gatsby-plugin-sitemap

Create a sitemap for your Gatsby site.

NOTE: This plugin only generates output when run in `production` mode! To test your sitemap, run: `gatsby build && gatsby serve`

Install

`npm install gatsby-plugin-sitemap`

How to Use

```
// In your gatsby-config.js
siteMetadata: {
  // If you didn't use the resolveSiteUrl option this needs to be set
  siteUrl: `https://www.example.com`,
},
plugins: [`gatsby-plugin-sitemap`]
```

Above is the minimal configuration required to have it work. By default, the generated sitemap will include all of your site's pages, except the ones you exclude.

You then can point your service (e.g. Google Search Console) at `https://www.example.com/sitemap/sitemap-index.xml`.

Recommended usage

You probably do not want to use the defaults in this plugin. Here's an example of the default output:

```
<?xml version="1.0" encoding="UTF-8"?>
<urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">
  <url>
    <loc>https://example.net/blog/</loc>
    <changefreq>daily</changefreq>
    <priority>0.7</priority>
  </url>
  <url>
    <loc>https://example.net/</loc>
    <changefreq>daily</changefreq>
    <priority>0.7</priority>
  </url>
</urlset>
```

See the `changefreq` and `priority` fields? Those will be the same for every page, no matter how important or how often it gets updated. They will most likely be wrong. But wait, there's more, in their docs Google says:

> - *Google ignores `<priority>` and `<changefreq>` values, so don't bother adding them.*
> - *Google reads the `<lastmod>` value, but if you misrepresent this value, we will stop reading it.*

You really want to customize this plugin config to include an accurate `lastmod` date. Checkout the example for an example of how to do this.

Options

The `default config` can be overridden.

The options are as follows:

- `output` (string = `/sitemap`) Folder path where sitemaps are stored.
- `createLinkInHead` (boolean = true) Whether to populate the `<head>` of your site with a link to the sitemap.
- `entryLimit` (number = 45000) Number of entries per sitemap file. A sitemap index (as `sitemap-index.xml`) will always be created and multiple sitemaps are created for every `entryLimit` increment (e.g under 45000 entries only `sitemap-0.xml` will be created).
- `excludes` (string[] = []) An array of paths to exclude from the sitemap. While this is usually an array of strings it is possible to enter other data types into this array for custom filtering. Doing so will require customization of the `filterPages` function.
- `query` (GraphQL Query) The query for the data you need to generate the sitemap. It's required to get the site's URL, if you are not fetching it from `site.siteMetadata.siteUrl`, you will need to set a custom `resolveSiteUrl` function. If you override the query, you may need to pass in a custom `resolvePagePath`, `resolvePages` to keep everything working. If you fetch pages without using `allSitePage.nodes` query structure you will definitely need to customize the `resolvePages` function.
- `resolveSiteUrl` (function) Takes the output of the data query and lets you return the site URL. Sync or async functions allowed.
- `resolvePagePath` (function) Takes a page object and returns the uri of the page (no domain or protocol).
- `resolvePages` (function) Takes the output of the data query and expects an array of page objects to be returned. Sync or async functions allowed.
- `filterPages` (function) Takes the current page and a string (or other object) from the `exclude` array and expects a boolean to be returned. `true` excludes the path, `false` keeps it.
- `serialize` (function) Takes the output of `filterPages` and lets you return a sitemap entry. Sync or async functions allowed.

The following pages are **always** excluded: `/dev-404-page`, `/404` & `/offline-plugin-app-shell-fallback`, this cannot be changed even by customizing the `filterPages` function.

Example:

```
const siteUrl = process.env.URL || `https://fallback.net`

// In your gatsby-config.js
module.exports = {
  plugins: [
    {
      resolve: "gatsby-plugin-sitemap",
      options: {
        query: `
```

```
        {
          allSitePage {
            nodes {
              path
            }
          }
          allWpContentNode(filter: {nodeType: {in: ["Post", "Page"]}}) {
            nodes {
              ... on WpPost {
                uri
                modifiedGmt
              }
              ... on WpPage {
                uri
                modifiedGmt
              }
            }
          }
        }
      `,
        resolveSiteUrl: () => siteUrl,
        resolvePages: ({
          allSitePage: { nodes: allPages },
          allWpContentNode: { nodes: allWpNodes },
        }) => {
          const wpNodeMap = allWpNodes.reduce((acc, node) => {
            const { uri } = node
            acc[uri] = node

            return acc
          }, {})

          return allPages.map(page => {
            return { ...page, ...wpNodeMap[page.path] }
          })
        },
        serialize: ({ path, modifiedGmt }) => {
          return {
            url: path,
            lastmod: modifiedGmt,
          }
        },
      },
    },
  ],
}
```

API Reference

resolveSiteUrl ⇒ `string`

Sync or async functions allowed.

Returns: `string` - - site URL, this can come from the graphql query or another scope.

Param	Type	Description
data	`object`	Results of the GraphQL query

resolvePagePath ⇒ `string`

If you don't want to place the URI in `path` then `resolvePagePath` is needed.

Returns: `string` - - uri of the page without domain or protocol

Param	Type	Description
page	`object`	Array Item returned from resolvePages

resolvePages ⇒ `Array`

This allows custom resolution of the array of pages. This also where users could merge multiple sources into a single array if needed. Sync or async functions allowed.

Returns: `object[]` - - Array of objects representing each page

Param	Type	Description
data	`object`	results of the GraphQL query

filterPages ⇒ `boolean`

This allows filtering any data in any way.

This function is executed via:

```
allPages.filter(
  page => !excludes.some(excludedRoute => thisFunc(page, excludedRoute, tools))
)
```

`allPages` is the results of the `resolvePages` function.

Returns: `Boolean` - - `true` excludes the path, `false` keeps it.

Param	Type	Description
page	`object`	
excludedRoute	`string`	Element from `exclude` Array in plugin config.

tools	`object`	contains tools for filtering `{ minimatch, withoutTrailingSlash, resolvePagePath }`

serialize ⇒ `object`

This function is executed by:

```
allPages.map(page => thisFunc(page, tools))
```

`allpages` is the result of the `filterPages` function. Sync or async functions allowed.

Kind: global variable

Param	Type	Description
page	`object`	A single element from the results of the `resolvePages` function
tools	`object`	contains tools for serializing `{ resolvePagePath }`