Tomcat / MariaDB / Docker V2

En este documento vamos a describir cómo desplegar una instancia de MariaDB y una de Tomcat en un equipo, cómo desplegar una aplicación, y cómo arrancar y parar los contenedores.

Todos los ficheros necesarios están en el fichero “practica2_material.zip”. Debemos descomprimir el contenido de ese fichero en un directorio, que utilizaremos como directorio para configurar nuestro despliegue. No será el directorio donde desplegaremos la aplicación ni la base de datos, sino un directorio “fuente”, donde tendremos todos los scripts que nos permitirán realizar ese despliegue.

1.- Desplegar MariaDB

Para desplegar la base de datos, basta con que, desde el directorio de ficheros de despliegue, ejecutemos el script “create_mariadb.sh”

```
#!/bin/bash

PORTAL_PATH=$HOME/sisinf
DIR="$( cd "$( dirname "${BASH_SOURCE[0]}" )" >/dev/null 2>&1 && pwd )"

if [ ! -d $PORTAL_PATH ]; then
    mkdir $PORTAL_PATH
    chmod 755 $PORTAL_PATH
fi

if [ ! -d $PORTAL_PATH/mysql ]; then
    mkdir $PORTAL_PATH/mysql
    chmod 777 $PORTAL_PATH/mysql
fi

docker stop sisinf-mariadb
docker rm sisinf-mariadb

docker run -d --name sisinf-mariadb \
    -e MARIADB_ROOT_PASSWORD=sisinf \
    -e MARIADB_DATABASE=sisinf  \
    -v $PORTAL_PATH/mysql:/bitnami/mariadb \
    -p 3306:3306 \
    bitnami/mariadb:latest
```

¿Qué hace este script?

En primer lugar, declaramos la ruta hacia el directorio donde vamos a crear nuestra base de datos. Por defecto, es $HOME/sisinf, aunque podemos cambiar este directorio al que queramos. El script creará ese directorio, si no existe, y creará un directorio “mysql” bajo él.

A continuación, para (docker stop) y borra (docker rm) cualquier contenedor previo que tuviéramos creado previamente, y crea un nuevo contenedor (docker run) para la base de datos.

-d => ejecutamos el contenedor en modo “detached”. Es algo similar a cuando ejecutamos un script en linux, y ponemos al final “&”. Lo lanzamos en otro proceso autónomo, que continuará ejecutándose en background.

-name => Nombre que tendrá nuestro contenedor

-e MARIADB_ROOT_PASSWORD => variable de entorno del contenedor, para definir el password del usuario *root* de nuestra base de datos

-e MARIADB_DATABASE => le indicamos al contenedor que, tras arrancar, cree automáticamente una base de datos llamada así (sisinf en nuestro ejemplo).

- v $PORTAL_PATH/mysql:/bitnami/mariadb => Con esta opción, mapeamos un directorio de nuestro sistema local con un directorio del sistema de ficheros del contenedor. Es una forma de "compartir" una carpeta entre nuestro sistema operativo y el contenedor. En este caso, el directorio compartido es la ubicación de los ficheros de la base de datos, tanto los datos propiamente dichos, como los ficheros de configuración de la base de datos. Esto nos permite, entre otras cosas, no perder los datos si decidimos recrear el contenedor (por ejemplo, porque instalamos una versión nueva).

-p 3306:3306 => Hacemos un NAT entre el puerto 3306 de nuestro sistema operativo, y el puerto 3306 del contenedor. Eso supone que si llamamos a localhost:3306, nos responderá el contenedor (es decir, el gestor de mariadb). Si no hacemos esta configuración, para conectarnos a la base de datos desde nuestro sistema, tendríamos que llamar a la IP del contenedor, y desde el exterior de nuestro sistema, no sería posible llegar. Eso podría dar mayor seguridad al sistema, pero de momento, vamos a dejarlo así por comodidad. Estamos aún creando un entorno de desarrollo.

bitnami/mariadb:latest => Utilizamos para crear el contenedor la última versión disponible de la imagen docker "bitnami/mariadb", es decir, la imagen de mariadb desarrollada por bitnami, y desplegada en DockerHub.

Una vez ejecutado el script, deberíamos tener ya la base de datos funcionando. Una manera de comprobarlo es ejecutar:

```
$ sudo docker ps
CONTAINER ID        IMAGE                  COMMAND                  CREATED             STATUS              PORTS                    NAMES
3ecdfbdb8d61        bitnami/mariadb:latest   "/entrypoint.sh /run…"   6 days ago          Up 2 seconds        0.0.0.0:3306->3306/tcp   sisinf-mariadb
```

Si no vemos el proceso arrancado, ha habido algún error. Para comprobar los errores, podemos ejecutar:

```
$ sudo docker logs sisinf-mariadb
```

Tal como hemos configurado el contenedor, cada vez que reiniciemos el equipo, el contenedor estará parado (aunque está creado). Para arrancarlo basta con que ejecutemos:

```
$ sudo docker start sisinf-mariadb
```

Tomcat

Una vez que tenemos arrancada la base de datos, vamos a desplegar la aplicación, dentro de un contenedor Tomcat.

Para ello, seguiremos los siguientes pasos:

1.- Nos aseguramos de que en la carpeta *tomcat* de nuestra carpeta de scripts de despliegue, tenemos todos los ficheros necesarios:

- context.xml (definición de contexto del servidor. Contiene la referencia al datasource de la base de datos)

- server.xml (configuración del server de Tomcat. Contiene los parámetros del datasource de la base de datos)

- tomcat-users.xml (configuración de usuarios de Tomcat, por si queremos utilizar las herramientas de administración de Tomcat desde interfaz web)

- mysql-connector-java-5.1.40-bin.jar (driver JDBC de MaríaDB y MySQL). Si vamos a utilizar otro gestor de base de datos, como Postgres u Oracle, debemos incluir en esta carpeta el driver correspondiente.

- Dockerfile (fichero con la especificación para construir nuestra imagen Docker)

- xxx.war (fichero WAR con nuestra aplicación web)

2.- Ejecutamos el script *create_tomcat_v2.sh.*

```
#!/bin/bash

# Paramos y borramos la versión anterior del portal, si existe
docker stop sisinf-tomcat
docker rm sisinf-tomcat

cd tomcat
sudo docker build -t sisinf/tomcat:latest .

docker run -d --name sisinf-tomcat \
    --link sisinf-mariadb \
    -p 8080:8080 \
    sisinf/tomcat:latest
```

Este script ejecuta unas operaciones muy sencillas:

- Detiene y elimina el contenedor tomcat previo (sisinf-tomcat), si es que existe.

- Entra en el directorio *tomcat*, y crea una nueva imagen docker, a la que llama *sisinf/tomcat:latest*.

- Arranca un contenedor utilizando esa imagen *sisinf/tomcat:latest*. Los demás parámetros significan lo siguiente:

* --link sisinf-mariadb => crea una entrada en el /etc/hosts del contenedor tomcat, apuntando al contenedor sisinf-mariadb. De esa manera, nuestra aplicación será capaz de conectarse a la base de

datos con una URL que apunte a *sisinf/mariadb:3306*, sin preocuparnos de qué IP tiene el contenedor de la base de datos.

- p 8080:8080 => Al igual que con la base de datos, creamos un NAT entre los puertos 8080 del contenedor y del sistema anfitrión. De esa manera, podemos conectarnos a Tomcat llamando a http://localhost:8080 o a http://<ip del equipo>:8080

Desplegar una nueva versión de la aplicación

Para desplegar una nueva versión de nuestra aplicación, basta que hagamos lo siguiente:

1.- Copiar el WAR de la nueva versión de la aplicación en el subdirectorio *tomcat* de nuestro directorio de scripts de despliegue.

2.- Ejecutar *create_tomcat_v2.sh*. Eso eliminará la versión anterior, y creará un nuevo contenedor con la nueva versión de la aplicación en cuestión de segundos.