Resumen de comandos Docker

#systemctl status/start/stop/restart/stop docker SERVICIOS

#/var/lib/docker FICHERO CONFIGURACIÓN

#docker version versión

#docker help ayuda de comandos docker

#docker run --name nombreContenedor hello-world crear un contendor

#docker run –d nginx crear un contenedor en segundo plano

#docker run –name -it Ubuntu /bin/bash crear un contendor interactivo

#docker rm $(docker ps -aq) –f forzar borrar todos los contenedores

#docker ps muestra los contenedores

#docker ps -a muestra los contenedores que están ejecutandose

#docker run –p 81:80 httpd:latest crear un contenedor al que se le asigna el puerto 80 y el 81 en el host

#netstat –lntu (o nte-tools) para ver los puertos abiertos

#docker inspect obtener información

#docker inspect ID|grep –i tcp para ver puertos abiertos en el contenedor

#docker exec –it nombreContenedor /bin/bash acceder a un contenedor

#docker attach nombreContenedor acceder a un contenedor

#docker start/stop/restart/ iniciar, parar, resetear

#cd /var/lib/docker/containers ver el tamaño de los contenedores

#docker stats estadísticas

#docker ps –s tamaño

RED

#docker network –help comandos del network.

#docker network create red1 crear una red

docker network create --help

#docker network create –subnet 192.168.3.0/24 red2

#docker network ls muestra las redes

#docker network inspect bridge inspeccionar una red

#ip a información ip

#nmcli con conexiones

#docker run -it --name mi-debian --network red1 debian Asociar a un contenedor una red en la creación

#docker network connect red2 mi-debian Asociar a un contenedor una red en CALIENTE

#docker run -it --name mi-busybox-3 --link mi-busybox-1:busy-1 busybox Crear un enlace

VOLUMENES

#/vae/lib/docker/volumes/ID/_data ubicación volumenes

#docker volumen ls mostrar volúmenes

#docker volume inspect ID Más información del volumen

#docker volume rm ID Borrar un volumen

#docker volume prune Borrar todos los volúmenes que no se usen

#docker run -it -v /datos --name mi-ubuntu ubuntu bash crear un contenedor con un volumen en la carpeta /datos

#docker volume create volumen-1 Crear un volumen independiente

#docker run -it --name mi-ubuntu -v volumen-1:/datos ubuntu bash Asociamos el volumen a un contenedor

#docker run -it --name mi-ubuntu-4 -v volumen-1:/datos:ro ubuntu bash Crear contenedor con un volumen en modo solo lectura

#docker diff mi-debian Para ver los cambios en un contenedor

IMAGENES

#docker commit mi-debian debian-manolo Creamos la imagen de forma manual

#docker images mostrar imágenes

#docker rmi images ID -f Borrar imagen

#docker search nombre_imagen Buscar

#docker pull nombre_imagen bajar

#docker run -it mi-imagen:v1 bash Construimos un contenedor con esa imagen

#docker build -t mftienda/cowsay:v1 . Creamos la imagen con el nombre de la cuenta de github (mftienda)

#docker image tag Nombre-Imagen:tag mftienda/Nombre-Imagen:tag Si ya tenemos la imagen creada, podemos cambiar el tag

SUBIR IMAGEN GITHUB

#docker login

#docker push mftienda/cowsay:v1 subir imagen a github

docker pull mftienda/cowsay:v1 bajar imagen a github

DOCKERFILE

vi Dockerfile crear archivo

FROM para descargar el sistema

RUN para ejecutar los comandos

#docker build -t mi-imagen:v1 . Construimos la imagen a partir del Dockerfile

#docker image history mi-imagen:v1 Comprobamos las capas de esa imagen

#docker run -it --name c1 mi-image:v1 bash Creamos contenedor

DOCKER COMPOSE

#curl -L "https://github.com/docker/compose/releases/download/1.27.4/dockercompose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose Instalar docker compose

#chmod +x /usr/local/bin/docker-compose le damos permiso al archivo

#docker-compose --version Comprobamos que está instalado

vi docker-compose.yml Creamos el fichero

#docker-compose up -d lanzar en segundo plano

#docker compose ENTER Vemos todos los comandos

#docker-compose ps vemos los procesos

#docker-compose start iniciar

#docker-compose stop pararlo

#docker-compose down eliminar contenedores y redes

SERVIDOR WEB

APACHE = apache2

var/www/html

usr/bin/apache2clt -D FOREGROUND

volumen var/www/html

SERVIDOR NGINX

nginx

usr/share/nginx/www

usr/sbin/nginx -g daemon off

volumen usr/share/nginx/www

REGISTRY

El puerto que utiliza por defecto es 5000

#docker pull registry Bajar imagen registry

#docker run -d -p 5000:5000 --name registro-1 registry Creamos un contenedor, registro-1 (Dpto Desarrollo), basado en esta imagen.

#docker tag debian localhost:5000/mi-debian creamos una imagen cambiando el nombre con el tag, debian es la imagen y lo demás el nombre de la imagen nuestra

#curl http://localhost:5000/v2/_catalog Ver imágenes

#docker pull localhost:5000/mi-hello Bajar imágenes

#chmod -R 777 /home/asir/plantilla/web dar permiso a la carpeta