TaskBuster Documentation

Release 0.1.0

Marina Mele

October 01, 2014

Contents

This is an awesome **Django Project Boilerplate**!!

With this code you can start a *complex* Django Project very quickly, with just a few steps!

Some of the TaskBuster Django Project Boilerplate functionalities are:

- **different virtual environments** for developing, testing and production
- **Internationalization** and **localization** to support different languages
- Project structure
- **HTML5 Boilerplate**
- Template Inheritance
- Functional **tests**
- robots.txt and humans.txt configured

Moreover, you can learn how to create this boilerplate **step by step** in the . There you can learn, step by step, how TaskBuster has been done, and even do it yourself if you want to!!

To start using the Boilerplate, check out the *Requirements* and next the *Quick Start Guide*.

CHAPTER 1

Contents

1.1 Requirements

The requirements necessary to use this Django Project Boilerplate are:

- **python3** and **pip3**
- **virtualenv and virtualenvwrapper**
- **Firefox** (to use Selenium's Webdriver in functional Tests)
- **GNU gettext** (to use Internationalization)

If you don't have the first two requirements, you may find this post useful: .

You can download Firefox from the official web page: .

And if you don't have GNU gettext, check this .

Ready!? Continue to the *Quick Start Guide*!

1.2 Quick Start Guide

1.2.1 Download TaskBuster Django Project Boilerplate

First, you need to download the BoilerPlate from GitHub.

You can visit the repository webpage in and download it as a zip file.

You can also do the same using your terminal with:

```
$ git clone git://github.com/mineta/taskbuster-boilerplate.git
```

This will download the repository in your current direcotry.

1.2.2 Secret Django Key

This boilerplate has the **DJANGO_KEY** setting variable hidden.

You can generate your DJANGO_KEY .

Keep reading to include your new Django key into your project.

1.2.3 Project Name

This project is named *TaskBuster*, so if you are using this Boilerplate to create your own project, you'll have to change the name in a few places:

- *taskbuster_project* **folder** (your top project container)
- *taskbuster_project/taskbuster* **folder** (your project name)
- virtual environment names: **tb_dev** and **tb_test** (name them whatever you want)
- in virtual environments **postactivate** files (see section below), you have to change **taskbuster.settings.development** for your **projectname.settings.development**. Same works for the testing environment.

1.2.4 Virtual environments and Settings Files

First, you must know your Python 3 path:

```
$ which python3
```

which is something similar to /usr/local/bin/python3.

Next, create a Development virtual environment with Python 3 installed:

```
$ mkvirtualenv --python=/usr/local/bin/python3 tb_dev
```

where you might need to change it with your python path.

Go to the virtual enviornment folder with:

```
$ cd $VIRTUAL_ENV/bin
```

and edit the postactivate file.:

    $ vi postactivate

You must add the lines:

```
export DJANGO_SETTINGS_MODULE="taskbuster.settings.development"
export SECRET_KEY="your_secret_django_key"
```

with your project name and your own secret key.

Next, edit the **predeactivate** file and add the line:

```
unset SECRET_KEY
```

Repeat the last steps for your testing environment:

```
$ mkvirtualenv --python=/usr/local/bin/python3 tb_test
$ cd $VIRTUAL_ENV/bin
$ vi postactivate
```

where you have to add the lines:

```
export DJANGO_SETTINGS_MODULE="taskbuster.settings.testing"
export SECRET_KEY="your_secret_django_key"
```

and in the predeactivate file:

```
unset SECRET_KEY
```

Next, install the packages in each environment:

```
$ workon tb_dev
$ pip install -r requirements/development.txt
$ workon tb_test
$ pip install -r requirements/testing.txt
```

1.2.5 Internationalization and Localization

Settings

The default language for this Project is **English**, and we use internatinalization to translate the text into Catalan.

If you want to change the translation language, or include a new one, you just need to modify the **LANGUAGES** variable in the file *settings/base.py*. The language codes that define each language can be found .

For example, if you want to use German you should include:

```
LANGUAGES = (
    ...
    'de', _("German"),
    ...
)
```

You can also specify a dialect, like Luxembourg's German with:

```
LANGUAGES = (
    ...
    'de-lu', _("Luxemburg's German"),
    ...
)
```

Note: the name inside the translation function _("") is the language name in the default language (English).

More information on the .

Translation

Go to the terminal, inside the taskbuster_project folder and create the files to translate with:

```
$ python manage.py makemessages -l ca
```

change the language "ca" for your selected language.

Next, go to the locale folder of your language:

```
$ cd taskbuster/locale/ca/LC_MESSAGES
```

where taskbuster is your project folder. You have to edit the file *django.po* and translate the strings. You can find more information about how to translate the strings .

Once the translation is done, compile your messages with:

```
$ python manage.py compilemessages -l ca
```

Tests

We need to update the languages in our Tests to make sure the translation works correclty. Open the file *functional_tests/test_all_users.py*:

- in **test_internationalization**, update your languages with the translation of title text, here "Welcome to TaskBuster!"
- in **test_localization**, update your languages.

1.2.6 Useful commands

A list of all the commands used to run this template:

```
$ workon tb_dev
$ workon tb_test

$ python manage.py makemessages -l ca
$ python manage.py compilemessages -l ca
```

CHAPTER 2

Indices and tables

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