Docker Básico

Arturo Silvelo

Try New Roads

Ejercicio Docker Compose

Crear la estructura que llevamos usando hasta ahora pero en formato `compose`.

Ejercicio 1: Crear Compose de backend

1. Crear un fichero `docker-compose.yml` ó `compose.yml`

2. Configurar el servicio `backend`

```
docker run -d \
-p 3005:3000 \
--network course-network \
--hostname course-compose-backend \
--name ccb \
-e USE_DB=false \
ghcr.io/trynewroads/course-backend:1.0.0
```

3. Levantar el `compose`

```
docker compose -f 06-compose/soluciones/compose.yml up -d

docker compose -f ./06-compose/soluciones/compose.yml ps
```

4. Comprobar que el servicio.

Acceder al **servidor**

Ejercicio 2: Crear Compose del frontend

1. Editar el fichero `docker-compose.yml` ó `compose.yml`

2. Configurar el servicio `frontend`

```
docker run \
-d
-p 8080:80 \
--network course-network \
--hostname course-compose-frontend \
-v ./nginx/default.conf.template:/etc/nginx/templates/default.conf.template:ro \
--name ccf \
ghcr.io/trynewroads/course-frontend:1.0.0
```

3. Levantar el `compose`

```
docker compose -f 06-compose/soluciones/compose.yml up -d

docker compose -f ./06-compose/soluciones/compose.yml ps
```

4. Comprobar que el servicio.

Acceder al **Cliente**

Ejercicio 3: Crear Compose de la base de datos

1. Editar el fichero `docker-compose.yml` ó `compose.yml`

2. Configurar el servicio `postgres`

```
docker run
-d
--name ccdb
-e POSTGRES_PASSWORD=12345678
--hostname course-database
--network course-network
-v postgres_data:/var/lib/postgresql/data postgres
```

3. Levantar el `compose`

```
docker compose -f 06-compose/soluciones/compose.yml up -d

docker compose -f ./06-compose/soluciones/compose.yml ps
```

4. Comprobar que el servicio.

```
docker exec -it ccdb psql -U postgres -d postgres -c "SELECT * FROM task;"
```

Ejercicio 4: Mejorar compose

1. Establecer redes independientes para aislar los servicios que se comuniquen entre ellos.

```
course-compose-backend:
  networks:
    - course-compose-back
    - course-compose-front

course-compose-frontend:
  networks:
    - course-compose-front

course-compose-database:
  networks:
    - course-compose-back

networks:
  course-compose-back:
    driver: bridge

  course-compose-front:
    driver: bridge
```

2. Establecer dependencia entres los servicios para que se inicien en orden (`depends_on`)

```
course-compose-backend:
  depends_on:
    - course-compose-database

course-compose-frontend:
  depends_on:
    - course-compose-backend
```

3. Eliminar las exposición de puertos no necesarios (`puerto de backend`)

```
course-compose-backend:
  #port:
  #  - 3005:3000
```

4. Limpiar sistema

Probamos el compose que funciona con los cambios

```
docker compose -f ./06-compose/soluciones/4.compose.yml up -d
docker compose -f ./06-compose/soluciones/4.compose.yml ps
```

Comprobamos si podemos acceder al **Cliente**, que no podemos acceder al **Servidor**.

```
docker compose -f ./06-compose/soluciones/4.compose.yml down --volumes
```