


Docker Commands — Clean & Unified Explanation

IMAGE MANAGEMENT

List images

`docker images`

`docker image ls`

 Shows all Docker images stored locally.

`docker images` is just an **older alias**.

Useful options:

`docker image ls -a` # include dangling/intermediate images

`docker image ls --help`

`docker image --help`

Build images


`docker build -t app-react:1 .`

`docker build -t andonyabdala77/app-react:3 .`

Builds an image from a Dockerfile in the **current directory**.

Flags:

- `-t` → name and tag (image:version)
- `.` → build context (files Docker can access)


 This is the **core command** behind everything Docker does.

Tag images

`docker image tag app-react:1 app-react:2`

`docker image tag app-react:2 andonyabdala77/app-react:2`

Creates a **new name/tag pointing to the same image**.

 Required before pushing to Docker Hub.

✖ Remove images

`docker image rm app-react:1`

`docker image rm -f app-react:2`

`docker image rm mongo:5.0.19-focal`

Deletes images from local storage.

- `-f` → force removal
- ✖ Fails if a container is still using the image (unless forced)

🧹 Cleanup unused images

`docker image prune`

Deletes images **not used by any container**.

⚠ Frees disk space, but can remove images you may need later.

💾 Image backup & restore

Save image to file

`docker image save -o app-react.tar andonyabdala77/app-react:3`

Exports an image to a .tar file.

Load image from file

`docker image load -i app-react.tar`

Imports an image from a .tar file.

📌 Useful when:

- no internet
 - sharing images manually
 - backups
-

DOCKER HUB (REGISTRY)

Login

`docker login`

Authenticates with Docker Hub.

Push images

`docker push andonyabdala77/app-react:2`

`docker push andonyabdala77/app-react:3`

Uploads images to Docker Hub.

CONTAINER LIFECYCLE

Create containers (not running)

`docker create app-react:2`

`docker create --name holamundo app-react:2`

Creates a container **without starting it**.

Run containers

`docker run -d --name holarun app-react:2`

`docker run -d -p 3000:3000 -v datos:/app/datos app-react:4`

`docker run -d -p 80:5173 -v ./src:/app/src --name holamundo app-react:4`

Creates **and starts** a container.

Flags:

- `-d` → detached (background)
- `-p` → port mapping (host:container)
- `-v` → volume / bind mount
- `--name` → container name

Start / Stop containers

`docker start holamundo`

`docker stop holamundo`

Starts or stops an existing container.

List containers

`docker ps`

`docker ps -a`

- `docker ps` → running containers only
- `docker ps -a` → all containers (running + stopped)

Remove containers

`docker container rm holamundo`

`docker container rm -f holamundo`

Deletes containers.

- `-f` → force stop + remove

Cleanup stopped containers

`docker container prune`

Deletes **all stopped containers**.

 Irreversible.

VOLUMES & BIND MOUNTS

Docker-managed volumes

`-v datos:/app/datos`

- Managed by Docker
 - Persistent
 - Safe for production data
-

Bind mounts (host files)

`-v ./src:/app/src`

`-v ./:/app/datos`

Maps host files directly into the container.

📌 Common for:

- development
 - live reload
 - editing code on host
-

👤 EXEC & INTERACTION

Open shell inside a container

`docker exec -it holamundo sh`

`docker exec -it -u root holamundo-app-1 sh`

Runs a command **inside a running container**.

Flags:

- `-i` → interactive
- `-t` → terminal
- `-u root` → run as root
- `sh` → shell

👉 This is how you debug containers.

📁 COPY FILES BETWEEN HOST & CONTAINER

Container → Host

`docker cp holamundo:/app/contenedor.txt .`

Host → Container

`docker cp anfitrión.txt holamundo:/app/`



DOCKER COMPOSE COMMANDS



Build services

`docker compose build`

`docker compose build --no-cache`

Builds images defined in `docker-compose.yml`.



Run services

`docker compose up`

`docker compose up -d`

`docker compose up --build -d`

Creates and starts **all services together**.



Stop & remove services

`docker compose down`

Stops containers and removes:

- containers
 - default network
-



List compose containers

`docker compose ps`

`docker compose ps -a`

Logs

`docker compose logs`

`docker compose logs -f`

`docker compose logs -t`

`docker compose logs -ft`

Shows logs from all services.

NETWORK COMMANDS

`docker network`

`docker network ls`

Lists Docker networks.

 Docker Compose **automatically creates networks.**