Note on Deploying a Website on Github

Author: Di Yu (email: yudi.0211@foxmail.com)

Create a Markdown File

---

1. Write note with Markdown.

Convert the Markdown File to a Local Website

Install hugo and configure the website folder

1. Install Hugo following the guidance.
2. Install powershell.
3. Open `powershell` and change the directory to the location you want to place the website files by running `cd <loc>`.
4. Create a new site using Hugo by running `hugo new site <site_name>` in `powershell`.
5. Edit the `hugo.toml` file, leaving `baseURL` empty for now.
6. Change the directory to the website folder in `powershell`.
7. Create a document content by running `hugo new docs/test.md`.

Install the theme and add contents

1. Run `git init`.
2. Find a nice theme at Hugo –> Download –> Install (run the `git clone` command under method 1 and rugo n the `git submodule add` command under method 2).
3. Edit the `hugo.toml` file, adding `theme = <theme_name>`.
4. Edit the content file, setting `draft` to `false` and add some contents.
5. Run `hugo server` and access it locally at `http://localhost:1313/`.

Deploy the Website on Github

Push website framwork to Github

1. Create an empty github repository.
2. Copy the `git remote add origin <website_path>.git` command.
3. Add a `README.md` file and git push –> `git add README.md` –> `git commit -m "initial commit"` –> `git branch -M main` –> `git remote add origin <website_path>.git` -> `git push -u origin main`.

Setup a branch for Github pages

1. Create a `gh-pages` branch.
2. Go to repository's `settings` –> `action` –> `general` –> enable `Read and write permissions` –> `save`.

Create a Github workflow

1. Create an empty file in your local repository at `.github/workflows/hugo.yml`.
2. Copy the YML specified in Deploying a Blog Powered by Hugo to Github Pages w/ Custom Domain via Github Actions to the `hugo.yml` file.

3. Update the Hugo source link, `user.name`, and `user.email` in the `hugo.yml` file.
4. Modify the `hugo.toml` file –> `baseURL = 'https://nagato-D.github.io/<repo_name>/'`.

Git push

1. Run `git add .` –> `git commit -m "add the first test page"` –> `git push`.
2. Waiting for Github to deploy your website.
3. Access your website at `https://<user_name>.github.io/<repo_name>/`.

How to Update the Website

---

1. Add (`hugo new <path/filename>`) or modify the markdown files in the `content` folder (remember to set `draft` keyword to `false` for markdown files in the `content` folder).
2. Run `hugo server` to rebuild the website locally.
3. Post the website online: `git add .` –> `git commit -m "message"` –> git push.
4. Verify the release.

How to Configure Website

---

1. Copy the contents in `themes/<theme_name>/config.toml` to `hugo.toml`.
2. Modify the parameters specified in `hugo.toml`, such as website title, avatarURL, and social links.
3. In order to add a menu in the homepage, like the tags in the right top, following the menu setting in the `config.toml` file to configure the `hugo.toml` file in the website root directory.
4. A recommended file structure/organization of the contents can be found here.

Add Icons

- LoveIt utilizes the icon library provided by Font Awesome for seamless integration of icons into your content.
- Icon references adhere to markdown syntax. To incorporate an icon, simply use the format `:icon_name:`, where the specific `icon_name` can be found on the Font Awesome website.
- As an illustration, let's say you want to insert a code icon. You can locate the corresponding HTML code here, which is `<i class="fa-solid fa-code"></i>`. Subsequently, you can introduce the code icon using `:fa-solid fa-code:`, with the `fa-solid fa-code` (namely `icon_name`) derived directly from the HTML code.

Reference

---

1. Deploy Hugo Blog to Github Pages via Github Actions w/ a Custom Domain
2. Deploying a Blog Powered by Hugo to Github Pages w/ Custom Domain via Github Actions
3. Hugo: Host on GitHub Pages
4. LoveIt Theme Documentation - Content