ginger - Manual

Release 2.0.1379

Josef Hahn

Jul 26, 2020

CONTENTS

CHAPTER

ONE

LICENSE

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

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

CHAPTER

TWO

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.

CHAPTER

THREE

UP-TO-DATE?

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://pseudopolis.eu/wiki/pino/projs/ginger and check that. You are currently reading the manual for version 2.0.1379.

CHAPTER

FOUR

MATURITY

ginger is in production-stable state.

CHAPTER

FIVE

DEPENDENCIES

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

 *Python 3.4*, required

 *Django 1.8*, required

 *Typical GNU/Linux Desktop*, recommended

 *python-feedparser*, required

 *apache 2.4*, recommended : as web server

 *mod_wsgi*, recommended : apache module

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

 *font 'Symbola'*, included : for logo symbol; free for use; copied from here.

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

 *all files in /_meta*, included : if not mentioned otherwise, Copyright 2015 Josef Hahn under license CC BY-SA 3.0 license.

CHAPTER

SIX

FIRST STEPS

Please read how to make Ginger ready for the first steps in *Appendix: Installation*.

The following steps require that all preparation steps are finished. In order to login, visit the Ginger root url (e.g. `http://localhost:8000`) in your favorite browser and follow the instructions there.

Depending on your installation procedure, the initial admin user is directly preconfigured, so you can directly login and add some feeds.

6.1 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 :)

CHAPTER

SEVEN

HOW TO USE GINGER

7.1 Screen Structure

When you are logged in, you see the Ginger main application. It always has a top bar showing available actions. Right after login, you get the default screen, which shows a filter panel on the left side and a message list on the right side. If you use a mobile device, switch between them instead (see top bar).

The top bar also includes the *Feeds* and *Settings* view. While the latter is interesting for experts, the former one is the first step with a new account.

7.2 Feeds View

Ginger works by fetching news messages from some sources. Those sources have to comply to the rss or atom (or some more) standard. They are addressed by a http(s) url that you typically can find on the homepage of a news provider. The feeds view is used for adding those sources to your Ginger account.

7.3 Settings View

The settings view allows to customize some aspects of Ginger. Feel free to take a look there. Most things should be easy to understand.

7.4 Default View

The default view is made for browsing all the news that came from your feeds so far. It allows to attach tags to your messages for querying them later. One feed-specific tag is assigned automatically to each message for convenience.

In mobile device mode, you can switch between the list of your messages and between a panel that allows to filter this list (in order to make searches). In large mode, you see both.

7.4.1 Message List

Each message builds one box. You can expand them by selecting them. All actions on a message are available from here. You can remove them, mark them as favorite and manage it's tags. You can also open the corresponding web page in a new browser tab.

Some keyboard support is available (when the message list has the focus):

- `ArrowUp`/`ArrowDown`: navigation
- `ArrowRight`: open website for message
- `Delete`: delete selected message
- `Plus`: expand/collapse the selected box
- `F`: mark/unmark selected message as favorite
- `T`: edit tags for the selected message

7.4.2 Filter Panel

By default, the message list shows all available messages. The filter panel allows to restrict that list by selecting some filter criteria.

You see a summary of the current filter at the bottom and a cloud for picking filter criteria above.

For instance, click on a tag to restrict your view to that tag. Select multiple ones to get an even smaller message list. Clear filter criteria for returning to a full list.

Advanced Filtering

If you have unlocked the *advanced* features in *Settings*, you can do some more filtering.

You can append additional or-connected filter rule lists (like '(tag A and tag B) *or* (tag X and tag Y and tag Z)').

It is also possible to negate most filter criteria by clicking on them in the filter summary view.

Store Filters

You can store you complete filter with a meaningful name and load it again in later sessions. You can remove unneeded stored filters in the *Settings*.

CHAPTER

EIGHT

APPENDIX: EXTERNAL AUTHENTICATION

Ginger can use an external authentication method instead of the internal user database. Write an executable (e.g. a shell script) with the following behavior:

- 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!

CHAPTER

NINE

APPENDIX: INSTALLATION

Install Ginger via the installation package for your environment, if a suitable one exists for download. This also takes care of installing dependencies and doing preparation (unless mentioned otherwise in the installation procedure). Use the source code archive as fallback. Please read more details about your installation flavour in the following.

Installation on Debian/Ubuntu

Just download the Ginger package for Debian/Ubuntu and install it. After the installation, the start menu contains a browser link to an automatically configured (apache2/wsgi-based) Ginger site. Try the *First Steps*.

Installation as Python wheel package

You need a Python environment with enabled 'wheel' functionality. Download the Ginger wheel package and install it. Proceed with *Appendix: Setting Up Ginger*.

Installation on MS Windows

Just download the Ginger package for Windows and install it. After the installation, the start menu contains a link for starting a personal Ginger server and a browser link to that site. Try to start the server and do the *First Steps*.

Installation From Source

Download the Ginger source package and extract it to a destination. Install the *Dependencies* and proceed with *Appendix: Setting Up Ginger* afterwards.

CHAPTER

TEN

APPENDIX: SETTING UP GINGER

For some platforms, the installation automatically sets up a Ginger service. If so, those are automatically enabled in background on some platforms and to be explicitly started by the user on other ones. The degree of service quality (in regards of performance, security, ...) can also differ between platforms. Read more about your scenario in *Appendix: Installation*.

If your installation package does automatically install a Ginger service and you are fine with this one, you can skip this preparation step, ensure that the service is started, and proceed with *First Steps*.

Otherwise there are many options to set up a Ginger service manually. The easiest way is to use the included personal Ginger server (technically it uses the Django development server). Other ways are to let Ginger run in a full web server. Please find out all existing possibilities in the Django documentation. Real web servers typically provide much higher service quality.

The first step for any installation way is to run the Ginger admin tool. Open a terminal window. You need to have write permissions for the Ginger installation directory for the following steps.

Finding ginger_admin: Now you have to find out the path to `ginger_admin`. For some platforms, the name can be used directly, but on other ones (e.g. if you use the source package) you have to find it somewhere in the Ginger installation directory; typically in `.../_meta/misc/ginger_admin.py` or likewise.

Call this tool this way for setting up a brand new Ginger installation:

```
ginger_admin setup "/home/foo/.ginger"
```

The second parameter specifies a storage location for Ginger runtime data like the database files.

Afterwards you can start the included personal server this way and browse to the printed destination:

```
ginger_admin runserver
```

This way you could - at least for the moment - proceed with *First Steps*.

For public servers you have to run Ginger in a real web server.

The following gives some short hints for installing Ginger in Apache 2.4 with `mod_wsgi`. If you plan to use a different software stack, you should also read the Django documentation. You probably need certain write permissions during the process.

Finding the local settings file: Find the file `settings_local.py`, which is typically stored in the Ginger installation directory or in its `ginger` subdirectory after you set up Ginger.

The `settings_local.py` must be adapted. Adapt the following settings, but don't remove unlisted ones; just ignore them!:

```
DEBUG = False  # at least when everything works
DATABASES = ...  # as it was or with another database
```

(continues on next page)

(continued from previous page)

```
STATIC_ROOT = "/var/lib/ginger/static/"  # used for static files
...
```

Afterwards, run this on a terminal:

```
ginger_admin init
```

Embed Ginger as wsgi application into an Apache2. Use `_meta/misc/apache2.conf` and `_meta/misc/ginger.wsgi` for inspiration.

If you are more comfortable with a different approach to hosting Django applications, find the application in the Ginger installation directory and host it like a typical Django application without doing any `ginger_admin` calls!

CHAPTER

ELEVEN

APPENDIX: MANUALLY UPDATING THE DATABASE SCHEME

For installations without a package manager, you have to manually update the database scheme whenever a new version brings some changes in the Ginger data storage structures. Update the source code first and then call this command:

```
ginger_admin update
```

CHAPTER

TWELVE

API REFERENCE

12.1 ginger package

12.1.1 Subpackages

ginger.main package

Subpackages

ginger.main.migrations package

Submodules

ginger.main.migrations.0001_initial module

Module contents

Submodules

ginger.main.admin module

ginger.main.core module

ginger.main.crawler module

ginger.main.feedadmin module

ginger.main.htmlcleaner module

ginger.main.htmlcleaner.**_translatetag**(*s*, *linktarget*, *tree*)

ginger.main.htmlcleaner.**cleanhtml**(*s*, *linktarget*)

ginger.main.messagefilter module

ginger.main.models module

ginger.main.projectinformations module

ginger.main.tags module

ginger.main.userscripting module

class ginger.main.userscripting.**Message**(*nativemessage*)

Bases: object

addtag(*tag*)

delete()

getcreated()

getfeed()

getsummary()

gettags()

gettitle()

geturl()

removetag(*tag*)

setcreated(*v*)

setsummary(*v*)

settitle(*v*)

seturl(*v*)

ginger.main.userscripting.**execute_user_script**(*user*, *scriptname*, *params*)

Module contents

12.1.2 Submodules

12.1.3 ginger.settings module

12.1.4 ginger.urls module

12.1.5 Module contents

12.2 manage module

PYTHON MODULE INDEX

g

m

INDEX

Symbols

A

C

D

E

G

M

R

S