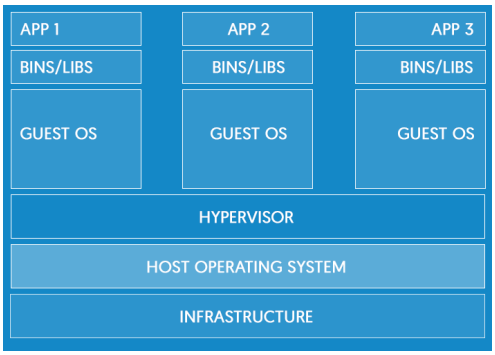
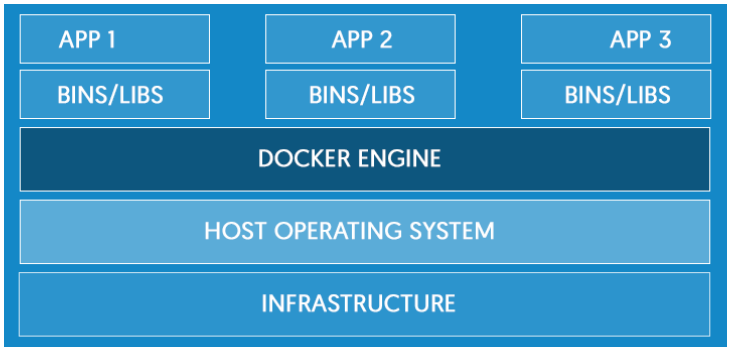
Site: <http://docker.com>

Documentação: <https://docs.docker.com>



Máquina Virtual



Container

Instalação:
Verificar se o kernel suporta AUFS: *grep AUFS /boot/config**

Instalar os pacotes: *sudo apt-get install apt-transport-https ca-certificates apparmor lxc cgroup-lite*

Adicionar chave:
*sudo apt-key adv --keyserver hkp://p80.pool.sks-keyservers.net:80 *
--recv-keys 58118E89F3A912897C070ADBF76221572C52609D

Adicionar repositório fornecido na documentação:
sudo nano /etc/apt/sources.list.d/docker.list
e adicionar a linha: *deb https://apt.dockerproject.org/repo ubuntu-trusty main*
Obs.:Escolha o repositório compatível com sua versão do linux:
Precise 12.04 (LTS): *deb https://apt.dockerproject.org/repo ubuntu-precise main*
Trusty 14.04 (LTS): *deb https://apt.dockerproject.org/repo ubuntu-trusty main*
Wily 15.10: *deb https://apt.dockerproject.org/repo ubuntu-wily main*
Xenial 16.04 (LTS) : *deb https://apt.dockerproject.org/repo ubuntu-xenial main*

Atualizar repositório e instalar o pacote: *sudo apt-get update;sudo apt-get install docker-engine*

Adicionar o usuário ao grupo docker: *sudo usermod -aG docker $USER*
Agora, deve-se deslogar e logar novamente. Ou reiniciar o computador.

Iniciar o serviço: *sudo service docker start*

Primeiros comandos:
id | grep -q docker && echo "Você está no grupo docker" || echo "Você não está no grupo docker"
pidof dockerd && echo "Docker iniciado" || echo "Inicie o serviço do docker"
docker version
docker run hello-world
docker ps -a
docker run -it ubuntu bash

Criação do container para sua aplicação

Crie o arquivo Dockerfile com o conteúdo:
FROM ubuntu
RUN echo deu certo > /CERTO
CMD /bin/bash

Opções do arquivo Dockerfile:
FROM: Informa a imagem de origem que deve ser utilizada para gerar a nova imagem.
Opções de imagens em: https://hub.docker.com/explore/
Obs.: deve ser a primeira linha do arquivo Dockerfile.
MAINTAINER: nome do mantenedor da nova imagem (opcional);
RUN: Especifica o que deve ser executado após ter a imagem pronta/criada (opcional);
CMD: Define um comando a ser executado quando um executar esse container;
LABEL: Adiciona metadados a uma imagem;
EXPOSE: Expõem uma ou mais portas que o container ficará em listen;
ENV: Define uma variável de ambiente dentro do container;
ADD: Adiciona arquivos locais ou de uma URL para dentro da imagem;
COPY: Copia arquivos ou diretórios locais para dentro da imagem.
ENTRYPOINT: Comando principal do container, que sempre será executado quando o container for aberto/executado.
Outras opções em: https://docs.docker.com/engine/userguide/eng-image/dockerfile_best-practices/

Para criar a imagem:
docker run -t NOME .
Opcionais: Em NOME, pode ser: categoria/nome ou categoria/nome:versão

Para visualizar todos os containers:
docker images

Para nomear seu container:
docker tag IMAGE_ID nome_desejado

Para remoer um container:
docker rmi IMAGE_ID
#ou
docker rmi REPOSITORY
Para forçar a remoção:
docker rmi -f IMAGE_ID
#ou
docker rmi -f REPOSITORY

Para usar seu container:
docker run -it NOME bash
Assim, você estará em um sistema enclausurado, sem acesso direto aos arquivos do sistema operacional que está utilizando fora do container. Aqui, você pode dar o comando apt-get update e instalar o que deseja. Exemplo da instalação do Apache2:
apt-get update
apt-get install net-tools apache2
ifconfig

Para verificar o que está em execução:

docker --config ~/testconfigs/ ps

Mais comandos em:

<https://docs.docker.com/engine/reference/commandline>