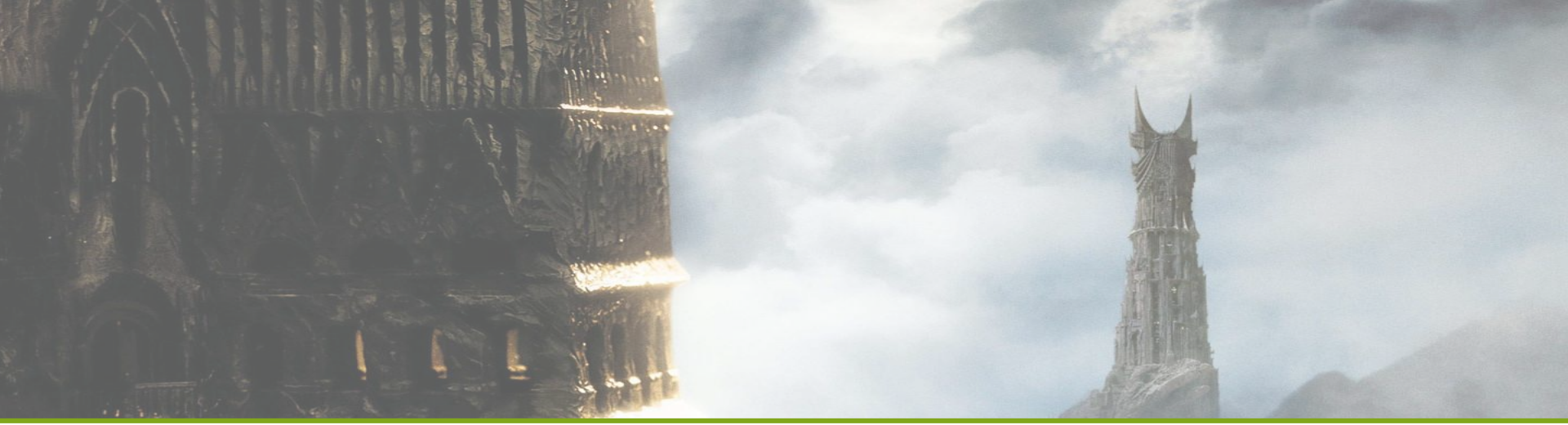
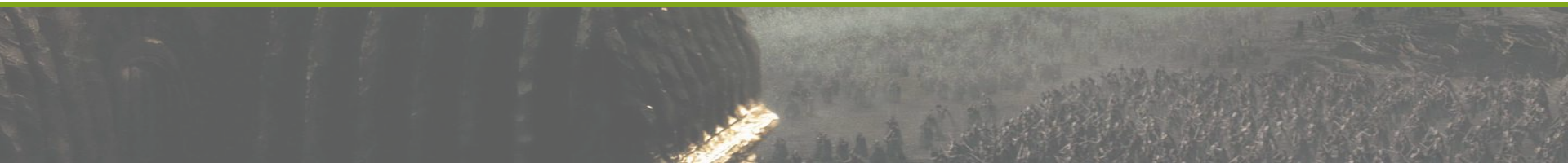


Docker - Commandes principales

ESGI - 2016 / 2017



Commandes principales

Pour obtenir la liste complète des commandes

- docker help
- docker <command> --help

https://docs.docker.com/engine/reference/commandline/

docker info

Affiche les informations sur l'installation de docker

- le nombre de containers (lancés, en pause, arrêtés)
- le nombre d'images
- la version
- la mémoire utilisée, le nombre de cpu
- l'utilisateur du hub

L'option -D fournit toutes les informations disponibles

docker run

Exécute un container depuis une image

- il existe de nombreux paramètres pour sa configuration, et qui peuvent être combinés
- docker run [options] image[:tag|@digest] [command] [arg1, arg2, …]
  - ex. : docker run -ti esgi-sshd:2.0 /bin/sh
  - -d lance en background

docker ps

Liste et détaille les containers (par défaut ceux lancés)

- -a : liste tous les containers (même arrêtés)
- -f ou --filter pour filtrer par nom, label, exit, …
- de nombreuses options existent pour afficher + ou - d'informations
- -q retourne seulement l'id (ex: sert pour concaténer le résultat à d'autres commandes linux)

docker logs

Récupère les logs d’un container

- docker logs [options] container (id ou nom)

docker images

Liste et donne des détails sur les images

- affichage LIFO (l'image la plus récent est tout en haut)
- indique le repository, tag et taille de l'image
- docker images comporte des couches intermédiaires qui permettent la réutilisation rapide avec docker build, toutes les étapes sont mises en cache (mais n'apparaissent pas par défaut)
  - utiliser l'option -a pour tout afficher
- la taille indiquée est la taille de l'image + les images parents (pas forcément affichées)
- la même image est affichée autant de fois qu'elle a de tags

docker exec

Lance une commande dans un container lancé

- ex. : docker exec -ti <image> /bin/sh, lance un processus de shell et le retourne, et permet ainsi de se connecter en terminal au container
- cette commande n’est pas relancée si le container est relancé

docker start / stop

La commande **docker start** lance un ou plusieurs containers arrêtés

La commande **docker stop** arrête un ou plusieurs containers lancés

docker rm

Supprime un ou plusieurs containers

- peut être supprimé par l'id ou le nom du container
- un container lancé ne peut pas être supprimé
  - pour forcer la suppression, utiliser l'option -f

docker rmi

Supprime une ou plusieurs images

- aucun container ne doit utiliser l'image (sinon erreur)
- sinon passer le paramêtre -f pour forcer (les containers utilisant l'image seront orphelins, mais pourront continuer à tourner ou à être démarrés)

docker pull

Télécharge les images du docker hub ou d'un registre privé

- est exécuté lorsque vous lancez docker run et que l'image n'est pas en local
- par défaut, le tag latest est utilisé. Il est possible de spécifier le tag en rajoutant :tag (ex. esgi:2.0)
- par défaut, télécharge l'image du docker hub

docker login / logout

Pour pouvoir travailler avec le hub (répertoire public d'images), vous devez créer un compte sur hub.docker.com

- l'authentification est obligatoire (pour les repos privées également)
- les informations d'authentification sont nécessaires pour la commande docker search

docker commit

Cette commande permet de créer une nouvelle image à partir de modifications effectuées depuis un shell interactif depuis une autre image

- une des manières de créer une image (l'autre sera en utilisant un dockerfile)
- par défaut, le container est arrêté durant le commit (--pause=true)
- vous pouvez fournir le nom de la nouvelle image et du repo après le nom/id du container à utiliser

docker push

Envoi (partage) une image de votre hôte vers le docker hub ou un repo privé

- l'image doit respecter une certaine nomenclature de nom et tag (https://docs.docker.com/engine/reference/commandline/tag/)

docker search

Recherche une image sur le docker hub

- l'option -f / --filter-value permet d'appliquer un filtre sur le nombre d'étoiles, si l'image est officielle ou si elle correspond à des builds automatiques

docker update

Permet d’actualiser la configuration d’un ou plusieurs containers

- fonctionne même si le container est lancé

Test final

Maintenant à vous de jouer

Dans un premier terminal (terminal 1)

- récupérer alpine
- vérifier que l'image alpine est bien présente
- démarrer un container avec un shell de manière interactive avec l'image alpine
  - lister les processus en cours (2 processus)

Test final

Dans un nouveau terminal (terminal 2)

- lister les containers lancés
- ajouter un nouveau processus (ping www.esgi.com) au container (attention il vous faut le nom du container)

Dans votre container (terminal 1)

- lister de nouveau les processus en cours (3 processus)
- créer un fichier avec votre nom

Test final

Dans le terminal 2

- créer une nouvelle image à partir du container que vous avez modifié (vous y avez ajouté un fichier lors de la précédente étape)
- créer (commit) une nouvelle image à partir de votre container
- se log au docker hub
- envoyer votre image sur le hub
- vérifier sur hub.docker.com que votre image est présente

Test final

Dans le terminal 2

- arrêter et retirer votre container
- retirer l'image alpine et votre alpine-esgi de la liste des images sur votre ordinateur
- démarrer un nouveau container alpine-esgi (qui va donc être récupérer du hub) en lançant la commande de votre choix (/bin/sh, cat votre_fichier, …)