Tecnologías Específicas en la Ingeniería Informática - Curso 2021/2022

Prácticas de Contenedores con Docker

Grado Ingeniería Informática

Universidad de Murcia

Introducción a los Contenedores con Docker

Listado de contenedores en ejecución

```
docker ps
```

Información sobre la versión instalada

```
docker version
```

Detalles sobre la versión instalada

```
docker info
```

Prueba de ejecución con una imagen Docker simple

```
docker run hello-world
```

Listado de las imágenes

```
docker image ls
```

Listado de contenedores en ejecución o que se han ejecutado antes

```
docker container ls --all
```

Fichero *Dockerfile*

- Especifica cómo se va a construir una imagen/componente.
- Normalmente, parte de una imagen existente (hay muchas en el *hub*) y especifica las acciones que hacer para configurarla, qué programas instalar, qué ficheros copiar y qué programa ejecutar como punto de entrada si no se especifica ninguno.

***Dockerfile* de ejemplo**

```
FROM openjdk:8-jre-alpine
MAINTAINER autor@um.es
ENV JAVA_HOME =/usr/lib/jvm/default - jvm
ENV PATH =${JAVA_HOME}/bin:$PATH
ENV JAVA_OPTIONS =- Djava.awt.headless = true
COPY pruebaDocker.jar /directorio
WORKDIR /directorio
CMD ["java", "-jar", "pruebaDocker.jar"]
```

Comandos (para más detalles ver https://docs.docker.com/engine/reference/builder/)

- ***FROM imagen*** Especifica la imagen usada de base. Normalmente las imágenes se codifican como "autor/imagen:versión". Si no se especifica la versión, se utiliza "latest".
- ***ENV VAR=valor*** Especifica valores de variables de entorno que estarán disponibles cuando se ejecute esa imagen.
- ***EXPOSE puertos*** Especifica que se exportarán los puertos dados por parte del software de la imagen cuando se ejecute.
- ***COPY from to*** Copia los ficheros o directorios especificados dentro de la imagen en un directorio interno *to.*
- ***WORKDIR dir*** Especifica el directorio de trabajo del contenedor.
- ***RUN programa*** Ejecuta los programas para configurar la imagen. La imagen quedará con el estado del último programa ejecutado. Por ejemplo, *apt* o *apk* para instalar paquetes.
- ***CMD orden args ó CMD ["orden", "args", ...]*** Especifica el ejecutable a usar.

Construcción, Ejecución y Eliminación de una Imagen

- Una vez escrito el fichero *Dockerfile* podemos pasar a construir la imagen.
- La construcción sólo especifica un directorio donde buscar el fichero *Dockerfile* (“.” en este caso) y, opcionalmente, un nombre para la imagen.

Crear una imagen docker con un nombre

```
docker build –t nuevaimagen .
```

Comprobar que la imagen se ha creado

```
docker image ls
```

Ejecutar la imagen

```
docker run nuevaimagen
```

Parar un contenedor

```
docker stop container_id
```

Eliminar un contenedor

```
docker rm container_id
```

Eliminar la imagen (primero tienen que estar parados todos los contenedores asociados)

```
docker rmi nuevaimagen
```

docker-compose

- *docker-compose* es una utilidad para construir, lanzar y gestionar un conjunto de servicios, donde cada servicio es uno o más contenedores Docker.
- Permite:
  - Construir imágenes Docker.
  - Lanzar aplicaciones basadas en componentes.
  - Gestionar el estado de cada contenedor.
  - Escalar servicios.
  - Ver los *logs* como un todo.
- Fichero principal de Compose es el fichero *docker-compose.yml*
  - En formato YAML (http://yaml.org/)

***docker-compose* de ejemplo**

```
version: "3"
services:
  web:
    # replace username/repo:tag with your name and image details
    image: username/repo:tag
    deploy:
      replicas: 5
      resources:
        limits:
          cpus: "0.1"
          memory: 50M
      restart_policy:
        condition: on-failure
    ports:
      - "80:80"
    networks:
      - webnet
networks:
  webnet:
```

Comandos (para más detalles ver https://docs.docker.com/compose/compose-file/)

- *services* establece el conjunto de servicios, cada uno con su nombre.
- Se pueden escalar, y se le añade un número a cada servicio.
- Por defecto se crea una red con un servicio DNS.
- El nombre de los ordenadores corresponde con los servicios.
- Los puertos exportados por los contenedores se acceden directamente en la red privada creada.

Inicio de los contenedores en segundo plano

```
docker-compose up -d
```

Ver los logs generados

```
docker-compose logs
```

Eliminar los contenedores

```
docker-compose rm
```