# Cours : Comprendre un fichier docker-compose.yml

Ce cours explique simplement, pas à pas, ce qu’est un fichier docker-compose.yml, à quoi il sert, et comment le construire. L’objectif est que tu comprennes le rôle de chaque partie sans jargon technique.

## 1. À quoi sert Docker Compose ?

Quand tu développes une application web, tu as souvent plusieurs parties : ton serveur web (comme Express), une base de données ou un cache (comme Redis), et d’autres services. Docker Compose permet de démarrer tous ces services ensemble avec un seul fichier, le docker-compose.yml.

## 2. Le principe de base

Chaque service (comme ton serveur Express ou Redis) est une 'petite boîte' indépendante appelée conteneur. Le fichier docker-compose.yml dit à Docker : 'Crée ces boîtes, connecte-les et fais-les communiquer.'

## 3. Structure d’un fichier docker-compose.yml

Voici la structure de base d’un fichier docker-compose.yml :

version: "3.9"  
  
services:  
 web:  
 redis:  
  
volumes:  
 ...

• version : indique la version du format Docker Compose.

• services : liste des conteneurs à lancer.

• volumes : pour sauvegarder les données.

## 4. Construire le fichier pas à pas

Tu veux deux services : ton serveur Express (web) et Redis.

Étape 1 : le squelette

version: "3.9"  
  
services:  
 web:  
 redis:

Étape 2 : ajouter le service web

web:  
 build: .  
 ports:  
 - "8000:8000"

Étape 3 : connecter web à redis via une variable d’environnement

environment:  
 - REDIS\_URL=redis://redis:6379

Ici, 'redis' dans l’URL correspond au nom du service Redis défini dans le fichier. Docker Compose crée automatiquement un petit réseau interne entre les services.

Étape 4 : indiquer que web dépend de redis

depends\_on:  
 - redis

Étape 5 : définir le service Redis

redis:  
 image: redis:7-alpine  
 volumes:  
 - redis-data:/data

Étape 6 : déclarer le volume

volumes:  
 redis-data:

## 5. Exemple complet final

version: "3.9"  
  
services:  
 web:  
 build: .  
 ports:  
 - "8000:8000"  
 environment:  
 - REDIS\_URL=redis://redis:6379  
 depends\_on:  
 - redis  
 restart: unless-stopped  
  
 redis:  
 image: redis:7-alpine  
 volumes:  
 - redis-data:/data  
 restart: unless-stopped  
  
volumes:  
 redis-data:

## 6. Explication visuelle du fonctionnement

Le service 'web' (ton app Express) se connecte au service 'redis' grâce à l’adresse REDIS\_URL. Redis sauvegarde ses données dans un volume nommé redis-data pour qu’elles persistent après un redémarrage.

## 7. Commandes Docker Compose utiles

• docker compose up -d → démarre les conteneurs en arrière-plan.

• docker compose down → arrête et supprime les conteneurs.

• docker compose down -v → supprime aussi les volumes (attention, cela efface les données).

• docker compose logs -f → affiche les logs en direct.

• docker compose exec web sh → ouvre un terminal dans le conteneur web.

## 8. Résumé du cours

Docker Compose simplifie le travail avec plusieurs conteneurs. En un seul fichier YAML, tu décris comment ton application doit démarrer, comment les services communiquent et comment les données sont conservées. Cela permet d’avoir une application complète (Express + Redis) prête à tourner en quelques commandes.