

PROGRAMAÇÃO CONCORRENTE E DISTRIBUÍDA

Docker Hub e Boas Práticas

Aula 06 — Publicando imagens e construindo com múltiplos estágios

Prof. Newton S. B. Miyoshi

newton.miyoshi@baraodemaua.br

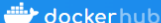
Centro Universitário Barão de Mauá — Ribeirão Preto

PARTE 1

Docker Hub

Publicando uma imagem com
`docker compose`.

Enviando com Docker Compose — criar o repositório

 Explore **Repositories** Organizations Usage

Search Docker Hub ctrl+K ⌕ ⚙ 🌙 ☰ N

[Repositories](#) / [Create](#) Using 0 of 1 private repositories.

Create repository

Namespace
newtonsbm

Repository Name *
cafecompao



Short description

Repositório de exemplo do projeto Cafe com Pão do curso de Ciência da Computação da Barão de Mauá

A short description to identify your repository. If the repository is public, this description is used to index your content on Docker Hub and in search engines, and is visible to users in search results.

Visibility

Using 0 of 1 private repositories. [Get more](#)

☒ Public 
Appears in Docker Hub search results

☐ Private 
Only visible to you

Cancel

Create

Pushing images

You can push a new image to this repository using the CLI:

```
docker tag local-image:tagname new-repo:tagname
docker push new-repo:tagname
```

Make sure to replace `tagname` with your desired image repository tag.

3

Esteja logado no Docker Desktop

The screenshot shows the Docker Desktop application window. The top bar is blue with the Docker logo, the text 'docker desktop PERSONAL', a search bar with the placeholder 'Search for images, containers, volume...', and a 'Ctrl+K' button. The left sidebar contains navigation icons for Containers, Images, Volumes, Builds, Docker Scout, and Extensions. The main area is titled 'Containers' and shows 'No containers are running.' Below this is a search bar and a toggle for 'Only show running containers'. A table lists containers with columns for Name, Image, Status, and Port(s). A user profile dropdown menu is open, showing the user's name 'newtonsbm' and options for 'Account Settings', 'Upgrade', and 'Sign out'. The bottom status bar shows 'Engine running', system resources (RAM 1.40 GB, CPU 0.83%, Disk -- GB avail. of -- GB), and version information (v4.33.1).

Containers [Give feedback](#)

Container CPU usage ⓘ
No containers are running.

Search ☐ Only show running containers

	Name	Image	Status	Port(s)	
☐	meucafezi-run-9e502e6a	cafecompao-meucafezinho	Exited		☐ ☐ ☐
☐	hardcore_	cafecompao	Exited	8000:8000	☐ ☐ ☐
☐	gifted_bat	cafecompao	Exited	8000:8000	☐ ☐ ☐
☐	romantic_	cafecompao	Exited		☐ ☐ ☐
☐	django-run-c1a1b7bb	cafecompao-django	Created (127)		☐ ☐ ☐

Showing 11 items

Engine running ☐ ☐ ☐ RAM 1.40 GB CPU 0.83% Disk -- GB avail. of -- GB BETA [Terminal](#) ☒ v4.33.1 ☐ 2

Enviando com Docker Compose — propriedade `image`

```
1 services:
2   meucafezinho:
3     build: .
4     image: newtonsbm/cafecompao_aula
5     ports:
6       - "8000:8000"
7     command: /bin/sh -c "python manage.py migrate && python manage.py runserver 0.0.0.0:8000"
```

No serviço que será enviado, `image` segue o padrão `nome_usuario/repositorio` referente ao Docker Hub.

Garanta que a imagem foi buildada

```
1 $ docker compose build
2 [+] Building 12.5s (8/10)
3  => [meucafezinho 1/5] FROM docker.io/library/python:3-alpine
4  => [meucafezinho 2/5] COPY . /app
5  => [meucafezinho 3/5] WORKDIR /app
6  => [meucafezinho 4/5] RUN pip install -r requirements.txt
7  => [meucafezinho 5/5] ...
```

Envie para o Docker Hub

```
1 $ docker compose push meucafezinho
2 [+] Pushing 9/9
3   - Pushing newtonsbm/cafecompao_aula: 2f44f485586a Pushed
4   - Pushing newtonsbm/cafecompao_aula: 699a8ba52c91 Pushed
5   - Pushing newtonsbm/cafecompao_aula: 5f70bf18a086 Mounted from newtonsbm/cafecompao_exemplo
```

Verifique no Docker Hub



Search Docker Hub

ExploreRepositoriesOrganizationsHelp

Upgrade

 newtonsbm

newtonsbm>Repositories>cafepao>latest



newtonsbm/cafepao:latest

Delete Tag

DIGEST: sha256:3f6f6143eb27871398bcd99fe10ea5e58a815705688e48a0c5d5f63a766d2c3f

OS/ARCH	COMPRESSED SIZE ⓘ	LAST PUSHED	TYPE
linux/amd64	76.34 MB	a minute ago by newtonsbm	Image

Image LayersVulnerabilities

IMAGE LAYERS ⓘ

1

ADD file ... in /

27.78 MB

2

CMD ["bash"]

0 B

3

ENV PATH=/usr/local/bin:/usr/local/sbin:/usr/local/...

0 B

4

ENV LANG=C.UTF-8

0 B

5

RUN /bin/sh -c set -eux;

3.34 MB

6

ENV GPG_KEY=A035C8C19219BA821ECEA86B64E628F8D684696D

0 B

7

ENV PYTHON_VERSION=3.11.1

0 B

Command

ADD file:997f5a9b32407d96efac41a1cfafb318f77de077c8b5cd7065b6ec9796b4bf5e in /

Opcional: tag

```
1 services:
2   meucafezinho:
3     build: .
4     image: newtonsbm/cafecompao_aula:pcd2024
5     ports:
6       - "8000:8000"
```

A tag é opcional — quando omitida, o padrão é `latest`.

Tags no Docker Hub

[General](#) [Tags](#) [Builds](#) [Collaborators](#) [Webhooks](#) [Settings](#)

☐ Sort by

Newest ▾

Filter Tags

Delete

☐

TAG

pcd2024

Last pushed 2 minutes ago by [newtonsbm](#)

docker pull newtonsbm/cafecompao_aula:pcd2024

Copy

Digest	OS/ARCH	Last pull	Compressed Size ⓘ
90f79a113497	linux/amd64	---	49.76 MB

☐

TAG

latest

Last pushed 20 minutes ago by [newtonsbm](#)

docker pull newtonsbm/cafecompao_aula:latest

Copy

PARTE 2

Boas Práticas — Multi-stage builds

Imagens menores, separando build de execução.

Use builds de múltiplos estágios

- Muitas vezes precisamos de **mais de uma imagem** para uma aplicação.
- Uma imagem para **fazer o build**, outra para **fazer o deploy**
 - ex.: Maven para Java, Tomcat para deploy.
- Vamos ver um exemplo: aplicação em **Go**, com build no **golang** e deploy no **alpine**.

Usando um estágio anterior para construir outro

```
1 FROM alpine:latest as builder
2 RUN apk --no-cache add build-base
3
4 FROM builder as build1
5 COPY source1.cpp source.cpp
6 RUN g++ -o /binary source.cpp
7
8 FROM builder as build2
9 COPY source2.cpp source.cpp
10 RUN g++ -o /binary source.cpp
```

Os estágios `build1` e `build2` reaproveitam o estágio `builder` — sem repetir a instalação das dependências.

EXEMPLO PRÁTICO

Passo a passo — Pelican SSG

- Criar `Dockerfile`.
- Criar `compose.yaml`.
- Gerar o projeto Pelican.
- Configurar.

Gerando o projeto — pelican-quickstart

```
1 $ docker compose run -it cafeblog pelican-quickstart
2 Welcome to pelican-quickstart v4.9.1.
3
4 > Where do you want to create your new web site? [.]
5 > What will be the title of this web site? Cafezando
6 > Who will be the author of this web site? Newton Miyoshi
7 > Do you want to specify a URL prefix? (Y/n) y
8   blog.cafecompao.com.br
9 Done. Your new project is available at /app
```

Docker Multi Stage — Pelican: Dockerfile (single / dev)

```
FROM python:3-slim
RUN mkdir /app
COPY . /app
WORKDIR /app
RUN apt-get update
RUN apt-get install -y git
RUN pip install pelican[markdown] git+https://github.com/arulrajnet/attila.git@master
```

Uma única imagem: Python + Git + todas as dependências de build, também usada para servir o site em desenvolvimento.

Docker Multi Stage — Pelican: DockerfileProd (multi / prod)

```
1 FROM python:3-slim AS builder
2 RUN mkdir /app
3 COPY . /app
4 WORKDIR /app
5 RUN apt-get update
6 RUN apt-get install -y git
7 RUN pip install pelican[markdown] git+https://github.com/arulrajnet/attila.git@master
8 RUN pelican content -o output -s publishconf.py
9
10 FROM nginx:1-alpine
11 COPY --from=builder /app/output/ /usr/share/nginx/html
```

O estágio **builder** gera o site estático; só o resultado (**output/**) vai para a imagem final do nginx — Python e Git ficam de fora.

compose.yaml — dev (single-stage)

```
1 services:
2   cafeblog:
3     build: .
4     image: cafecompao_blog:dev
5     ports:
6       - "3000:3000"
7     volumes:
8       - ./app
9     command: pelican --autoreload --listen --bind 0.0.0.0 --port 3000
```

compose.prod.yaml — prod (multi-stage)

```
1 services:
2   cafeblog:
3     image: cafecompao_blog:prod
4     build:
5       context: .
6       dockerfile: DockerfileProd
7     ports:
8       - "8080:80"
```

`docker compose -f compose.prod.yaml up` usa outro arquivo e outro `dockerfile` — o nginx passa a servir na porta 80.

Comparação

 **docker desktop** PERSONAL

 Search for images, containers, volume... Ctrl+K

    N — □

 Containers

 **Images**

 Volumes

 Builds

 Docker Scout

 Extensions

Images [Give feedback](#)

Local Hub

1.76 GB / 4.91 GB in use

25 images

Last refresh: 59 minutes ago 

 Search

☐	Name	Tag	Status	Created	Size	Actions		
☐	cafecompao_blog fcc9188e5dee 	prod	Unused	18 seconds ago	46.42 MB			
☐	cafecompao_blog 1771b2dd1868 	dev	Unused	14 minutes ago	290.91 MB			

MATERIAL EM VÍDEO



YOUTUBE Docker Multistage Build — O segredo pra enxugar suas imagens [youtube.com/watch?v=zP4lInlwXg](https://www.youtube.com/watch?v=zP4lInlwXg)

Referências bibliográficas

- DOCKER. Docker Hub quickstart. docs.docker.com/docker-hub/quickstart
- DOCKER. Multi-stage builds. docs.docker.com/build/building/multi-stage