django-scribbler Documentation

Release 0.1.1

Mark Lavin

October 12, 2012

CONTENTS

django-scribbler is an application for managing snippets of text for a Django website. Similar projects include django-flatblocks, django-chunks and django-pagelets. This project attempts to take some of the best concepts from those previous projects as well as focus on giving the users instant feedback inspired by Bret Victor's Inventing on Principle talk.

CHAPTER

ONE

FEATURES

- Simple template tag for defining snippet blocks with default text
- Front-end editing of snippets with the powerful CodeMirror editor
- Live in-place preview of content while editing
- The full power of the Django template language in the snippet blocks

CHAPTER

TWO

INSTALLATION

django-scribbler requires Django >= 1.3 and Python >= 2.6 (but < 3.0)

To install from PyPi:

```
pip install django-scribbler
```

CHAPTER

THREE

DOCUMENTATION

Documentation on using django-scribbler is available on Read The Docs.

CHAPTER

FOUR

LICENSE

django-scribbler is released under the BSD License. See the LICENSE file for more details.

CHAPTER

FIVE

CONTRIBUTING

If you think you've found a bug or are interested in contributing to this project check out django-scribbler on Github.

CHAPTER

SIX

CONTENTS

6.1 Getting Started

Below are the basic steps need to get django-scribbler integrated into your Django project.

6.1.1 Configure Settings

You need to include `scribbler` to your installed apps. django-scribbler requires `django.contrib.auth` which in turn requires `django.contrib.sessions` which are enabled in Django by default. You will also need to include a context processor to include the current request in the template context.

```
INSTALLED_APPS = (
    # Required contrib apps
    'django.contrib.auth',
    'django.contrib.sessions',
    # Other installed apps would go here
    'scribbler',
)

TEMPLATE_CONTEXT_PROCESSORS = (
    # Other context processors would go here
    'django.core.context_processors.request',
)
```

Note that `TEMPLATE_CONTEXT_PROCESSORS` is not included in the default settings created by `startproject`. You should take care to ensure that the default context processors are included in this list. For a list of default `TEMPLATE_CONTEXT_PROCESSORS` please see the official Django docs.

For the context processor to have any effect you need to make sure that the template is rendered using a RequestContext. This is done for you with the render shortcut.

django-scribbler aggressively caches the scribble content. By default the scribble content is cached for 12 hours. You have the option to configure this cache timeout with the `SCRIBBLER_CACHE_TIMEOUT` setting. The value should be the timeout in seconds.

6.1.2 Configure Urls

You should include the scribbler urls in your root url patterns.

```
urlpatterns = patterns('',
    # Other url patterns would go here
    url(r'^scribbler/', include('scribbler.urls')),
)
```

6.1.3 Create Database Tables

You'll need to create the necessary database tables for storing scribble content. This is done with the `syncdb` management command built into Django:

```
python manage.py syncdb
```

django-scribbler uses South to handle database migrations. If you are also using South then you should run `migrate` instead:

```
python manage.py migrate scribbler
```

6.1.4 User Permissions

To edit scribbles on the front-end users must have the `scribbler.add_scribble` and `scribbler.change_scribble` permissions. You can configure uses to have these permissions through the users section of the Django admin. Superusers have all of these permissions by default.

6.1.5 Include Static Resources

django-scribbler includes both CSS and JS resources which need to be included in your templates to handle the front-end content management. Since you may want to include scribbles on any page on your site these should be included in your base template `<head>`.

```
<link rel="stylesheet" href="{{ STATIC_URL }}scribbler/libs/codemirror.css">
<link rel="stylesheet" href="{{ STATIC_URL }}scribbler/css/scribbler.css">
<script data-main="{{ STATIC_URL }}scribbler/js/scribbler" src="{{ STATIC_URL }}scribbler/libs/requi
```

This uses RequireJS to load the additional JS resources. The front-end editor uses CodeMirror (currently using v2.32) which is included in the distribution. Both RequireJS and CodeMirror are available a MIT-style license compatible with this project's BSD license. You can find the license files included in `scribbler/static/scribbler/libs/`.

6.1.6 Place Scribbles in Your Template

You are now ready to place the scribble content blocks throughout your templates. This is done with the `scribble` block tag. It takes one argument which is the slug name for the scribble. Slugs must be unique per url/slug pair. That means you cannot use the same slug more than once in the template but you can use the same slug in different templates as long as they are rendered on different urls.

```
{% load scribbler_tags %}
{% scribble 'header' %}
    <p>Blip {% now 'Y' %} {{ STATIC_URL|upper }}</p>
{% endscribble %}
```

The content inside the block is the default content that will be rendered if a matching scribble in the database is not found.

Note: Scribble content can be any valid Django template. However the content does not include all of the context of the template. Only the context provided by the set of `TEMPLATE_CONTEXT_PROCESSORS`.

That should be enough to get you up and running with django-scribbler.

6.2 Release History

Release and change history for django-scribbler

6.2.1 v0.1.1 (Released 2012-08-25)

Minor bug fix release for some JS and CSS issues.

Bug Fixes

- Fixed issue with the content editor z-index allowing content in front when open
- Fixed issue where links within editable content could not be clicked by editors

6.2.2 v0.1.0 (Released 2012-07-28)

- Initial public release.

Features

- Template tag for rendering content blocks
- CodeMirror editor integration

CHAPTER

SEVEN

INDICES AND TABLES

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