

Writing good documentation is important for your project maintainers (and for your future self). [Docz](#) is a very nice documentation generator. It enables you to write interactive documentation for your React components with very little effort.

Docz leverages `mdx` files – short for Markdown with JSX – which brings **React components** to Markdown files. From your PropTypes, or Flow types or TypeScript types, it can generate **property tables** to document properly how to use your components. In addition, you can provide a **coding playground** for your components, so that anyone can see them in action, modify the code and see the changes live, or copy the snippet to use it somewhere else.

If you're starting your Gatsby project from scratch and would like to have great documentation, with Docz support out of the box, you can use [gatsby-starter-docz](#). You can also find more information at the end of this guide in the [Other resources](#) section.

Alternatively, the following guide should help you to get Docz working within an existing Gatsby project.

Setting up your environment

First, if you do not have a Gatsby project set up yet, use the Gatsby CLI to create a new site:

```
gatsby new my-gatsby-site-with-docz
```

To set up Docz you need to install the Docz Gatsby theme and add some custom configuration. Make sure you are in the root directory of your Gatsby project:

```
cd my-gatsby-site-with-docz
```

And install required packages:

```
npm install gatsby-theme-docz docz
```

Add `gatsby-theme-docz` under `plugins` in `gatsby-config.js`:

```
module.exports = {
  plugins: [
    //highlight-next-line
    `gatsby-theme-docz`,
    // Your plugins go here
  ],
}
```

Writing documentation

Docz searches your directories for `mdx` files and renders them. Create a `docs` folder at the root of your project. Place an `index.mdx` file inside this directory with the following content:

```
---
name: Getting Started
route: /
---
```

```
# Getting Started

## Hello world

Type here the most beautiful getting started that you ever saw!
```

Run the development server with `gatsby develop` and you should be greeted with the default Docz layout and a "Getting Started" heading. Stop the development server after verifying that everything works.

Let's spice things up by adding and rendering a React component. For the sake of simplicity you can create the following button component in the same `docs` directory.

```
import React from "react"
import PropTypes from "prop-types"

const scales = {
  small: {
    fontSize: "16px",
  },
  normal: {
    fontSize: "18px",
  },
  big: {
    fontSize: "22px",
  },
}

export const Button = ({ children, scale }) => (
  <button style={scales[scale]}>{children}</button>
)

Button.propTypes = {
  children: PropTypes.node.isRequired,
  scale: PropTypes.oneOf(["small", "normal", "big"]),
}

Button.defaultProps = {
  scale: "normal",
}
```

The button will display its text by default with a `font-size` of `18px` however you can also pass in `small` & `big` as a size. These properties will later be displayed by Docz.

Note: If your component relies on `StaticQuery` or `graphql`, consider splitting it into two smaller components:

- one React component dealing only with the **UI layer**, and
- another dealing with the **data layer**.

You could showcase the UI layer React component in your `mdx` files and your data layer component could use it to render the data it fetched thanks to `StaticQuery` and `graphql`.

Create a new file in the `docs` directory to document your newly created button component. Call the file `button.mdx` :

```
---
name: Button
menu: Components
---

# Button

Buttons make common actions more obvious and help users more easily perform them.
Buttons use labels and sometimes icons to communicate the action that will occur
when the user touches them.
```

Docz offers some internal components you can use to display the component and its properties. Import both these and your component itself into the document and use them:

```
---
name: Button
menu: Components
---

// highlight-start
import { Playground, Props } from "docz"
import { Button } from "../button"
// highlight-end

# Button

Buttons make common actions more obvious and help users more easily perform them.
Buttons use labels and sometimes icons to communicate the action that will occur
when the user touches them.

// highlight-start

## Properties

<Props of={Button} />

## Basic usage

<Playground>
  <Button>Click me</Button>
</Playground>

## With different sizes

<Playground>
  <Button scale="small">Click me</Button>
  <Button scale="normal">Click me</Button>
  <Button scale="big">Click me</Button>
```

```
</Playground>
// highlight-end
```

Start the development server again and you should see the properties (children and scale), one playground displaying the normal button, and one playground showing the button in its three sizes.

Configuration

You can usually set your config using a `doczrc.js` file ([see all available options](#)) or if you want to set some default options for your theme, you can set `options` in the theme definition.

```
module.exports = {
  plugins: [
    //highlight-start
    {
      resolve: `gatsby-theme-docz`,
      options: {
        // Your options here
      },
    },
    //highlight-end
    // Your plugins go here
  ],
}
```

We highly recommend that you set your configuration using `doczrc.js` ! Live reloading will only work with the configuration file, not the settings inside the theme definition.

Other resources

- For more information on Docz visit [the Docz site](#) and in particular the [Gatsby theme documentation](#)
- Check out the official [Docz starter](#)