04-docker

August 21, 2020

1 Docker

1.1 Conteneur

Technologie logicielle datant de 2013 permettant d'éxécuter un système d'exploitation isolé et automatisé dans un autre système. La technique est proche de la virtualisation mais sans l'inconvénient de la latence.

Un conteneur permet également de sauvegarder l'état d'un ordinateur à un instant donné et de pouvoir rédemmarrer de cet état à tout moment.

1.2 Points forts

- Installation très simple `https://www.docker.com/get-started`
- Ligne de commande facile à utiliser (`docker help`)
- Beaucoup de possibilités (hébergement web, cloud, intégration continue, …)

1.3 Docker en 1 ligne

Pour vérifier que docker est bien installé

```
docker run helloworld
```

```
docker run --rm -it ubuntu /bin/bash
```

- `run` : on veut lancer le conteneur
- `-it` : on veut un terminal et être interactif avec lui
- `ubuntu` : l'image à utiliser pour ce conteneur
- `/bin/bash` : commande éxécutée au démarrage qui permet l'interactivité

1.4 Les étapes du démarrage d'un image docker

- Recherche de l'image -> Si l'image n'existe pas en local, alors téléchargement via le hub. Construction du système de fichiers.
- Démarrage du container
- Configuration de l'adresse IP du container -> Ainsi que de la communication entre l'extérieur et le conteneur
- Capture des messages entrées-sorties

1.5 Principales commandes sur linux

```
sudo /usr/bin/docker -d & # run the daemon
sudo docker search ubuntu # give ubuntu images from public index
sudo docker pull ubuntu # pull latest ubuntu images
sudo docker history ubuntu # view history for this images
sudo docker images # show local images
docker ps # show active containers
sudo docker logs ubuntu
sudo docker attach ubuntu # retake the hand on the container
sudo docker run -d -p 8888:80 ubuntu # export 8888 on master
sudo docker stop # SIGTERM suivi d'un SIGKILL
sudo docker kill # SIGKILL directement
```

1.6 Création d'images Docker

Il est possible de créer une image docker en utilisant des commandes en ligne mais préférez la méthode du fichier Dockerfile qui vous permettra de conserver les etapes et une trace de la procédure.

1.6.1 Premier Dockerfile

```
FROM rocker/shiny

FROM rocker/r-ver:3.6.3

RUN apt-get update && apt-get install -y \
    sudo \
    gdebi-core \
    pandoc \
    pandoc-citeproc \
    libcurl4-gnutls-dev \
    libcairo2-dev \
    libxt-dev \
    xtail \
    wget


# Download and install shiny server
RUN wget --no-verbose https://download3.rstudio.org/ubuntu-14.04/x86_64/VERSION -O "version.tx
    VERSION=$(cat version.txt)  && \
    wget --no-verbose "https://download3.rstudio.org/ubuntu-14.04/x86_64/shiny-server-$VERSION-
    gdebi -n ss-latest.deb && \
    rm -f version.txt ss-latest.deb && \
    . /etc/environment && \
    R -e "install.packages(c('shiny', 'rmarkdown'), repos='$MRAN')" && \
    cp -R /usr/local/lib/R/site-library/shiny/examples/* /srv/shiny-server/ && \
    chown shiny:shiny /var/lib/shiny-server

EXPOSE 3838
```

```
COPY shiny-server.sh /usr/bin/shiny-server.sh

CMD ["/usr/bin/shiny-server.sh"]
```

1.7 Instructions

Les instructions sont peu nombreuses : FROM, RUN, CMD, LABEL, MAINTAINER, EXPOSE, ENV, ADD, COPY, ENTRYPOINT , VOLUME, USER, WORKDIR, ARG, ONBUILD, STOPSIGNAL, HEALTHCHECK, SHELL

- Pour chaque instruction RUN, un conteneur temporaire (8xxxxxxxx) est créé depuis l'image de base.
- La commande RUN est exécutée dans ce conteneur,
- Le conteneur est commité en une image intermédiaire (7yyyyyyyy),
- Le conteneur intermédiaire (8xxxxxxxx) est supprim´e, Le résultat, l'image intermédiaire, servira d'image de base pour l'´etape suivante, etc..

1.8 Docker Hub

Site public permettant de sauvegarder et partager ses images docker:

- Déposer son fichier Dockerfile sur un dépôt github
- Créer une image docker sur https://hub.docker.com et lier cette image au dépôt github
- Chaque modification sur le dépôt va mettre à jour l'image

1.9 Exercice

- Creez un dépôt github personnel et déposer le ficher Dockerfile

[ ]: