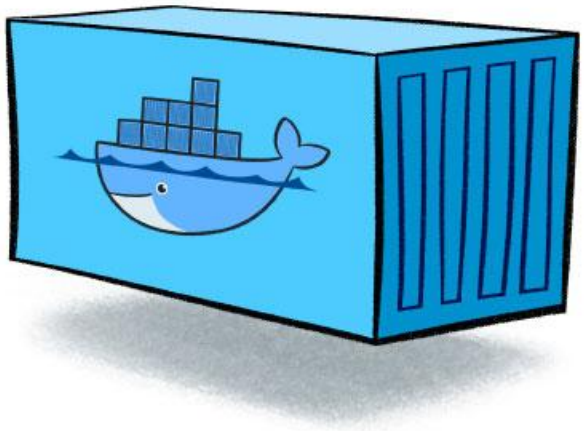


Apresentação

Docker

A plataforma que os Devs amam

Time:

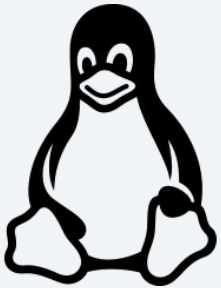
Eliabe Leal - TSI

Fabrício Liberato - TSI

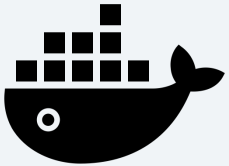
Luiz Henrique - Assert

PESQUISA STACK OVERFLOW

83.1%



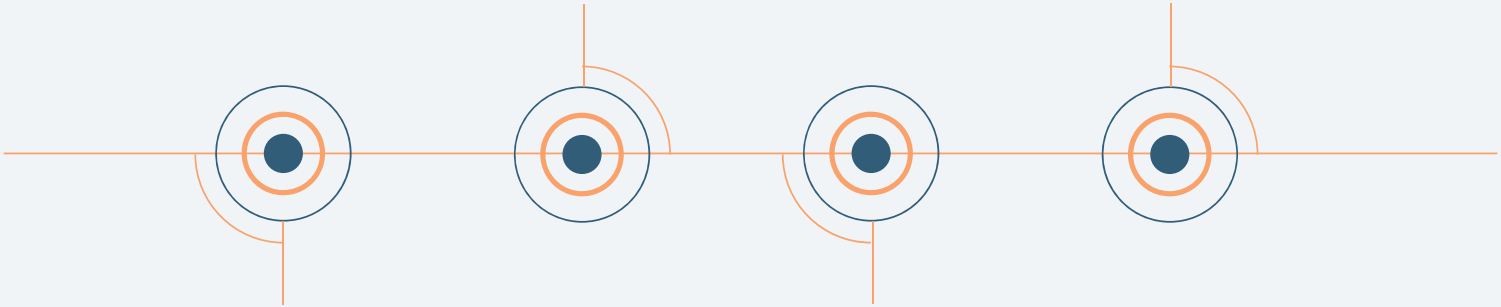
77.8%



76.8%

Fonte: https://insights.stackoverflow.com/survey/2019#technology-_-most-loved-dreaded-and-wanted-platforms

HISTÓRIA DOS CONTAINERS



Unix V7

1979

O chroot foi introduzido, mudando o diretório raiz de um processo e seus filhos para um novo local no sistema de arquivos.

Jails - FreeBSD

2000

Um pequeno provedor de hospedagem surgiu com o Jails do FreeBSD, para conseguir uma separação clara entre os seus serviços e os de seus clientes para fins de segurança e facilidade de administração.

Process Containers

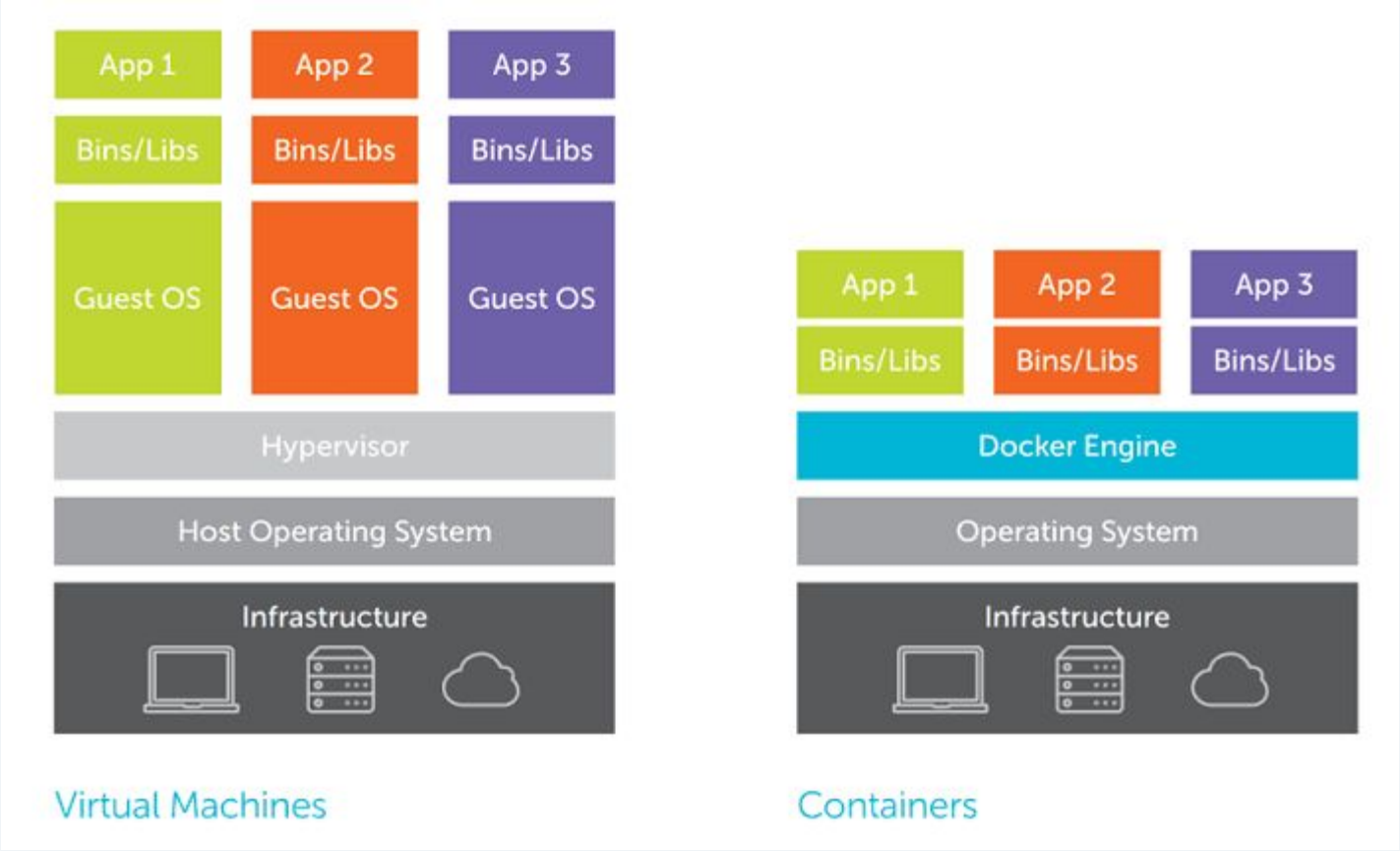
2006

Despite being red, Mars is a cold place, not hot. It’s full of iron oxide dust

2013

Quando o Docker surgiu em 2013, os conteiners explodiram em popularidade.

VM X Container



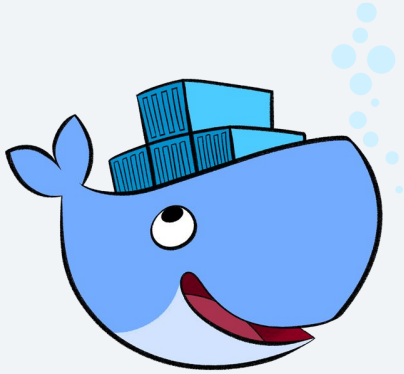
O QUE É UMA IMAGEM E UM CONTAINER?

DOCKERFILE

build

DOCKER IMAGE

run



DOCKER CONTAINER

COMANDOS

PULL

docker pull hello-world

RUN

docker run hell-world

IMAGES

docker images

PS

docker ps

BUILD

Build an image from a Dockerfile

REMOVER

docker rm CONTAINER
docker rmi IMAGE

docker run

- O comando docker run tem como primeira ação criar um container a partir de uma imagem específica, e então inicia o container com o comando específico. Simplificando, o docker run usa:
  - docker create
  - docker start

```
docker run [OPTIONS] IMAGE [COMMAND] [ARG...]
```

Reference: https://docs.docker.com/engine/reference/commandline/run/

docker run - Options

- **--name** para adicionar um nome ao container (Serve como idenficador);
- **--rm** para apagar o container após sua execução;
- **-v** para realizar a ligação entre duas pastas (volumes)
- **-p** para configurar portas disponíveis no host e dentro de sua rede;
- **--expose** para configurar portas disponíveis apenas para o host;
- **-w** para para configurar o diretório de trabalho dentro do container;
- **-it** para abrir um terminal interativo;
- **-e** para adicionar uma variável de ambiente.

Reference: https://docs.docker.com/engine/reference/commandline/run/

Docker Hub

- https://hub.docker.com/
- Docker Hub é o caminho mais fácil do mundo para criar, gerenciar e entregar as imagens;
- Repositório de imagens;
- Tags.

Docker Hub - Postgres

- Pesquisar Postgres no Docker Hub;
- docker run;
- docker ps;
- docker exec;
- docker run novamente;
- docker stop/start;
- docker rm.

Docker Hub - Python

- Pesquisar Python no Docker Hub;
- Executar o interpretador do python;
- Executar um script python.

Exercício - Docker

- Pesquisar o Node no Docker Hub;
- Executar o interpretador do node;
- Executar um script JS.

Dockerfile

a/Dockerfile

```
1  FROM python:3.5
2
3  WORKDIR /home/pokedex
4
5  COPY requirements.txt requirements.txt
6  RUN pip install -r requirements.txt
7
8  COPY .flaskenv .flaskenv
9  COPY templates/ templates/
10 COPY static/ static/
11 COPY app.py app.py
12
13 EXPOSE 5000
14 CMD ["flask", "run", "--host", "0.0.0.0"]
```

Dockerfile

- Um arquivo de texto com um conjunto de instruções com a finalidade de construir uma imagem;
- Essa imagem pode ser usada para criar e executar um aplicativo em um ou mais containers sem a necessidade de se preocupar com a instalação e/ou depedências;

Dockerfile - Comandos Mais Comuns

- **FROM:** Define uma imagem base para sua imagem.
- **WORKDIR:** Criar um novo diretório onde estará as todas as configurações da sua aplicação.
- **RUN:** Executa qualquer comando e comitta os resultados. O resultado é uma nova imagem que será usada no próximo comando do Dockerfile.
- **COPY:** Copia arquivos e/ou pastas do ambiente local para o filesystem do container.
- **EXPOSE:** A instrução EXPOSE informa ao Docker que o contêiner escuta nas portas de rede especificadas em tempo de execução.

Dockerfile - Comandos Mais Comuns

- **CMD:** O principal objetivo é fornecer padrões para um contêiner em execução. Esses padrões podem incluir um executável ou podem omitir o executável.

Dockerfile - docker build

- Depois de definir os comandos no Dockefile, podemos executar o comando docker build para construir uma imagem apontando para o caminho onde está o Dockerfile. Também podemos usar as seguintes opções:
  - **-t:** Definimos um nome ou **tag** para referenciamos a imagem em vez do id da imagem gerada.
  - **-f:** O nome do Dockerfile

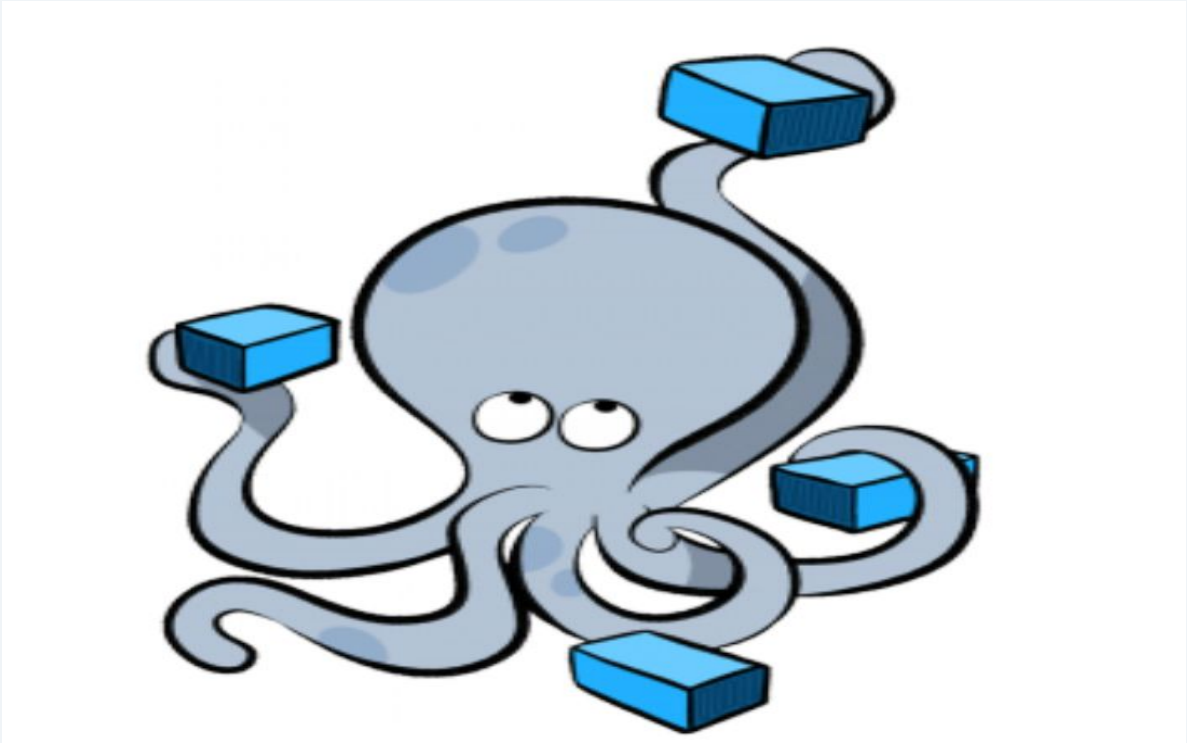
```
docker build [OPTIONS] PATH | URL | -
```

Dockerfile - Executando o container

- Para executarmos o container baseado na nova image, executamos o docker run apontando para o id da imagem ou tag(no exemplo abaixo a tag da imagem é pokeset):

```
docker run -d -p 3000:3000 --name pokeset_app pokeset
```

Docker-compose



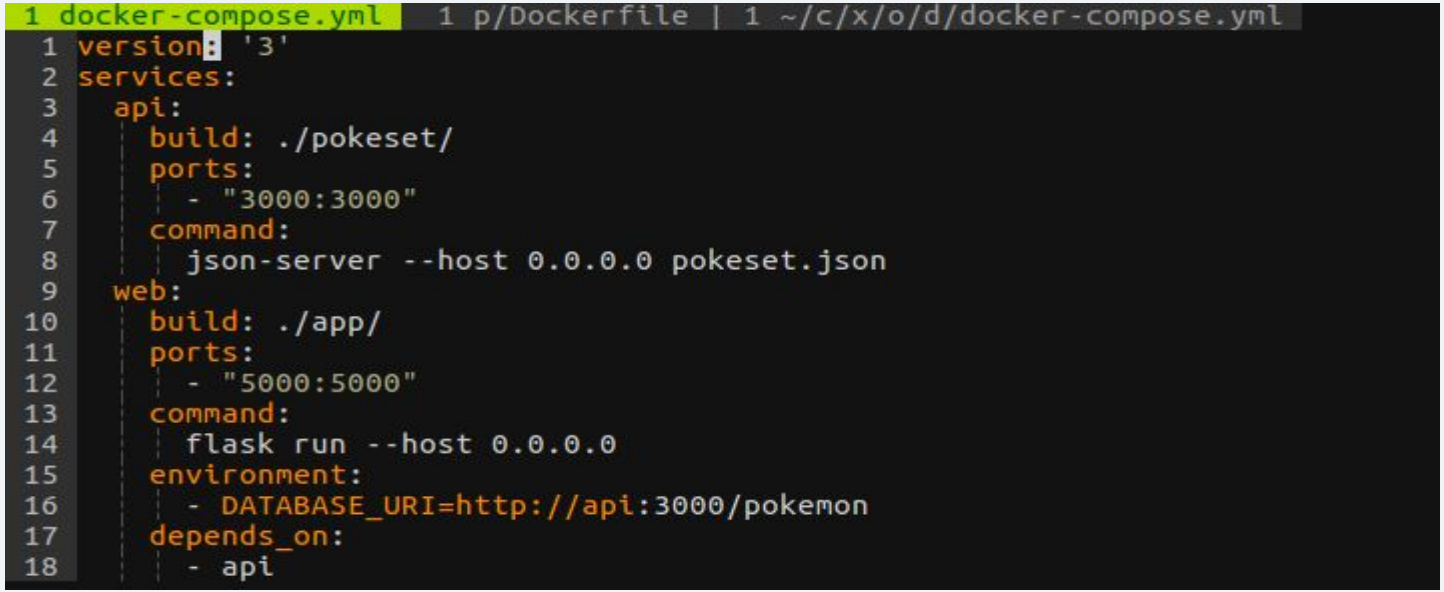
Docker-compose

- Geralmente com o aumento do número de containers em execução, fica evidente a necessidade de um melhor gerenciamento da sua comunicação

- O Docker Compose é uma ferramenta desenvolvida para gerenciar um maior número de containers.

Docker-compose - docker-compose.yml

- Arquivo de definição usado pelo Docker Compose para definirmos os containers como serviços e suas configurações como portas expostas, variáveis de ambiente, etc:



Docker-compose - docker-compose up

- O comando docker-compose up é usado para criar, executar e linkar um ou mais containers a partir de um docker-compose.yml definido. As opções mais usadas são:
  - **-d:** Executar os containers em background.
  - **-t timeout:** estabele timeout em segundos para o encerramento dos containers (default: 10).

```
up [options] [--scale SERVICE=NUM...] [SERVICE...]
```

Desafio

Compose and Django - https://docs.docker.com/compose/django/