django12factor Documentation

Release 1.3

Kristian Glass

July 29, 2016

Contents

CHAPTER 1

What is it?

Django is an awesome Python web framework.

"The Twelve-Factor App" is an awesome methodology for building SaaS apps.

`django-12factor` makes Django more 12factor-y. Right now, this focuses on the Config - "Store config in the environment"; Heroku users with addons will be particularly familiar with this.

Still not sure of the benefits? Check out "Twelve-Factor Config: Misunderstandings and Advice".

CHAPTER 2

Usage

Add the following to the bottom of your `settings.py`:

```
import django12factor
d12f = django12factor.factorise()
```

`factorise()` returns a `dict` of settings, so you can now use and assign them as you wish, e.g.

```
DEBUG = d12f['DEBUG']
LOGGING = d12f['LOGGING']
```

If you don't like that repetition, you can (ab)use `globals()` like so:

```
import django12factor
d12f = django12factor.factorise()

def f(setting):
    globals()[setting] = d12f[setting]

f('DEBUG')
f('LOGGING')
```

You can also add non-Django settings this way, e.g. keys to APIs:

```
custom_settings = (
    "GOOGLE_ANALYTICS_KEY",
    "MAILCHIMP_API_KEY",
)
d12f = django12factor.factorise(custom_settings=custom_settings)

MAILCHIMP_API_KEY = d12f['MAILCHIMP_API_KEY']
GOOGLE_ANALYTICS_KEY = d12f['GOOGLE_ANALYTICS_KEY']
```

In the event of a `custom_setting` not being set in the environment, it will default to `None`.

2.1 Give me everything!

If you say so...

```
import django12factor
globals().update(django12factor.factorise())
```

CHAPTER 3

Utilities

`django12factor.getenv_bool` is a utility function that takes the name of an environment variable, and returns `True` _unless_ it is set to either a "falsey" string (e.g. `"no"`) or not set.

CHAPTER 4

Settings

The following settings are currently supported:

4.1 DEBUG

Defaults to `False` for safety reasons, otherwise `True` unless `os.environ("DEBUG")` is a "falsy" string.

4.2 TEMPLATE_DEBUG

As for `DEBUG`, but defaults to the value of `DEBUG`.

4.3 CACHES

Uses django-cache-url to parse `os.environ("CACHE_URL")`.

4.4 LOGGING

A static `LOGGING` dict that configures 12factor-style logging.

4.5 DATABASES

Uses dj-database-url - parses `DATABASE_URL` if it exists, otherwise falls back to in-memory sqlite.

Anything of the form `FOO_DATABASE_URL` will be parsed as `DATABASES['foo']`, allowing you to configure multiple databases via the environment.

4.6 ALLOWED_HOSTS

Treats `os.environ("ALLOWED_HOSTS")` as a comma-separated list.

4.7 SECRET_KEY

Uses `os.environ("SECRET_KEY")` - required if `DEBUG==False`.

CHAPTER 5

Indices and tables

- genindex
- modindex
- search