Overseer Documentation

Release 0.2.0

DISQUS

June 24, 2013

CONTENTS

Overseer is a simple status board app written in Django.

For a live example, see http://status.disqus.com

CHAPTER

ONE

SETUP

If you haven't already, start by downloading Overseer. The easiest way is with *pip*:

```
pip install overseer --upgrade
```

Or with *setuptools*:

```
easy_install -U overseer
```

Once installed, you're going to need to configure a basic Django project. You can either use Overseer within your existing project, or base it on the provided shell in `example_project`.

1.1 Existing Project

To use it within an existing project, adjust the following in `settings.py`:

```
INSTALLED_APPS = (
    ...
    'overseer',
)
```

You'll also need to include the appropriate `urls.py`:

```
urlpatterns = patterns('',
    (r'^status', include('overseer.urls', namespace='overseer')),
)
```

1.2 New Project

Simple copy the `example_project` directory included with the package, and adjust ''settings.py' as needed.

You may now continue to *Config* for configuration options.

CHAPTER

TWO

CONFIG

Several configuration variables are available within Overseer. All of these are handled with a single dictionary configuration object:

```
OVERSEER_CONFIG = {
    # the title for your page
    'TITLE': 'DISQUS Service Status',

    # the heading text for your page
    'NAME': 'status.disqus.com',

    # the prefix for overseer's media -- by default this is handled using Django's static media serve
    'MEDIA_PREFIX': '/media/',

    # the base url to overseer -- highly recommended
    'BASE_URL' : 'http://status.disqus.com',
}
```

2.1 Email Subscriptions

Allow users to subscribe to email notifications on event updates by configuring the following settings in OVERSEER_CONFIG:

```
OVERSEER_CONFIG = {
    # Enable subscriptions
    'ALLOW_SUBSCRIPTIONS': True,

    # Specify an email from address
    'FROM_EMAIL': 'overseer@domain.com',

    # Ensure base url is set
    'BASE_URL': 'http://status.disqus.com',
}
```

2.2 Twitter Integration

Enabling Twitter integration a few things, first a foremost, you must register an application via Twitter. You only need to fill out the basics. Don't worry about callback URL, *ensure your Application Type is set to Client*.

Once you've obtained an API KEY and API SECRET, you'll need to fill in these values in your configuration:

```
OVERSEER_CONFIG = {
    'TWITTER_CONSUMER_KEY': '...',
    'TWITTER_CONSUMER_SECRET': '...',
}
```

Now you need to obtain an authenticated access token. To do this you're going to need to obtain a PIN number using OAuth. Overseer includes a manage.py command to obtain these values:

```
python manage.py overseer_twitter_auth
```

This will open a new webpage which will ask for authentication to your Twitter account. Click Allow and paste the PIN code given when Overseer asks for it (via the CLI). When finished, you should see something like this:

```
(overseer)[~/Development/overseer/example_project] ./manage.py overseer_twitter_auth
Request Token:
    - oauth_token        = tSsXTKV8NjKDNU364umDp9OYqv2dTFLWrsVML3T3vU
    - oauth_token_secret = uawnL5iieRI7GFOHuBuOACJhGT2WEbvJLMlJhF2E

We are opening a new browser window to authorize your account

Enter your PIN number once authorized: 5541487

Configuration changes:

    'TWITTER_ACCESS_TOKEN':  '241259795-wVnWHaoaI08hbucKhJF1y2CaI4wKvbGyEoF0Ange',
    'TWITTER_ACCESS_SECRET': 'lNkf6CKYPmh3szc1f3ueQdg076facQM3qGKFYzwT2A',

Add the above values to your OVERSEER_CONFIG setting
```

Simply take the values given, and add them to your configuration:

```
OVERSEER_CONFIG = {
    'TWITTER_CONSUMER_KEY': '...',
    'TWITTER_CONSUMER_SECRET': '...',
    'TWITTER_ACCESS_TOKEN':  '241259795-wVnWHaoaI08hbucKhJF1y2CaI4wKvbGyEoF0Ange',
    'TWITTER_ACCESS_SECRET': 'lNkf6CKYPmh3szc1f3ueQdg076facQM3qGKFYzwT2A',
}
```

You'll see now an option to "Post to Twitter" within the Event administration. While this is selected (it's value does not persist) any changes made to event updates (additions, for example) will also create a Twitter posting with a permalink (assuming BASE_URL is configured) and the #status hash tag.

CHAPTER

THREE

ADMINISTRATION

As of the current version, the only way to administer the application is via the `django.contrib.admin` integration.