Website content guide — Sister City

Overview

This is an instructional guide to help you make edits to text content on the Sister City website.

The website frontend is a powered by a complex JavaScript application called Vue. You don't need to know much about Vue; only that you probably don't want to be mucking around in its files regularly. Project scope didn't allow for a dedicated graphical content management system, but we didn't want to leave you out in the cold, so we've created a very simple way for you to edit some site text.

Web UI feature guide →

What you need

To get started, you'll need a rudimentary understanding of GitHub and a way to edit text files.

- A GitHub account
- Permission to access to the Sister City repository
- The GitHub desktop app
- A text editor, like Sublime Text — or just use the GitHub text editor

It's beyond the scope of this document to give a full GitHub tutorial. The short of it is that GitHub is a version control service for code. Developers like to keep versions of code so they can "roll back" to a previous version if something goes wrong. Check out the GitHub guides if you're just starting out.

Organizations have different ways of working in GitHub; you'll need to work with your development team to figure out your workflow. We suggest something like this:

- Create a new branch for your work. Don't work on master. Use your initials in the branch name, so it's easy to identify who's working on what, e.g. `FD-content-edits`.

- Make the edits you need to make.
- Commit your changes frequently, with brief, useful messages. Other folks read these!
- When you're done, create a pull request. This tells other developers on the project that you think your work is ready to be merged into master.
- Someone else in your organization should review the pull request. They can approve it, or request changes.
- When your pull request is approved, merge it to master. Once it's merged, you can delete the branch you were working on — you don't need it anymore.

What can be edited, and what can't (easily)

You will be able to edit:

- 404 page
- All the text in the *Info* view ("Something Else"), like About, Food & drink, FAQ, and so on
- Hotel manual
- *Your stay* and *Requests* menu text
- Rooms details text and slideshow

Like we said, the website frontend is powered by a JavaScript application. There's a lot of text, mostly interface text and some error messages, that won't be easily editable by the layperson. But nothing's impossible, and a developer on your team should be able to help you make a change to something if it becomes necessary.

You'll need a developer's help to find, edit, or otherwise modify things like:

- Interface text, like buttons, form labels, and page headers and introductions
- Items in the requests list
- Error messages

File locations

The files are hosted on a web server, and we manage their versions in a GitHub repository (repo). Once you've got a local version of the site up and running, you'll be able to navigate the site's backend directory structure using Finder.

All of the files you'll want to edit are in the `src > assets > data` directory. If you need to change something that isn't in there, ask a developer for help. Trust us, you really do not want to poke around in those other files.

The files are named in a way that should make it pretty clear what's what. Here's a rundown.

- `404.md` — The 404, File not found page is where folks end up if they access a bad link.

Info ("Something Else")

These files are all in the *Info* view, called "Something Else" in the site's navigation. Each file is appended with it page name, e.g., the About page is `info-about.md`.

- `info-about.md`
- `info-contact.md`
- `info-faq.md`
- `info-food.md`
- `info-press.md`
- `info-privacy-policy.md`
- `info-rooms.yml` — File with data pertaining to room type, description, photo, floor plan, plus the intro text on this page.

Requests

These files pertain to the *Requests* view. This is one of the most complex parts of the platform

- `requests-menu.yml` — The top-level menu in requests, which splits out to *Supplies* and *Services*. You can edit the text here, but you'll need to work with a developer to add, modify, or delete entries.
- TK TK TK TK

Booking

These files relate to pages and data accessed during the booking process.

- `reservation-policy.md`
- `rooms-details.md` — Rooms details a user can view when she opens the sidebar panel from the rooms and rates view.
- `rooms-slideshow.yml` — A gallery accessible from the *Rooms + rates* view.
- `rooms-unavailable.md` — Text displayed on the *Calendar* view flash message when the hotel is sold out encouraging the user to book at Ace NY.

Your stay

- `stay-bailout.md` — The "bailout" help text that's appended to each stay page.

These files are part of the *Hotel manual* view. The manual index is compiled automatically elsewhere, so you'll need to work with a developer to add, modify, or delete entries.

- `stay-manual-ada-requests.md`
- `stay-manual-basics.md`
- `stay-manual-fire.md`
- `stay-manual-food.md`
- `stay-manual-housekeeping.md`
- `stay-manual-money.md`
- `stay-manual-music.md`
- `stay-manual-necessities.md`
- `stay-manual-neighborhood.md`
- `stay-manual-packages.md`
- `stay-manual-safety.md`
- `stay-manual-smoking.md`
- `stay-manual-telephone.md`
- `stay-manual-television.md`
- `stay-manual-transportation.md`
- `stay-manual-wake-up-calls.md`
- `stay-manual-welcome.md`
- `stay-menu.yml` — The menu for the entire Stay section, including section titles and descriptions. You can edit the text here, but to add or delete sections, you'll need to work with a developer.

File types

There are two file types in the directory; both are text files. Markdown files are appended with `.md` and YAML files are appended with `.yml`. Markdown is super easy to work with, and we've written up a basic guide below. YAML is more picky about syntax, so you'll likely want to work with a developer to edit any content in YAML files. In short, Markdown files turn into HTML, while YAML files turn into data to be used in some other part of the site.

Markdown

Markdown is a text-to-HTML conversion tool for web writers. Markdown allows you to write using an easy-to-read, easy-to-write plain text format, which we can easily

convert to HTML.

Daring Fireball, GitHub Guides, and Learn X in Y Minutes have great tutorials to help you learn Markdown. If you're feeling nervous about jumping into files right away, you can practice writing Markdown using John Gruber's Dingus tool. You can preview Markdown right in GitHub, too. Below, we'll cover some basics.

Syntax

Paragraphs

A paragraph is just one or more consecutive lines of text separated by one or more blank lines. (A blank line is just an empty line with nothing on it.) For single line breaks (e.g., an address), follow your line with two spaces and a hard return. Many text editors will strip out trailing spaces. If this happens, you can re-edit the text (space, space, return) in GitHub's editor and re-commit.

Some pages have larger text style on the opening paragraph — this happens automatically in the code; there's nothing you have to do to create this effect.

Headers

To create a header, preface the header text with the pound sign — one for a top-level header, and two for a second-level header. Remember to put a space after the pound sign.

```
# This is a primary header
## This is a secondary header
### If you really need one, here's a tertiary header
```

You should need only primary and secondary headers in the text, though no matter which header you use, it'll look the same on the website (though not in GitHub preview) — there's only one header style on the site. But it's worth structuring the content a little bit for assistive devices and SEO bots.

Lists

You can create a bulleted list using either asterisks or dashes before each list item.

```
* List item
* Another list item
```

```
* Yet another list item
```

Ordered lists use regular old numbers followed by periods.

```
1. List item 1
2. List item 2
3. List item 3
```

Links
For offsite links, wrap link text in brackets, and the destination URL in parenthesis immediately following the link text.

```
This paragraph links to [Sister City](http://hellosister.city).
```

Linking within the Sister City platform is more complicated because the site is a dynamic application, not a collection of static pages. Work with a developer if you need to link to another part of the site.

Images
Image syntax is a lot like link syntax. Preface an image with an exclamation point, wrap the alt text in brackets, and include the relative path to the image in parenthesis. (Alt text is what's read by screen readers and other assistive devices and should be a very short note about the image.) If the image gets a caption, put it right below the image itself, wrapped in asterisks.

```
![alt text](/path/to/img.jpg)
*Caption goes in asterisks right below the image.*
```

Image files should be uploaded to `static > images`. You can use the GitHub desktop app to add and commit images, or work with Vinh.

If you add room renderings, they should be transparent pngs. If they must have a background color, it should be white. (We multiply the image background color to match the site background color, to account for discrepancies in color rendering across devices.)

YAML

YAML is a human-readable data serialization language. This means that it's really good for things like configuration files, or other places where data is stored and read by computers, but it's also easy for humans to understand.

Learn X in Y Minutes has a good YAML tutorial. You can also test-run your YAML using a validator. Because YAML has some picky rules about structure and syntax, we recommend that you work with a developer if you need to edit content in a YAML file.

Structure and syntax

The top-level structure of a YAML file can be either a list (array) or object (property/value pairs). We usually use an array. But each structure can be nested inside the other.

```
- id: 1
  title: First item
- id: 2
  title: second item
```

```
property: value
other_property: other value
third_property:
  - contains a list of strings
```

Here's a real-life example from `info-rooms.yml`.

```
header: "All rooms have private bathrooms, soft beds, windows,
a TV, a bluetooth speaker, and custom bath products."
list:
- title: Single
  description: Our coziest little room, with a standard double
-sized bed. Sleeps one human, or two small ones. Average size
```

```
150 square feet.
  photoPath: '/static/images/room-renderings/SC_rendering_Queen_web.jpg'
  floorplanPath: '/static/images/floorplans/single.svg'
```

YAML files use whitespace indentation to denote structure. In the example above, the hyphen on the third line means that this text defines an item in a list. All of the property names (`description`, `photoPath`, `floorplanPath`) belong to this item and are indented accordingly. YAML allows you to condense the hyphen and the first property into a single line to save space.

To add a new item to the list, add a new line that is not indented and begins with a hyphen. The property names should be identical as well. We recommend copying an existing item and pasting it where you need it, then changing the values. Just make sure the indentation lines up correctly. (This can be hard to do in the GitHub editor, but simple in a stand-alone text editor.)

Comments are prefaced with a pound sign.

```
# This file controls the content of the slideshow component in the Rooms view.
```