

Activité : découvrir Docker

Pour cette découverte de Docker, vous pouvez réaliser au préalable :

- avoir **installé** Docker.
- utiliser l'environnement de découverte appelé **Play with Docker (PWD)** : <https://labs.play-with-docker.com/>

Un premier conteneur

Lancement sans option d'un conteneur

Cette commande lance le classique Hello world ! :

```
btssio@ubuntudocker:~$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
1b930d010525: Pull complete
Digest: sha256:92695bc579f31df7a63da6922075d0666e565ceccad16b59c3374d2cf4e8e50e
Status: Downloaded newer image for hello-world:latest
```

```
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/>

```
btssio@ubuntudocker:~$
```

Ce qu'il faut retenir de l'exécution de cette commande :

- La première ligne indique que l'image du conteneur **'hello-world:latest'** n'est pas disponible **localement** c'est-à-dire sur le serveur Ubuntu. De plus, comme aucune version du conteneur n'est précisée, c'est la dernière version disponible, la **latest** qui est recherchée.
- L'image recherchée est alors **tirée (pull)** de Docker Hub depuis Internet.
- Une explication des étapes réalisées est indiquée : le client Docker, la session de terminal de votre serveur Ubuntu, contacte le **daemon** Docker du serveur Ubuntu (le service Docker) qui tire l'image du conteneur depuis Docker Hub, **crée un conteneur, exécute** ce qui est prévu dans l'image du conteneur (la production du texte à afficher) et envoie cette sortie texte vers le client Docker votre terminal.
- Le conteneur est ensuite **arrêté automatiquement** car le traitement prévu s'est effectué, l'affichage du texte et rien d'autre.

Afficher les images Docker présentes sur la machine

```
btssio@ubuntudocker:~$ docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
hello-world   latest   fce289e99eb9   4 months ago   1.84kB
btssio@ubuntudocker:~$
```

Vous pouvez vérifier la présence de l'image qui a été tirée (pull) de Docker Hub et stockée sur votre serveur. Docker gère un **cache** des images sur la machine. Lors du prochain démarrage d'une instance de conteneur sur la même image il n'y aura pas de nouveau téléchargement sauf si elle a été modifiée entre-temps.

INFORMATION

Pour en savoir plus sur une image il suffit de faire une recherche sur Docker Hub. Pour le conteneur hello-world : https://hub.docker.com/_/hello-world

Explication des tags

- Le **TAG** associé à l'image ubuntu est latest et est le marqueur de version de l'image.
- **IMAGE ID** est l'identifiant unique de l'image.
- **CREATED** est la date de création de l'image publiée sur le site Docker Hub.
- **SIZE** est sa taille virtuelle, c'est-à-dire de la couche logicielle téléchargée.

Lister les conteneurs actifs

L'image du conteneur hello-world est présente sur votre serveur mais après le lancement d'un conteneur basé sur cette image, vous avez retrouvé l'invite de commandes de votre serveur. Le conteneur lancé s'est ensuite arrêté automatiquement. Mais existe-il toujours ?

Pour constater que le conteneur lancé est bien arrêté vous pouvez visualiser les conteneurs actifs :

```
btssio@ubuntudocker:~$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
btssio@ubuntudocker:~$
```

Le conteneur hello-world n'est effectivement **plus actif**.

Lister tous les conteneurs y compris ceux qui sont inactifs

Relancez le conteneur hello-world. Vous pouvez à nouveau vérifier que celui n'est pas actif après l'affiche du texte prévu.

Visualisez maintenant tous les conteneurs créés, qu'ils soient actifs ou non :

```
btssio@ubuntudocker:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
9bbd425c86f4   hello-world "/hello" 2 seconds Exited (0) 1 s ago   vibrant_goldberg
ae50ecb3a66d   hello-world "/hello" 49 seconds Exited (0) 47 s ago   quirky_elbakyan
btssio@ubuntudocker:~$
```

Les 2 conteneurs basés sur la même image hello-world, sont bien présents mais arrêtés sans erreur normale (code de sortie 0) comme le montre le champ STATUS : Exited (0). Le premier conteneur s'est arrêté il y a 1 seconde, le 2ème il y a 47 secondes. Chaque conteneur : * possède un identifiant de conteneur CONTAINER ID, * et un nom, NAMES, qui a été généré par le démon Docker. La colonne COMMAND nous renseigne sur le processus qui avait été lancé, à savoir un script hello. ===== Comprendre le système de couche logicielle des images Docker ===== ===== Récupérer une image dans sa dernière version ===== <code shell> btssio@ubuntudocker:~\$ docker pull ubuntu Using default tag: latest latest: Pulling from library/ubuntu f476d66f5408: Pull complete 8882c27f669e: Pull complete d9af21273955: Pull complete f5029279ec12: Pull complete Digest: sha256:d26d529daa4d8567167181d9d569f2a85da3c5ecaf539cace2c6223355d69981 Status: Downloaded newer image for ubuntu:latest btssio@ubuntudocker:~\$ </code> Le TAG est latest par défaut si vous ne précisez pas la version du conteneur voulu. Cette image ubuntu est une image officielle ; elle n'est pas préfixée et vous pouvez avoir des informations sur Docker hub. Lien : <https://hub.docker.com/>



ubuntu ☆

Docker Official Images

Ubuntu is a Debian-based Linux operating system based on free software.

10M+

- Container
- Linux
- x86-64
- ARM
- ARM 64
- 386
- PowerPC 64 LE
- IBM Z
- Base Images
- Operating Systems
- Official Image

Linux - x86-64 (latest)

Copy and paste to pull this image

docker pull ubuntu

View Available Tags

Cette image est basée sur quatre couches logicielles : `shell` ... f476d66f5408: Pull complete 8882c27f669e: Pull complete d9af21273955: Pull complete f5029279ec12: Pull complete ...

INFORMATION

Le chargement de ce conteneur ubuntu et des couches logicielles nécessaires a lieu une fois pour toutes. Tous les lancements de conteneur suivant se baseront sur cette image, y compris - et c'est une des grands avantages de l'architecture par couches - toutes les images qui se trouvent sur Docker Hub et qui ont été construites à partir de cette image ubuntu. Et il existe sur Docker Hub de nombreuses images basées sur Ubuntu qui est une distribution de base fréquemment utilisée pour les images Docker.

Lancez un conteneur Ubuntu. Comme précédemment, le conteneur est ensuite arrêté mais existe bien. La colonne COMMAND indique que le processus qui a été lancé est cette fois-ci un shell Bash. `btssio@ubuntudocker:~$ docker run ubuntu btssio@ubuntudocker:~$ docker ps -a CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES b5ec671c6d11 ubuntu "/bin/bash" 21 s ago Exited ... laughing_ride ...` Vous pouvez prendre connaissance de la configuration du conteneur sur Docker Hub en consultant son fichier Dockerfile.

IMPORTANT

Le fichier Dockerfile associé à une image Docker, décrit exactement comment l'image a été construite. Quand vous recherchez sur Docker Hub une image qui doit répondre à votre besoin et qui a été publiée par un membre de la communauté, ce fichier vous permet de savoir ce que contient l'image afin de vous permettre de faire un choix éclairé parmi la profusion d'images possibles. Une bonne pratique consiste donc :

- à choisir une image officielle si celle-ci répond à votre besoin,
- et, si cela n'est pas le cas, à choisir une image de la communauté en regardant si elle est souvent utilisée (sa popularité), sa documentation mais aussi son fichier Dockerfile pour savoir exactement ce qu'elle contient.

Sur Docker Hub plusieurs Dockerfile sont indiqués pour l'image officielle ubuntu :

Supported tags and respective Dockerfile links

- 18.04 , bionic-20210827 , bionic
- 20.04 , focal-20210827 , focal , latest
- 21.04 , hirsute-20210825 , hirsute , rolling
- 21.10 , impish-20210827 , impish , devel
- 14.04 , trusty-20191217 , trusty
- 16.04 , xenial-20210804 , xenial


```
d=00;36:.midi=00;36:.mka=00;36:.mp3=00;36:.mpc=00;36:.ogg=00;36:.ra=00;36:.wav=00;36:.oga=00;36:.opus=00;36:.spx=00;36:*.xsp
f=00;36: TERM=xterm SHLVL=1 PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin =/usr/bin/env </code> * Création d'une
variable d'environnement qui n'existe pas sur l'hôte pour spécifier au conteneur l'adresse IP 192.168.1.200 d'un service Web nécessaire
pour la réalisation de sa tâche <code shell> $ docker run -e=IPAPIWEB=192.168.1.200 ubuntu env
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin HOSTNAME=2352eb733166 IPAPIWEB=192.168.1.200 HOME=/root </code> *
Modification du hostname au démarrage du conteneur <code shell> $ docker run -it -h idefix ubuntu root@idefix:/# </code> =====
Quelques autres commandes de Docker =====
=== Supprimer un conteneur === <code shell> docker rm [identifiant ou nom du
conteneur] </code>
=== Faire le ménage === La commande system et sa sous-commande prune permettant de réaliser le ménage dans
les conteneurs arrêtés, les images orphelines et d'autres ressources a priori non utilisées, sans supprimer les images. <code shell> docker
system prune </code>
=== Supprimer une image === Les conteneurs qui utilisent cette image doivent au préalable avoir été supprimés.
Le paramètre -f force cependant la suppression si cela n'est pas le cas. <code shell> docker rmi [identifiant ou nom de l'image] </code>
=== Lancer un conteneur avec un nom donné === <code shell> docker run --name=[nom fourni] [image] </code>
=== Lancer un
conteneur en mode détaché === Lors du lancement d'un conteneur, on perd l'accès à la console du serveur Ubuntu. Cette commande
permet de lancer le conteneur et de retrouver le shell du serveur sans arrêter le conteneur. <code shell> btssio@ubuntudocker:~$ docker
run -i -t -d ubuntu e70a11313849c4c2ce8beebe72935368b909759ab5eb4e2c2f1bcf715c1a2bd1 btssio@ubuntudocker:~$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES e70a11313849 ubuntu "/bin/bash" 3 seconds ago Up 2 s gallant_kilby
</code>
=== Se connecter à un conteneur lancé === Pour accéder ensuite à ce conteneur on utilise la commande attach avec l'identifiant
ou le nom du conteneur : <code shell> btssio@ubuntudocker:~/Documents/lab_exercice$ docker attach e70a11313849
root@e70a11313849:/# </code>
=== Arrêter un conteneur === <code shell> docker stop [conteneur] </code>
=== Démarrer un
conteneur arrêté === <code shell> docker start [conteneur] </code>
```

Docker propose également un **guide de découverte de Docker (en anglais)** avec son ordinateur personnel ou bien en ligne dans un lab (Cloud) : <https://www.docker.com/101-tutorial>

Autres ressources :

- <https://devopssec.fr/article/cours-complet-apprendre-technologie-docker>

===== Mémento Docker =====

Mémento Docker :
<https://www.digitalocean.com/community/tutorials/how-to-remove-docker-images-containers-and-volumes-fr>

===== Retour Accueil Docker ===== * Docker

From:
 / - **Les cours du BTS SIO**

Permanent link:
[/doku.php/reseau/docker/utiliserdocker?rev=1679922739](https://doku.php/reseau/docker/utiliserdocker?rev=1679922739)

Last update: **2023/03/27 15:12**

