Deferred Static Generation (DSG) allows you to defer non-critical page generation to the first user request, speeding up build times. Instead of generating *every* page up front, you can decide to generate certain pages at build time and others only when a user accesses the page for the first time. Subsequent page requests use the same HTML and JSON generated during the very first request to this page.

> ***Note:*** *This feature requires running NodeJS server. It is currently fully supported with* `gatsby serve` *and in Gatsby Cloud.*

Creating deferred pages

`config`

Inside File System Route API templates you can export an async function called `config` that returns an object with the key `defer`:

```
export async function config() {
  // Optionally use GraphQL here

  return ({ params }) => {
    return {
      defer: true,
    }
  }
}
```

Read the Deferred Static Generation guide to see a real-world example.

`createPage`

Creating deferred pages is almost identical to creating regular pages. The only difference is the new `defer` argument for `createPage` action. When set to `true`, it tells Gatsby to exclude the page from the build step and instead generate it during the first HTTP request:

```
exports.createPages = async function ({ actions, graphql }) {
  actions.createPage({
    path: "/the-page-path/",
    component: require.resolve("./src/templates/template.js"),
    context: {},
    defer: true, // highlight-line
  })
}
```

The `defer` argument is optional. If it's excluded, the page will be generated at build time by default.

Working with deferred pages locally

Deferred static generation has no effect when using `gatsby develop`. You can work with pages locally as usual.

If you want to test deferred generation specifically, run `gatsby build` and `gatsby serve` from the command line. Deferred pages will be generated during the very first request to the page via `gatsby serve`.

Using in production

Deferred static generation requires a running NodeJS server. You can put NodeJS running `gatsby serve` behind a content delivery network (CDN) like Fastly, however that also requires additional infrastructure (like monitoring, logging, and crash-recovery).

Complete setup is available for you in Gatsby Cloud out-of-the-box.

How it works

The first request against a deferred page is a cache miss because the HTML/JSON wasn't generated yet. On Gatsby Cloud the request is sent to a worker process that leverages an internal database to generate the data of the page. Once the page is generated it'll be sent back to the user immediately. In the background, all generated artifacts are stored so that on a second request the cached response is directly served from the CDN. For that it bypasses the worker process completely.

When you directly visit a page you'll get served the HTML. If you request a page on client-side navigation through Gatsby's Link component the response will be JSON. Gatsby's router uses this to render the page on the client. In the background, the JSON is stored and the HTML is generated so that on a second request of the page (including a direct visit) the page can be served from the CDN.

This all happens automatically and you only need to configure the `defer` key.

Additional Resources

- How-To Guide: Using Deferred Static Generation
- Conceptual Guide: Rendering Options