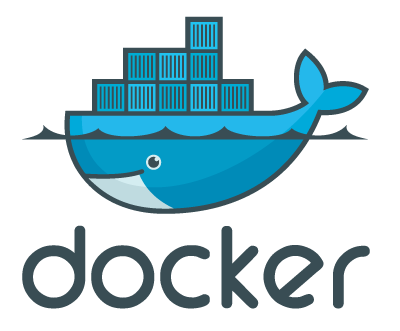
Compte Rendu

Semaine 07/02 au 11/02

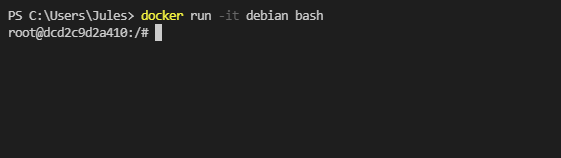
Premier pas vers Docker



Jules MORFIN

**Première étape : Création d’un container**

après avoir téléchargé tout les outils, je passe a la création d’un container via Docker



Avec cette commande J’ai lancé le téléchargement d’une image Debian dans un container

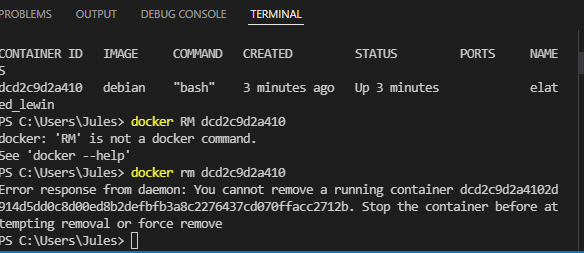
Donc la je lance l’application bash dans mon container, en l’occurrence j’avais déjà télécharger Debian.

Une image contenant texte

Description générée automatiquement

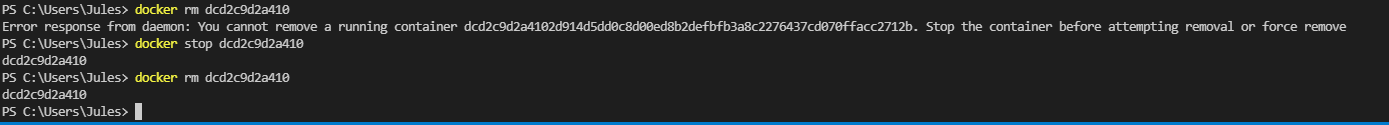
Maintenant grâce a cette commande je peu visualisé les containers que j’ai, dont aussi l’image qu’il possède, sont temps de création etc, son ID et son nom.

La je me retrouve à la première problématique, je voulais lui donner un nom personnalisé pas celui que Docker donne au hasard. Je décide donc de le supprimé via un Docker rm et de le recrée.



Sauf que première erreur, je ne pouvais pas supprimé un container qui tourne actuellement

Après mettre renseigné, il faut faire un stop pour fermer le container avant de le supprimé ce qui semble pour logique.



Une fois que mon container est supprimé je vais en recrée un avec un nom cette fois « TEST ».

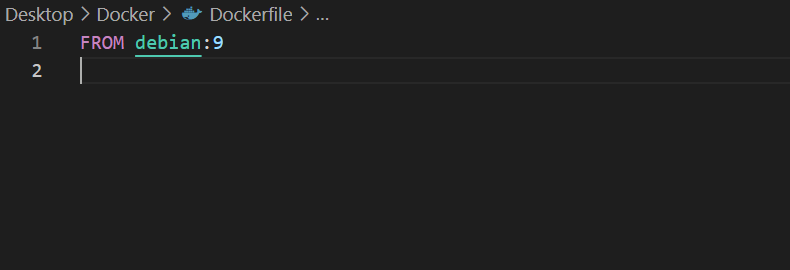
J’affine donc ma commande en « docker container run -d --publish 80:80 --name TEST debian bash »

Docker container pour spécifier ce que je veux donc configurer le container, -d c’est deamon pour l’arrière plan, --publish 80 :80, c’est pour mappé le port 80 de ma machine et de mon container, debian c’est pour spécifié l’image debian, puis enfin name, c’est pour donner un nom à ma machine.

**Deuxième étape : Création d’un Docker file**

Pour crée le docker file, il suffit de crée un fichier de notre dossier de travail, en l’occurrence pour moi c’est sur le bureau dans le un dossier Docker . (Qui est synchroniser avec mon GIThub)

Ensuite j’ai spécifié l’image et la version que je veux utiliser comme ceci :



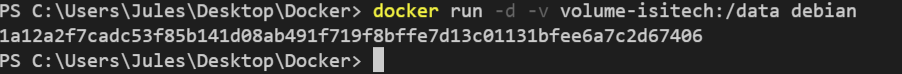
Ensuite je sauvegarde le fichier dans mon dossier local DOCKER, puis via comme ceci pour pousser la conf sur github :

Une image contenant texte

Description générée automatiquement

Ayant déjà sauvegarder en amont, il me dit que tout est déjà synchroniser.

Ensuite, comme demande je crée un volume dans mon image :

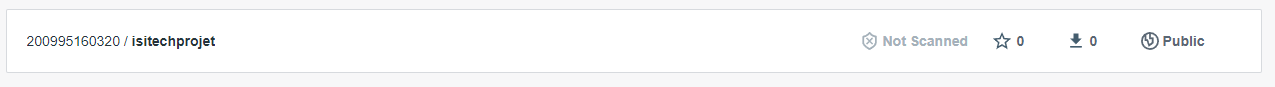


Je peu donc vérifié via la commande suivante si mon volume a bien été crée dans le dossier data.

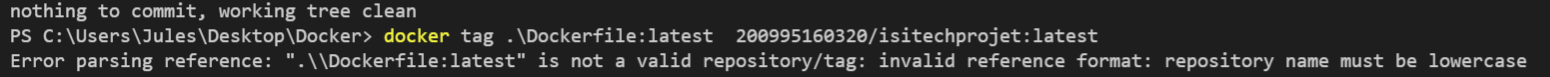
Une image contenant texte

Description générée automatiquement

Maintenant il faut le sauvegarder sur dockerHub donc pour ça j’ai créé un compte, et un Repo.



Bon malheureusement je n’arrive pas a push mon image dans mon dockerhub, Donc la commande que je voulais faire passer est celle-ci :



Après j’aurais fait un docker push 200995160320/isitechprojet et je pense que cela aurait pousser dans mon dockerhub.

J’aurais pu ensuite partager mon repo

**Etape commune :SERV MINECRAFT :**

**Explication du projet**

1: création d'un dossier avec le nom du container,

2: ensuite dans ce répertoire créer un fichier docker-compose.yml qui servira à créer le container

3: création du docker files afin de créer l'images pour le docker-compose

4: création du .env, qui sert a à incrémenter des variable d'environnements

5: via PowerShell crée un fichier server-data avec le nom du container qui permet de stocker la data base du server

6: dans la data base intégrer le dossier d'installation de Minecraft

**Etape 2 création docker Files**

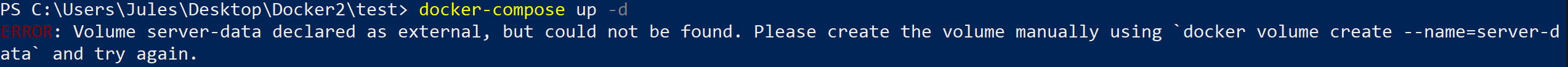
Création d'un docker Files sous Ubuntu 18.04 qui installer une version de java pour faire tourner le server Minecraft

Le docker compose-yml, crée un container baser sur l'image docker files crée peu avant, il créer un container Minecraft-server sur lequel est héberger le serveur Minecraft

après le codage du docker files et du dockercompose.yml, on lance la commande "dockercompose up --build, cela créer le container et permet de jouer en local

**Etape 3 en cours de réalisation ouverture au multi**

Premièrement après que la config a été faite par mes collègues, j’ai moi aussi essayer de la configurer sur mon environnement, et essayer en local de vérifié si cela fonctionner.



Après de nombreuse erreur (doublon de docker-compose avec l’import des fichiers de Jeremy dans mes fichiers, problème de version de Docker-compose, problème de synthaxe des commandes etc),

Nous avons pu arriver jusqu’à l’étape du lancement du serv :

Une image contenant texte

Description générée automatiquement

Malheureusement, après le building une nouvelle erreur est apparue, et cause de manque de temps, nous avons pas pu trouver la solution. Une image contenant table

Description générée automatiquement

Nous avons donc essayer rapidement de le faire en multi grâce à Ngrok, solution pratique au vue du manque de temps, mais cette tentative a elle aussi échoué.



Une image contenant texte

Description générée automatiquement

**Conclusion :**

Pour moi, cette expérience a été très compliqué a réalisé, d’une part sans le cour de lundi matin (malgré l’enregistrement d’une trentaine de minutes) j’ai mis énormément de temps a comprendre la logique de cette outil, et son fonctionnement, aussi pour git, un outil que je n’avais jamais utilisé avant. Je suis conscient que je n’ai pas pu rendre grand-chose, mais tout de même partant de rien, j’ai l’impression maintenant d’avoir compris beaucoup de chose sur ce sujet et c’est un réel soulagement après un tel casse-tête, je me rends compte que nous pouvons réaliser énormément de chose grâce a Docker, et que ça, c’était seulement la partie émerger de l’iceberg.

PS : (dans mon GIT il y’a deux branches, je ne sais pas si je devais vous le préciser)