Acciones principales con Docker

Sistemas de Gestión Empresarial

IES Clara del Rey

Gestión de Docker

Existen varias herramientas que permiten trabajar con Docker tanto desde línea de comandos como desde interfaz gráfica.

En **Visual Studio Code** hay **extensiones** que permiten trabajar con Docker:

- Docker
- Docker Explorer
- Docker Compose
- Docker Run

Además, hay herramientas dedicadas como **Docker Desktop** para la gestión de contenedores.

En este tema nos vamos a centrar en el uso mediante consola hasta entender cómo funcionan Docker.

Imágenes

- La imagen es una plantilla de solo lectura que se utiliza para crear contenedores. A partir de una imagen pueden crearse múltiples contenedores.
- Las imágenes, tienen su sistema de ficheros predefinido, parámetros predefinidos (comandos, de variables de entorno, etc.) con valores por defecto y personalizables al crear el contenedor.
- Docker permite crear nuevas imágenes basándose en imágenes anteriores. Una imagen puede estar formada por un conjunto de **capas** que han modifican una imagen base.
- Al crear una nueva imagen, simplemente estamos añadiendo una capa a la imagen anterior, la que actúa como base.

Contenedores

- Son instancias de una imagen.
- Pueden ser arrancados, parados y ejecutados.
- Cada contenedor Docker posee un identificador único de 64 caracteres. Habitualmente se utiliza una versión corta con los primeros 12 caracteres o se le asigna un nombre.

Dónde se almacenan imágenes y contenedores

- Información del sistema: **docker info**

Proporciona directorio de Docker y driver de almacenamiento.

- Valores habituales:
  - Directorio: **/var/lib/docker**
  - Al usar el driver *overlay2*
    - imágenes: **/var/lib/docker/overlay2**
    - contenedores: **/var/lib/docker/containers**

Registro: Docker Hub

Docker Hub es una **plataforma de registro** de Docker. Los servicios básicos son gratuitos y permite registrar imágenes Docker, haciéndolas públicas o privadas.

https://hub.docker.com/

Docker utiliza esta plataforma registro como **registro por defecto**

Documentación de referencia de Docker

Enlace a la documentación de referencia

Creación y arranque de contenedores

docker run

Este comando crea un contenedor a partir de una imagen y lo arranca.

Importante: un error común es creer que **docker run** solo arranca contenedores. Si se ejecutan varios **docker run**, se crean varios contenedores, no arrancando varias veces un contenedor.

docker create

Crea un contenedor sin arrancarlo

Crear el primer contenedor

docker run hello-world

La primera ejecución

- *hello-word:latest* indica que la imagen no está localmente en nuestro sistema.
- Ha descargado una imagen llamada *hello-world:latest* , que instala la última versión.
- Se crea el contenedor, se inicia y ejecuta un proceso.

```
alumno@linuxserver2324:~/dockers$ docker run --name hola1 hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/

alumno@linuxserver2324:~/dockers$
```

Figure 1: Salida de la ejecución del hello world

Listar los contenedores disponibles

- **docker ps**: Lista los contenedores en ejecución en el sistema
- **docker ps -a**: Muestra un listado de todos los contenedores, tanto aquellos en funcionamiento como los que están parados

Información que se obtiene:

- **CONTAINER_ID**: identificador único del contenedor (versión 12 primeros caracteres).
- **IMAGE**: imagen utilizada para crear el contenedor.
- **COMMAND**: comando que se lanza al arrancar el contenedor.
- **CREATED**: cuando se creó el contenedor.
- **STATUS**: si el contenedor está en marcha o no (cuánto lleva en marcha o cuánto hace que se paró).
- **PORTS**: redirección de puertos del contenedor.
- **NAMES**: nombre del contenedor.

Parada y arranque de contenedores

- **docker start**
- **docker stop**
- **docker restart**

Forma de uso:

docker start 434d318b3771 (con identificador)

docker start stupefied_colden (con nombre)

- **docker inspect**: muestra detalles de configuración del contenedor

Ejecución de comandos

- **docker exec**: permite ejecutar un comando dentro de un contenedor que esté en ese momento en ejecución.

```
docker exec [OPCIONES] IDENTIFICADOR/NOMBRE
            COMANDO [ARGUMENTOS]
```

```
docker exec -d contenedor touch /tmp/prueba
```

```
docker exec -it contenedor bash
```

```
docker exec -it -e VAR1=1 contenedor bash
```

Copia de ficheros a un contenedor

- **docker cp**: permite copiar ficheros y directorios del anfitrión a un contenedor o viceversa.

Actualmente no se permite la copia de ficheros entre contenedores.

```
docker cp idcontainer:/tmp/prueba ./
```

```
docker cp ./miFichero idcontainer:/tmp
```

Acceso a un contenedor en ejecución

La orden **docker attach** permite enlazar la entrada o salida estándar de nuestra terminal a un contenedor que está ejecutando un proceso en segundo plano.

```
docker attach [OPCIONES] IDENTIFICADOR/NOMBRE
```

Ejecutar el ejemplo:

```
docker run -d --name=muchotexto busybox sh -c \
    "while true; do $(echo date); sleep 1; done"
docker attach muchotexto
...
docker stop muchotexto
```

Los logs en contenedores

docker logs permite consultar la información generada por el contenedor.

```
docker logs [OPCIONES] IDENTIFICADOR/NOMBRE
```

En el ejemplo anterior

```
docker logs -f --until 2s muchotexto
```

```
-f, --follow mantiene abierta la salida
--until string Muestra los logs durante un tiempo
```

Renombrado y borrado de contenedores

docker rename permite cambiar el nombre de un contenedor

```
docker rename contenedor1 contenedor2
```

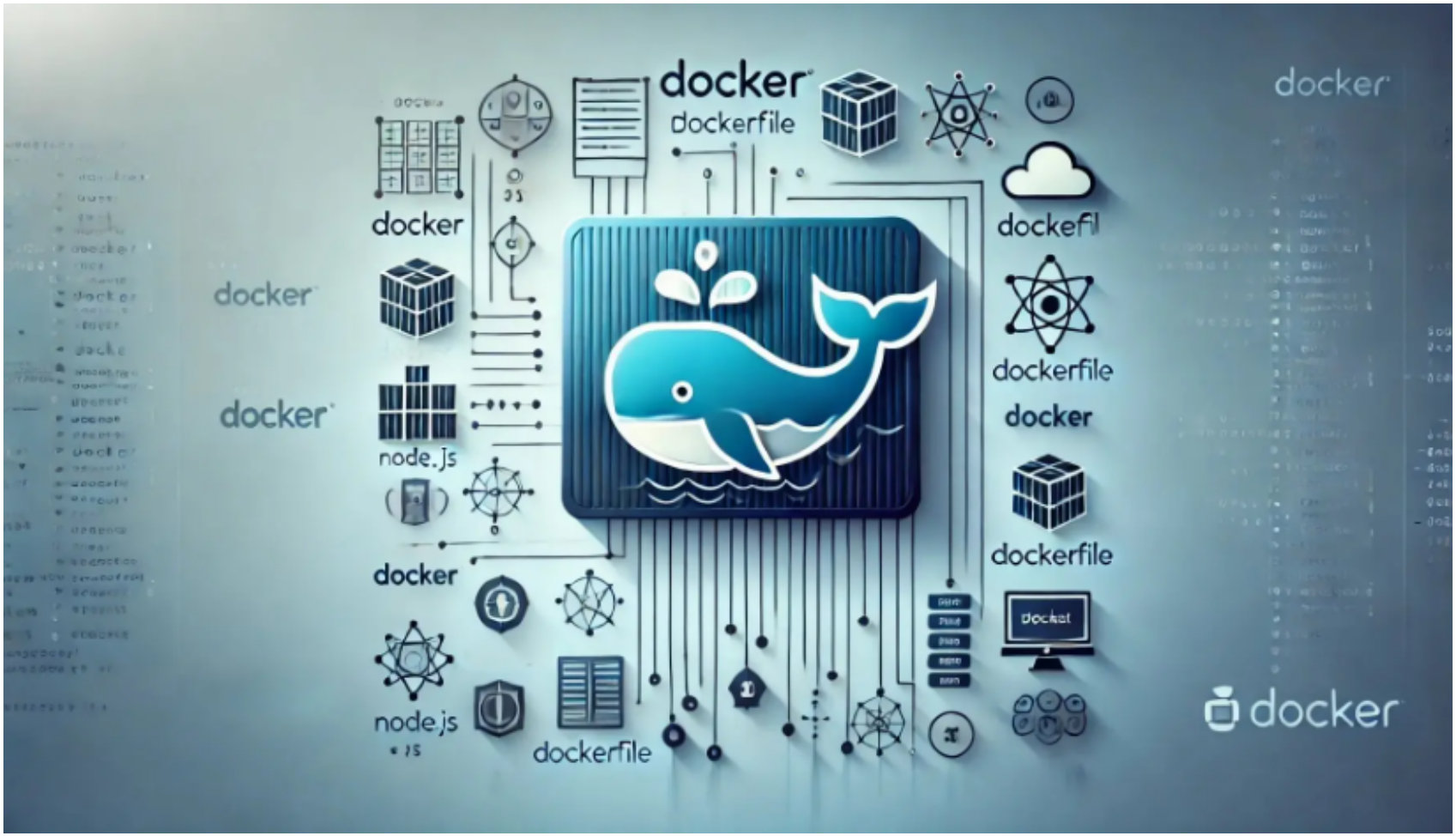
docker rm permite borrar uno o varios contenedores

```
docker rm [OPCIONES] IDENTIFICADOR/NOMBRE [CONTAINER...]
```

Ejercicios prácticos con docker run

Práctica 1. Ejercicios iniciales con contenedores

A por ello!


docker
Dockerfile
docker
docker
node.js
docker
node.js
dockerfile
dockerfil
dockerfile
docker
dockerfile
docker
docker