DEV REFERENCE / CONTAINERS

docker & podman cheat sheet

Commands are interchangeable — substitute `podman` for `docker` unless noted.

IMAGES

Pull an image	`docker pull <image>[:tag]`
List local images	`docker images`
Build from Dockerfile	`docker build -t <name>[:tag] <path>`
Tag an image	`docker tag <image> <new-name>[:tag]`
Remove an image	`docker rmi <image>`
Remove unused images	`docker image prune`
Inspect an image	`docker inspect <image>`
Show image history	`docker history <image>`

CONTAINERS — RUN

Run a container	`docker run <image>`
Run and remove on exit	`docker run --rm <image>`
Run detached (background)	`docker run -d <image>`
Run with interactive shell	`docker run -it <image> bash`
Name the container	`docker run --name <name> <image>`
Map ports	`docker run -p <host>:<container> <image>`
Mount a volume	`docker run -v <host-path>:<container-path> <image>`

Set environment variable	`docker run -e KEY=value <image>`
Load env from file	`docker run --env-file .env <image>`
Limit memory	`docker run -m 512m <image>`

CONTAINERS — MANAGE

List running containers	`docker ps`
List all containers	`docker ps -a`
Stop a container	`docker stop <container>`
Start a stopped container	`docker start <container>`
Restart a container	`docker restart <container>`
Kill a container (force)	`docker kill <container>`
Remove a container	`docker rm <container>`
Remove all stopped containers	`docker container prune`
Rename a container	`docker rename <old> <new>`

CONTAINERS — INSPECT & DEBUG

View logs	`docker logs <container>`
Tail logs	`docker logs -f <container>`
Exec into running container	`docker exec -it <container> bash`
Copy file from container	`docker cp <container>:<path> <dest>`
Inspect container details	`docker inspect <container>`
Show resource usage	`docker stats`
Show running processes	`docker top <container>`

Show port mappings	`docker port <container>`

VOLUMES

Create a volume	`docker volume create <name>`
List volumes	`docker volume ls`
Inspect a volume	`docker volume inspect <name>`
Remove a volume	`docker volume rm <name>`
Remove unused volumes	`docker volume prune`

NETWORKS

List networks	`docker network ls`
Create a network	`docker network create <name>`
Connect container to network	`docker network connect <network> <container>`
Disconnect from network	`docker network disconnect <network> <container>`
Inspect a network	`docker network inspect <name>`
Remove a network	`docker network rm <name>`

REGISTRY

Log in to a registry	`docker login [registry]`
Push an image	`docker push <image>[:tag]`
Log out	`docker logout [registry]`
Search Docker Hub	`docker search <term>`

DOCKER COMPOSE

Start services	`docker compose up`
Start detached	`docker compose up -d`
Stop services	`docker compose down`
Stop and remove volumes	`docker compose down -v`
Rebuild images	`docker compose build`
View service logs	`docker compose logs -f [service]`
Run one-off command	`docker compose run <service> <cmd>`
List running services	`docker compose ps`

PODMAN-SPECIFIC

Run rootless container	`podman run <image>` `# rootless by default — no daemon required`
Generate systemd unit	`podman generate systemd --name <container>`
Create a pod	`podman pod create --name <name>`
List pods	`podman pod ls`
Start a pod	`podman pod start <name>`
Stop a pod	`podman pod stop <name>`
Remove a pod	`podman pod rm <name>`
Play Kubernetes YAML	`podman play kube <file.yaml>`

SYSTEM & CLEANUP

Show disk usage	`docker system df`
Remove all unused resources	`docker system prune`
Prune including volumes	`docker system prune --volumes`

Show system info	`docker info`
Show version	`docker version`

DOCKERFILE REFERENCE

Dockerfile

```
# Base image
FROM node:20-alpine

# Set working directory
WORKDIR /app

# Copy dependency manifests first (layer cache)
COPY package*.json ./

# Install dependencies
RUN npm ci --only=production

# Copy source
COPY . .

# Expose port
EXPOSE 3000

# Default command
CMD ["node", "server.js"]
```