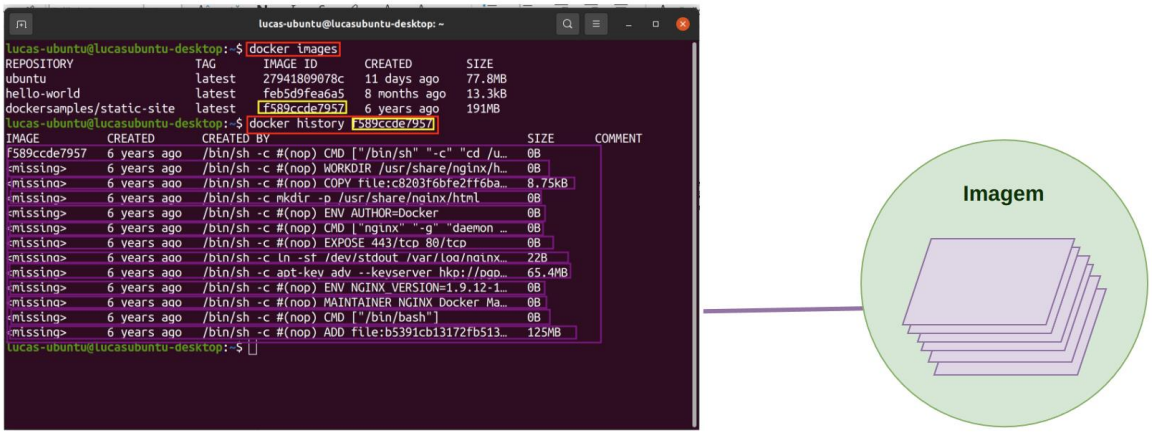
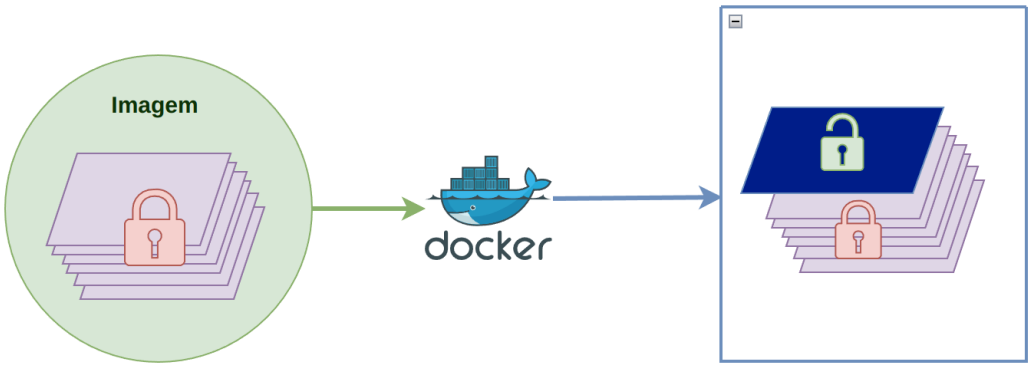
O que são **Imagens**

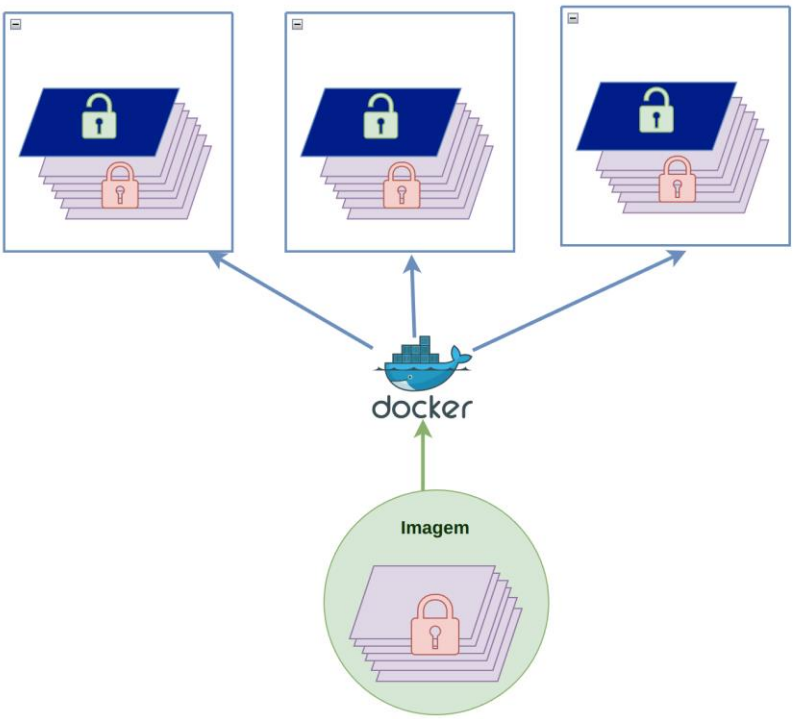
Uma **imagem** é composta por camadas, quando as camadas são empilhadas elas formam regras de execução do **container**. Falando assim é meio abstrato, então vou usar o **elemento visual**, ***camadas*** são os "comandos" que **compõe a imagem**, então se você usar o comando ***"docker history [idImagem]"*** as camadas dela podem ser visualizadas:



As imagens são **IMUTÁVEIS**, o que significa que são apenas para leitura, uma vez **montada** não podem ser alteradas. Mas como é possível escrever dentro de um **container** se as **imagens** são read only? Isso porque o **container** é uma **camada extra que permite a leitura/escrita:**



Esse conceito de camadas também ajuda a entender porque containers são mais leves que máquinas virtuais. Basicamente falando, o **Docker** é inteligente o bastante pra reaproveitar **camadas** já existentes na máquina, então toda vez que um **container** é montado, as **camadas** em comum de uma **imagem** são reaproveitadas:



Como são criadas Imagens

O Docker tem um arquivo especial que define como uma imagem deve ser criada, esse arquivo é o **Dockerfile**. O interessante é que **imagens** podem ser criadas com base em imagens já existentes, onde você pode definir ***variáveis de ambiente, portas para expor o container, diretórios principais, comandos e etc:***

Dockerfile

```
#define a imagem base da imagem que vai ser criada
FROM node:14

#cria uma pasta principal de trabalho dentro do container
WORKDIR /app-node

#variavel de ambiente durante a construção da imagem/ build do container
ARG PORT_BUILD=6000

#varial de ambiente dentro do container
ENV PORT=$PORT_BUILD

# deixa claro o valor da porta durante a listagem do container
EXPOSE $PORT_BUILD

# copia o que ta no diretório raiz do dockerfile pra dentro do workdir definido
COPY . .

# aqui eu to executando um comando de node na pasta de workdir, então ele sabe se virar com esse comando
RUN npm install

# executa toda vez que o container é iniciado
ENTRYPOINT npm start
```

Esse arquivo pode ser transformado em imagem com o comando ***"docker build –t xxxxxxxxxxx"*** o –t é uma flag para podermos nomear a imagem.