Docker

O que é Docker?

O Docker é uma plataforma de software que facilita a criação, implantação e execução de aplicações usando contêineres. Contêineres são como pacotes padronizados que incluem tudo o que um software precisa para funcionar: o código, o tempo de execução (runtime), bibliotecas, variáveis de ambiente e arquivos de configuração.

Principais Conceitos

- **Contêiner:** Uma instância de uma imagem. É um ambiente isolado e leve onde sua aplicação é executada.
- **Imagem (Image):** Um modelo de "somente leitura" que contém as instruções para criar um contêiner. É a base para a criação de um ou mais contêineres.
- **Dockerfile:** Um arquivo de texto que contém uma série de instruções para construir uma imagem Docker. É o "receita" para a imagem.
- **Docker Hub:** Um registro online e público de imagens Docker. É como um "GitHub" para imagens, onde você pode encontrar e compartilhar imagens.
- **Docker Compose:** Uma ferramenta para definir e executar aplicações multi-contêineres. Permite gerenciar todos os serviços de uma aplicação em um único arquivo de configuração.

Prática: Construindo sua Primeira Aplicação Docker

Passo 1: Verifique a instalação do Docker

Abra o terminal ou prompt de comando e digite:

Bash

```
docker --version
```

Isso mostrará a versão do Docker instalada, confirmando que ele está pronto para uso.

Passo 2: Entendendo os comandos básicos

- docker pull [nome-da-imagem]: Baixa uma imagem de um registro.
- docker images: Lista as imagens que você tem localmente.
- docker run [nome-da-imagem]: Cria e executa um novo contêiner a partir de uma imagem.
- docker ps: Mostra os contêineres em execução.
- docker ps -a: Mostra todos os contêineres (em execução ou parados).
- docker stop [id-do-contêiner]: Para a execução de um contêiner.
- docker start [id-do-contêiner]: Inicia um contêiner parado.
- docker rm [id-do-contêiner]: Remove um contêiner.
- docker rmi [id-da-imagem]: Remove uma imagem.

Passo 3: Executando um contêiner simples (Exemplo: Nginx)

Vamos executar um servidor web Nginx em um contêiner. O Docker fará o download da imagem do Docker Hub automaticamente, se você não a tiver.

`docker run -d -p 8080:80 --name meu-nginx nginx`

- docker run: Comando para criar e rodar um contêiner.
- -d: Roda o contêiner em modo "detached" (em segundo plano).
- -p 8080:80: Mapeia a porta 8080 do seu computador para a porta 80 do contêiner. Isso permite que você acesse o servidor web.
- --name meu-nginx: Atribui um nome para o seu contêiner.
- nginx: O nome da imagem que o Docker vai usar.

Agora, abra seu navegador e acesse `http://localhost:8080`. Você verá a página de boas-vindas do Nginx.

Para parar o contêiner:

`docker stop meu-nginx`

Para remover o contêiner:

`docker rm meu-nginx`

Passo 4: Criando sua própria imagem com um Dockerfile

Vamos criar um pequeno aplicativo web em Python e empacotá-lo em uma imagem.

1. Crie uma pasta para o projeto e entre nela:

`mkdir minha-app-docker`

`cd minha-app-docker`

2. Crie um arquivo chamado app.py com o seguinte conteúdo:

Python

```
from flask import Flask

app = Flask(__name__)


@app.route('/')

def hello_world():

    return '<h1>Olá do meu Contêiner Docker!</h1>'


if __name__ == '__main__':

    app.run(host='0.0.0.0', port=5000)
```

3. Crie um arquivo chamado requirements.txt com o conteúdo:

Flask

4. Crie um arquivo chamado Dockerfile (sem extensão) com as instruções para a imagem:

Dockerfile

```
# Use uma imagem base do Python

FROM python:3.9-slim


# Defina o diretório de trabalho no contêiner

WORKDIR /app


# Copie os arquivos de requisitos para o diretório de trabalho

COPY requirements.txt .


# Instale as dependências

RUN pip install --no-cache-dir -r requirements.txt


# Copie o código da sua aplicação
```

```
COPY . .

# Expõe a porta em que a aplicação será executada
EXPOSE 5000

# Comando para rodar a aplicação quando o contêiner for iniciado
CMD ["python", "app.py"]
```

Passo 5: Construindo a imagem e rodando o contêiner

Agora, no mesmo diretório onde estão os arquivos, execute o comando para construir a imagem.

```
docker build -t minha-app-python .
```

- docker build: Comando para construir uma imagem a partir de um Dockerfile.
- -t minha-app-python: Atribui um nome (tag) para sua nova imagem.
- .: Indica que o Dockerfile está no diretório atual.

Depois de construída, sua imagem estará na lista de imagens. Para verificar: docker images

Finalmente, rode sua aplicação em um contêiner:

```
docker run -d -p 5000:5000 --name meu-app-python minha-app-python
```

Acesse http://localhost:5000 no seu navegador e você verá a mensagem "Olá do meu Contêiner Docker!".

Para parar e remover o contêiner, use os comandos docker stop e docker rm.