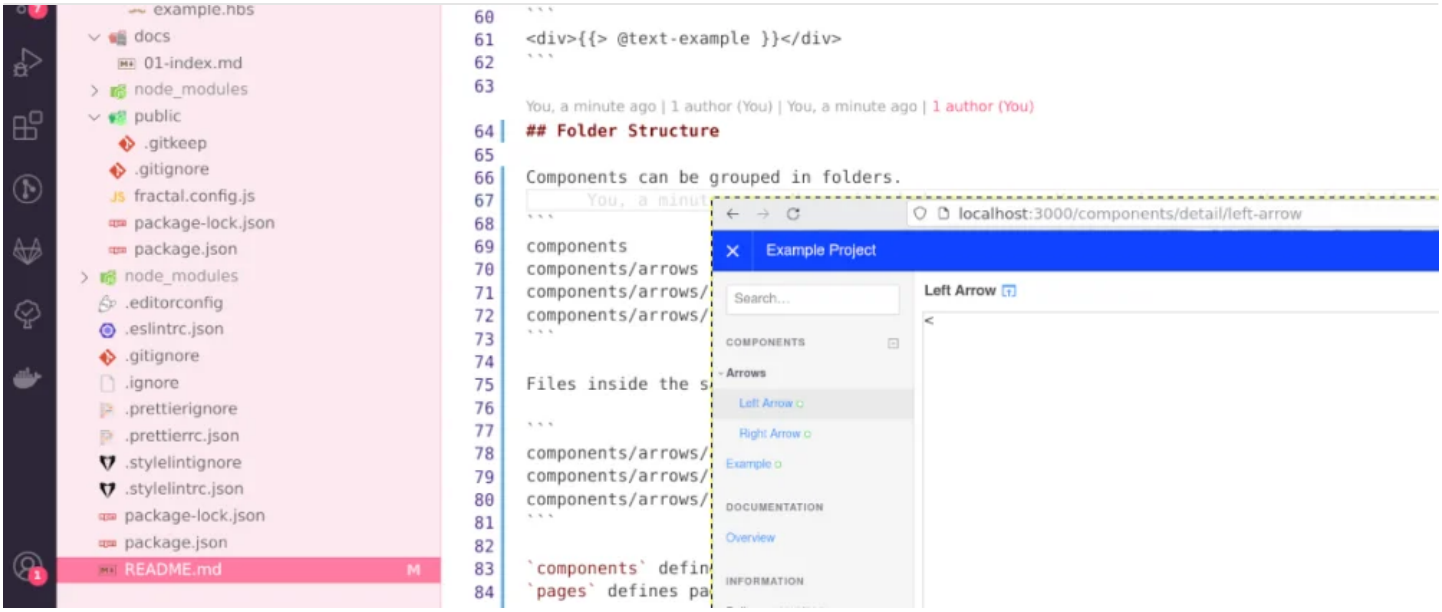

Ingo Steinke
Posted on 19 Oct 2021 • Updated on 9 Dec 2021

fractal.build as an atomic design tool

#todayilearned #programming #webdev #javascript

Recently, I discovered a tool that helped me build a design system: fractal.build. Described in by Rachel Andrew in her article Pattern Library First some years ago, fractal does look a little old school, but it can be customized and does a good job without getting into your way.

to do

Add testing tools and more complex examples to my example setup. Codecept.io seems like a promising candidate for a more lightweight, simple and easy solution to add some front-end tests to prevent regressions, compared to cypress, storybook, and storyshots. Evaluation results will be pushed to GitHub and updated here.

alternatives to fractal

Fractal looks less shiny than Storybook, that I have used for ReactJS projects, but it can easily be used for projects without any JavaScript framework.

Fractal seemed easier, at least to me, to understand and maintain, than PatternLab, which I failed to install due a bug in the current installer (and when I managed to install

the grunt version, I was already told that there is fractal as a possible alternative).

atomic design and design systems

So what are design systems and what is atomic design?
Much has been said and written about CSS methodologies like BEM, ABEM, ITCSS, and utility-based approaches like Tailwind or Bootstrap. Follow the links for further reading, if you like.

agnostic fractal

Fractal is quite agnostic about tools, methods, and coding style. Which also allows for a pragmatic approach that does not adhere to one single methodology.

The default setup allows you to build and compose components using handlebars, HTML, and CSS. Fractal can be customized to use any other markup language like Twig or Nunjucks, so you could probably use Liquid markup for a JAMStack setup with 11ty as well.

boilerplates to start with

Other users have created boilerplates for using ABEM CSS in fractal or ditching handlebars to use fractal with twig templates instead.

To use CSS on a component level, you can add a tool chain of your choice (or just the first copy-and-paste-able example you find on Google), like SASS or PostCSS, together with a build process based on Webpack, Gulp, or plain Node.js.

avoiding webpack

In my first fractal.build example, I started with a gulp setup with SASS for a quick proof of concept, changing the setup to use the popular FractalWebpackPlugin without having to modify any component code.

While webpack may be a valid choice for maintaining single page applications mainly written in JavaScript, and I do not recommend it for a simple front-end setup that aims to produce static CSS files. (A rant about webpack, its ecosystem, and its breaking changes with every major release would fill another article of its own.)

In a future JAMStack project, I would rather go for PostCSS to use native CSS 3 / CSSnext features and try to avoid unnecessary tool dependencies.

But still, after changing one's mind about tools or language choices, any existing code

could be refactored easily while keeping the same folder structure.

advantages and suggestions

Apart from its agnostic and pragmatic approach, fractal has some other advantages.

preview theme customization

Fractal's user interface can be themed / customized, so we do not have to stick to the original UI. We can set colors, logo, and fonts to match our customers' corporate design before a presentation.

component composition

Components can include other components, so we can build a design system bottom-up starting with colors, icons, buttons etc. to be used in forms, paragraphs, sliders, navigation which can then be composed to larger blocks and pages.

variants

Components can have variants, either by configuration (in a config file) or by using file names accordingly, like in this example:

```
src/coponents/my-component/
  my-component.config.yml (or .json)
  my-component.hbs (default variant)
  my-component.css (classes used by my component)
  my-component--with-arrow.hbs
  my-component--with-arrow-without-borders.hbs
```

This can get confusing quickly, but you can (mis)use the default variant to display an overview page.

```
<!-- my-component.hbs -->

<h2>Component with Arrow</h2>
{{>@my-component--with-arrow}}

<h2>Component with Arrow but without Borders</h2>
{{>@my-component--with-arrow-without-borders}}
```

configuration

Use `fractal.config.js` in the project root directory to configure paths and options.

outputs paths

outputs paths

Don't confuse `static.path` where the assets are built for the development preview (localhost server) and `builder.dest` where a static HTML version is built after running `fractal build`.

Static component files have a timestamp by default, so every file seems to be changed after a build, even if you only modified a single component. If you want to automate regression testing, or if you have to commit the static builds, you don't need that noise.

As fractal developer Mihkel Eidast helpfully explained, you can use a custom theme to override the timestamp in `fractal.cofig.js`:

```
const mandelbrot = require('@frctl/mandelbrot');

const myCustomisedTheme = mandelbrot({
  information: [{ }],
});
fractal.web.theme(myCustomisedTheme);
```

disadvantages

Some aspects to consider before choosing fractal:

invalid markup breaks the preview

Some invalid markup can break larger parts of your preview. One single mistyped character inside a handlebars include will show an error message instead of the preview, and this error can bubble up to composed higher order components and overview pages.

learn to understand handlebars

Nesting complex components requires some handlebars knowledge, especially when dealing with optional values. And be careful to clear attributes before inadvertently passing them to descendant nodes, like a CSS class name that you want to set on a parent, but not on every child and grandchild element.

Poor syntax highlighting and missing linting and IDE assistance for handlebars made me list handlebars as a slight disadvantage, but if you handle it correctly, handlebars does a good job!

component names must be unique

This might be an advantage or a disadvantage, according to your own point of view: while components can be nested and composed, there is no hierarchy.

Instead, all components exist on the same level and share the same namespace, so their technical names have to be unique.

you must do it by yourself

Apart from its agnostic and pragmatic approach being an advantage for me, it might be a disadvantage to you.

Fractal is just a tool, and quite a simple one, at least when you have experience with other tools and frameworks. It is up to you to complete the setup by making further choices and implementations.

conclusion

Despite fractal being not the latest fad (or maybe even because of that) I have discovered it as a practical development and preview tool that does not get in your way.

Ingo Steinke

Web Development, Sustainability, Art and Music, Nature and Travel, Sustainability

LOCATION
Germany

WORK

Creative Web Developer at Ingo Steinke

JOINED

21 Sep 2019

More from Ingo Steinke

Animated Gradient Text Color

#webdev #showdev #css #tutorial

Aspect ratio: no need for container units!

#css #html #webdev #todayilearned

CSS :has(.parent-selectors)

#css #webdev #todayilearned #javascript