1. Premier container
2. Hello World

Pour vous assurer que Docker est fonctionnel sur votre poste de travail, exécutez la commande suivante :

docker run hello-world

Pouvez-vous expliquer avec vos mots ce qui s’est passé suite à l’exécution de cette commande?

Afficher l’image d’un texte qui dit bonjours.

Resultat obtenu :

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/>

1. Observer un container

Exécutez à présent la commande : docker container ls -a

Retrouvez les informations suivantes sur le container lancé précédemment :

Quel est son identifiant ? 99f56f84cb29

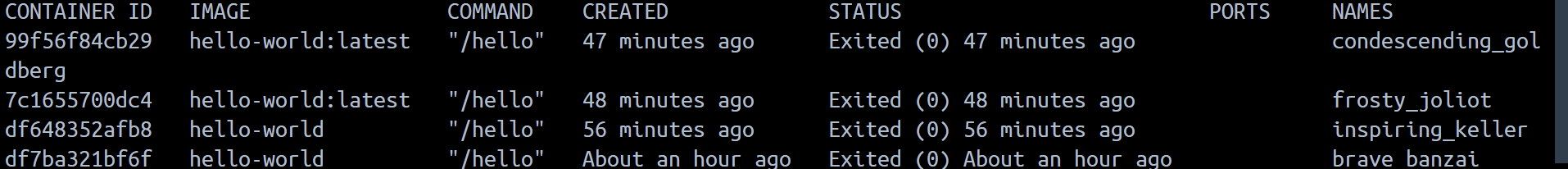
Quel est son nom ? condescending\_gol

Quel est son état ? Exited (0) 47 minutes ago

Quel est le nom de son image? Avez-vous vu au point 1.1. d’‘où cette image provenait? hello-world:latest

Quelle commande le container a-t-il exécuté? docker container list,

Si vous avez installer Docker Desktop, pouvez-vous retrouver ces mêmes informations dans l’interface graphique ?



1. Les images

Tapez à présent la commande suivante :

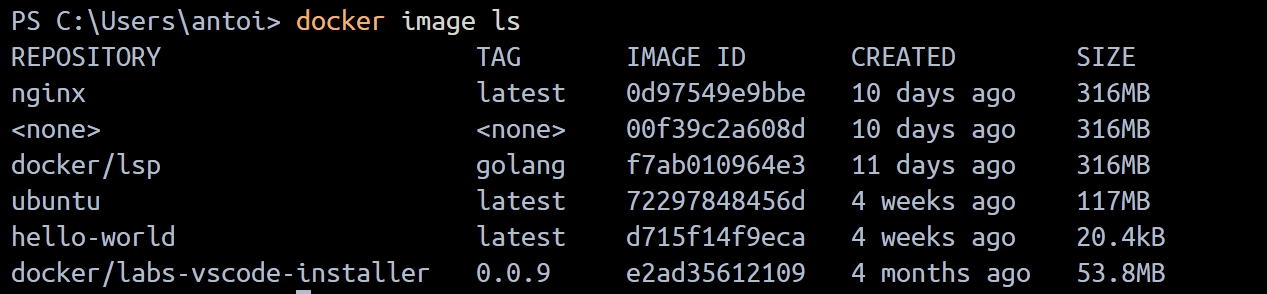
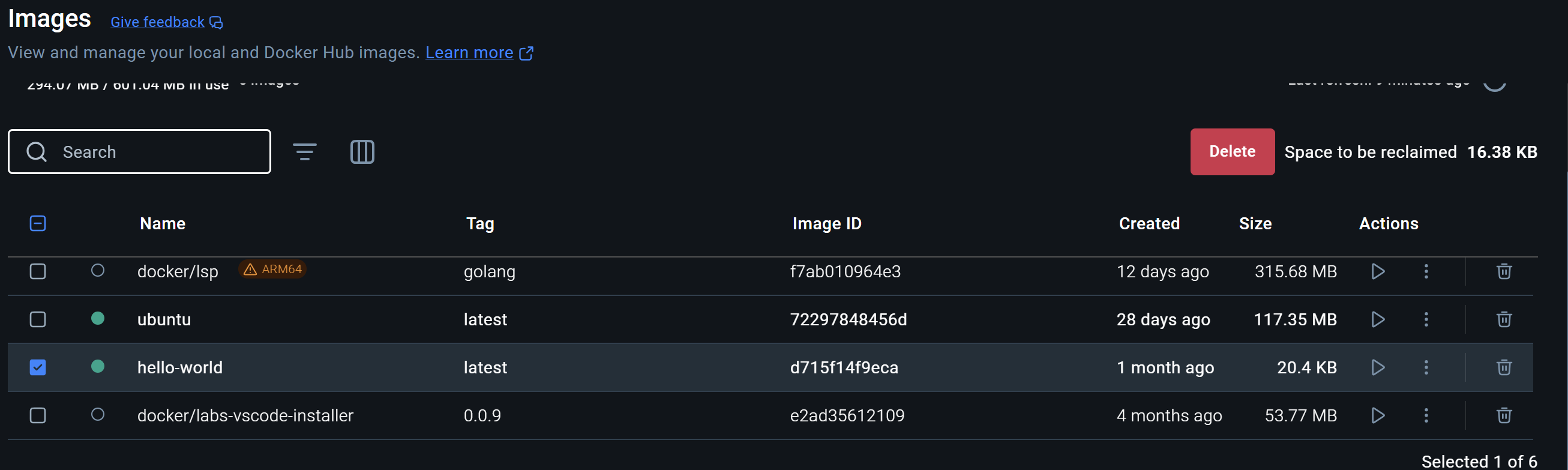
docker image ls

Quelles informations voyez-vous?

REPOSITORY TAG IMAGE ID CREATED SIZE

Quel est le lien avec ce que vous avez observé auparavant? Hello world

Comparez l’output de cette commande avec la vue correspondante de l’interface graphique.



Essayez de trouver la commande qui vous permettra de supprimer cette image. C’est une bonne idée de ne pas conserver les images non utilisées sur votre système de fichiers : même avec la mutualisation de couches, elles prennent de l’espace sur le disque!

Docker rmi image

REPOSITORY TAG IMAGE ID CREATED SIZE

hello-world latest d715f14f9eca 2 weeks ago 20.4kB

docker/labs-vscode-installer 0.0.9 e2ad35612109 3 months ago 53.8MB

1. Utiliser un container

Nous allons à présent voir comment il est possible d’interagir avec un container. Pour cela, nous allons prendre un simple container basé sur une distribution Linux Ubuntu, et interagir avec lui comme nous le ferions dans le cas d’une machine “normale” ou d’une machine virtuelle, via un shell interactif.

1. Interagir avec un container

Lancez un container Ubuntu avec la commande suivante :

docker run -it ubuntu bash

Normalement, après téléchargement de l’image ubuntu, le container se lance et vous obtenez un “prompt” dans votre console indiquant que vous pouvez interagir avec ce container.

A quoi servent les options i et t dans la commande ci-dessus?

I : --interactive Keep STDIN open even if not attached

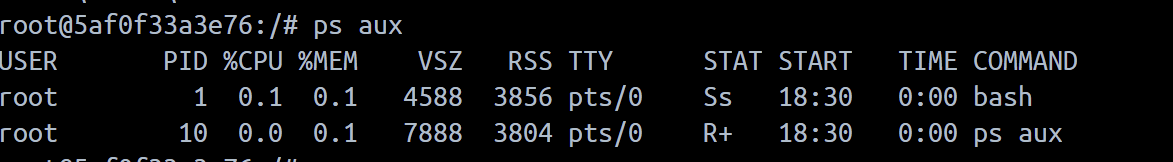
T : -t, --tty Allocate a pseudo-TTY

Chaque container Docker est destiné à exécuter une commande unique. Quelle est-elle dans ce cas-ci?

Dans le terminal bash ouvrir l’image d’ubuntu si installer sinon l’installer sur la machine

Dans le container, quels sont les processus présents? Et leurs PIDs?

Le pid est bash



Avec quel utilisateur êtes-vous loggé?

root

Votre container a-t’il accès à Internet? Qui est son résolveur?

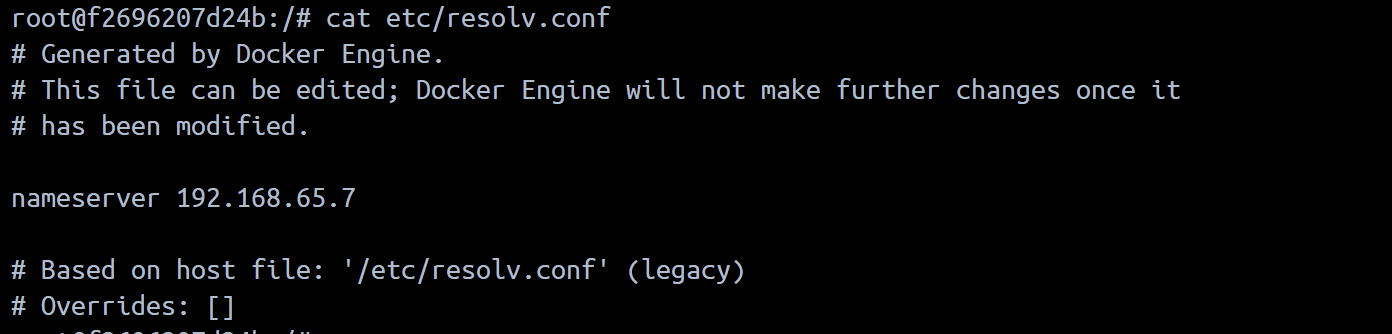
Oui.

Le résolveur est le DNS.

1. Inspecter un container

Dans un autre terminal et la commande docker inspect ou via l’interface graphique, essayez d’obtenir des informations sur ce container. Conseil : la commande grep peut vous être d’une grande aide!

Chaque container dispose d’une interface réseau. Quelle est l’adresse IP de l’interface de votre container?



Votre container a-t’il des ports ouverts?

Non



Une fois que vous avez pu retrouver l’information demandée, vous pouvez à présent éteindre votre container ubuntu. Deux options sont possibles :

Utiliser la commande docker stopdepuis la GUI ou un autre terminal.

Taper exit dans le shell du container. Cela aura pour effet de quitter le processus bash, qui est le processus principal du container. Une fois ce processus quitté, le container n’a plus de raison d’être et s’arrête automatiquement (il est néanmoins possible de le redémarrer).

1. Faire tourner un service dans un container

Les containers sont utiles pour faire tourner des services de manière isolée, tout en permettant beaucoup de souplesse pour la migration ou la mise à jour : les containers peuvent être détruits et redémarrés très facilement, simplement sur base d’une image. Nous allons ici lancer un petit serveur web et voir comment le rendre accessible.

Nous allons commencer par exécuter un container sur base de l’image officielle “nginx” (serveur web concurrent d’Apache). Cette image est disponible sur le Docker Hub, et sera automatiquement téléchargée lorsque vous tenterez de l’utiliser. Il s’agit d’une image sur laquelle nginx et ses dépendances ont été installée, et qui peut donc être utilisée telle quelle.

Instructions

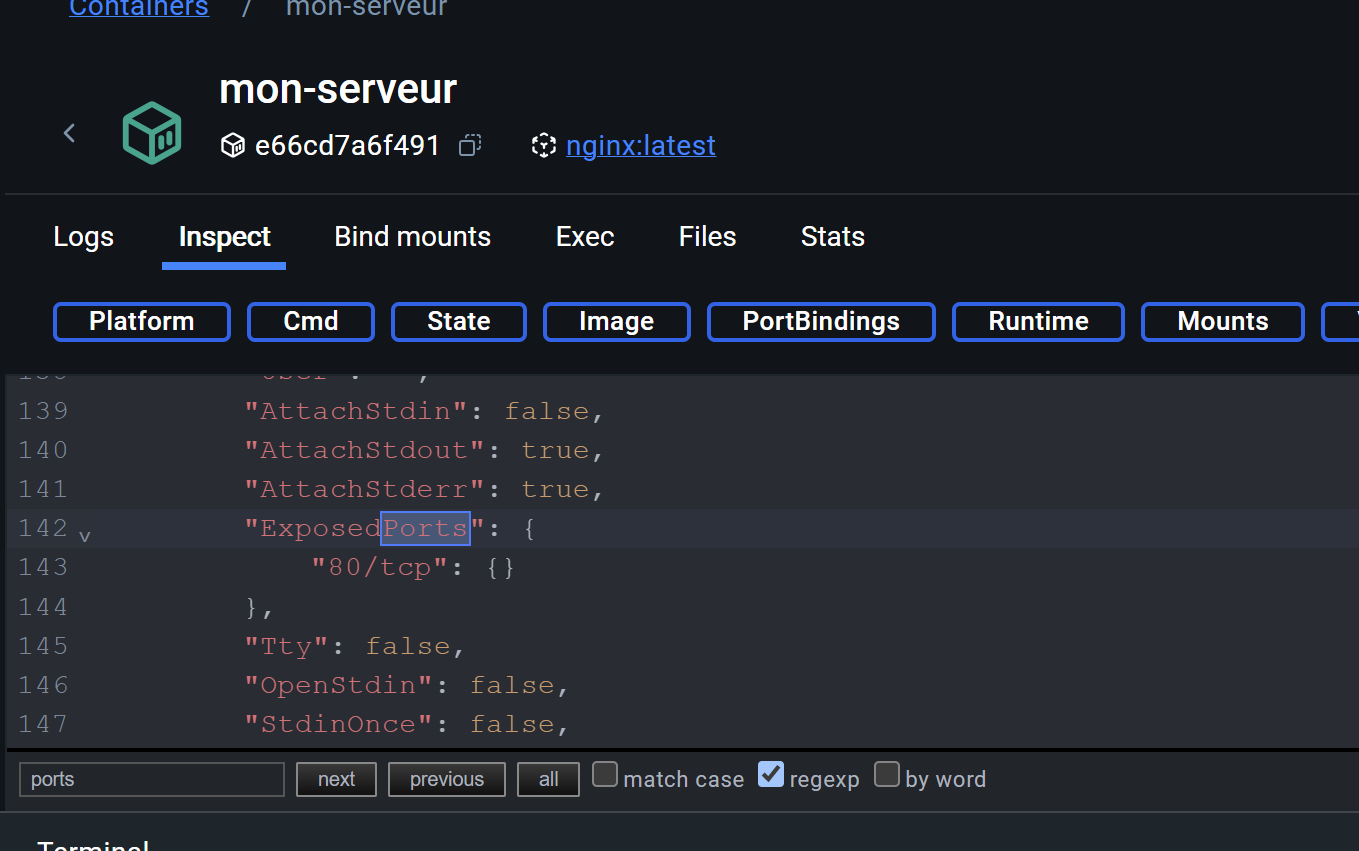
Exécutez la commande docker run --name=mon-serveur-web nginx dans un terminal.

Examinez à présent le container ainsi lancé, soit depuis l’interface graphique, soit depuis la console. Observez ce que vous voyez au niveau des “ports”.

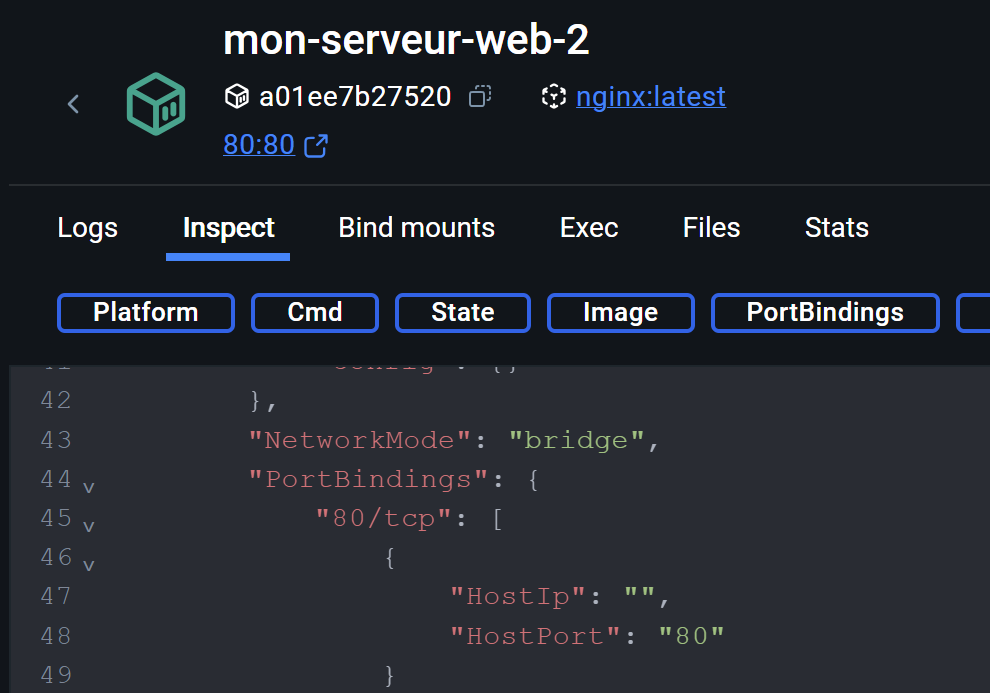
Le serveur nginx tourne donc sur le port 80, sur l’interface du container. Cependant, ce serveur n’est pas accessible en dehors de l’environnement Docker. Pour solutionner ce problème, nous allons effectuer une redirection de port (aussi appelée publication de port en terminologie Docker). Eteignez le container, puis relancez-en un nouveau, cette fois avec la commande docker run -p80:80 --name mon-serveur-web nginx.

Analysez le container et comparez les informations relatives aux ports ouverts avec celles obtenues deux étapes plus tôt. Quelle est la différence ? Si votre hôte est un système Linux, n’hésitez pas à confronter ces informations à l’output de netstat.

Essayez à présent d’accéder au serveur web via votre navigateur. Examinez à présent le container ainsi lancé, soit depuis l’interface graphique, soit depuis la console. Observez ce que vous voyez au niveau des “ports”.



En utilisant la commandeOn peut observer que le port n’est pas défini. Par conséquent l’image n’est pas accessibles en dehors de la machine.



En utilisant la commande : « docker run -p80:80 --name mon-serveur-web-2 nginx ». on peut observer que le l’image est accessible en dehors de la machine via au port 80.

1. Construire des images

Pour la suite de ce TP, référez-vous à cette section du support théorique.

1. Figer un container

Imaginons que nous souhaitions configurer un service conformément à nos besoins. Par exemple, dans le cas de notre petit serveur web, nous souhaitons modifier la page d’accueil pour la personnaliser.

Nous pouvons lancer un container sur base d’une image de base, ouvrir un shell interactif et aller éditer nos configurations.

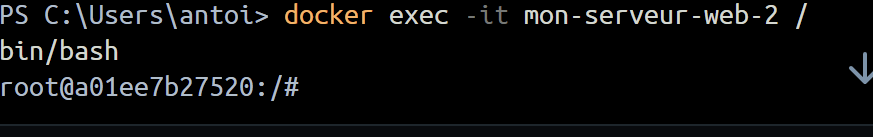
Instructions

Vérifiez que votre serveur web tourne toujours. Son processus principal (“commande”) est normalement le processus nginx.

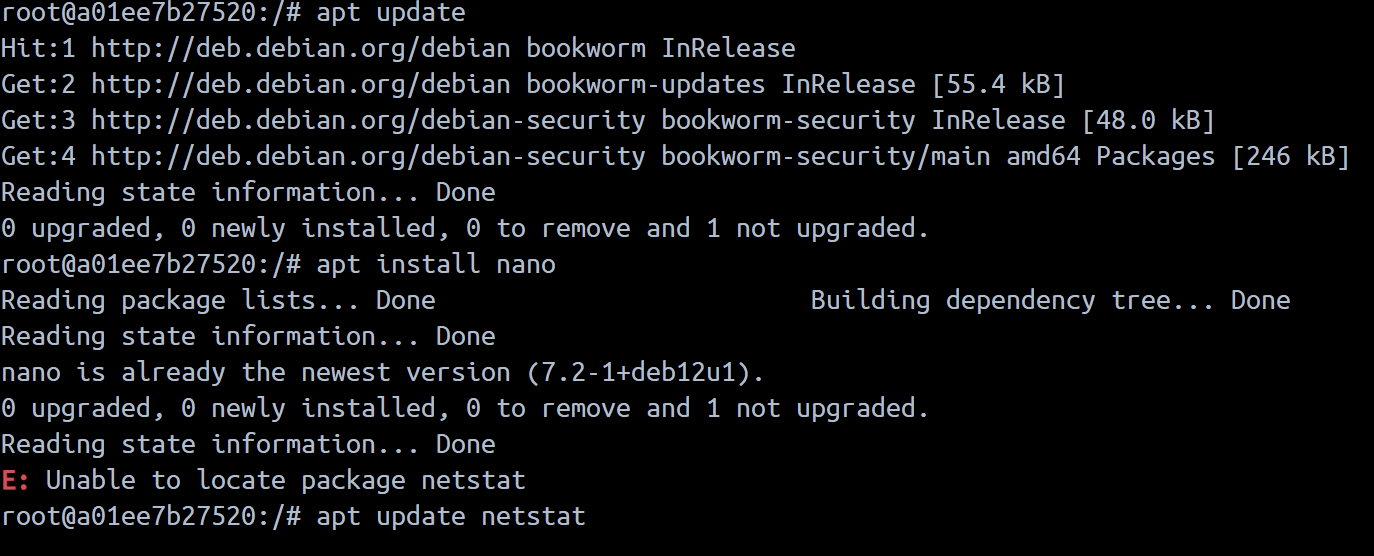
ok

Démarrez terminal bash. Pour cela, nous allons lancer un second processus sur le container, en y attachant notre terminal afin de pouvoir interagir : docker exec -it \<nom-du-container> /bin/bash

Nous allons à présent commencer à personnaliser notre container. Nous pouvons installer des outils supplémentaires, via un apt updatepuis apt install. Installez ce que vous voulez, par exemple nano, ou encore votre outil favori netstat.



Pour personnaliser la page d’accueil, vous pouvez ensuite aller éditer le fichier /usr/share/nginx/html et le transformer à votre goût. Tester ensuite avec votre navigateur que les modifications ont bien eu lieu.



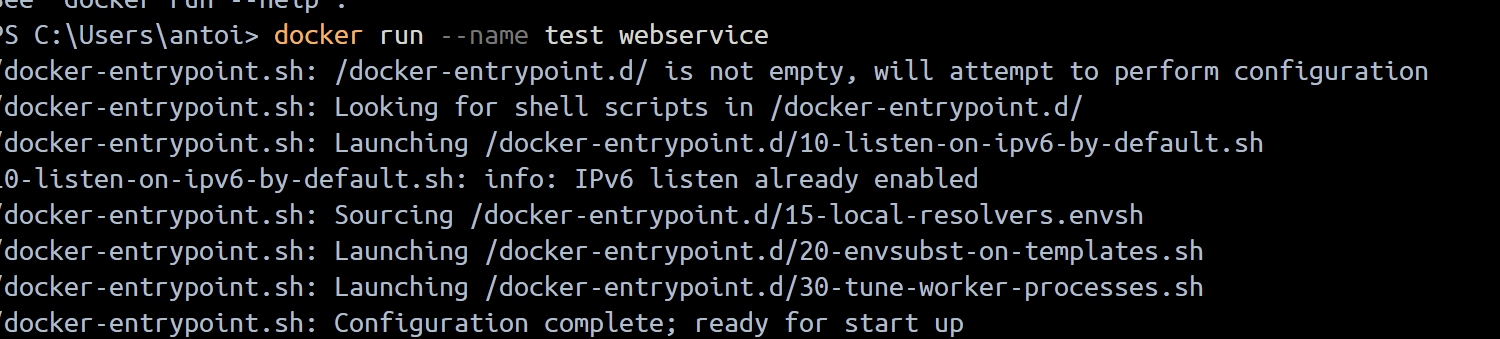
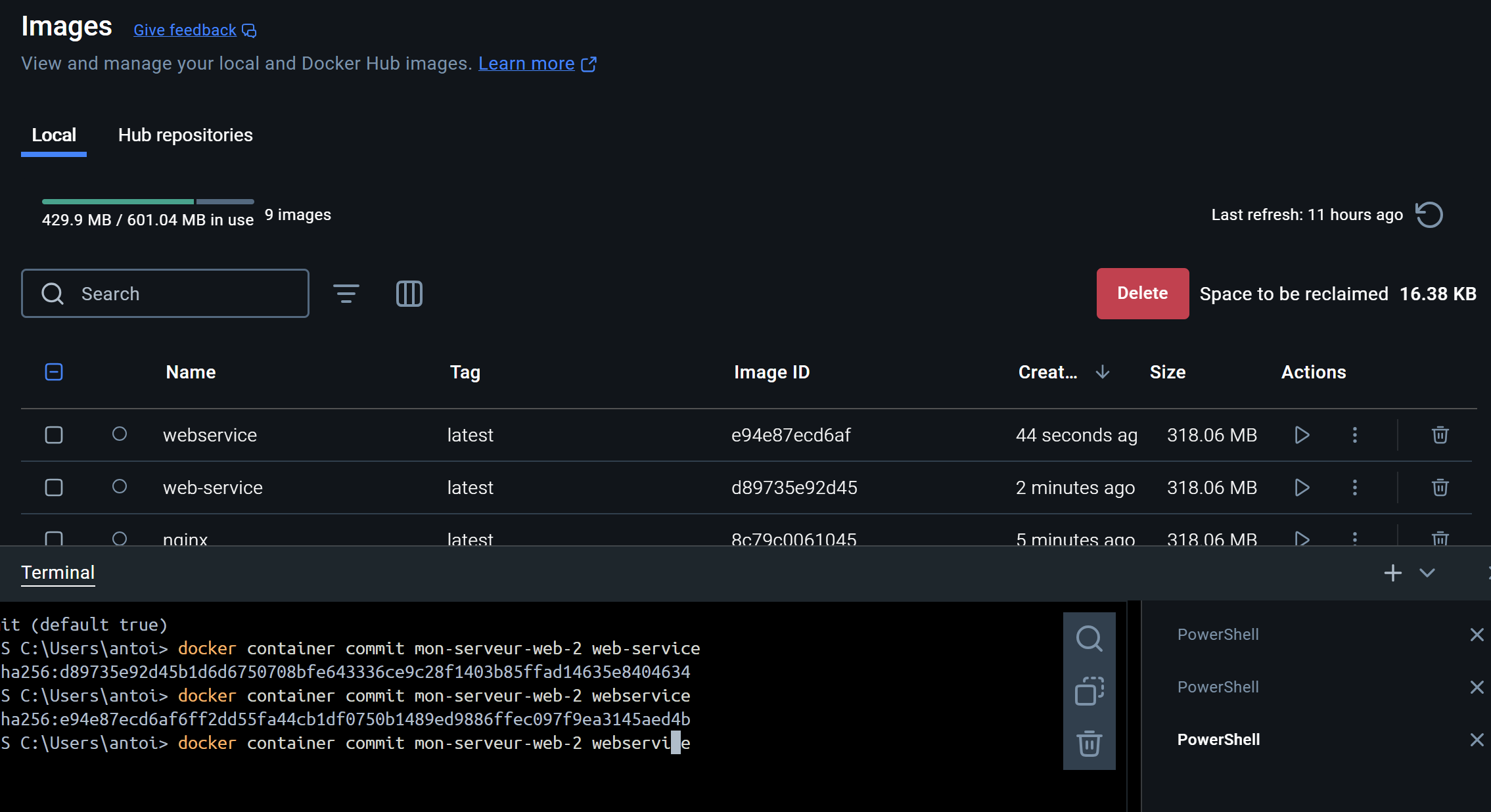
Instructions

Utilisez la commande docker container commit \<nom-container> \<nom-image>pour créer une image sur base de votre serveur web.

Vérifiez dans la liste de vos images que vous retrouvez bien celle de votre serveur web.

Arrêtez votre précédent container si ce n’est pas déjà fait, afin de libérer le port 80.

Démarrez un nouveau container sur base de votre image personnelle, et vérifiez qu’elle fonctionne comme prévu



1. Créer une image sur base d’un Dockerfile

Instructions

Créez un Dockerfile réalisant les mêmes opérations que celles que vous avez effectuées plus tôt, sur base de cet exemple. Enregistrez-le dans un répertoire de travail, en l’appelant Dockerfile.

FROM nginx:latest

RUN apt update

RUN apt install -y nano net-tools

COPY index.html /usr/share/nginx/html

Mettez également dans ce répertoire un fichier index.htmlqui sera votre page d’accueil personnalisée.

Exécutez ensuite la commande docker build -t <nom de l'image> . dans ce même répertoire. Cela aura pour effet de déclencher la construction d’une nouvelle image, sur base de l’image nginx.

Vérifiez ensuite que cette image fonctionne comme prévu.

w

