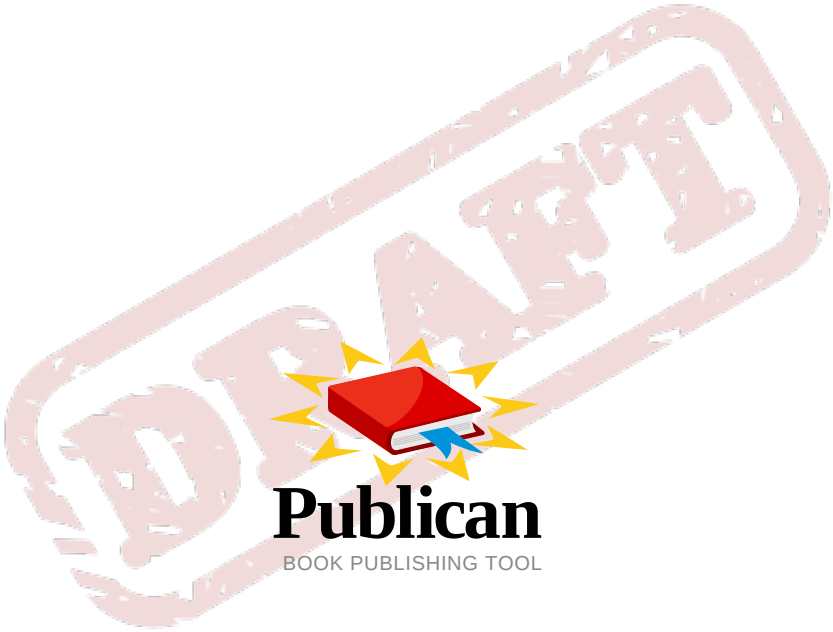
Simple-Shop 0.4

Simple-Shop Documentation

User documentation for simple-shop *v0.4.0-alpha*



Sascha Peter

Simple-Shop 0.4 Simple-Shop Documentation
User documentation for simple-shop *v0.4.0-alpha*
Edition 0.1

Author Sascha Peter *sascha.o.peter@gmail.com*

Copyright © 2015 Sascha Peter This material may only be distributed subject to the terms and conditions set forth in the GNU Free Documentation License (GFDL), V1.2 or later (the latest version is presently available at *http://www.gnu.org/licenses/fdl.txt*).

This documentation will guide users how to setup this project and give an overview about the code and technology used.


DRAFT

Preface

1. Document Conventions

This manual uses several conventions to highlight certain words and phrases and draw attention to specific pieces of information.

In PDF and paper editions, this manual uses typefaces drawn from the *Liberation Fonts*[1] set. The Liberation Fonts set is also used in HTML editions if the set is installed on your system. If not, alternative but equivalent typefaces are displayed. Note: Red Hat Enterprise Linux 5 and later includes the Liberation Fonts set by default.

1.1. Typographic Conventions

Four typographic conventions are used to call attention to specific words and phrases. These conventions, and the circumstances they apply to, are as follows.

`Mono-spaced Bold`

Used to highlight system input, including shell commands, file names and paths. Also used to highlight keycaps and key combinations. For example:

> To see the contents of the file **`my_next_bestselling_novel`** in your current working directory, enter the **`cat my_next_bestselling_novel`** command at the shell prompt and press **`Enter`** to execute the command.

The above includes a file name, a shell command and a keycap, all presented in mono-spaced bold and all distinguishable thanks to context.

Key combinations can be distinguished from keycaps by the hyphen connecting each part of a key combination. For example:

> Press **`Enter`** to execute the command.
>
> Press **`Ctrl`**+**`Alt`**+**`F1`** to switch to the first virtual terminal. Press **`Ctrl`**+**`Alt`**+**`F7`** to return to your X-Windows session.

The first paragraph highlights the particular keycap to press. The second highlights two key combinations (each a set of three keycaps with each set pressed simultaneously).

If source code is discussed, class names, methods, functions, variable names and returned values mentioned within a paragraph will be presented as above, in **`mono-spaced bold`**. For example:

> File-related classes include **`filesystem`** for file systems, **`file`** for files, and **`dir`** for directories. Each class has its own associated set of permissions.

Proportional Bold

This denotes words or phrases encountered on a system, including application names; dialog box text; labeled buttons; check-box and radio button labels; menu titles and sub-menu titles. For example:

[1] https://fedorahosted.org/liberation-fonts/

Choose **System** → **Preferences** → **Mouse** from the main menu bar to launch **Mouse Preferences**. In the **Buttons** tab, click the **Left-handed mouse** check box and click **Close** to switch the primary mouse button from the left to the right (making the mouse suitable for use in the left hand).

To insert a special character into a **gedit** file, choose **Applications** → **Accessories** → **Character Map** from the main menu bar. Next, choose **Search** → **Find…** from the **Character Map** menu bar, type the name of the character in the **Search** field and click **Next**. The character you sought will be highlighted in the **Character Table**. Double-click this highlighted character to place it in the **Text to copy** field and then click the **Copy** button. Now switch back to your document and choose **Edit** → **Paste** from the **gedit** menu bar.

The above text includes application names; system-wide menu names and items; application-specific menu names; and buttons and text found within a GUI interface, all presented in proportional bold and all distinguishable by context.

Mono-spaced Bold Italic or ***Proportional Bold Italic***

Whether mono-spaced bold or proportional bold, the addition of italics indicates replaceable or variable text. Italics denotes text you do not input literally or displayed text that changes depending on circumstance. For example:

To connect to a remote machine using ssh, type **ssh *username@domain.name*** at a shell prompt. If the remote machine is **example.com** and your username on that machine is john, type **ssh john@example.com**.

The **mount -o remount *file-system*** command remounts the named file system. For example, to remount the **/home** file system, the command is **mount -o remount /home**.

To see the version of a currently installed package, use the **rpm -q *package*** command. It will return a result as follows: ***package-version-release***.

Note the words in bold italics above — username, domain.name, file-system, package, version and release. Each word is a placeholder, either for text you enter when issuing a command or for text displayed by the system.

Aside from standard usage for presenting the title of a work, italics denotes the first use of a new and important term. For example:

Publican is a *DocBook* publishing system.

1.2. Pull-quote Conventions

Terminal output and source code listings are set off visually from the surrounding text.

Output sent to a terminal is set in **mono-spaced roman** and presented thus:

```
bin var home etc lib mnt proc opt
```

Source-code listings are also set in **mono-spaced roman** but add syntax highlighting as follows:

```
#!/usr/bin/env python

class ExampleClass(object):
    """This is an example class.

    @author: Sascha Peter <sascha.o.peter@stickyboard.co.uk>
    @since: 2015-07-14
    @version: 0.1.0-alpha
    """

    def __init__(self):
        print("Init isn't it?")

    def main(self):
        print("Main method")

if __name__ == '__main__':
    example_class = ExampleClass()
    print example_class.main()
```

1.3. Notes and Warnings

I'm using three kinds of ways to visually highlight important information.

Note

Notes are tips to make your life easier.

Important

Important boxes are things which you need to make sure you followed or considered.

Warning

Warnings are something which should not be ignored as it may makes your life more troublesome than it needs be.

2. Feedback

This document has been created with big care, however, it is always possible that small errors find their way into the project.

If you have any feedback, suggestions or found typos and alike, please feel free to contact me under *sascha.o.peter@gmail.com* to let me know and I will adjust them in accordingly.


DRAFT

Installation

The following sections will guide you through the setup and installation of the software on your device.

1.1. Setup Python 3 on MAC

Simple-Shop has been built and developed in Python 3.3.2 and Django 1.8. One of the easiest ways to set up python3, or generally multiple versions of python on your environment is through *https://github.com/yyuu/pyenv*.

The following commands are a quick cheat sheet for you if you installed everything correctly with pyenv.

To install a version **`pyenv install version.number`**

To set it/switch to it **`pyenv global version.number`**

Then to check the set version **`pyenv version`**

And the overall system version **`python --version`**

Python 3.3.2 required!
The software has been tested with Python 3.3.2 and Django 1.8. Please make sure you have the right version installed.

1.2. Setup Virtual Environment

When working with different Python versions or different projects, it's suggested to use separate environments for each project to handle their requirements.

One way to achieve this is to use virtualenv. Pyenv has a plugin for virtualenv *https://github.com/yyuu/pyenv-virtualenv*. But you are free to use any virtualenv solution you prefer.

Make sure you activate your virtual environment!
Depending on your operating system and virtualenv solution you went for, the command to activate the virtualenv may vary. With pyenv-virtualenv the command would be **`pyenv activate your-virtual-env`**

1.3. Install project dependencies

When working with different Python versions or different projects, it's suggested to use separate environments for each project to handle their requirements.

After activating the virtual environment for your simple-shop project, you can install all project requirements through PIP. All project dependencies are listed in the pip-requirements.txt file in the root of the project.

```
pip install -r pip-requirements.txt
```


DRAFT

Configuration

The following sections will guide you through the configuration and setup of the website project itself and how to locally launch it in your pre-prepared environment of the previous chapter.

2.1. Configure the project

The root folder of the project will present you with the following folder hiearchy:

- cart/
- djangocart/
- docs/
- product/
- simple_shop/
- **`.gitignore`**
- **`README.md`**
- **`manage.py`**
- **`pip-requirements.txt`**

Please change into simple_shop and create a copy of **`settings_local.local`** with the name **`settings_local.py`**.

```
cd simple_shop/
cp settings_local.template settings_local.py
vi settings_local.py
```

Editor of choice

The last line with **`vi settings_local.py`** is just an example. Please use your editor of choice for editing the python file.

Most varaibles in the settings_local.py will already be as they are needed. The only part which needs to be filled in will be SECRET_KEY.

SECRET_KEY is required for the project to run!

SECRET_KEY is used in many security related places within the django framework and should never be shared nor exposed. Here is an example of a secret key (don't use for productive use): 6*kpa+0m+%k(d-$_xbh#b7zch1_9ox0!35a^@q4fv=i*p!4*to

2.2. Create the database

Change into the root folder of simple-shop if you have not already and execute **`python manage.py migrate`**. This will setup the database hierarchy and tables.

After the database is created, you can import the initial data with **`python manage.py loaddata data`**. This creates all product and category entries in the database, including an superuser called admin and password admin.

The information asked will consist out of *`username`*, *`email address`*, *`password`*.

Make sure your virtual environment is active!

In order to execute the manage.py script you'll have to have your environment active with the right python version, otherwise you'll encounter potential issues / errors.

2.3. Compile the LessCSS

The project is using LessCSS to write less CSS and align the CSS more with well structured code. There are a multitude of LessCSS compilers out there which can be used to compile the style.less in **`simple_shop/media/css/`**.

In case you can't compile

The project in it's current version doesn't make heavy use of css besides what twitter's bootstraps provides. You could just create **`style.css`** in **`simple_shop/media/css`** and add:

```
body
{
    padding-top:70px;
}
```

2.4. Start the webapplication

When you finished the configuration, please execute **`python manage.py runserver`** and head over to *http://localhost:8000* in your browser of choice.

The admin panel is accessible under *http://localhost:8000/admin/*[1]. The username and password when you used the initial data is "admin".

[1] http://localhost:8000

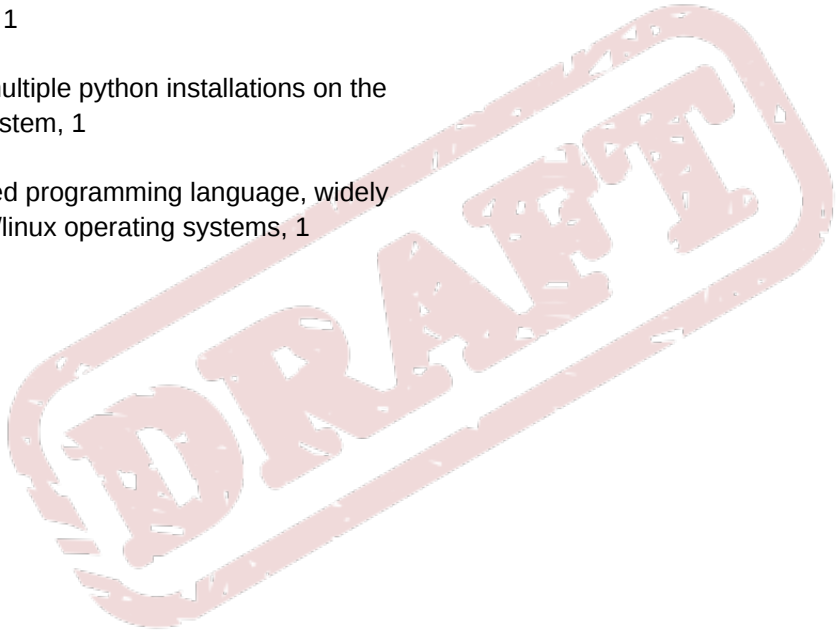
Appendix A. Revision History

Revision 0 **Tue Jul 14 2015** **Sascha Peter** *sascha.o.peter@gmail.com*

Initial creation of book by publican


DRAFT

Index




DRAFT