# Docker-Compose

**Docker Compose** é unha ferramenta oficial de Docker que permite **definir e xestionar contedores múltiples** como unha soa aplicación.

En lugar de lanzar cada contedor a man con `docker run`, podes describir toda a aplicación nun ficheiro chamado `docker-compose.yml`.

Exemplo típico: unha aplicación web precisa tres servizos:

- un contedor con **Nginx**,
- un contedor con **aplicación Node.js**,
- e un contedor con **base de datos MySQL**.

Con Compose, descríbese todo iso nun único ficheiro e arráncase con: ***docker compose up [-d]***

```
version: "3.9"

services:
  web:
    image: nginx:alpine
    ports:
      - "8080:80"
    volumes:
      - ./html:/usr/share/nginx/html
    depends_on:
      - app

  app:
    build: ./app
    environment:
      - NODE_ENV=production
    ports:
      - "${APP_PORT}:3000"
    volumes:
      - ./app:/usr/src/app
    command: ["npm", "start"]

  db:
    image: mysql:8
    environment:
      - MYSQL_ROOT_PASSWORD=secret
      - MYSQL_DATABASE=mydb
    volumes:
      - dbdata:/var/lib/mysql

volumes:
  dbdata:
```

APP_PORT é unha variable de entorno, normalmente almacenada en .env

docker compose stop
docker compose down [--volumes]
docker compose ps
docker compose logs -f

## Exemplo 2: PHP + MySQL + phpMyAdmin

```
services:
  db:
    image: mysql:8
    environment:
      MYSQL_ROOT_PASSWORD: example
    volumes:
      - db_data:/var/lib/mysql

  php:
    image: php:apache
    ports:
      - "8080:80"
    volumes:
      - ./php:/var/www/html
    depends_on:
      - db

  phpmyadmin:
    image: phpmyadmin/phpmyadmin
    ports:
      - "8081:80"
    environment:
      PMA_HOST: db
    depends_on:
      - db

volumes:
  db_data:
```

Por defecto, Compose crea unha **rede bridge interna** co nome do proxecto. Todos os servizos comparten esa rede e comunícanse entre si polo nome de servizo.
Si desexamos personalizar as redes, podemos facelo:

```
networks:
  frontend:
  backend:

services:
  web:
    image: nginx
    networks:
      - frontend
  app:
    image: node
    networks:
      - frontend
      - backend
  db:
    image: postgres
    networks:
      - backend
```

Por defecto, Compose crea unha **rede bridge interna** co nome do proxecto. Todos os servizos comparten esa rede e comunícanse entre si polo nome de servizo. O seguinte exemplo levanta un servidor DNS con SSH

```
services:
  xavi-dns:
    image: debian-bind
    container_name: xavi-dns
    hostname: xavi-dns
    restart: always
    environment:
      SSH_USER: xavi
      SSH_PASSWORD: abc123.
      BIND_IP: 172.30.0.100
      BIND_FQDN: xavi.daw.iesrodeira.com
      BIND_RECORDS: mysql IN A 172.30.1.202
      DNS: 192.168.1.10
      DEFAULT_ROUTE: 172.30.0.1
      ENTITTY_NAME: IES de Rodeira
      WELCOME_MESSAGE: Servidor dns de xavi.daw.iesrodeira.com.\n\nCando modifiques a zona DNS recarga a
configuracion escribindo "load-zones" como root\n
    networks:
      daw-network:
        ipv4_address: 172.30.0.100
    cap_add:
      - NET_ADMIN
    volumes:
      - bind9_config-xavi-dns:/etc/bind
      - bind9_cache-xavi-dns:/var/cache/bind
      - bind9_lib-xavi-dns:/var/lib/bind
      - ssh-xavi:/etc/ssh/host_keys

networks:
  daw-network:
    external: true  «« A rede non é creada por docker-compose

volumes:
  bind9_config-xavi-dns:
    name: bind9_config-xavi-dns
  bind9_cache-xavi-dns:
    name: bind9_cache-xavi-dns
  bind9_lib-xavi-dns:
    name: bind9_lib-xavi-dns
  ssh-daniel:
    name: ssh-xavi
```