Docker Cheat Sheet

docker

BUILD

Build an image from the Dockerfile in the current directory and tag the image
docker build -t myimage:1.0 .

List all images that are locally stored with the Docker Engine
docker image ls

Delete an image from the local image store
docker image rm alpine:3.4

SHARE

Pull an image from a registry
docker pull myimage:1.0

Retag a local image with a new image name and tag
docker tag myimage:1.0 myrepo/myimage:2.0

Push an image to a registry
docker push myrepo/myimage:2.0

www.docker.com
www.docker.com/products/kubernetes

RUN

Run a container from the Alpine version 3.9 image, name the running container "web" and expose port 5000 externally, mapped to port 80 inside the container.
docker container run --name web **-p** 5000:80 alpine:3.9

Stop a running container through SIGTERM
docker container stop web

Stop a running container through SIGKILL
docker container kill web

List the networks
docker network ls

List the running containers (add **--all** to include stopped containers)
docker container ls

Delete all running and stopped containers
docker container rm -f $(docker ps -aq)

Print the last 100 lines of a container's logs
docker container logs --tail 100 web

DOCKER MANAGEMENT

All commands below are called as options to the base **docker** command. Run **docker <command> --help** for more information on a particular command.

app*	Docker Application
assemble*	Framework-aware builds (Docker Enterprise)
builder	Manage builds
cluster*	Manage Docker clusters (Docker Enterprise)
config	Manage Docker configs
context	Manage contexts
engine	Manage the docker Engine
gmsa*	Manage Docker gMSA configs (Docker Enterprise)
image	Manage images
network	Manage networks
node	Manage Swarm nodes
plugin	Manage plugins
registry*	Manage Docker registries
secret	Manage Docker secrets
service	Manage services
stack	Manage Docker stacks
swarm	Manage Swarm
system	Manage Docker
template*	Quickly scaffold services (Docker Enterprise)
trust	Manage trust on Docker images
volume	Manage volumes

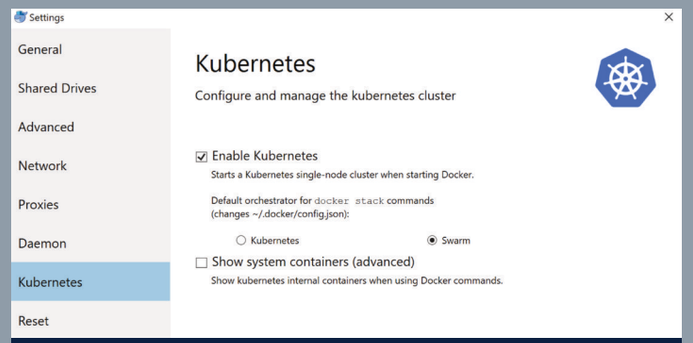
* New Docker CLI plugins for Docker Engine 19.03

Docker Kubernetes Service

docker

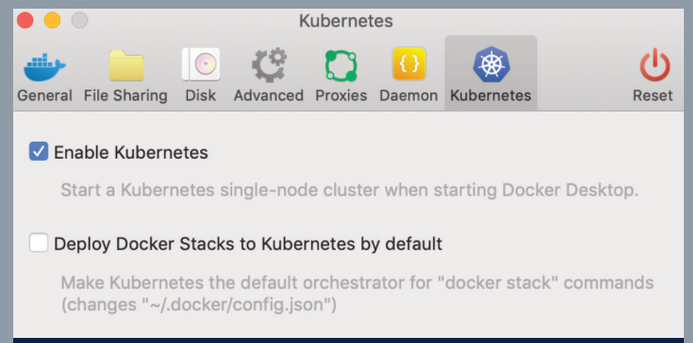
DOCKER DESKTOP

Windows



Change default orchestrator for **stack deploy** commands:
`DOCKER_STACK_ORCHESTRATOR=[kubernetes | swarm]`

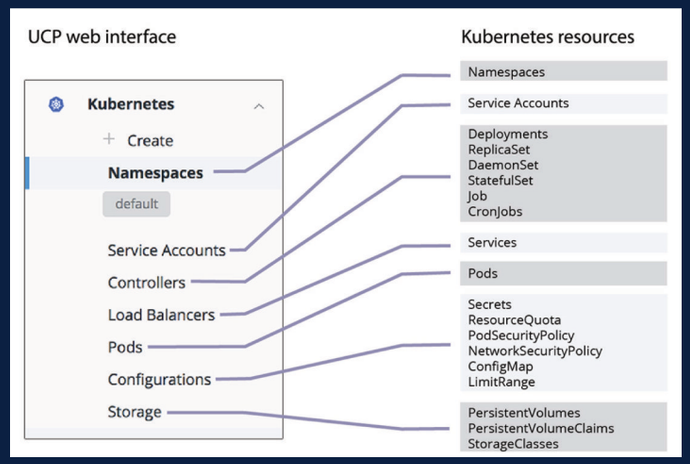
macOS



Change default orchestrator for **stack deploy** commands:
`DOCKER_STACK_ORCHESTRATOR=[kubernetes | swarm]`

DOCKER ENTERPRISE UNIVERSAL CONTROL PLANE

Both Swarm and Kubernetes are installed and available by default in Docker Enterprise 2.0 and later



Docker Desktop includes the Docker and Kubernetes CLIs which can be used for remote cluster access.

> **TIP: Docker Desktop Enterprise** Version Packs ensure desktop APIs are the same version as the UCP cluster.

Authenticate to a UCP cluster from the CLI using Client Bundles:
https://docs.docker.com/ee/ucp/user-access/cli/

Deploy a workload to the Docker Enterprise Kubernetes service:

```
kubectl apply -f deployment.yaml
kubectl get deployments
```

Docker Kubernetes Service includes *Compose on Kubernetes* by default for both UCP clusters and Desktop. You can add *Compose on Kubernetes* for other Kubernetes distributions:
https://github.com/docker/compose-on-kubernetes

Deploy a Compose-based application to default orchestrator in the current context:

```
docker stack deploy
--compose-file /path/to/
docker-compose.yml mystack
```

Deploy a Compose-based app with Kubernetes:

```
docker stack deploy
--orchestrator kubernetes
--namespace my-app
--compose-file /path/to/
docker-compose.yml mystack
```

View deployed services:

```
kubectl get services
```

View deployed services in namespace “my-app”:

```
kubectl get services -n my-app
```