1. Activité : publier le service Web avec Docker

Pour rendre un service accessible, le conteneur doit être lancé en arrière plan.

Docker utilise la technique classique du mappage de port :

- un **port de l'hôte** va être **redirigé** vers un **port du container**.

Docker dispose pour cela d'une **interface réseau** sur la machine hôte.

1.1. Sous Linux

```
ip a
…
docker0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:f6:de:8b:5b brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
...
```

1.2. Sous Windows

```
PS > ipconfig /all
...
Carte Ethernet vEthernet (WSL) :

   Suffixe DNS propre à la connexion. . . :
   Description. . . . . . . . . . . . . . : Hyper-V Virtual Ethernet Adapter
   Adresse physique . . . . . . . . . . . : 00-15-5D-61-F5-91
   DHCP activé. . . . . . . . . . . . . . : Non
   Configuration automatique activée. . . : Oui
   Adresse IPv6 de liaison locale. . . . .: fe80::a2f2:e355:be04:5818%80(préféré)
   Adresse IPv4. . . . . . . . . . . . . .: 172.30.16.1(préféré)
   Masque de sous-réseau. . . . . . . . . : 255.255.240.0
   Passerelle par défaut. . . . . . . . . :
   IAID DHCPv6 . . . . . . . . . . . : 1342182749
   DUID de client DHCPv6. . . . . . . . : 00-01-00-01-2A-76-DE-73-80-FA-5B-3F-40-3C
   Serveurs DNS. . .  . . . . . . . . . . : fec0:0:0:ffff::1%1
                                    fec0:0:0:ffff::2%1
                                    fec0:0:0:ffff::3%1
   NetBIOS sur Tcpip. . . . . . . . . . . : Activé
...
```

Une commande possible est la suivante :

```
docker run -d -p <IP:port-hôte:port-conteneur> --name <nom conteneur> <image> COMMANDE
```

Le paramètre permettant le mappage est **-p <IP:port-hôte:port-container>** :

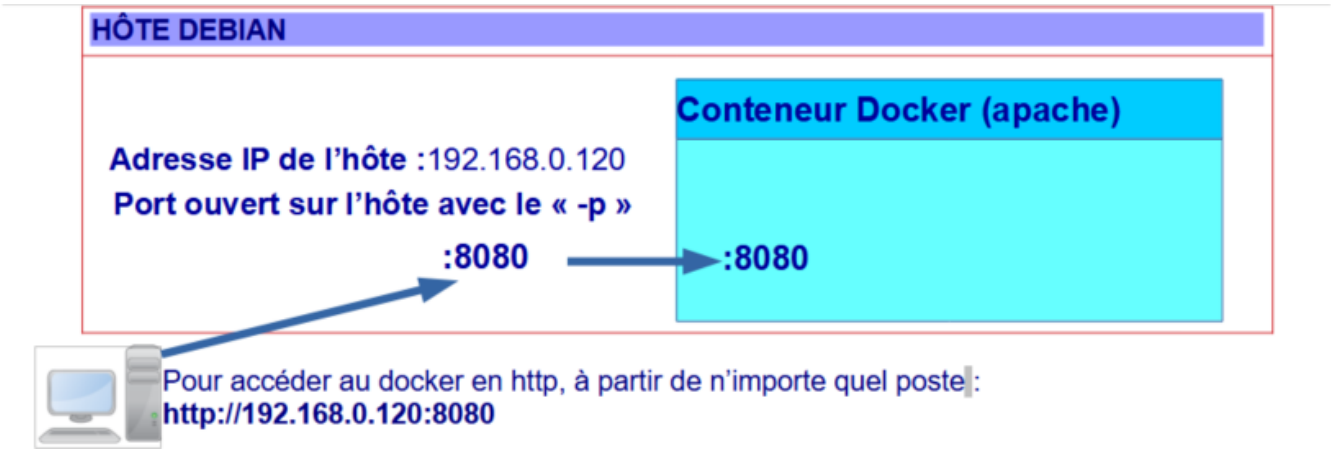
- si l'adresse IP réelle de l'hôte n'est pas indiquée, le système n'écoutera que sur localhost ;
- si le port de l'hôte à mapper n'est pas indiqué, Docker en choisira un automatiquement.

Le principe est ensuite de lancer le conteneur sera lancé en **arrière plan** (sans laisser de console ouverte) en mode **détaché** : c'est l'option **-d**» qui permet cela.

1.2.1. Création d'un conteneur Apache

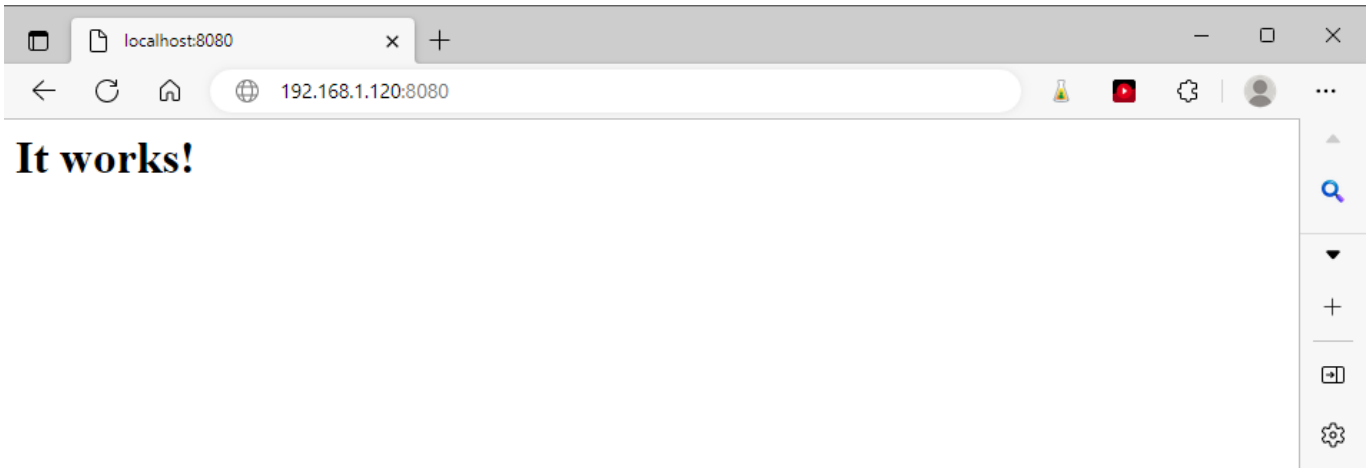
- **image** : ubuntu/apache2
- **ports** :
    - 80 → http

```
> docker run  -d -p 8080:80 --name servweb ubuntu/apache2
bd68b3b35b01aacb11148ad9073c538a6cf59e5bc1dbfc9fe03b2f8212cbd963
```



Pour accéder au site par défaut en http à partir de n'importe quel poste :

- http://192.168.0.120:8080



Nous pouvons constater via la commande **docker ps** que l'attribut PORT est maintenant rempli :

```
PS C:\docker ps
CONTAINER ID   IMAGE            COMMAND                  CREATED          STATUS          PORTS
NAMES
f8d3548ef747   ubuntu/apache2   "apache2-foreground"   24 seconds ago   Up 23 seconds
0.0.0.0:8080->80/tcp   servweb

…
```

L'option **-p** peut être multiplié autant de fois que nécessaire (si par exemple il y a plusieurs services à exposer dans le conteneur).

- utiliser la commande Powershell

```
Get-NetTCPConnection
```

1.3. La persistance des données

- Les **images** utlisée par les instances de conteneur sont en **lecture seule**.
- **Toutes les modifications** sur le conteneur sont réalisées **dans une couche supplémentaire**.
- Les **données ajoutées** dans un conteneur **disparaissent avec lui lors de sa destruction**.

Pour **sauvegarder** les modifications dans une autre image, il est nécessaire d'utiliser la commande **commit**.

Cela est pertinent pour construire une nouvelle image personnalisée mais pas pour utiliser le conteneur pour un service en ligne où les données sont modifiées sans cesse ou bien s'il est important de conserver certaines données comme les traces (logs).

Pour un **site Web dynamique** il est nécessite de disposer :

- d'un **espace de stockage** des **fichiers Web du site** ;
- d'un **espace de stockage** pour la **base de données** ;
- d'un **espace de stockage** pour les logs.

Le mécanisme le plus couramment utilisé avec Docker consiste à utiliser un **stockage externe au conteneur** et donc sur l'hôte ou sur n'importe quel autre support (baies SAN ou NAS) appelé **volume de données**.

1.4. Volume de données pour les pages Web

Le paramètre **-v (volume)** de la commande Docker permet :

- de définir un **mappage** entre un **dossier local** du PC et un **dossier du conteneur** ;
- et d'avoir un **stockage qui persistera** même lorsque le conteneur sera supprimé.

De cette manière, une arborescence de fichiers du PC sera montée à l'intérieur du container permettant la publication du site Web utilisant le **serveur Web Apache** et le **moteur de script php**.

- **image utilisée** : php:8.1-apache
- **dossier du serveur Apache** : /var/www/html
- **dossier local sur le PC** : par exemple C:\www\html
- Commande sous **Windows**. La commande à utiliser sera la suivante si les fichiers du site Web sont situés dans le dossier local **c:\www\html\siteweb** et les logs du serveur Apache externalisés dans le dossier c:\www\log\ :

```
docker run --name servweb -d -v c:\www\html\siteweb:/var/www/html -v c:\www\log:/var/log/apache2 -p
8001:80  php:8.1-apache
```

- Commande sous **Linux**

La commande à utiliser sera la suivante si les fichiers du site Web sont situés dans le dossier local **/home/user/www/html/siteweb** et les logs du serveur Apache externalisés dans le dossier **/home/user/www/log** :

```
docker run --name servweb -d -v /home/user/www/html/siteweb:/var/www/html -v
/home/user/www/log:/var/log/apache2 -p 8001:80  php:8.1-apache
```

Le site Web est accessible à l'URL : http://localhost:8001

Toute page créée ou modifiée dans le dossier C:\www\html\siteweb est publiée dans le site Web.

1.5. Activité à faire

Mission 1 :

- Créez un conteneur Docker à partir de l'image **httpd** permettant d'accéder à la page Web par défaut du serveur Web en configurant la persistance du site en local. Le dossier de publication du serveur httpd est **/usr/local/apache2/htdocs/**
- Modifiez la page Web par défaut pour afficher vos nom et prénom ainsi qu'un lien vers une 2e page html.
- Créez cette 2e page présentant le playgroud de Docker.
- Testez l'accès à vos deux pages : page d'accueil et page de présentation du playground Docker.

Mission 2 : Vous devez **externaliser les logs du serveur Apache** afin de pouvoir les consulter en cas de besoin et d'erreur de fonctionnement du site Web **Geststage** et du serveur.

- Créez un conteneur Docker à partir de l'image **httpd** affichant la page Web par défaut et en configurant la persistance des logs en local en créeant un volume local pour le dossier **/usr/local/apache2/logs/** du conteneur.
- **Vérifiez la présence** des fichiers de logs dans le **dossier local** (access.log ; error.log ; other_vhosts_access.log)

1.6. Créer une image personnalisée

L'image **php:8.1-apache** contient une **configuration de php** qui ne possède de pilote que pour gérer les accès à la base de données **sqlite**.

Pour pouvoir disposer d'une image contenant le **serveur Web Apache** mais aussi **php** configuré pour communiquer avec la base de données **MariaDB**, vous allez créer une **nouvelle image** à partir de l'image **php:8.1-apache** qui contiendra les pilotes nécessaires.

- Créer un ficher appelé **Dockerfile** (sans extension) ;
- Editez ce fichier pour y mettre les instructions suivantes :

Dockerfile

```
FROM php:8.1-apache
RUN apt-get update && apt-get upgrade -y
```

```
RUN docker-php-ext-install mysqli && docker-php-ext-enable mysqli
RUN docker-php-ext-install pdo pdo_mysql
```

- Depuis l'invite de commande **Powershell**, dans le dossier contenant le **fichier Dokerfile**, lancez la commande suivante pour créer :
    - une image **appelée php** et **préfixée** par votre pseudo Docker :

```
docker build -t charles/php-mariadb .
```

1.7. Mémento Docker

2. Retour Accueil Docker

- Docker

From:
https://siocours.lycees.nouvelle-aquitaine.pro/ - **Les cours du BTS SIO**

Permanent link:
https://siocours.lycees.nouvelle-aquitaine.pro/doku.php/reseau/docker/webdocker

Last update: **2026/06/21 09:56**