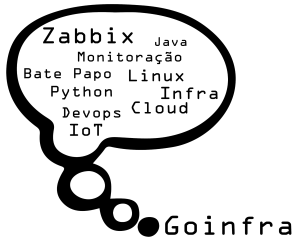
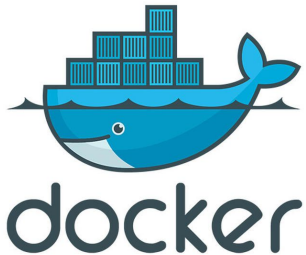
Construindo Container com o Docker

Bezaliel Ramos





O que vamos falar

O que é o docker
chroot…..Docker
Plataforma suportada
Conceitos
Namespace
cgroup
Rede
Volume
Imagem

Containers
Comandos(docker ps,
log,inspect,containers,images
Dockerfile
FROM e RUN
ADD e COPY
CMD e ENTRYPOINT
docker compose

ONX
Solutions

$whoami

Beza

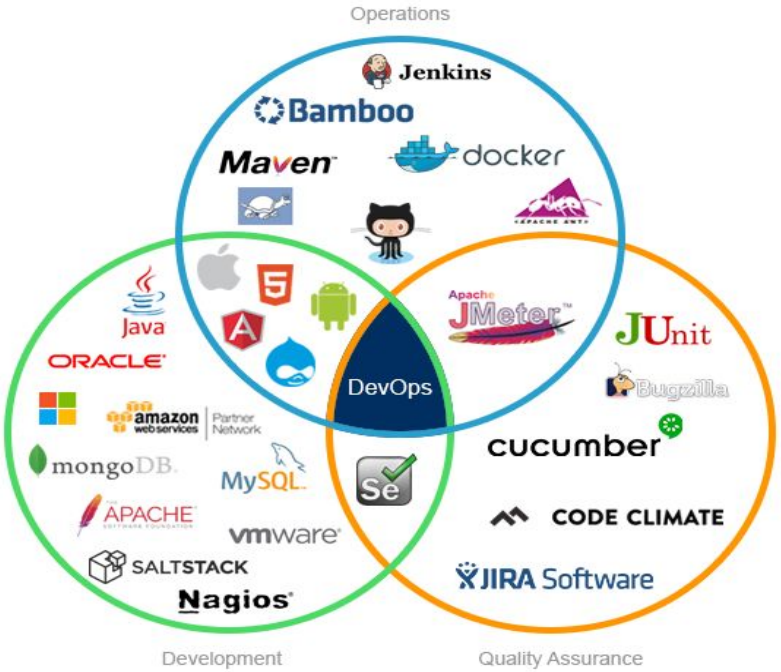
Zabbix Certified Specialist and Zabbix for Large Environments

LPIC

Onx Solutions

DevOps

- Evento intitulado “DevOps day” 2009 na Belgica
- Alinhamento do time de desenvolvimento com o de operações
- Visando acelerar as entregas em produção com um grau de qualidade
- Promove um conjunto de processos e métodos



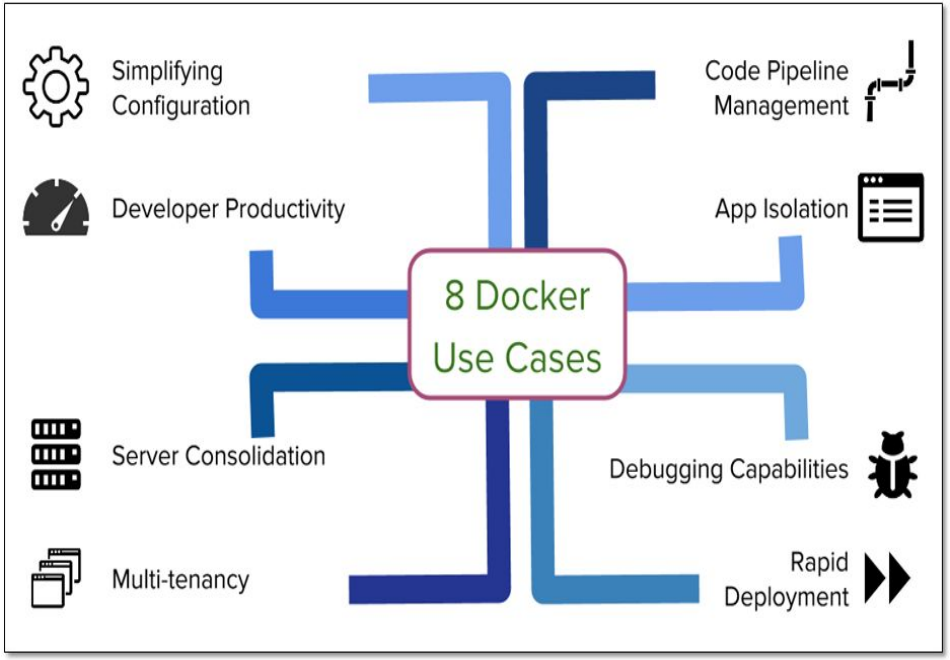
docker

ONX Solutions

O que é o docker

Criado em 2013 por Solomon Hykes em um projeto interno na dotCloud.

É uma tecnologia de código aberto que permite criar, executar, testar e implantar aplicações distribuídas dentro de **contêineres** de software.



docker

ONX Solutions

Plataforma suportadas

DOCKER CE

Platform	x86_64 / amd64	ARM	ARM64 / AARCH64	IBM Power (ppc64le)	IBM Z (s390x)
CentOS	✓		✓		
Debian	✓	✓	✓		
Fedora	✓				
Ubuntu	✓	✓	✓	✓	✓

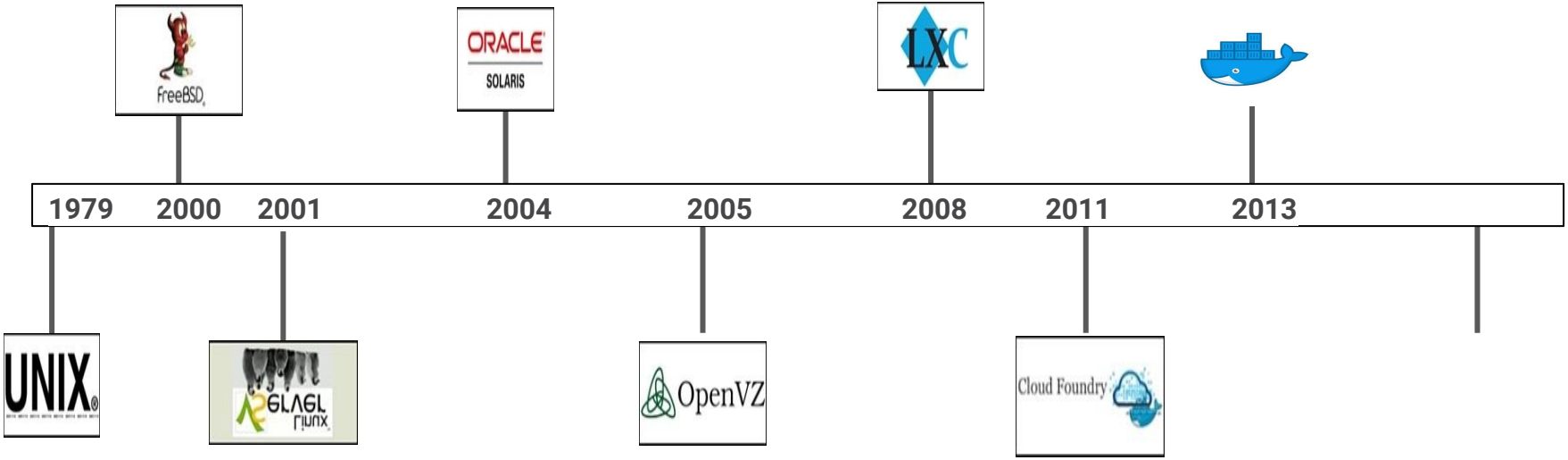
Desktop

Platform	Docker CE x86_64	Docker CE ARM	Docker EE
Docker for Mac (macOS)	✓		
Docker for Windows (Microsoft Windows 10)	✓		

docker

ONX Solutions

chroot ... docker

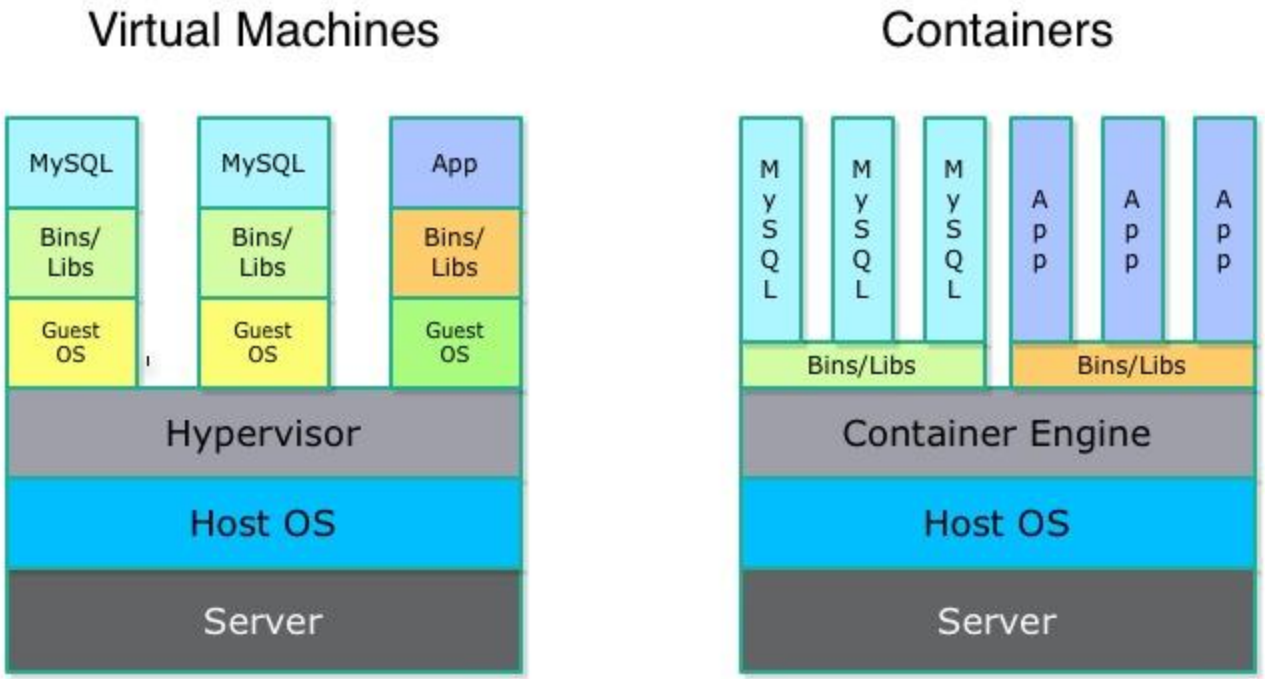


https://blog.aquasec.com/a-brief-history-of-containers-from-1970s-chroot-to-docker-2016

docker

ONX Solutions

Containers vs Virtual Machine



docker

ONX Solutions

Containers

- Criado a partir de uma imagem
- Geram novas imagens
- Conectividade com o host e outros containers
- Execução controlada - CPU, RAM, I/O, etc;
- Descartáveis ou persistentes

docker

ONX Solutions

Containers

```
$ docker run busybox echo ‘hellow word’
Unable to find image 'busybox:latest' locally
latest: Pulling from library/busybox
57310166fe88:  Pull complete
Digest:
sha256:1669a6aa7350e1cdd28f972ddad5aceba2912f589f19a090ac75b7083da
748db
Status: Downloaded newer image for busybox:latest
hellow word
```

ONX Solutions

Imagem

- Coleção de arquivos + alguns metadata(arquivos do root flilesystems);
- Layers empilhadas uma na outra;
- Pode ter camadas compartilhadas para otimizar uso do disco, tempo de transferência e memória

Exemplo:

- Centos,
- JRE
- Tomcat
- Application Jar

Container vs imagens

Imagem:

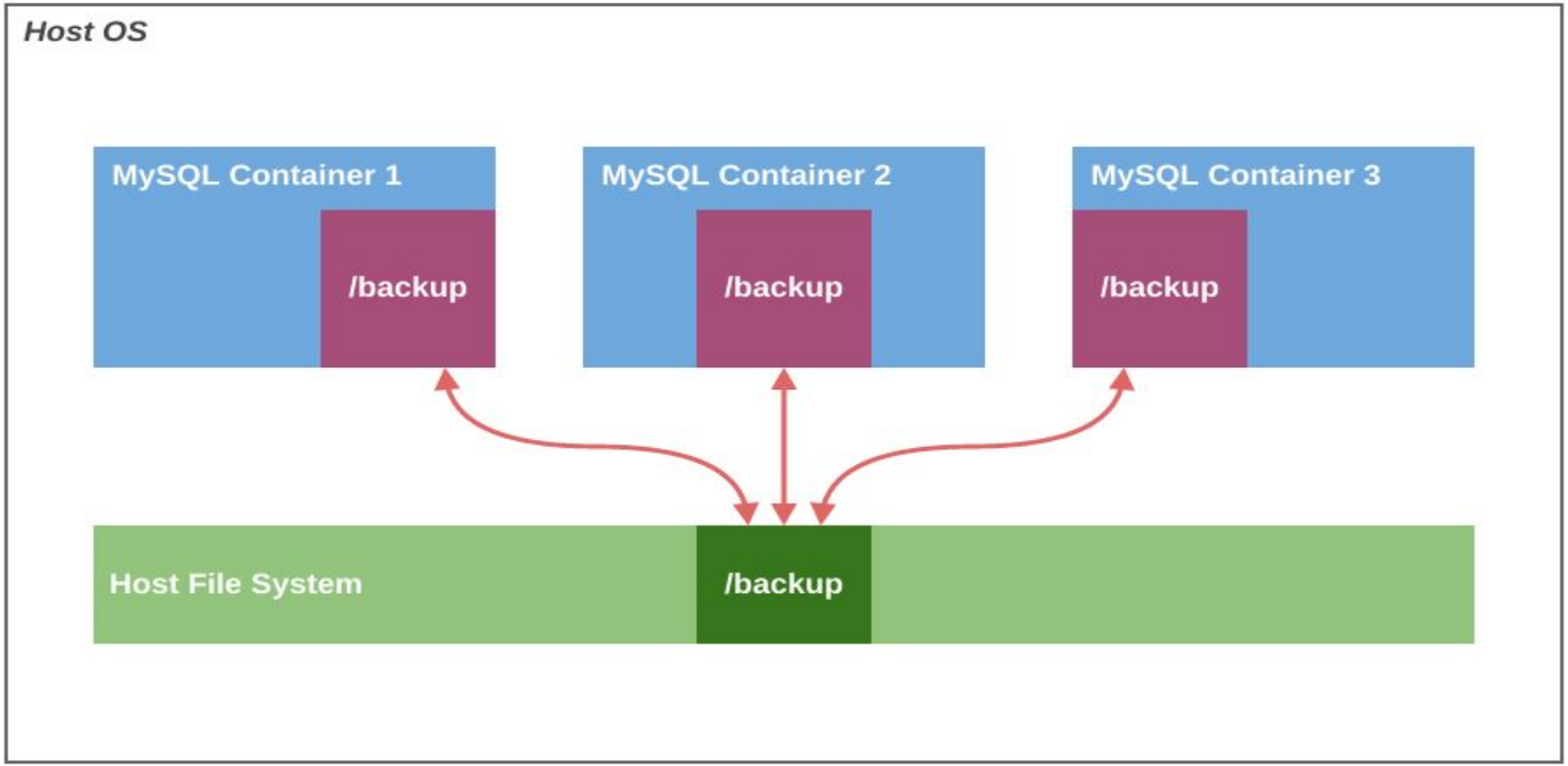
- Contém arquivos read-only do filesystem
- Pode executar vários containers
- Criado através do “docker build”

Container:

- Container encapsula o ambiente para rodar a aplicação **“docker run”** iniciando um container a partir de uma imagem
- Você pode ter +1000 container a partir de uma imagem

Volumes

- São diretórios configurados dentro de um container
- Fornece recursos de compartilhamento e persistência de dados.
- Podendo ser:
  - Compartilhado entre containers
  - Compartilhado entre host e um container.
  - Toda alteração feita em um volume é de forma direta.



docker

ONX Solutions

Volume, executando entre containers

```
$ mkdir /opt/nginx/
$ echo “Hello Word” > /opt/nginx/index.html

# Container 1:
$ docker run -d -p 8080:8080 -v /opt/nginx/:/usr/share/nginx/html:ro nginx

#Container 2 rodando um volume em outro container:
$ docker run -i -t -p 8081:8080 --volumes-from  <CONTAINER ID> nginx
# ls -lh /usr/share/nginx/html/index.html
-rw-rw-r-- 1 1000 1000 26 Feb 21 19:15 /usr/share/nginx/html/index.html
```

$man namespaces

Namespace:

- Camada isolamento para um processo dentro do container
- PID’s,UID’s namespace é remapeado dentro do container
- Importante para o docker pois fornece um modelo de isolamento
- Inclui net,mnt,pid,uts e user

```
$ docker run -it busybox /bin/sh
/ # ps -ef
PID   USER     TIME  COMMAND
    1 root      0:00 /bin/sh
    5 root      0:00 ps -ef
```

$man cgroups

Libera apenas o recursos necessários:

- CPU
- Memória
- I/O disco
- rede etc

```
$ docker run  -it  --memory=128m  --cpus=0.5
ubuntu
```

$man cgroups

Libera apenas o recursos necessários:

- CPU
- Memória
- I/O disco
- rede etc

```
$ docker run  -it  --memory=128m  --cpus=0.5
ubuntu
```

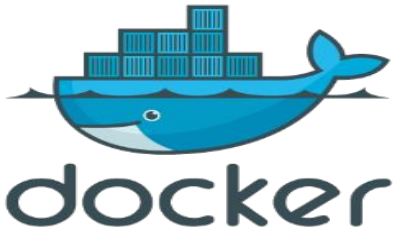
Rede

Quando o docker inicializado se cria interface “docker” e “veth” para o container.

docker0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default

veth10dc201@if8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master **docker0** state UP group default

vethbc4955e@if10: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master **docker0** state UP group default



Linux Kernel

Storage

Device Mapper | Btrfs | Aufs

Namespaces

PID | MNT | IPC | UTS | NET

Networking

veth | bridge | iptables

Cgroups

cpu | cpuset | memory | device

Security

Capability | SElinux | seccomp

Comandos

(Docker 1.13+)$ man docker container

```
# Lista todos os containers em
execução
$docker container ls

#Lista todos os container, mesmo
aqueles que não estão em execução
$docker container ls -a

# Para o container
$docker container stop <hash>
```

```
#Força o shutdown de um container
específico
$docker container kill <hash>

# Remove o container
$docker container rm <hash>

#Remove todos os containers
$docker container rm $(docker
container ls -a -q)
```

$ man docker ps

```
# Mostra o último container iniciado
$ docker ps -l

# Mostra somente o ID
$docker ps -q

# Mostra o ID  ultimo container inciado
$ docker ps -lq

# Lista todos os containers
docker ps -a
```

ONX
Solutions

$ man docker log e man docker inspect

```
# Mostra log dos container
$ docker logs  <ID CONTAINER>
$ docker logs --tail  10 4061001525f3 -- flat tail 10 == tail  10
Thu Feb 22 16:44:51 UTC 2018
Thu Feb 22 16:44:52 UTC 2018
Thu Feb 22 16:44:53 UTC 2018
Thu Feb 22 16:44:54 UTC 2018
Thu Feb 22 16:44:55 UTC 2018
Thu Feb 22 16:44:56 UTC 2018

# Inspecionar informações do Container
docker inspect  7546e2c63e90
docker inspect --format '{{ .State.Status }}'  7546e2c63e90
```

$ man docker images

```
#Lista todas as imagens
$docker image ls -a

$Remove uma imagem específica
$docker image rm <HASH IMAGE>

#Remove todas as imagens
$docker image rm $(docker image ls -a -q)

#Busca no Docker Hub as imagens
$docker search django
```

ONX
Solutions

$man docker commit and docker save

#salva todas as alterações feita no container

```
$docker commit <CONTAINERID>
$docker tag <CONTAINERID> name
```

Salva a imagem em formato tar

```
docker save --output busyboy.tar busybox
$ docker save busybox > busybox.tar
```

$ docker exec

```
#Acessando o container
$docker exec -ti <ContainerID> /bin/bash
```

ONX
Solutions

Criando container

ONX
Solutions

Criando intimidade com o container

```
$ docker run -it ubuntu
root@166b4eedaa43:/# apt-get update
root@166b4eedaa43:/# apt-get install figlet -y
```

Dockerfile

$ man Dockerfile

É um arquivo de texto que possui as rotinas de construção de uma imagem

Instruções que passada no Dockerfile:

- FROM,RUN
- COPY /ADD
- WORKDIR
- ENV
- EXPOSE
- VOLUME
- WORKDIR

ONX
Solutions

FROM e RUN

FROM

- Indica qual imagem base para geração
- Exemplo:

```
FROM ubunut:latest
FROM image:tag
```

RUN

- Usado para instalar bibliotecas, pacotes, arquivos e qualquer instrução
- Grava alterações feita no sistema de arquivo
- Exemplo:

```
#Forma de execução
RUN ["apt-get", "install","figlet"]
# Executa um shell - /bin/sh -c
RUN apt-get install figle
```

CMD e ENTRYPOINT

CMD

- Define um comando a ser executado quando um container for iniciado.

```
$docker run -it figlet Hellow GoInfra
```

- Providência argumentos default para o ENTRYPOINT.

```
CMD ["param1", "param2"]
```

- Executa no shell /bin/sh -c

```
CMD command param1 param2
```

ENTRYPOINT

- Define comando base para o container
- Não é sobrescrito
- COmbinando com CMD permite executar a imagem com argumentos:

```
FROM ubuntu:trusty
ENTRYPOINT ["/bin/ping","-c","3"]
CMD ["localhost"]
```

docker

ONX Solutions

COPY e ADD

ADD

- Equivalente ao COPY, mas contém features extras
- Faz requisição de arquivos por URL
- ADD automaticamente executa o unzip zip e tar do arquivo
- Contudo o ADD não descompacta arquivos remoto

  Exemplo:

  ```
  ADD test.tgz /absoluteDir/
  ```

COPY

- Similar ao ADD, porém com limitações
- Contém uma camada de transparência
- Copia arquivos locais

  Exemplo:

  ```
  COPY requirements.txt /tmp/
  ```

ONX Solutions

```
$ vim Dockefile

FROM ubuntu
RUN apt-get update
RUN ["apt-get", "install","figlet"]
ENTRYPOINT ["figlet", "-f","script"]
CMD ["Hellow"]
```

```
$ vim Dockerfile

FROM ubuntu
RUN apt-get update
RUN apt-get install -y nginx
RUN echo "daemon off;" >> /etc/nginx/nginx.conf
EXPOSE 80
CMD service nginx start
```

Rodando a imagem no Dockerfile

```
# Parametro -t usado para tag a imagem
$ docker build -t goinfra/nginx  .

# Executar o container com o nome na porta 8080 -i Mantém o STDIN aberto
-t aloca um Pseudo TTY
$docker run --name meetupgoinfra -p 8080:8080  -it goinfra/nginx

# Executa o container em uma porta aleatoria -P
docker run --name meetupgoinfra -P  -d -t goinfra/nginx

# Executa o container em background na porta 8080
docker run --name meetupgoinfra01 -p 8080:8080  -d goinfra/nginx
```

docker-compose

ONX
Solutions

docker compose

- Facilita a criação e uso de aplicações multi-container no Docker
- Múltiplos environments em um single host
- Automatizador de ambiente de teste
- Recria o contêiner quando tem alguma alteração

$vim docker-compose.yml

```
version: '3'
services:
  db:
    image: postgres
  web:
    build: .
    volumes:
      - .:/myapp
    ports:
      - "80:80"
    depends_on:
      - db
```

ONX Solutions

Algumas ideias

Docker em aplicação desktop

```
$docker pull andrey01/chrome
$docker run --rm -ti -v chrome_data:/data busybox
/bin/sh -c "chown 1000:1000 /data"
$
```

https://github.com/arno01/chrome/blob/master/docker-compose.yml

```
$  docker-compose up -d
```

docker

ONX Solutions

Docker Microsoft SQL Server

```
$docker pull microsoft/mssql-server-linux
```

```
$docker run -e 'ACCEPT_EULA=Y' --env 'SA_PASSWORD=123mudar!' -p
1433:1433 -d microsoft/mssql-server-linux
```

```
52f4e350aff3fb77d546ed75988b7e2ae4cf305f3a7463280efc9ca9aa775b4e
```

docker

ONX Solutions

Fonte

DevOps https://pt.wikipedia.org/wiki/DevOps
namespace:
https://success.docker.com/article/Introduction_to_User_Namespaces_in_Docker_Engine
cgroup: https://docs.docker.com/config/containers/resource_constraints
container vs image:
https://stackoverflow.com/questions/23735149/docker-image-vs-container
Networking: https://docs.docker.com/network/#docker-ee-networking-features
DOcker CE vs EE :https://www.mundodocker.com.br/docker-ce-ee/
ADD or Copy:
https://docs.docker.com/develop/develop-images/dockerfile_best-practices/#add-or-copy

Obrigado

Linkedin: @bezarsnba
GitHub:@bezarsnba
E-mail: bramos@onxsolutions.net

ONX
Solutions