

Les volumes

# Introduction

Les volumes constituent le mécanisme à privilégier :

- ✓ Ils servent à conserver les données générées et utilisées par les conteneurs Docker.

Bien que les montages liés dépendent de la structure des répertoires et du système d'exploitation de la machine hôte :

- les volumes qu'en à eux sont entièrement gérés par Docker.

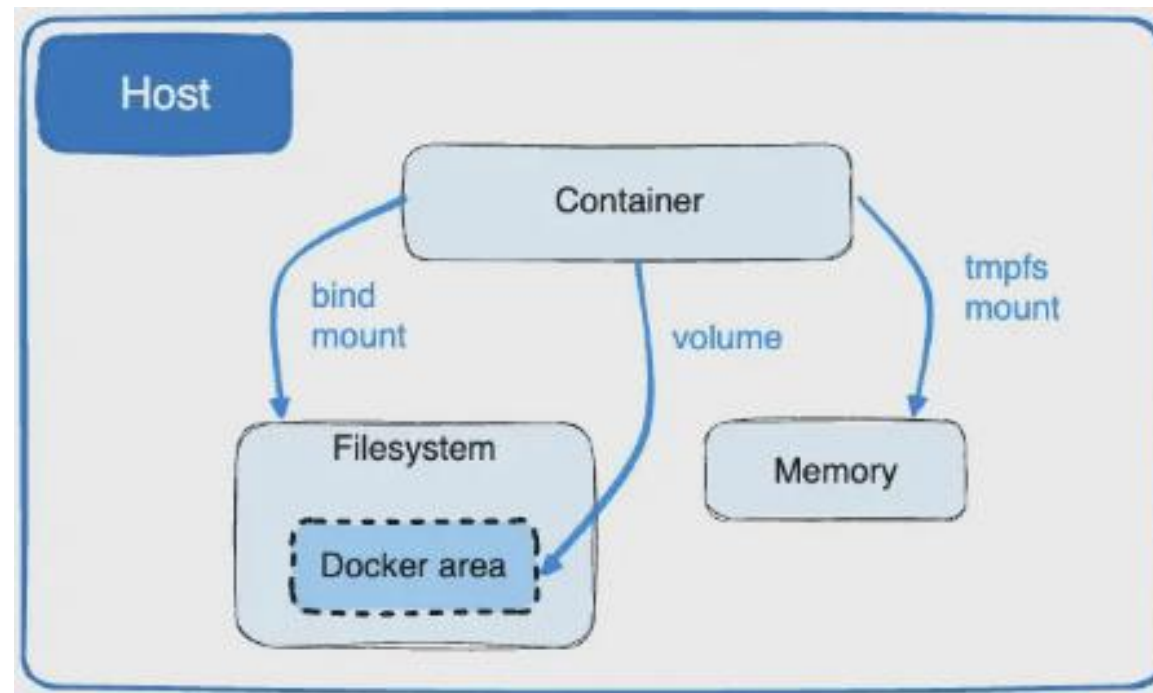
Source : <https://docs.docker.com/storage/volumes/>

Les volumes présentent plusieurs avantages par rapport aux montages liés :

- ✓ Les volumes sont plus faciles à sauvegarder ou à migrer que les montages liés.
- ✓ Vous pouvez gérer les volumes à l'aide des commandes Docker CLI ou de l'API Docker.
- ✓ Les volumes fonctionnent sur les conteneurs Linux et Windows.
- ✓ Les volumes peuvent être partagés de manière plus sûre entre plusieurs conteneurs.
- ✓ Les pilotes de volume vous permettent de stocker des volumes sur des hôtes distants ou des fournisseurs de cloud, de chiffrer le contenu des volumes ou d'ajouter d'autres fonctionnalités.
- ✓ Les nouveaux volumes peuvent avoir leur contenu prérempli par un conteneur.
- ✓ Les volumes sur Docker Desktop ont des performances bien supérieures à celles des montages liés à partir d'hôtes Mac et Windows.

De plus, les volumes constituent souvent un meilleur choix que la persistance des données dans la couche inscriptible d'un conteneur :

- ✓ car un volume n'augmente pas la taille des conteneurs qui l'utilisent
- ✓ et le contenu du volume existe en dehors du cycle de vie d'un conteneur donné.



Si votre conteneur génère des données d'état non persistantes :

- envisagez d'utiliser un montage tmpfs pour :
  - ✓ éviter de stocker les données n'importe où de manière permanente
  - ✓ et pour augmenter les performances du conteneur en évitant d'écrire dans la couche inscriptible du conteneur.

Les volumes utilisent la propagation de liaison privée (rprivate Bind Propagation) :

- Les changements n'affectent pas l'hôte.

De plus la propagation de liaison n'est pas configurable pour les volumes.

**Choisissez l'indicateur -v ou --mount**



En général, `--mount` est plus explicite et verbeux.

La plus grande différence est que :

- ✓ la syntaxe `-v` combine toutes les options dans un seul champ,
- ✓ tandis que la syntaxe `--mount` les sépare.

Si vous devez spécifier les options du pilote de volume, vous devez utiliser `--mount`.

-v ou --volume :

- ✓ Se compose de trois champs, séparés par des caractères deux-points (:).
- ✓ Les champs doivent être dans le bon ordre et la signification de chaque champ n'est pas immédiatement évidente.

Exemple :

```
docker run -d --name devtest -v myvol:/app nginx:latest
```

Dans le cas de volumes nommés :

```
docker run -d --name devtest -v myvol:/app nginx:latest
```

- ✓ le premier champ est le nom du volume et est unique sur une machine hôte donnée.

Pour les volumes anonymes, le premier champ est omis.

- ✓ Le deuxième champ est le chemin où le fichier ou le répertoire est monté dans le conteneur.
- ✓ Le troisième champ est facultatif et constitue une liste d'options séparées par des virgules, telles que ro.

--mount :

- ✓ se compose de plusieurs paires clé-valeur, séparées par des virgules
- ✓ et chacune constituée d'un tuple <key>=<value>.

La syntaxe --mount est plus détaillée que -v ou --volume, mais :

- ✓ l'ordre des clés n'est pas significatif
- ✓ et la valeur de l'indicateur est plus facile à comprendre.

--mount :

- ✓ Le type de montage peut être bind, volume ou tmpfs.
- ✓ La source de montage (source ou src).
  - Pour les volumes nommés, il s'agit du nom du volume.
  - Pour les volumes anonymes, ce champ est omis.
- ✓ La destination prend comme valeur le chemin où le fichier ou le répertoire est monté dans le conteneur.
  - Peut être spécifié comme destination, dst ou target.
- ✓ L'option en lecture seule, si elle est présente, entraîne le montage du montage lié dans le conteneur en lecture seule.
  - Peut être spécifié en readonly ou ro.

--mount -v comparaison :

✓ -v

```
docker run -d --name devtest -v myvol:/app nginx:latest
```

✓ --mount

```
docker run -d --name devtest --mount source=myvol,target=/app nginx:latest
```

C'est 2 instructions amènent au même résultat.

# **Créer et gérer des volumes**

Contrairement à un montage lié, vous pouvez créer et gérer des volumes en dehors de la portée de n'importe quel conteneur.

**Créer un volume :**

```
docker volume create monvolume
```



Lister les volumes :

```
docker volume ls
```

Résultat :

```
DRIVER      VOLUME NAME
local       fifo-ipc-exemple
local       monvolume
local       my-vol
local       myvol
```

Inspecter un volume :

```
docker volume inspect monvolume
```

Résultat :

```
[
  {
    "CreatedAt": "2023-12-06T15:55:00Z",
    "Driver": "local",
    "Labels": null,
    "Mountpoint": "/var/lib/docker/volumes/monvolume/_data",
    "Name": "monvolume",
    "Options": null,
    "Scope": "local"
  }
]
```

Inspecter un volume :

```
docker volume rm monvolume
```

Résultat :

DRIVER	VOLUME NAME
local	fifo-ipc-exemple
local	my-vol
local	myvol

**Démarrer un conteneur avec un volume**

Si vous démarrez un conteneur avec un volume qui n'existe pas encore :

- Docker crée le volume pour vous.

```
C:\Users\conta>docker volume ls
DRIVER      VOLUME NAME
local      myvol
```


C:\Users\conta>docker run -d --name devtest --mount source=monvolume,target=/app nginx:latest  
e6351cfc90360d80a303db2bb0b023154847b33afc90b9c8544089305b414fed

```
C:\Users\conta>docker volume ls
DRIVER      VOLUME NAME
local      monvolume
local      myvol
```

A terminal window with a black background and white text. It shows three commands and their outputs. The first command is 'docker volume ls', which outputs a table with two columns: 'DRIVER' and 'VOLUME NAME'. The output shows 'local' as the driver and 'myvol' as the volume name. The 'myvol' text is highlighted with a red box, and a red arrow points to it from the right. The second command is 'docker run -d --name devtest --mount source=monvolume,target=/app nginx:latest', which outputs a long alphanumeric string. The third command is another 'docker volume ls', which outputs a table with two columns: 'DRIVER' and 'VOLUME NAME'. The output shows two entries: 'local' with 'monvolume' and 'local' with 'myvol'. The 'monvolume' text is highlighted with a red box, and a red arrow points to it from the right.


Exemple avec le flag -v :

```
C:\Users\conta>docker volume ls
DRIVER      VOLUME NAME
local      myvol
```



```
C:\Users\conta>docker run -d --name devtest -v monvolume:/app nginx:latest
0fe0b08208f9b7c50064176366b7996658f9b460e4fad6a97598c0c7b0b383ee
```

```
C:\Users\conta>docker volume ls
DRIVER      VOLUME NAME
local      monvolume
local      myvol
```



Vérifions en inspectant notre volume :

```
docker inspect monvolume
```

Résultat :

```
[
  {
    "CreatedAt": "2023-12-06T16:58:34Z",
    "Driver": "local",
    "Labels": null,
    "Mountpoint": "/var/lib/docker/volumes/monvolume/_data",
    "Name": "monvolume",
    "Options": null,
    "Scope": "local"
  }
]
```

## Nettoyage :

```
C:\Users\conta>docker ps -a 1
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS      NAMES
0fe0b08208f9   nginx:latest  "/docker-entrypoint..." 10 minutes ago Up 10 minutes 80/tcp     devtest

C:\Users\conta>docker stop devtest 2
devtest

C:\Users\conta>docker rm devtest 3
devtest

C:\Users\conta>docker volume ls 4
DRIVER      VOLUME NAME
local      monvolume
local      myvol

C:\Users\conta>docker volume rm monvolume 5
monvolume

C:\Users\conta>docker volume ls 6
DRIVER      VOLUME NAME
local      myvol
```



# Exercise

1. Créer un volume monsite
2. Créer et lancer 3 conteneurs qui :
  - ✓ montent notre volume sur le répertoire `‘/usr/share/nginx/html’`
  - ✓ Soient accessibles sur un navigateur
3. Modifier le fichier `‘index.html’` dans le volume
4. Vérifier le résultat dans le navigateur

**Correction**

# 1. Créer un volume monsite

```
C:\Users\conta>docker volume create monsite  
monsite
```

```
C:\Users\conta>docker volume ls  
DRIVER      VOLUME NAME  
local      monsite  
local      myvol
```

## 2. Créer et lancer 3 conteneurs qui :

- ✓ Montent notre volume sur le répertoire '/usr/share/nginx/html'
- ✓ Soient accessibles sur un navigateur

```
C:\Users\conta>docker run -d -p 8080:80 --name monsiteweb1 -v monsite:/usr/share/nginx/html nginx:latest 1
fa1b49378cd5420bd399a965063ec5f2e02c009ef6944eedc441574b883f34ee

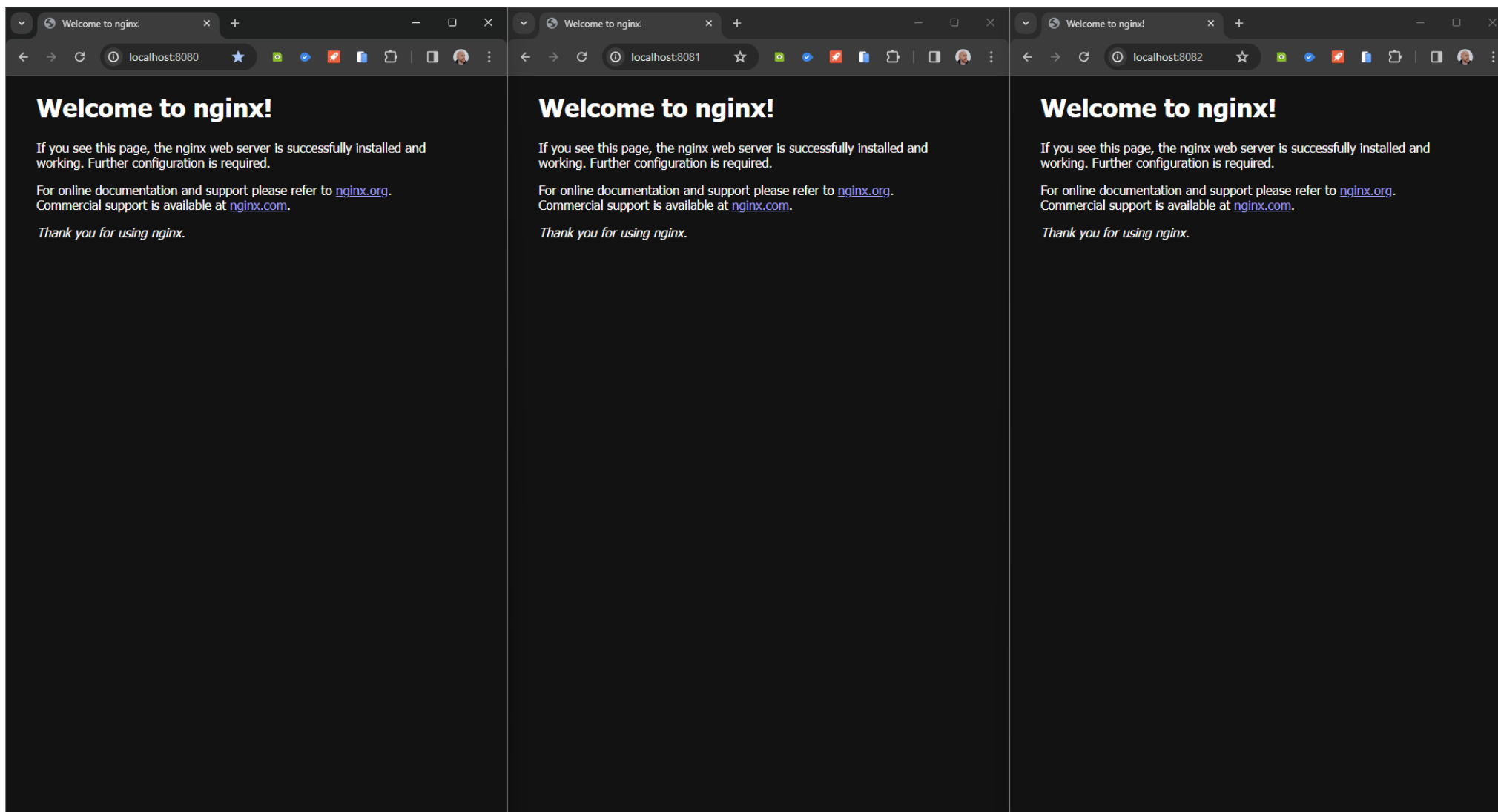
C:\Users\conta>docker run -d -p 8081:80 --name monsiteweb2 -v monsite:/usr/share/nginx/html nginx:latest 2
7ac6791f7b3f53ecda09199337f49d6ec1606f02467c7c54ba69fcc3e4a6b4bf

C:\Users\conta>docker run -d -p 8083:80 --name monsiteweb3 -v monsite:/usr/share/nginx/html nginx:latest 3
75518af061656c59721cb1b114b278d7d64e1fc029947faac77afbe8fa10111d

C:\Users\conta>docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
75518af06165	nginx:latest	"/docker-entrypoint..."	21 seconds ago	Up 20 seconds	0.0.0.0:8083->80/tcp	monsiteweb3
7ac6791f7b3f	nginx:latest	"/docker-entrypoint..."	41 seconds ago	Up 40 seconds	0.0.0.0:8081->80/tcp	monsiteweb2
fa1b49378cd5	nginx:latest	"/docker-entrypoint..."	3 minutes ago	Up 3 minutes	0.0.0.0:8080->80/tcp	monsiteweb1

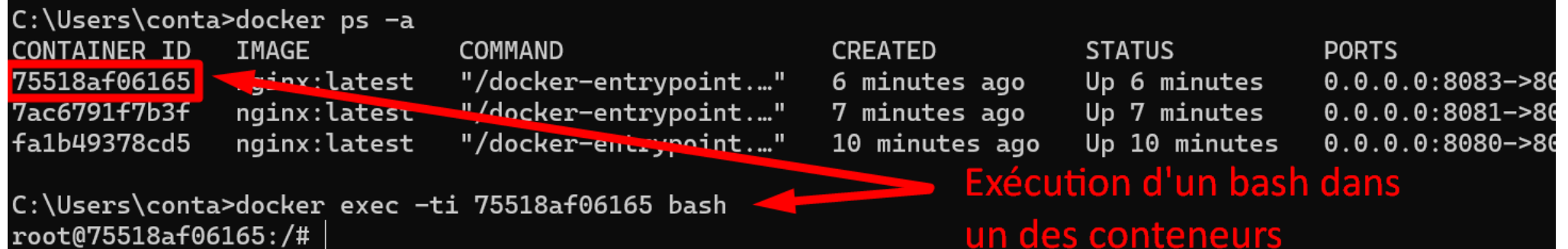
## 2. Résultat :



### 3. Modifier le fichier 'index.html' dans le volume

```
C:\Users\conta>docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS
75518af06165   nginx:latest   "/docker-entrypoint...." 6 minutes ago  Up 6 minutes  0.0.0.0:8083->80
7ac6791f7b3f   nginx:latest   "/docker-entrypoint...." 7 minutes ago  Up 7 minutes  0.0.0.0:8081->80
fa1b49378cd5   nginx:latest   "/docker-entrypoint...." 10 minutes ago Up 10 minutes  0.0.0.0:8080->80

C:\Users\conta>docker exec -ti 75518af06165 bash
root@75518af06165:/# |
```



Exécution d'un bash dans un des conteneurs

### 3. Installer un éditeur de texte

```
C:\Users\conta>docker exec -ti 75518af06165 bash
root@75518af06165:/# apt update
Get:1 http://deb.debian.org/debian bookworm InRelease [151 kB]
Get:2 http://deb.debian.org/debian bookworm-updates InRelease [52.1 kB]
Get:3 http://deb.debian.org/debian-security bookworm-security InRelease [48.0 kB]
Get:4 http://deb.debian.org/debian bookworm/main amd64 Packages [8780 kB]
Get:5 http://deb.debian.org/debian bookworm-updates/main amd64 Packages [6668 B]
Get:6 http://deb.debian.org/debian-security bookworm-security/main amd64 Packages [106 kB]
Fetched 9144 kB in 3s (3290 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
2 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@75518af06165:/# |
```

*Mise à jour des dépôts...*



### 3. Installer un éditeur de texte

```
2 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@75518af06165:/# apt-get install nano 1
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libgpm2 libncursesw6
Suggested packages:
  gpm hunspell
The following NEW packages will be installed:
  libgpm2 libncursesw6 nano
0 upgraded, 3 newly installed, 0 to remove and 2 not upgraded.
Need to get 837 kB of archives.
After this operation, 3339 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y 2
Get:1 http://deb.debian.org/debian bookworm/main amd64 libncursesw6 amd64 6.4-4 [134 kB]
Get:2 http://deb.debian.org/debian bookworm/main amd64 nano amd64 7.2-1 [689 kB]
Get:3 http://deb.debian.org/debian bookworm/main amd64 libgpm2 amd64 1.20.7-10+b1 [14.2 kB]
Fetched 837 kB in 1s (1600 kB/s)
debconf: delaying package configuration, since apt-utils is not installed
```

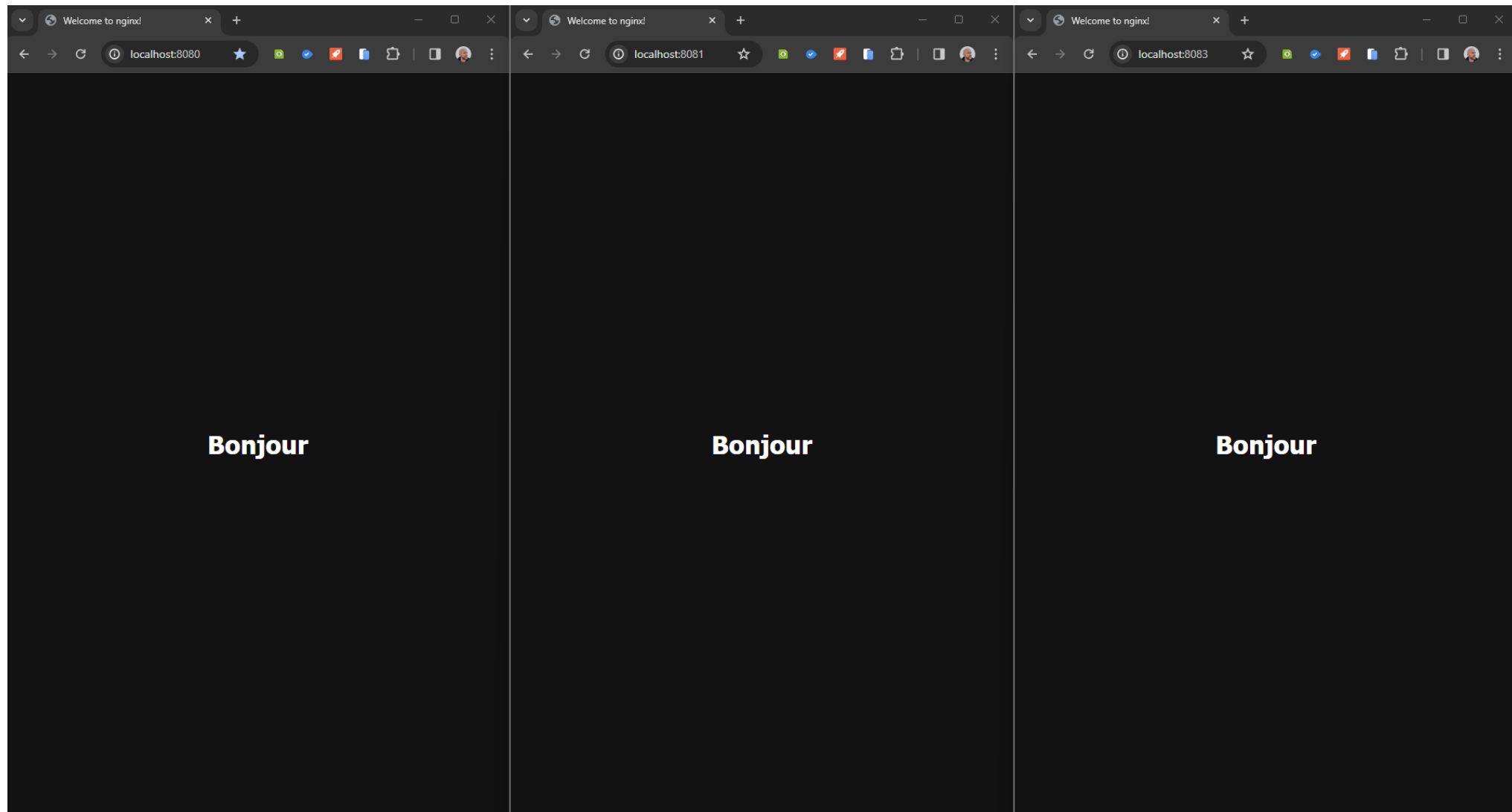
*Installation de l'éditeur ...*

### 3. Modification du fichier

```
Processing triggers for libc-bin (2.36-9+deb12u3) ...  
root@75518af06165:/# nano /usr/share/nginx/html/index.html  
root@75518af06165:/#
```

```
GNU nano 7.2 /usr/share/nginx/html/index.html *  
<!DOCTYPE html>  
<html>  
<head>  
<title>Welcome to nginx!</title>  
<style>  
html { color-scheme: light dark; }  
body { display: grid; min-height:100vh; margin: 0 auto;  
font-family: Tahoma, Verdana, Arial, sans-serif; }  
h1 { place-self: center; }  
</style>  
</head>  
<body>  
<h1>Bonjour</h1>  
</body>  
</html>
```

### 3. Vérifier le résultat dans le navigateur



**Questions ??**

Prochain chapitre: Les points de montage.