Eggplant Documentation

Release 0.1

The Eggplant Open Source Community & Hacklab Team

February 03, 2016

Contents

Contents:

CHAPTER 1

Installation

TODO: Move from README.rst

CHAPTER 2

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

2.1 Types of Contributions

2.1.1 Report Bugs

Report bugs at https://github.com/kbhff/eggplant/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

2.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" is open to whoever wants to implement it.

2.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "feature" is open to whoever wants to implement it.

2.1.4 Write Documentation

eggplant could always use more documentation, whether as part of the official eggplant docs, in docstrings, or even on the web in blog posts, articles, and such.

2.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/kbhff/eggplant/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

2.2 Get Started!

Ready to contribute? Here's how to set up *eggplant* for local development.

If you are completely new to source code versioning using git, please search for a video explaining it. And ask for help getting a git tool set up on your machine.

Note: We have decided to use the very conventional a simple git branching model. Read the guide to get a good introduction to Git workflows.

```
$ git clone git@github.com:kbhff/eggplant.git
```

2.2.1 Virtualenv

The project is pretty basic, these are classical just steps. Just make note that it's a Python 3.4 only project. Enter the git project folder.

```
$ pip install virtualenvwrapper
```

To get the mkvirtualenv command you need to:

```
source  /usr/local/bin/virtualenvwrapper.sh
```

On debian this file in:

```
/etc/bash_completion.d/virtualenvwrapper
```

start a new bash session to source it.

```
$ mkvirtualenv eggplantenv -p python3.4
$ workon eggplantenv
$ pip install -r requirements/development.txt
$ python manage.py syncdb
$ python manage.py runserver
```

Use "workon eggplantenv" to activate the eggplan virtual environment, and "deactivate" to exit.

This will deploy a local SQLite database and run a local webserver. If you are completely new to Django and Python, notice that you need [pip](https://pip.pypa.io/en/stable/installing.html), too.

2.2.2 Vagrant

(Optional) This takes some time (downloading) and some harddisk capacity.

Another way to get started contributing to this project is to download and install git and [Vagrant](http://vagrantup.com/), Clone the project (as mentioned above), change directory into the eggplant folder and then run the following commands:

```
$ vagrant up
$ vagrant ssh
$ python manage.py migrate
$ python manage.py test
$ python manage.py createsuperuser
$ python manage.py runserver
```

This will download and bootstrap an ubuntu 14.04 vagrant box, connect to it, start the django development server. The project should now be available at http://192.168.33.28:8000/.

2.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 3.4. https://travis-ci.org/kbhff/eggplant/pull_requests and make sure that the tests pass for all supported Python versions.

2.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_eggplant
```

Lots of people contributed to this group through the Eggplant Meetup group in Copenhagen.

http://www.meetup.com/Eggplant/

CHAPTER 3

History

3.1 0.1 (2015-08-15)

- Adding a history file for the Eggplant application

CHAPTER 4

Developer documentation

4.1 Component overview

The Eggplant application contains the following sub-applications (python packages nested in `eggplant/` containing models and migrations).

Applications inside `eggplant/` are interdependent, except `core` which should not depend on anything else and `permissions` which all other applications are expected to use.

- `accounts` - Administration and user pages related to accounts, being everything that has to do with user's transactions inside the market.
- `core` - an application with models that all other applications may depend on, **dependency of others**
- `dashboard` - Functionality for workflow of members, administrators etc.
- `departments` - Administration logic regarding a department such as shifts.
- `invitations` - Sending of invitation emails to create new user profiles that join accounts, departments etc.
- `membership` - Extends from Django allauth and contains the models of memberships and their organizations.
- `market` - Contains everything related to paying for stuff, invoicing, and creating and managing goods such as grocery bags.
- `permissions` - Models and decorators for permissions, **dependency of others**
- `profiles` - User information

4.1.1 Coupling

Currently, we expect applications to depend on each other's models. But it would get messy if templates, views, and static assets also started being interdependent.

Possible scheme: Only allow models and decorators to be interdependent, all other common elements should live in `eggplant.core`.

4.1.2 Philosophy

To quote Two Scoops of Django (1.8 version) that in turn quotes James Bennett (who in turn quotes Douglas McIlroy):

> James Bennett volunteers as both a Django core developer and as its release manager. He taught us everything that we know about good Django app design. We quote him:

> "The art of creating and maintaining a good Django app is that it should follow the truncated Unix philosophy according to Douglas McIlroy:
>
> > 'Write programs that do one thing and do it well.'
>
> In essence, each app should be tightly focused on its task. If an app can't be explained in a single sentence of moderate length, or you need to say 'and' more than once, it probably means the app is too big and should be broken up.

4.1.3 TODO

The scope of the applications may need to be consolidated.

4.2 Dependency overview

askdasdk

CHAPTER 5

Eggplant

Eggplant is an open source web application that provides simple and flexible infrastructure for organizing food coops and other local community-driven projects.

CHAPTER 6

How to contribute

Read our documentation to get started reporting bugs, developing code etc. The project description, organisation and goals are on our website eggplant.dk. The list of tickets is available on our GitHub project.

Write code, write tests, have fun.

CHAPTER 7

Get in touch

For AFK stuff, you can join us in Copenhagen on meetup.com.

We use Slack for ad-hoc communication: Click to recieve an invitation. The techical discussion takes place on Slack#teamblue. The design and organisational issues can also be raised on Slack#teamgreen.

CHAPTER 8

Indices and tables

- genindex
- modindex
- search