About This Example Site

This example site demonstrates the capabilities of Typst SSG. Typst SSG takes `.typ` files and compiles them into PDFs, which are then converted into static HTML pages using a custom PDFium-based PDF viewer, which supports text selection and clickable links.

Technical Specifications

Feature	Description
Font	Junicode (main text), JetBrains Mono (code)
Page Width	`42em`
Margins	`2em` horizontal, `2.5em` top, `3em` bottom
Background	Off-white (`#fefefe`)
Link Color	Blue (`#2980b9`)
Rendering	PDFium WASM in browser

Component Example

Custom "components" can be created by defining Typst functions to render specific elements with pre-defined styles. For example, this site uses a `card` component defined in `src/components/card.typ` to create styled content blocks.

```
#import "components/card.typ": card

#card(
  title: "Card Title",
  meta: "Card Meta Info",
)[
  Card component example.
]
```

Card Title

Card Meta Info

Card component example.

Layout Files

Layout files can be defined to create consistent structures across multiple pages, by defining a `#let layout(body) = {...}` function that wraps the page content with specific styles and elements. `layout` functions are defined in an `index.typ` file within the directory of the pages that should use that layout.

The file tree gets read from `src/pages` for each page, the parent directories are chalked for `index.typ` files that define `layout` functions. There are three layout inheritance modes (which can be set in `tssg.config.js`):

1. `none`: only checks the immediate directory for a layout file.
2. `fallback`: uses the closest layout file found in the directory hierarchy.
3. `merge`: merges all layout functions from the page up to the root. The `set` and `show` functions from all layout files are applied, with the closest layout file taking precedence.

In this example site, the `layoutInheritance` mode is set to `merge` in the `tssg.config.js` file. The `index.typ` file in the `src/pages` directory defines the base layout for all pages. The `blog/` and `projects/` sub-directories each have their own `index.typ` files that define specific layouts for the pages within these directories, which are merged with the base layout.

Layout File Example

The `index.typ` file in the root `src/pages/` directory defines the base layout for all pages. The `layout` function takes the page content as the `body` parameter and applies `show` and `set` rules to style various elements, as well as defining elements that should appear on all pages. Note that the content elements such as the navbar are *not* inherited by sub-directory layouts; only the `set` and `show` rules are merged. Setting page properties:

```
set page(
  width: 42em,
  height: auto,
  margin: (x: 2em, top: 2.5em, bottom: 3em),
  fill: rgb("#fefefe")
)
```

Setting text properties, heading styles, and link styles:

```
set text(
  font: "Junicode",
  size: 11pt,
  fill: rgb("#040608")
)

set heading(numbering: none)

// ...

show link: it => underline(text(fill: rgb("#2c3e50"))[#it])
```

Defining how raw code blocks should be displayed:

```
show raw.where(block: true): it => {
  set text(
    size: 1em,
    fill: rgb("#333")
  ) if it.at(
    "label",
    default: none
  ) != <tssg-code>

  if it.at(
    "label",
    default: none
  ) != <tssg-code> {
    block(
      width: 100%,
      inset: 1em,
      radius: 0.2em,
      fill: rgb("#f5f5f5"),
      stroke: 0.5pt + rgb("#ddd"),
      it,
    )
  } else {
    it
  }
}
```

The above rule defines styles for all raw code blocks, giving them a light gray background, some padding, and a border. It excludes code blocks labeled `<tssg-code>`, which are used internally by the `#code-block` and should not be styled.

CSS Styling

Most styling is done within the Typst document itself. However, to change the overall appearance of your site, you can create a CSS file in the root of a directory.

For example, the `index.css` file in the `src/pages/` directory changes the background color of the entire page, to match the Typst page background color:

```
#pdf-container {
  background: #fefefe;
}
```

Getting Started

To create your own site using Typst SSG:

1. Install the Typst SSG framework.
2. Download the Typst SSG utility package.
3. Run `tssg init` to set up a new project.
4. Create your pages in the `src/pages` directory as `.typ` files.
5. Run `tssg dev` to start a local development server and `tssg build` to generate the static site.
6. Deploy the generated static files to a hosting service.

Documentation

- Typst Documentation
- Typst SSG Documentation

For more examples, visit the Blog or check out the Projects page.