DjangoPyPi2 Documentation

Release 0.6.2

Zohar Zilberman

Apr 29, 2017

Contents

DjangoPyPI is a Django application that provides a re-implementation of the Python Package Index. Using Twitter Bootstrap for UI, forked from the original DjangoPyPi project, DjangoPyPi2 provides an easy to use and manage interface.

CHAPTER 1

Compatibility with original DjangoPyPi

This is a fork of the original `djangopypi` package. This version is somewhat different than the original version by its design, and it might affect older version in that the database table names are different than the original ones. It is highly recommended that you install a fresh copy of this package and manually transfer you data from your installation.

Since the table names in this installation are different, the same database can be used for the migration. Unfortunately there are too many versions of `djangopypi` our there, so it's quite dangerous to create `south` migrations for them. Sorry for the inconvenience.

Migrating from DjangoPyPi

If you do want to migrate from DjangoPyPi, the best method is to install a fresh copy of DjangoPyPi2 and import all packages from it.

CHAPTER 2

Installation & Configuration

Installation

DjangoPyPi2 is a self-contained Django project along with its apps. If you want fine-grained control, you can looks at the sources of the apps found in the `djangopypi2.apps` package.

The most simple way to install `djangopypi2` is by:

```
# Make sure we run with Bash, create a virtualenv and install packages
$ bash
$ virtualenv pypi-site
$ source pypi-site/bin/activate
$ pip install gunicorn djangopypi2

# Configure our installation
$ manage-pypi-site syncdb
$ manage-pypi-site collectstatic
$ manage-pypi-site loaddata initial
```

That's it, we're now ready to run our server

Where data is kept

By default `djangopypi2` installs and runs from `~/.djangopypi2`, meaning the `.djangopypi2` directory inside the homedir of the user running the web server.

This can be overridden by setting the `DJANGOPYPI2_ROOT` environment variable.

For example, to install with a specific `PROJECT_ROOT` /etc/djangopypi2:

```
# Configure our installation
$ DJANGOPYPI2_ROOT=/etc/djangopypi2 manage-pypi-site syncdb
```

```
$ DJANGOPYPI2_ROOT=/etc/djangopypi2 manage-pypi-site collectstatic
$ DJANGOPYPI2_ROOT=/etc/djangopypi2 manage-pypi-site loaddata initial
```

Running

Gunicorn

It's easiest to see our server running by executing:

```
$ gunicorn_django djangopypi2.website.settings
```

Then surfing to http://localhost:8000/ .

For a permanent setup, simply create a `supervisor <http://supervisord.org/>` configuration (you can omit the `environment` setting if you didn't specify a different project root):

```
[program:djangopypi2]
user = www-data
directory = /path/to/virtualenv
command = /path/to/virtualenv/bin/gunicorn_django djangopypi2.website.settings
environment = DJANGOPYPI2_ROOT='/path/to/djangopypi2'
```

Apache + mod_wsgi

If you used `DJANGOPYPI2_ROOT=/etc/djangopypi2`

```
WSGIPythonPath /usr/lib/python2.6/site-packages/djangopypi2/
WSGIPassAuthorization On
<VirtualHost *:80>

  Servername pip.example.com
  ServerAlias *.pip.example.com

  WSGIScriptAlias / /etc/djangopypi2/wsgi.py

  CustomLog logs/pip-access_log combined
  ErrorLog  logs/pip-error_log

</VirtualHost>
```

Note : Adjust `site-packages` path with your python version (2.6 on centos6, 2.7 on Ubuntu as of Apr 2013)

As for /etc/djangopypi2/wsgi.py:

```
import os

os.environ.setdefault("DJANGO_SETTINGS_MODULE", "website.settings")
os.environ.setdefault("DJANGOPYPI2_ROOT", "/etc/djangopypi2")

from django.core.wsgi import get_wsgi_application
application = get_wsgi_application()
```

Configuration

When first running `djangopypi2`, a file called `settings.json` will be created in the `PROJECT_ROOT` directory:

```
{
    "DEBUG": true,
    "ADMINS": [],
    "LANGUAGE_CODE": "en-us",
    "TIME_ZONE": "America/Chicago",
    "WEB_ROOT": "/",
    "ALLOW_VERSION_OVERWRITE: "",
    "USE_HTTPS": false,

    "EMAIL_SERVER": "smtp://localhost:1025/",
    "EMAIL_USE_TLS": false,
    "EMAIL_DEFAULT_SENDER": "sender@example.com",

    "DB_ENGINE": "django.db.backends.sqlite3",
    "DB_FOLDER": "PROJECT_ROOT",
    "DB_NAME": "db.sqlite3",
    "DB_HOST": "",
    "DB_PORT": "",
    "DB_USER": "",
    "DB_PASSWORD": ""
}
```

The `DEBUG`, `ADMINS`, `LANGUAGE_CODE` and `TIME_ZONE` are exactly the same as in any Django `settings.py` file.

The `WEB_ROOT` setting allows for reverse proxy support. By specifying any other root than `/` you can move the entire site to be served on a different web root.

The `ALLOW_VERSION_OVERWRITE` setting allows you to selectively allow clients to overwrite package distributions based on the version number. This is a regular expression, with the default empty string meaning 'deny all'. A common use-case example of this is to allow development versions to be overwritten, but not released versions:

```
"ALLOW_VERSION_OVERWRITE": "\\.dev.*$"
```

This will match `1.0.0.dev`, `1.0.0.dev3`, but not `1.0.0`. Note the escaping of the backslash character - this is required to conform to the json format.

The `USE_HTTPS` setting should be set to true if `djangopypi2` is served over HTTPS.

The `EMAIL_SERVER` should contain the SMTP server address in this format:

```
smtp://username:password@host:port/
```

If no authentication is needed, then `smtp://host:port/` is sufficient. To see the email messages sent with the default value of this setting, run `python -m smtpd -n -c DebuggingServer localhost:1025` in a terminal.

The `EMAIL_USE_TLS` should be set to true if TLS should be used to connect to the SMTP server.

The `EMAIL_DEFAULT_SENDER` setting allows you to set the default sender email for the SMTP server.

The settings with name starting with `DB_` are the database settings. If you are using sqlite3, then only these settings matter:

```
"DB_ENGINE": "django.db.backends.sqlite3",
"DB_FOLDER": "PROJECT_ROOT",
"DB_NAME": "db.sqlite3",
```

DB_FOLDER and DB_NAME are the directory and filename where the sqlite3 file resides respectively. For DB_FOLDER, you may use the special value PROJECT_ROOT to use the value of the DJANGOPYPI2_ROOT environment variable as the directory containing the sqlite3 file.

If you are not using sqlite3, then all settings except DB_FOLDER matters. As an example, to setup djangopypi2 with postgreSQL:

```
"DB_ENGINE": "django.db.backends.postgresql_psycopg2",
"DB_NAME": "djangopypi2",
"DB_HOST": "localhost",
"DB_PORT": "5432",
"DB_USER": "postgres",
"DB_PASSWORD": "password"
```

Make sure to actually create the database with name djangopypi2 beforehand. You also need to install psycopg2 package separately.

Package upload directory

Packages are uploaded to PROJECT_ROOT/media/dists/ by default.

You can change this setting by setting up a Django project with more specific settings, or have a look at the admin interface's Global Configuration section to see if you configure your desired behavior in there.

CHAPTER 3

Developers: How to upload packages

Uploading to your PyPI

Assuming you are running your Django site locally for now, add the following to your `~/.pypirc` file:

```
[distutils]
index-servers =
    pypi
    local

[pypi]
username:user
password:secret

[local]
username:user
password:secret
repository:http://localhost:8000/pypi/
```

Uploading a package: Python >=2.6

To push the package to the local pypi:

```
$ python setup.py register -r local sdist upload -r local
```

Uploading a package: Python <2.6

If you don't have Python 2.6 please run the command below to install the backport of the extension for multiple repositories:

```
$ easy_install -U collective.dist
```

Instead of using register and dist command, you can use `mregister` and `mupload` which are a backport of python 2.6 register and upload commands that supports multiple servers.

To push the package to the local pypi:

```
$ python setup.py mregister -r local sdist mupload -r local
```

CHAPTER 4

Users: How to use this server

Installing a package with pip

To install your package with pip:

```
$ pip install -i http://my.pypiserver.com/simple/ <PACKAGE>
```

If you want to fall back to PyPi or another repository in the event the package is not on your new server, or in particular if you are installing a number of packages, some on your private server and some on another, you can use pip in the following manner:

```
$ pip install -i http://localhost:8000/simple/ \
  --extra-index-url=http://pypi.python.org/simple/ \
  -r requirements.txt
```

(substitute your djangopypi2 server URL for the `localhost` one in this example)

The downside is that each install of a package hosted on the repository in `--extra-index-url` will start with a call to the first repository which will fail before pip falls back to the alternative.

CHAPTER 5

Authors / Contributors

Full list of contributors

- Ask Solem <askh@opera.com>
- Rune Halvorsen <runeh@opera.com>
- Russell Sim <russell.sim@gmail.com>
- Brian Rosner <brosner@gmail.com>
- Hugo Lopes Tavares <hltbra@gmail.com>
- Sverre Johansen <sverre.johansen@gmail.com>
- Bo Shi <bs@alum.mit.edu>
- Carl Meyer <carl@dirtcircle.com>
- Vinícius das Chagas Silva <vinimaster@gmail.com>
- Vanderson Mota dos Santos <vanderson.mota@gmail.com>
- Stefan Foulis <stefan.foulis@gmail.com>
- Michael Richardson <michael@michaelrichardson.me>
- Benjamin Liles <benliles@gmail.com>
- Halldór Rúnarsson <halldor89@gmail.com>
- Jannis Leidel <jannis@leidel.info>
- Sebastien Fievet <zyegfryed@gmail.com>
- Jaap Roes <jaap@u-e-h.net>
- Stefano Rivera <stefano@rivera.za.net>
- David Miller <david@deadpansincerity.com>
- Tomasz Wysocki

- Zohar Zilberman <popen2@gmail.com>
- Toby Champion <toby@tobychampion.com>
- Edward Easton <eeaston@gmail.com>
- Guilherme Souza <guivideojob@gmail.com>
- Julien Rottenberg <julien@rottenberg.info>
- Marcus Cobden <marcus@marcuscobden.co.uk>
- Heryandi <heryandi@gmail.com>
- EisenSheng <eisensheng@gmail.com>

External libraries or resources

Originally forked from http://github.com/benliles/djangopypi by Benjamin Liles

This software uses Twitter Bootstrap for UI: http://twitter.github.com/bootstrap/

Twitter Bootstrap graphics ships with Glyphicons: http://glyphicons.com/

Favicon taken from http://pypi.python.org/favicon.ico

RSS and XML icons taken from Wikipedia.

CHAPTER 6

History

0.6.1 (2013-04-??)

- Apache+mod_wsgi documentation (thanks Julien Rottenberg <julien@rottenberg.info>)

0.6.0 (2013-04-28)

- Upgrade to Twitter Bootstrap 2.3.1
- Django 1.5 compatibility
- Added 'delete' button to package and release
- Added missing links to internal pages
- Remove useless "empty" operations like editing a package

0.5.9 (2013-04-14)

- Require Django 1.4.5; 1.5 is not yet supported

0.5.8 (2013-03-05)

- Fix supervisor config so 'environment' works (thanks Toby Champion)
- Distribution deletion now removes the underlying files (thanks Edward Easton)
- Added ALLOW_VERSION_OVERWRITE user config (thanks Edward Easton)

0.5.7 (2012-11-15)

- Fix broken `admin` link
- When a package name contains a '-' sign, try to redirect to an equivalent one with '_' in the name if it's not found

0.5.6.1 (2012-10-23)

- Fix ~/.djangopypi2 to expand according to os.environ['USER'] (thanks davedash)
- Support for DJANGOPYPI2_ROOT environment variable, for explicit project root

0.5.6 (2012-10-10)

- Fix DOAP views and add links to them from package and release views
- Add pypi_manage app

0.5.5 (2012-10-08)

- New pypi_metadata app, holds only package metadata
- /simple/ interface is case insensitive
- New pypi_packages app, split out of pypi_frontend, doing only package management
- From now on pypi_frontend only implements scripting interfaces (xmlrpc, distutils, doap)
- Removed pypi_config app
- Add missing TEMPLATE_CONTEXT_PROCESSORS to website.settings
- Add pypi_users for showing user profiles
- Add Shpinx docs

0.5.4 (2012-10-05)

- Allow additional settings in ~/.djangopypi2/settings.py
- Fix bug in distutils' upload causing upload to fail
- Fix bug causing uploaded files to be saved with the wrong name

0.5.3.1 (2012-10-04)

- Fix mirroring not handling `simple` method

0.5.3 (2012-10-04)

- Remove policy from MirrorSite
- Redirect to first enabled mirror site when package is not found locally

0.5.2 (2010-10-03)

- Organize code in distutils views
- Detect binary platform (in case of bdist_*) from filename

0.5.1 (2012-10-03)

- Provide ready-to-deploy Django project within the package

0.5.0 (2012-10-03)

- Removed south support (too many changes), hopefully added in a future version
- Added bootstrap-based user interface
- Split djangopypi to several Django apps
- Switched to relative imports
- All configuration resides in the database and editable from the admin
- Static files are automatically served when DEBUG = True
- Removed loadclassifiers command
- Contains fixtures with initial data for all configuration models

0.4.4 (2012-04-18)

- xmlrpc bug fixes
- CSRF token template tags on forms
- Transaction bug fixes
- Switched to logging over stdout
- Proxy simple and detail views when necessary
- Removed unused legacy view, submit_package_or_release
- ppadd management command working again

0.4.3 (2011-02-22)

- Moved xmlrpc views into views folder
- Moved xmlrpc command settings to the settings file
- Cleaned up xmlrpc views to remove django.contrib.sites dependency

0.4.2 (2011-02-21)

- Added CSRF support for Django>=1.2
- Added conditional support to proxy packages not indexed

0.4.1 (2010-06-17)

- Added conditional support for django-haystack searching

0.4 (2010-06-14)

- 'list_classifiers' action handler
- Issue #3: decorators imports incompatible with Django 1.0, 1.1
- RSS support for release index, packages
- Distribution uploads (files for releases)

0.3.1 (2010-06-09)

- Installation bugfix

0.3 (2010-06-09)

- Added DOAP views of packages and releases
- Splitting djangopypi off of chishop
- Switched most views to using django generic views

Backwards incompatible changes

- Refactored package/project model to support multiple owners/maintainers
- Refactored release to match the metadata only that exists on pypi.python.org
- Created a Distribution model for distribution files on a release

0.2.0 (2009-03-22)

- Registering projects and uploading releases now requires authentication.
- Every project now has an owner, so only the user registering the project can add releases.
- md5sum is now properly listed in the release link.
- Project names can now have dots ('.') in them.
- Fixed a bug where filenames was mangled if the distribution file already existed.
- Releases now list both project name and version, instead of just version in the admin interface.
- Added a sample buildout.cfg. Thanks to Rune Halvorsen (runeh@opera.com).

Backwards incompatible changes

- Projects now has an associated owner, so old projects must be exported and imported to a new database.

0.1.0 (2009-03-22)

- Initial release