django-newsletters Documentation

Release 0.2

coordt

February 24, 2016

Contents

Django TWT Newsletters manages subscriptions to one or more newsletters.

It does not manage the content, but provides ways for producers to link content to newsletters. Delivery of the newsletters is provided through a pluggable backend, allowing integration with outside services

Users can manage the newsletters they get, without registering. A confirmation e-mail is sent each time a change is made.

Contents:

CHAPTER 1

Installation

Installation is easy using `pip` or `easy_install`.

```
pip install django-newsletters
```

or

```
easy_install django-newsletters
```

Add `newsletters` and `categories` into your `settings.py`'s `INSTALLED_APPS`:

```
INSTALLED_APPS = {
    # ...
    'newsletters',
    'categories',
    'editor', # required for categories
}
```

1.1 Dependencies

Django-Categories

CHAPTER 2

Getting Started

1. Follow instructions for *Installation*.
2. Configure the *Settings*, if necessary.
3. Set up the URLs in your `urls.py`:

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

4. Make sure your server can send e-mail notifications.
5. Override the *Templates*, if necessary.
6. Write a signal handler for subscription and unsubscription events.

CHAPTER 3

Subscriptions

3.1 Basic API workflow

index: Allows new users to sign up for newsletters

user newsletter management: Allows a user change the subscription status of all the newsletters. This does not do atomic changes. It changes the status for all newsletters at once.

newsletter detail: Provides the HTML of the newsletter. Renders the first template it finds in the order:

1. `newsletters/<newsletter-slug>.html`
2. `newsletters/category.html`
3. `NEWSLETTERS_SETTINGS['DEFAULT_TEMPLATE']`

The template will have the following variables in the context:

- `newsletter`: The Newsletter object
- `category`: The Category object of the newsletter
- `ads`: A list of Advertisement objects scheduled for this newsletter

Management of newsletter subscriptions is managed through a simple API:

A `GET` or `POST` request to the **newsletter index page** returns a page with a form for signing up, a list of the available newsletters with checkboxes, and a field to enter the e-mail address.

If the request also includes `format=JSON`, the newsletter list and signup URL are sent in `JSON` format.

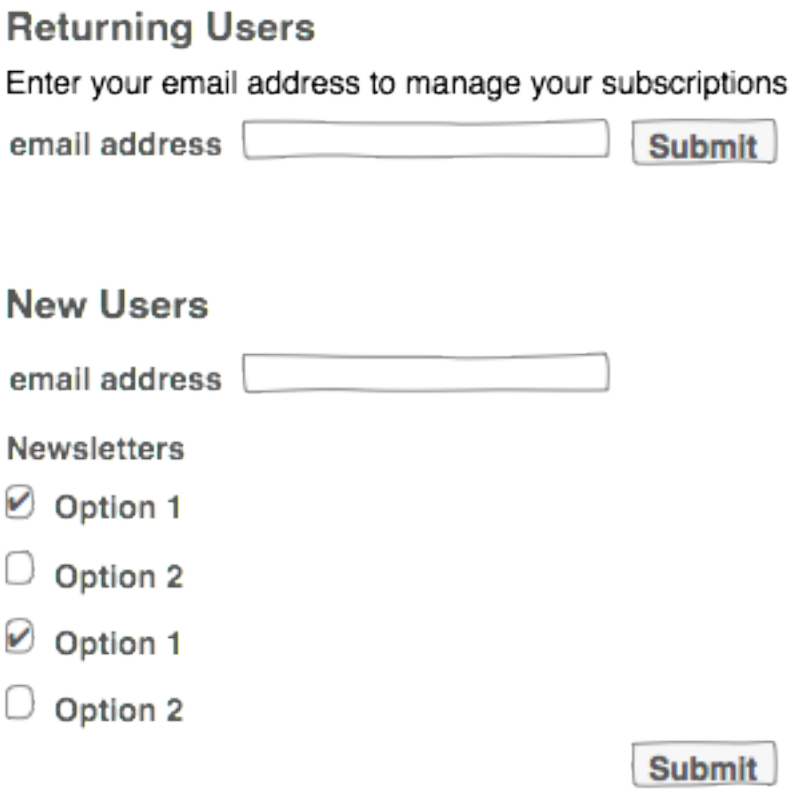
A `GET` or `POST` request to the newsletter index page with the parameter `u` and the users e-mail address (`u=username@example.com`) will redirect to the **newsletter manage page,** which includes the list of newsletters that the e-mail address is subscribed and allows the user to modify the list.

A summary is provided below:

Request Type	Has u	has format	Result
GET	No	No	HTML newsletter sign up form
GET	Yes	No	Redirect to manage page for user
GET	No	Yes	JSON newsletter list and sign up URL
GET	Yes	Yes	Redirect to manage page for user
POST	No	No	HTML newsletter sign up form
POST	Yes	No	Redirect to manage page for user
POST	No	Yes	JSON newsletter list and sign up URL
POST	Yes	Yes	Redirect to manage page for user

3.2 Subscriptions

Here is an example subscription landing page.



3.2.1 Start

1. User goes to the mailing list landing page.
2. They have the choice to manage their newsletter subscriptions (returning user), or sign up for newsletters (new user)

3.2.2 New Users

1. A list of available newsletters with checkboxes, a text field for their e-mail address, and a submit button are available.
2. The user enters their email address, checks the appropriate newsletters and submits the form.
3. If there are errors in the submission, the form will appear again with an error message.
4. If there are no errors, a success page is displayed informing the user a confirmation email was sent.

3.2.3 Returning Users

1. User submits their email address to `./manage/`
2. The form shows all currently subscribed newsletters with checked checkboxes.

3. The user can check or uncheck the boxes to subscribe or unsubscribe from the newsletters.
4. Submission of the form will either provide information regarding errors in the form or displays a success page informing the user a confirmation email was sent.

3.2.4 Confirmation

1. The confirmation email tells the user that there was a change to their newsletter subscriptions. It contains a link to manage their subscriptions if there was an error.

CHAPTER 4

External Mailing List Managers

Upon each successful subscription or unsubscription event, Django Newsletters sends a signal. This allows you to connect outside services.

Both signals provide the same two arguments: email and newsletter. The sender is always the newsletter.

```
from newsletters.signals import subscription, unsubscription

def print_subscription(sender, email, newsletter, *args, **kwargs):
    print "Subscription Event!"
    print email, newsletter

subscription.connect(print_subscription)

def print_unsubscription(sender, email, newsletter, *args, **kwargs):
    print "Unsubscription Event!"
    print email, newsletter

unsubscription.connect(print_unsubscription)
```

To make sure that newsletters are available, you can also listen to newsletter create and delete signals.

```
from django.db.models.signals import post_save, pre_delete
from newsletters.models import Newsletter

def create_external_list(sender, instance, created, *args, **kwargs):
    if created:
        external_service.create_list(sender.name, sender.id)

post_save.connect(create_external_list, sender=Newsletter)

def delete_external_list(sender, instance, *args, **kwargs):
    external_service.pre_delete(sender.id)

pre_delete.connect(delete_external_list, sender=Newsletter)
```

CHAPTER 5

Reference

5.1 Settings

5.1.1 DEFAULT_SETTINGS

```
DEFAULT_SETTINGS = {
    'POSITIONS': (
        (1, 'Leaderboard'),
        (2, 'Medium Rectangle'),
        (3, 'Button 1 (1)'),
        (4, 'Button 1 (2)'),
    ),
    'DEFAULT_TEMPLATE': 'newsletters/default.html',
    'ADVERTISEMENT_STORAGE': settings.DEFAULT_FILE_STORAGE,
    'FROM_EMAIL': 'no-reply@%s' % current_site.domain,
    'AUTO_CONFIRM': True,
    'EMAIL_NOTIFICATION_SUBJECT': '[%s] Newsletter Subscription Change' % current_site.name

}
```

5.1.2 Overriding default settings

In your `settings.py` file, create a dictionary named `NEWSLETTER_SETTINGS`. You only need to include keys for the specific settings you are going to change. For example, to change the default `FROM_EMAIL`:

```
NEWSLETTER_SETTINGS = {
    'FROM_EMAIL': 'our-newsletter-admin@example.com'
}
```

5.1.3 POSITIONS

Advertising positions available for scheduling. Consists of a tuple of `int` and `string` tuples.

Default: `((1, 'Leaderboard'), (2, 'Medium Rectangle'), (3, 'Button 1 (1)'), (4, 'Button 1 (2)'),)`

5.1.4 DEFAULT_TEMPLATE

The default template to use when rendering a newsletter. The app looks for templates in the following order:

1. `newsletters/<newsletter-slug>.html`
2. `newsletters/<category-slug>.html`
3. `DEFAULT_TEMPLATE`

Default: `newsletters/default.html`

5.1.5 ADVERTISEMENT_STORAGE

Storage engine to use when saving advertising images.

Default: settings.DEFAULT_FILE_STORAGE

5.1.6 FROM_EMAIL

The sender of the subscription change notification emails.

Default: `no-reply@<current site domain>`

5.1.7 AUTO_CONFIRM

Currently not working! It is meant to allow a user to click a confirmation email to subscribe/unsubscribe. Currently not working.

Default: `True`

5.1.8 EMAIL_NOTIFICATION_SUBJECT

The subject of the subscription change notification e-mail.

Default: `[<current site name>] Newsletter Subscription Change`

5.2 Templates

- *newsletters/base.html*
- *newsletters/list.html*
- *newsletters/manage.html*
- *newsletters/subscribe.html*
- *newsletters/unsubscribe.html*
- *newsletters/default.html*
- *newsletters/notification_email.txt*

5.2.1 newsletters/base.html

All other templates extend this template.

If your project's primary content block is not called `content`, override `newsletters/base.html` and include `{% block content %}{% endblock %}` within the name of your primary content block.

For example, if your primary content block is called `core_content`, override `newsletters/base.html` to be:

```
{% extends 'base.html' %}
{% block core_content %}
    {% block content %}{% endblock %}
{% endblock %}
```

The change allows all the other default templates to at least work.

5.2.2 newsletters/list.html

This template displays a list of available newsletters.

Context:

- `newsletters` a list of Newsletter objects
- `form` the form for signing up for a bunch of newsletters

5.2.3 newsletters/manage.html

This template allows bulk subscription management for a user.

Context:

- `newsletters` a list of Newsletter objects
- `form` the form for signing up for a bunch of newsletters
- `messages` a list of success or failure messages relating to the last action.

5.2.4 newsletters/subscribe.html

The successful result of the subscription to a specific newsletter.

Context:

- `newsletter` the Newsletter to which the user subscribed

5.2.5 newsletters/unsubscribe.html

The successful result of an unsubscription to a specific newsletter.

Context:

- `newsletter` the Newsletter to which the user unsubscribed

5.2.6 newsletters/default.html

The default template for rendering a newsletter.

5.2.7 newsletters/notification_email.txt

The email sent to users notifying them of changes in their subscription status.

Context:

- `unsubscriptions` a list of `Newsletter` objects from which the user unsubscribed.
- `subscriptions` a list of `Newsletter` objects to which the user subscribed.
- `site` the current `Site` object.
- `email` the email of the subscriber.

CHAPTER 6

Indices and tables

- genindex
- modindex
- search