

# 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 conteneurs sont 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 temps et en espace différentes.

**Déf Docker :** Docker est une plate-forme logicielle qui vous permet de concevoir, tester et déployer des 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 pouvez facilement 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 :

#### Étape 1 : 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 :

```
root@debian-xfce:~# echo \"deb[arch=\"$(dpkg--print-architecture)signed-by=/etc
/apr/keyrings/docker.gpg]https://download.docker.com/linux/debian \"$(. /etc/
os-release && echo \"$VERSION_CODENAME\")\" stable\" | \ tee /etc/apr/sources.list
.d/docker.list > /dev/null
-bash: tee : commande introuvable
-bash: dpkg--print-architecture : commande introuvable
root@debian-xfce:~# echo \"deb [arch=\"$(dpkg --print-architecture)\" sig
ned-by=/etc/apr/keyrings/docker.gpg] https://download.docker.com/linux/debian
\"
\"$(. /etc/os-release && echo \"$VERSION_CODENAME\")\" stable\" | \
tee /etc/apr/sources.list.d/docker.list > /dev/null
apt-get update
Réception de :1 http://depot.stsio.lan/debian bookworm InRelease [151 kB]
Atteint :2 http://depot.stsio.lan/debian bookworm-updates InRelease
Atteint :3 http://depot.stsio.lan/security bookworm-security/updates InRelease
Atteint :4 https://download.docker.com/linux/debian bookworm InRelease
151 ko réceptionnés en 0s (376 ko/s)
Lecture des listes de paquets... Fait
N: Le dépôt « Debian bookworm » a modifié sa valeur « firmware component » de
« non-free » à « non-free-firmware »
N: Plus d'information disponible dans la note de mise à jour ici : https://www
.debian.org/releases/bookworm/amd64/release-notes/ch-information.html#non-free
-split
root@debian-xfce:~#
```

```
# 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 :

```
root@debian-xfce:~# apt-get install docker-ce docker-ce-cli containerd.io dock  
er-buildx-plugin docker-compose-plugin  
Lecture des listes de paquets... Fait  
Construction de l'arbre des dépendances... Fait  
Lecture des informations d'état... Fait  
docker-ce est déjà la version la plus récente (5:24.0.6-1~debian.12~bookworm).  
docker-ce-cli est déjà la version la plus récente (5:24.0.6-1~debian.12~bookwo  
rm).  
docker-buildx-plugin est déjà la version la plus récente (0.11.2-1~debian.12~b  
ookworm).  
docker-compose-plugin est déjà la version la plus récente (2.21.0-1~debian.12~  
bookworm).  
Les paquets suivants seront mis à jour :  
  containerd.io  
1 mis à jour, 0 nouvellement installés, 0 à enlever et 15 non mis à jour.  
Il est nécessaire de prendre 28,6 Mo dans les archives.  
Après cette opération, 560 ko d'espace disque supplémentaires seront utilisés.  
Souhaitez-vous continuer ? [0/n] o  
Réception de :1 https://download.docker.com/linux/debian bookworm/stable amd64  
  containerd.io amd64 1.6.24-1 [28,6 MB]  
28,6 Mo réceptionnés en 1s (40,0 Mo/s)  
Lecture des fichiers de modifications (« changelog »)... Terminé  
(Lecture de la base de données... 142432 fichiers et répertoires déjà installé  
s.)  
Préparation du dépaquetage de .../containerd.io_1.6.24-1_amd64.deb ...  
Dépaquetage de containerd.io (1.6.24-1) sur (1.6.22-1) ...  
Paramétrage de containerd.io (1.6.24-1) ...  
Traitement des actions différées (« triggers ») pour man-db (2.11.2-2) ...  
root@debian-xfce:~# █
```

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

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

Exemple en image :

```
root@debian-xfce:~# docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/

root@debian-xfce:~#
```

## **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
ab8d8938f431	hello-world	"/hello"	9 minutes ago	Exited (0) 9 minutes ago
b8229dc6d38e	f5a6b296b8a2	"/docker-entrypoint..."	13 days ago	Exited (0) 13 days ago
cc05e18a006b	hello-world	"/hello"	13 days ago	Exited (0) 13 days ago
ba6ea2af92ae	hello-world	"/hello"	13 days ago	Exited (0) 13 days ago
b6e1addb8b49	hello-world	"/hello"	13 days ago	Exited (0) 13 days ago

```
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 seconds

```
root@debian-xfce:~#
```

status: up

## executer un shell bash:

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



Exemple en image :

```
root@debian-xfce:~# docker exec -it nginx bash
root@e5bbec68fa64:/#
```

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 :

```
root@debian-xfce:~# docker stop nginx
nginx
root@debian-xfce:~#
```

## Lister les processus afin de lister les conteneurs :

```
# docker ps -a
```

Exemple en image :

```

root@debian-xfce:~# docker ps -a
CONTAINER ID   IMAGE          PORTS          COMMAND          CREATED          STATUS
              NAMES
e5bbec68fa64   nginx         (0) 25 seconds ago   "/docker-entrypoint..."   2 minutes ago   Exited
ab8d8938f431   hello-world   (0) 11 minutes ago   "/hello"          11 minutes ago   Exited
              magical_shannon
b8229dc6d38e   f5a6b296b8a2 (0) 13 days ago       "/docker-entrypoint..."   13 days ago     Exited
              some-nginx
cc05e18a006b   hello-world   (0) 13 days ago       "/hello"          13 days ago     Exited
              busy_goldwasser
ba6ea2af92ae   hello-world   (0) 13 days ago       "/hello"          13 days ago     Exited
              quizzical_faraday
b6e1addb8b49   hello-world   (0) 13 days ago       "/hello"          13 days ago     Exited
              hopeful_jennings
root@debian-xfce:~#

```

Status : exited

## Lancer un conteneur :

```
# docker start nom_du_conteneur
```

Exemple en image :

```

root@debian-xfce:~# docker start nginx
nginx
root@debian-xfce:~#

```

## 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 :

stopper le container puis exécuter 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 before
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      PORTS      COMMAND      CREATED      STATUS
                NAMES
ab8d8938f431   hello-world  (0) 14 minutes ago  magical_shannon  Exited
b8229dc6d38e   f5a6b296b8a2  (0) 13 days ago  some-nginx      Exited
cc05e18a006b   hello-world  (0) 13 days ago  busy_goldwasser  Exited
ba6ea2af92ae   hello-world  (0) 13 days ago  quizzical_faraday  Exited
b6e1addb8b49   hello-world  (0) 13 days ago  hopeful_jennings  Exited
root@debian-xfce:~#

```

Le container nginx n'apparaît 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@sha256:32da30332506740a2f7c34d5dc70467b7f14ec67d912703568daff790ab3f755
Deleted: sha256:61395b4c586da2b9b3b7ca903ea6a448e6783dfdd7f768ff2c1a0f3360aaba99
Deleted: sha256:1c69f36a0d9b59b762eaba410fa9fd01b85140670a8d49199a7b37702cc956c0
Deleted: sha256:bac209bfab6997cccf20779ae98d5f77a66867734499ecf604a50a5826f6b8a4
Deleted: sha256:859676f4cd3004af025a02dade096ad6f9391d94d1b1a983fc6098debe435055
Deleted: sha256:cbbd97cee0d824e5e82f9a4b2e93c5eb3c66fd72a2624d5c1521dc3395bfd1e2
Deleted: sha256:0b41545d8c3de3b78778d591a8da3d9dfa5fa8807baef5edf21e5eb94ded792d
Deleted: sha256:7e87866b23143eb30086086a669b2e902368a5836446a885b2411d3feef18bef
Deleted: sha256:d310e774110ab038b30c6a5f7b7f7dd527dbe527854496bd30194b9ee6ea496e
root@debian-xfce:~#

```

## Vérification de la suppression de l'image :

```
# docker images
```

Exemple en image :

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

L'image nginx n'apparaît 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:

```
GNU nano 7.2 Dockerfile
FROM debian:latest

USER root

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

CMD echo "Hello World!"

```

[ Lecture de 7 lignes ]

<b>^G</b> Aide	<b>^O</b> Écrire	<b>^W</b> Chercher	<b>^K</b> Couper	<b>^T</b> Exécuter
<b>^X</b> Quitter	<b>^R</b> Lire fich.	<b>^\</b> Remplacer	<b>^U</b> Coller	<b>^J</b> Justifier

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

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

Exemple en image :

```
Terminal - etul@debian-xfce: ~
Fichier  Édition  Affichage  Terminal  Onglets  Aide
root@debian-xfce:~/dockerd# docker build -t my-hello-world .
[+] Building 7.9s (6/6) FINISHED                                docker:default
=> [internal] load build definition from Dockerfile             0.0s
=> => transferring Dockerfile: 134B                             0.0s
=> [internal] load .dockerignore                               0.1s
=> => transferring context: 2B                                   0.0s
=> [internal] load metadata for docker.io/library/debian:latest 1.3s
=> [1/2] FROM docker.io/library/debian:latest@sha256:eaace54a93d7b69c7 2.8s
=> => resolve docker.io/library/debian:latest@sha256:eaace54a93d7b69c7 0.0s
=> => sha256:eaace54a93d7b69c7c52bb8ddf9b3fcb80c106a49 1.85kB / 1.85kB 0.0s
=> => sha256:8a6e23e1b192b30eff14036a92e9ecdb551a1a10aa853 529B / 529B 0.0s
=> => sha256:2657a4a0a6d5e8b3515004185275768f115a64a83 1.46kB / 1.46kB 0.0s
=> => sha256:167b8a53ca4504bc6aa3182e336fa96f4ef7687 49.56MB / 49.56MB 0.9s
=> => extracting sha256:167b8a53ca4504bc6aa3182e336fa96f4ef76875d158c1 1.7s
=> [2/2] RUN apt-get update && apt-get upgrade -y             3.5s
=> exporting to image                                           0.2s
=> => exporting layers                                          0.2s
=> => writing image sha256:2b93326fcc75aa24d6019c8de464c78ae7fc2b06d33 0.0s
=> => naming to docker.io/library/my-hello-world              0.0s
root@debian-xfce:~/dockerd#
```

## Créer une image et un conteneur nginx avec un html personnalisé :

-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 :

```
GNU nano 7.2                                index.html
<body>
<p>Hello World!</p>
</body>

I

[ Lecture de 5 lignes ]
^G Aide      ^O Écrire    ^W Chercher  ^K Couper    ^T Exécuter  ^C Emplacement
^X Quitter   ^R Lire fich.^_ Remplacer  ^U Coller    ^J Justifier ^/ Aller ligne
```

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

# ls -l

```
root@debian-xfce:~# ls -l
total 8
-rw-r--r-- 1 root root 96 27 sept. 13:55 Dockerfile
-rw-r--r-- 1 root root 37 27 sept. 16:23 index.html
root@debian-xfce:~#
```

-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 :



```
GNU nano 7.2 Dockerfile
FROM nginx

COPY ./index.html /usr/share/nginx/html/index.html

```

[ Lecture de 3 lignes ]

<b>^G</b> Aide	<b>^O</b> Écrire	<b>^W</b> Chercher	<b>^K</b> Couper	<b>^T</b> Exécuter	<b>^C</b> Emplacement
<b>^X</b> Quitter	<b>^R</b> Lire fich.	<b>^_</b> Remplacer	<b>^U</b> Coller	<b>^J</b> Justifier	<b>^/</b> Aller ligne

-créer l'image

**nom\_du\_dossier# docker build -t nom\_de\_l'image .**

Exemple en image :

```
Terminal - etu1@debian-xfce: ~
Fichier  Édition  Affichage  Terminal  Onglets  Aide

root@debian-xfce:~/dockerd# docker build -t my-nginx .
[+] Building 6.5s (7/7) FINISHED                                docker:default
=> [internal] load build definition from Dockerfile             0.0s
=> => transferring dockerfile: 100B                             0.0s
=> [internal] load .dockerignore                               0.0s
=> => transferring context: 2B                                   0.0s
=> [internal] load metadata for docker.io/library/nginx:latest 1.3s
=> [internal] load build context                               0.0s
=> => transferring context: 74B                                  0.0s
=> [1/2] FROM docker.io/library/nginx@sha256:32da30332506740a2f7c34d5d 4.3s
=> => resolve docker.io/library/nginx@sha256:32da30332506740a2f7c34d5d 0.1s
=> => sha256:32da30332506740a2f7c34d5dc70467b7f14ec67d 1.86kB / 1.86kB 0.0s
=> => sha256:b2888fc9cfe7cd9d6727aeb462d13c7c45dec413b 1.78kB / 1.78kB 0.0s
=> => sha256:61395b4c586da2b9b3b7ca903ea6a448e6783dfdd 8.15kB / 8.15kB 0.0s
=> => sha256:8b625c47d69711d95708566cd97b72bca565679 41.34MB / 41.34MB 1.7s
=> => sha256:4d3239651a63f0595b1c047313d6f5c63e1e69c834d31 627B / 627B 0.2s
=> => sha256:a803e7c4b030119420574a882a52b6431e160fc 29.12MB / 29.12MB 1.1s
=> => sha256:0f816efa513d909851c457ae41744fe3ff36ab19ebc2d 959B / 959B 0.4s
=> => sha256:01d159b8db2f24da97028c26bf6622e249e162bladab0 370B / 370B 0.6s
=> => sha256:5fb9a81470f3644c474192baf0827a34749286cb6 1.22kB / 1.22kB 0.8s
=> => sha256:9b1e1e7164db75ad0f64e8deeb33e771d455fa590 1.41kB / 1.41kB 1.1s
=> => extracting sha256:a803e7c4b030119420574a882a52b6431e160fceb7620f 1.4s
=> => extracting sha256:8b625c47d69711d95708566cd97b72bca565679d034ee0 1.3s
=> => extracting sha256:4d3239651a63f0595b1c047313d6f5c63e1e69c834d315 0.0s
=> => extracting sha256:0f816efa513d909851c457ae41744fe3ff36ab19ebc2d7 0.0s
=> => extracting sha256:01d159b8db2f24da97028c26bf6622e249e162bladab06 0.0s
=> => extracting sha256:5fb9a81470f3644c474192baf0827a34749286cb6d9330 0.0s
=> => extracting sha256:9b1e1e7164db75ad0f64e8deeb33e771d455fa590126b2 0.0s
=> [2/2] COPY ./index.html /usr/share/nginx/html/index.html 0.7s
=> exporting to image                                           0.1s
=> => exporting layers                                           0.0s
=> => writing image sha256:fbf4e218b76a1c88b7282b63f8173f2433e43cf8db8 0.0s
=> => naming to docker.io/library/my-nginx                       0.0s
root@debian-xfce:~/dockerd#
```

-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 port 8080 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 :

```
GNU nano 7.2                                compose.yml
version: '3'
services:
  web:
    image: nginx:latest
    ports:
      - "8080:80"
    volumes:
      - /usr/share/nginx/html

[ Lecture de 8 lignes ]
^G Aide      ^O Écrire    ^W Chercher  ^K Couper    ^T Exécuter
^X Quitter   ^R Lire fich.^_ Remplacer  ^U Coller    ^J Justifier
```

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

**Nom\_du\_dossier# docker compose up**

Exemple en image :

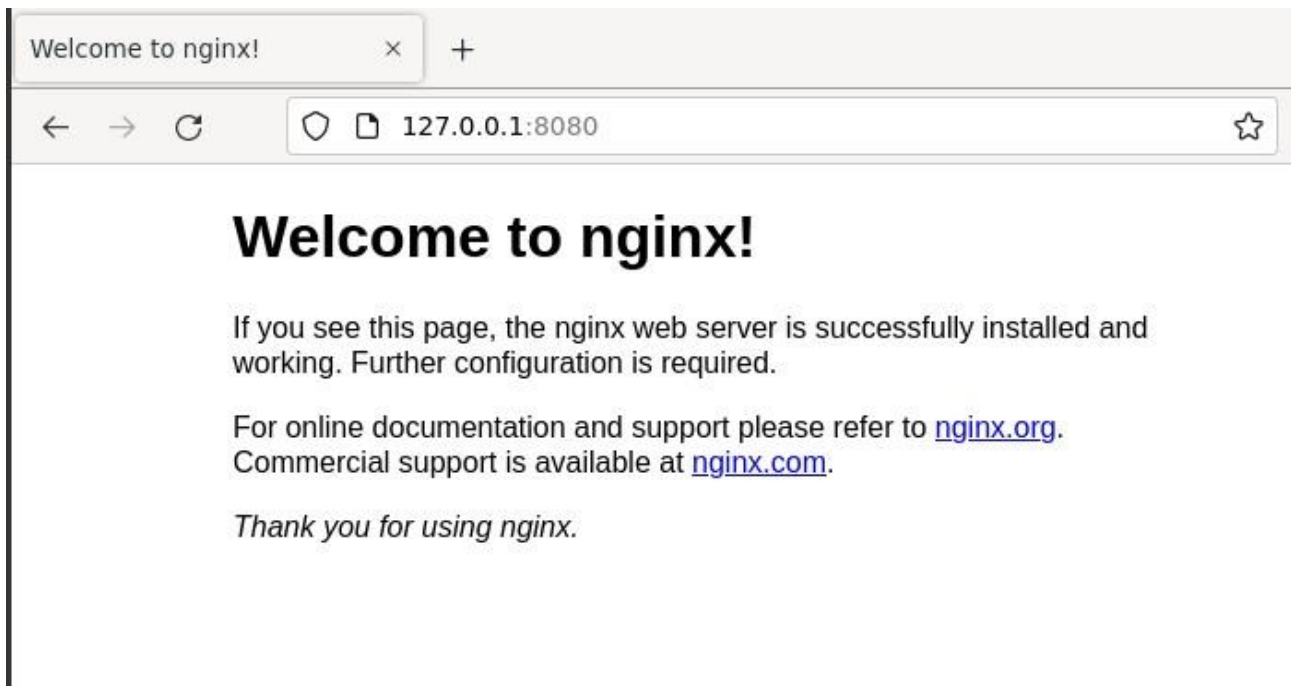
```

root@debian-xfce:~/dockerd# docker compose up
[+] Running 1/0
 ✓ Container dockerd-web-1 Created 0.0s
Attaching to dockerd-web-1
dockerd-web-1 | /docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, wi
ll attempt to perform configuration
dockerd-web-1 | /docker-entrypoint.sh: Looking for shell scripts in /docker-e
ntrypoint.d/
dockerd-web-1 | /docker-entrypoint.sh: Launching /docker-entrypoint.d/10-list
en-on-ipv6-by-default.sh
dockerd-web-1 | 10-listen-on-ipv6-by-default.sh: info: IPv6 listen already en
abled
dockerd-web-1 | /docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local
-resolvers.envsh
dockerd-web-1 | /docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envs
ubst-on-templates.sh
dockerd-web-1 | /docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune
-worker-processes.sh
dockerd-web-1 | /docker-entrypoint.sh: Configuration complete; ready for star
t up
dockerd-web-1 | 2023/10/02 11:34:11 [notice] 1#1: using the "epoll" event met
hod
dockerd-web-1 | 2023/10/02 11:34:11 [notice] 1#1: nginx/1.25.2
dockerd-web-1 | 2023/10/02 11:34:11 [notice] 1#1: built by gcc 12.2.0 (Debian
12.2.0-14)
dockerd-web-1 | 2023/10/02 11:34:11 [notice] 1#1: OS: Linux 6.1.0-11-amd64
dockerd-web-1 | 2023/10/02 11:34:11 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1
048576:1048576
dockerd-web-1 | 2023/10/02 11:34:11 [notice] 1#1: start worker processes
dockerd-web-1 | 2023/10/02 11:34:11 [notice] 1#1: start worker process 21
c

```

- ouvrir le navigateur

- taper l'url 127.0.0.1:8080



## **Stopper le conteneur lancer avec docker compose lancer 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 en arrière-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:latest
  restart: always
  ports:
    - 80:80
  environment:
    WORDPRESS_DB_HOST: postgres
    WORDPRESS_DB_USER: postgres
    WORDPRESS_DB_PASSWORD:
    postgres
    WORDPRESS_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
  ports:
    - '5432:5432'
  volumes:
    - ./postgres-data:/var/lib/postgresql/data
```

volumes:

db: