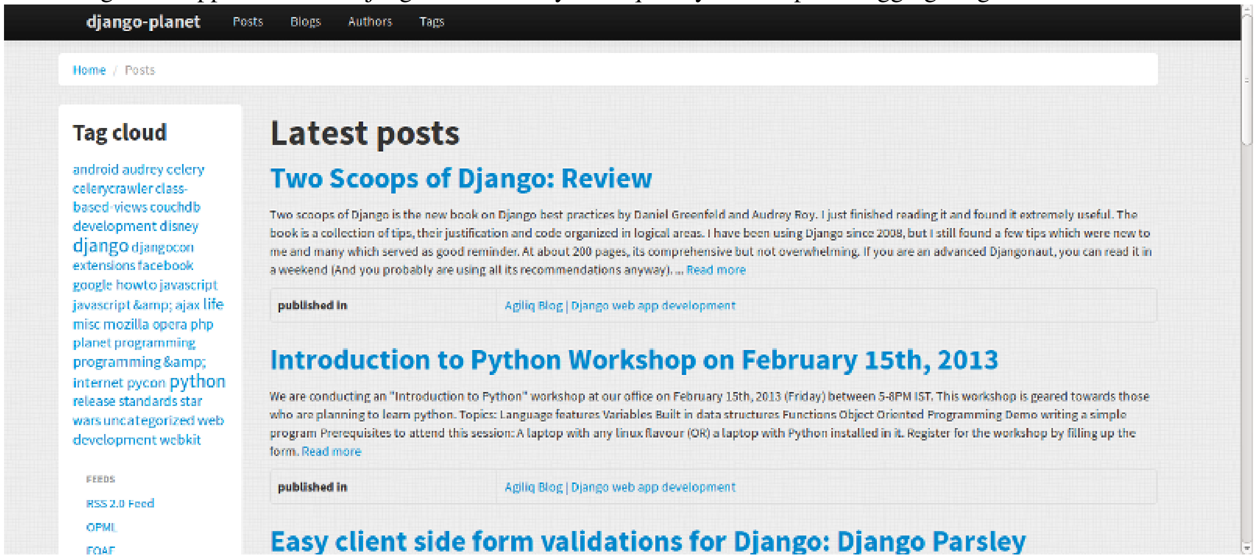
django-planet Documentation

Release 0.5.1

Matías Agustín Méndez

August 13, 2016

This is a generic application for Django that allows you to quickly build a planet aggregating RSS and ATOM feeds.


django-planet
Posts
Blogs
Authors
Tags
Home / Posts
Tag cloud
android audrey celery celerycrawler class-based-views couchdb development disney django djangocon extensions facebook google howto javascript javascript &amp; ajax life misc mozilla opera php planet programming programming &amp; internet pycon python release standards star wars uncategorized web development webkit
FEEDS
RSS 2.0 Feed
OPML
FOAF

Latest posts

Two Scoops of Django: Review

Two scoops of Django is the new book on Django best practices by Daniel Greenfeld and Audrey Roy. I just finished reading it and found it extremely useful. The book is a collection of tips, their justification and code organized in logical areas. I have been using Django since 2008, but I still found a few tips which were new to me and many which served as good reminder. At about 200 pages, its comprehensive but not overwhelming. If you are an advanced Djangonaut, you can read it in a weekend (And you probably are using all its recommendations anyway). ... Read more

published in Agiliq Blog | Django web app development

Introduction to Python Workshop on February 15th, 2013

We are conducting an "Introduction to Python" workshop at our office on February 15th, 2013 (Friday) between 5-8PM IST. This workshop is geared towards those who are planning to learn python. Topics: Language features Variables Built in data structures Functions Object Oriented Programming Demo writing a simple program Prerequisites to attend this session: A laptop with any linux flavour (OR) a laptop with Python installed in it. Register for the workshop by filling up the form. Read more

published in Agiliq Blog | Django web app development

Easy client side form validations for Django: Django Parsley

Some parts of this help docs has been copied from django-tastypie and then readapted to django-planet. Kudos to django-tastypie for its docs!

CHAPTER 1

Content:

1.1 Install

```
pip install django-planet
```

1.1.1 Required settings

Modify your projects `settings.py` file following the next steps:

In `INSTALLED_APPS`:

```
'pagination',
'tagging',
'planet',
```

Be sure to set the site id:

```
SITE_ID = 1
```

Include the context processor:

```
TEMPLATE_CONTEXT_PROCESSORS = (
    #...
    'planet.context_processors.context',
)
```

Add the pagination middleware:

```
MIDDLEWARE_CLASSES = (
    # ...
    'pagination.middleware.PaginationMiddleware',
)
```

1.1.2 Urls

In `urls.py`:

```
urlpatterns = patterns('',
    # ...
    url(r'^planet/', include('planet.urls')),
)
```

1.1.3 Optional settings

```
# To modify cookie names so you don't have login conflicts with other projects
LANGUAGE_COOKIE_NAME = "myplanetlng"
SESSION_COOKIE_NAME = "myplanetid"
# user agent
PLANET_USER_AGENT = "Django Planet"
# limit the number of posts to be retrieved by feed (default is 30)
PLANET_MAX_POSTS = 20
# base template to be used (base.html is the default)
PLANET_BASE_TEMPLATE ="my_base.html"
```

Select the async backend:

```
ASYNC_BACKEND = "huey"
```

This will be detailed in the next section.

1.2 Setup auto updates

By default auto updates are not enabled.

There are 2 options to set auto update for the feeds: a cron job or an async job.

1.2.1 Cron job

30 * * * * python manage.py planet_update_all_feeds

This attempts to pull in new posts every 30 minutes.

1.2.2 Celery

Install Celery `pip install celery redis`. You will need a `celery.py` file as explained here. A Redis or RabbitMQ instance is also required, check the Celery docs. In settings.py:

```
    PLANET_ASYNC_BACKEND = "celery"

from datetime import timedelta
    CELERYBEAT_SCHEDULE = {
        'update-feeds': {
            'task': 'planet.tasks.update_feeds',
            'schedule': timedelta(minutes=30),
        },
    }
```

Then launch the worker: start a beat and a worker:

```
celery -A project_name beat  -l info --broker='redis://localhost:6379/0'
celery -A project_name worker  -l info --broker='redis://localhost:6379/0'
```

1.2.3 Huey

Huey is easier to configure than Celery. If you are not familiar with Celery you might want to use it for an easy start.

Install Huey: `pip install huey`. Add `huey.contrib.djhuey` in INSTALLED_APPS. In settings.py:

```
PLANET_ASYNC_BACKEND = "huey"

from huey import RedisHuey
HUEY = RedisHuey('your_project_name')
```

Launch the worker:

```
python manage.py run_huey
```

1.3 Management commands

Add some feeds:

```
python manage.py planet_add_feed https://www.djangoproject.com/rss/weblog/
python manage.py planet_add_feed https://djangopackages.org/feeds/packages/latest/rss/
```

Update a feed:

```
python manage.py planet_update_feed https://www.djangoproject.com/rss/weblog/
```

Update all the feeds:

```
python manage.py planet_update_all_feeds
```

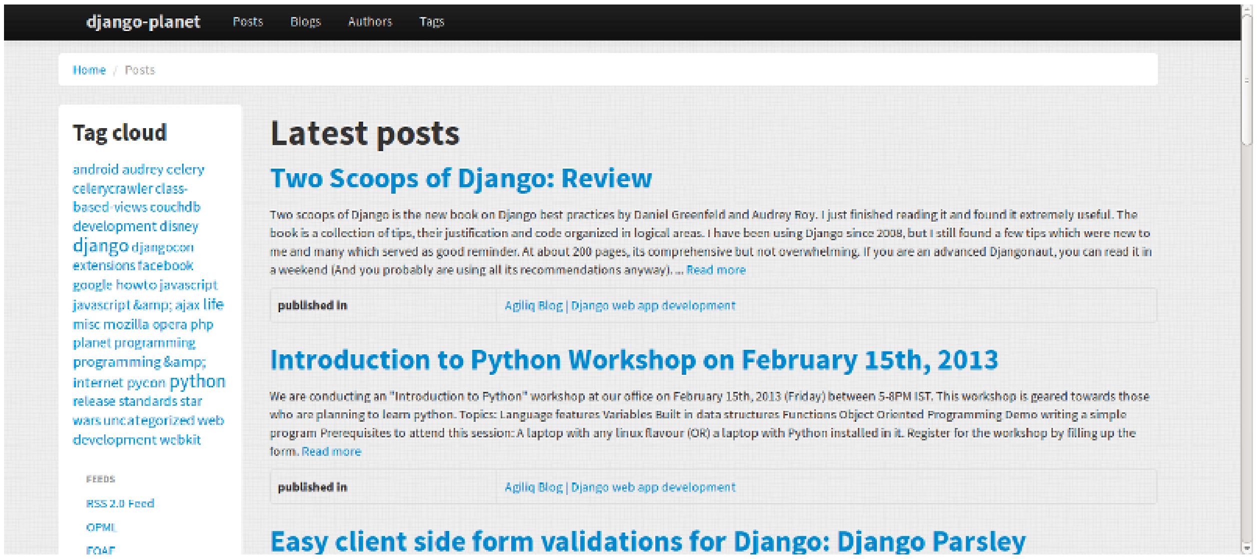
Empty all feeds:

```
python manage.py planet_flush_all_feeds
```

1.4 How does it looks like?

The following screenshots are just for demonstration purposes only:

1.4.1 Index page



django-planet Posts Blogs Authors Tags

Home / Posts

Tag cloud

android audrey celery celerycrawler class-based-views couchdb development disney django djangocon extensions facebook google howto javascript javascript &amp; ajax life misc mozilla opera php planet programming programming &amp; internet pycon python release standards star wars uncategorized web development webkit

FEEDS

RSS 2.0 Feed

OPML

FOAF

Latest posts

Two Scoops of Django: Review

Two scoops of Django is the new book on Django best practices by Daniel Greenfeld and Audrey Roy. I just finished reading it and found it extremely useful. The book is a collection of tips, their justification and code organized in logical areas. I have been using Django since 2008, but I still found a few tips which were new to me and many which served as good reminder. At about 200 pages, its comprehensive but not overwhelming. If you are an advanced Djangonaut, you can read it in a weekend (And you probably are using all its recommendations anyway). ... Read more

published in	Agiliq Blog | Django web app development

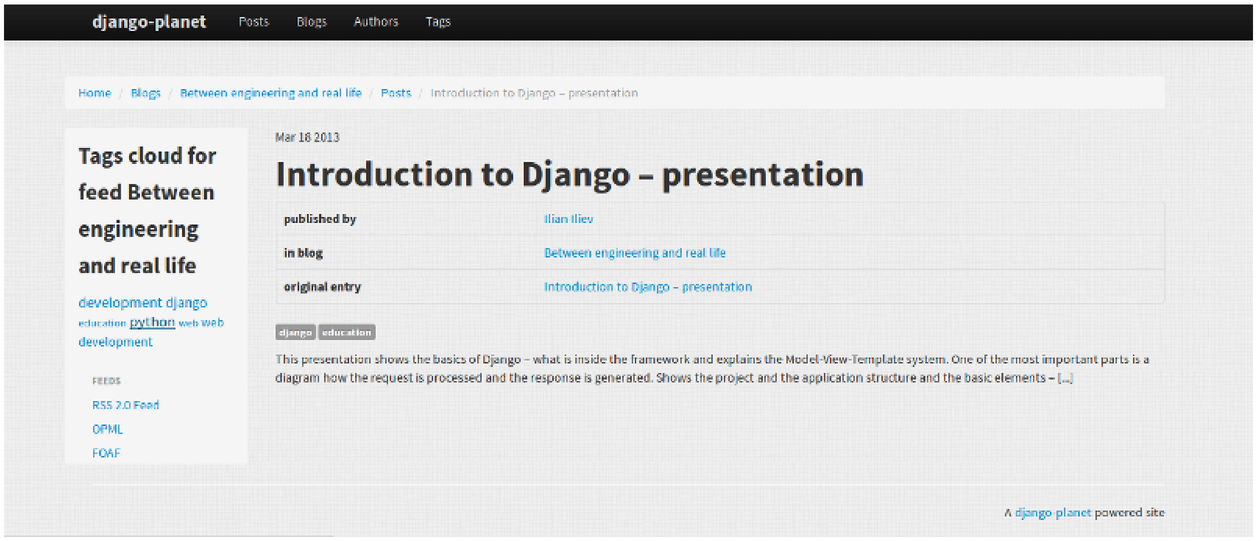
Introduction to Python Workshop on February 15th, 2013

We are conducting an "Introduction to Python" workshop at our office on February 15th, 2013 (Friday) between 5-8PM IST. This workshop is geared towards those who are planning to learn python. Topics: Language features Variables Built in data structures Functions Object Oriented Programming Demo writing a simple program Prerequisites to attend this session: A laptop with any linux flavour (OR) a laptop with Python installed in it. Register for the workshop by filling up the form. Read more

published in	Agiliq Blog | Django web app development

Easy client side form validations for Django: Django Parsley

1.4.2 Post view



django-planet Posts Blogs Authors Tags

Home / Blogs / Between engineering and real life / Posts / Introduction to Django – presentation

Tags cloud for feed Between engineering and real life

development django education python web web development

FEEDS

RSS 2.0 Feed

OPML

FOAF

Mar 18 2013

Introduction to Django – presentation

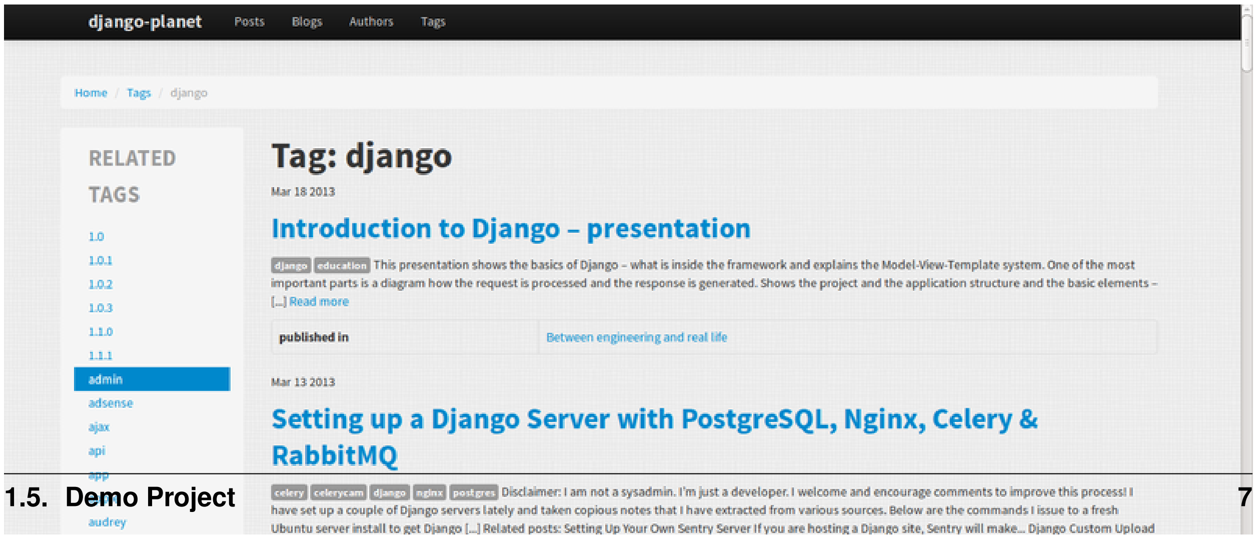
published by	Ilian Iliev
in blog	Between engineering and real life
original entry	Introduction to Django – presentation

django education

This presentation shows the basics of Django – what is inside the framework and explains the Model-View-Template system. One of the most important parts is a diagram how the request is processed and the response is generated. Shows the project and the application structure and the basic elements – [...]

A django planet powered site

1.4.3 Tag view



django-planet Posts Blogs Authors Tags

Home / Tags / django

RELATED TAGS

1.0
1.0.1
1.0.2
1.0.3
1.1.0
1.1.1
admin
adsense
ajax
api
app
audrey

Tag: django

Mar 18 2013

Introduction to Django – presentation

django education This presentation shows the basics of Django – what is inside the framework and explains the Model-View-Template system. One of the most important parts is a diagram how the request is processed and the response is generated. Shows the project and the application structure and the basic elements – [...] Read more

published in	Between engineering and real life

Mar 13 2013

Setting up a Django Server with PostgreSQL, Nginx, Celery & RabbitMQ

celery celerycam django nginx postgres Disclaimer: I am not a sysadmin. I'm just a developer. I welcome and encourage comments to improve this process! I have set up a couple of Django servers lately and taken copious notes that I have extracted from various sources. Below are the commands I issue to a fresh Ubuntu server install to get Django [...] Related posts: Setting Up Your Own Sentry Server If you are hosting a Django site, Sentry will make... Django Custom Upload

1.5 Demo Project

1.6 Contributing

django-planet is open-source and, as such, grows (or shrinks) & improves in part due to the community. Below are some guidelines on how to help with the project.

1.6.1 Philosophy

- django-planet is BSD-licensed. All contributed code must be either
  - the original work of the author, contributed under the BSD, or...
  - work taken from another project released under a BSD-compatible license.
- GPL'd (or similar) works are not eligible for inclusion.
- django-planet's git master branch should always be stable, production-ready & passing all tests.

1.6.2 Guidelines For Reporting An Issue/Feature

So you've found a bug or have a great idea for a feature. Here's the steps you should take to help get it added/fixed in Tastypie:

- First, check to see if there's an existing issue/pull request for the bug/feature. All issues are at https://github.com/matagus/django-planet/issues and pull reqs are at https://github.com/matagus/django-planet/pulls.
- If there isn't one there, please file an issue. The ideal report includes:
  - A description of the problem/suggestion.
  - How to recreate the bug.
  - If relevant, including the versions of your:
    * Python interpreter
    * Django
    * django-planet
    * Optionally of the other dependencies involved
  - Ideally, creating a pull request with a (failing) test case demonstrating what's wrong. This makes it easy for us to reproduce & fix the problem. Instructions for running the tests are at Django Planet

1.6.3 Guidelines For Contributing Code

If you're ready to take the plunge & contribute back some code/docs, the process should look like:

- Fork the project on GitHub into your own account.
- Clone your copy of django-planet.
- Make a new branch in git & commit your changes there.
- Push your new branch up to GitHub.
- Again, ensure there isn't already an issue or pull request out there on it. If there is & you feel you have a better fix, please take note of the issue number & mention it in your pull request.

- Create a new pull request (based on your branch), including what the problem/feature is, versions of your software & referencing any related issues/pull requests.

In order to be merged into django-planet, contributions must have the following:

- A solid patch that:
  - is clear.
  - works across all supported versions of Python/Django.
  - follows the existing style of the code base (mostly PEP-8).
  - comments included as needed.
- A test case that demonstrates the previous flaw that now passes with the included patch.
- If it adds/changes a public API, it must also include documentation for those changes.
- Must be appropriately licensed (see "Philosophy").
- Adds yourself to the AUTHORS file.

If your contribution lacks any of these things, they will have to be added by a core contributor before being merged into django-planet proper, which may take substantial time for the all-volunteer team to get to.

1.6.4 Guidelines For Core Contributors

If you've been granted the commit bit, here's how to shepherd the changes in:

- Any time you go to work on djano-planet, please use `git pull --rebase` to fetch the latest changes.
- Any new features/bug fixes must meet the above guidelines for contributing code (solid patch/tests passing/docs included).
- Commits are typically cherry-picked onto a branch off master.
  - This is done so as not to include extraneous commits, as some people submit pull reqs based on their git master that has other things applied to it.
- A set of commits should be squashed down to a single commit.
  - `git merge --squash` is a good tool for performing this, as is `git rebase -i HEAD~N`.
  - This is done to prevent anyone using the git repo from accidently pulling work-in-progress commits.
- Commit messages should use past tense, describe what changed & thank anyone involved.

  """Added template tag to get all posts.

  Further description appears here if the change warrants an explanation as to why it was done."""
- For any patches applied from a contributor, please ensure their name appears in the AUTHORS file.
- When closing issues or pull requests, please reference the SHA in the closing message (i.e. `Thanks! Fixed in SHA: 6b93f6`). GitHub will automatically link to it.

Requirements

django-planet requires the following modules but simply installing it using **Pip_** will also install them: `pip install django-planet`

2.1 Required

- Python 2.6+
- Django 1.6/1.7
- django-tagging 0.3.6
- django-pagination 1.0.0+
- feedparser
- BeautifulSoup4

Optionally, install celery if you want to add and update feeds using async & parallel tasks:

- Celery or Huey

CHAPTER 3

Why django-planet?

There are other feed aggregators out there for Django. You need to assess the options available and decide for yourself. That said, here are some common reasons for django-planet.

- You need to quickly create a blog aggregator website with a nice look & feel.
- You want a full website for browsing blog posts and its authors and tags, feeds and blogs.
- SEO matters to you: django-planet has templates with SEO metatags and it includes sitemaps so you may submit them to your favorite search engines.
- You want searching posts, blogs, tags and authors.
- You need to customize templates and have a rich set of template tags to do it.
- You want complete ATOM & RSS support

CHAPTER 4

Running The Tests

The easiest way to get setup to run django-planet's tests looks like:

```
$ git clone https://github.com/matagus/django-planet.git
$ cd django-planet
$ virtualenv env
$ . env/bin/activate
$ ./env/bin/pip install -U -r requirements.txt
$ ./env/bin/pip install -U mock django-discover-runner factory-boy tox
```

Then running the tests is as simple as:

```
# From the same directory as above:
$ tox
```

That will test django-planet using Python 2.7 combinated with Django 1.4, Django 1.5 and Django 1.6.

CHAPTER 5

Indices and tables

- genindex
- modindex
- search