Docker Compose Multi-container stacks

CONCEPTS

A project is one Compose file's services, networks and volumes. Commands act on the current project; the folder name is its default name.

Project one compose.yaml
- **Service** definition + replicas
  - **Container** running instance

Services reach each other by service name on the project network. Use the container port, not the published one.

LIFECYCLE

docker compose up
Create and start services

docker compose up -d
Start in the background

docker compose up --build
Rebuild images, then start

docker compose up -d --wait
Start and wait until healthy

docker compose up -d --force-recreate
Recreate even if unchanged

docker compose up -d --no-deps *<service>*
Start one service only

docker compose up -d --scale *<service>*=*<n>*
Run n replicas

docker compose up --watch
Sync or rebuild on file changes

docker compose down
Remove containers and networks

docker compose down -v
Also delete the volumes

docker compose down --remove-orphans
Drop containers of removed services

docker compose stop
Stop without removing

docker compose restart *<service>*
Restart a service

docker compose start
Start stopped containers

INSPECT

docker compose ps
List the project's containers

docker compose ps -a
Include stopped containers

docker compose logs *<service>*
Show a service's output

docker compose logs -f *<service>*
Follow a service's output

docker compose logs --tail 100 *<service>*
Only the last 100 lines

docker compose logs --since 10m
Only recent lines

docker compose logs -t
Add timestamps

docker compose port *<service> <port>*
Show the published host port

docker compose ls
List all projects on this host

RUN COMMANDS

docker compose exec *<service>* **sh**
Shell inside a running service

docker compose exec -T *<service> <cmd>*
No TTY, for scripts and pipes

docker compose run --rm *<service>* **sh**
One-off container, removed after

docker compose run --no-deps --rm *<service> <cmd>*
Skip starting dependencies

docker compose cp *<service>*:*<path> <dir>*
Copy files out of a service

BUILD AND IMAGES

docker compose build
Build all services with a build key

docker compose build *<service>*
Build one service

docker compose build --no-cache
Ignore the build cache

BUILD AND IMAGES (cont.)

docker compose build --pull
Fetch newer base images

docker compose pull
Pull the services' images

docker compose push
Push the services' images

FILES AND CONFIG

docker compose config
Print the merged, resolved file

docker compose config -q
Validate only, print nothing

docker compose -f *<file>* **up**
Use a specific Compose file

docker compose -f *<a>* **-f** *<b>* **up**
Merge files, later ones win

docker compose -p *<name>* **up**
Set the project name

docker compose --env-file *<f>* **up**
Read variables from another file

docker compose --profile *<p>* **up**
Enable a profile

COMPOSE.YAML

Service keys

Key	Use
`image`	Image to run
`build`	Build context or object
`command`	Override the command
`ports`	Publish "host:ctr"
`environment`	Variables as map or list
`env_file`	Load variables from files
`volumes`	Mounts and volumes
`depends_on`	Start order, optional health
`restart`	Restart policy
`healthcheck`	Readiness test
`profiles`	Start only when enabled

Minimal file

```
services:
  web:
    build: .
    ports: ["8080:80"]
    depends_on:
      db: { condition:
service_healthy }
  db:
    image: postgres:17
    volumes: ["dbdata:/var/lib/
postgresql/data"]
volumes:
  dbdata:
```