

# Firefish!

## 1. Install dependencies

Make sure that you can use the `sudo` command before proceeding.

### Utilities

```
sudo apt update
sudo apt install build-essential python3 curl wget git lsb-release
```

### Node.js and pnpm

Instructions can be found at [this repository](#).

```
NODE_MAJOR=20
curl -fsSL "https://deb.nodesource.com/setup_${NODE_MAJOR}.x" | sudo -E bash -
sudo apt install nodejs

# check version
node --version
```

You also need to enable `pnpm`.

```
sudo corepack enable
corepack prepare pnpm@latest --activate

# check version
pnpm --version
```

### PostgreSQL and PGroonga

PostgreSQL install instructions can be found at [this page](#).

```
sudo sh -c 'echo "deb https://apt.postgresql.org/pub/repos/apt $(lsb_release -cs)-pgdg main" >
/etc/apt/sources.list.d/pgdg.list'
wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -
sudo apt update
sudo apt install postgresql-16

sudo systemctl enable --now postgresql

# check version
psql --version
```

PGroonga install instructions can be found at [this page](#).

```
wget "https://apache.jfrog.io/artifactory/arrow/$(lsb_release --id --short | tr 'A-Z' 'a-z')/apache-arrow-apt-source-
latest-$(lsb_release --codename --short).deb"
sudo apt install "./apache-arrow-apt-source-latest-$(lsb_release --codename --short).deb"
wget "https://packages.groonga.org/debian/groonga-apt-source-latest-$(lsb_release --codename --short).deb"
sudo apt install "./groonga-apt-source-latest-$(lsb_release --codename --short).deb"
sudo apt update
sudo apt install postgresql-16-pgdg-pgroonga

rm "apache-arrow-apt-source-latest-$(lsb_release --codename --short).deb" "groonga-apt-source-latest-
$(lsb_release --codename --short).deb"
```

# Redis

Instructions can be found at [this page](#).

```
curl -fsSL https://packages.redis.io/gpg | sudo gpg --dearmor -o /usr/share/keyrings/redis-archive-keyring.gpg
echo "deb [signed-by=/usr/share/keyrings/redis-archive-keyring.gpg] https://packages.redis.io/deb $(lsb_release
-cs) main" | sudo tee /etc/apt/sources.list.d/redis.list
sudo apt update
sudo apt install redis

sudo systemctl enable --now redis-server
```

```
# check version
redis-cli --version
```

## FFmpeg

```
sudo apt install ffmpeg
```

## 2. Set up a database

1. If you forgot the password you typed, you can reset it by executing `sudo -u postgres psql -c "ALTER USER firefish PASSWORD 'password';"`.

Create a database

```
sudo -u postgres createdb --encoding='UTF8' --owner=firefish firefish_db
```

Create a database user

```
sudo -u postgres createuser --no-createdb --no-createrole --no-superuser --encrypted --pwprompt firefish
```

Enable PGronnga extension

```
sudo -u postgres psql --command='CREATE EXTENSION pgroonga;' --dbname=firefish_db
```

## 3. Configure Firefish

Create an user for Firefish and switch user

```
sudo useradd --create-home --user-group --shell /bin/bash firefish
sudo su --login firefish

# check the current working directory
# the result should be /home/firefish
pwd
```

Install Rust toolchain

Instructions can be found at [this page](#).

```
curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh
. "${HOME}/.cargo/env"

# check version
cargo --version
```

Clone the Firefish repository

```
git clone --branch=main https://firefish.dev/firefish/firefish.git
```

Copy and edit the config file

```
cd firefish
cp .config/example.yml .config/default.yml
nano .config/default.yml
```

```
url: https://your-server-domain.example.com # change here
port: 3000

db:
  host: localhost
  port: 5432
  db: firefish_db
  user: firefish
  pass: your-database-password # and here
```

## 4. Build Firefish

Build

```
pnpm install --frozen-lockfile
NODE_ENV=production NODE_OPTIONS='--max-old-space-size=3072' pnpm run build
```

Execute database migrations

```
pnpm run migrate
```

exit

# 5. Preparation for publishing a server

## 1. Set up a firewall

To expose your server securely, you may want to set up a firewall. We use [ufw](#) in this instruction.

```
sudo apt install ufw
# if you use SSH
# SSH_PORT=22
# sudo ufw limit "${SSH_PORT}/tcp"
sudo ufw default deny
sudo ufw allow 80
sudo ufw allow 443
sudo ufw --force enable

# check status
sudo ufw status
```

## 2. Set up a reverse proxy

In this instruction, we use [Caddy](#) to make the Firefish server accesible from internet. However, you can also use [Nginx](#) if you want ([example Nginx config file](#)).

Install Caddy

```
sudo apt install debian-keyring debian-archive-keyring apt-transport-https
curl -1sLf 'https://dl.cloudsmith.io/public/caddy/stable/gpg.key' | sudo gpg --dearmor -o
/usr/share/keyrings/caddy-stable-archive-keyring.gpg
curl -1sLf 'https://dl.cloudsmith.io/public/caddy/stable/debian.deb.txt' | sudo tee /etc/apt/sources.list.d/caddy-
```

```
stable.list
sudo apt update
sudo apt install caddy

# check version
caddy version
```

Replace the config file

```
sudo mv /etc/caddy/Caddyfile /etc/caddy/Caddyfile.bak
sudo nano /etc/caddy/Caddyfile
```

```
your-server-domain.example.com {
  reverse_proxy http://127.0.0.1:3000

  log {
    output file /var/log/caddy/firefish.log
  }
}
```

Restart Caddy

```
sudo systemctl restart caddy
```

## 6. Publish your Firefish server

Create a service file

```
sudo nano /etc/systemd/system/firefish.service
```

```
[Unit]
Description=Firefish daemon
Requires=redis.service caddy.service postgresql.service
After=redis.service caddy.service postgresql.service network-online.target

[Service]
Type=simple
User=firefish
```

```
Group=firefish
UMask=0027
ExecStart=/usr/bin/pnpm run start
WorkingDirectory=/home/firefish/firefish
Environment="NODE_ENV=production"
Environment="npm_config_cache=/tmp"
Environment="NODE_OPTIONS=--max-old-space-size=3072"
# uncomment the following line if you use jemalloc (note that the path varies on different environments)
# Environment="LD_PRELOAD=/usr/lib/x86_64-linux-gnu/libjemalloc.so.2"
StandardOutput=journal
StandardError=journal
SyslogIdentifier=firefish
TimeoutSec=60
Restart=always

CapabilityBoundingSet=
DevicePolicy=closed
NoNewPrivileges=true
LockPersonality=true
PrivateDevices=true
PrivateIPC=true
PrivateMounts=true
PrivateUsers=true
ProtectClock=true
ProtectControlGroups=true
ProtectHostname=true
ProtectKernelTunables=true
ProtectKernelModules=true
ProtectKernelLogs=true
ProtectProc=invisible
RestrictNamespaces=true
RestrictRealtime=true
RestrictSUIDSGID=true
SecureBits=noroot-locked
SystemCallArchitectures=native
SystemCallFilter=~@chown @clock @cpu-emulation @debug @ipc @keyring @memlock @module @mount
@obsolete @privileged @raw-io @reboot @resources @setuid @swap
SystemCallFilter=capset pipe pipe2 setpriority
```

[Install]

WantedBy=multi-user.target

## Start Firefish

```
sudo systemctl enable --now firefish
```

---

Revision #3

Created 22 September 2024 10:01:12 by Svetsen

Updated 22 September 2024 10:03:35 by Svetsen