Criação de Imagens Docker

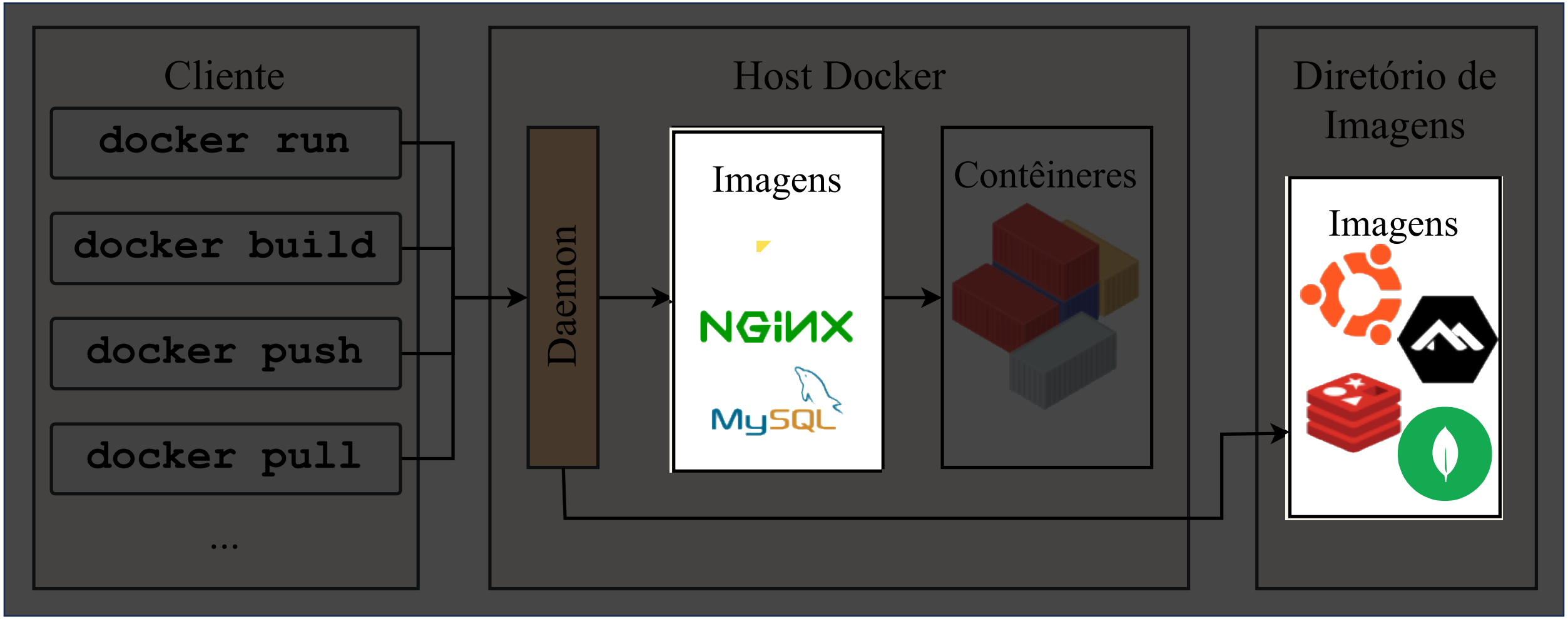
Professor: Pedro Horchulhack

Agenda

1. O que são imagens
2. Tipos de criação de imagens
3. Estratégias e boas práticas
4. Atividade

O que é uma imagem Docker?

Relembrando o fluxo de um contêiner



O que é o Docker? – Importante lembrar

- **Imagem**: Conjunto de instruções que empacotam, de modo padronizado, como um contêiner deve executar (inclui bibliotecas, binários, arquivos auxiliares e configurações)

- **Container**: Processo isolado, em execução, contendo os itens empacotados por uma imagem específica

- **Diretório de imagens**: Repositório centralizado para armazenar e compartilhar imagens Docker

O que é uma imagem Docker?

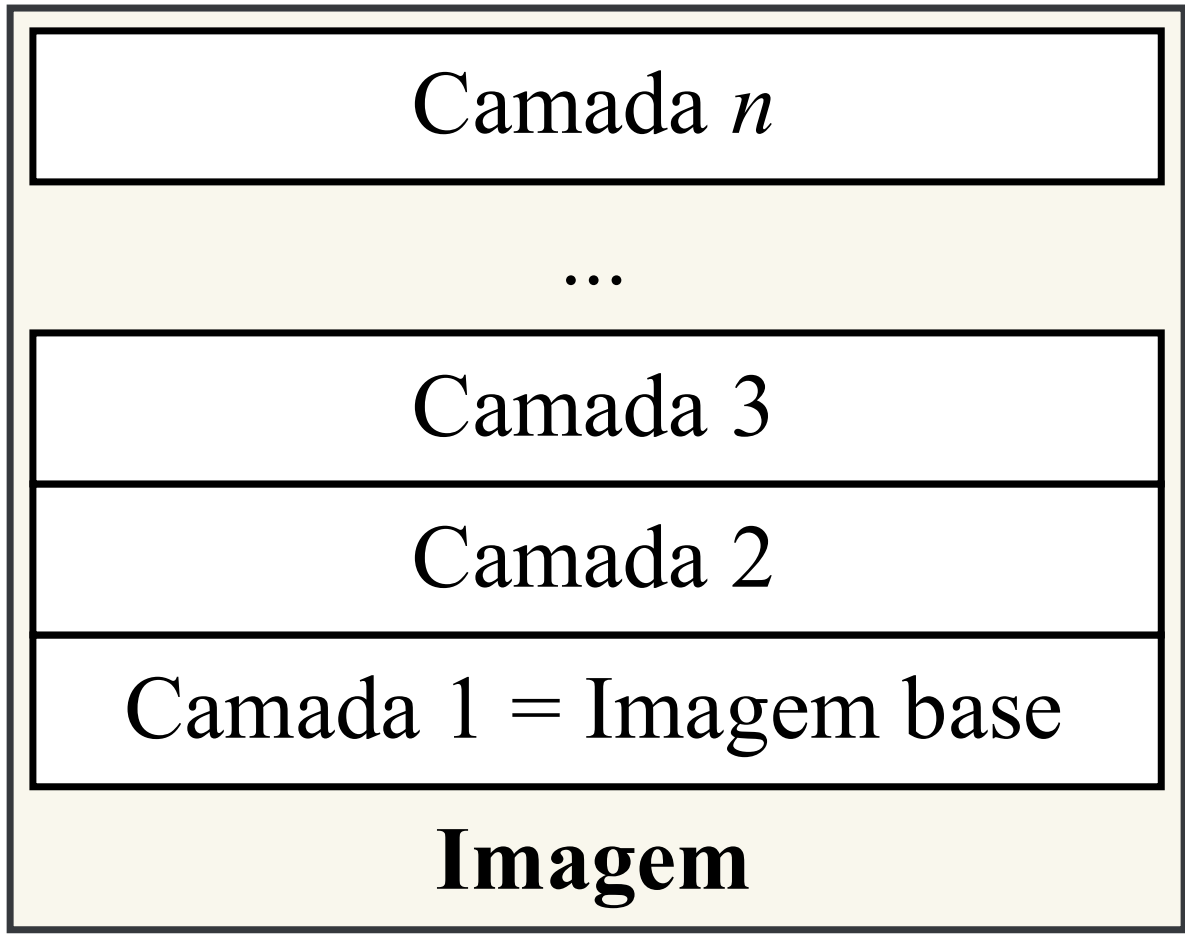
- Ela é composta por múltiplas camadas empilhadas
  - Cada modificação representa uma **camada** (adicionar arquivo, atualizar sistema, especificar usuário, ...)

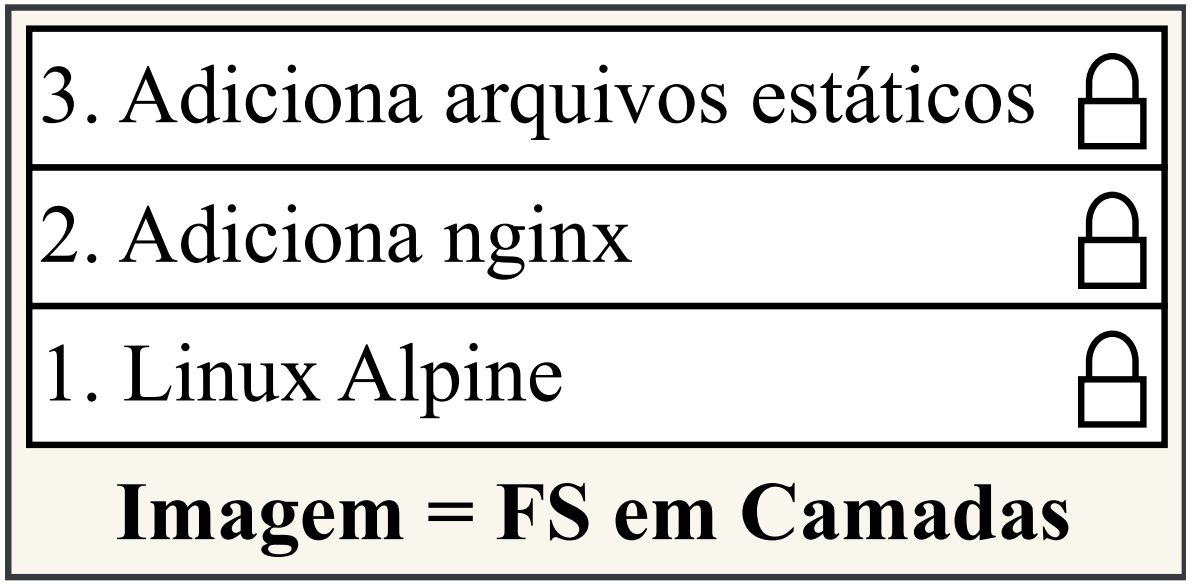
- Nós criamos uma imagem através de um arquivo `Dockerfile` (sem extensões) ou interativamente
  - Cada **instrução** ou **comando** representará uma camada na imagem

- A partir desse `Dockerfile` nós iremos construí-la e atribuí-la um **nome** e uma **tag** específica

- Para quem quiser se aprofundar: Utiliza o conceito de Union File System (UFS)

O que é uma imagem Docker?

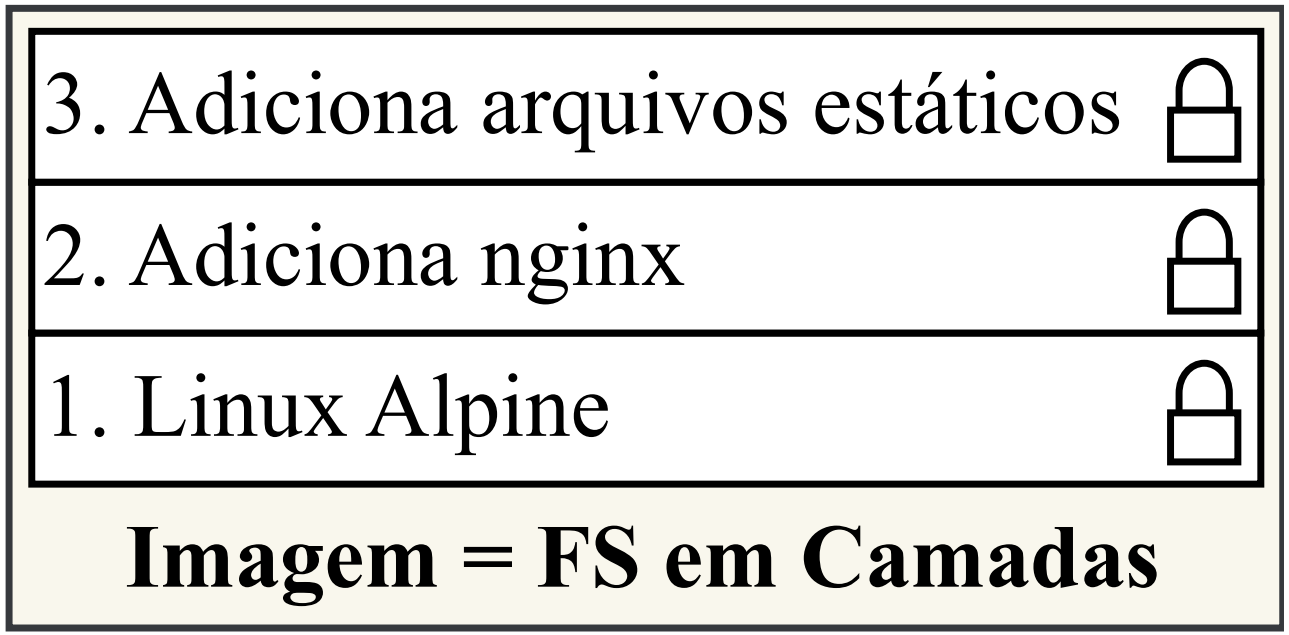




O que é uma imagem Docker?

- As camadas das imagens são imutáveis

- Permite que múltiplos contêineres, até imagens, de compartilhar camadas
  - Reutilização de espaço (disco/memória)
  - Reduz tempo de inicialização de contêineres



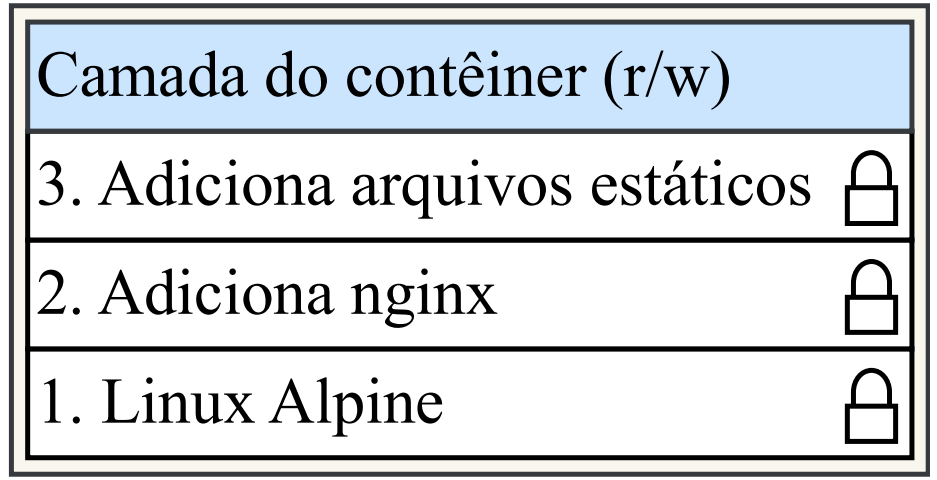
O que é uma imagem Docker? – Exemplo

O que é uma imagem Docker? – Exemplo

```
~ » docker pull nginx                                                    hrchlhck@hrchlhck
Using default tag: latest
latest: Pulling from library/nginx
24c63b8dcb66: Pull complete
ac894f1d1dfb: Pull complete
2572d4eb2260: Pull complete
0ac3805c647c: Pull complete
da20f09652a8: Pull complete
2de21a3abd85: Pull complete
77cea143f3c3: Pull complete
Digest: sha256:a484819eb60211f5299034ac80f6a681b06f89e65866ce91f356ed7c72af059c
Status: Downloaded newer image for nginx:latest
docker.io/library/nginx:latest
```

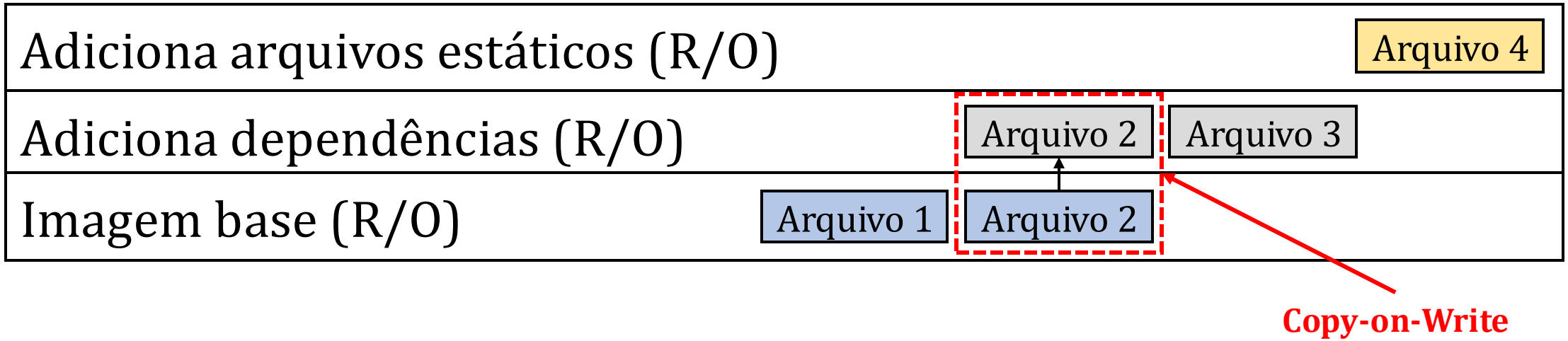
O que é uma imagem Docker?

- Aos contêineres são adicionados uma camada “fina” para leitura e escrita
  - Dados que estão nas camadas inferiores não são alterados
- Estratégia *Copy-on-Write* (CoW)



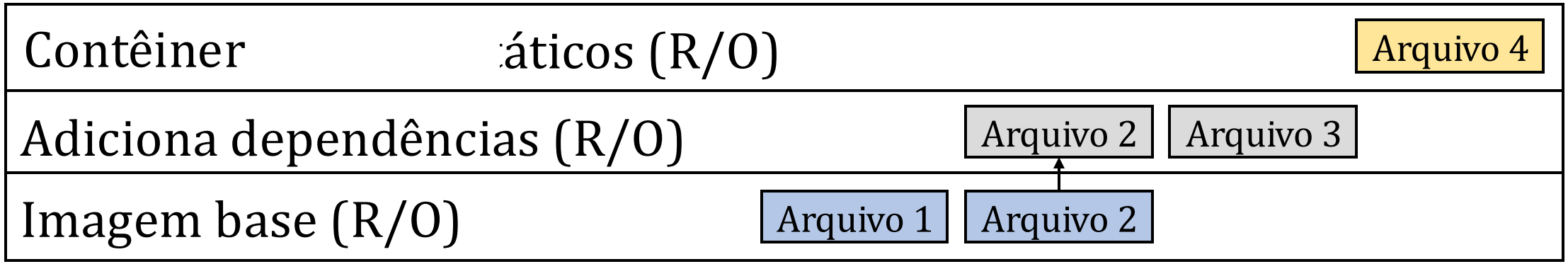
O que é uma imagem Docker?

Usando *Copy-on-Write*



O que é uma imagem Docker?

Usando *Copy-on-Write*



Criando uma imagem

- Podemos criar imagens de duas formas:
  - Por interação
  - Por declaração

Criando uma imagem por **interação**

Criando uma imagem – Interação

- Vamos criar um contêiner da imagem **ubuntu**
  - Baixar o `curl` (`apt update && apt install curl -y`)
  - Testar o comando baixado (`curl -I google.com`)
  - Sair do contêiner (`exit`)

- Liste e filtre os contêineres pelo id ou o nome especificado
- Execute o comando `docker container diff <nome ou id>`
  - Obs: também podemos executar assim `docker diff <nome ou id>`

Criando uma imagem – Interação

```
~ » docker container diff 495f6be64f67                                    hrchlhck@hrchlhck
C /usr
C /usr/lib
A /usr/lib/sasl2
A /usr/lib/ssl
A /usr/lib/ssl/cert.pem
A /usr/lib/ssl/certs
A /usr/lib/ssl/misc
A /usr/lib/ssl/misc/tsget.pl
A /usr/lib/ssl/misc/CA.pl
A /usr/lib/ssl/misc/tsget
A /usr/lib/ssl/openssl.cnf
A /usr/lib/ssl/private
C /usr/lib/aarch64-linux-gnu
A /usr/lib/aarch64-linux-gnu/libgssapi_krb5.so.2.2
A /usr/lib/aarch64-linux-gnu/libpsl.so.5
A /usr/lib/aarch64-linux-gnu/libkrb5support.so.0.1
A /usr/lib/aarch64-linux-gnu/libldap.so.2
A /usr/lib/aarch64-linux-gnu/libkrb5.so.3
```

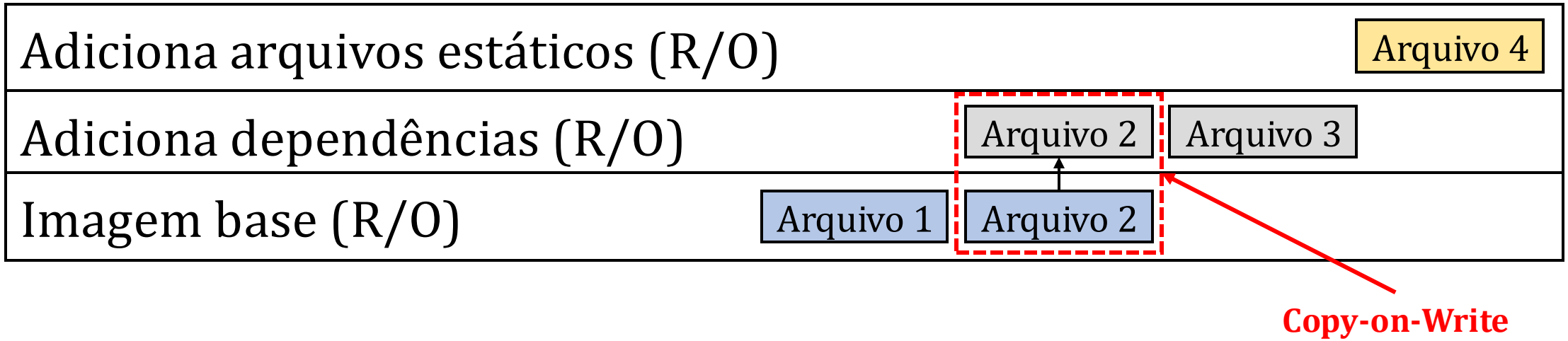
Criando uma imagem – Interação

- A lista denota quais arquivos foram
  - (**A**)dicionados,
  - Alterados ((**C**)hanged)
  - Removidos ((**D**)eleted)

```
~ » docker container diff 495f6be64f67                                    hrchlhck@hrchlhck
C /usr
C /usr/lib
A /usr/lib/sasl2
A /usr/lib/ssl
A /usr/lib/ssl/cert.pem
A /usr/lib/ssl/certs
A /usr/lib/ssl/misc
A /usr/lib/ssl/misc/tsget.pl
A /usr/lib/ssl/misc/CA.pl
A /usr/lib/ssl/misc/tsget
A /usr/lib/ssl/openssl.cnf
A /usr/lib/ssl/private
C /usr/lib/aarch64-linux-gnu
A /usr/lib/aarch64-linux-gnu/libgssapi_krb5.so.2.2
A /usr/lib/aarch64-linux-gnu/libpsl.so.5
A /usr/lib/aarch64-linux-gnu/libkrb5support.so.0.1
A /usr/lib/aarch64-linux-gnu/libldap.so.2
A /usr/lib/aarch64-linux-gnu/libkrb5.so.3
```

Criando uma imagem – Interação

Usando *Copy-on-Write*, nós alteramos a imagem original adicionando um novo pacote (`curl`) e conseguimos salvá-la com essas alterações.



Criando uma imagem – Interação

Usando *Copy-on-Write*, nós alteramos a imagem original adicionando um novo pacote (`curl`) e conseguimos salvá-la com essas alterações.

Para salvar a nova imagem:

- `docker commit <id ou nome do contêiner> <novo nome da imagem>`

Criando uma imagem – Interação

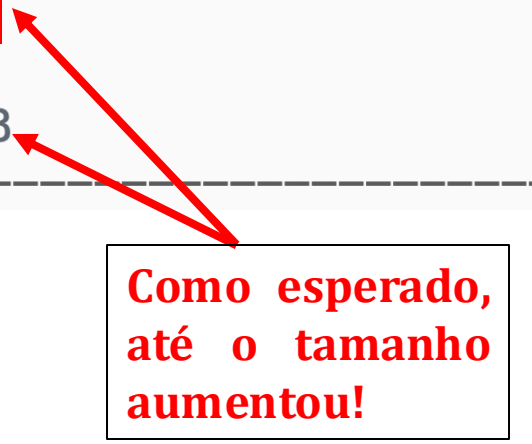
Antes do commit

```
~ » docker images                                                    hrchlhck@hrchlhck
REPOSITORY   TAG       IMAGE ID       CREATED        SIZE
nginx        latest    8dd77ef2d82e   12 days ago    193MB
ubuntu       latest    fabf3a8d4949   2 weeks ago    98.8MB
```

Criando uma imagem – Interação

Depois do commit

```
[~ » docker commit 495f6be64f67 nova-imagem                    hrchlhck@hrchlhck ]
sha256:39195e321917459a8fa3b4c42bd297d646951806960bf91ed8153d23c0097228
------------------------------------------------------------------------------
[~ » docker images                                             hrchlhck@hrchlhck ]
REPOSITORY    TAG       IMAGE ID       CREATED          SIZE
nova-imagem   latest    39195e321917   3 seconds ago    145MB
nginx         latest    8dd77ef2d82e   12 days ago      193MB
ubuntu        latest    fabf3a8d4949   2 weeks ago      98.8MB
------------------------------------------------------------------------------
```



Criando uma imagem – Interação

Conseguimos visualizar as alterações feitas

`docker history <nome ou id da imagem>`

```
~ » docker history nova-imagem
IMAGE          CREATED         CREATED BY                                      SIZE      COMMENT
e1dc010df15f   6 seconds ago   bash                                            46.7MB
fabf3a8d4949   2 weeks ago     /bin/sh -c #(nop)  CMD ["/bin/bash"]            0B
<missing>      2 weeks ago     /bin/sh -c #(nop) ADD file:d1bd5209fbd828a2a…   98.8MB
<missing>      2 weeks ago     /bin/sh -c #(nop)  LABEL org.opencontainers.…   0B
<missing>      2 weeks ago     /bin/sh -c #(nop)  LABEL org.opencontainers.…   0B
<missing>      2 weeks ago     /bin/sh -c #(nop)  ARG LAUNCHPAD_BUILD_ARCH     0B
<missing>      2 weeks ago     /bin/sh -c #(nop)  ARG RELEASE                  0B
```

Criando uma imagem – Interação

Crie uma imagem de forma **interativa**, de algum dos exercícios anteriores

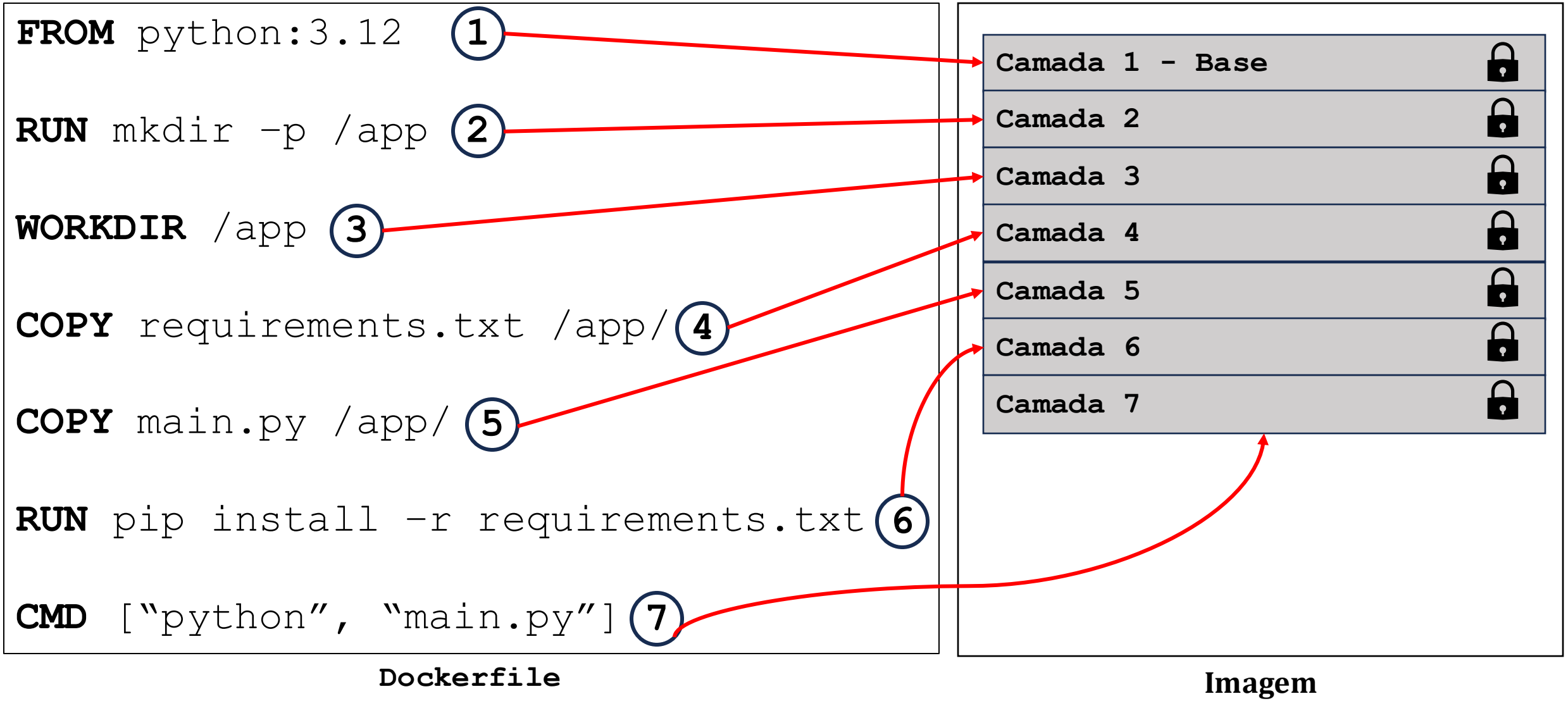
Criando uma imagem por **declaração**

Criando uma imagem – Declaração

- Nesta estratégia nós **declaramos** ao docker o que queremos que ele faça

- Criamos imagens **camada por camada**, através de instruções e utilizando um arquivo chamado **`Dockerfile`**

Criando uma imagem – Declaração



Criando uma imagem – Declaração

- Baixe os códigos disponibilizados pelo professor
  - `requirements.txt`
  - `main.py`

- Coloque-os no mesmo diretório que o seu `Dockerfile`

- Precisamos construir a nossa imagem a partir do `Dockerfile`, associando-a à uma **tag**
  - `docker build -t` **`imagem-declarada .`**

- Vamos olhar o histórico

Criando uma imagem – Declaração

```
~ » docker history imagem-declarada                                                                 hrchl
IMAGE          CREATED          CREATED BY                                      SIZE      COMMENT
1e92367b20ac   12 seconds ago   CMD ["python" "main.py"]                        0B        buildkit.dockerfile.v0
<missing>      12 seconds ago   RUN /bin/sh -c pip install -r requirements.t…   15.2MB    buildkit.dockerfile.v0
<missing>      15 seconds ago   COPY main.py /app/ # buildkit                   195B      buildkit.dockerfile.v0
<missing>      15 seconds ago   COPY requirements.txt /app/ # buildkit          6B        buildkit.dockerfile.v0
<missing>      15 seconds ago   WORKDIR /app                                    0B        buildkit.dockerfile.v0
<missing>      15 seconds ago   RUN /bin/sh -c mkdir -p /app # buildkit         0B        buildkit.dockerfile.v0
<missing>      5 weeks ago      CMD ["python3"]                                 0B        buildkit.dockerfile.v0
<missing>      5 weeks ago      RUN /bin/sh -c set -eux;   wget -O get-pip.p…   10.5MB    buildkit.dockerfile.v0
<missing>      5 weeks ago      ENV PYTHON_GET_PIP_SHA256=dfe9fd5c28dc98b5ac…   0B        buildkit.dockerfile.v0
<missing>      5 weeks ago      ENV PYTHON_GET_PIP_URL=https://github.com/py…   0B        buildkit.dockerfile.v0
<missing>      5 weeks ago      ENV PYTHON_PIP_VERSION=24.0                     0B        buildkit.dockerfile.v0
<missing>      5 weeks ago      RUN /bin/sh -c set -eux;  for src in idle3 p…   32B       buildkit.dockerfile.v0
<missing>      5 weeks ago      RUN /bin/sh -c set -eux;   wget -O python.ta…   62MB      buildkit.dockerfile.v0
<missing>      5 weeks ago      ENV PYTHON_VERSION=3.12.3                       0B        buildkit.dockerfile.v0
<missing>      5 weeks ago      ENV GPG_KEY=7169605F62C751356D054A26A821E680…   0B        buildkit.dockerfile.v0
<missing>      5 weeks ago      RUN /bin/sh -c set -eux;  apt-get update;  a…   19.1MB    buildkit.dockerfile.v0
<missing>      5 weeks ago      ENV LANG=C.UTF-8                                0B        buildkit.dockerfile.v0
<missing>      5 weeks ago      ENV PATH=/usr/local/bin:/usr/local/sbin:/usr…   0B        buildkit.dockerfile.v0
<missing>      2 days ago       /bin/sh -c set -ex;  apt-get update;  apt-ge…   560MB
<missing>      2 days ago       /bin/sh -c set -eux;  apt-get update;  apt-g…   183MB
<missing>      2 days ago       /bin/sh -c set -eux;  apt-get update;  apt-g…   48.5MB
<missing>      2 days ago       /bin/sh -c #(nop)  CMD ["bash"]                 0B
<missing>      2 days ago       /bin/sh -c #(nop) ADD file:b7c55dda8ded4219a…   139MB
```

Criando uma imagem – Declaração

- Vamos dar uma olhada nas diretivas que colocamos no `Dockerfile`

Criando uma imagem – Declaração

```
FROM python:3.12

RUN mkdir -p /app

WORKDIR /app

COPY requirements.txt /app/

COPY main.py /app/

RUN pip install -r requirements.txt

CMD ["python", "main.py"]
```

- **FROM**: indica qual a imagem base
- **RUN**: executa comandos dentro do contêiner
- **WORKDIR**: Muda o diretório padrão
- **COPY**: Copia arquivos de fora do contêiner para dentro dele
- **CMD**: Especifica os parâmetros a serem executados pelo contêiner

Criando uma imagem – Declaração

- Alguns comandos adicionais:
  - **`ENTRYPOINT:`** Especifica o comando base que irá executar os parâmetros fornecidos pela instrução CMD
  - **`ADD:`** Baixa o arquivo e adiciona ao contêiner
  - **`ARG:`** Variáveis usadas durante o build
  - **`ENV:`** Variáveis de ambiente que irão ser passadas ao contêiner
  - **`USER:`** Especifica o usuário sobre qual o contêiner irá executar
  - **`EXPOSE:`** Especifica quais portas o contêiner irá executar (não substitui a flag –– publish)
  - **`SHELL:`** Especifica o shell padrão da imagem

Estratégias e boas práticas na criação de imagens

Estratégias e boas práticas na criação de imagens

- Temos um tipo de imagem **especial** chamado de `scratch`
  - Ela permite criarmos uma imagem que só contém um arquivo binário, estaticamente *linkado*
  - Exemplo: imagem `hello-world`

Estratégias e boas práticas na criação de imagens

- Um dos principais problemas ao criar imagens é o seu **tamanho**

- Estratégias:
  - Realizar *builds* multi-estágio para reduzir o tamanho
  - Reduzir a quantidade de camadas
  - Limpar as referências

```
nginx          latest    8dd77ef2d82e   13 days ago    193MB
ubuntu         latest    fabf3a8d4949   2 weeks ago    98.8MB
python         3.12      ca8cc0e526c3   5 weeks ago    1.02GB
alpine         latest    ace17d5d883e   3 months ago   7.73MB
```

Estratégias e boas práticas na criação de imagens

Builds multi-estágio

- Aqui conseguimos pegar arquivos específicos de outras imagens, **sem que toda a imagem seja concatenada** a imagem que você está criando

- Façamos um experimento

Estratégias e boas práticas na criação de imagens

Builds multi-estágio

1. Crie um novo diretório chamado `multiestagio` e entre nesse diretório
2. Crie um novo arquivo chamado `ola.c` que mostre na tela "`Olá , mundo!`" (Linguagem C)
3. Baixe o arquivo `Dockerfile.multiestagio` disponibilizado pelo professor
4. Realize o build da imagem com o nome `multiestagio`
   - `docker build -t multiestagio -f Dockerfile.multiestagio .`
5. Veja o tamanho resultante da imagem

Estratégias e boas práticas na criação de imagens

Builds multi-estágio

6. Vamos alterar como declaramos a imagem
7. Vamos fazer o build da imagem com um novo nome
8. Vejamos o tamanho da imagem final

Atividade

Vamos criar uma imagem personalizada para nosso projeto

Referências

1. Unionfs: A Stackable Unification File System. **Unionfs: A Stackable Unification File System**. 2024. Disponível em: <https://unionfs.filesystems.org/>. Acesso em 30 de abril de 2024.
2. Docker. **docker image**. Docker Inc, 2024. Disponível em: https://docs.docker.com/reference/cli/docker/image/>. Acesso em: 12 de maio de 2024.
3. Docker. **Docker overview**. Docker Inc., 2024. Disponível em: <https://docs.docker.com/get-started/overview/#docker-architecture>. Acesso em 12 de maio de 2024.
4. Docker. **Dockerfile reference**. Docker Inc., 2024. Disponível em: <https://docs.docker.com/reference/dockerfile/>. Acesso em 12 de maio de 2024.

Referências

5. RICE, Liz. **Container security: fundamental technology concepts that protect containerized applications**. First edition. Beijing Boston Farnham Sebastopol Tokyo: O'Reilly, 2020.
6. SCHENKER, Gabriel Nicolas. **The ultimate Docker container book: build, test, ship, and run containers with Docker and Kubernetes**. Third edition. Birmingham, UK: Packt Publishing Ltd., 2023.

Contato: p.horchulhack@pucpr.br