Docker login | à ne pas oublier de se connecter

Docker pull hello-world. Il va chercher l'image du hello-world pour l'utiliser

Docker images | Montre les images présentes dans notre docker

Docker run hello-world | Il va créer un conteneur sur la base de l'image que nous l'avons donné

Si on fait run sans faire pull, run tout seul va faire un pull pour chercher l'image

Docker ps

Docker container ps On voit pas le hello-world c'est parce qu'il est pas démarrer

Docker ps -a ou -all c'est pour voir tous les conteneurs

On le voyé pas parce que il tourné pas en continue

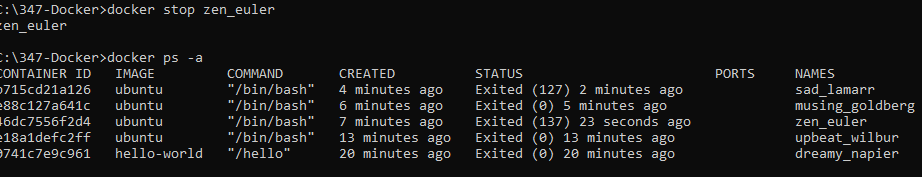
Docker pull ubuntu

Docker run -dti ubuntu | Disstach

Docker run -it ubuntu | Terminal interactif



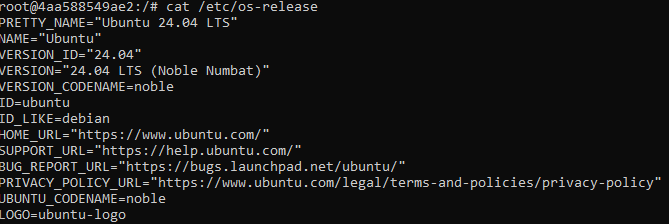
Docker ps -a



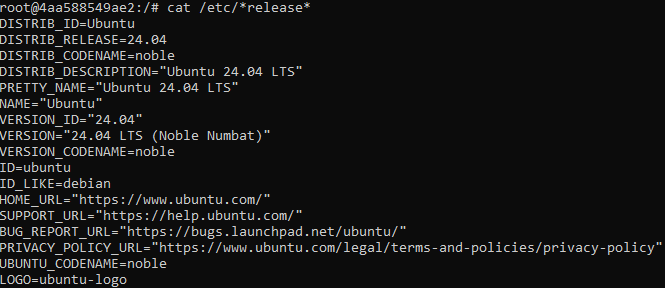
Docker stop nomcontainer

docker ps -a Tout les conteneurs sont exited

docker run -it ubuntu



cat /etc/\*release\*



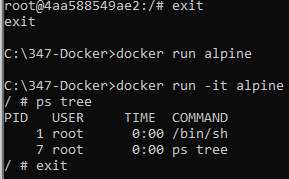
ls -l /etc/\*release\*

cat /etc/os-release

exit

docker run -it alpine

ps tree



Docker exec -it containerid /bin/bash

apt update

apt upgrade -y

apt -y install nano

nano

docker exec -it container-id ls -l | Nous sommes pas obliger de rentrer dans le conteneur pour exécuter la commande

docker rm container-id | supprimer le conteneur sans supprimer l’image

docker ps -q | % {docker stop $\_ }

stoppé les conteneurs utlisant powershell

docker ps -aq | % {docker rm $\_ } tout les conteneurs et quiet et seulement pour avoir un affichage réduit

docker system prune

docker run -dti –name Ubuntu-A ubuntu

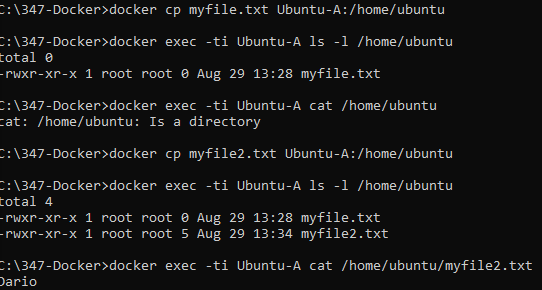
docker run -dti –name Cent0s-A centos

docker run -dti –name Ubuntu-B ubuntu

docker cp myfile.txt Ubuntu-A:/home/ubuntu

docker exec -ti Ubuntu-A ls -l /home/ubuntu

docker exec -ti Ubuntu-A cat /home/ubuntu/myfile.txt



Echo “test3 >> test3.txt

Docker exec -ti Ubuntu-A apt update

Docker exec -ti Ubuntu-A apt upgrade -y

Docker exec -ti Ubuntu-A apt -y install zip

Docker cp Myzip.zip Ubuntu-A:/home/ubuntu

Docker cp Ubuntu-A:/home/ubuntu/Myzip.zip Zipcopy.zip

Docker pull devian:11

Docker run -dti Debian:11

Il faut faire attn. car si la version n’est pas spécifié docker va faire un pull de lastest version

docker run -e MYSQL\_ROOT\_PASSWORD=pwd123 --name mysql-A -d -p 3306 mysql

Si on écrit pas les : il sélection le même port de deux côtés

-d detach shell c’est une session, comme sa on pourra continuer à écrire sans problème parce que il tourne en arrière

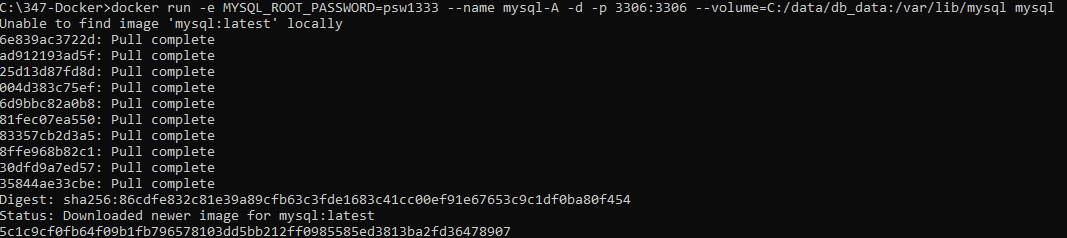
Docker exec -it mysql-A bash

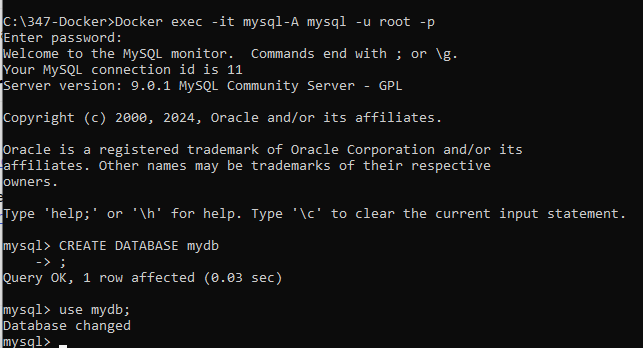
Docker exec -it mysql-A mysql -u root -p

CREATE DATABSE mydb;

SHOW DATABASES;

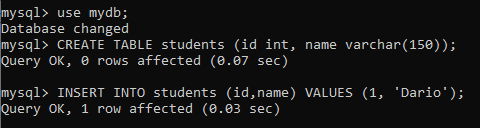
docker run -e MYSQL\_ROOT\_PASSWORD=psw1333 --name mysql-A -d -p 3306:3306 --volume=C:/data/db\_data:/var/lib/mysql mysql





CREATE TABLE students (id int, name varchar(150));

INSERT INTO students (id,name) VALUES (1, 'Dario');



docker run -dit --name apache-A -p 8080:80 -v C:/data/html:/usr/local//htdocs/ httpd:2.4



docker run -it --rm --name php-A -v C:/data/html:/usr/local//htdocs/ -w /usr/src/myapp php:8.2-cli php index.php

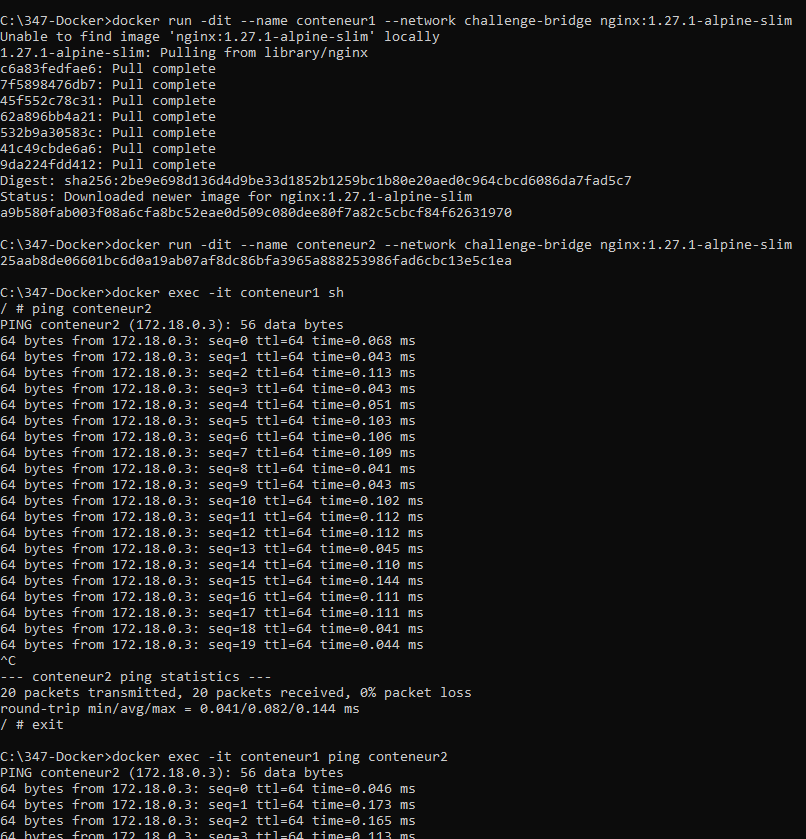


docker network create --driver bridge challenge-bridge

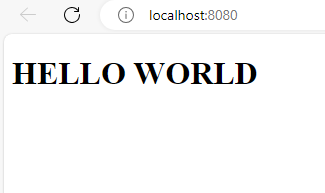
docker run -dit --name conteneur1 --network challenge-bridge nginx:1.27.1-alpine-slim

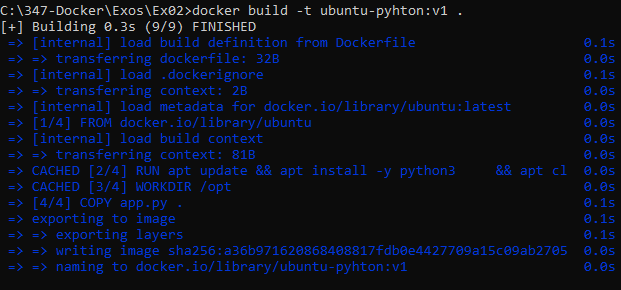
docker run -dit --name conteneur2 --network challenge-bridge nginx:1.27.1-alpine-slim

docker exec -it conteneur1 ping conteneur2

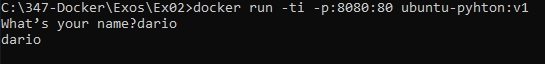








docker build -t ubuntu-pyhton:v1 .



docker run -ti -p:8080:80 ubuntu-pyhton:v1