Tailwind is a utility-first CSS framework for rapidly building custom user interfaces. This guide will show you how to get started with Gatsby and Tailwind CSS.

Overview

There are three ways you can use Tailwind with Gatsby:

1. Standard: Use PostCSS to generate Tailwind classes, then you can apply those classes using `className`.
2. CSS-in-JS: Integrate Tailwind classes into Styled Components.
3. SCSS: Use gatsby-plugin-sass to support Tailwind classes in your SCSS files.

You have to install and configure Tailwind for all of these methods, so this guide will walk through that step first, then you can follow the instructions for PostCSS, CSS-in-JS or SCSS.

Installing and configuring Tailwind

This guide assumes that you have a Gatsby project set up. If you need to set up a project, head to the **Quick Start guide**, then come back.

1. Install Tailwind

```
npm install tailwindcss autoprefixer
```

2. Generate Tailwind config file (optional)

Note: A config file isn't required for Tailwind 1.0.0+

To configure Tailwind, you'll need to add a Tailwind configuration file. Luckily, Tailwind has a built-in script to do this. Just run the following command:

```
npx tailwindcss init
```

Option #1: PostCSS

1. Install the Gatsby PostCSS plugin **gatsby-plugin-postcss**.

```
npm install postcss gatsby-plugin-postcss
```

2. Include the plugin in your `gatsby-config.js` file.

```
plugins: [`gatsby-plugin-postcss`],
```

3. Configure PostCSS to use Tailwind

Create a `postcss.config.js` in your project's root folder with the following contents.

```
module.exports = () => ({
  plugins: [require("tailwindcss")],
})
```

4. Use the Tailwind Directives in your CSS

You can now use the `@tailwind` directives to add Tailwind's utilities, preflight, and components into your CSS. You can also use `@apply` and all of Tailwind's other directives and functions!

To learn more about how to use Tailwind in your CSS, visit the Tailwind Documentation

Option #2: CSS-in-JS

These steps assume you have a CSS-in-JS library already installed, and the examples are based on Emotion.

1. Install the Twin Babel Macro and Emotion

```
npm install -D twin.macro @emotion/react @emotion/styled gatsby-plugin-emotion
```

2. Import the Tailwind base styles

```
import "tailwindcss/dist/base.min.css"
```

3. Enable the Gatsby emotion plugin

```
module.exports = {
  plugins: [`gatsby-plugin-emotion`],
}
```

4. Use `twin.macro` to create your styled component

```
import React from "react"
import tw, { styled } from "twin.macro" //highlight-line

const Button = styled.button`
  ${tw`bg-blue-500 hover:bg-blue-800 text-white p-2 rounded`}
`

// or use the shorthand version

const Button = tw.button`
  bg-blue-500 hover:bg-blue-800 text-white p-2 rounded
`

const IndexPage = () => (
  <div>
    <h1>Hi people</h1>
    <Button>Activate</Button> // highlight-line
  </div>
)

export default IndexPage
```

See the Twin + Gatsby + Emotion installation guide for more information.

Option #3: SCSS

1. Install the Gatsby SCSS plugin **gatsby-plugin-sass** and `sass`.

```
npm install sass gatsby-plugin-sass
```

2. To be able to use Tailwind classes in your SCSS files, add the `tailwindcss` package into the `postCssPlugins` parameter in your `gatsby-config.js`.

```
plugins: [
  {
    resolve: `gatsby-plugin-sass`,
    options: {
      postCssPlugins: [
        require("tailwindcss"),
        require("./tailwind.config.js"), // Optional: Load custom Tailwind CSS
configuration
      ],
    },
  },
],
```

Note: Optionally you can add a corresponding configuration file (by default it will be `tailwind.config.js`). If you are adding a custom configuration, you will need to load it after `tailwindcss`.

3. Add base CSS/SCSS files

Note: This approach is not needed if you chose CSS-in-JS above, as you can already nest styles and `@apply` rules directly from your `.js` files.

In case you need to create custom classes for elements for nested selectors, or for overriding external packages, you can create your own CSS/SCSS files.

1. Create a new file and import your Tailwind directives.

This will be your 'master' CSS file, which you will import all other CSS within.

```
@tailwind base;
@tailwind components;
@tailwind utilities;
```

Note: if using SCSS (or another supported language, rename files/folders appropriately).

2. Import any custom CSS files or add any custom CSS you need (optional)

```
@tailwind base;
@tailwind components;
@tailwind utilities;

@import popup.css body {
```

```
  @apply bg-purple-200;
}
```

Tailwind will swap these directives out at build-time with all of the styles it generates based on your configured design system.

3. Import this file into your `gatsby-browser.js`

In `gatsby-browser.js` add an import rule for your Tailwind directives and custom CSS to pull them into your site.

```
import "./src/css/index.css"
```

4. Configuring your content path

By default, Tailwind ensures that only the classes we need are delivered to the browser. Rather than including every combination of every class you might think of, Tailwind automatically removes unused classes. Because of this, it requires a configuration file to tell it which content to scan.

3.0.0 and above

You can mark files to process directly from your Tailwind config. You need to provide an array of strings telling it which files to process.

```
module.exports = {
  content: ["./src/**/*.{js,jsx,ts,tsx}"],
  theme: {},
  variants: {},
  plugins: [],
}
```

See Tailwind's documentation for usage with Gatsby and extended content configuration options for more information.

> ***Note:*** *It is* ***not recommended*** *that you include Gatsby's output directories (e.g.* `public` , `.cache` *) in your content array in your Tailwind config. You should only need to include your source files to have Tailwind work as expected.*

Older versions

It is recommended you install the latest version of Tailwind CSS to get all available features. If you need to use an older version, you can follow the instructions on the PurgeCSS website - Purge CSS manually in older Tailwind versions

Other resources

- Introduction to PostCSS
- Tailwind Documentation
- Gatsby starters that use Tailwind