Introducción a Docker  
UD 04. Caso práctico 03 - Apache 2 con PHP desde Alpine

short line

Autor: Sergi García Barea

Actualizado Marzo 2022

Licencia

**Reconocimiento – NoComercial - CompartirIgual (BY-NC-SA)**: No se permite un uso comercial de la obra original ni de las posibles obras derivadas, la distribución de las cuales se debe hacer con una licencia igual a la que regula la obra original.

Nomenclatura

A lo largo de este tema se utilizarán distintos símbolos para distinguir elementos importantes dentro del contenido. Estos símbolos son:

📖 **Importante**

❕ **Atención**

💬 **Interesante**

[**Introducción**](#_dbh0n1vac4c8) **3**

[**Descargando la aplicación**](#_qbo60tdb3ooa) **3**

[**Preparando el Dockerfile y creando la imagen**](#_vyhbfp4t666x) **3**

[**Probando la imagen**](#_14wpn6n5xh6b) **4**

[**Bibliografía**](#_72yv1gty1dke) **4**

UD04. Caso práctico 03

# Introducción

En este caso práctico vamos a crear una imagen que partiendo de la imagen base “alpine”, una de las imágenes más reconocidas como “imagen ligera”, le instalaremos Apache 2, PHP y pondremos un simple ***<?php phpinfo(); ?>*** obtenido desde un fichero de Internet.

# Descargando la aplicación

Antes de empezar, debemos crear un directorio (por ejemplo, “Caso4-3”) y en el descargar la sencilla aplicación con la aplicación que ejecuta el comando ***<?php phpinfo(); ?>*** que encontraremos tanto en un fichero comprimido “zip” de este curso como en <https://gist.github.com/SyntaxC4/5648247>

# Preparando el Dockerfile y creando la imagen

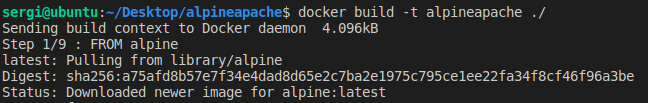
En el directorio creando anteriormente, o crearemos (o en el caso del “zip” del curso, ya tendremos listo) el siguiente fichero “***Dockerfile***”:

| ***FROM*** *alpine* ***MAINTAINER*** *Sergi <sergi.profesor@gmail.com> #Actualizamos e instalamos paquetes con APK para Alpine* ***RUN*** *apk update && apk add apache2 php php-apache2 openrc tar #Copiamos script para lanzar Apache 2* ***ADD*** *./start.sh /start.sh #Descargamos un ejemplo de <?php phpinfo(); ?> por enseñar como bajar algo de Internet #Podría haber sido simplemente #RUN echo "<?php phpinfo(); ?>" > /var/www/localhost/htdocs/index.php* ***ADD*** *https://gist.githubusercontent.com/SyntaxC4/5648247/raw/94277156638f9c309f2e36e19bff378ba7364907/info.php /var/www/localhost/htdocs/index.php   # Si quisiéramos algo como Wordpress haríamos #ADD http://wordpress.org/latest.tar.gz /var/www/localhost/htdocs/wordpress.tar.gz #RUN tar xvzf /var/www/localhost/htdocs/wordpress.tar.gz && rm -rf /var/www/localhost/htdocs/wordpress.tar.gz  # Usamos usuario y grupo www-data. El grupo lo crea Apache, pero si quisiéramos crear grupo # Grupo www-data RUN set -x && addgroup -g 82 -S www-data # Creamos usuario www-data y lo añadimos a ese grupo* ***RUN*** *adduser -u 82 -D -S -G www-data www-data # Hacemos todos los ficheros de /var/www propiedad de www-data # Y damos permisos s esos ficheros y a start.sh* ***RUN*** *chown -R www-data:www-data /var/www/ && chmod -R 775 /var/www/ && chmod 755 /start.sh #Indicamos puerto a exponer (para otros contenedores) 80* ***EXPOSE*** *80 #Comando lanzado por defecto al instalar el contendor* ***CMD*** */start.sh* |
| --- |

El funcionamiento del propio “Dockerfile” está definido por sus propios comentarios. Una vez preparado, crearemos la imagen con:

| docker build -t alpineapache ./ |
| --- |

Con esa línea indicamos que creamos la imagen “***alpineapache***” basándose en el fichero “***Dockerfile***” del directorio actual.



# Probando la imagen

Con el siguiente comando, podremos lanzar la aplicación en el puerto 80 de nuestra máquina:

| docker run -dp 80:80 alpineapache |
| --- |

Una vez hecho esto, podremos probar la aplicación accediendo a <http://localhost/index.html> y <http://localhost/index.php>.

# Bibliografía

[1] Docker Docs <https://docs.docker.com/>