# 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`

4. Comprobar que el servicio.

# 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`

4. Comprobar que el servicio.

# 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 cdb
-e POSTGRES_PASSWORD=12345678
--hostname course-database
--network course-network
-v postgres_data:/var/lib/postgresql/data postgres
```

3. Levantar el `compose`
4. Comprobar que el servicio.

# Ejercicio 4: Mejorar compose

1. Establecer redes independientes para aislar los servicios que se comuniquen entre ellos. (`Crear dos redes course-compose-public y course-compose-private`)
2. Establecer dependencia entres los servicios para que se inicien en orden (`depends_on`)
3. Eliminar las exposición de puertos no necesarios (`puerto de backend`)
4. Limpiar sistema