Apache dan PHP di dalam Image (Docker)

Husni <husni@if.trunojoyo.ac.id>

Ada cukup banyak referensi bagaimana menjalankan Apache dan PHP di atas Docker. Sayangnya, jarang langsung mendapatkan apa yang diharapkan, perlu penyesuaian agar Web server apache + PHP berjalan baik sebagai container di dalam Docker. Tutorial ini memberikan langkah-langkah pasti untuk membuat Image Docker yang berisi Web Server apache + PHP dan telah dicoba dapat berjalan dengan baik saat dijadikan container.

Berikut ini adalah langkah-langkah yang dimaksud (instalasi dan konfigurasi semua software yang dibutuhkan didasarkan pada Linux distro Ubuntu 14.04.2 LTS):

1. Pastikan Docker telah terinstall. Jika belum, berikan perintah berikut di console (command prompt):

   ```
   sudo apt-get install docker.io
   ```

2. Buat direktori untuk menyimpan semua file yang diperlukan selama pembuatan Image dan eksekusi image menjadi container, misalnya direktori /home/username/.webserver. Adanya titik (.) diawal nama direktori, secara default, menyebabkan direktori tersebut tidak terlihat pada file manager atau di console (perintah ls).

3. Masuk (dengan perintah cd) ke dalam direktori yang telah dibuat di atas (~/.webserver).

4. Buat file bernama Dockerfile. Isinya adalah sebagai berikut:

```
# Dockerfile untuk Web server apache + PHP

# Image diturunkan dari Ubuntu LTS
FROM ubuntu:14.04.2

MAINTAINER Husni <husni@if.trunojoyo.ac.id>

#U pdate database paket dan instal semua yang diperlukan
# curl dan lynx-cur digunakan untuk debugging container
# Enable apache mods.
# Update file PHP.ini , enable-kan tag <? ?> dan quieten logging.
RUN apt-get update \
 && DEBIAN_FRONTEND=noninteractive \
 apt-get -y install apache2 \
 libapache2-mod-php5 \
 php5-mysql \
 php5-gd \
 php-pear \
 php-apc \
 php5-curl \
 curl \
 lynx-cur && \
 a2enmod php5 && \
 a2enmod rewrite && \
 sed -i "s/short_open_tag = Off/short_open_tag = On/" /etc/php5/apache2/php.ini && \
 sed -i "s/error_reporting = .*$/error_reporting = E_ERROR | \
 E_WARNING | E_PARSE/" /etc/php5/apache2/php.ini
```

```
# Setup variabel lingkungan dari apache
ENV APACHE_RUN_USER www-data
ENV APACHE_RUN_GROUP www-data
ENV APACHE_LOG_DIR /var/log/apache2
ENV APACHE_LOCK_DIR /var/lock/apache2
ENV APACHE_PID_FILE /var/run/apache2.pid

# Port 80 dipublish ke Host
EXPOSE 80

# Tambahkan  halaman web (aplikasi) default ke /var/www/
ADD webdata/ /var/www/

# Update situs apache default dengan konfigurasi yang kita buat.
ADD apache-config.conf /etc/apache2/sites-enabled/000-default.conf

# Secara default, jalankan apache.
CMD /usr/sbin/apache2ctl -D FOREGROUND
```

5. Buat file konfigurasi apache khusus (digunakan sebagai default, mengganti 000-default.conf). Isinya sebagai berikut:

```
# apache-config.conf
<VirtualHost *:80>
 ServerAdmin admin@nama_domain.com
 DocumentRoot /var/www/

 <Directory /var/www/>
    Options Indexes FollowSymLinks MultiViews
    AllowOverride All
    Order deny,allow
    Allow from all
 </Directory>

 ErrorLog ${APACHE_LOG_DIR}/error.log
 CustomLog ${APACHE_LOG_DIR}/access.log combined

</VirtualHost>
```

6. Buat direktori baru di dalam direktori kerja, misalnya "~/.webserver/webdata" yang akan digunakan sebagai direktori web semua file/aplikasi web yang dibuat. Di dalam diektori ini, setidaknya buat sebuah file bernama index.php yang isinya:

```
<? echo "<p>Hello?</p>"; ?>
```

7. Buat image, misalnya bernama "webserver2015" menggunakan perintah docker build:

```
docker build -t webserver2015 .
```

8. Jalankan image tersebut menjadi container. Web server biasanya berjalan di belakang layar sebagai daemon. agar dapat diakses dari host atau komputer lain, maka port 80 dari container (dimana web server berjalan) harus dipetakan ke port tertentu pada Host, misalnya 8080.

Berikut ini contoh menjalankan image webserver2015 di atas:

```
docker run -p 8080:80 -d webserver2015
```

Keterangan:

- -p 8080:80 mempublikasikan port 80 pada container ke port 8080 pada mesin host (dimana Docker berjalan).
- -d melepaskan diri dari proses berjalannya container, gunakan docker ps untuk mendapatkan ID atau nama dari proses yang berjalan dan docker stop ID untuk menghentikannya.

Catatan:

- Kadang kala kita ingin men-debug dari container; mungkin ada masalah konfigurasi PHP atau ingin menampilkan catatan (log) error. Ini dapat dilakukan dengan memulai containerdalam modus interaktif dan menjalankan apache secara manual:

```
docker run -i -t -p 8080:80 mysite /bin/bash

apachectl start
```

- Perubahan halaman web atau aplikasi? Ini mungkin terjadi. Bagaimana agar update aplikasi PHP berpengaruh terhadap container yang berjalan tanpa harus membangun ulang image? Perintah docker run mempunyai flag -v yang memungkinkan kita me-mount suatu direktori pada mesin host ke dalam container:

  ```
  docker run -p 8080:80 -d -v ~/.webserver/webdata:/var/www --name webserverku webserver2015
  ```

  Sekarang, apapun perubahan yang dilakukan di dalam direktori ~/.webserver/webdata (pada host) akan mengupdate isi direktori /var/www di dalam container.

selamat mencoba...semoga bermanfaat.