docker-build

Construire une image depuis le code source

Cette commande lit un Dockerfile depuis le répertoire spécifié. Il a besoin également d'autres fichiers et répertoires trouvés dans le répertoire courant du service docker. Le contenu de ce répertoire sera utilisé par les commandes AD dans le Dockerfile.

Note, cela envoie beaucoup de données au service Docker en fonction du contenu du répertoire courant. La construction est lancé par le service Docker, par par le client, donc tout le contexte doit être transféré au service. Le client affiche "Sending Build context to Docker daemon" quand le contexte est envoyé au service.

Quand l'URL d'une archive tar ou un simple Dockerfile est donné, aucun contexte est envoyé au service. Dans ce cas, le Dockerfile à la racine de l'archive et le reste de l'archive seront utilisé comme contenu de construction. Dans un dépôt git est utilisé comme url, le dépôt est cloné localement et envoyé comme contexte.

OPTIONS

-f, –file=PATH/Dockerfile Chemin du fichier Dockerfile à utiliser.

–build-arg=variable Nom et valeur d'un buildarg. Docker les utilise comme contexte d'environnement via l'instruction RUN du dockerfile.

–force-rm=true|false Supprime les conteneurs intermédiaires, même après des constructions échouées. défaut : false

–isolation="default" spécifique le type de technologie d'isolation utilisée par les conteneurs

–no-cache=true|false Ne pas utiliser de cache pour construire l'image. Défaut : false

–pull-true|false Tente de récupérer une version plus récente de l'image. défaut : false

-q, –quiet=true|false Supprime la sortie de la construction et affiche l'ID de l'image en cas de succès. Défaut : false

–rm=true|false Supprime les conteneurs intermédiaire en cas de réussite. Défaut : true.

-t, –tag="" Noms de dépôts (et optionnellement avec tag) à appliquer à l'image résultante en cas de succès.

-m, –memory=MEMORY Limite mémoire

–memory-swap="LIMIT" Une valeur limite égale à la mémoire plus le swap. Doit être utilisé avec -m.

–shm-size="" Taille de /dev/shm. format : <number><unit>

–cpu-shares=0 Partage CPU

–cpu-period=0 Limite la période CFS

–cpu-quota=0 Lisite le quota CPU CFS

–cpuset-mems="" Nœuds mémoire autorisés.

–cpuset-cpus=CPUSET-CPUS CPU dans lesquels autoriser l'exécution

–cgroup-parent="" Chemin des cgroups sous lesquels créer le conteneur. Si le chemin n'est pas absolu, le chemin est relatif aux cgroups du processus init.

–ulimit=[] Options ulimit

-v|–volume[=[[HOST-DIR:]CONTAINER-DIR[:OPTIONS]]]

Exemples

Construire une image en utilisant un Dockerfile dans le répertoire courant :

docker build .
Construire une image et nommer cette image
docker build -t myimage .
Une meilleur approche est de nommer le répertoire, nom et tag :
docker build -t fedora/jboss :1.0
Construire une image avec une URL :
docker build github.com/scollier/purpletest
Construire une image en utilisant une URL dans un contexte tar
docker build -f dev/Dockerfile https ://10.10.10.1/docker/context.tar.gz