



(https://profile.intra.42.fr)

(https://profile.intra.42.fr/searches) **SCALE FOR PROJECT DOCKER-1 (/PROJECTS/DOCKER-1)**

You should correct 1 student in this team



Git repository

vogsphere@vogsphere.42.fr:intra/2018/activities/docker_1/rnu{

Introduction

Nous vous demandons, pour le bon déroulement de cette notation :

- De rester courtois, poli, respectueux, constructif lors de cet échange. Le lien de confiance entre la communauté 42 et vous en dépend.
- De bien mettre en évidence auprès de la personne notée (ou du groupe) les dysfonctionnements éventuels.
- D'accepter qu'il puisse y avoir parfois des différences d'interprétation sur les demandes du sujet ou l'étendue des fonctionnalités. Restez ouvert d'esprit face à la vision de l'autre (a-t-il raison ou tort ?), et notez le plus honnêtement possible.

Bonne soutenance à tous !


Guidelines


RAPPELÉZ VOUS QUE VOUS NE DEVEZ CORRIGER QUE CE QUI SE TROUVE SUR LE DÉPÔT DE RENDU DE L'ÉTUDIANT.

Il s'agit de faire un "git clone" du dépôt de rendu, et de corriger ce qui s'y trouve.

Si le correcteur n'a pas encore fait ce projet, il est obligatoire de lire le sujet en entier avant de commencer cette soutenance. |

Attachments

 Subject (<https://cdn.intra.42.fr/pdf/pdf/1008/docker.en.pdf>)

 Sujet (<https://cdn.intra.42.fr/pdf/pdf/1007/docker.fr.pdf>)

Préliminaires

Consignes préliminaires

Avant de commencer, prenez bien soin de vérifier les éléments suivants :

- Il y a bien un rendu (dans le dépôt git), respectant la hiérarchie demandée
- Pas de triche, l'étudiant doit pouvoir expliquer les commandes tapées.

Si un élément de cette liste n'est pas respecté, la notation s'arrête là. Utilisez le flag approprié. Vous êtes encouragés à continuer de débattre du projet, mais le barème n'est pas appliqué.

 Yes

 No

Mise en place de l'environnement de travail

Verifiez que les points suivant sont bien appliqués (faire le necessaire si besoin)

- La commande "docker version" affiche bien la version de docker et de docker-machine
- Virtualbox est bien installé sur le dump de correction
- Un lien symbolique permet de relire les dossiers .docker et VirtualBox VMs du goinfre vers le home.

 Yes

 No

How to Docker

Cette partie vous permet d'évaluer la première partie du sujet. Pour des questions d'optimisation, vous allez noter le rendu par ensemble de questions répondant à une même thématique. Pour chaque commande, vous allez devoir l'exécuter via votre shell en faisant `cat 01` ou \$(cat 01). Pour partir sur de bonnes bases, vous allez déjà supprimer la machine virtuelle Char si elle existe.

Avant de commencer

Verifiez bien que :

- 01 : Une machine virtuelle s'est bien créée au nom de Char avec le driver virtualbox (docker-machine ls)
- 02 : Vous devriez avoir affiché quelque chose comme 192.168.99.xxx

- 03 : la commande "env" dans votre shell affiche bien 4 variables "DOCKER_*".
En revanche, aucune commande d'assignation de variable d'environnement ne doit
etre present dans le script, sinon c'est zero et fin de la correction.

Si l'un de ces points n'est pas respecté, cette partie est comptée fausse et vous passez a la suivante

✓ Yes

✗ No

Mon premier container (04-09)

Verifiez bien que :

- 04 : l'image hello-world est bien disponible en faisant un docker images (sans erreurs)
- 05 : Un message d'accueil est affiché sur le terminal.
- 06 : Une fois le container lancé, verifiez que :
 - * Le container s'appelle bien overlord (docker ps)
 - * Le port 80 du container est bien bindé au port 5000 de la machine virtuelle (docker ps)
 - * La commande "curl http://:5000" renvoie bien du HTML correspondant a la page de test de nginx
 - * La commande "docker inspect -f "{ .HostConfig.RestartPolicy } }" overlord" renvoie bien {always ..}
- 07 : Une adresse IP de la forme "172.X.0.X" apparait (verifiable avec un docker inspect)
- 08 : Une fois le container alpine lancé, verifiez que :
 - * Vous avez bien l'invite " / \# "
 - * Un whoami sur cette console vous donne bien "root".
 - * Un "exit" quitte bien le container
 - * Un docker ps -a ne retrouve aucune trace d'un container alpine.
- 09 : Une fois le container debian lancé, verifiez que :
 - * Vous avez bien l'invite " root@hostname: / \# "
 - * L'execution des commandes donnés dans le fichier s'exécute.
 - * Vous avez la possibilité de faire un "git clone https://github.com/docker/docker.git"
 - * Un petit programme en C devrait etre compilable et executable dans le contexte du container (installez vim si besoin)
 - * Un docker ps -a ne retrouve aucune trace d'un container debian.

Si l'un de ces points n'est pas respecté, cette partie est comptée fausse et vous passez a la suivante

✓ Yes

✗ No

Wordpress (10-18)

Verifiez bien que :

- 10-11 : le volume "hatchery" soit bien créé, et que la commande de 11 vous montre bien un volume "hatchery" en plus des autres

- 12-13 : Une fois le container mysql lancé:
 - * Vous revenez bien sur votre shell classique (seul le digest du container lancé apparait)
 - * La commande 13 montre bien que MYSQL_ROOT_PASSWORD est set à "Kerrig@n" et que MYSQL_DATABASE est bien set sur "zerglings"
 - * Un "docker inspect spawning-pool" vous affiche bien qu'un volume "hatchery" est monté sur la destination "/var/lib/mysql" (Clé "Mounts")
 - * La commande "docker inspect -f '{{ .HostConfig.RestartPolicy }}' spawning-pool" renvoie bien {always ..}
 - * La commande "docker exec -it spawning-pool mysql -uroot -p" vous demande bien de rentrer un mot de passe
 - * Le mot de passe est bien "Kerrig@n" et l'invite de commande mysql est bien visible
 - * la commande SQL "show databases" montre bien une database "zerglings"
- 14 : Une fois le container wordpress lancé:
 - * Vous revenez bien sur votre shell classique (seul le digest du container lancé apparait)
 - * Vous avez possibilité de lancer un navigateur et aller sur <http://:8080>, et configurer Wordpress pour que la db utilisée soit celle de spawning-pool
 - * Tentez une connexion sur
- 15 : Une fois le container phpmyadmin lancé :
 - * Vous revenez bien sur votre shell classique (seul le digest du container lancé apparait)
 - * Vous avez possibilité de lancer un navigateur et aller sur <http://:8081>
 - * Vous pouvez accéder a la base de données dispo sur spawning pool et vérifier que vous avez des tables wordpress créées
- 16 : Les logs du container mysql s'affichent en direct
- 17 : Ca parle de lui meme
- 18 : Utiliser la commande "docker exec -it overlord /bin/sh -c "kill 1"", puis utiliser la commande "docker inspect -f '{{ .RestartCount }}' overlord" qui devrait alors s'être incrémenter de 1.

Si l'un de ces points n'est pas respecté, cette partie est comptée fausse et vous passez a la suivante

✓ Yes

✗ No

Abathur (19)

Effectuez toutes les commandes de l'exercice afin de mettre en place le container.
Le container doit embarquer le framework Flask (pip install Flask).
Un script python doit etre present dans le dossier partagé entre l'hôte et le container.
Lancer ce script depuis le contexte du container,
Ainsi, aller sur <http://:3000> affichera bien Hello World en titre.
De meme, des logs doivent apparaitre sur le terminal.

Si l'un de ces points n'est pas respecté, cette partie est comptée fausse et vous passez a la suivante

✓ Yes

✗ No

The Swarm (20-30)

Vérifiez bien que:

- 20 : un "docker node ls" affiche bien Char dans les HOSTNAME et son MANAGER STATUS est bien Leader
- 21 : meme punition qu'avec la question 01.
- 22 : un "docker node ls" affiche bien Aiur dans les HOSTNAME et son MANAGER STATUS soit autre que Leader
- 23 : la commande 23 fonctionne et fait apparaitre les 2 nodes.
- 24-25 : une fois le service rabbitmq lancé:
 - * la commande `25` affiche bien le service "orbital-command" en 1 réplique sur 1 en mode répliqué sur une image rabbitmq:latest
 - * un "docker service ps orbital-command" vous affiche bien le service en status "Running"
 - * un "docker service inspect -f "{{ .Spec.TaskTemplate.ContainerSpec }}" orbital-command" vous donne bien deux variables d'environnement qui set un user et un password spécifique
- 26-27 : une fois le service engineering-bay lancé:
 - * un "docker service ps engineering-bay" vous affiche bien les services en status "Running" avec 2 répliques de faite
 - * un "docker service inspect -f "{{ .Spec.TaskTemplate.ContainerSpec }}" engineering-bay" vous donne bien deux variables d'environnement qui set un user et un password qui permettent la connexion au service orbital-command
 - * la commande `27` vous fait bien defiler les logs d'une des 2 tasks du service... et vous montre bien des zergs attaquant orbital-command
- 28-29 : une fois le service marines lancé:
 - * un "docker service ps marines" vous affiche bien les services en status "Running" avec 2 répliques de faite
 - * un "docker service inspect -f "{{ .Spec.TaskTemplate.ContainerSpec }}" marines" vous donne bien deux variables d'environnement qui set un user et un password qui permettent la connexion au service orbital-command
 - * scruter les logs du service marines montre bien que les marines sont en train de poutrer du Zerg
- 30 : un "docker service ps marines" vous affiche bien les services en status "Running" avec 20 répliques de faite. Le service en lui meme n'est pas arrete, mais bel et bien updaté.

Si l'un de ces points n'est pas respecté, cette partie est comptée fausse et vous passez a la suivante

✓ Yes

✗ No

Viscera Cleanup Detail (31-34)

Toutes les commandes de cette partie doivent ne faire chacune qu'une seule ligne. (allez y à grand renfort de wc -l)

Vérifiez bien que:

- 31 : un "docker service ls" n'affiche plus aucun service independemment de l'état de celui-ci

- 32 : un "docker ps -a" n'affiche plus aucun container independemment de l'état de celui-ci
- 33 : un "docker images ls" n'affiche plus aucune image
- 34 : un "docker-machine ls" n'affiche plus la machine virtuelle Aiur

Si l'un de ces points n'est pas respecté, cette partie est comptée fausse et vous passez a la suivante

☒ Yes

☐ No

DockerFiles

Cette partie vous permet d'évaluer la seconde partie du sujet. Vous allez devoir construire chaque Dockerfile et évaluer la bonne mise en place de l'application Pour partir sur de bonne bases, vous allez déjà repartir sur de bonnes bases. Soit vous refaites une machine virtuelle, soit vous utilisez Docker for Mac, au choix. L'important est qu'un `docker ps -a` dans le terminal vous montre absolument rien.

Vim // Emacs

Buildez ce dockerfile puis lancez le.

Vim ou emacs doit se lancer, et le mode "explorer" de l'éditeur doit vous montrer que vous êtes bien dans le contexte du container et non de votre hôte.

Faites les tests nécessaires.

Si l'un de ces points n'est pas respecté, cette partie est comptée fausse et vous passez a la suivante

☒ Yes

☐ No

BYOTSS

Buildez ce dockerfile puis lancez le.

Il doit apparaitre en tâche de fond.

Vous pouvez vous connecter aisément avec un client TeamSpeak classique dessus (le prendre sur le MSC si il n'est pas installé)

Si l'un de ces points n'est pas respecté, cette partie est comptée fausse et vous passez a la suivante

☒ Yes

☐ No

Dockerfile in a Dockerfile... in a Dockerfile ?

Buildez le dockerfile et poussez le quelque part (hub docker, registry local...).

Profitez de ce moment pour créer une application Rails vierge dans le repertoire (prenez un container Ruby et faites le nécessaire).

Copiez le dockerfile du sujet, tentez un build et lancez le container avec le necessaire (expose de port, mode detache...) .

Vérifiez que vous pouvez accéder à l'application Rails en essayant d'accéder sur l'IP de la machine via le port exposé.

Si l'un de ces points n'est pas respecté, cette partie est comptée fausse et vous passez a la suivante

☒ Yes

☐ No

Salade Tomates Oignons

Buildez le dockerfile et lancez le avec le necessaire (expose de port, mode detache...) .

Vérifiez bien que Gitlab est disponible, que vous pouvez créer des repos dessus et que vous pouvez push autant en HTTPS qu'en SSH.

Si l'un de ces points n'est pas respecté, cette partie est comptée fausse et vous passez a la suivante

☒ Yes

☐ No

Bonus

Les bonus ne doivent être évalués que si et seulement si la partie obligatoire est PARFAITE. Par PARFAITE, on entend bien évidemment qu'elle est entièrement réalisée, qu'il n'est pas possible de mettre son comportement en défaut, même en cas d'erreur, aussi vicieuse soit-elle, de mauvaise utilisation, etc. Concrètement, cela signifie que si la partie obligatoire n'a pas obtenu TOUS les points pendant cette soutenance, les bonus doivent être intégralement IGNORÉS.

🎵 I feel it coming... I feel it coming... I feel it coming... I feel it coming... 🎵

A vous d'évaluer les différents dockerfiles du dossier `02_bonus` .

L'attribution des points est à la libre discrétion du correcteur.

Rate it from 0 (failed) through 5 (excellent)

5

Ratings

Don't forget to check the flag corresponding to the defense

☒ Ok

★ Outstanding project

📁 Empty work

📁 Incomplete work

📁 Cheat

💥 Crash

Conclusion

Leave a comment on this evaluation

bonne explication good
project !

Finish evaluation