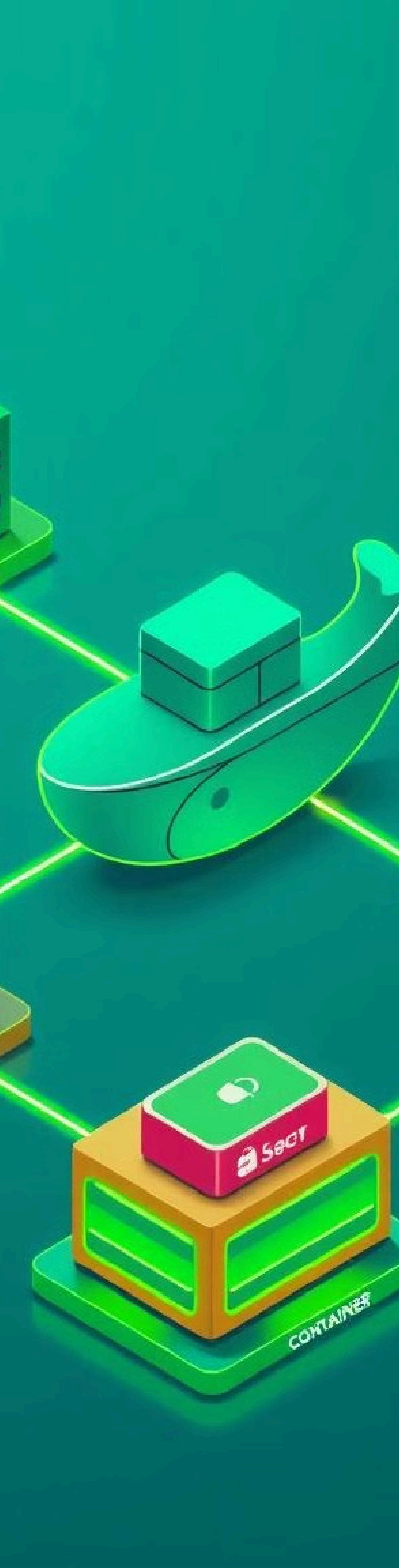


Docker Compose: Guia Completo

Este guia completo sobre Docker Compose aborda desde conceitos básicos até boas práticas e considerações de segurança. Ele explica como definir e gerenciar ambientes multi-container Docker de forma declarativa, utilizando um arquivo YAML para orquestrar serviços. O documento inclui exemplos práticos, comandos essenciais e um exercício para consolidar o aprendizado.

por Everton Coimbra de Araújo



Introdução ao Docker Compose

Docker Compose é uma ferramenta que permite definir e gerenciar ambientes multi-container Docker de forma declarativa. Com Docker Compose, você pode definir um ambiente multi-container em um único arquivo YAML, facilitando a orquestração e automação dos serviços que compõem a sua aplicação. É especialmente útil para configurar, iniciar e gerenciar aplicações que dependem de múltiplos containers, como um servidor web, um banco de dados e um serviço de cache.

Conceitos Básicos do Docker Compose

Serviços

Cada container em uma aplicação Docker Compose é definido como um serviço. Um serviço pode ser um banco de dados, um servidor web ou qualquer outro componente da sua aplicação.

Volumes

Volumes são usados para persistir dados entre os containers e o host. Eles permitem que os dados não sejam perdidos quando os containers são removidos ou recriados.

Redes

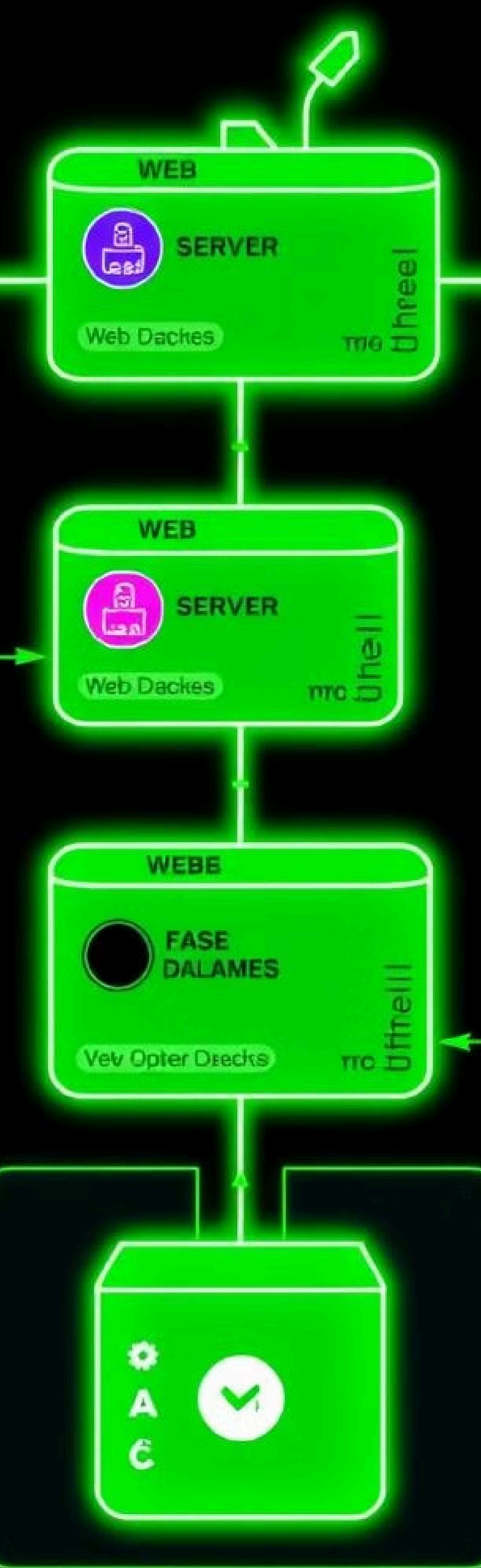
Docker Compose configura redes para permitir que os serviços se comuniquem entre si. Por padrão, os serviços na mesma rede podem se comunicar usando seus nomes de serviço.

Estrutura de um Arquivo docker-compose.yml

O arquivo docker-compose.yml é onde você define a configuração da sua aplicação. Aqui está um exemplo básico:

```
version: '3'
services:
  web:
    image: nginx:latest
    ports:
      - "8082:80"
    volumes:
      - ./web:/usr/share/nginx/html
    networks:
      - mynetwork
  database:
    image: mysql:5.7
    environment:
      MYSQL_ROOT_PASSWORD: example
    volumes:
      - db_data:/var/lib/mysql
    networks:
      - mynetwork
networks:
  mynetwork:
volumes:
  db_data:
```

O código acima define dois serviços, **web** e **database**, que utilizam o servidor web **Nginx** e o banco de dados **MySQL**, respectivamente. Além disso, cria uma rede chamada **mynetwork** para conectar os dois serviços, e um volume **db_data** para persistir os dados do MySQL, garantindo que eles não sejam perdidos se o container for removido ou recriado. Veja a seguir a descrição dos componentes do arquivo.



Descrição do Arquivo docker-compose.yml

- **version:** Define a versão do Docker Compose. Aqui usamos a versão 3.
- **services:** Define os serviços que compõem a aplicação.
 - **web:** Serviço para o servidor web Nginx.
 - **image:** Imagem Docker a ser usada.
 - **ports:** Porta mapeada do host para o container.
 - **volumes:** Volume montado do host para o container.
 - **networks:** Rede na qual o serviço está conectado.
 - **database:** Serviço para o banco de dados MySQL.
 - **image:** Imagem Docker a ser usada.
 - **environment:** Variáveis de ambiente para o serviço.
 - **volumes:** Volume montado do host para o container.
 - **networks:** Rede na qual o serviço está conectado.
- **networks:** Define as redes utilizadas pelos serviços.
- **volumes:** Define os volumes utilizados pelos serviços.

Comandos Básicos do Docker Compose

Os comandos básicos do Docker Compose são essenciais para gerenciar o ciclo de vida dos serviços definidos no arquivo docker-compose.yml. Com eles, você pode iniciar, parar, construir e monitorar os containers que compõem sua aplicação de forma prática e eficiente. A seguir, veremos alguns dos principais comandos usados para interagir com ambientes multi-container, explicando o que cada um faz, o motivo de utilizá-los e o que esperar de cada operação.

docker compose up

1

O comando docker compose up cria e inicia todos os serviços definidos no arquivo docker-compose.yml. Ele é utilizado para iniciar todo o ambiente de uma aplicação com apenas um comando, facilitando a automação e a orquestração dos containers.

docker compose build --no-cache

3

O comando docker compose build constrói ou reconstrói as imagens dos serviços definidos no arquivo docker-compose.yml. Ele é utilizado principalmente quando houve alterações no código ou nas dependências dos serviços, e você precisa refletir essas mudanças ao iniciar os containers.

A opção **--no-cache** garante que a construção das imagens seja feita sem utilizar o cache de etapas anteriores, o que é útil quando se deseja garantir que tudo seja baixado e recompilado do zero, evitando a reutilização de camadas que possam estar desatualizadas.

Utilizar este comando é essencial para garantir que os containers serão executados com as versões atualizadas do código ou das dependências, evitando problemas de inconsistência no ambiente.

2

docker compose down

O comando docker compose down é utilizado para parar e remover todos os containers, redes e volumes criados com o docker compose up. Ele é importante quando você quer liberar recursos, interromper o ambiente, ou garantir que tudo seja limpo após o uso.

Testando os Serviços do Arquivo docker-compose.yml

Para testar os dois serviços definidos no arquivo docker-compose.yml — o servidor web Nginx e o banco de dados MySQL — siga os passos abaixo:

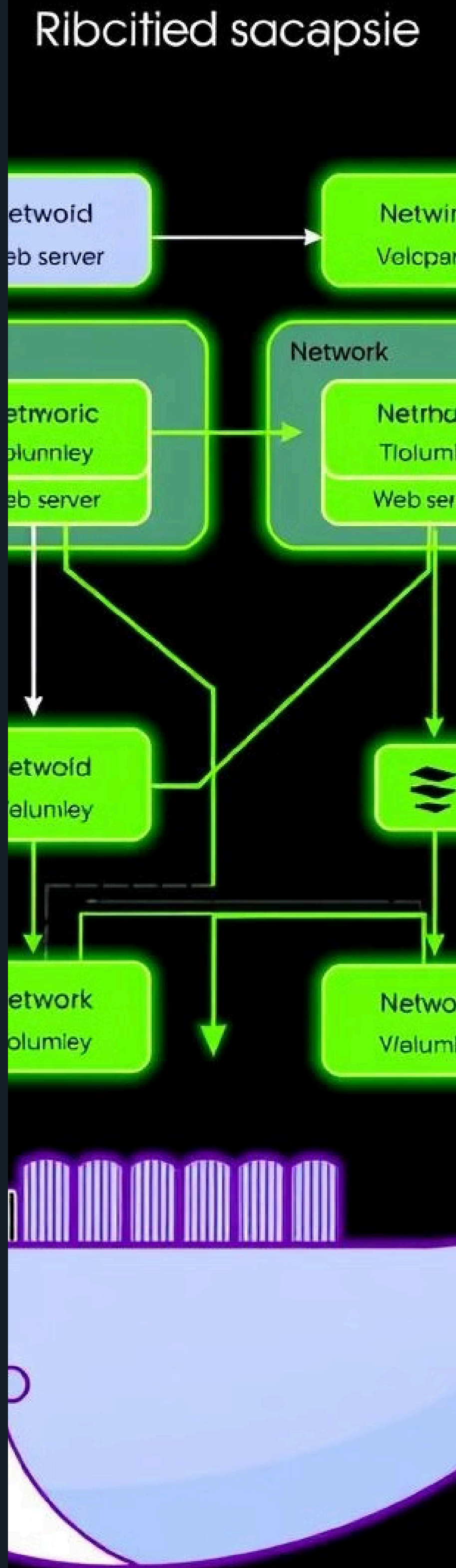
1. **Testar o Serviço Web (Nginx):**
 - Após executar o comando `docker compose up -d`, abra um navegador e acesse `http://localhost:8082`. Se tudo estiver correto, você deverá ver a página padrão do Nginx ou qualquer conteúdo que tenha colocado no diretório mapeado `./web`.
2. **Testar o Serviço de Banco de Dados (MySQL):**
 - Verifique se o container do MySQL está em execução utilizando o comando:

```
docker compose ps
```

- Em seguida, conecte-se ao container do MySQL para verificar se ele está funcionando corretamente:

```
docker exec -it mysql -p
```

Substitua pelo nome do container exibido pelo comando `docker compose ps`. Você será solicitado a inserir a senha que foi definida como **example**. Após a autenticação, você poderá executar comandos SQL para testar o funcionamento do banco de dados.



Comandos para Testar o Banco de Dados MySQL

1. Listar Bancos de Dados:

```
SHOW DATABASES;
```

Este comando permite visualizar todos os bancos de dados disponíveis no MySQL. É útil para garantir que a conexão foi bem-sucedida e o banco de dados está acessível.

1. Criar um Novo Banco de Dados:

```
CREATE DATABASE test_db;
```

Este comando cria um novo banco de dados chamado test_db, ajudando a verificar se o MySQL está permitindo operações de escrita.

1. Usar um Banco de Dados:

```
USE test_db;
```

Com este comando, você muda o contexto para o banco de dados test_db criado anteriormente.

1. Criar uma Tabela:

```
CREATE TABLE users (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  name VARCHAR(100),  
  email VARCHAR(100)  
);
```

Este comando cria uma tabela chamada users no banco de dados test_db, permitindo testar a criação de estruturas dentro do banco.

1. Inserir Dados na Tabela:

```
INSERT INTO users (name, email) VALUES ('Alice', 'alice@example.com');
```

Este comando insere um registro na tabela users, verificando se o MySQL está aceitando inserções de dados corretamente.

1. Consultar Dados da Tabela:

```
SELECT * FROM users;
```

Este comando retorna todos os registros da tabela users, permitindo verificar se os dados foram inseridos corretamente.

Boas Práticas e Considerações de Segurança

- **Uso de Imagens Oficiais:** Sempre que possível, utilize imagens oficiais do Docker Hub.
- **Gerenciamento de Senhas e Credenciais:** Use variáveis de ambiente para gerenciar senhas e credenciais, e considere usar o Docker Secrets para informações sensíveis.
- **Versão de Compose:** Utilize a versão mais recente do Docker Compose que seja compatível com sua aplicação.
- **Limpeza de Recursos:** Utilize `docker compose down --volumes` para limpar recursos quando não forem mais necessários.
- **Documentação:** Documente o arquivo `docker-compose.yml` de forma clara, incluindo descrições sobre cada serviço e suas dependências. Isso facilita a manutenção futura e permite que outros desenvolvedores entendam rapidamente a configuração e o propósito de cada componente.

Docker Compose é uma ferramenta poderosa para definir e gerenciar ambientes multi-container. Com um único arquivo YAML, você pode orquestrar diversos serviços que compõem a sua aplicação, facilitando o desenvolvimento, testes e deploy. As boas práticas e exercícios aqui apresentados ajudam a consolidar o conhecimento necessário para utilizar o Docker Compose de forma eficaz e segura.

Conclusão

Docker Compose é uma ferramenta poderosa para definir e gerenciar ambientes multi-container. Com um único arquivo YAML, você pode orquestrar diversos serviços que compõem a sua aplicação, facilitando o desenvolvimento, testes e deploy. As boas práticas e exercícios aqui apresentados ajudam a consolidar o conhecimento necessário para utilizar o Docker Compose de forma eficaz e segura.

FAQ - "Não Existem Perguntas Idiotas"

Por que usar Docker Compose em vez de comandos Docker individuais?

Docker Compose facilita a automação e o gerenciamento de ambientes multi-container. Em vez de iniciar cada container individualmente, você define todos os serviços em um único arquivo `docker-compose.yml` e os gerencia com comandos simples, economizando tempo e reduzindo erros.

Docker Compose

ser
er compose

ser

porter;
pose to
ose
s.

S

ker, serving.
nent ler detting
for
hencr emplher,



Updasterder
the asserts th

- Dockers cont
inschal pose
that fon the
the read en
poviler and

Perting

Docker composed
Docker Comnant

DOCKER SERVICE

- Docker compose is the easiest way to manage Docker containers and the Compose.
- Extended decomp big estlets
- Leased complete indertsill ere. Dockere atome docker that sponse.

Qual é a diferença entre **docker compose up** e **docker compose start**?

docker compose up cria e inicia os containers a partir do arquivo **docker-compose.yml**, enquanto **docker compose start** apenas inicia containers que já foram criados. Use **up** para iniciar o ambiente desde o início e **start** para reiniciar containers parados.

Posso usar Docker Compose em produção?

Sim, é possível usar Docker Compose em produção, especialmente para pequenos projetos ou em ambientes onde a simplicidade é desejada. No entanto, para ambientes mais complexos e com maior escalabilidade, ferramentas como Kubernetes são mais recomendadas.

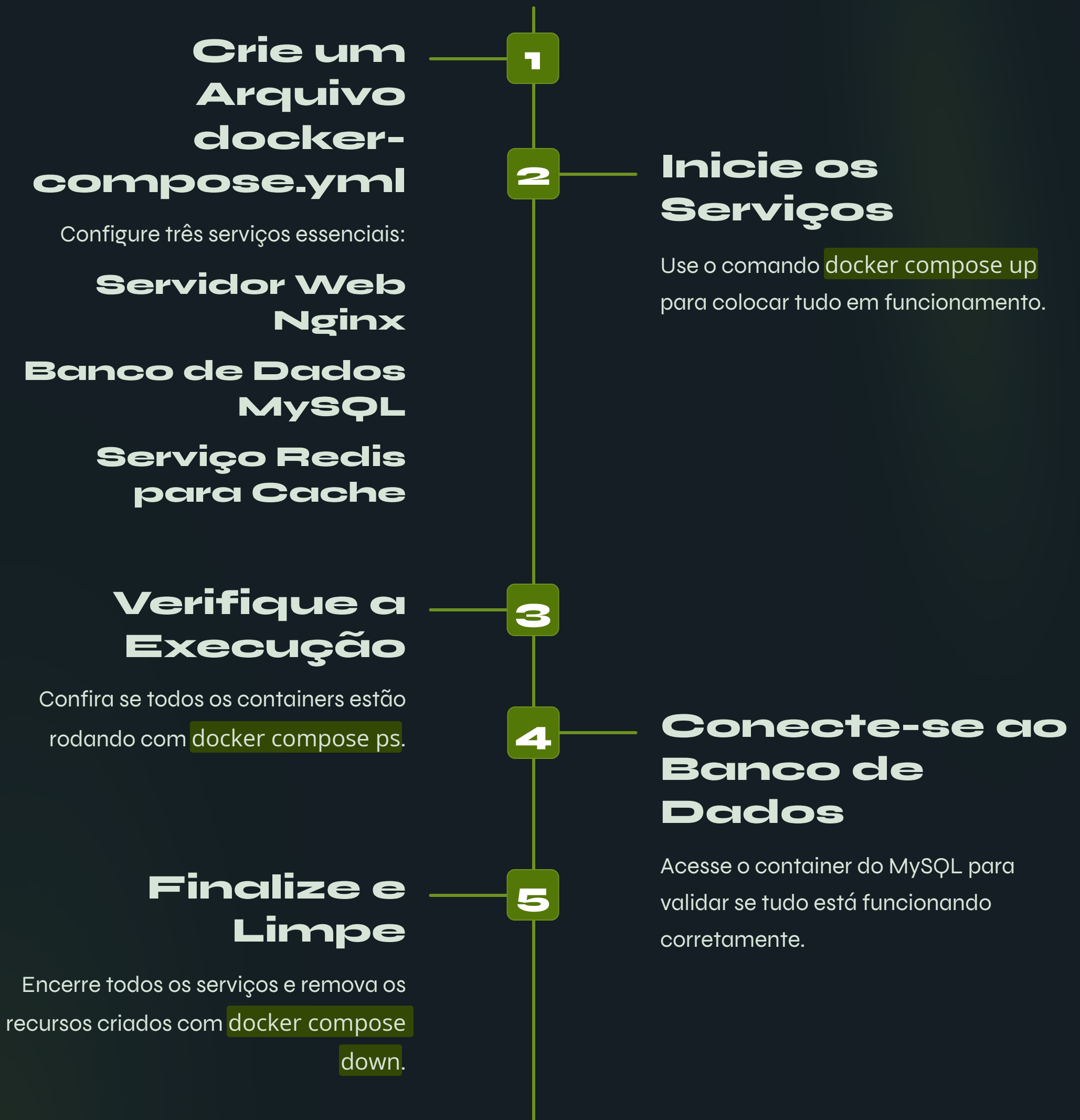
Como faço para atualizar um serviço no Docker Compose?

Para atualizar um serviço, você pode modificar o arquivo **docker-compose.yml** e executar **docker compose up -d --build**. Isso reconstruirá a imagem do serviço e reiniciará o container com a nova configuração.

Qual é a função do **--no-cache** no comando **docker compose build**?

A opção **--no-cache** força o Docker a ignorar o cache durante a construção da imagem, garantindo que todos os pacotes e dependências sejam baixados novamente. Isso é útil para garantir que as últimas versões de todas as dependências sejam utilizadas.

Exercício Prático



Respostas Comentadas

Criação do Arquivo `docker-compose.yml`

```
version: '3'
services:
  web:
    image: nginx:latest
    ports:
      - "80:80"
  database:
    image: mysql:5.7
    environment:
      MYSQL_ROOT_PASSWORD: example
  cache:
    image: redis:alpine
```

Neste arquivo, definimos três serviços (`web`, `database` e `cache`) usando imagens oficiais. Cada serviço é configurado com suas respectivas imagens e, no caso do MySQL, usamos uma variável de ambiente para definir a senha root.

Iniciar os Serviços

```
docker compose up -d
```

O comando inicia todos os serviços em segundo plano, garantindo que a aplicação esteja em execução e permitindo continuar utilizando o terminal para outros comandos.

Verificar Containers em Execução

```
docker compose ps
```

O comando lista todos os containers que estão em execução, confirmando que todos os serviços foram iniciados corretamente.

Conectar-se ao Container do MySQL

```
docker exec -it <nome_do_container_mysql> mysql -p
```

Utilizamos o comando `docker exec` para acessar o container do MySQL e verificar se ele está funcionando corretamente. Substitua `<nome_do_container_mysql>` pelo nome real do container exibido pelo `docker compose ps`.

Finalizar e Limpar os Serviços

```
docker compose down --volumes
```

Este comando para todos os containers e remove os volumes associados, garantindo que não haja resíduos de dados dos testes realizados.

Docker Compose Command

or staptivadl the

- Eptarits of the a

compasse waitlting

ker cther and

flet morh
comnanly pacitions.

kestonitality tine

Gondatter conmad.s.

- Shep Dockffer C
Folcter is Becker
Plepcet lint, teet

SEPERTIHC: COMPOSE ENJLURI



UPDAT DOCKER COMPOSE

