

Installation from binary



- [Install Forgejo and Git, create git user](#)
 - [Create directories Forgejo will use](#)
 - [Optional: Set up database](#)
 - [Install systemd service for Forgejo](#)
 - [Forgejo's web-based configuration](#)
 - [Further configuration in Forgejo's app.ini](#)
 - [General hints for using Forgejo](#)
-

Install Forgejo and Git, create git user

“ [!note] Note This guide uses git.example.com and x.y.z for illustrative purpose. Replace with your domain and the used Forgejo version as appropriate.

First, download the Forgejo binary for your CPU architecture and verify the GPG signature, as described on [the Forgejo download page](#).

Next, copy the downloaded Forgejo binary to `/usr/local/bin/` (renaming it to "forgejo") and make it executable:

```
sudo cp forgejo-x.y.z-linux-amd64 /usr/local/bin/forgejo
sudo chmod 755 /usr/local/bin/forgejo
```

Make sure `git` and `git-lfs` are installed on your system. On Debian GNU/Linux you can use:

```
sudo apt install git git-lfs
```

Create a user `git` on the system. Forgejo will run as that user, and when accessing git through SSH (which is the default), this user is part of the URL (for example in `git clone git@git.example.com:YourOrg/YourRepo.git` the `git` before the `@` is the user you'll create now). On **Debian, Ubuntu** and their derivatives that's done with:

```
sudo adduser --system --shell /bin/bash --gecos 'Git Version Control' \
--group --disabled-password --home /home/git git
```

On **Linux distributions not based on Debian/Ubuntu** (this should at least work with Red Hat derivatives like Fedora, CentOS etc.), run this instead:

```
sudo groupadd --system git
sudo useradd --system --shell /bin/bash --comment 'Git Version Control' \
--gid git --home-dir /home/git --create-home git
```

Create directories Forgejo will use

Now create the directories Forgejo will use and set access permissions appropriately:

```
sudo mkdir /var/lib/forgejo
sudo chown git:git /var/lib/forgejo && sudo chmod 750 /var/lib/forgejo
```

This is the directory Forgejo will store its data in, including your Git repositories.

```
sudo mkdir /etc/forgejo
sudo chown root:git /etc/forgejo && sudo chmod 770 /etc/forgejo
```

This is the directory Forgejo's config, called `app.ini`, is stored in. **Initially it needs to be writable by Forgejo**, but after the installation you can make it read-only for Forgejo because then it shouldn't modify it anymore.

Optional: Set up database

When using sqlite as Forgejo's database, nothing needs to be done here.

If you need a more powerful database, you can use MySQL/MariaDB or PostgreSQL (apparently sqlite is good enough for at least 10 users, but might even suffice for more).

See [Forgejo's Database Preparation guide](#) for setup instructions.

Install systemd service for Forgejo

Forgejo provides a [systemd service script](#). Download it to the correct location:

```
sudo wget -O /etc/systemd/system/forgejo.service
https://codeberg.org/forgejo/forgejo/raw/branch/forgejo/contrib/systemd/forgejo.service
```

If you're *not* using sqlite, but MySQL or MariaDB or PostgreSQL, you'll have to edit that file (`/etc/systemd/system/forgejo.service`) and uncomment the corresponding `Wants=` and `After=` lines. Otherwise it *should* work as it is.

After adding or modifying a systemd service file, you will need to reload systemd.

```
sudo systemctl daemon-reload
```

Now enable and start the Forgejo service, so you can go on with the installation:

```
sudo systemctl enable forgejo.service
sudo systemctl start forgejo.service
```

Forgejo's web-based configuration

Access the Forgejo web UI by opening `http://localhost:3000/` in your local web browser. Replace `localhost` with the server's local IP address if necessary.

If it doesn't work:

- Make sure the forgejo service started successfully by checking the output of:

```
sudo systemctl status forgejo.service
```

If that indicates an error but the log lines underneath are too incomplete to tell what caused it,

```
sudo journalctl -n 100 --unit forgejo.service
```

will print the last 100 lines logged by Forgejo.

You should be greeted by Forgejo's "Initial Configuration" screen. The settings should be mostly self-explanatory, some hints:

- Select the correct database (SQLite3, or if you configured something else in the "Set up database" step above, select that and set the corresponding options)
- **Server Domain** should be `git.example.com` (or whatever you're actually using), **Forgejo Base URL** should be `http://git.example.com:3000` (assuming you won't change `HTTP_PORT` to a different value than 3000)
- Check the **Server and Third-Party Service Settings** settings for settings that look relevant for you.
- It may make sense to create the administrator account right now (**Administrator Account Settings**), even more so if you disabled self-registration.
- Most settings can be changed in `/etc/forgejo/app.ini` later, so don't worry about them too much.

Once you're done configuring, click `Install Forgejo` and a few seconds later you should be on the dashboard (if you created an administrator account) or at the login/register screen, where you can create an account to then get to the dashboard.

So far, so good, but we're not quite done yet - some manual configuration in the `app.ini` is needed.

Further configuration in Forgejo's app.ini

Stop the forgejo service:

```
sudo systemctl stop forgejo.service
```

While at it, make `/etc/forgejo/` and the `app.ini` read-only for the git user (Forgejo doesn't write to it after the initial configuration):

```
sudo chmod 750 /etc/forgejo && sudo chmod 640 /etc/forgejo/app.ini
```

Now, (as root), edit `/etc/forgejo/app.ini`.

“ [!note] Note You'll probably find the [Configuration Cheat Sheet](#) and the [Example app.ini](#) that contains all options incl. descriptions helpful.

The following changes are recommended if dealing with many large files:

- Forgejo allows uploading files to Git repositories through the web interface. By default the **file size for uploads** is limited to 3MB per file, and 5 files at once. To increase it, under the `[repository]` section, add a `[repository.upload]` section with a line like `FILE_MAX_SIZE = 4095` (that would be 4095MB, about 4GB) and `MAX_FILES = 20`. It'll look somewhat like this:

```
...
[repository]
ROOT = /var/lib/forgejo/data/forgejo-repositories

[repository.upload]
;; max size for files to the repo via web interface, in MB,
;; defaults to 3 (this sets a limit of about 4GB)
FILE_MAX_SIZE = 4095
;; by default 5 files can be uploaded at once, increase to 20
MAX_FILES = 20

[server]
...
```

Similar restrictions exist for attachments to issues/pull requests, configured in the

`[attachment]` [sections](#) `MAX_SIZE` (default 4MB) and `MAX_FILES` (default 5) settings.

- By default **LFS data uploads expire** after 20 minutes - this can be too short for big files, slow connections or slow LFS storage (git-lfs seems to automatically restart the upload then - which means that it can take forever and use lots of traffic).. If you're going to use LFS with big uploads, increase this limit, by adding the line `LFS_HTTP_AUTH_EXPIRY = 180m` (for 180 minutes) to the `[server]` section.
- Similarly, there are timeouts for all kinds of git operations, that can be too short. Increasing all those git timeouts by adding a `[git.timeout]` section below the `[server]` section:

```
;; Git Operation timeout in seconds
;; increase the timeouts, so importing big repos (and presumably
;; pushing large files?) hopefully won't fail anymore
[git.timeout]
DEFAULT = 3600 ; Git operations default timeout seconds
MIGRATE = 6000 ; Migrate external repositories timeout seconds
MIRROR  = 3000 ; Mirror external repositories timeout seconds
CLONE   = 3000 ; Git clone from internal repositories timeout seconds
PULL    = 3000 ; Git pull from internal repositories timeout seconds
GC       = 600  ; Git repository GC timeout seconds
```

The example values are increased by a factor of 10 for illustration. Increasing all timeouts, particularly by this magnitude, may not be necessary. Evaluate your specific performance requirements and adjust values accordingly.

- By default LFS files are stored in the filesystem, in `/var/lib/forgejo/data/lfs`. In the `[lfs]` section you can change the `PATH = ...` line to store elsewhere, but you can also configure Forgejo to store the files in an S3-like Object-Storage.
- If you want to use the systemwide sendmail, enable sending E-Mails by changing the `[mailer]` section like this:

```
[mailer]
;; send mail with systemwide "sendmail"
ENABLED = true
PROTOCOL = sendmail
FROM = "Forgejo Git" <noreply@yourdomain.com>
```

- By default, Forgejo will listen on the port 3000, but that can be [changed to 80 with `HTTP\_PORT`](#) like this:

```
[server]
HTTP_PORT = 80
```

When you're done editing the app.ini, save it and start the Forgejo service again:

```
sudo systemctl start forgejo.service
```

You can test sending a mail by clicking the user button on the upper right of the Forgejo page ("Profile and Settings"), then `Site Administration`, then `Configuration` and under `Mailer Configuration` type in your mail address and click `Send Testing Email`.

General hints for using Forgejo

Sometimes you may want/need to use the Forgejo [command line interface](#). Keep in mind that:

- You need to **run it as the `git` user**, for example with:

```
$ sudo -u git forgejo command --argument
```

- You need to specify the **Forgejo work path**, either with the `--work-path /var/lib/forgejo` (or `-w /var/lib/forgejo`) commandline option or by setting the `FORGEJO_WORK_DIR` environment variable before calling `forgejo`:

```
$ export FORGEJO_WORK_DIR=/var/lib/forgejo
```

- You need to specify the path to the config (app.ini) with either `--config /etc/forgejo/app.ini` or `-c /etc/forgejo/app.ini`.

So all in all your command might look like:

```
$ sudo -u git forgejo -w /var/lib/forgejo -c /etc/forgejo/app.ini admin user list
```

“**For convenience**, you could create a `/usr/local/bin/forgejo.sh` with the following contents:

```
#!/bin/sh
sudo -u git forgejo -w /var/lib/forgejo -c /etc/forgejo/app.ini "$@"
```

and make it executable:

```
sudo chmod 755 /usr/local/bin/forgejo.sh
```

Now if you want to call `forgejo` on the commandline (for the default system-wide installation in `/var/lib/forgejo`), instead of the long line shown above, use:

```
$ forgejo.sh admin user list
```

You can always call `forgejo` and its subcommands with `-h` or `--help` to make it output usage information like available options and (sub)commands, for example to show available subcommands to administrate users on the commandline:

```
$ forgejo admin user -h
```

📄 #4

📄 cocobook-sync 📄 2026-10-28 16:40:55 UTC

📄 cocobook-sync 📄 2026-10-29 03:45:43 UTC