django-wiki Documentation

Release 0.0.24.2

Benjamin Bach

April 26, 2015

Contents

Contents:

CHAPTER 1

Installation

1.1 Pre-requisites

For image processing, django-wiki uses the Pillow library (a fork of PIL). The preferred method should be to get a system-wide, pre-compiled version of Pillow, for instance by getting the binaries from your Linux distribution repos.

1.1.1 Debian-based Linux Distros

You may find this a bit annoying: On Ubuntu 12.04 and Debian, PIL is satisfied by installing `python-imaging`, however Pillow is not! On later versions of Ubuntu (tested on 13.10), Pillow is satisfied, but PIL is not. But since PIL no longer compiles on later releases of Ubuntu, we have opted to use Pillow. The alternative would be that django-wiki's requirements would be installed and silently fail (i.e. PIL from pip compiles on Ubuntu 13+ but finds no system libraries for image processing).

If you are on Ubuntu 13+, you may install a system-wide Pillow-adequate library like so:

```
sudo apt-get install python-imaging
```

After, you can verify that Pillow is satisfied by running `pip show Pillow`.

```
$ pip show Pillow
---
Name: Pillow
Version: 2.0.0
Location: /usr/lib/python2.7/dist-packages
```

On Ubuntu 12.04, Debian Wheezy, Jessie etc., you should acquire a system-wide installation of Pillow, read next section...

1.1.2 Pip installation

Firstly, you need to get development libraries that PIP needs before compiling. For instance on Debian/Ubuntu 12.04:

```
sudo apt-get install libjpeg8 libjpeg-dev libpng libpng-dev
```

Later versions of Ubuntu:

```
sudo apt-get install libjpeg8 libjpeg-dev libpng12-0 libpng12-dev
```

After that, install with `sudo pip install Pillow`. You might as well install Pillow system-wide, because there are little version-specific dependencies in Django applications when it comes to Pillow, and having multiple installations of the very same package is a bad practice in this case.

1.1.3 Mac OS X 10.5+

Ethan Tira-Thompson has created ports for OS X and made them available as a .dmg installer. Download and install the universal combo package here.

Once you have the packages installed, you can proceed to the pip installation. PIL will automatically pick up these libraries and compile them for django use.

1.2 Install

To install the latest stable release:

```
pip install wiki
```

Install directly from Github (in case you have no worries about deploying our master branch directly):

```
pip install git+git://github.com/benjaoming/django-wiki.git
```

1.2.1 Configure `settings.INSTALLED_APPS`

The following applications should be listed - NB! it's important to maintain the order due to database relational constraints:

```
'django.contrib.sites', # django 1.6.2
'django.contrib.humanize',
'django_nyt',
'mptt',
'sekizai',
'sorl.thumbnail',
'wiki',
'wiki.plugins.attachments',
'wiki.plugins.notifications',
'wiki.plugins.images',
'wiki.plugins.macros',
```

1.2.2 Django < 1.7

If you run older versions of django, please point south to the correct migrations modules like so:

```
INSTALLED_APPS += ['south',]
SOUTH_MIGRATION_MODULES = {
    'django_nyt': 'django_nyt.south_migrations',
    'wiki': 'wiki.south_migrations',
    'images': 'wiki.plugins.images.south_migrations',
    'notifications': 'wiki.plugins.notifications.south_migrations',
    'attachments': 'wiki.plugins.attachments.south_migrations',
}
```

1.2.3 Database

To sync and create tables, do:

```
python manage.py syncdb
python manage.py migrate
```

1.2.4 Configure TEMPLATE_CONTEXT_PROCESSORS

Add 'sekizai.context_processors.sekizai' and 'django.core.context_processors.debug' to settings.TEMPLATE_CONTEXT_PROCESSORS. Please refer to the Django docs to see the current default setting for this variable.

In Django 1.5, it should look like this:

```
TEMPLATE_CONTEXT_PROCESSORS = (
    "django.contrib.auth.context_processors.auth",
    "django.core.context_processors.debug",
    "django.core.context_processors.i18n",
    "django.core.context_processors.media",
    "django.core.context_processors.request",
    "django.core.context_processors.static",
    "django.core.context_processors.tz",
    "django.contrib.messages.context_processors.messages",
    "sekizai.context_processors.sekizai",
)
```

1.2.5 Set SITE_ID

If you're working with fresh Django installation, you need to set the SITE_ID

```
SITE_ID = 1
```

1.2.6 Include urlpatterns

To integrate the wiki to your existing application, you shoud add the following lines at the end of your project's urls.py.

```
from wiki.urls import get_pattern as get_wiki_pattern
from django_nyt.urls import get_pattern as get_nyt_pattern
urlpatterns += patterns('',
    (r'^notifications/', get_nyt_pattern()),
    (r'', get_wiki_pattern())
)
```

Please use these function calls rather than writing your own include() call - the url namespaces aren't supposed to be customized.

The above line puts the wiki in / so it's important to put it at the end of your urlconf. You can also put it in */wiki* by putting '^wiki/' as the pattern.

CHAPTER 2

Plugins

Add/remove the following to your `settings.INSTALLED_APPS` to enable/disable the core plugins:

- `'wiki.plugins.attachments'`
- `'wiki.plugins.images'`
- `'wiki.plugins.notifications'`

The notifications plugin is mandatory for an out-of-the-box installation. You can safely remove it from `INSTALLED_APPS` if you also override the **wiki/base.html** template.

CHAPTER 3

Customization

See *Settings* for the settings that can be used to configure django-wiki. Other ways to customize django-wiki for your use are listed below.

3.1 Templates

django-wiki can be customized by providing your own templates.

All templates used by django-wiki inherit from `wiki/base.html`, which in turn simply inherits from `wiki/base_site.html` (adding nothing). `wiki/base_site.html` provides a complete HTML page, but provides a number of blocks that you might want to override. The most useful are:

- `wiki_header_branding`
- `wiki_header_navlinks`

These can be overridden to provide your own branding and links in the top bar of the page. The `wiki/base_site.html` template uses Bootstrap 3, so the following example shows how to use this in practice, assuming you want a single link to your home page, and one to the wiki. Add the following as `wiki/base.html` somewhere in your `TEMPLATE_DIRS`:

```
{% extends "wiki/base_site.html" %}

{% block wiki_header_branding %}
<a class="navbar-brand" href="/">Your brand</a>
{% endblock %}

{% block wiki_header_navlinks %}
<ul class="nav navbar-nav">
  <li class="active"><a href="{% url 'wiki:root' %}">Wiki</a></li>
</ul>
{% endblock %}
```

CHAPTER 4

Settings

For now, look in wiki/conf/settings.py to see a list of available settings.

CHAPTER 5

Other tips

1. **Account handling:** There are simple views that handle login, logout and signup. They are on by default. Make sure to set `settings.LOGIN_URL` to point to your login page as many wiki views may redirect to a login page.
2. **Syntax highlighting:** Python-Markdown has a pre-shipped codehilite extension which works perfectly, so add something like:

   ```
   WIKI_MARKDOWN_KWARGS = {'extensions': ['footnotes', 'attr_list', 'headerid', 'extra', 'codehilit
   ```

   to your settings. Currently, django-wiki ships with a stylesheet that already has the syntax highlighting CSS rules built-in. Oh, and you need to ensure `pip install pygments` because Pygments is what the codehilite extension is using!
3. **Project Templates:** Create new django-wiki projects quickly and easily using django-wiki project templates https://github.com/django-wiki/django-wiki-project-template

CHAPTER 6

Release notes

6.1 About the versioning

Up until the django-wiki 0.1 release, versions have been 0.0.1-0.0.24 with migrations kept in South and without any serious issues of upgrading, `python manage.py migrate` was enough.

django-wiki 0.1 is cutting ties in the sense that migrations are being reset. This means that you can upgrade directly the upcoming 0.0.24 to 0.1 but upgrades from previous versions are not possible.

6.2 django-wiki 0.0.24

This release is a transitional release for anyone still using an older version of django-wiki. The code base has been heavily refactored and this is hopefully the final release.

> **Warning:** 0.0.24 is actually mainly a transitional release.

Compatibility

- Django < 1.7 (That means Django 1.7 is **not** supported)
- South 1.0+ (if you are on an older South, you **need** to upgrade)

Notifications fixed

Through its history, django-wiki has maintained **'a very weird migration'_**. It caused for the notifications plugin's table to be removed, but luckily that makes it quite easy to detect and restore, which the new migrations now do.

However, you may want to bootstrap having notifications. You can ensure that all owners and editors of articles receive notifications using the following management command:

    python manage.py wiki_notifications_create_defaults

6.2.1 Troubleshooting

If you have been running from the git master branch, you may experience problems and need to re-run the migrations entirely.

```
python manage.py migrate notifications zero --delete-ghost-migrations
python manage.py migrate notifications
```

If you get *DatabaseError: no such table: notifications_articlesubscription*, you have been running django-wiki version with differently named tables. Don't worry, just fake the backwards migration:

```
python manage.py migrate notifications zero --fake
```

If you get `relation "notifications_articlesubscription" already exists` you may need to do a manual `DROP TABLE notifications_articlesubscription;` using your DB shell (after backing up this data).

After this, you can recreate your notifications with the former section's instructions.

6.3 django-wiki 0.1

Warning: If you are upgrading from a previous release, please ensure that you firstly install django-wiki 0.0.24 because it contains the final migrations necessary before entering the django-wiki 0.1+ migration tree.
If you are using django 1.7 and have an old installation of django-wiki (which should be impossible since it wouldn't run) please downgrade to 1.6,

```
$ pip install django-wiki==0.0.24
$ python manage.py migrate
$ # EDIT YOUR PROJECT'S SETTINGS
$ pip install django-wiki==0.3
$ python manage.py migrate
```

Ammending settings: If you are running django < 1.7, you need the following in your project's settings:

```
INSTALLED_APPS.append('south')
SOUTH_MIGRATION_MODULES = {
    'django_nyt': 'django_nyt.south_migrations',
    'wiki': 'wiki.south_migrations',
    'images': 'wiki.plugins.images.south_migrations',
    'notifications': 'wiki.plugins.notifications.south_migrations',
    'attachments': 'wiki.plugins.attachments.south_migrations',
}
```

Supported

- Python 2.7, 3.3, and 3.4 (3.2 is not)
- Django 1.5, 1.6 and 1.7
- Django < 1.7 still needs south, and migration trees are kept until next major release.

Release plan:

Until django-wiki 0.2 is released, table names of plugins will defer depending on whether you are using South or django.db.migrations. If you want to upgrade your django to 1.7, please rename tables manually.

Django-wiki 0.2 will *not* support South.. but development will remain in the 0.1 branch for now.

New features are introduced in the 0.1 branch until something seriously has to break due to some force majeure.

CHAPTER 7

Known Issues

7.1 Django 1.4

1.4.2+

- In the requirements is stated `sorl-thumbnail>11.12.1b` which is the newest beta of sorl-thumbnail. However, you need to downgrade after.
- [django-mptt issue #271](https://github.com/django-mptt/django-mptt/issues/271) means that you should use Django 1.4.2+.

7.2 Django 1.7 and Python 2.6

Those are incompatible with each other and can never be combined as Dj 1.7 dropped Py 2.6 support.

CHAPTER 8

Indices and tables

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