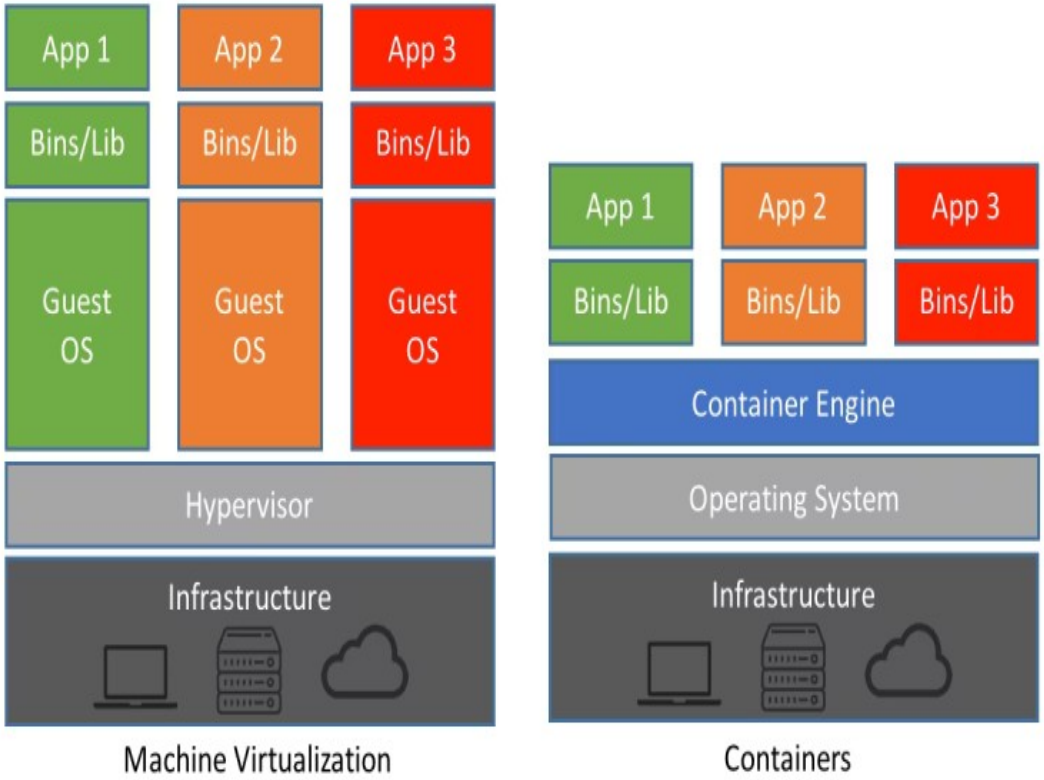
Docker básico

Contenedores



- Más liviano y rápido que una VM, fácil de distribuir
- NO virtualiza el sistema operativo completo, usa el kernel del host
- Es efímero, los datos se deben guardar en “volúmenes” separados

Contenedores

- Se puede envolver una aplicación en un contenedor para mantener un entorno de ejecución consistente
- Se aisla de las demás aplicaciones del sistema
- Se puede definir la distro de linux a usar
- Se pueden ajustar los recursos de hardware a usar
- Puedes definir las dependencias que necesites para la aplicación

Docker

- Es la herramienta más popular para manejar contenedores
- Término importantes:
  - Un contenedor es una copia “viva” de una imágen, como un objeto de una clase
  - Una imágen es una copia estática de una aplicación y sus dependencias, como una clase
  - Dockerfile es una archivo que se una para construir la imágen.
  - Volumen es un sistema de almacenamiento de datos que se conecta a los contenedores, pero no depende de ellos

Docker: comandos básicos

- Correr un contenedor
  - docker run [options] image[:tag] [comand] [arg...]
  - Ejemplos:
    - docker run -it -d –name mycontainer ubuntu:latest bash
    - docker run hello-world
    - -i: mantiene la entrada estandar abierta (permite escribir en el contenedor)
    - -t: permite pseudo-tty (consola)
    - -d: detach ejecuta el contenedor en background
    - --name nombre que se le asigna al contenedor

Docker: comandos básicos

- Descargar una imágen
  - docker pull image[:tag]
  - Ejemplo:
    - docker pull alpine
    - docker pull debian:bullseye

Docker: comandos básicos

- Ejecutar un comando
  - docker exec [options] container command [arg...]
  - Ejemplo:
    - docker exec mycontainer ls -l
    - docker exec -it mycontainer bash

Docker: comandos básicos

- Copiar datos
  - docker cp [options] container:SRC DST (del contenedor al host)
  - docker cp [options] SRC container:DST (del host al contenedor)
  - Ejemplo:
    - docker cp ./hola.txt mycontainer:/
    - docker cp mycontainer:/hola.txt ./

Docker: comandos básicos

- Listar contenedores
  - docker ps [-a] → docker container ls [options]
  - Ejemplo:
  - docker ps (lista contenedores activos)
  - docker ps -a (lista también los inactivos)
  - docker container ls
  - docker container ls -a

Docker: comandos básicos

- Construir una imágen
- Requiere un archivo Dockerfile donde se describen los pasos para construir la nueva imágen
  - docker build [options] FilePath
  - Ejemplo:
    - docker build -t my-image .
    - docker build -t my-image:2.0 ../NoDockerfile

Dockerile

- FROM image[:tag] (especifica la imágen base)
- RUN command (ejecutar comandos en shell, genera contenedores intermedios o capas)
- WORKDIR PATH (indica un nuevo directorio de trabajo y se posiciona en el)
- ADD SRC[...] DST copia archivos desde el directorio actual del host a la imágen, relativo a la ruta del Dockerfile)
- ENTRYPOINT [Command] (indica qué debe pasar al “hacer run” en la imágen

Dockerile

```
FROM ubuntu:latest
RUN apt-get -y update
RUN apt-get -y upgrade
RUN apt-get install -y build-essential
WORKDIR /calculator_app
ADD ./calculator/ ./
RUN make clean; make
ENTRYPOINT ["./calculator"]
```

Dockerile

```
FROM ubuntu:latest
RUN apt -y update
RUN apt install -y figlet
ENTRYPOINT ["figlet", “hola mundo”]
```

- docker build -t my-image . (crear imágen apartir del Dockerfile que hay en el directorio actual)
- docker run my-image (ejecutar un contenedor a partir de la imágen que acabo de crear)

Docker: otros comandos

- docker image ls (Listar imágenes)
- docker rm [options] nombre[…] (Eliminar contenedores)
  - -f (elimina contenedores que están activos)
- docker image rm image[…] (Eliminar imágenes)
- docker container prune (Elimina todos los contenedores inactivos)
- docker image prune [options] (similar al anterior, pero para imágenes)
- docker [re]start (iniciar o reiniciar un contenedor)
- docker stop (parar un contenedor) [mucho más en https://docs.docker.com/reference/cli/docker/]

Exponer una API en contenedor

- https://raw.githack.com/OAI/OpenAPI-Specification/master/examples/v3.0/petstore-expanded.yaml
- docker run -it -p 4010:4010 -v $(pwd):/tmp stoplight/prism mock -h 0.0.0.0 /tmp/refactored-petstore-openapi.yaml
  - -p CONTAINER_PORT:HOST_PORT (expone la aplicación en un puerto del host)
  - -v PATH_HOST:PATH_CONTAINER (mapea una ruta del host al contenedor como volumen)