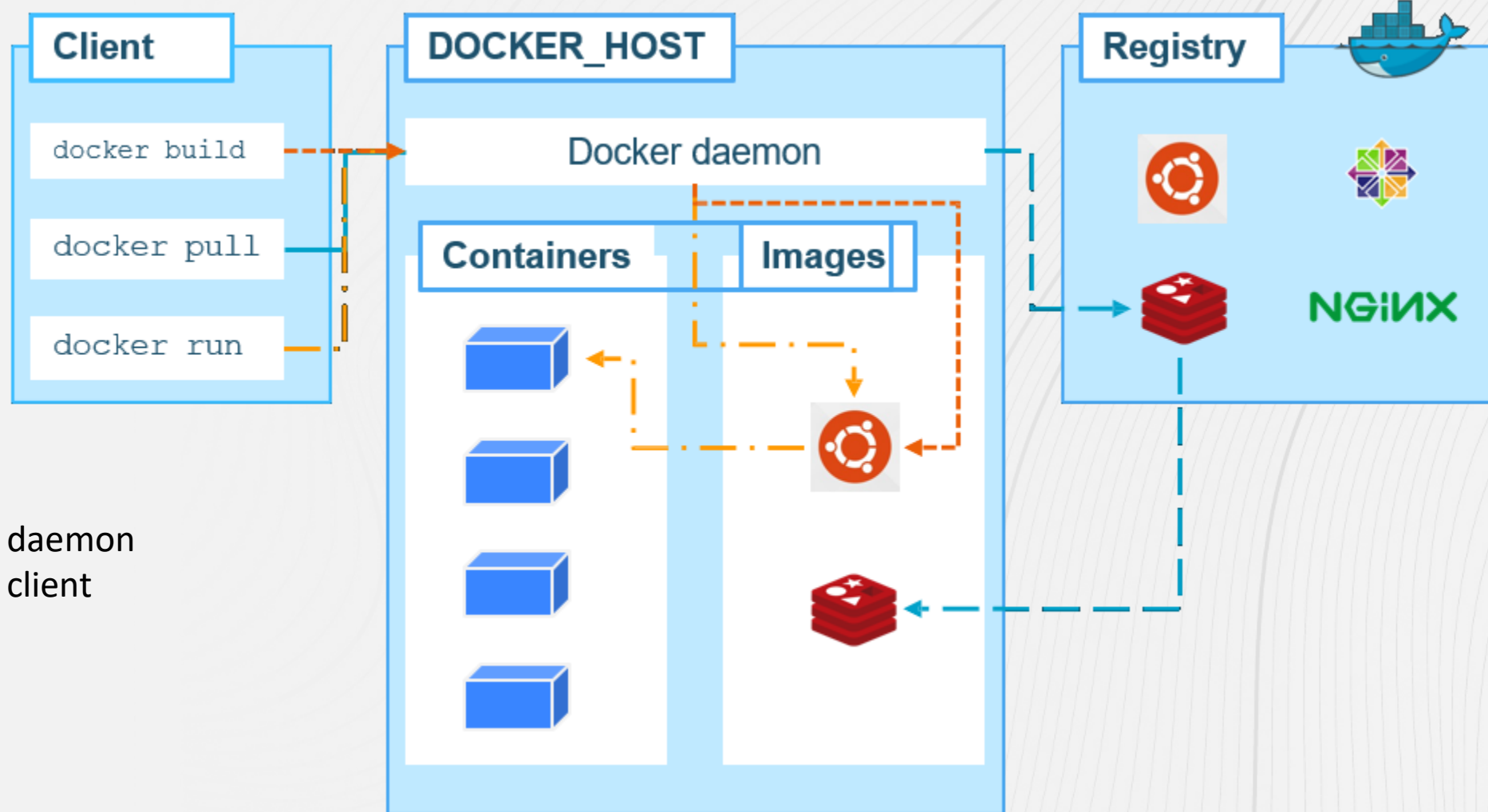
# Docker

---

- Plataforma leve para virtualização de containers.
- Conjunto de ferramentas para gerencia e deploy das aplicações.
- Licenciado sob a licença Apache 2.0.
- Construído pelas Docker, Inc.
- Plataforma Docker = Docker Engine + Docker Hub



# Docker Engine



Docker daemon  
Docker client

# Registro de containers – Docker Hub

---

- Registro Docker em nuvem.
- Biblioteca de imagens públicas (repositórios oficiais).
- Armazenamento para imagens privadas (repositórios públicos e privados).
- Criação de imagens automatizado.
- Amazon EC2 Container Registry (Amazon ECR).

# Registro de containers – Docker Hub

<https://hub.docker.com>

<username>/foobar ← Repositório mantido por um usuário

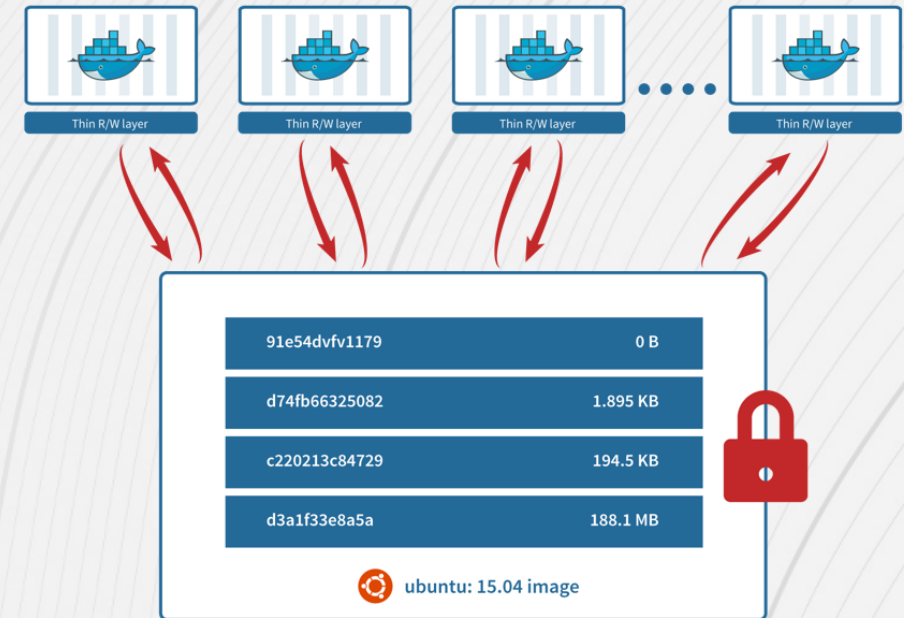
ubuntu:14.04 ← Repositório mantido pelo Docker

↑  
Image name

↙  
Image tag

# Docker Image

- Template apenas de leitura.
- Usado para criar containers.
- Une sistemas de arquivos para combinar diferentes camadas em uma única imagem.
- Imagens são criadas a partir de uma base.
- Instruções escritas no Dockerfile.





# Docker Image

---

- Criamos um Dockerfile com os passos necessários para criar e executar uma imagem.
- Algumas instruções criam camadas na imagem.
- Quando modificamos o Dockerfile e recriamos a imagem, apenas as camadas que foram modificadas são recriadas.



# Pull de imagem do Docker Hub

```
# Pegar a imagem mais recente do Ubuntu
```

```
$ docker pull ubuntu
```

```
# Pegar uma versão específica da imagem
```

```
$ docker pull ubuntu:14.04
```

```
# Listar imagens locais
```

```
$ docker images
```

# Criar uma imagem a partir do Dockerfile

```
FROM ubuntu:14.04
RUN apt-get update
RUN apt-get install -y redis-server
EXPOSE 6379
```

**FROM <image>:<tag>**

Imagem base. Deve ser o primeiro comando do Dockerfile

**RUN <command>**

Executa qualquer comando em uma nova camada em cima da imagem atual e salva o resultado

**EXPOSE <port>**

Informa o Docker que o container escuta em uma porta específica na rede durante a execução

# Criar uma imagem a partir do Dockerfile

`CMD ["exec", "param1", "param2"]`

Configura um comando para executar quando lançar um container

`ENTRYPOINT ["exec", "param1", "param2"]`

Configura um container para rodar como um executável

`ENV <key> <value>`

Configura uma variável de ambiente <key> com o valor <value>

`COPY <src> <dest>`

Copia novos arquivo ou diretórios de <src> e os adiciona no sistema de arquivos do container no caminho <dest>

`ADD <src> <dest>`

Faz o mesmo que COPY, mas também aceita URLs de arquivos como <src>

`VOLUME <path>`

Cria um ponto de montagem com um nome específico

# Criar uma imagem a partir do Dockerfile

# Criar uma imagem e usar uma tag

```
$ docker build -t <username>/myredis:v1 /path/to/Dockerfile
```

# Entrar no Docker Hub

```
$ docker login
```

# Publicar a imagem no Docker Hub

```
$ docker push <username>/myredis:v1
```



# Executar um container

# Rodar um container de uma imagem Ubuntu e executar o bash

```
$ docker run -i -t --name my_container ubuntu /bin/bash
```

# Listar containers em execução

```
$ docker ps
```

# Rodar um container de uma imagem Ubuntu e executar um loop infinito

```
$ docker run -d --name my_deamon_container ubuntu  
/bin/bash -c "while true; do echo hello world; sleep 1; done"
```

# Visualizar logs de um container

```
$ docker logs -f my_deamon_container
```

# Retorna informações sobre um container

```
$ docker inspect my_container
```



# Executar um container

```
# Mapeia a porta 6379 no container para a porta 45000 no host
$ docker run -d -p 45000:6379 <username>/myredis:v1 redis-cli
```

```
# Monta um diretório do host em um local específico no container
e o dá permissão apenas para leitura
$ docker run -d -p 80 --name mywebsite
-v /home/user/mywebsite:/var/www/html/website:ro <username>/nginx nginx
```

```
# Rodar um container PostgreSQL
$ docker run -d --name my_postgres_db postgres
# Rodar um container de aplicação web com um "link" para o container PostgreSQL
$ docker run -d -P --name my_webapp
--link my_postgres_db:db <username>/webapp python app.py
```

Documentação Docker

<https://docs.docker.com/reference/>

Tópicos avançados:

Docker compose

Docker swarm

Curso Udemy:

Docker Mastery: with Kubernetes + Swarm from a  
Docker Capitan





---

Vinícius Lima  
[vinicius@infomach.com.br](mailto:vinicius@infomach.com.br)