Welcome to `gatsby@4.8.0` release (February 2022 #2)

Key highlights of this release:

- Support for TypeScript in `gatsby-browser` and `gatsby-ssr`
- New TypeScript option when creating Gatsby projects from the CLI
- Significant memory usage reduction when filtering and sorting nodes
- New APIs in `gatsby-core-utils` and `gatsby-plugin-utils`

Also check out notable bugfixes.

Bleeding Edge: Want to try new features as soon as possible? Install `gatsby@next` and let us know if you have any issues.

Previous release notes

Full changelog

Support for TypeScript in `gatsby-browser` and `gatsby-ssr`

In addition to Gatsby's current TypeScript support for your site's `.ts` / `.tsx` files, you can now use TypeScript with your `gatsby-browser` and `gatsby-ssr` files.

The `GatsbyBrowser` and `GatsbySSR` types can now be used in `gatsby-browser.tsx` and `gatsby-ssr.tsx` respectively.

```
import * as React from "react"
import { GatsbyBrowser } from "gatsby"

export const wrapPageElement: GatsbyBrowser["wrapPageElement"] = ({ element }) => {
  return (
    <div>
      <h1>Hello World</h1>
      {element}
    </div>
  )
}
```

```
import * as React from "react"
import { GatsbySSR } from "gatsby"

export const wrapPageElement: GatsbySSR["wrapPageElement"] = ({ element }) => {
  return (
    <div>
      <h1>Hello World</h1>
      {element}
    </div>
  )
}
```

Work is in progress for TypeScript support for `gatsby-config` and `gatsby-node` files. Check out the RFC to learn how to test it out today.

New TypeScript option when creating Gatsby projects from the CLI

When initializing new Gatsby projects with `gatsby new`, `npm init gatsby`, or `npx create-gatsby`, you can now select JavaScript or TypeScript as an option to start your project with.

After calling one of the above commands with no flags, the third question will now be: "Will you be using JavaScript or TypeScript?". Selecting JavaScript will start your project with `gatsby-starter-minimal`, and selecting TypeScript will start your project with `gatsby-starter-minimal-ts`.

A new `-ts` flag has been added for use with `npm init gatsby` and `npx create-gatsby`. Executing `npm init gatsby -ts` or `npx create-gatsby -ts` will skip the language question and start your project with `gatsby-starter-minimal-ts`.

Lastly, arguments for Gatsby initializer commands are now *not positional*. Any order of your desired site name and flags will work as expected (e.g. `npm init gatsby -ts hello-world -y`).

Significant memory usage reduction when filtering and sorting nodes

Gatsby no longer stores whole nodes in memory when executing filtering and sorting processes on your site. Now, Gatsby uses weak references and node partials, resulting in significant memory usage reduction for sites with many large nodes.

See PR #34747 for more information.

New APIs in `gatsby-core-utils` and `gatsby-plugin-utils`

Two new APIs have been added:

- `createMutex` in `gatsby-core-utils`, via PR #34761
- `hasFeature` in `gatsby-plugin-utils`, via PR #34748

Calling `createMutex` gives you a mutex that you can use to safely perform processes concurrently in places where that matters, such as in workers.

For example in the code below, one worker can execute while all others wait for it to finish. This is handy in scenarios like writing to the same file to disk.

```
const { createMutex } = require("gatsby-core-utils/mutex")

const mutex = createMutex("my-custom-mutex-key")
await mutex.acquire()

await fs.writeFile("pathToFile", "my custom content")

await mutex.release()
```

Calling `hasFeature` allows you to check if the current version of Gatsby has a certain feature. This is particularly useful for plugin authors.

```
const { hasFeature } = require(`gatsby-plugin-utils`)

if (!hasFeature(`image-service`)) {
  // You can do things like polyfill image-service here so older versions have
support as well
}
```

Note - The list of features available will be added in future PRs, the above is an example of how it will be used.

Notable bugfixes & improvements

- `gatsby-plugin-gatsby-cloud` : Improved UX for preview success and error notifications, via PR #34725
- `gatsby` :
  - Removal of `v8-compile-cache` for better ESM support, via PR #34672
  - Support use of `node.slug` to match node manifests to pages for Gatsby Preview, via PR #34790
  - Fix Content Sync when using DSG, via PR #34799
  - Allow referencing derived types in schema customization, via PR #34787
  - Refactor load plugin modules, via PR #34813
  - Upgrade from lmdb-store to lmdb, via PR #34576
- `gatsby-source-wordpress` : Fix Safari image loading bug, via PR #34727
- `gatsby-core-utils` : Improvements to `fetch-remote-file` , via PR #34758

Contributors

A big **Thank You** to our community who contributed to this release 💜

- kyleslie2: Typo fix PR #34749
- angeloashmore: chore(docs): Add Prismic to `using-gatsby-plugin-image` PR #34751
- paulduszak: chore(docs): Update "building a theme" tutorial PR #34732
- jhcao23: docs: update typo Forestry PR #34805
- VividhPandey003: documentation: Add Third Party Schema PR #34820
- axe312ger
  - refactor(gatsby-source-contentful): remove unnecessary check for existing node PR #34829
  - fix(gatsby-source-contentful): avoid confusion of Gatsby node and Contentful node count in logs PR #34830