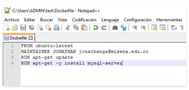
Cómo automatizar el proceso de construcción y configuración de una imagen utilizando un archivo Dockerfile

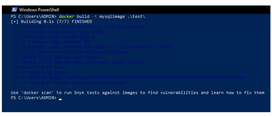
Este PDF muestra cómo automatizar el proceso de construcción y configuración de una imagen utilizando un archivo DockerFile en lugar de ejecutar comandos manualmente.

Para este caso se creará una imagen con sistema operativo ubuntu, con un servidor apache funcionando por el puerto 80. En la siguiente imagen se muestra el código de ejemplo usado en el Dockerfile. En el caso de Windows puede utilizar el editor de texto que considere más conveniente pero deberá tener en cuenta la ubicación del archivo, este deberá llamarse Dockerfile y adicionalmente este archivo no puede tener ninguna extensión.

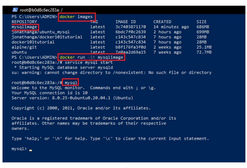


Se puede observar que la imagen se construirá con la última versión del sistema operativo ubuntu, en segundo lugar se indica el nombre y correo electrónico de la persona que genera la imagen nueva, en tercer lugar se ejecuta el comando **apt-get update** para actualizar los repositorios del sistema operativo ubuntu de la imagen, y finalmente se ejecuta el comando **apt-get -y install mysql-server** para ejecutar en la imagen la instalación de mysql, automatizando así todos los pasos ejecutados anteriormente pero desde un archivo Dockerfile.

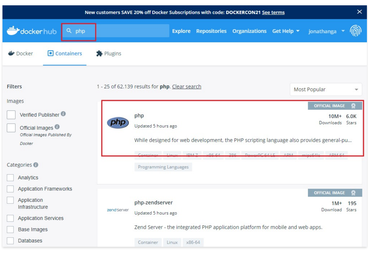
Para ejecutar el archivo Dockerfile se ejecuta el siguiente comando:
docker build -t nombreNuevaImagen ./directorioContenedor/



Donde **nombreNuevaImagen** corresponde al nombre de la nueva imagen a generar a partir de las sentencias del archivo Dockerfile y **./directorioContenedor/** corresponde al directorio de la máquina host donde se encuentra el archivo Dockerfile.



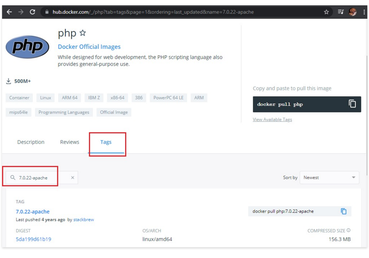
El proceso de construcción de la imagen de acuerdo con las sentencias definidas en el dockerfile se realiza de forma lineal de arriba hacia abajo, docker divide4 esta construcción en pasos dependiendo al número de comandos definidos en el archivo Dockerfile, el tamaño final de la imagen, la cantidad de comandos a ejecutar en la imagen construida y la velocidad de conexión a internet definen el tiempo que se puede demorar el sistema en la construcción de la imagen.



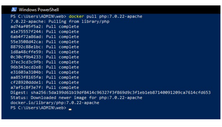
Así como se pueden crear imágenes propias iniciando desde lo básico e instalando de acuerdo a las necesidades, es posible encontrar imágenes públicas con todos los elementos ya pre-configurados listos para la creación de nuevos contenedores.

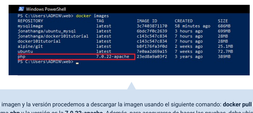
Para el siguiente ejemplo se ejecutará un servidor web que permita archivos de extensión php usando un contenedor a partir de una imagen publicada en Docker Hub.

Para esto debe ingresar a Docker hub, buscar php y verificar los detalles de la imagen oficial.



Se puede verificar en la pestaña de tags todas las versiones disponibles de imágenes de php, en este caso requerimos una versión específica de php y el servidor apache.





Una vez seleccionada la imagen y la versión procedemos a descargar la imagen usando el siguiente comando: **docker pull image:version**, para el ejemplo la imagen se llama **php** y la versión es la **7.0.22-apache**. Además, para asegurarse de hacer las pruebas, debe ubicar la terminal de ejecución de docker sobre un directorio donde contendrá los archivos a publicar en el servidor que se va a montar desde el contenedor.

A continuación, se creará el contenedor desde la imagen teniendo en cuenta que en el contenedor el servidor de php estará escuchando solicitudes desde el puerto 80 y debo indicar qué puerto en la máquina local será asignado para realizar el proceso de escucha en la ejecución del contenedor, para esto se usa el comando de ejecución del contenedor con la especificación de los puertos de la siguiente forma:

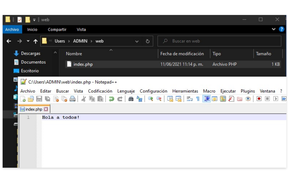
docker run -p puertolocal: puertoContenedor

Donde **puertolocal** corresponde al puerto donde escuchará el servicio en mi máquina local y **puertoContenedor** corresponde al puerto con el que se equipará en el contenedor.

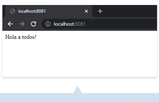
Como este contenedor va a ser utilizado para ir probando cambios en los archivos php que se van generando en el equipo local, es necesario indicar a docker para que haga un mapeo del directorio en la máquina local con el directorio donde se deben alojar los archivos que deben ser ejecutados por apache en el contenedor con lo cual el comando de ejecución puede tomar la siguiente forma:

docker run -p puertolocal:puertoContenedor -v directorioMaquinaLocal:DirectorioEnContenedor Imagen

Donde -v es la directriz para hacer mapeo y **directorioMaquinaLocal** corresponde a una ruta donde se aloja mi directorio de desarrollo local y **DirectorioEnContenedor** representa el directorio donde será mapeado toda la información en el contenedor. **Imagen** corresponde al nombre de la imagen previamente descargada. También se puede agregar la bandera -d para hacer que el contenedor que se ejecute lo haga en segundo plano.



En la siguiente imagen se puede observar que la terminal de comandos está ubicada sobre un directorio llamado **web** el cual corresponde con la ruta que se indica en el mapeo **C:\Users\ADMN\web** y el directorio mapeado del lado del contenedor corresponde a **/var/www/html** el cual es el sitio que utilizara el servidor Apache para la publicación de aplicaciones web.



Con este mapeo se logra que todos los cambios que haga en el directorio de la máquina local se verá publicado en el servidor de apache que está siendo ejecutado por el contenedor y que también está mapeado al puerto local 8081 en mi máquina local.