

Podman

- [Quadlets](#)

Quadlets

Podman Quadlets are declarative configuration files for defining your infrastructure (containers, volumes, etc.). You can define those files and the call of `systemctl daemon-reload` will generate systemd files, so you can handle your podman service like a usual systemd service.

Where do I define the files

There are multiple folders, where you can store config files. Some folders are temporary, some are permanent, some are rootless (runs with your user), some runs as root.

Rootless directories

- `$XDG_RUNTIME_DIR/containers/systemd/`, temporary for the current session
- `$XDG_CONFIG_HOME/containers/systemd/`
- `~/.config/containers/systemd/`
- `/etc/containers/systemd/users/${UID}/`, user specific, used by the administrator
- `/etc/containers/systemd/users/`, for all users
- `/usr/share/containers/systemd/users/${UID}/`
- `/usr/share/containers/systemd/users/`

I recommend using `~/.config/containers/systemd/`, because no root access is needed and it is permanent.

Rootful directories

- `/run/containers/systemd/`, temporary
- `/etc/containers/systemd/`
- `/usr/share/containers/systemd/`

For rootful services, I would recommend the directory `/etc/containers/systemd/`.

Define a container

To define a container, create a file with the ending `.container`, e.g. `test.container`.

Here is an example:

```
[Container]
ContainerName=test
Image=docker.io/nginx
PublishPort=8080:80

[Service]
Restart=on-failure

[Install]
WantedBy=default.target
```

In the `[Container]` section, you define the container itself, such as its name and the image to use. For this Nginx example, I also added a port mapping.

In the `[Service]` section, you define the behavior of the systemd service, such as when it should be restarted.

In the `[Install]` section, you define how the service should be integrated into systemd, for example which target it should be enabled under.

Volumes

To define a volume for your container, use the `Volume` keyword in the `Container` section. You can define local directories using an absolute path, or the name of a volume.

```
[Container]
Volume=/on/the/host:/in/your/container

# Or with a named volume, see "Define a volume"
Volume=my-named-volume:/in/container

# You can also add options, e.g. to set it read only
Volume=my-named-volume:/in/container:ro
```

Define a volume

To define a volume, create a file with the `.volume` ending, e.g. `test.volume`.

```
[Volume]
VolumeName=my-volume
```

