Podman Cheatsheet

By Dejan Panovski • Updated on Jun 3, 2026 • Download PDF

Quick reference for Podman commands and rootless container workflows

Podman is a daemonless container engine for running OCI containers, images, pods, and rootless workloads. This cheatsheet covers the Podman commands you use most often for containers, images, volumes, networks, pods, systemd, Compose, and cleanup.

Container Lifecycle

Create, start, stop, and remove containers.

`podman run image`	Create and start a container
`podman run -d image`	Run container in background
`podman run -it image sh`	Run with an interactive shell
`podman run --name web image`	Run with a custom name
`podman run -p 8080:80 image`	Map host port to container port
`podman run -v /host:/container image`	Bind mount a host path
`podman start container`	Start a stopped container
`podman stop container`	Gracefully stop a container
`podman restart container`	Restart a container
`podman rm container`	Remove a stopped container
`podman rm -f container`	Force remove a running container
`podman kill container`	Kill a container with SIGKILL
`podman pause container`	Pause a running container
`podman unpause container`	Resume a paused container

Container Inspection

List containers, inspect state, and watch resource use.

`podman ps`	List running containers
`podman ps -a`	List all containers
`podman ps -q`	Show only container IDs
`podman inspect container`	Show detailed JSON info
`podman inspect --format '{{.State.Status}}' container`	Print one field
`podman top container`	List processes in a container
`podman stats`	Show live resource usage
`podman stats --no-stream`	Print one usage snapshot
`podman port container`	Show port mappings
`podman diff container`	Show filesystem changes

Logs

Read and follow container logs.

`podman logs container`	View container logs
`podman logs -f container`	Follow live logs
`podman logs --tail 50 container`	Show the last 50 lines
`podman logs -t container`	Add timestamps
`podman logs --since 10m container`	Show recent logs
`podman logs --until 2026-06-03T12:00:00 container`	Stop at a time
`journalctl --user -u name.service`	View logs for a user service
`sudo journalctl -u name.service`	View logs for a system service

Images and Builds

Pull, build, tag, and manage images. Podman accepts Dockerfiles and Containerfiles.

`podman pull image:tag`	Pull image from a registry
`podman pull docker.io/library/nginx`	Pull with a full image name
`podman push image:tag`	Push image to a registry
`podman build -t name .`	Build image from Dockerfile
`podman build -f Containerfile .`	Build with a custom file
`podman images`	List local images
`podman rmi image`	Remove an image
`podman tag source target:tag`	Tag an image
`podman save image > image.tar`	Save image to tar archive
`podman load < image.tar`	Load image from tar archive
`podman history image`	Show image layer history

Exec and Copy

Run commands inside containers and copy files.

`podman exec -it container sh`	Open shell in container
`podman exec -it container bash`	Open Bash if installed
`podman exec container command`	Run command in container
`podman exec -u root container command`	Run as a specific user
`podman cp ./file container:/path/`	Copy file into container
`podman cp container:/path/file ./`	Copy file from container
`podman attach container`	Attach to main process
`podman wait container`	Wait for container to stop

Volumes

Manage persistent container data.

`podman volume create data`	Create a named volume
`podman volume ls`	List volumes
`podman volume inspect data`	Show volume details
`podman volume rm data`	Remove a volume
`podman volume prune`	Remove unused volumes
`podman run -v data:/data image`	Mount named volume
`podman run -v /host:/data image`	Bind mount host directory
`podman run -v /host:/data:Z image`	Bind mount with SELinux relabeling
`podman run --tmpfs /tmp image`	Mount tmpfs in a container

Networks

Create networks and connect containers.

`podman network create net`	Create a network
`podman network ls`	List networks
`podman network inspect net`	Show network details
`podman network rm net`	Remove a network
`podman network connect net container`	Connect container to network
`podman network disconnect net container`	Disconnect from network
`podman run --network net image`	Run container on network
`podman run --network host image`	Use host networking
`podman run --network none image`	Disable networking

Pods

Run groups of containers that share a network namespace.

`podman pod create --name app`	Create a pod
`podman pod create --name app -p 8080:80`	Create pod with port mapping
`podman run --pod app image`	Run container in pod
`podman pod ps`	List pods
`podman ps --pod`	Show containers with pod info
`podman pod inspect app`	Show pod details
`podman pod stop app`	Stop all containers in pod
`podman pod start app`	Start pod containers
`podman pod rm app`	Remove a stopped pod
`podman pod rm -f app`	Force remove a pod

Rootless and systemd

Inspect rootless mode and manage Podman with systemd.

`podman info`	Show Podman host configuration
`podman info --format '{{.Host.Security.Rootless}}'`	Check if rootless
`podman system migrate`	Apply user namespace changes
`systemctl --user enable --now podman.socket`	Start user Podman socket
`sudo systemctl enable --now podman.socket`	Start system Podman socket
`sudo loginctl enable-linger $USER`	Keep user services after logout
`systemctl --user daemon-reload`	Reload user units
`systemctl --user status name.service`	Check user service
`sudo systemctl status name.service`	Check system service
`podman auto-update`	Update containers with auto-update labels

Compose

Run Compose projects with Podman.

`podman compose up`	Start Compose project
`podman compose up -d`	Start in background
`podman compose down`	Stop and remove services
`podman compose ps`	List Compose services
`podman compose logs`	View Compose logs
`podman compose exec service sh`	Shell into a service
`podman-compose up -d`	Use podman-compose directly
`export DOCKER_HOST=unix://$XDG_RUNTIME_DIR/podman/podman.sock`	Point Docker Compose at Podman
`docker compose up -d`	Run Docker Compose against Podman socket

System and Cleanup

Check disk usage and remove unused data.

`podman system df`	Show storage usage
`podman system prune`	Remove unused data
`podman system prune --volumes`	Also remove unused volumes
`podman image prune`	Remove dangling images
`podman image prune -a`	Remove unused images
`podman container prune`	Remove stopped containers
`podman volume prune`	Remove unused volumes
`podman network prune`	Remove unused networks
`podman system reset`	Remove all Podman storage

Registry and Login

Authenticate and work with container registries.

`podman login docker.io`	Log in to Docker Hub
`podman login registry.example.com`	Log in to private registry
`podman logout docker.io`	Log out from registry
`podman search term`	Search configured registries
`podman pull user/image:tag`	Pull image from registry
`podman push user/image:tag`	Push image to registry
`podman tag image registry/user/image:tag`	Tag image for registry
`podman manifest create name`	Create manifest list
`podman manifest push name destination`	Push manifest list

Docker Compatibility

Use Podman with Docker-style commands and tooling.

`sudo apt install podman-docker`	Install Docker-compatible wrapper
`docker ps`	Run Podman through Docker wrapper
`alias docker=podman`	Add shell alias
`podman --remote ps`	Use remote Podman client
`podman system service --time=0`	Start API service manually
`podman info --format '{{.Host.RemoteSocket.Path}}'`	Show remote socket path
`podman run -d -p 8080:80 nginx`	Docker-like run command