Selfhosting music

A guide on how to selfhost music, from start to finish, trying to be as easy to understand!

Step 0: Prerequisites

For this guide, I will use Navidrome (since it is what I use currently). I will guide you through how to set it up as a server rather than an app, as this is also how I use it, but also because, installing it as an app, especially on Windows, shouldn't be hard.

You'll need a server. This can be anything that can run Linux on it. Old PC, laptop, SBC, anything really.
Navidrome is available for: amd64, armv6, armv7 and arm64. If you don't understand what these mean, you probably have an amd64 device. If it is a very old device, like 10-15+ years old, it might be i686, and Navidrome probably won't work. The device doesn't need to be very powerful, it can even run on a Pi Zero.
Your server will also have to have remote access set up. A guide on how to setup and secure SSH is coming!

You will have to install Docker on your server. Since I don't know which Linux distribution you are running, I can't help with specific commands to run, but here is the Docker documentation on how to install it. (~~explain what docker is?~~)

You will also need to install Docker Compose, which is a way to define and run multiple containers using a single configuration file.

Optionally, if you want remote access (i.e. access outside of your network), you will also need:

- a domain name
- nginx, as reverse proxy
- certbot, to handle ssl certificates

You will also need patience and/or motivation if you aren't familiar with selfhosting. Selfhosting isn't easy but very much worth it!

Step 1: Running Navidrome

As mentionned before, I am going to explain how to run Navidrome using Docker Compose. If you have trouble following some steps, I will link to short videos showing how to do them.

Start by creating a folder named `navidrome` (or naything you like!) and cd into it. video

Edit a file named `docker-compose.yml` and paste in this inside:

```
services:
  navidrome:
    image: deluan/navidrome:latest
    user: 1000:1000
    ports:
      - "4533:4533" # change it to `127.0.0.1:4533:4533` after you've set up the reverse proxy, if you plan on setting one up
    restart: unless-stopped
    environment:
      ND_SCANSCHEDULE: 1h # at what frequency navidrome should scan for new content. set to 0 to disable
      ND_LOGLEVEL: info
      ND_SESSIONTIMEOUT: 24h
      ND_BASEURL: "https://music.example.com" # change this to the url you want navidrome to be accessible at. remove this line if no remote access is planned
    volumes:
      - "./data:/data" # where navidrome's config etc. will be stored
      - "../music:/music:ro" # where we will put our music. `ro` means read-only so navidrome cannot edit/remove files
```

video

There are many, many more configuration options that you can find here.
Don't forget to create the folder you plan on putting your music in before starting Navidrome or you will have issues.

After that, run `sudo docker compose up -d` and `sudo docker compose logs -f`. The first command start the containers required for navidrome (which is just navidrome in this case) and the second one prints the containers logs in the terminal, `-f` meaning *follow*, so keep printing new logs as they arrive. video

If everything went correctly, Navidrome should now be accessible at `http://<server ip>:4533`! If not, read the logs and troubleshoot yourself! (~~jk, I will add some troubleshooting steps!~~)
It should ask you to create an admin user, enter a username and password (a strong one if you plan on having remote access!) and confirm.

You should now be on Navidrome's home page, which is recently added albums.

Step 2: Remote access

If you don't care about remote access, you can skip that part.

Step 3: Adding music

We need to place our music in the directory we specified in the `docker-compose.yml` file, in our case `../music`, which is most likely `~/music`. If you have the music you want to upload on the device you are accessing your server with, you can log out of SSH, and then run something like this: `scp -r <local folder> <username>@<server IP>:~/music`
`scp` (secure copy protocol) transfers files through ssh, `-r` means *recursive* and is required to transfer directories. `<local folder>` is the folder on the local device you want to transfer to your server. `<username>` is the remote username, `<server IP>` is the remote ip, the same one you use to connect to your server with ssh. `:~/music` means transfer the local files to the `~/music` folder.

Depending on the size of your library, this might take a while...

After the transfer is finished, you can click on the  icon in the top right,

and then on either of these icons  (left one is quick scan, right one is full scan). Music should now have appeared and you can click on play on any of these and it should just work!™

Step 4: Choosing a client

Now to listen to music, you can either directly listen to it through the website, or better, through clients. There are many, *many*, clients available and chosing one is dependent on personal preferences.

The clients I use are:

- Symfonium for Android (paid, ~5$ one time payment. one of the best apps I ever purchased)
- Feishin for Desktop, be it windows or linux.

To connect these clients to your server, either you input the url of the server or `<IP>:4533` and input your username and password. And playback should just work!™