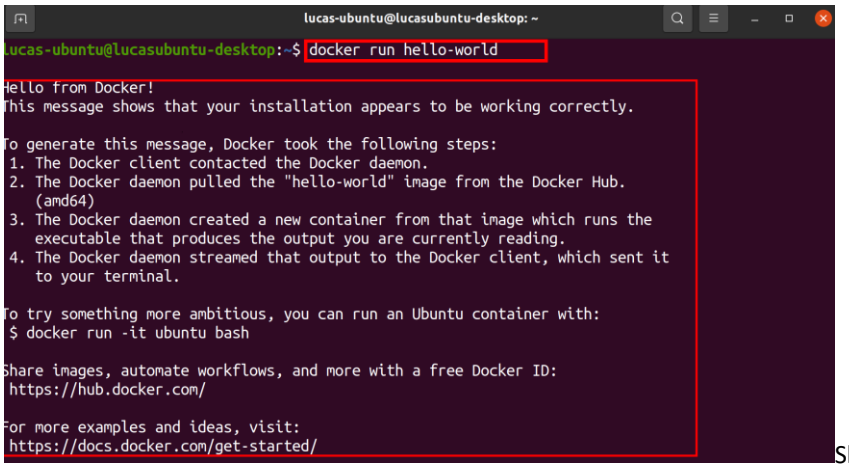
DevOps

Uma **definição rápida** sobre **DevOps** é que **ela** é uma cultura que **visa unificar práticas de desenvolvimento com práticas de operação**, em um português claro, o **dev** tem que saber como fazer um deploy e monitoramento decente da sua aplicação *(práticas de infra).*

Assim como o mundo de **desenvolvimento** tem diversas práticas *(SOLID, TDD, TESTES etc...)*, o mundo de **operações** também tem *(CI/CD, PROVISIONAMENTO, CONTEINERIZAÇÃO, etc..)* e todas elas merecem resenhas próprias.

Engine Docker

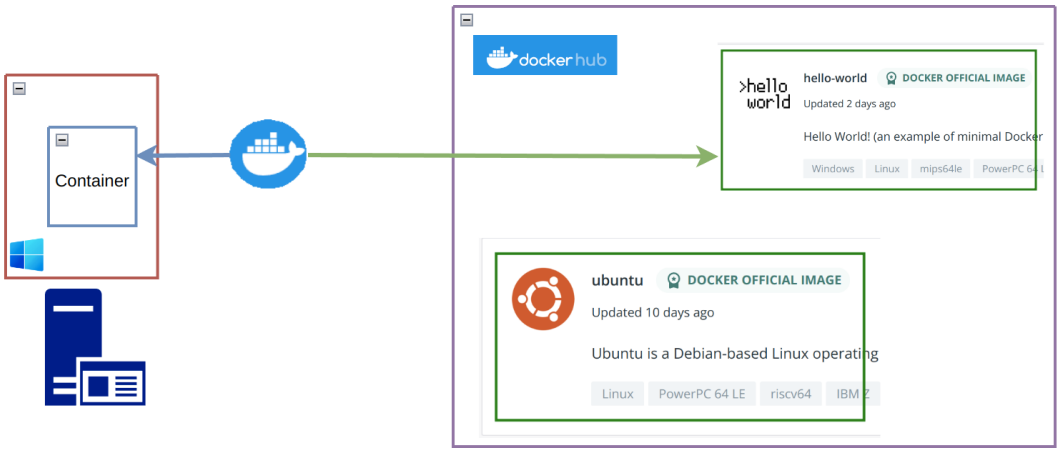
O **docker** é um motor de **contêiner,** ou seja, ele é capaz de criar e executar containers. Um exemplo da documentação é usar o comando ***"docker run hello-world"*** depois da instalação, e isso vai gerar uma saída desse tipo, (dá pra entender que executamos um container):



Show, só que **dá onde veio esse container? Isso é realmente um container?**

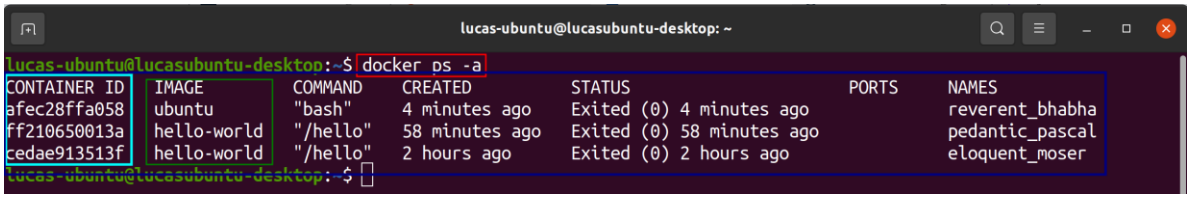
Imagens Docker

Na verdade, isso não é um **container**, isso é uma **IMAGEM**, e o docker buscou ela de um **repositório remoto (DockerHub) e** criou um **container**, então podemos entender que **imagens são receitas** para criação de um **contêiner.** Por mais que o **docker** não precise de um **SO** no **contêiner**, você ainda sim pode **criar containers com imagens baseadas em um SO específico:**



O comando ***"docker run [nomeImagem] "*** vai tentar encontrar a **imagem** na máquina, se não encontrar ele vai até o **repositório** atrás **dela**, se ele a encontrar vai executar o **container**.

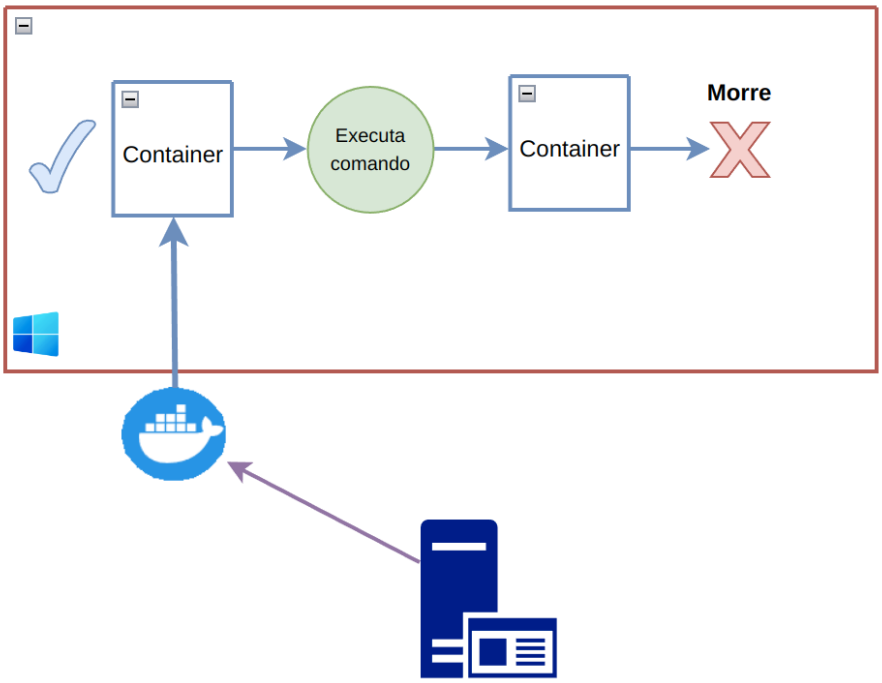
Dá pra baixar as **imagens** sem executa-las usando ***"docker pull [nomeImagem]"***, o **docker** vai baixa-la na **máquina local**, pra saber quais **imagens** você tem na máquina *(até mesmo as que não estão em execução)* dá pra usar ***"docker container ls -a"*** ou ***"docker ps -a"*** *(sim é a mesma coisa)*:



As i**nformações provenientes desse comando ajudam a entender o container,** então tem o id dele, a imagem usada e etc...

O mais importante nesse caso é a coluna de **COMMAND**, ela especifica qual comando o **contêiner** executou, então note que os **containers** de hello-world simplesmente ***printaram um hello na tela*** enquanto o **container** do ubuntu apenas ***abriu o terminal com "bash"***.

É por isso que os **containers** estão com o **status "Exited"**, eles foram executados, **fizeram o que tinha que fazer e pararam de rodar.** Para que um container se mantenha ativo, ele precisa de algum processo rodando nele, se ele não tiver nada para executar, ele morre:


Container
Executa
comando
Container
Morre