One of the challenges as your Gatsby site grows is that site build times can increase. Longer build times can be costly to your team in several ways:

- Longer build times slow your CI/CD pipelines, increasing the amount of time to verify, review, and share changes.
- Longer build times can cause friction for business stakeholders who want to publish content quickly

Improving build times

Improving build performance is an iterative process: locate key problem areas, identify potential changes, test the impact of those changes on build speed, and ship changes that help.

Some improvements require a larger architectural changes that require more in-depth work to ship. For these, it's worth doing a quick prototype to verify that the change will have its desired effect before embarking on a multi-day or multi-week project.

While the best options will depend on the kind of site you have, some places to look are below

1. General best practices

Upgrade to the latest versions of gatsby and source plugins

The Gatsby team is constantly updating plugins to use less memory and run faster. If you're not on the latest versions, consider upgrading. This may not do anything -- but it may make a large impact.

Audit plugin configuration and queries and remove unused ones

As your site's codebase evolves, you might accumulate plugins that are no longer needed. Try looking through your `gatsby-config.js` to make sure you're using all the plugins you have installed. In addition, you may want to look through queries to make sure you're using them (and the fields in each query).

Query only needed fields in `createPages`

Creating pages should take at most around 1-2 seconds / 10k pages. Some sites take however 30s to many minutes to create pages. This almost always happens because the graphql query in `createPages` includes many fields that aren't needed to create the pages. Most sites only need to query for the node id in `createPages`. All other fields needed for the page should be queried in the page component's query.

Check how long `createPages` takes for your build. If it's longer than 10s, check if there's fields you can remove from the query.

Query only needed fields in page queries

The more fields you query the longer each query will take to run. Gatsby's GraphQL schema gives you a rich array of filtering/sorting capabilities so you grab just the data you need. If you install gatsby-plugin-perf-budgets, it will show you the amount of data you query for each page so you can audit larger ones.

Make sure you're not clearing the cache between builds

In the past, Gatsby's cache was less reliable than it is now. As a result, some sites started clearing the cache between builds. With improvements we've made, that should not be necessary anymore, so if your build script says something like "gatsby clean && gatsby build" you may want to change it to just run gatsby build.

Run builds on a machine with more and higher-powered CPUs

The better the underlying hardware / VM / container you have, the faster your builds will go. As an anecdote, a Gatsby core team developer ran the same build on two machines they owned -- a five-year-old low-powered

Windows machine and a new MacBook Pro -- and found that the latter gave a 30x build speed improvement.

Use Gatsby Cloud

Gatsby Cloud offers image parallelization during image processing and other build optimizations that can speed things up significantly. Try running your site on Gatsby Cloud and comparing the speeds there (both warm and cold builds) to your current CI/CD setup.

2. Reducing build times from content sourcing and processing

Avoid pulling unnecessary locales or content models into your site.

Sometimes your headless CMS is storing content types that you're not using in your site, or content translations into other languages that you don't need. Most popular CMS integrations support adding options in `gatsby-config.js` to only pull specific locales; a few also support limiting specific content models.

If you have a lot of content you're not using in your Gatsby site, check the options reference of the source plugin you use to see if you can exclude it from getting pulled in altogether!

Check your headless CMS or other data source

With similar amounts of pages and content, Gatsby builds work significantly faster with some headless CMSs than others (largely due to the way various source-plugins are implemented).

Note that generally popular and official source plugins have built in a number of optimizations that custom or less-used source plugins may not have, so if you're sourcing large amounts of data from other plugins, you may want to monitor the build performance of those plugins.

If build speeds are sufficiently painful, your CMS is the primary contributor, and there are significantly faster alternative(s), you may want to consider migrating some or all content.

Conditionally applying plugin options using environment variables

Your source plugin may have options that control which or how many entries, files, etc. are sourced. In order to speed up sourcing for e.g. `gatsby develop` or your staging environment, you can use environment variables to apply different plugin options depending on the current environment. To see an example of this pattern using `gatsby-source-filesystem` you can visit the sourcing from the filesystem documentation.

If your site is multi-language, consider breaking it into several sites

Some organizations with larger Gatsby sites and a strong emphasis on fast build times, have found that breaking a large, multi-language site into smaller sites is effective in significantly reducing build times. The team at Daniel Wellington reduced their build times from 20 minutes to 3 minutes through this technique.

3. Reducing build times from schema creation, image processing, and bundle generation

Use gatsby-plugin-schema-snapshot

If your "build schema" step is slow, try using gatsby-plugin-schema-snapshot, since it will eliminate the need for schema inference.

Reconsider use of advanced options for gatsby-plugin-image

Some options offered in `gatsby-plugin-image`, such as AVIF and traced-svg, increase build time significantly. If image processing is a large proportion of your builds, and those builds are painful, consider whether using `gatsby-plugin-image` on lower-traffic pages is a net benefit.

Decrease bundle size

The build time cost of generating Javascript bundles is proportional to the size of your bundles. Optimizations to improve bundle size, especially of the templates that are most commonly used, should generate significant benefits. For example, if you have a 30k-page site with 25k product pages, to optimize build time start by optimizing your product page template.

4. Advanced options

Look at current experimental flags

The current list of experimental flags available is located in a file in the Gatsby repository. Take a look at this file to see if there are any flags currently available that can speed up your builds.

Increase query concurrency and core count

There are two options available that will increase build speeds, at the cost of also increasing Gatsby's memory usage.

There's a flag called `GATSBY_EXPERIMENTAL_QUERY_CONCURRENCY` which represents the number of queries Gatsby runs in parallel. You can try set this from 4 (the default) to 8, 16, or 32, and seeing what the effect is on build times.

In addition, if your hardware supports logical cores, you can try increasing the `GATSBY_CPU_COUNT` variable (its default is the number of physical cores available on your machine).

Improving develop times

For development purposes, you rarely need the entire site available locally -- a subset is usually fine. For example, if you have 2,000 blog posts, it's probably okay to only have the latest 20 posts available on local development servers.

This can be achieved in two ways:

- For source plugins (eg WordPress) that allow you to specify the number of objects to pull down for any given content type, you can pull down a smaller number of nodes if it's in development.
- For source plugins (like Contentful) that don't offer this option, you can write code in `gatsby-node.js` to restrict the number of pages built in development; eg, iterate only through the first X elements of an array instead of all elements.

In addition, everything above applicable to build times is also applicable to develop times. However, you should probably start by slimming down the number of objects sourced and pages built in development, as that is generally a simpler and quicker to implement solution.

Additional Resources

- We did a deep dive on this with additional information and context in the Optimizing Build Performance On Gatsby webinar.