Docker Compose Cheatsheet

By Dejan Panovski • Updated on Jul 15, 2026 •

Docker Compose quick reference: lifecycle commands, Compose file directives, health checks, environment variables, profiles, scaling, and cleanup.

Docker Compose defines and runs multi-container applications from a single YAML file. This cheatsheet covers current docker compose commands for starting, inspecting, and cleaning up services, plus common Compose file directives; it does not cover the legacy docker-compose v1 binary.

Start & Stop

Bring services up and take them down.

`docker compose up`	Create and start all services
`docker compose up -d`	Start in background (detached)
`docker compose up -d --build`	Rebuild images before starting
`docker compose up -d service`	Start one service and its dependencies
`docker compose up --wait`	Detach and wait until services are running or healthy
`docker compose stop`	Stop services without removing them
`docker compose start`	Start stopped services
`docker compose restart`	Restart all services
`docker compose down`	Stop and remove containers and networks
`docker compose down -v`	Also remove project-owned named and anonymous volumes

Inspect & Logs

Check service status and output.

`docker compose ps`	List running project containers
`docker compose ps -a`	List all project containers, including stopped
`docker compose logs`	Show logs for all services
`docker compose logs -f service`	Follow logs for one service
`docker compose logs --tail 100 service`	Show last 100 log lines
`docker compose top`	Running processes per service
`docker compose events`	Stream container events
`docker compose port service 80`	Host port mapped to container port 80
`docker compose ls`	List running Compose projects

Exec & Run

Run commands inside service containers.

`docker compose exec service sh`	Open shell in a running service
`docker compose exec service command`	Run command in a running service
`docker compose exec -u root service sh`	Shell as a specific user
`docker compose run --rm service command`	Run a one-off container
`docker compose run --rm --no-deps service sh`	One-off without starting dependencies
`docker compose cp service:/path ./local`	Copy from a service container
`docker compose cp ./local service:/path`	Copy into a service container
`docker compose attach service`	Attach to a service's output

Build & Images

Build, pull, and push service images.

`docker compose build`	Build all service images
`docker compose build service`	Build one service image
`docker compose build --no-cache`	Build without layer cache
`docker compose pull`	Pull service images
`docker compose push`	Push service images to a registry
`docker compose images`	List images used by services
`docker compose create`	Create containers without starting them
`docker compose watch`	Sync files and rebuild on changes

Compose File Basics

Core service directives in docker-compose.yml .

`services:`	Top-level map of containers to run
`image: postgres:16`	Run an existing image
`build: .`	Build from a Dockerfile in a path
`command: ["npm", "run", "dev"]`	Override the image default command
`entrypoint: ["/entrypoint.sh"]`	Override the image entrypoint
`ports: ["8080:80"]`	Publish host port 8080 to container port 80
`volumes: ["db_data:/var/lib/postgresql/data"]`	Mount a named volume
`environment: ["DEBUG=1"]`	Set environment variables
`env_file: .env.local`	Load variables from a file
`container_name: myapp`	Fixed container name (prevents scaling)

Health & Startup Order

Control dependency order and restart behavior.

`depends_on: [db]`	Start db before this service
`depends_on: {db: {condition: service_healthy}}`	Wait until db passes its health check
`healthcheck: {test: ["CMD", "pg_isready"]}`	Define a health probe
`interval, timeout, retries, start_period`	Health check timing fields
`restart: "no"`	Never restart (default)
`restart: always`	Always restart
`restart: on-failure`	Restart on non-zero exit
`restart: unless-stopped`	Restart unless manually stopped

Environment Variables

Pass configuration into services. Keep `.env` files out of version control; use Compose secrets for sensitive data.

`environment: ["API_KEY=${API_KEY}"]`	Interpolate from the shell or `.env`
`${VAR:-default}`	Use default when VAR is unset or empty
`${VAR:?message}`	Fail with message when VAR is unset or empty
`env_file: .env.local`	Load variables from a specific file
`.env`	Loaded automatically from the project directory
`docker compose --env-file .env.prod up -d`	Use an alternate env file
`docker compose config`	Render final config with variables resolved

Projects & Files

Compose looks for `compose.yaml` first; `docker-compose.yml` is still supported.

`docker compose -f compose.prod.yaml up -d`	Use a specific file
`docker compose -f base.yml -f override.yml up -d`	Merge files; later files can add, merge, or override values
`docker compose -p myproject up -d`	Set the project name
`COMPOSE_FILE=base.yml:override.yml`	Default files via environment variable
`COMPOSE_PROJECT_NAME=myproject`	Default project name
`docker compose config`	Validate and print the merged config
`docker compose config --services`	List service names
`docker compose version`	Show Compose version

Profiles & Scaling

Run optional services and multiple replicas.

`profiles: ["debug"]`	Assign a service to a profile
`docker compose --profile debug up -d`	Start with a profile enabled
`COMPOSE_PROFILES=debug,test`	Enable profiles via environment variable
`docker compose up -d --scale worker=3`	Run 3 replicas of a service
`docker compose up -d --no-deps service`	Start or update one service without its dependencies
`docker compose up -d --force-recreate`	Recreate containers even without changes

Cleanup

Remove what a project created.

`docker compose down`	Remove containers and networks
`docker compose down -v`	Also remove project-owned named and anonymous volumes
`docker compose down --rmi all`	Also remove images
`docker compose down --remove-orphans`	Remove containers for removed services
`docker compose rm`	Remove stopped service containers
`docker compose stop service`	Stop a single service