Bambu API Documentation

Release 2.0

Steadman

April 27, 2014

A shortcut function for sending template-based emails in HTML and plain-text format, and subscribing users to newsletters

CHAPTER 1

About Bambu Mail

This package allows for the quick and simple sending of emails in both plain-text and HTML format, without needing to specify both. A base HTML template is supplied which can be overridden. Markdown is the expected format for the email body, which means it gets transformed into HTML but left alone for the text-only version.

In addition, Bambu Mail has a simple provider system for newsletter subscriptions, allowing new signups to be automatically added to a mailing list (should they choose to). The only current provider in this package is for Mailchimp, but the base type is easily extensible.

Finally, Bambu Mail provides a Postmark email backend, based on the django-postmark but with a slight bug fix.

CHAPTER 2

About Bambu Tools 2.0

This is part of a toolset called Bambu Tools. It's being moved from a namespace of `bambu` to its own 'root-level' package, along with all the other tools in the set. If you're upgrading from a version prior to 2.0, please make sure to update your code to use `bambu_mail` rather than `bambu.mail`.

CHAPTER 3

Installation

Install the package via Pip:

```
pip install bambu-mail
```

Add it to your `INSTALLED_APPS` list along with Bambu Markup:

```
INSTALLED_APPS = (
    ...
    'bambu_mail',
    'bambu_markup'
)
```

Add `bambu_mail.urls` to your URLconf:

```
urlpatterns = patterns('',
    ...
    url(r'^mail/', include('bambu_mail.urls')),
)
```

CHAPTER 4

Questions or suggestions?

Find me on Twitter (@iamsteadman) or visit my blog.

CHAPTER 5

Contents

5.1 Settings

5.1.1 Newsletter settings

NEWSLETTER_PROVIDER The fully-qualified module and class name of the provider to use for subscribing users to mailing lists

NEWSLETTER_SETTINGS A dictionary of settings to pass to the newsletter provider when a new subscription is required

5.1.2 Postmark settings

POSTMARK_API_KEY The Postmark API key

POSTMARK_SSL Send data to Postmark's API via SSL (defaults to `False`)

POSTMARK_TEST_MODE Rather than send the message to Postmark's API, simply print out the JSON data that would have been sent

5.2 Newsletter providers

A newsletter provider is a fairly simple class that exposes a `subscribee()` method. That method calls an API to add the supplied email address to a mailing list. Which list to add the email address to is determined by the value of the `kwargs` passed to that method. These should match settings specified in the Django settings file for that newsletter provider.

The simplest way to setup a provider is to specify the following in your Django settings file:

```
>>> NEWSLETTER_SETTINGS = {
...     'API_KEY': 'mykey',
...     'LIST_IDS': {
...         'new_signup': '1234567890'
...     }
... }
```

The idea here is that the provider you use will translate the `API_KEY` argument to one which it understands, and the same with the list IDs. That way you can refer to multiple mailing lists within your application code without having to store the ID of that list in more than one place. Also, if you change providers, you don't need to change anything other than the value of each item in the `LIST_IDS` dict.

class bambu_mail.newsletter.**ProviderBase**(***kwargs*)

The provider class is instantiated by bambu_mail.shortcuts.subscribe. Kwargs are gathered from the NEWSLETTER_SETTINGS variable within the Django settings file. Because Bambu Mail is provider agnostic, it's likely that one provider won't understand the kwargs of another, so a keyword argument of ARG_MAPPINGS can be set, where the key is the globally-recognised name for an argument, and the value is the name for that argument that only the specific provider understands.

class bambu_mail.newsletter.mailchimp.**MailChimpProvider**(***kwargs*)

Mailchimp provider that allows new signups to be automatically added to a Mailchimp list.

Parameters

- **email** – The email address to add to the mailing list
- **list_id** – The mailing list to subscribe the user to (as defined in the NEWSLETTER_SETTINGS setting)
- **kwargs** – Additional arguments to be passed to the subscription system

CHAPTER 6

API reference

6.1 Shortcut functions

bambu_mail.shortcuts.**render_to_mail**(*subject*, *template*, *context*, *recipient*, *fail_silently=False*)

Parameters

- **subject** – The subject line of the email
- **template** – The name of the template to render the HTML email with
- **context** – The context data to pass to the template
- **recipient** – The email address or `django.contrib.auth.User` object to send the email to
- **fail_silently** – Set to `True` to avoid errors being raised by the sender

This function acts as an alias to one of two functions, depending on your setup. If you use Celery, this function will perform the compositing and sending of the email asynchronously. Otherwise the process will take place on the same thread.

bambu_mail.shortcuts.**subscribe**(*email*, ***kwargs*)

Parameters

- **email** – The email address of the user to add to the mailing list
- **kwargs** – Keyword arguments to pass to the individual newsletter provider

This function acts as an alias to one of two functions, depending on your setup. If you use Celery, this function will perform the API calls asynchronously. Otherwise the process will take place on the same thread.

6.2 Email backends

exception bambu_mail.backends.postmark.**PostmarkMailServerErrorException**(*value*, *inner_exception=None*)

An exception fired when an HTTP Internal Server Error is raised by Postmark

exception bambu_mail.backends.postmark.**PostmarkMailUnauthorizedException**(*value*, *inner_exception=None*)

An exception fired when an HTTP 401 Unauthorized error is raised by Postmark

exception bambu_mail.backends.postmark.**PostmarkMailUnprocessableEntityException**(*value*, *inner_exception=Non*

An exception fired when an HTTP 422 Unprocessable Entity error is raised by Postmark

class bambu_mail.backends.postmark.**PostmarkMessage**(*message*, *fail_silently=False*)

Original code from django-postmark

6.3 Views

bambu_mail.views.**subscribe**(*request*)

Takes POST data (`email` and optional `next` fields), submitting the `email` field to the newsletter provider for subscription to a mailing list, and redirecting the user to the value of `next` (this can also be provided in the querystring), or the homepage if no follow-on URL is supplied, with a message in the `django.contrib.messages` queue to let them know it was successful.

If the email address is invalid or the subscription process was unsuccessful, the user is redirected to the follow-on URL and a message placed in the `django.contrib.messages` queue letting them know what the issue was.

CHAPTER 7

Indices and tables

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

b