



Concepts clés de Docker

Sommaire

- [Image](#)
- [Container](#)
- [Dockerfile](#)
- [Volumes](#)
- [Réseaux Docker](#)
- [Variables d'environnement](#)

Image

- 🛠 **But** : Définir un environnement logiciel complet (OS, dépendances, code...)
- 💡 **Explication** : Une image est un **modèle figé** que Docker peut exécuter.
- ☑ **Exemple** :

```
FROM python:3.11
```

Container

- 🛠 **But** : Exécuter l'application dans un environnement isolé
- 💡 **Explication** : Un container est une **image qui tourne**. Il est temporaire et autonome.
- ☑ **Exemple** :

```
docker run python:3.11 python --version
```

Dockerfile

- 🛠 **But** : Automatiser la création d'une image
- 💡 **Explication** : Le Dockerfile est un **script de construction** d'image.
- ☑ **Exemple** :

```
FROM python:3.11
COPY . /app
RUN pip install -r /app/requirements.txt
CMD ["python", "/app/main.py"]
```

Volumes

- 🛠 **But** : Conserver les données même si le container est supprimé
- 💡 **Explication** : Les volumes sont **liés à l'hôte** et permettent de sauvegarder ou partager des fichiers.
- ☑ **Exemple** :

```
docker run -v /data:/app/data my-image
```

Réseaux Docker

- 🛠 **But** : Permettre aux containers de se **parler entre eux**
- 💡 **Explication** : Docker crée des réseaux virtuels pour connecter les containers ensemble.
- ☑ **Exemple** :

```
services:
  api:
    networks: [backend]
  db:
    networks: [backend]
```

Variables d'environnement

- 🛠 **But** : Configurer dynamiquement une application
- 💡 **Explication** : Ces variables sont passées au container au lancement.
- ☑ **Exemple** :

```
environment:
  - ENV=production
```