Step-by-Step

With Pelican installed, it's time to create the structure of your blog and configure it using the `pelican-quickstart` command.

Initial Setup

Running `pelican-quickstart` will guide you through setting up the basic framework for your blog. Below is an example of the process and the choices I made:

```
Welcome to pelican-quickstart v4.8.0.

This script will help you create a new Pelican-based website.

Please answer the following questions so this script can generate the files needed by Pelican.

> Where do you want to create your new web site? [.]
> What will be the title of this web site? My Awesome Blog
> Who will be the author of this web site? Your Name
> What will be the default language of this web site? [en] en
> Do you want to specify a URL prefix? e.g., https://example.com   (Y/n) Y
> What is your URL prefix? (see above example; no trailing slash) https://example.com
> Do you want to enable article pagination? (Y/n) Y
> How many articles per page do you want? [10] 10
> What is your time zone? [Europe/London] Europe/Brussels
> Do you want to generate a tasks.py/Makefile to automate generation and publishing? (Y/n) Y
> Do you want to upload your website using FTP? (y/N) N
> Do you want to upload your website using SSH? (y/N) N
> Do you want to upload your website using Dropbox? (y/N) N
> Do you want to upload your website using S3? (y/N) N
> Do you want to upload your website using Rackspace Cloud Files? (y/N) N
> Do you want to upload your website using GitHub Pages? (y/N) N
Done. Your new project is available.
```

Explanation of Choices

- **Source Directory**: We’re creating the blog in our current `blog/` directory, so the default `.` is used as the source.
- **URL Prefix**: This is your public URL of your future website.

After answering these questions, Pelican generates the basic structure and configuration files for your blog.

Blog Structure Overview

Here’s what the structure looks like:

```
blog/
??? .venv/               # Virtual environment
??? content/             # Directory for your articles and pages
 ?   ??? (images)/       # Directory for storing images
 ?   ??? (pages)/        # For non-chronological content (optional)
??? output/              # Directory for the generated site
??? tasks.py             # Automation tasks
??? Makefile             # Makefile for easier commands
??? pelicanconf.py       # Main configuration file
??? publishconf.py       # Configuration for publishing
```

Creating Content

I’ll be using Markdown for writing posts, though Pelican also supports reStructuredText.

All blog posts go into the `content/` directory. If you have static pages (like an About or Contact page), place them in the `content/pages/` directory.

Here’s an example of a simple first article:

```
Title: Hello, World
Date: 2024-08-17 12:42
Tags: hello
Category: hello
Authors: Your Name
Summary: A short introduction to my blog

# Hello, world!

Welcome to my blog, folks!
```

The first few lines are **metadata** that provides information about your article, which templates can use. Depending on the theme, you might be able to include additional metadata.

Publishing Your Blog

The Canonical Way

To generate your website, use the following command:

```
pelican content/
```

This command processes your content and generates the static files in the specified output directory.

Automating the Process

If you opted to generate a `tasks.py/Makefile`, you can use the predefined commands to simplify your workflow. For example, to start the development server and regenerate content on changes, use:

```
make devserver
```

This command will:

- Automatically regenerate your site as you edit content.
- Serve your site locally at http://127.0.0.1:8000.

Customizing with Themes

The default design might not be very exciting, but don't worry—you can easily change it by using a theme.

Pelican offers a wide range of themes, which you can browse here. Once you find one you like, you can integrate it into your blog.

Installing a Theme

For this example, let's say you've chosen the Attila theme.

1. Create a `themes/` directory in your `blog/` folder.
2. Download the theme and place it in the `themes/` directory.

Here's what your updated structure might look like:

```
blog/
??? .venv/
??? content/
 ?   ??? (images)/
 ?   ??? (pages)/
 ?   ??? hello.md
??? output/
 ?   ??? ...
??? themes/
 ?   ??? attila/
 ?       ??? ...
??? tasks.py
```

```
??? Makefile
??? pelicanconf.py       # Main configuration file
??? publishconf.py       # Settings for publishing
```

Next, install the theme using Pelican's built-in command:

```
pelican-themes -U themes/attila/
```

You can verify that the theme was installed with:

```
pelican-themes -l
```

Finally, update your `pelicanconf.py` file to use the new theme:

```
THEME = 'attila'
```

And that’s it! Your blog is now set up with a stylish new theme, ready to go live.

Happy me! □□

Revision #2
Created 17 August 2024 23:06:54 by Tiffanie BOREUX
Updated 16 October 2024 01:58:37 by Tiffanie BOREUX