Sébastien Masset

Franck Pagès



Déploiement projet Docker – Haute disponibilité



# **Création des dossiers et des Dockerfiles afin de build nos images**

1. Images NGINX 

Contenu du Dockerfile :

FROM debian:wheezy

MAINTAINER m.sebastien.gm@gmail.com

RUN apt-get update && apt-get install -y nano wget dialog net-tools procps inetutils-telnet curl

RUN mkdir /app

RUN apt-get install -y nginx-full

RUN rm -v /etc/nginx/nginx.conf

RUN mkdir -p /app/conf /app/docs

ADD index.html /app/docs/

ADD nginx.conf /app/conf/

EXPOSE 80

CMD nginx -c /app/conf/nginx.conf

1. Images php-fpm1 

Afin de pouvoir lancer nos conteneurs php-fpm1 nous devons créer les 4 fichiers suivants :

Contenu du Dockerfile :

FROM debian:wheezy

MAINTAINER fpages26@gmail.com@gmail.com

RUN apt-get update && apt-get install -y nano wget dialog net-tools procps inetutils-telnet vim curl

RUN mkdir /app

RUN apt-get install -y php5-fpm

ADD php.ini /etc/php5/fpm/

ADD php-fpm.conf /etc/php5/fpm/

ADD www.conf /etc/php5/fpm/pool.d/

ADD app /app

EXPOSE 9000

CMD ["/usr/sbin/php5-fpm"]

Création du fichier php.ini avec configuration standard.

Création du fichier php-fpm.conf avec configuration standard.

Création du dernier fichier de configuration www.conf.

Création du dossier app

1. Images MySQL 

Afin de pouvoir lancer nos conteneurs MySQL nous devons créer les 3 fichiers suivants :

Contenu du Docker file :

FROM debian:wheezy

MAINTAINER fpages26@gmail.com

RUN apt-get update && apt-get install -y nano wget dialog net-tools procps inetutils-telnet vim curl

RUN mkdir /app

RUN apt-get install -y mysql-server

RUN mkdir -p /app/conf /app/data

ADD mysql.conf /app/conf/

ADD entrypoint.sh /usr/local/bin/entrypoint.sh

WORKDIR /usr

EXPOSE 3306

ENTRYPOINT ["/usr/local/bin/entrypoint.sh"]

Création du fichier mysql.conf avec configuration standard.

Création du fichier entrypoint.sh

## Busybox pour le data MySQL

Contenu du Dockerfile :

FROM busybox

VOLUME /var/lib/mysql

CMD /bin/bash

//////////////

//docker run -d -volumes-from $mysql\_container data/latest

////////// ///

git pull github.com/pojkofd00m/EPSI\_DOCKER

cd /vagrant

cd base

docker BUILD –t =”base/latest” ./

cd ../data

docker BUILD –t=”data/latest” ./

cd ../mysql

docker BUILD –t=”mysql/latest” ./

cd ../nginx

docker BUILD –t=”nginx/latest” ./

cd ../php-fpm1

docker BUILD –t=”php-fpm/latest” ./

cd ../php-fpm2

docker BUILD –t=”php-fpm2/latest” ./

cd ../

docker images

docker run –d mysql/latest

docker run -d -volumes-from mysql/latest data/latest

docker run –d php-fpm1/latest

docker run –d php-fpm2/latest

docker run –d –p 80:80 nginx/latest

docker ps

# Build des conteneurs

1. Build de l’image nginx puis attribution d’un tag

Afin de build l’image nginx nous devons tout d’abord se placer dans le dossier contenant le Dockerfile : 

Puis nous lançons la commande pour build l’image et on lui donne un tag : 

La machine docker exécute les commandes passé dans le Dockerfile et installe donc nginx via apt-get

1. Build de l’image php-fpm puis attribution d’un tag

Idem, nous devons nous placer dans le dossier ou se situe le Dockerfile :

1. Build de l’image php-fpm
2. Build de l’image MySql