Docs Components

The Gatsby docs site has a handful of components that have been developed to facilitate writing new content for the docs. There are also components that help organize and lay out content in various pages across the website.

This guide documents what components are available and explains how to use them. You can also refer to the code for this page on GitHub to see to how each component can be used, because they are all embedded here!

Information about authoring in Markdown on the site is also listed.

Globally available components

A variety of components have been written to help with formatting code and content across the docs and don't need to be imported.

Guide List

The `<GuideList />` is a component that renders an `h2` heading and a list of links to child docs nested below the current doc in the site's information architecture. It is often used on overview pages where many other pages are nested below it to show what a docs section contains.

Usage The Guide List component takes one prop:

- `slug` (required) - the value of which is already available on every page's context on the site by default

The slug is used to find a matching value in one of the sidebar files that powers this website. It finds the matching entry in the hierarchy and renders the pages that are children of it in a list.

The component can be used like this:

```
---
title: Headless CMS
---
```

```
Overview information about headless CMSs...

<GuideList slug={props.slug} /> // highlight-line
```

Sample When rendered in a page, the Guide List looks like this:

Egghead Embed

To embed video content from Egghead on the site, there is an `<EggheadEmbed />` component that adds an iframe with the video inline with other content.

Usage The Egghead Embed component takes two props:

- `lessonLink` - the URL of the lesson from Egghead
- `lessonTitle` - the name of the lesson that is used to add more information to the embedded iframe.

It can be used like this:

```
---
title: Using Gatsby Image
---

This is how you use `gatsby-image`...

<!-- highlight-start -->

<EggheadEmbed
  lessonLink="https://egghead.io/lessons/gatsby-use-gatsby-image-s-graphql-fragments-for-blu
  lessonTitle="Use gatsby-image's GraphQL fragments for blurred-up and traced SVG images"
/>

<!-- highlight-end -->
```

Sample Rendered, it looks like this:

Pull Quote

The `<Pullquote />` component is used to call out a quote. It applies borders and styles that make a section of the content pop out to readers.

Usage The Pull Quote component takes two optional props, and uses the children it wraps to populate its inner content:

- `citation` - the reference of the person or entity that made the quoted statement
- `narrow` - styles the pull quote by removing the left and right negative margins, keeping it inside the parent container. This prop could be used in docs where it's necessary to keep content from overlapping with other sections of the layout, such as the Table of Contents.

It is used like this:

```
---
title: Doc Name
---

<!-- highlight-start -->
<Pullquote citation="Winston Churchill" narrow={true}>
  To improve is to change, so to be perfect is to have changed often.
</Pullquote>
<!-- highlight-end -->
```

Sample Rendered, the component looks like this:

To improve is to change, so to be perfect is to have changed often.

Gatsby Cloud Callout

The `<CloudCallout />` component is used to call attention and link to Gatsby Cloud. It wraps a small call to action in styles that make content stand out.

Usage The Cloud Callout component takes one optional prop, and prepends the children it wraps to the general call to action:

- `narrow` - styles the component by removing the left and right negative margins, keeping it inside the parent container. This prop is by default set to true since it's most commonly used in the docs where it's necessary to keep content from overlapping with other sections of the layout, such as the Table of Contents.

It is used like this:

```
---
title: Deploying to Storage Provider
---

<!-- highlight-start -->
<CloudCallout>
  Connect your Gatsby site to Storage Provider for automatic deployments.
</CloudCallout>
<!-- highlight-end -->
```

Sample Rendered, the component looks like this:

Connect your Gatsby site to Storage Provider for automatic deployments.

Component Model

The `<ComponentModel />` was made to help explain concepts of how Gatsby works at a high level. It conceptually breaks Gatsby into different layers and shows how data is pulled, aggregated, and eventually rendered as an app. It's used on docs pages to help explain how Gatsby works at different levels.

Usage The Component Model component takes one prop:

- `initialLayer` - defaults to `Content`, can be one of `Content`, `Build`, `Data`, `View`, `App` that correspond to the different layers

```
---
title: GraphQL Concepts in Gatsby
---

To help understand how GraphQL works in Gatsby...

<ComponentModel initialLayer="Data" />
```

Importing other components

If you need to use a component that is not globally available, you can do by importing it using the special `@components` alias, which points to `www/src/components`:

```
import EmailCaptureForm from "@components/email-capture-form"

<EmailCaptureForm />
```

NOTE: Do *not* import a component using relative path directories:

```
// DO NOT DO THIS
import EmailCaptureForm from "../../www/src/components/email-capture-form"
```

Doing so will break localized versions of the page, which are stored in other repos.

Writing content in markdown

New docs are added to the docs folder inside the Gatsby repository. They are stored as `.md` (Markdown) or `.mdx` (MDX) files and can be written using Markdown, or using inline JSX thanks to MDX. Writing in Markdown will output tags that are styled according to Gatsby's design guidelines.

You can read more about writing in Markdown in the Markdown syntax guide.

Frontmatter

Frontmatter is a set of key-value pairs in your Markdown and MDX files that defines the metadata for a page. While authoring new docs for the Gatsby site, it may be helpful to understand what frontmatter is available to you.

General

- `title` (string)

  The title of the doc. Gatsby renders the value in `og:title`, `<title>` and `<h1>`.

- `excerpt` (string)

  The excerpt for the post. Gatsby renders the value in `description`, `og:description`, and `twitter:description`.

Documentation

- `description` (string, default `excerpt`)

  A description of the doc. Gatsby renders the value in the `description` and `og:description` metadata.

- `issue` (string)

  The issue URL relating to a stub on GitHub. Gatsby renders a link to the stub.

- `disableTableOfContents` - (boolean)

  Determines if the table of contents is output.

- `tableOfContentsDepth` - (integer)

  The number of levels to render the table of contents.

Relevant links

- About Twitter Cards
- Facebook Sharing - Best Practices
- Making Your Website Shareable on LinkedIn

Code blocks

Code can be formatted using regular Markdown syntax, but the docs site has additional enhancements that can be used thanks to various Gatsby plugins that aren't all native Markdown.

Usage Code blocks are wrapped in 3 backticks. A language can be added immediately after the first set of back ticks to provide syntax highlighting for the language. A `title` of the file can be added after the language. Line highlighting can be included in the code block by commenting `highlight-line`, or `highlight-start` followed by a `highlight-end`.

````
```javascript:title=gatsby-config.js
// In your gatsby-config.js
plugins: [
  {
    resolve: `gatsby-transformer-remark`,
    options: {
      plugins: [
        `gatsby-remark-prismjs`
      ],
    },
  },
]
```
````

In order to demonstrate how to use code blocks in a doc without your triple backticks being styled and formatted automatically (just like the example above), you can wrap a set of triple backticks in quadruple backticks like this:

`````
````
```javascript:title=gatsby-config.js
// In your gatsby-config.js
plugins: [
  {
    resolve: `gatsby-transformer-remark`,
    options: {
      plugins: [
        `gatsby-remark-prismjs`
      ],
    },
  },
]
```
````
`````

Sample The above code block is rendered like this:

```javascript:title=gatsby-config.js
// In your gatsby-config.js
plugins: [
  {
    resolve: `gatsby-transformer-remark`,
    options: {
```

```
      plugins: [
        `gatsby-remark-prismjs`
      ],
    },
  },
]
```

Line numbers and line highlighting can be added to code blocks as well, and is explained in detail in the `gatsby-remark-prismjs` README.