# django Organice Documentation

***Release 0.2***

**Peter Bittner <django@bittner.it>**

11.06.2014

Inhaltsverzeichnis

Inhalt:

KAPITEL 1

# django Organice

All-in-one collaboration solution. For non-profit organizations, sports clubs, small to medium-sized businesses. Nice, very nice, http://organice.io

django Organice is a nice way to run a collaboration platform, an intranet, and websites for your business. It's a compilation of the best Django packages, preconfigured for getting you started quickly. Being powered by the well-known Django Web framework it's a safe choice for your IT investment providing an easily extensible architecture.

Auf http://organice.io finden Sie weitere Informationen.

## 1.1 Fundamental Features

- Enables everyone to contribute to your content—while retaining your full control.
- Continually involves, helps and optimally supports your collaborators by sending out reminders, editing and submission hints to avoid losing traction or interest.
- Supports all four common collaboration and support patterns in several ways:
  1. *Active contribution* (create content, contribute corrections)
  2. *Passive contribution* (report issues, assign tasks, provide feedback)
  3. *Active consumption* (search and browse for content)
  4. *Passive consumption* (subscribe to notifications)

## 1.2 Core Components

Wesentliche Komponenten von django Organice sind:

- Django Web Framework
- leistungsstarkes Content Management (django CMS)
- leistungsstarker, vielseitiger Blog (Zinnia, cmsplugin)
- Veranstaltungskalender (via Zinnia)
- flexibles Kontaktformular (cmsplugin)
- Seiten für Linksammlungen

- leistungsstarker Newsletter (Emencia)
- todo/issue lists
- social login and user profiles (allauth)
- generic analytics support (analytical)
- Mehrsprachenfähigkeit
- Multi-Websitefähigkeit
- database support for PostgreSQL, MySQL, Oracle, and more

## 1.3 Roadmap

We are thrilled to get these awesome features out to you:

- group spaces with automatic mailing list (groupname@example.com)
- documentation area / wiki (probably by use of CMS functionality)
- discussion forum / Q&A board (Askbot, Misago)
- content download for magazine generation
- unobtrusive, integrated, transparent document and asset management
- DropBox, ownCloud, Windows OneDrive, Google Drive, Apple iCloud, etc. integration
- fully integrated search (Haystack)
- web analytics respecting your privacy (Piwik)
- live chat or chat integration

## 1.4 Wer benutzt django Organice?

Beispiele von Websites, die auf django Organice laufen:

- Rudolf Steiner Schule Kreuzlingen

## 1.5 Download and Contributions

Official repositories: (kept in sync)

1. Bitbucket: https://bitbucket.org/bittner/django-organice
2. GitHub: https://github.com/bittner/django-organice

## 1.6 Getting Help

- Dokumentation ist verfügbar unter http://docs.organice.io
- Questions? Please use StackOverflow. Tag your questions with `django-organice`.

- Found a bug? Please use either the Bitbucket or GitHub issue tracker (you choose)

KAPITEL 2

# Installation

This document assumes you are familiar with basic Python and Django development and their tools. If not, please read up on pip, virtualenv, and virtualenvwrapper first. A basic understanding is sufficient.

## 2.1 Requirements

- Python 2.6 or higher

All other dependencies are resolved by the django Organice installer.

### 2.1.1 Recommended for installation

- pip
- virtualenv
- virtualenvwrapper

## 2.2 Installing django Organice

1. We recommend preparing a virtual environment for running django Organice:

   ```
   $ mkvirtualenv example
   $ workon example
   ```

   The prompt will change to something like `(example)~$` to reflect that your new virtual environment is active.

2. The easiest way is using `pip` for installation:

   ```
   $ pip install django-organice
   ```

   This will pull the latest django Organice package from the Internet and install all dependencies automatically.

   *NOTE:* Since pip 1.5+ external repositories are deprecated and require additional command line switches:

   ```
   $ pip install django-organice --allow-external django-cms --allow-unverified django
   ```

If you're a developer you may want to run django Organice with the latest sources: (don't do this as a user)

```
$ git clone git@github.com:bittner/django-organice.git
$ cd django-organice
$ python setup.py install
```

or, alternatively, using pip:

```
$ pip install git+https://github.com/bittner/django-organice.git#egg=django-organic
```

3. Install the adapter suitable for your database (PostgreSQL `psycopg2`, MySQL `MySQL-python`, Oracle `cx_Oracle`, etc.), e.g.

   ```
   $ pip install psycopg2
   ```

   The Django project recommends PostgreSQL.

   > *NOTE:* You can skip this step if you decide to use SQLite, e.g. for evaluation purposes.

4. Run the Organice setup command to create your new project: (e.g. *example*)

   ```
   $ organice-setup example
   ```

5. Edit your settings in `example/settings/common.py`, `example/settings/develop.py`, etc. See the Django documentation on settings if you're not familiar with it. The `develop` settings are used by your project by default (local development), `common` is included in all profiles.

6. Initialize your database and start rocking:

   ```
   $ python manage.py syncdb --migrate
   $ python manage.py runserver
   ```

   You can now point your browser to http://127.0.0.1:8000/ and start developing your project locally.

   > *NOTE:* If you're planning to create your content locally make sure you use the same database engine for local development and production. Your plan of moving the whole database content from development to production will give you major headaches otherwise. And, use Sqlite for evaluating only!

## 2.3 Initial Configuration

1. Follow the instructions given to you by the django Organice installer `organice-setup` after setup has completed. You have to adapt some values in your project settings!
2. If you want your site to use a language other than English, or you want to use several languages: Adapt the values of `LANGUAGE_CODE` and `LANGUAGES`, and set `USE_I18N = True` in your project settings.
3. Add your first pages, blog posts, and newsletter data:

- Add some pages and navigation in the Django administration at Cms > Pages, and publish your changes.

- Surf your new website, and fill your new pages with content using the front-end editing feature.
- Surf to `/blog/` on your website, and start adding Blog posts.
- Add a user in the Django administration at Newsletter > Contacts.
- Add `localhost` (or appropriate server) to Newsletter > SMTP servers.
- To allow subscribing from the website (from `/newsletter/subscribe`) add a list to Newsletter > Mailing lists.
- Finally, add your first newsletter to Newsletter > Newsletters.
- For adding templates to Emencia Newsletter please consult the related section in the TinyMCE 3.x documentation.

4. For sending newsletters to work you must configure a cronjob polling on `python manage.py send_newsletter` every half an hour. If that was just Greek to you go ask your server admin for help. She knows!

## 2.4 Deployment to Production

During the installation `organice-setup` prepared 3 different environments that help you with deployment:

```
example
-- settings
|   -- __init__.py
|   -- common.py
|   -- develop.py
|   -- staging.py
|   -- production.py
```

This modularized setup is described in Solution 2 of Tommy Jarnac's blog on Django settings best practices [1]. The `develop` settings are active by default (for local development), `common` is included by all profiles.

For deployment to environments other than `develop` the settings module location must be overridden by setting the Django environment variable `DJANGO_SETTINGS_MODULE`. For example, if you use Apache as your Django web server adapt your Apache configuration file for `example` with:

```
SetEnv DJANGO_SETTINGS_MODULE example.settings.production
```

> *NOTE:* To test different settings locally you can start the Django webserver with the `--settings` option:
>
> ```
> $ python manage.py runserver --settings=example.settings.staging
> ```

[1] David Cramer from DISQUS has described a similar solution at http://justcramer.com/2011/01/13/settings-in-django/

KAPITEL 3

# Konfiguration

django Organice kommt mit vernünftigen Vorgabewerten für so gut wie Alles. Darüber hinaus können Sie sein Verhalten über Django settings anpassen, die hier aufgelistet sind.

## 3.1 Settings

Sie können jede der folgenden Optionen in den `settings` Ihres Projekts definieren, um den Vorgabewert zu übersteuern.

### 3.1.1 ORGANICE_URL_PATH_ADMIN

**Default** `admin`

The URL path for accessing the Django Administration backend (e.g. `www.example.com/admin`). Must be non-empty. Use an identifier only, no white space, no leading or trailing slash.

### 3.1.2 ORGANICE_URL_PATH_BLOG

**Default** `blog`

The URL path for the blog's start page (e.g. `www.example.com/blog`). Must be non-empty. Use an identifier only, no white space, no leading or trailing slash.

### 3.1.3 ORGANICE_URL_PATH_NEWSLETTER

**Default** `newsletter`

The URL path for accessing newsletter functionality on the front-end (e.g. `www.example.com/newsletter`). Must be non-empty. Use an identifier only, no white space, no leading or trailing slash.

### 3.1.4 ORGANICE_URL_PATH_TODO

**Default** `todo`

The URL path for accessing todo list functionality on the front-end (e.g. `www.example.com/todo`). Must be non-empty. Use an identifier only, no white space, no leading or trailing slash.

## 3.2 Third Party Settings

### 3.2.1 Analytics Providers

The analytics services supported by `django-analytical` are enabled by setting various service properties in your settings file. The properties are documented in their documentation. For security reasons you shouldn't add those to your common settings file, but to settings/production.

KAPITEL 4

# Themes

One of the nice things of django Organice is that themes are handled as separate projects. In fact, they are pluggable Django apps composed of assets and templates that you can simply install and activate.

## 4.1 Official Themes

Here is a list of django Organice themes officially supported by us:

1. RSSK Theme: django-organice-theme-rssk
2. Fullpage Theme: django-organice-theme-fullpage

If you have a nice theme and would like to include it in this list let us know by e-mail or make a pull request on this page of the documentation.

### 4.1.1 Mother Theme

`django-organice-theme` is the mother of all themes for django Organice. This theme is installed automatically when you install django Organice. From the development perspective all themes are derived from the mother theme, which contains a collection of static files (assets) and templates, as well as a `Makefile` for asset management. The mother theme is composed of:

- bootstrap-sass (Sass version of Twitter Bootstrap v3)
- Compass (CSS authoring framework using Sass)
- UglifyJS v2 (JavaScript minifier)

The `Makefile` also supports you with updating those components on your development system.

## 4.2 Rolling Your Own Theme

Preparations:

- Visit http://organice.io/themes and find a theme that is as close as it gets of what you want.
- Go to that theme's repository page, make a copy of the whole project, and rename it (e.g. to `mytheme`).

Loop until you're happy:

- Add or adapt the style sheet (`.scss`), JavaScript (`.js`), and other files in `mytheme/static/`.
- Run `make assets` in order to compile the Sass files to CSS, and combine and minify both CSS und JavaScript.
- Adapt the template files in `mytheme/templates/`, and test the results on your development system.

KAPITEL 5

# User Manual

django Organice is composend of the following main components:

1. Content Management (Cms)
2. Blog (Zinnia)
3. Newsletter

## 5.1 Content Management

Editing your website is not much different from surfing the web. When you're logged in to your website you will have the django CMS toolbar on top of the page. Slide the "Edit mode" button to the "ON" state, and some elements of the page you're currently on start having additional toolbars to add and edit content in those areas.

You can safely modify content on the page without the website to change. Only you can see the changes, and only as long as you are in Edit mode. When you're happy with the changes press the blue "Publish" button on the django CMS toolbar to make your version visible to the world.

For more technical tasks like creating a navigation structure, adding pages, etc. the django CMS toolbar redirects you to the Django administration interface. Want to have an upfront first impression before using it? Check out the (slightly outdated) django CMS video on frontend-editing.

## 5.2 Blog

Writing and publishing interesting articles now and then is called "blogging". It's very similar to the usual editing of your website, but still it's different because you get more features as a natural add-on: Categories, tagging, comments, publication dates, archives (yearly, monthly, daily), related entries, RSS feeds, etc.

On your website you simply go to the Blog area, and click on the "Add" link, which is available when you're logged in. In the blog entry editor you have all functionality you already know from the content management section. Modifying existing blog entries works identical to the usual content changes on your website with the exception of publishing, which is not available with an intermediate draft step: All your changes are immediately visible online.

Note that in django Organice we also use the blog functionality for other use cases that are very similar to the usual blogging, such as event agendas, job postings, press releases, download lists, etc.

## 5.3 Newsletter

Sending out news or updates to a circle of friends or customers is managed by the Newsletter component, which is available from the Django administration interface when you're logged in to your website. Simply click on "Admin > Site Administration" on the django CMS toolbar, and go to the Newsletter section in Django administration.

This component has a couple of powerful features. It's best explained having you watch the interesting French overview video from the Emencia website.

KAPITEL 6

# Contributing to django Organice

Official repositories: (kept in sync)

1. Bitbucket: https://bitbucket.org/bittner/django-organice
2. GitHub: https://github.com/bittner/django-organice

Fork any of the repositories, make your code changes or additions, and place a pull request.

## 6.1 Help Wanted

- Writing tests
- Translations (blog posts, documentation, user interface)

### 6.1.1 Translation

Translation is done on Transifex for both the project text strings and the documentation.

The multilingual documentation is written in reStructuredText syntax, and built using Sphinx. As a translator you can simply jump on Transifex and get your hands dirty for your language. Some helpful background reading is available from Read the Docs and Transifex support.

## 6.2 Bumping Versions, Package, Release

As for the version numbers of django Organice we use Semantic Versioning.

KAPITEL 7

# Alternativen zu django Organice

Mit django Organice versuchen wir Menschen glücklich zu machen, die keine Technik- oder Prozessexperten sind. Menschen, die einfach und unkompliziert zusammenarbeiten wollen, ohne vorher an umfangreichen Schulungen teilgenommen zu haben, nur um das Arbeitsinstrument zu verstehen. Einfachheit ist der Schlüssel, alles soll selbsterklärend sein, damit die Arbeit leicht von der Hand geht. Dabei sollen gelegentliche Nutzer genauso glücklich werden wie Experten, die die Plattform täglich nutzen.

Natürlich kann django Organice nicht für alles die Lösung sein. Wenn Sie das Gefühl haben, Ihre speziellen Bedürfnisse kann es nicht befriedigen, könnte es sich auszahlen, einen Blick auf die uns bekannten Alternativen zu werfen. Falls Sie weitere Angebote kennen, geben Sie uns Bescheid, damit wir diese Liste sinnvoll ergänzen können.

## 7.1 Software-Alternativen

In alphabetischer Reihenfolge.

- Bitrix Intranet (PHP, kommerziell, wird mit Quellcode geliefert)
- Coyo (Java/Play, kommerziell, wird mit Quellcode geliefert)
- Fairgate (PHP, kommerziell, nur Schweiz)
- Microsoft SharePoint + Yammer [1] (.NET + Rails, kommerziell)
- Open Atrium (Drupal, open source)

## 7.2 Cloud-Mitbewerber

In alphabetischer Reihenfolge.

- Bitrix24.com / Bitrix24.de [1]
- Igloo

[1] http://intranetsdoneright.blogspot.com/2013/11/what-is-social-intranet.html

# CHANGELOG

**Changes** https://github.com/bittner/django-organice/compare/v0.2...HEAD

## 8.1 0.2

- Project-level Makefile
- Automation of translation processes with Transifex (for documentation only)
- New options for settings (`ORGANICE_URL_PATH_...`)
- Newsletter editor configuration, newsletter template sample
- Added social login and user profiles (django-allauth)
- Added assets pipeline (bootstrap-sass, Compass, UglifyJS v2)
- Upgraded jQuery to v1.11.0, template overhaul with Bootstrap
- Added language selection dropdown menu
- Migrated theme data (templates, styles, and javascript) and assets pipeline to separate projects
- Generation of server configuration (lighttpd) and more options in organice-setup
- Added media management (django-media-tree)
- Added todo lists (django-todo)
- Added generic analytics (django-analytical)

  **Changes** https://github.com/bittner/django-organice/compare/v0.1...v0.2

  **Packages** https://github.com/bittner/django-organice/blob/v0.2/docs/requirements.txt

## 8.2 0.1

- Initial release
- Based on Django 1.5.5, django CMS 2.4.3, Zinnia blog 0.13, Emencia newsletter 0.3.dev
- A more natural i18n mechanism than vanilla Django, no language prefix for default language
- Setup script with project generation, deployment settings, custom templates, Bootstrap 3

**Packages** https://github.com/bittner/django-organice/blob/v0.1/docs/requirements.txt

KAPITEL 9

# Verzeichnisse und Tabellen

- *genindex*
- *modindex*
- *search*