Containers com *Docker*

Professor: José Eurípedes Ferreira de Jesus Filho
jeferreirajf@gmail.com

Universidade Federal de Jataí – UFJ

Aula anterior

- Mundo interno:
  - As imagens já possuem definido as portas que elas irão expor ao mundo interno do Docker

```
docker run –itd --rm --name nginx nginx:1.25.1
```

- Mundo externo:

```
docker run –itd --rm --name nginx -p 8000:80 nginx:1.25.1
```

Introdução

- Até agora no curso nós estamos rodando imagens que vem do ***dockerhub***.

- Sabemos rodar um **`container`**, fazer as modificações que precisamos nele e então criar uma imagem a partir desse **`container`** modificado rodando.

- Existe alguma maneira de personalizar uma imagem sem precisar rodar um **`container`** antes?

Dockerfile!

Dockerfile

- Um `dockerfile` é a descrição passo a passo de como uma imagem deve ser.
  - ➢Geralmente uma imagem sempre é derivada de alguma imagem base.
    - ❖A imagem mais básica de todas se chama `scratch`
  - ➢A partir da imagem base, executa-se comandos e personalizações.
  - ➢É possível definir uma série de características da imagem.

Dockerfile

```
FROM ubuntu:23.10
```

Dockerfile

```
FROM ubuntu:23.10
```

- Para transformar a definição do dockerfile em uma imagem:
  - `docker build –t <nome>:<tag> <source>`

Workdir

- O **`workdir`** define um diretório padrão para o container.

```
FROM ubuntu:23.10
WORKDIR /app
```

- `docker build -t meuubuntu:latest .`

Run

- O `run` roda um comando na construção da imagem.

```
FROM ubuntu:23.10
WORKDIR /app
RUN apt-get update
RUN apt-install vim -y
```

- `docker build –t meuubuntu:latest .`

Cmd

- O **cmd** executa um comando depois da montagem da imagem. O comando pode ser substituído!

```
FROM ubuntu:23.10
WORKDIR /app
RUN apt-get update && apt-install vim -y
CMD [“echo”, “ola”, “mundo”]
```

- `docker build -t meuubuntu:latest .`

Entrypoint

- O **`entrypoint`** executa um comando depois da montagem da imagem. O comando não pode ser substituído, mas pode ser complementado!

```
FROM ubuntu:23.10
WORKDIR /app
RUN apt-get update && apt-install vim -y
ENTRYPOINT [“echo”, “ola”, “mundo”]
```

- `docker build –t meuubuntu:latest .`

Cmd e Entrypoint

- Você pode combinar **`entrypoint`** e **`cmd`** de diversas formas!

```
FROM ubuntu:23.10
WORKDIR /app
RUN apt-get update && apt-install vim -y
ENTRYPOINT [“echo”, “ola”]
CMD [“mundo”]
```

- `docker build -t meuubuntu:latest .`

Copy

- O **copy** copia algo do computador de quem está construindo a imagem para a imagem.

```
FROM ubuntu:23.10
WORKDIR /app
COPY <source> <destiny>
ENTRYPOINT [“cat”, <filename>]
```

- `docker build -t meuubuntu:latest .`

Exercício

- O que são `dockerfiles`? O que eles nos permitem fazer?

Exercício

- Crie uma **imagem Docker** chamada **node-app** utilizando um **Dockerfile** a partir de um **ubuntu:23.10**. A imagem deve possuir o **NodeJS** instalado e ao executar um container a partir dela, o container deve executar uma aplicação “**Hello world**” em **Node** e deve encerrar a execução.