There are so many ways to add styles to your website; Gatsby supports almost every possible option, through official and community plugins.

Using global CSS files without a Layout component

Prerequisites

- An existing Gatsby site with an index page component
- A `gatsby-browser.js` file

Directions

1. Create a global CSS file as `src/styles/global.css` and paste the following into the file:

```
html {
  background-color: lavenderblush;
}

p {
  color: maroon;
}
```

2. Import the global CSS file in the `gatsby-browser.js` file such as the following:

```
import "./src/styles/global.css"
```

> ***Note:*** *You can also make use of* `require('./src/styles/global.css')` *to import the global CSS file in your* `gatsby-browser.js` *file.*

3. Run `gatsby-develop` to observe the global styling being applied across your site.

> ***Note:*** *This approach is not the best fit if you are using CSS-in-JS for styling your site, in which case a layout page with all the shared components should be used. This is covered in the next recipe.*

Additional resources

- More on adding global styles without a layout component

Using global styles in a layout component

Prerequisites

- A Gatsby site with an index page component

Directions

You can add global styles to a shared layout component. This component is used for things that are common throughout the site, like a header or footer.

1. If you don't already have one, create a new directory in your site at `/src/components`.
2. Inside the components directory, create two files: `layout.css` and `layout.js`.
3. Add the following to `layout.css`:

```
body {
  background: red;
}
```

4. Edit `layout.js` to import the CSS file and output layout markup:

```
import React from "react"
import "./layout.css"

export default function Layout({ children }) {
  return <div>{children}</div>
}
```

5. Now edit your site's homepage at `/src/pages/index.js` and use the new layout component:

```
import React from "react"
import Layout from "../components/layout"

export default function Home() {
  return <Layout>Hello world!</Layout>
}
```

Additional resources

- Standard Styling with Global CSS Files
- More about layout components

Using Styled Components

Prerequisites

- A Gatsby site with an index page component
- gatsby-plugin-styled-components, styled-components, and babel-plugin-styled-components installed in `package.json`

Directions

1. Inside your `gatsby-config.js` file add `gatsby-plugin-styled-components`

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

2. Open the index page component (`src/pages/index.js`) and import the `styled-components` package
3. Style components by creating style blocks for each element type
4. Apply to the page by including styled components in the JSX

```
import React from "react"
import styled from "styled-components" //highlight-line

const Container = styled.div`
  margin: 3rem auto;
  max-width: 600px;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
`

const Avatar = styled.img`
  flex: 0 0 96px;
  width: 96px;
  height: 96px;
  margin: 0;
`

const Username = styled.h2`
  margin: 0 0 12px 0;
  padding: 0;
`

const User = props => (
  <>
    <Avatar src={props.avatar} alt={props.username} />
    <Username>{props.username}</Username>
  </>
)

export default function UsersList() {
  return (
    <Container>
      <h1>About Styled Components</h1>
      <p>Styled Components is cool</p>
      <User
        username="Jane Doe"

avatar="https://s3.amazonaws.com/uifaces/faces/twitter/adellecharles/128.jpg"
      />
      <User
        username="Bob Smith"
        avatar="https://s3.amazonaws.com/uifaces/faces/twitter/vladarbatov/128.jpg"
      />
    </Container>
  )
}
```

4. Run `gatsby develop` to see the changes

Additional resources

- More on Using Styled Components
- Egghead lesson

Using CSS Modules

Prerequisites

- An existing Gatsby site with an index page component

Directions

1. Create a CSS module as `src/pages/index.module.css` and paste the following into the module:

```
.feature {
  margin: 2rem auto;
  max-width: 500px;
}
```

2. Import the CSS module as a JSX object `style` in the `index.js` file by modifying the page so it looks like the following:

```
import React from "react"

// highlight-start
import * as style from "./index.module.css"

export default function Home() {
  return (
    <section className={style.feature}>
      <h1>Using CSS Modules</h1>
    </section>
  )
}
// highlight-end
```

3. Run `gatsby develop` to see the changes.

Note: Notice that the file extension is `.module.css` instead of `.css`, which tells Gatsby that this is a CSS module.

Additional resources

- More on Using CSS Modules
- Live example on Using CSS modules

Using Sass/SCSS

Sass is an extension of CSS that gives you more advanced features like nested rules, variables, mixins, and more.

Sass has 2 syntaxes. The most commonly used syntax is "SCSS", and is a superset of CSS. That means all valid CSS syntax, is valid SCSS syntax. SCSS files use the extension `.scss`

Sass will compile `.scss` and `.sass` files to `.css` files for you, so you can write your stylesheets with more advanced features.

Prerequisites

- A Gatsby site.

Directions

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

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

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

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

3. Write your stylesheets as `.sass` or `.scss` files and import them. If you don't know how to import styles, take a look at Styling with CSS

Note: You can use Sass/SCSS files as modules too, like mentioned in the previous recipe about CSS modules, with the difference that instead of `.css` the extensions have to be `.scss` or `.sass`

Using `.scss`:

```
$font-stack: Helvetica, sans-serif;
$primary-color: #333;

body {
  font: 100% $font-stack;
  color: $primary-color;
}
```

```
import "./styles.scss"
```

Using `.sass`:

```
$font-stack:    Helvetica, sans-serif
$primary-color: #333

body
  font: 100% $font-stack
  color: $primary-color
```

```
import "./styles.sass"
```

Additional resources

- Difference between `.sass` and `.scss`
- Sass guide from the official Sass website
- A more complete installation tutorial on Sass with some more explanations and more resources

Adding a Local Font

Prerequisites

- A Gatsby site
- A font file: `.woff2`, `.ttf`, etc.

Directions

1. Copy a font file into your Gatsby project, such as `src/fonts/fontname.woff2`.
2. Import the font asset into a CSS file to bundle it into your Gatsby site:

```
@font-face {
  font-family: "Font Name";
  src: url("../fonts/fontname.woff2");
}
```

Note: Make sure the font name is referenced from the relevant CSS, e.g.:

```
body {
  font-family: "Font Name", sans-serif;
}
```

By targeting the HTML `body` element, your font will apply to most text on the page. Additional CSS can target other elements, such as `button` or `textarea`.

If fonts are not updating following steps above, make sure to replace the existing font-family in relevant CSS.

Additional resources

- More on importing assets into files

Using Emotion

Emotion is a powerful CSS-in-JS library that supports both inline CSS styles and styled components. You can use each styling feature individually or together in the same file.

Prerequisites

- A Gatsby site

Directions

1. Install the Gatsby Emotion plugin and Emotion packages.

```
npm install gatsby-plugin-emotion @emotion/react @emotion/styled
```

2. Add the `gatsby-plugin-emotion` plugin to your `gatsby-config.js` file:

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

3. If you don't already have one, create a page in your Gatsby site at `src/pages/emotion-sample.js`.

Import Emotion's `css` core package. You can then use the `css` prop to add Emotion object styles to any element inside a component:

```
import React from "react"
import { css } from "@emotion/react"

export default function EmotionSample() {
  return (
    <div>
      <p
        css={{
          background: "pink",
          color: "blue",
        }}
      >
        This page is using Emotion.
      </p>
    </div>
  )
}
```

4. To use Emotion's styled components, import the package and define them using the `styled` function.

```
import React from "react"
import styled from "@emotion/styled"

const Content = styled.div`
  text-align: center;
  margin-top: 10px;
  p {
    font-weight: bold;
  }
`

export default function EmotionSample() {
  return (
    <Content>
      <p>This page is using Emotion.</p>
    </Content>
  )
}
```

Additional resources

- Using Emotion in Gatsby
- Emotion website
- Getting started with Emotion and Gatsby

Using Google Fonts

Hosting your own Google Fonts locally within a project means they won't have to be fetched over the network when your site loads, increasing your site's speed index by up to ~300 milliseconds on desktop and 1+ seconds on 3G. It's also recommended to limit custom font usage to only the essential for performance.

Prerequisites

- A Gatsby site
- The Gatsby CLI installed
- A chosen font package from Fontsource

Directions

This example shows how to set up the Open Sans font. If you have a different Google Font you want to use, you can find the corresponding package in NPM or the packages directory in the Fontsource repository.

1. Run `npm install @fontsource/open-sans` to download the necessary package files.
2. Then within your app entry file or site component, import the font package. It is recommended you import it via the layout template (`layout.js`). However, importing via page component (`index.js`), or `gatsby-browser.js` are viable alternatives.

```
import "@fontsource/open-sans" // Defaults to weight 400.
```

If you wish to select a particular weight or style, you may specify it by changing the import path.

```
import "@fontsource/open-sans/500.css" // Weight 500.
import "@fontsource/open-sans/900-italic.css" // Loads the italic variant.
```

Note: The range of supported weights and styles a font may support is shown in each package's README file.

3. Once it's imported, you can reference the font name in a CSS stylesheet, CSS Module, or CSS-in-JS.

```
body {
  font-family: "Open Sans";
}
```

Additional resources

- Fontsource repo on GitHub
- Typography.js - Another option for using Google fonts on a Gatsby site

Using Font Awesome

Using Font Awesome gives you access to thousands of icons for use on your site. Since Gatsby sites are React sites, it's recommended to use the react-fontawesome SVG library.

Prerequisites

- The Gatsby CLI installed
- A Gatsby site

Directions

1. Install the `react-fontawesome` dependencies.

```
npm install @fortawesome/fontawesome-svg-core  @fortawesome/free-brands-svg-icons
@fortawesome/react-fontawesome
```

> *Note that there are multiple icon libraries within `react-fontawesome`. You may also be interested in `free-regular-svg-icons` and `free-solid-svg-icons` which you would install the same way.*

2. Import the `FontAwesomeIcon` component and the icon you want to use. Then use the icon as a component directly in your JSX files:

```
import { FontAwesomeIcon } from "@fortawesome/react-fontawesome"
import { faReact } from "@fortawesome/free-brands-svg-icons"

const IndexPage = () => (
  <Layout>
    <FontAwesomeIcon icon={faReact} /> //highlight-line
  </Layout>
)

export default IndexPage
```

> *This example imports a single, specific icon and uses it for improved performance. As an alternative, you can import the icons and build a library.*

Additional resources

- Font Awesome
- react-fontawesome