Docker Labs Appendix 1
Working with Rancher Desktop

1 References

The main official reference for Rancher Desktop

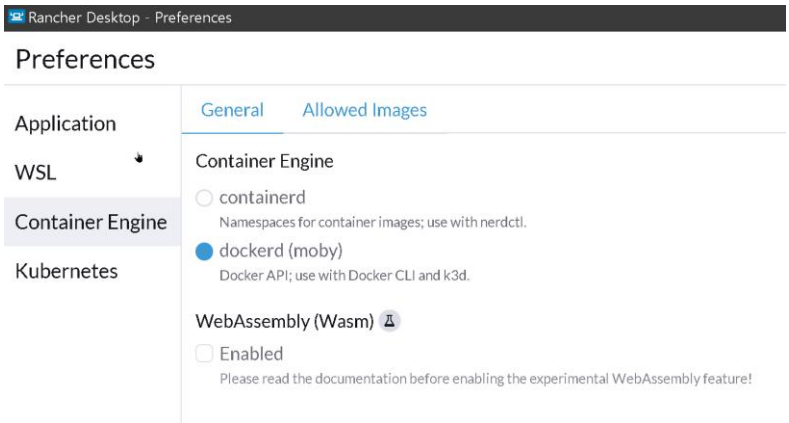
https://docs.rancherdesktop.io/

Reference for working with the UI

https://docs.rancherdesktop.io/ui/general

2 Configuration

By default, Rancher is configured to use dockerd (moby) as the container engine; this is the open source version of the engine used by Docker.



With this engine selected, you can use your typical Docker commands via:

`docker command`

If you use the containerd engine, then you would precede your typical Docker commands with:

`nerdctl command`

instead

3 Specific issues

Rancher Desktop uses K3s, a light weight Kubernetes distribution. This is already activated when Rancher Desktop starts. K3s is implemented through a set of containers that are generated from existing images.

This means that whenever you run the standard Docker commands to view images or containers,

```
docker ps
```

```
docker images
```

you will see all these additional images and containers that are associated with K3s, in addition to any of your own images that you have pulled from DockerHub and any containers that you have started from them.

These extra images / containers that are part of the K3s system will clutter your listing of images or containers.

To work around this, you can view the images / containers using the Dashboard UI, which will automatically exclude the K3s images or containers.

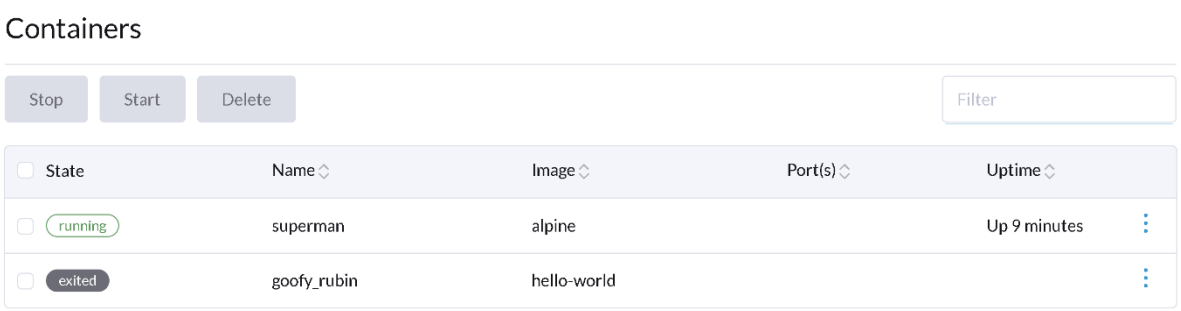
If you are working purely using CLI commands, you can use a `select-string` Powershell function to exclude the K3s related images / containers, for e.g.

```
docker ps -a | select-string -Pattern "k8s" -NotMatch
```

```
docker images | select-string -Pattern "rancher" -NotMatch
```

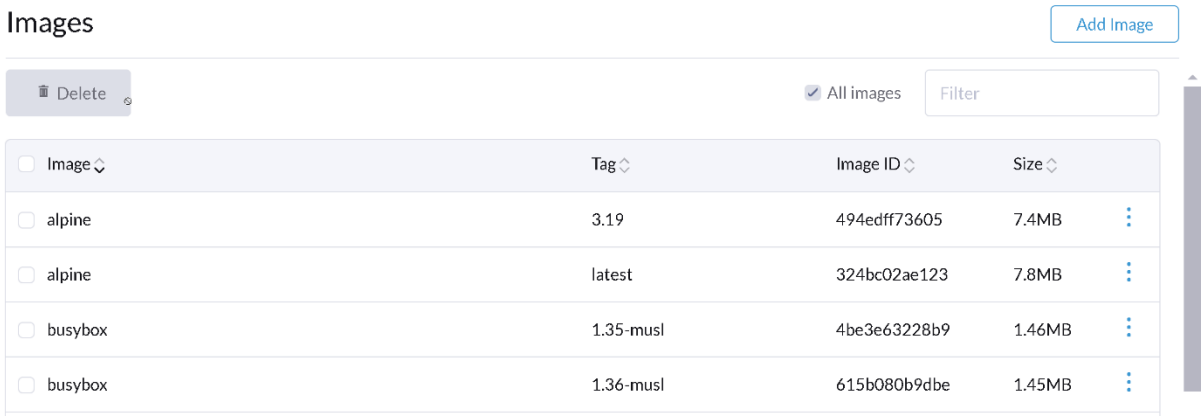
4 Image / Container dashboard UI

You can manage your containers / images directly from the dashboard as an alternative to the standard Docker CLI commands.



You can perform the following container related functions from the UI

- Determine which containers are stopped or running
- Selecting specific containers to stop, start and delete (functionally equivalent to `docker stop`, `docker start` and `docker rm`)
- Filtering on containers based on their names



You can perform the following image related functions from the UI

- Select specific images to delete (equivalent to `docker rmi`)
- Filter and sort on existing images in the local registry

You can also add an image by pulling it from DockerHub (equivalent to `docker pull`) or by running a build on a Dockerfile contained in a specific directory (equivalent to `docker build`)