gatsby-source-drupal

Source plugin for pulling data (including images) into Gatsby from Drupal sites.

It pulls data from Drupal 8 sites with the Drupal JSONAPI module installed.

An example site built with the headless Drupal distro ContentaCMS is at https://using-drupal.gatsbyjs.org/

`apiBase` Option allows changing the API entry point depending on the version of jsonapi used by your Drupal instance. The default value is `jsonapi`, which has been used since jsonapi version `8.x-1.0-alpha4`.

Install

`npm install gatsby-source-drupal`

How to use

```
// In your gatsby-config.js
module.exports = {
  plugins: [
    {
      resolve: `gatsby-source-drupal`,
      options: {
        baseUrl: `https://live-contentacms.pantheonsite.io/`,
        apiBase: `api`, // optional, defaults to `jsonapi`
      },
    },
  ],
}
```

On the Drupal side, we highly recommend installing JSON:API Extras and enabling "Include count in collection queries" `/admin/config/services/jsonapi/extras` as that speeds up fetching data from Drupal by around 4x.

Filters

You can use the `filters` option to limit the data that is retrieved from Drupal. Filters are applied per JSON API collection. You can use any valid JSON API filter query. For large data sets this can reduce the build time of your application by allowing Gatsby to skip content you'll never use.

As an example, if your JSON API endpoint (https://live-contentacms.pantheonsite.io/api) returns the following collections list, then `articles` and `recipes` are both collections that can have a filters applied:

```
{
  ...
  links: {
    articles: "https://live-contentacms.pantheonsite.io/api/articles",
    recipes: "https://live-contentacms.pantheonsite.io/api/recipes",
    ...
```

```
  }
}
```

To retrieve only recipes with a specific tag you could do something like the following where the key (recipe) is the collection from above, and the value is the filter you want to apply.

```
// In your gatsby-config.js
module.exports = {
  plugins: [
    {
      resolve: `gatsby-source-drupal`,
      options: {
        baseUrl: `https://live-contentacms.pantheonsite.io/`,
        apiBase: `api`,
        filters: {
          // collection : filter
          recipe: "filter[tags.name][value]=British",
        },
      },
    },
  ],
}
```

Which would result in Gatsby using the filtered collection https://live-contentacms.pantheonsite.io/api/recipes?filter[tags.name][value]=British to retrieve data.

Basic Auth

You can use `basicAuth` option if your site is protected by basicauth. First, you need a way to pass environment variables to the build process, so secrets and other secured data aren't committed to source control. We recommend using `dotenv` which will then expose environment variables. Read more about dotenv and using environment variables here. Then we can *use* these environment variables and configure our plugin.

```
// In your gatsby-config.js
module.exports = {
  plugins: [
    {
      resolve: `gatsby-source-drupal`,
      options: {
        baseUrl: `https://live-contentacms.pantheonsite.io/`,
        apiBase: `api`, // optional, defaults to `jsonapi`
        basicAuth: {
          username: process.env.BASIC_AUTH_USERNAME,
          password: process.env.BASIC_AUTH_PASSWORD,
        },
      },
    },
  ],
}
```

Fastbuilds

You can use the `fastBuilds` option to enable fastbuilds. This requires the Gatsby Drupal module (called gatsby_fastbuilds) to be enabled. This will speed up your development and build process by only downloading content that has changed since you last ran `gatsby build` or `gatsby develop`.

This will require authentication to your Drupal site and a Drupal user with the Drupal permission to `sync gatsby fastbuild log entities`.

```
// In your gatsby-config.js
module.exports = {
  plugins: [
    {
      resolve: `gatsby-source-drupal`,
      options: {
        baseUrl: `https://live-contentacms.pantheonsite.io/`,
        apiBase: `api`, // optional, defaults to `jsonapi`
        basicAuth: {
          username: process.env.BASIC_AUTH_USERNAME,
          password: process.env.BASIC_AUTH_PASSWORD,
        },
        fastBuilds: true,
      },
    },
  ],
}
```

Request Headers

You can add optional request headers to the request using `headers` param.

```
// In your gatsby-config.js
module.exports = {
  plugins: [
    {
      resolve: `gatsby-source-drupal`,
      options: {
        baseUrl: `https://live-contentacms.pantheonsite.io/`,
        apiBase: `api`, // optional, defaults to `jsonapi`
        headers: {
          Host: "https://example.com", // any valid request header here
        },
      },
    },
  ],
}
```

GET Search Params

You can append optional GET request params to the request url using `params` option.

```
// In your gatsby-config.js
module.exports = {
  plugins: [
    {
      resolve: `gatsby-source-drupal`,
      options: {
        baseUrl: `https://live-contentacms.pantheonsite.io/`,
        apiBase: `api`, // optional, defaults to `jsonapi`
        params: {
          "api-key": "your-api-key-header-here", // any valid key value pair here
        },
      },
    },
  ],
}
```

File Downloads

You can use the `skipFileDownloads` option if you do not want Gatsby to download files from your Drupal website. This is useful if you are using another option for processing/serving images.

```
// In your gatsby-config.js
module.exports = {
  plugins: [
    {
      resolve: `gatsby-source-drupal`,
      options: {
        baseUrl: `https://live-contentacms.pantheonsite.io/`,
        apiBase: `api`, // optional, defaults to `jsonapi`
        skipFileDownloads: true,
      },
    },
  ],
}
```

You can also filter out temporary files. This will help to avoid Gatsby throwing an error when a 404 is returned from a file that does not exist:

```
// In your gatsby-config.js
module.exports = {
  plugins: [
    {
      resolve: `gatsby-source-drupal`,
      options: {
        baseUrl: `https://live-contentacms.pantheonsite.io/`,
        apiBase: `api`,
        filters: {
          // collection : filter
          "file--file": "filter[status][value]=1",
        },
```

```
      },
    },
  ],
}
```

Concurrent File Requests

You can use the `concurrentFileRequests` option to change how many simultaneous file requests are made to the server/service. This benefits build speed, however too many concurrent file request could cause memory exhaustion depending on the server's memory size so change with caution.

```
// In your gatsby-config.js
module.exports = {
  plugins: [
    {
      resolve: `gatsby-source-drupal`,
      options: {
        baseUrl: `https://live-contentacms.pantheonsite.io/`,
        apiBase: `api`, // optional, defaults to `jsonapi`
        concurrentFileRequests: 60, // optional, defaults to `20`
      },
    },
  ],
}
```

Concurrent API Requests

You can use the `concurrentAPIRequests` option to change how many simultaneous API requests are made to the server/service. 20 is the default and seems to be the fastest for most sites.

Disallowed Link Types

You can use the `disallowedLinkTypes` option to skip link types found in JSON:API documents. By default it skips the `self`, `describedby`, `contact_message--feedback`, and `contact_message--pesonal` links, which do not provide data that can be sourced. You may override the setting to add additional link types to be skipped.

```
// In your gatsby-config.js
module.exports = {
  plugins: [
    {
      resolve: `gatsby-source-drupal`,
      options: {
        baseUrl: `https://live-contentacms.pantheonsite.io/`,
        // skip the action--action resource type.
        disallowedLinkTypes: [
          `self`,
          `describedby`,
          `contact_message--feedback`,
```

```
          `contact_message--personal`,
        ],
      },
    },
  ],
}
```

NOTES:

When using includes in your JSON:API calls the included data will automatically become available to query, even if the link types are skipped using `disallowedLinkTypes`.

This enables you to fetch only the data you need at build time, instead of all data of a certain entity type or bundle.

```
// In your gatsby-config.js
module.exports = {
  plugins: [
    {
      resolve: `gatsby-source-drupal`,
      options: {
        baseUrl: `https://live-contentacms.pantheonsite.io/`,
        // Skip the node--page resource type and paragraph components.
        disallowedLinkTypes: [
          `self`,
          `describedby`,
          `node--page`,
          `paragraph--text`,
          `paragraph--image`,
        ],
        filters: {
          // Use includes so only the news content paragraph components are fetched.
          "node--news": "include=field_content",
        },
      },
    },
  ],
}
```

Entity Reference revisions and relationships

By default `gatsby-source-drupal` resolves Entity Reference relationships using just ID. If you are using the contrib module Entity reference revisions and Paragraphs, you may have advanced use-cases such as fetching drafts where you want to resolve these relationships using both ID and revision ID. You can nominate entity-type IDs where you wish to resolve relationships using the revision ID by adding them to the `entityReferenceRevisions` configuration option. Please note that `gatsby-source-drupal` only ever fetches the default (published) revision, so this functionality is only needed in advanced cases where you have custom code Drupal side that is applying additional logic.

```
// In your gatsby-config.js
module.exports = {
```

```
  plugins: [
    {
      resolve: `gatsby-source-drupal`,
      options: {
        baseUrl: `https://live-contentacms.pantheonsite.io/`,
        apiBase: `api`,
        entityReferenceRevisions: ["paragraph"], // optional, defaults to `[]`
      },
    },
  ],
}
```

Translations

If you have translations or multilingual enabled on your Drupal site, you can opt-in to sourcing translations of entities. To do this, enable in your plugin's configuration the languages and entity types you'd like to source. E.g.

```
// In your gatsby-config.js
module.exports = {
  plugins: [
    {
      resolve: `gatsby-source-drupal`,
      options: {
        baseUrl: `https://live-contentacms.pantheonsite.io/`,
        languageConfig: {
          defaultLanguage: `en`,
          enabledLanguages: [`en`, `fil`],
          translatableEntities: [`node--article`],
          nonTranslatableEntities: [`file--file`],
        },
      },
    },
  ],
}
```

Some entities are not translatable like Drupal files and will return null result when language code from parent entity doesn't match up. These items can be specified as nonTranslatableEntities and receive the defaultLanguage as fallback.

Gatsby Preview (experimental)

You will need to have the Drupal module installed, more information on that here: https://www.drupal.org/project/gatsby

In your Drupal module configuration, set the update URL to your Gatsby Preview instance URL.

NOTES:

- This is experimental feature in active development. APIs used for this feature are not yet stable - it can break while we iterate on API design (particularly when versions of `gatsby-source-drupal` and `Gatsby Live Preview` Drupal module are incompatible).

Preview Secret

While you don't need to pass any additional options for preview to work, you can pass a `secret` for added security between your Drupal instance and Gatsby preview. Ensure this secret matches the one set in your Drupal Gatsby Preview settings.

```
// In your gatsby-config.js
module.exports = {
  plugins: [
    {
      resolve: `gatsby-source-drupal`,
      options: {
        baseUrl: `https://live-contentacms.pantheonsite.io/`,
        secret: process.env.PREVIEW_SECRET, // optional, must match Drupal instance
preview secret
      },
    },
  ],
}
```

How to query

You can query nodes created from Drupal like the following:

```
{
  allArticle {
    edges {
      node {
        title
        internalId
        created(formatString: "DD-MMM-YYYY")
      }
    }
  }
}
```