Adding Markdown Pages

Gatsby can use Markdown files to create pages in your site. You add plugins to read and understand folders with Markdown files and from them create pages automatically.

Here are the steps Gatsby follows for making this happen.

1. Read files into Gatsby from the filesystem
2. Transform Markdown to HTML and frontmatter to data
3. Add a Markdown file
4. Create a Collection Route component for the Markdown files

Read files into Gatsby from the filesystem

Use the plugin `gatsby-source-filesystem` to read files.

Install

```
npm install gatsby-source-filesystem
```

Add plugin

Open `gatsby-config.js` to add the `gatsby-source-filesystem` plugin. The `path` option is how you set the directory to search for files.

```js
module.exports = {
  siteMetadata: {
    title: "My Gatsby Site",
  },
  plugins: [
    {
      resolve: `gatsby-source-filesystem`,
      options: {
        name: `markdown-pages`,
        path: `${__dirname}/src/markdown-pages`,
      },
    },
  ],
}
```

Completing the above step means that you've "sourced" the Markdown files from the filesystem. You can now "transform" the Markdown to HTML and the YAML frontmatter to JSON.

Transform Markdown to HTML and frontmatter to data using `gatsby-transformer-remark`

You'll use the plugin `gatsby-transformer-remark` to recognize files which are Markdown and read their content. The plugin will convert the frontmatter metadata part of your Markdown files as `frontmatter` and the content part as HTML.

Install transformer plugin

```
npm install gatsby-transformer-remark
```

Configure plugin

Add this to `gatsby-config.js` after the previously added `gatsby-source-filesystem`.

```js
module.exports = {
  siteMetadata: {
    title: "My Gatsby Site",
  },
  plugins: [
    {
      resolve: `gatsby-source-filesystem`,
      options: {
        name: `markdown-pages`,
        path: `${__dirname}/src/markdown-pages`,
      },
    },
    // highlight-next-line
    `gatsby-transformer-remark`,
  ],
}
```

Add a Markdown file

Create a folder in the `/src` directory of your Gatsby application called `markdown-pages`. Now create a Markdown file inside it with the name `post-1.md`.

Frontmatter for metadata in Markdown files

When you create a Markdown file, you can include a set of key/value pairs that can be used to provide additional data relevant to specific pages in the

GraphQL data layer. This data is called "frontmatter" and is denoted by the triple dashes at the start and end of the block. This block will be parsed by `gatsby-transformer-remark` as YAML. You can then query the data through the GraphQL API from your React components.

```
---
slug: "/blog/my-first-post"
date: "2019-05-04"
title: "My first blog post"
---
```

What is important in this step is the key pair `slug`. The value that is assigned to the key `slug` is used in order to navigate to your post.

Create a Collection Route for the Markdown files

Create `src/pages/{MarkdownRemark.frontmatter__slug}.js` and add the following code:

```jsx
import React from "react"
import { graphql } from "gatsby"

export default function Template({
  data, // this prop will be injected by the GraphQL query below.
}) {
  const { markdownRemark } = data // data.markdownRemark holds your post data
  const { frontmatter, html } = markdownRemark
  return (
    <div className="blog-post-container">
      <div className="blog-post">
        <h1>{frontmatter.title}</h1>
        <h2>{frontmatter.date}</h2>
        <div
          className="blog-post-content"
          dangerouslySetInnerHTML={{ __html: html }}
        />
      </div>
    </div>
  )
}

export const pageQuery = graphql`
  query($id: String!) {
    markdownRemark(id: { eq: $id }) {
      html
      frontmatter {
        date(formatString: "MMMM DD, YYYY")
```

```
        slug
        title
      }
    }
  }
}
`
```

Two things are important in the file above:

1. A GraphQL query is made in the second half of the file to get the Markdown data. Gatsby has automagically given you all the Markdown metadata and HTML in this query's result.

   Note: To learn more about GraphQL, consider this excellent resource

2. The result of the query is injected by Gatsby into the component as the `data` prop. `props.data.markdownRemark` is the property that has all the details of the Markdown file.

Next you could create a page component at `src/pages/blog/index.js` to serve as a listing page for all your blog posts.

This should get you started on some basic Markdown functionality in your Gatsby site. You can further customize the frontmatter and the component file to get desired effects!

For more information, have a look in the working example `using-markdown-pages`. You can find it in the Gatsby examples section.

Other tutorials

Check out tutorials listed on the Awesome Gatsby page for more information on building Gatsby sites with Markdown.

Gatsby Markdown starters

There are also a number of Gatsby starters that come pre-configured to work with Markdown.