

Docker

Stefan Kummer | <https://www.skweb.ch>

pour le **développement local et le partage immédiat de configuration.**

Conteneur

```
ps                : liste conteneurs actifs
ps -a             : liste tous les conteneurs
rm <id> <id>      : supprimer un ou plusieurs
run <image>       : créer et lancer
run -it <image>   : permet de se connecter
                  i = interagir / t = text
run -it --rm <image> : supprime à la sortie
start <id>        : démarre un conteneur
start -ai <i>      : reste interactif
stop <id>         : stop un conteneur
run <image> sleep infinity : reste en vie
exec -it <id> bash : entrer dans un conteneur
                  mais ne l'arrête pas
exec -it <id> touch <f> : execute une commande
--name=<name>      : nommer le conteneur
```

Images

```
ls ou images      : liste les images
image rm <id> <id> : supprime un ou des
rmi <id> <id>      : idem
pull <image>      : télécharge l'image
```

Conteneurs connectés

- Communiquer par les adresses ip des conteneurs : ip -c a
- On peut aussi communiquer par les noms des conteneurs.

Réseaux personnalisés (cf dos)

```
network ls        : liste les réseaux
--network=<network> : connecte le conteneur à ce
                    Réseau / none = isoler les
                    Conteneurs
network create --driver=bridge <name>
                  : créé un réseau de type
                  bridge
network inspect    : informations réseau
```

Volumes

```
run ... -v <chemin_local>:<chemin_docker> <image>
                  : mapper un volume
volume create test : manager un volume
volume ls          : liste les volumes
run ... -v <volume>:<chemin_docker>
                  : manager un volume
volume rm <volume> : supprime un ou des
volume inspect     : infos sur le volume
```

Volume managé (Volume nommé) : méthode de stockage **entièrement gérée par Docker**, stockée dans une zone dédiée de l'hôte (souvent /var/lib/docker/volumes), et **recommandée pour la persistance des données en production** grâce à sa portabilité et sa sécurité.

Volume Mappé (Bind Mount) : montage qui lie **directement un chemin spécifique (un fichier ou un répertoire)** de la machine hôte à un chemin dans le conteneur, et est principalement utilisé

Ports

```
-p <port_local>:<port_docker>
      : mappe un port
```

Dockerfile

```
FROM <image>:<version>
      : base
RUN <command>      : exécute une commande
RUN apt update
RUN apt install -y <paquets>
CMD                : un seul par Dockerfile
                  Commande à exécuter au
                  démarrage
ADD <src> <dest>   : ajouter des fichiers (ext)
COPY <src> <dest>  : copier des fichiers (int)
ENV <key>=<value>   : variables
ENV [<key>=<value>,...]
EXPOSE <port>(<udp>) : expose un port
EXPOSE [...]
VOLUME </vol>       : créé un volume
VOLUME [...]
```

<https://docs.docker.com/reference/dockerfile/>

```
build -t <nom_image_a_créer> <dockerfile|. >
      : création de l'image
```

Docker compose

```
compose up        : construction
compose up -d     : mode détaché (daemon)
compose stop      : depuis le dossier docker-
                  compose.yml
compose rm        : idem
```

docker-compose.yml (ou compose.yml)

```
services:
      : début d'un conteneur basé
      sur une image
  <name>:
      : définit le nom service
    image:
      : image à utiliser
    container_name:
    stdin_open: true
    tty: true
      : ces 2 lignes permettent
      d'interagir avec le
      conteneur (il ne s'arrête
      pas)

volumes:
  - <local>:<docker>
      : exemple ./data:/distant
      le dossier distant est créé
      automatiquement
volumes:
  - test_volume:<docker>
      : pour les volumes managés
networks:
  - Test_network

volumes:
      : création d'un volume
  test_volume:
      : managé

networks:
  test_network
  driver: bridge|host|overlay|macvlan
```

Type de Réseau	Isolation	Multi-Hôte (Swarm)	Performance	Cas d'Utilisation Principal
Bridge (Pont)	Élevée (par défaut)	Non (limité à l'hôte)	Modérée (via NAT)	Conteneurs communicants sur un seul hôte (ex: Web + DB).
Host (Hôte)	Aucune	Non (limité à l'hôte)	Très Élevée (pas de NAT)	Applications sensibles à la latence ou sans souci de conflit de ports.
Overlay (Superposition)	Élevée (y compris chiffrement)	Oui (essentiel)	Modérée (implique encapsulation)	Clusters Docker Swarm et communication multi-hôtes distribuée.
Macvlan	Modérée (apparaît comme un hôte distinct)	Non (sauf configuration complexe)	Élevée (pas de NAT)	Conteneurs nécessitant leur propre IP physique sur le réseau local.
None (Aucun)	Totale (pas de réseau)	Non	N/A	Tâches batch isolées qui n'ont besoin d'aucune connectivité réseau.

Points Clés à Retenir :

- Le **Bridge** est le choix par défaut et le plus courant pour la plupart des applications simples sur un seul serveur.
- L'**Overlay** est indispensable dès que vous passez à un environnement orchestré ou distribué (plusieurs serveurs).
- Le mode **Host** est le plus rapide, mais le moins isolé.