TP1 :

Déf container : Un conteneur est une structure de données, une classe, ou un type de données abstrait, dont les instances représentent des collections d'autres objets. Autrement dit, les conteneurssont utilisés pour stocker des objets sous une forme organisée qui suit des règles d'accès spécifiques.On peut implémenter un conteneur de différentes façons, qui conduisent à des complexités en tempset en espace différentes.

Déf Docker : Docker est une plate-forme logicielle qui vous permet de concevoir, tester et déployerdes applications rapidement. Docker intègre les logiciels dans des unités normalisées appelées conteneurs, qui rassemblent tous les éléments nécessaires à leur fonctionnement, dont les bibliothèques, les outils système, le code et l'environnement d'exécution. Avec Docker, vous pouvezfacilement déployer et dimensionner des applications dans n'importe quel environnement, avec l'assurance que votre code s'exécutera correctement.

Étape 1 TP :

Fiche procédure installation docker :

Étape1 : mettre à jour le dépôt Debian avec le dépôt docker

Taper dans l'ordre les commandes suivantes :

```
$ su -
```

```
# apt-get update
```

```
# apt-get install ca-certificates curl gnupg
```

Exemple en image :

```
root@debian-xfce:~# $ apt-get install ca-certificates curl gnupg
-bash: $ : commande introuvable
root@debian-xfce:~# apt-get install ca-certificates curl gnupg
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
ca-certificates est déjà la version la plus récente (20230311).
curl est déjà la version la plus récente (7.88.1-10+deb12u1).
gnupg est déjà la version la plus récente (2.2.40-1.1).
0 mis à jour, 0 nouvellement installés, 0 à enlever et 16 non mis à jour.
root@debian-xfce:~#
```

```
# install -m 0755 -d /etc/apt/keyrings
```

Exemple en image :

```
root@debian-xfce:~# install -m 0755 -d /etc/apt/keyrings
root@debian-xfce:~#
```

```
# curl -fsSL https://download.docker.com/linux/debian/gpg |  gpg
--dearmor -o /etc/apt/keyrings/docker.gpg
```

Exemple en image :

```
root@debian-xfce:~# curl -fsSL https://download.docker.com/linux/debian/gpg |
 gpg --dearmor -o /etc/apt/keyrings/docker.gpg
Le fichier « /etc/apt/keyrings/docker.gpg » existe. Faut-il réécrire par-dessu
s ? (o/N) o
root@debian-xfce:~#
```

```
# chmod a+r /etc/apt/keyrings/docker.gpg
```

Exemple en image :

```
root@debian-xfce:~# chmod a+r /etc/apt/keyrings/docker.gpg
root@debian-xfce:~#
```

```
# echo \ "deb [arch="$(dpkg --print-architecture)"
signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/debian \

  "$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \
   tee /etc/apt/sources.list.d/docker.list > /dev/null
 apt-get update
```

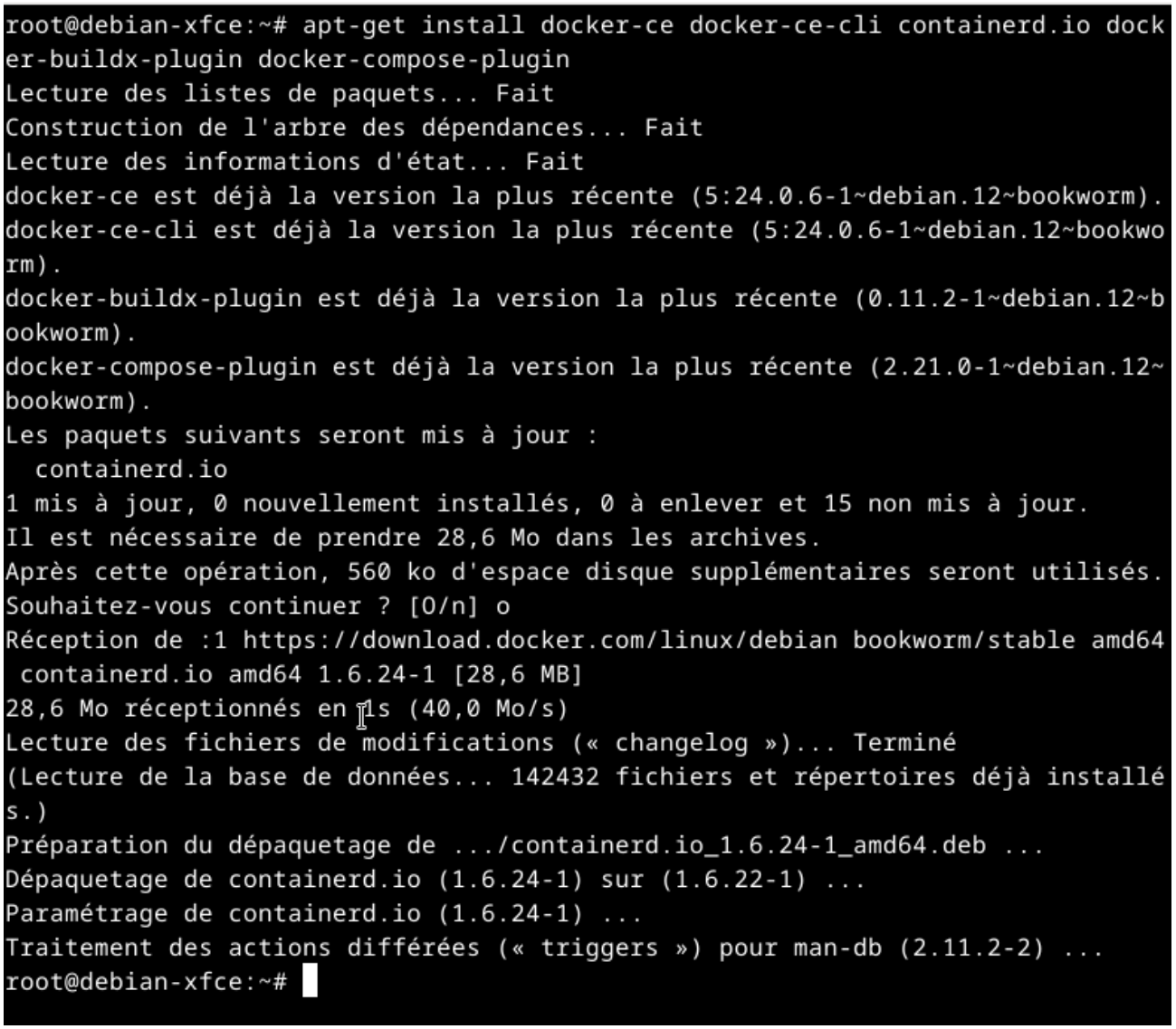
Exemple en image :


```
# apt-get update
```

Étape 2 : installer docker via son paquet d’installation

```
# apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

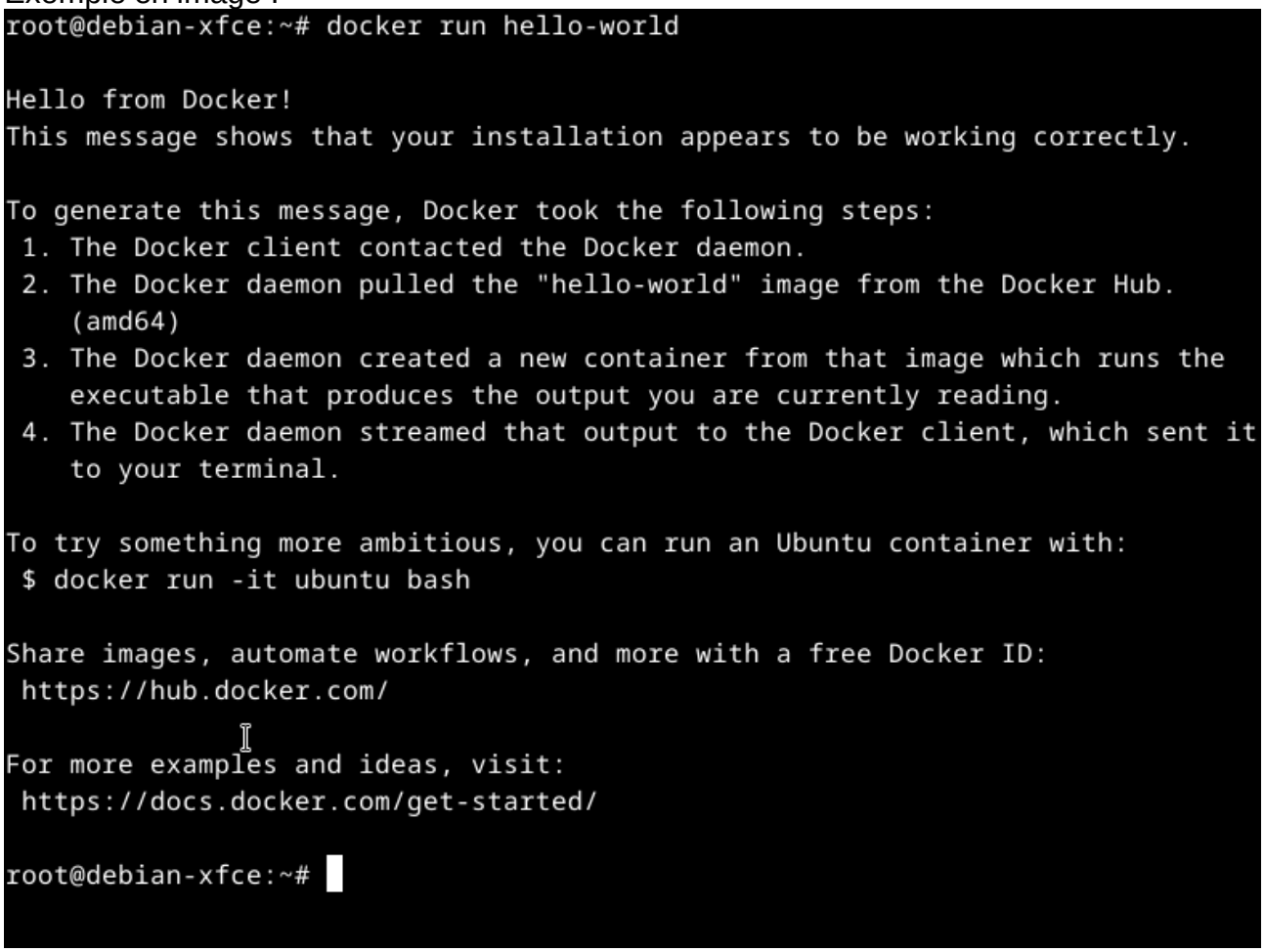
Exemple en image :



Étape 3 : vérifier l'installation en lançant l'image hello-world


```
# docker run hello-world
```

Exemple en image :



Étape 2:

Découverte commande de base docker :

Récupérer une image :

```
# docker pull nom_de_l'image
```

Exemple en image :

```
root@debian-xfce:~# docker pull nginx
Using default tag: latest
latest: Pulling from library/nginx
a803e7c4b030: Pull complete
8b625c47d697: Pull complete
4d3239651a63: Pull complete
0f816efa513d: Pull complete
01d159b8db2f: Pull complete
5fb9a81470f3: Pull complete
9b1e1e7164db: Pull complete
Digest: sha256:32da30332506740a2f7c34d5dc70467b7f14ec67d912703568daff790ab3f75
5
Status: Downloaded newer image for nginx:latest
docker.io/library/nginx:latest
root@debian-xfce:~#
```

Lister les images :

```
# docker images
```

Exemple en image :

```
root@debian-xfce:~# docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
nginx         latest    f5a6b296b8a2   2 weeks ago    187MB
hello-world   latest    9c7a54a9a43c   4 months ago   13.3kB
root@debian-xfce:~#
```

Démarrer un conteneur Nginx sur un autre port :

```
# docker run --name nom_du_conteneur -d -p 8080:80 nom_de_l'image
```

Exemple en image :

```
root@debian-xfce:~# docker run --name nginx -d -p 8080:80 nginx
e5bbec68fa64cd298e0a9b974c7ebaa63719ca1b994179a54f781e7826cb13b7
root@debian-xfce:~#
```

lister tous les conteneurs:

```
# docker container ls -a
```

Exemple en image :

```
root@debian-xfce:~# docker container ls -a
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS
                    PORTS                                      NAMES
e5bbec68fa64   nginx          "/docker-entrypoint.…"   23 seconds ago   Up 22
seconds             0.0.0.0:8080->80/tcp, :::8080->80/tcp   nginx
ab8d8938f431   hello-world    "/hello"                 9 minutes ago    Exited
 (0) 9 minutes ago                                          magical_shannon
b8229dc6d38e   f5a6b296b8a2   "/docker-entrypoint.…"   13 days ago      Exited
 (0) 13 days ago                                            some-nginx
cc05e18a006b   hello-world    "/hello"                 13 days ago      Exited
 (0) 13 days ago                                            busy_goldwasser
ba6ea2af92ae   hello-world    "/hello"                 13 days ago      Exited
 (0) 13 days ago                                            quizzical_faraday
b6e1addb8b49   hello-world    "/hello"                 13 days ago      Exited
 (0) 13 days ago                                            hopeful_jennings
root@debian-xfce:~#
```

id nginx:
f9d8e99fc572f03472017519034b973d3062fde533d2f679cd88627cea86b56e

nom nginx:
some-nginx

Lister les conteneurs :

```
# docker container ls
```

Exemple en image :

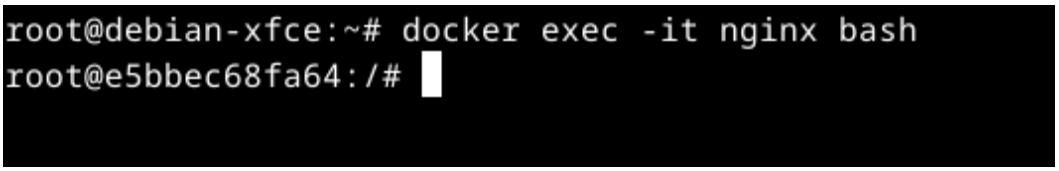
```
root@debian-xfce:~# docker container ls
CONTAINER ID   IMAGE     COMMAND                  CREATED              STATUS
         PORTS                                    NAMES
e5bbec68fa64   nginx     "/docker-entrypoint.…"   About a minute ago   Up 59 s
econds   0.0.0.0:8080->80/tcp, :::8080->80/tcp   nginx
root@debian-xfce:~#
```

status: up

executer un shell bash:

```
# docker exec -it nom_du_conteneur bash
```

Exemple en image :



Retour uname : Linux f9d8e99fc572 6.1.0-11-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.38-4 (2023-08-08) x86_64 GNU/Linux

Stopper un conteneur :

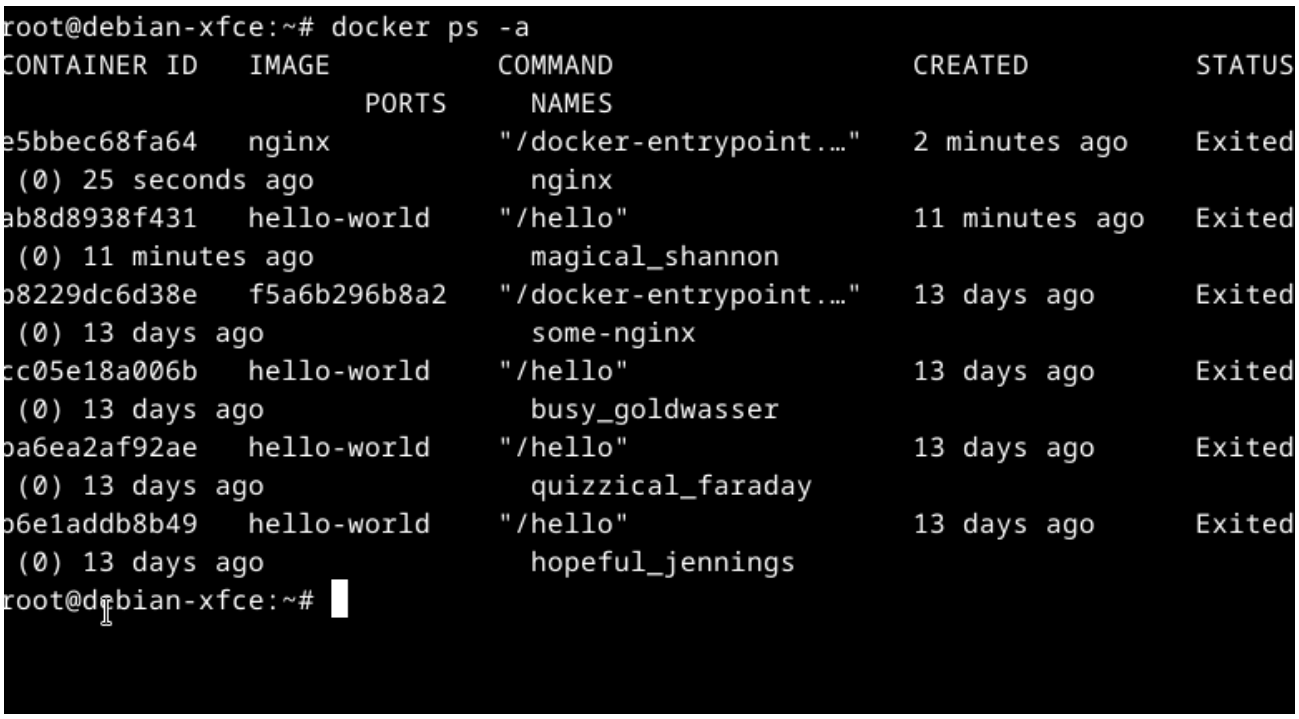
```
# docker stop nom_du_conteneur
```

Exemple en image :

Lister les processus afin de lister les conteneurs :

```
# docker ps -a
```

Exemple en image :

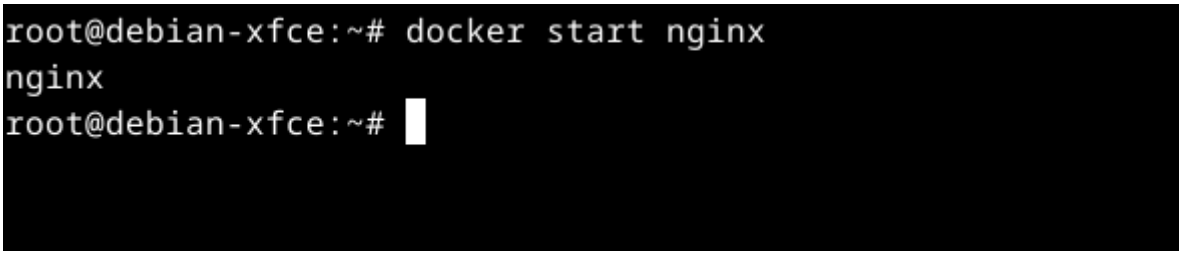


Status : exited

Lancer un conteneur :

```
# docker start nom_du_conteneur
```

Exemple en image :



Lister les processus afin de lister les conteneurs :

```
# docker ps -a
```

Exemple en image :

```
root@debian-xfce:~# docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS
                     PORTS                                       NAMES
e5bbec68fa64   nginx          "/docker-entrypoint.…"   4 minutes ago    Up 29
seconds              0.0.0.0:8080->80/tcp, :::8080->80/tcp   nginx
ab8d8938f431   hello-world    "/hello"                 12 minutes ago   Exited
 (0) 12 minutes ago                                          magical_shannon
b8229dc6d38e   f5a6b296b8a2   "/docker-entrypoint.…"   13 days ago      Exited
 (0) 13 days ago                                             some-nginx
cc05e18a006b   hello-world    "/hello"                 13 days ago      Exited
 (0) 13 days ago                                             busy_goldwasser
ba6ea2af92ae   hello-world    "/hello"                 13 days ago      Exited
 (0) 13 days ago                                             quizzical_farada
y
b6e1addb8b49   hello-world    "/hello"                 13 days ago      Exited
 (0) 13 days ago                                             hopeful_jennings
root@debian-xfce:~#
```

Status : up

Supprimer un conteneur :

stoper le container puis éxecuter la commande suivante:

```
# docker container rm nom_du_conteneur
```

Exemple en image :

```
root@debian-xfce:~# docker container rm nginx
Error response from daemon: You cannot remove a running container e5bbec68fa64
cd298e0a9b974c7ebaa63719ca1b994179a54f781e7826cb13b7. Stop the container befor
e attempting removal or force remove
root@debian-xfce:~# docker stop nginx
nginx
root@debian-xfce:~# docker container rm nginx
nginx
root@debian-xfce:~#
```

Vérification de la suppression du conteneur :

```
# docker ps -a
```

Exemple en image :

```
root@debian-xfce:~# docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS
                     PORTS     NAMES
ab8d8938f431   hello-world    "/hello"                 14 minutes ago   Exited
 (0) 14 minutes ago             magical_shannon
b8229dc6d38e   f5a6b296b8a2   "/docker-entrypoint.…"   13 days ago      Exited
 (0) 13 days ago                some-nginx
cc05e18a006b   hello-world    "/hello"                 13 days ago      Exited
 (0) 13 days ago                busy_goldwasser
ba6ea2af92ae   hello-world    "/hello"                 13 days ago      Exited
 (0) 13 days ago                quizzical_faraday
b6e1addb8b49   hello-world    "/hello"                 13 days ago      Exited
 (0) 13 days ago                hopeful_jennings
root@debian-xfce:~#
```

Le container nginx n'apparait plus

Supprimer une image :

```
# docker image rm nom_de_l'image
```

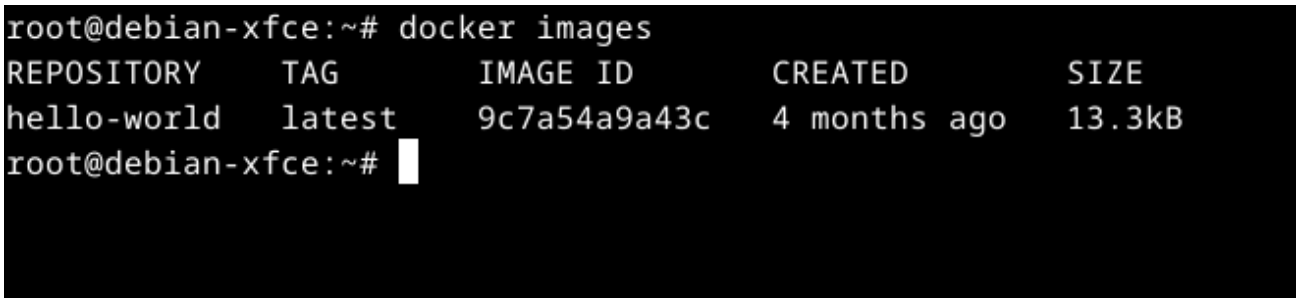
Exemple en image :

```
root@debian-xfce:~# docker image rm nginx
Untagged: nginx:latest
Untagged: nginx@sha256:32da30332506740a2f7c34d5dc70467b7f14ec67d912703568daff7
90ab3f755
Deleted: sha256:61395b4c586da2b9b3b7ca903ea6a448e6783dfdd7f768ff2c1a0f3360aaba
99
Deleted: sha256:1c69f36a0d9b59b762eaba410fa9fd01b85140670a8d49199a7b37702cc956
c0
Deleted: sha256:bac209bfab6997cccf20779ae98d5f77a66867734499ecf604a50a5826f6b8
a4
Deleted: sha256:859676f4cd3004af025a02dade096ad6f9391d94d1b1a983fc6098debe4350
55
Deleted: sha256:cbbd97cee0d824e5e82f9a4b2e93c5eb3c66fd72a2624d5c1521dc3395bfd1
e2
Deleted: sha256:0b41545d8c3de3b78778d591a8da3d9dfa5fa8807baef5edf21e5eb94ded79
2d
Deleted: sha256:7e87866b23143eb30086086a669b2e902368a5836446a885b2411d3feef18b
ef
Deleted: sha256:d310e774110ab038b30c6a5f7b7f7dd527dbe527854496bd30194b9ee6ea49
6e
root@debian-xfce:~#
```

Vérification de la suppression de l'image :

```
# docker images
```

Exemple en image :



L'image nginx n'apparait plus dans la liste

Étape 3 :

Créer une image à partir de debian:

-créer un dossier qui va contenir le Dockerfile
```
# mkdir nom_du_dossier
```

-se déplacer dans le dossier
```
# cd nom_du_dossier
```

-créer un fichier Dockerfile :

```
nom_du_dossier# touch Dockerfile
```

-écrire un fichier Dockerfile avec la syntaxe suivante :

```
nom_du_dossier# nano Dockerfile
```

Texte du Dockerfile :

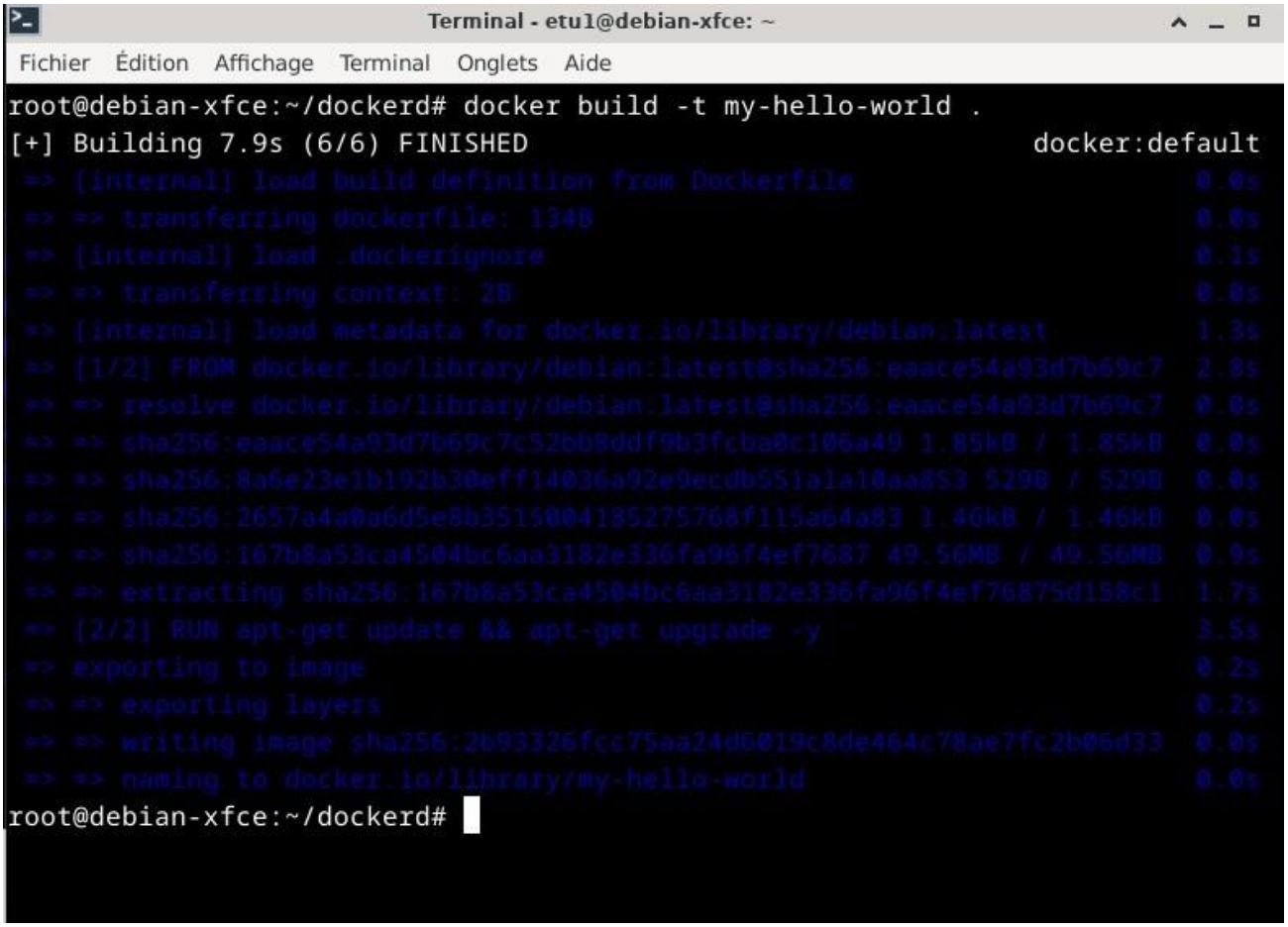
```
FROM debian:latest

USER root

RUN apt-get update && apt-get upgrade -y

CMD echo "Hello World!"
```

Exemple en image:

Commande a taper pour construire l’image avec le Dockerfile :

```
nom_du_dossier# docker build -t nom_de_l’image .
```

Exemple en image :



Créer une image et un conteneur nginx avec un html personalisé :

-créer le fichier html utilisé et placer le dans le dossiercontenant le Dockerfile

```
nom_du_dossier# touch index.html
```

-remplir le fichier html

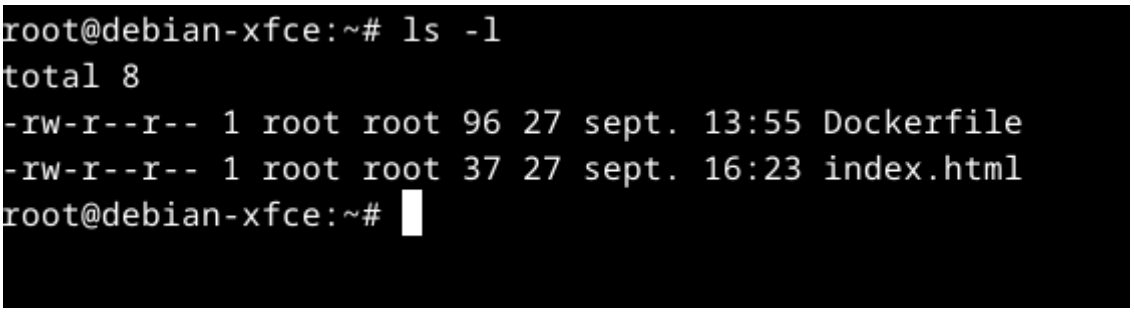
```
nom_du_dossier# nano index.html
```

```
<body>
<p>Hello World !</p>
</body>
```

Exemple en image :

-le fichier index.html est bien dans le dossier courant

ls -l

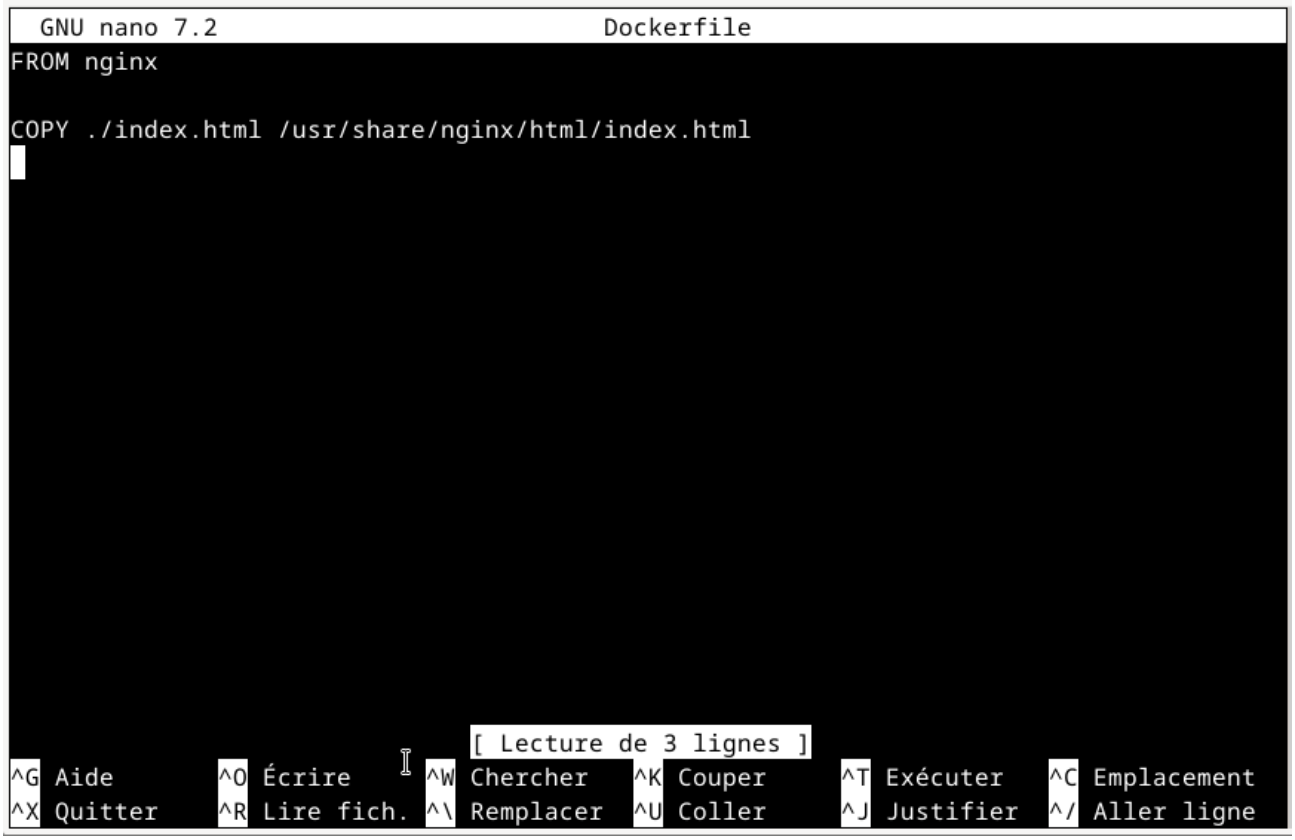


-modifier le Dockerfile pour utiliser l'image nginx
nom_du_dossier# nano Dockerfile

-remplir le Dockerfile avec les lignes suivantes

From nginx

Copy ./index.html /usr/share/nginx/html/index.html
Exemple en image :



-créer l'image

```
nom_du_dossier# docker build -t nom_de_l'image .
```

Exemple en image :

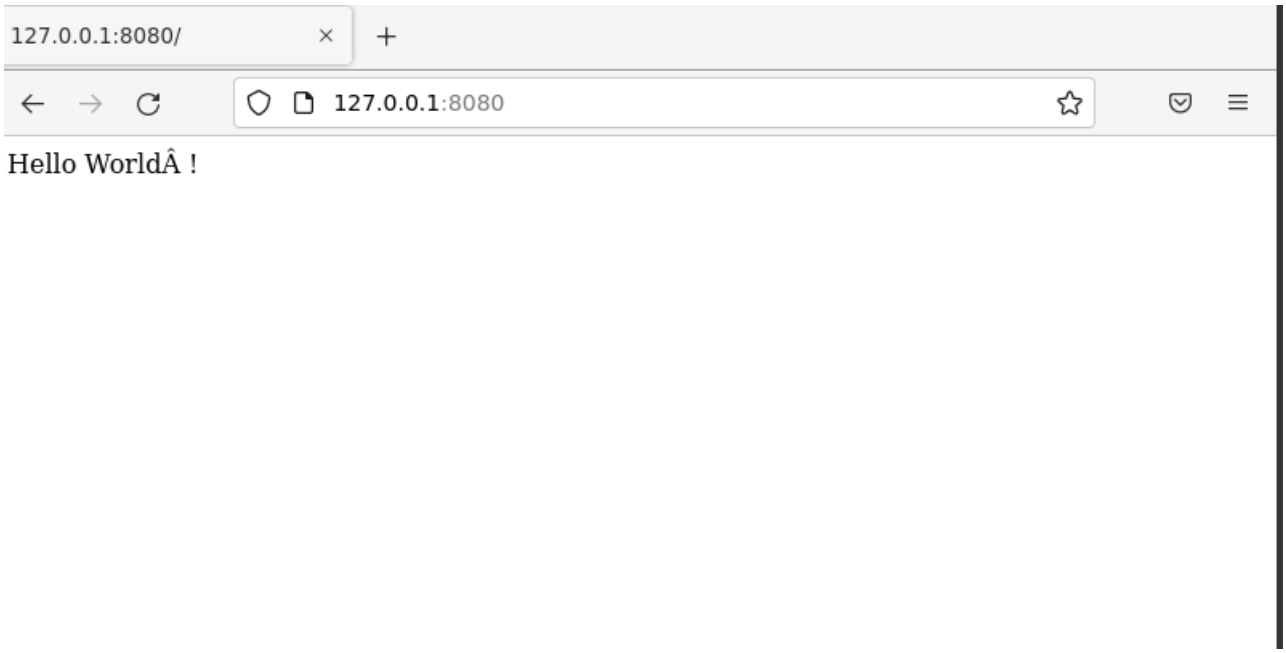
-lancer un conteneur à partir de l'image

nom_du_dossier# docker run -d –name nom_du_conteneur -p 8080:80 nom_de_l'image

Exemple en image :

```
root@debian-xfce:~/dockerd# docker run -d --name nginx1 -p 8080:80 my-nginx
9fcd3a03c69275d289018dfd07ed8f229186e47d4f6fca0405418f62dbf7dabe
root@debian-xfce:~/dockerd#
```

-ouvrir le navigateur
-taper l'url 127.0.0.1:8080



Étape 4:

Créer une image et un conteneur nginx sur le port8080 avec docker compose :

-créer un fichier compose.yml dans le dossier docker :

nom_du_dossier# nano compose.yml
-modifier le fichier compose.yml :

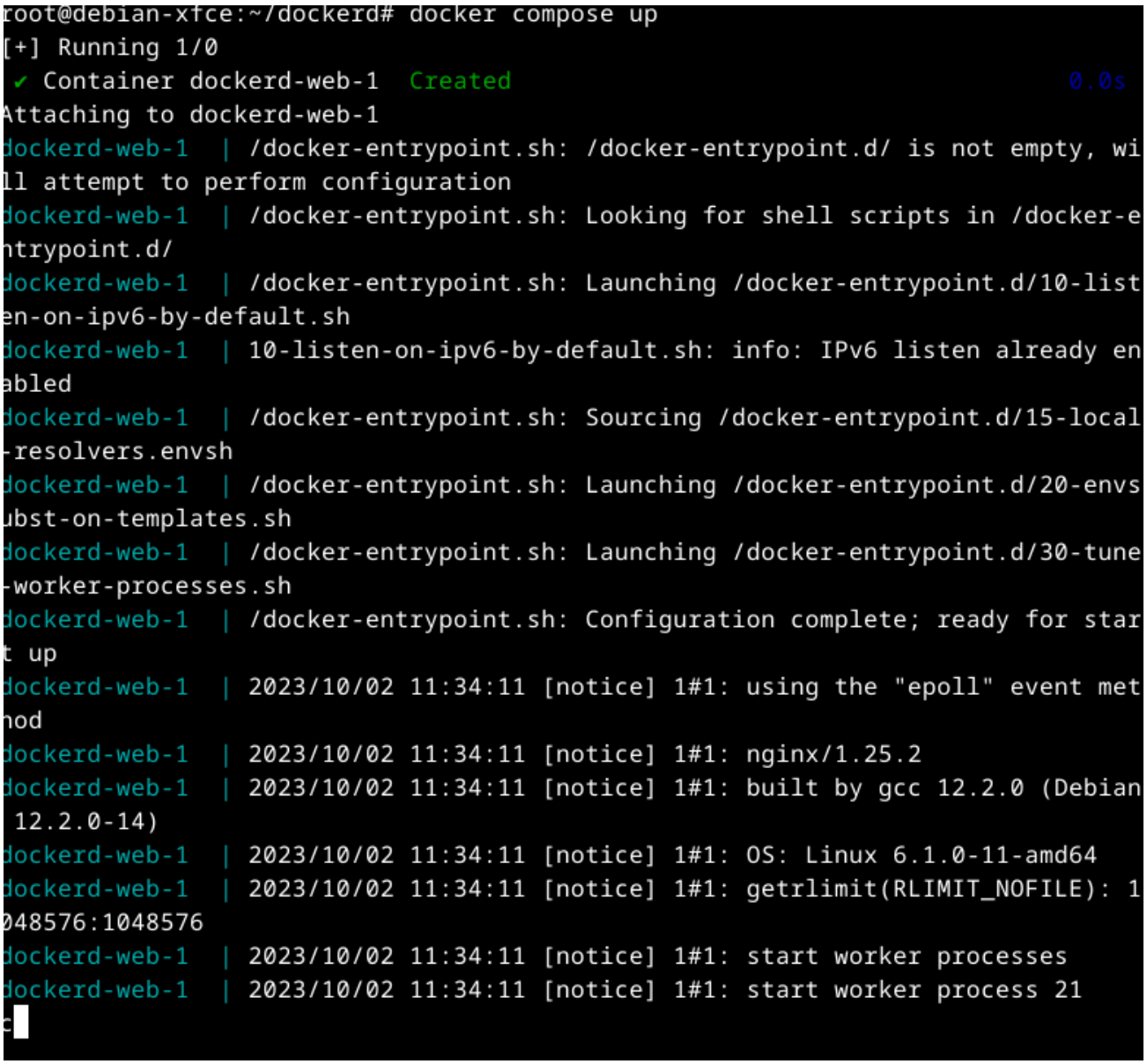
```
version:
'3'
services:
  web:
    image: nginx:latest
    ports:
    - "8080:80"
    volumes:
    - /usr/share/nginx/html
```

Exemple en image :

-taper la commande suivante pour lancer docker compose en premier plan :

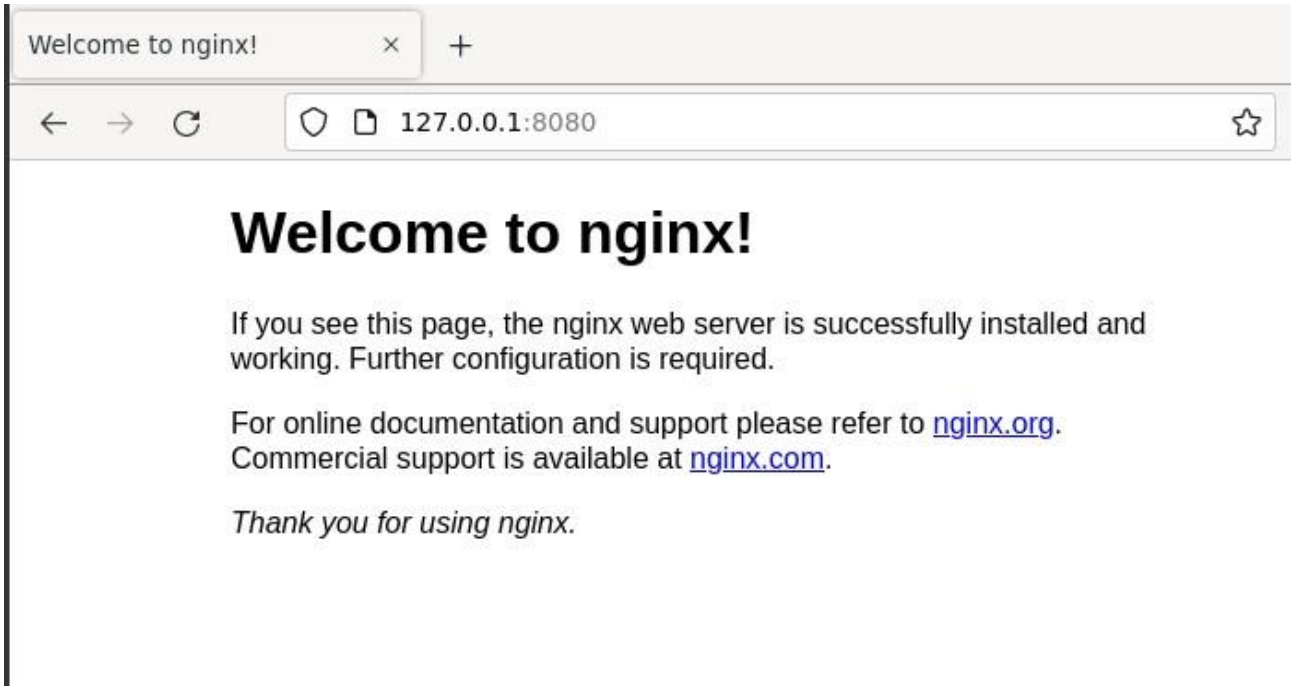
```
Nom_du_dossier# docker compose up
```

Exemple en image :



- -ouvrir le navigateur

-taper l’url 127.0.0.1:8080



Stopper le conteneur lancer avec docker composelancer en premier plan :

Appuyer sur ctrl + c dans le terminal afin de stopper le conteneur

Lancer le docker compose en arrière-plan :

Au lieu de taper la commande docker compose up taper la commande suivante :

```
nom_du_dossier# docker compose up -d
```

Récupérer les logs de docker compose :

```
nom_du_dossier# docker compose logs
```

Stopper un conteneur docker compose lancer enarrière-plan :

```
nom_du_dossier# docker compose stop
```

Installer wordpress via postgreSQL sur docker compose :

Compose.yml:

```
version: '3.8'

services:

    wordpress:
        depends_on:
            - "postgres"
        image: ntninja/wordpress-postgresql:latestrestart: always
        ports:
            - 80:80
        environment:
            WORDPRESS_DB_HOST: postgres
            WORDPRESS_DB_USER: postgres
            WORDPRESS_DB_PASSWORD:
            postgresWORDPRESS_DB_NAME:
            postgres
        volumes:
            - ./wp:/var/www/html

    postgres:
        image: postgres:10.5
        restart: always
        environment:
            - POSTGRES_DB=postgres
            - POSTGRES_USER=postgres
            - POSTGRES_PASSWORD=postgr
        esports:
            - '5432:5432'
        volumes:
            - ./postgres-data:/var/lib/postgresql/data

volumes:
    db:
```