Contents

Chapter 1

Ginger Manual

1.1 License

ginger is written by Josef Hahn under the terms of the GPLv3.

Please read the `LICENSE` file from the package and the Dependencies section for included third-party stuff.

1.2 About

Ginger is a news reader for rss news feeds. Once it is hosted in a web server, a user can login and read her news in the web browser.

Features:

- multi-user web application
- multiple news feeds per user
- message tagging
- powerful message filtering
- very customizable
- scripting interface

It is based on Django and Python and is real-world tested to run in Apache with the wsgi module on Debian Linux.

Are you currently reading from another source than the homepage? Are you in doubt if that place is up-to-date? If yes, you should visit `https://ginger.ws/wiki/pino/projs/ginger` and check that. You are currently reading the manual for version 1.0.490.

1.3 Maturity

In this version, the state of ginger is considered as production-stable.

Dependencies

There are external parts which are used by ginger. Many thanks to the projects and all participants.

`Python 2.7` *required*

`Django 1.6` *required*

`python-feedparser` *required*

`Typical Linux Desktop` *recommended (maybe has alternatives)*

`apache 2.2` *recommended (maybe has alternatives)* : as web server

`mod_wsgi` *recommended (maybe has alternatives)* : apache module

`jquery` *included* : licensed under terms of `GPLv2`.

`font 'Source Sans Pro'` *included* : license `OFL`; copied from `here`.

`banner image` *included* : _meta/homepage_bannerimage.png; license `public domain`; copied from `here`.

1.4 User Documentation

1.4.1 Install Ginger

You have a choice how to install ginger:

- Install a package for your operating system, if available.
- Download the source package (and some more) and start the Django development server. This is not recommended for production.
- Download the source package (and some more) and embed the application into your web server. This is the most complicated way.

Those are now explained in detail... After installation, follow-up in First Login.

1.4.1.1 Operating System Package

Nothing much to say here. Download and install it :-) It should come up with further instructions.

1.4.1.2 Using Django Development Server

Since ginger is a Django application, you can use the small and easy test & development server from Django. Please download the ginger source package and install all the required Dependencies.

Open a terminal and unpack ginger to its destination (e.g. `/home/pino/ginger_test`). Change to this destination folder and execute this:

```
$ ./manage.py syncdb --noinput
```

Once you have done so, a database is initialized. You can now (and later on) run the server by calling:

```
$ ./manage.py runserver
```

The output points to a web address. Put it into your browser and follow the instructions there. This procedure is very common for Django applications and you should find more on the web.

1.4.1.3 Embed Ginger In Existing Web Server

Please download the ginger source package and install all the required Dependencies. Unpack ginger to a good destination (e.g. `/usr/lib/ginger`). Also create a folder for runtime data (e.g. /var/lib/ginger).

Afterwards, create a `/usr/lib/ginger/ginger/settings_local.py` with this content:

```
DEBUG = False
TEMPLATE_DEBUG = DEBUG
SECRET_KEY = 's1e2c3r4e5t6s7e8c9r0e1t2s3e4c5r6e7t8'
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': 'var/lib/ginger/ginger-db',
        'USER': '',
        'PASSWORD': '',
        'HOST': '',
        'PORT': '',
    }
}
STATIC_ROOT = "/var/lib/ginger/static/"
```

Change `SECRET_KEY` to something long, random and secret. Change settings in `DATABASES` to your needs, e.g. for using another database engine.

Afterwards, run this on a terminal:

```
$ ./manage.py syncdb --noinput
$ ./manage.py collectstatic --noinput
```

Embed ginger as wsgi application into an Apache2. Use `_meta/misc/apache2.conf` and `_meta/misc/ginger.wsgi` for inspiration. Using other web servers should work as well somehow :-p

First Login

Visit the Ginger root url (e.g. `http://localhost:8000`) and follow the instrustions there.

It should create an admin user for you and logs you in with that user.

1.4.1.4 What's Next?

Ginger is a multi-user tool with an own user database.

User configuration must be made with the admin interface. You can create, remove and modify users (reset password, ...) there. The admin user can see it in the menu bar.

Create some users there for your friends/colleagues/clients. Give them your Ginger url and the user data. Let them insert theirs favorite feeds and read the messages there.

Or simply use Ginger with the admin account on a single-user machine :)

1.4.2 Advanced

1.4.2.1 External authentication

Ginger can use an external authentication method instead of the internal user database. Write an executable (e.g. a bash script) with the following behaviour: Read one line from stdin => this is a user name Read one line from stdin => this is the password Check that for correctness and print the username if and only if login data are correct Terminate

The returned username should be a modified version of the given one if it may contain problematic characters (like `@`, `%`, ...). The easiest solution is to just remove them. The executable must be allowed to be executed by the web server.

Write a line like this in your `ginger/settings_local.py`:

```
EXTERNAL_AUTH_HELPER = '/path/to/your/auth_helper'
```

Note that the admin interface will still use the internal database! Since you don't need it in this case, you should hide it in your server configuration or at least reset the default admin password!