mypelican_static_sites git project

Overview

This is a github project that uses a site generator called pelican to generate static websites.

At this writing this project maintains the content and generation for three seperate sites. Each site is rendered in 3 languages, english, french and spanish.

By convention we name each of these subdirectories using the domain name of the site. The scripts rely on this fact.

Currently Maintained Sites

1. www.bernatchez.net

   Hosted in Bunny.net: www-bernatchez-net

2. www.ogopogo.biz

   Hosted in Bunny.net: www-ogopogo-biz

3. blog.bernatchez.net

   Hosted in Bunny.net: blog-bernatchez-net

4. docs.pelican.bernatchez.net

   Hosted unpublished on local file system.

Web site generation

Generate a local version of a site so that you can browse and review it before publishing it.

```
cd ./blog.bernatchez.net
make html
xdg-open ./bucket/index.html
cd ..
```

Generate an online version of the site and publish it.

```
cd ./blog.bernatchez.net
# Make sure the deploy.bunny file actually runs an "nmx" command rather than just echoing it.
make pubhtml

# Manually Invoke the command printed as the last line by the above make
# This pushes everything out to CDN cache
npx --yes bunny-transfer@latest purge blog-bernatchez-net-zone
```

Warning

Once you have invoked "make pubhtml", your "local site" content will no longer be appropriate for local viewing. If you wish to browse both local and online versions interchangebly. You need to repeat the "make html" once you have published thereby leaving the local files configured for local viewing.

Web Site Maintenance

Scope

The git project is organized to have 2 scopes for site source content.

Global source scope

The .rst files contained in subdirectories of the main project directory. These files can be sourced by any of our websites.

Local source scope

The .rst files contained within subdirectories under a given site subdirectory. These files can only be sourced by the given website.

A given site's "content" subdirectoy contains subdirectories whose names match either global or local scope subdirectories. That identifies what will be included in the site.

What directory names appear in the "content" directory determines what is included in the generated site.

The make file handles populating such directories with with soft links to the .rst source files in their corresponding local or global scope directory.

Insofar as the repository is concerned you only "git add" .rst files which are in global or local scope directories. The subdirectories under content/ will only contain "placeholder.txt" within git to make sure their subdirectories are created upon clone.

pdf source

All our sites also contain .pdf versions of their pages. To produce those pdf files, each source directory also contains two executable script files: **article2pdf**, and **qarticle2pdf**. The latter is just a less verbose version. Whenever you modify or add content in a source directory, remember to re-regenerate the .pdf files accordingly.

Generally, you modify your content, and then invoke "qarticle2pdf" to make sure your pdfs are up to date.

If you have created a mew source directory you can generate those two scripts in it by invoking the command "genarticle2pdf -q" inside the new directdory.

github maintenance

When you clone the github project to a local directory, you end up with a main directory which contains web site subdirectories, each of those is an independant web site generation directory.

The project main directory contains various *Global source scope* site subdirectories (E.G. ./examples, ./artwork, ./travel, etc...). Each of these contain .rst files that can serve as source for site content generation. All under git revision control.

This *Global source scope* content can optionally be used as source for any of the web sites.

Each web site subdirectory can also contain its own subdirectories containing site specific *Local source scope* .rst files (E.G. ./blog.bernatchez.net/pages/) intended only for the given site.

To add any one of these *local or global scope* content directories to a given web site, you need to create a directory of the same name under the given site's content subdirectory. You must also add a "placeholder.txt" file and put it under revision control (e.g. git add placeholder.txt) to enssure that new directory will be created upon use of "git clone".

For example we did this to add ./examples to the ./blog.bernatchez.net web site:

```
mkdir ./blog.bernatchez.net/content/examples/
cp ./blog.bernatchez.net/content/pages/placeholder.txt ./blog.bernatchez.net/content/examples/
git add ./blog.bernatchez.net/content/examples/placeholder.txt
```

```
You also need to modify the project .gitignore file to avoid adding
the generated soft links under content/examples into the git
repository.  Those links are regenerated every make.

We added the lines below to .gitignore:

blog.bernatchez.net/content/examples/
blog.bernatchez.net/content/examples/*
```

Repository

The original central copy of this project's repository is:

git@github.com:pierrebernatchez/mypelican_static_sites.git