Agilidade em Hackathons com o Docker

—

Lucas de Castro Oliveira

Alguém já ouviu falar ?

Que baleia azul é essa ?

- Docker é ferramenta que permite isolar processos e recursos dentro de um sistema operacional, sem que o mesmo interfira com o restante do seu ambiente.

- Recursos como disco, rede, memória, processamento e até mesmo S.O podem ser definidos e isolados para que se rode um processo em específico.

- Idéia velha: vem do conceito de kernel namespaces lá dos anos 70.

Linux: Instalando em 1 linha

```
curl -fsSL https://get.docker.com -o
get-docker.sh  &&
sh get-docker.sh
```

Windows/Mac

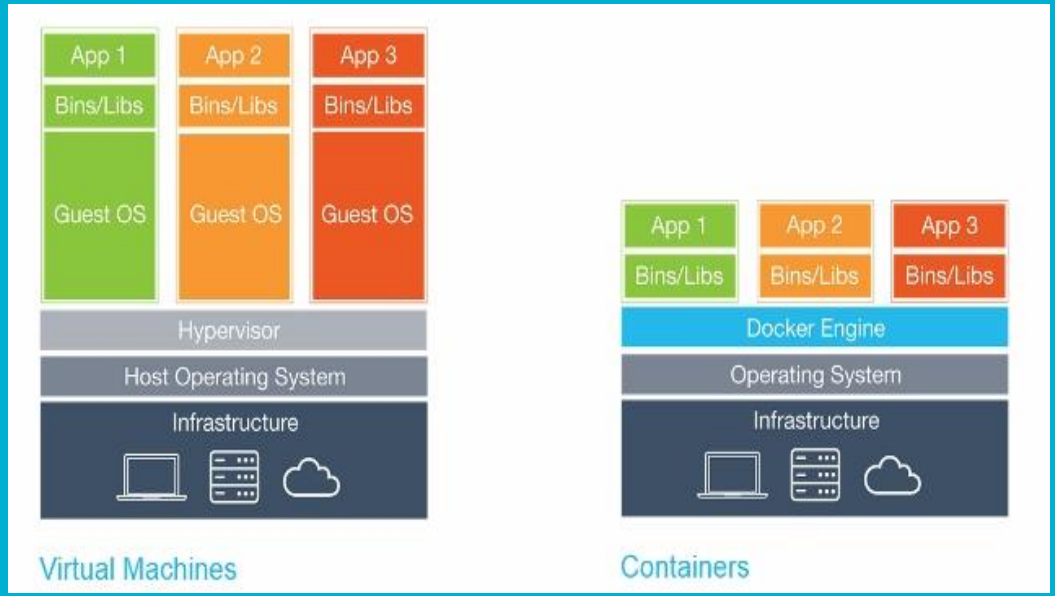
Instaladores em https://docs.docker.com/install/

Alguém já ouvir falar de máquina virtual ?

Docker ≠ VMs

Docker e VM(s) NÃO são a mesma coisa

- VM(s) criam uma instância inteira, ao passo que um docker host cria “containers”, que são processos isolados dentro de um S.O.
- Nada impede de um docker host ser instalado dentro de uma VM.



Fonte: Stack Exchange

Nossa você isolou meus parabéns

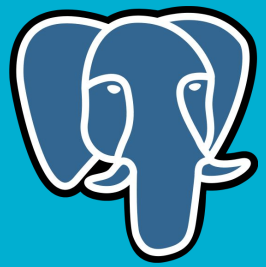
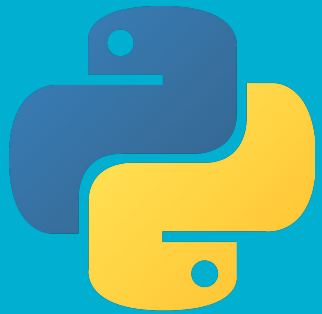
Tá mas o que eu faço com isso ???

- E aí que entra um lance muito interessante do docker

Fonte: Vida de Memes

Imagens Docker

- Imagens em docker são a base de todo o container, elas que definem o que irá rodar dentro do container.
- Pensem em imagens estão para containers assim como classes estão para objetos (paradigma OO).
- Com estas imagens é possível rodar os mais diversos serviços em um container:

Da imagem para o Container

- Existem diversas imagens docker em hub.docker.com (oficiais ou não).
- Para iniciar um container a partir de uma imagem, basta abrir o seu terminal e digitar alguns comandos para o docker.
- Básico do básico: hello-world em Docker
- `docker run hello-world`

Parâmetros adicionais

- São como flags que determinam propriedades do container que vai rodar. Existem várias, vou mostrar algumas:
    - **-e** : Variáveis de ambiente, úteis para quando a imagem proporciona configuração via variáveis de ambiente ex: MySql.
    - **-p**: Portas, recurso super interessante que permite conectar um container ao mundo exterior.
    - **-v**: Permite fazer mapeamento de volumes nos containers, ou seja, é possível persistir dados entre um container e outro. Além de inserir arquivos do host para o container.

Subindo um Mysql com Docker

- docker run -e MYSQL_ROOT_PASSWORD=secreta -e MYSQL_USER=system_abc -e MYSQL_PASSWORD=yourpass -e MYSQL_DATABASE=my_db -p 3306:3306 -d mysql:8.0.17

- Em que:
  - -e ... : Atribui variáveis de ambiente que serão incluídas no container
  - -p: Faz mapeamento de rede, um túnel entre a porta 3306 do host e a porta do 3306 do container.
  - -d: Este container é um *Daemon* ou seja, rode em background
  - Imagem: Qual imagem está container vai se basear e em que versão

Mais um exemplo

- docker run -v ./mosquitto/data:/mosquitto/data -v ./mosquitto/log:/mosquitto/log -v ./mosquitto/mosquitto.conf:/mosquitto/config/mosquitto.conf:ro -p 1883:1883 -p 9001:9001 -d eclipse-mosquitto:1.6.7

- -v: Volumes, lado esquerdo: fonte na minha máquina. Lado direito: destino no container

Quantos parâmetros, que sintaxe zuada!

Ainda bem que existe o docker-compose

- Permite definir parâmetros de containers num yaml todo organizadinho
- Ideal para colocar em versionamentos de código
- Uma forma fácil de coordenar múltiplos containers dentro da mesma máquina
- Possibilita fazer uma stack de serviços
- Fácil instalação
- Documentação extensa

Transformação comando para docker-compose

- docker run -e MYSQL_ROOT_PASSWORD=secreta -e MYSQL_USER=system_abc -e MYSQL_PASSWORD=yourpass -e MYSQL_DATABASE=my_db -p 3306:3306 -d mysql:8.0.17

```
version: '2.4'
services:
 mysql_db:
  image: mysql:8.0.17
  environment:
   - MYSQL_ROOT_PASSWORD=secreta
   - MYSQL_USER=system_abc
   - MYSQL_PASSWORD=yourpass
   - MYSQL_DATABASE=my_db
  ports:
   - 3306:3306
```

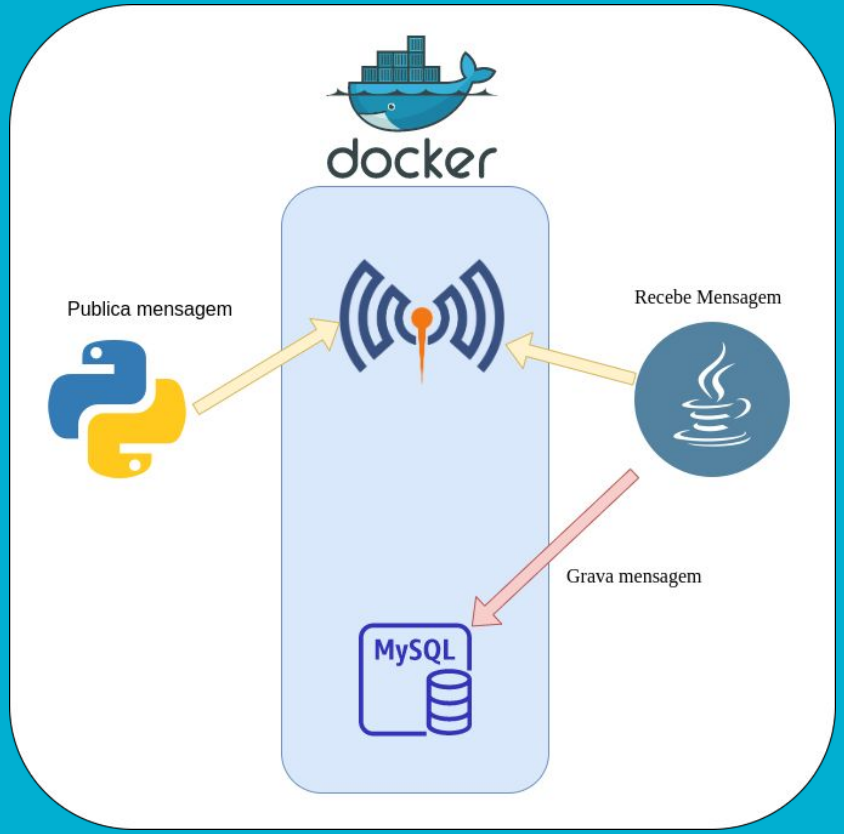
Transformação comando para docker-compose

- docker run -v ./mosquitto/data:/mosquitto/data -v ./mosquitto/log:/mosquitto/log -v ./mosquitto/mosquitto.conf:/mosquitto/config/mosquitto.conf:ro -p 1883:1883 -p 9001:9001 -d eclipse-mosquitto:1.6.7

```
version: '2.4'
services:
  mqtt:
    image: eclipse-mosquitto:1.6.7
    volumes:
      - ./mosquitto/data:/mosquitto/data
      - ./mosquitto/log:/mosquitto/log
      - ./mosquitto/mosquitto.conf:/mosquitto/config/mosquitto.conf:ro
    ports:
      - 1883:1883
      - 9001:9001
```

Demo

Arquitetura da Solução



Obrigado!!!

Dúvidas ???

Recursos

- https://labs.play-with-docker.com - Primeiros passos com Docker ? Comece por aqui.
- https://hub.docker.com/_/eclipse-mosquitto - Documentação da imagem do mosquitto.
- https://hub.docker.com/_/mysql - Documentação da imagem do MySQL.
- https://www.baeldung.com/java-mqtt-client - Guia para utilizar MQTT com Java/Spring.
- http://www.steves-internet-guide.com/publishing-messages-mqtt-client/ - Guia para utilziar MQTT com Python.
- https://github.com/eclipse/mosquitto/issues/1078 - Caso você tenha algum problema com permissões com a imagem do mosquitto.
- https://github.com/lcastrooliveira/demo-docker - Repositório onde contém arquivos de demonstração + apresentação.