*Youtube - Spoolman

Hallo. Hier zeige wir euch, wie ihr den Service "Spoolman" installiert, um eure Filamentspulen mit Klipper zu verwalten.

Original Dokumentation

Link zum Spoolman Repo:

https://github.com/Donkie/Spoolman

Installation - Spoolman als Service (Standalone)

1. System Updaten

```
sudo apt-get update
```

```
sudo apt-get upgrade -y
```

2. Curl installieren

```
sudo apt-get install -y curl jq
```

3. Sourche URL festlegen:

```
source_url=$(curl -s https://api.github.com/repos/Donkie/Spoolman/releases/latest | jq -r '.assets[] | select(.name == "spoolman.zip").browser_download_url')
```

4. Spoolman Repo downlload

```
curl -sSL $source_url -o temp.zip && unzip temp.zip -d ./Spoolman && rm temp.zip
```

5. Spoolman installieren

```
cd ./Spoolman
```

```
bash ./scripts/install_debian.sh
```

6. Aktueller Fix

```
sudo systemctl stop Spoolman.service
```

```
rm /home/biqu/.local/share/spoolman/spoolman.db
```

```
sudo systemctl start Spoolman.service
```

7. Prüfen ob spoolman läuft
http://<IP>:7912
Beispiel: http://192.168.1.10:7912

Spoolman standalone deinstallieren

```
sudo systemctl stop Spoolman.service
```

```
sudo systemctl disable Spoolman.service
```

```
pi@mainsailos:~ $ sudo systemctl disable Spoolman.service
Removed /etc/systemd/system/default.target.wants/Spoolman.service.
```

```
sudo rm /etc/systemd/system/Spoolman.service
```

Installation mit Docker

Docker installieren

```
sudo apt update && sudo apt upgrade -y
```

```
curl -fsSL https://get.Docker.com -o get-Docker.sh
```

```
sudo sh get-Docker.sh
```

```
sudo usermod -aG docker $USER
```

```
newgrp docker
```

```
docker run hello-world
```

Bild nach der Installation

```
Client: Docker Engine - Community
 Version:           24.0.7
 API version:       1.43
 Go version:        go1.20.10
 Git commit:        afdd53b
 Built:             Thu Oct 26 09:08:26 2023
 OS/Arch:           linux/arm
 Context:           default

Server: Docker Engine - Community
 Engine:
  Version:          24.0.7
  API version:      1.43 (minimum version 1.12)
  Go version:       go1.20.10
  Git commit:       311b9ff
  Built:            Thu Oct 26 09:08:26 2023
  OS/Arch:          linux/arm
  Experimental:     false
 containerd:
  Version:          1.6.24
  GitCommit:        61f9fd88f79f081d64d6fa3bb1a0dc71ec870523
 runc:
  Version:          1.1.9
  GitCommit:        v1.1.9-0-gccaecfc
 docker-init:
  Version:          0.19.0
  GitCommit:        de40ad0

================================================================================

To run Docker as a non-privileged user, consider setting up the
Docker daemon in rootless mode for your user:

    dockerd-rootless-setuptool.sh install

Visit https://docs.docker.com/go/rootless/ to learn about rootless mode.


To run the Docker daemon as a fully privileged service, but granting non-root
users access, refer to https://docs.docker.com/go/daemon-access/

WARNING: Access to the remote API on a privileged Docker daemon is equivalent
         to root access on the host. Refer to the 'Docker daemon attack surface'
         documentation for details: https://docs.docker.com/go/attack-surface/

================================================================================
```

Bild nach "docker run hello-word"

```
Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (arm32v7)
 3. The Docker daemon created a new container from that image which runs th
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

Spoolman Docker Container installieren

1. Verzeichnes für die Spoolman Datenbank anlegen

```
cd ~
```

```
mkdir spoolman
```

```
cd spoolman/
```

```
mkdir data
```

```
chown 1000:1000 data
```

```
nano docker-compose.yml
```

2. Docker-compose.yml

```
version: '3.8'
services:
  spoolman:
    image: ghcr.io/donkie/spoolman:latest
    restart: unless-stopped
    volumes:
      # Mount the host machine's ./data directory into the container's
/home/app/.local/share/spoolman directory
      - type: bind
        source: ./data # This is where the data will be stored locally. Could also be set to for example
`source: /home/pi/printer_data/spoolman`.
        target: /home/app/.local/share/spoolman # Do NOT change this line
    ports:
      # Map the host machine's port 7912 to the container's port 8000
      - "7912:8000"
    environment:
      - TZ=Europe/Berlin # Optional, defaults to UTC
```

3. Container starten

```
docker compose up -d
```

Spoolman könnt ihr wie folg erreichen

http://192.168.xxx.xxx:7912

Spoolman Datenbank von Standalone auf Docker

Die Datenbank wird standardmäßig bei der Standalone in folgenden Verzeichnis gespeichert:

~/.local/share/spoolman/spoolman.db

Ihr müsst sie dann in euer angelegtes "Data" Verzeichnis kopieren

```
cd ~
```

```
cp .local/share/spoolman/spoolman.db ~/spoolman/data/spoolman.db
```

Moonraker.conf und printer.cfg anpassen

moonraker.conf

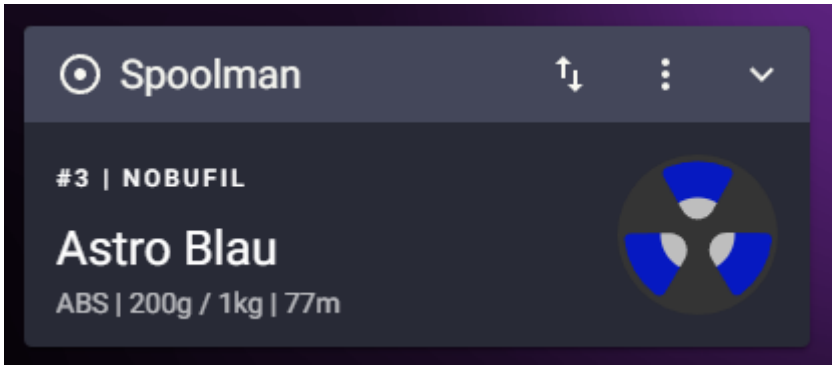
```
[spoolman]
server: http://192.168.0.123:7912
#   URL to the Spoolman instance. This parameter must be provided.
sync_rate: 5
#   The interval, in seconds, between sync requests with the
#   Spoolman server.  The default is 5.
```

printer.cfg | macros.cfg

```
[gcode_macro SET_ACTIVE_SPOOL]
gcode:
  {% if params.ID %}
    {% set id = params.ID|int %}
    {action_call_remote_method(
      "spoolman_set_active_spool",
      spool_id=id
    )}
  {% else %}
    {action_respond_info("Parameter 'ID' is required")}
  {% endif %}

[gcode_macro CLEAR_ACTIVE_SPOOL]
gcode:
  {action_call_remote_method(
    "spoolman_set_active_spool",
    spool_id=None
  )}
```

Beispiel Spoolman auf der Weboberfläche



Hilfe - Wenn ihr falsche Daten in der moonraker.conf eingeben habt

1. Per SSH auf den Pi verbinden
2. moonraker.conf bearbeiten

```
cd ~
nano printer_data/config/moonraker.conf
```

3. Nach der Bearbeitung Dienst neustarten:

```
sudo service moonraker restart
```

Optional - Bug bei Standalone Lösung

Es gibt aktuell noch folgenden Issue: (Stand 05.11.2023)
Problem:
In der Standalone Version beendet sich der spoolman Service wenn er mit --user Parameter unter dem User pi gestartet wird.
Lösung:

Links zum Issue
(root user notwendig)

```
systemctl --user stop Spoolman
systemctl --user disable Spoolman
sudo su
cd /home/pi/.config/systemd/user/
mv Spoolman.service /etc/systemd/system/
nano /etc/systemd/system/Spoolman.service
```

Folgendes unter [Service] eintragen:

```
User=pi
Group=pi
```

Dienst speichern und neustarten:

```
systemctl daemon-reload
systemctl enable Spoolman
systemctl start Spoolman
```

Prüfen ob es geklappt hat:

```
ps -axu |grep spoolman
```

-> pi 246703 0.3 1.9 686968 72340 ? Sl 15:55 1:28 /home/pi/Spoolman-0.13.1/.venv/bin/python /home/pi/Spoolman-0.13.1/.venv/bin/uvicorn spoolman.main:app --host 0.0.0.0 --port 7912

Revision #37
Created 5 November 2023 18:38:05 by Robin
Updated 13 November 2024 14:46:44 by Cryd