Dockerfile Best Pratices

ou comment optimiser ses images Docker

Constitution d’une image (layers)

- Une image est constituée d’un ou plusieurs layers, chaque layer est en cache est peut être réutilisé dans une autre image (gain de place)
  - ➢ docker history
- La taille d’un layer dépend des modifications apportées : il stocke la différence entre deux états
- Instruction / commit ↔ Layer
- Limite de 127 layers

Comment créer une image ?

- Dockerfile
- Modifier un container et le commiter (Update & Commit)

Dockerfile

- vérifier image de base : éviter les images trop complètes, elles contiennent des services inutiles
  - BuzyBox au lieu de Ubutu (-190MB)
  - Debian au lieu de Ubuntu (-100MB)
  - Fedora
  - CentOS

Dockerfile

```
RUN apt-get update && apt-get install -y ruby
ruby-dev
RUN gem install sinatra
```

devient

```
RUN apt-get update && apt-get install -y ruby
ruby-dev && gem install sinatra
```

- vérifier image de base : éviter les images trop complètes, elles contiennent des services inutiles

- Limiter le nombre de layers (instructions)
  - Consolider les instructions similaires
  - Le layer peut être réutilisé
  - Lors d'un ADD ou COPY les checksum et headers des fichiers sont utilisés pour vérifier l'existence en cache
  - Lors d'un "RUN apt-get" : pas de comparaison des fichiers téléchargés, uniquement sur la commande

Dockerfile

- vérifier image de base : éviter les images trop complètes, elles contiennent des services inutiles
- Limiter le nombre de layers (instructions)
- Différence entre CMD et ENTRYPOINT
  - CMD : utilisé quand aucun paramètre “docker run”
  - ENTRYPOINT : toujours exécuté
  - CMD + ENTRYPOINT : les arguments CMD sont envoyés à ENTRYPOINT

Dockerfile

- vérifier image de base : éviter les images trop complètes, elles contiennent des services inutiles
- Limiter le nombre de layers (instructions)
- Différence entre CMD et ENTRYPOINT
- Utiliser .dockerignore ou placer Dockerfile dans un dossier vide
  - ➢ le contexte du Dockerfile est envoyé au daemon : peut ralentir le build

Update & Commit

- Image de départ requise
- Facile à faire
- Difficile à maintenir

Update & Commit

- Image de départ requise
- Facile à faire
- Difficile à maintenir
- Image de départ concise
  - nettoyer le cache package

```
FROM fedora
RUN dnf install -y mariadb
RUN dnf install -y wordpress
RUN dnf clean all
```

537.7 MB

```
FROM fedora
RUN dnf install -y mariadb
wordpress && dnf clean all
```

377.9 MB

Update & Commit

- Image de départ requise
- Facile à faire
- Difficile à maintenir
- Image de départ concise
  - nettoyer le cache package
  - enlever les packages inutiles

Update & Commit

- Image de départ requise
- Facile à faire
- Difficile à maintenir
- Image de départ concise
  - nettoyer le cache package
  - enlever les packages inutiles
  - ne pas installer la doc des packages (si permis par le manager)

Update & Commit

- Image de départ requise
- Facile à faire
- Difficile à maintenir
- Image de départ concise
  - nettoyer le cache package
  - enlever les packages inutiles
  - ne pas installer la doc des packages (si permis par le manager)
- Squashing layers
  - build plus rapide car moins de layers

Update & Commit

- Image de départ requise
- Facile à faire
- Difficile à maintenir
- Image de départ concise
  - nettoyer le cache package
  - enlever les packages inutiles
  - ne pas installer la doc des packages (si permis par le manager)
- Squashing layers
  - build plus rapide car moins de layers
  - compression de la taille image

Update & Commit

- Image de départ requise
- Facile à faire
- Difficile à maintenir
- Image de départ concise
  - nettoyer le cache package
  - enlever les packages inutiles
  - ne pas installer la doc des packages (si permis par le manager)
- Squashing layers
  - build plus rapide car moins de layers
  - compression de la taille image
  - organiser les layers de son image de manière logique

Squashing layers

- docker export / import : squash de tous les layers en un seul
  - image plus petite
  - pas de gestion de cache au build
  - pas d'history

```
docker export container_id | docker import -
```

Commandes utiles

- supprime les images inutilisées

  docker rmi $(docker images -q -f dangling=true)

- démarre un container, lance une commande et supprime le container

  docker run --rm -ti image_name command

- supprime les containers stoppés

  rm $(docker ps -a -q)