



Certified Tech Developer

The Ultimate Degree

Objetivos

Construir um container com Docker para hospedar um projeto Front End.

O que vamos construir?

Vamos fazer um container com o S.O Ubuntu e instalar o Apache em seguida vamos clonar um projeto Front End e hospedar e acessar o projeto pelo navegador.

Instruções

Abra o Play with Docker <https://labs.play-with-docker.com/> ou se tiver o Docker instalado no seu computador abra seu terminal preferido.

Digite o comando, para baixarmos uma imagem do Ubuntu para nosso Docker:

```
docker image pull ubuntu
```

Teremos uma saída semelhante a esta:

```
Using default tag: latest
latest: Pulling from library/ubuntu
125a6e411906: Pull complete
Digest: sha256:26c68657ccce2cb0a31b330cb0be2b5e108d467f641c62e13ab40cbec258c68d
Status: Downloaded newer image for ubuntu:latest
docker.io/library/ubuntu:latest
```

Significa que a imagem foi baixada corretamente, podemos verificar utilizando o comando:

```
docker image ls
```

O Docker listará todas as imagens existente localmente:

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
ubuntu	latest	d2e4e1f51132	2 weeks ago	77.8MB

Agora vamos executar um comando para criar e já executar um container baseado neste imagem:

```
docker container run ubuntu
```

Foi criado um container e executado, porém como ele não tem nenhum comando ou aplicação rodando o mesmo desliga logo em seguida, podemos verificar com o comando:

```
docker container ls --all
```

Dica: A FLA `--all` serve para listar todos os containers existentes estando ligado ou não. O `ls` lista apenas os containers ligados:

Para saber mais sobre os parâmetros que podemos usar para o comando 'run', podemos executar:

```
docker container run --help
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
706a693ce6d9	ubuntu	"bash"	18 seconds ago	Exited (0) 16 seconds ago		crazy_nobel

Então vamos executar novamente, mas com um comando `sleep` de um dia para que o mesmo fique ligado:

```
docker container run --publish 80:80 --detach ubuntu sleep 1d
```

```
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
125a6e411906: Pull complete
Digest: sha256:26c68657ccce2cb0a31b330cb0be2b5e108d467f641c62e13ab40cbec258c68d
Status: Downloaded newer image for ubuntu:latest
560177fb245c14f3a9aa3de6a20684efc48ed1cbe4ff35546e3c85684971a0b4
```

Está é nossa saída, ou seja temos um container rodando perfeitamente.



Dica: A FLAG --publish ou -p serve para liberar o acesso na porta 80 do nosso container, porta essa que o Apache trabalha e a flag - -detach ou -d serve para rodarmos o container desanexado do nosso terminal, deixando o terminal livre para nosso uso.

Para saber mais sobre os parâmetros que podemos usar para o comando 'run', podemos executar:

```
docker container run --help
```

Mas iremos verificar o container criado:

```
docker container ls
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
560177fb245c	ubuntu	"sleep 1d"	7 minutes ago	Up 7 minutes		hungry gould

Precisamos acessar nosso container agora, de posse do ID do container vamos realizar o comando:

```
docker container exec -it 560177fb245c bash
```

Dica: O comando exec serve para executarmos alguma ação com o nosso container, no caso iremos executar um terminal interativo usando a FLAG -it e vamos passar qual terminal do Linux nós queremos no caso o bash.

Para saber mais sobre os parâmetros que podemos usar para o comando 'run', podemos executar:

```
docker container exec --help
```

A saída é um terminal Linux:

```
$ docker container exec -it 560177fb245c bash
root@560177fb245c:/#
```

Vamos executar uma sequência de comandos no bash do nosso container:

```
>> apt-get update
```

```
>> apt-get install apache2 -y
```

Após esses comando vamos verificar a instalação do Apache com o comando:

```
apachectl -v
```

```
root@94813e6eda25:/# apachectl -v
Server version: Apache/2.4.52 (Ubuntu)
Server built:   2022-03-25T00:35:40
```

Podemos também verificar o status do serviço do Apache com o comando:

```
service apache2 status
```

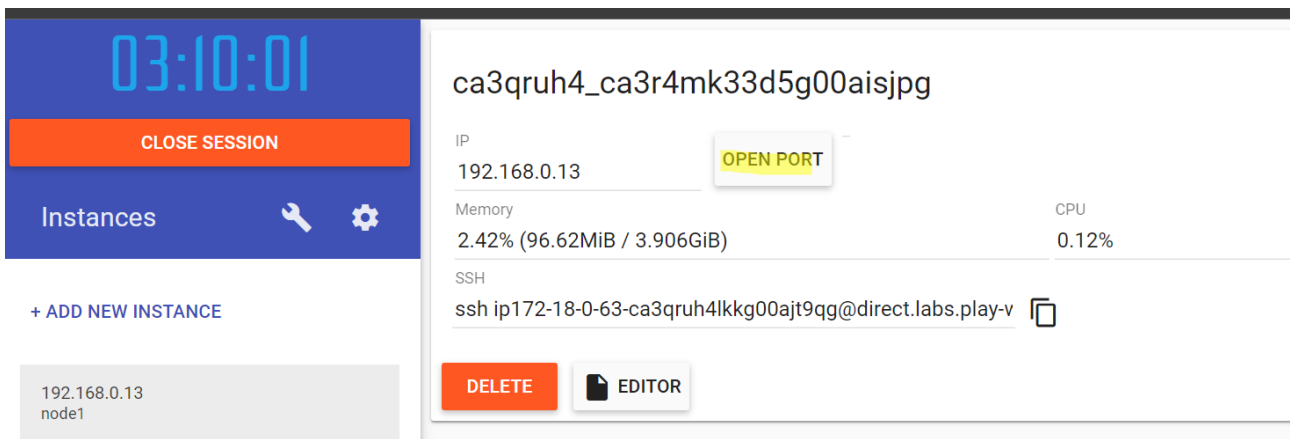
Caso tenha uma saída igual a esta:

```
* apache2 is not running
```

Você pode ativar o serviço do Apache com o comando:

```
service apache2 start
```

Agora podemos acessar a página default do Apache em nosso navegador, caso o acesso seja pelo nosso computador, digitaremos localhost:80 em nosso navegador caso esteja usando o Play with Docker, clique no OPEN PORT e digite o número da porta no caso 80





labs.play-with-docker.com diz

What port would you like to open?

OK

Cancelar

E teremos a página do Apache:



Apache2 Default Page

Ubuntu

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in `/usr/share/doc/apache2/README.Debian.gz`**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|   |-- ports.conf
|-- mods-enabled
|   |-- *.load
|   |-- *.conf
|-- conf-enabled
|   |-- *.conf
|-- sites-enabled
|   |-- *.conf
```

```
>> apt-get install git
```

E Instalaremos o git em nosso container para que possamos clonar nosso projeto:

```
>> git clone https://github.com/nidiodolfini/aula21
```

```
>> chmod 777 -R aula21/
```

```
>> cp -rf aula21/* /var/www/html/
```

Após esses comandos feitos no terminal já teremos a página de nosso projeto visível em nosso navegador pelo endereço localhost ou pelo OPEN PORT do Play with Docker:



The screenshot shows a web application interface with a light gray background. On the left, there is a vertical purple-to-blue gradient bar. The main content area is white and contains the following elements:

- The title "ToDo" in a large, dark gray font.
- The subtitle "Ingressar" in a slightly smaller, dark gray font.
- An "Email:" label followed by a light gray input field with a key icon on the right.
- A "Senha:" label followed by a light gray input field with a key icon on the right.
- A blue rectangular button with the text "Ingressar" in white.
- A link at the bottom that reads "Não tem conta? Cadastre se [aqui](#)".