This document uses the deprecated image plugin. Please check out how to work with the new gatsby-plugin-image.

Optimizing images is a challenge on any website. To utilize best practices for performance across devices, you need multiple sizes and resolutions of each image. Luckily, Gatsby has several useful plugins that work together to do that for images on page components.

The recommended approach is to use GraphQL queries to get images of the optimal size or resolution, then, display them with the `gatsby-plugin-image` component.

Query images with GraphQL

Querying images with GraphQL allows you to access the image's data as well as perform transformations with Sharp, a high-performance image processing library.

You'll need a few plugins for this:

- `gatsby-source-filesystem` plugin allows you to query files with GraphQL
- `gatsby-plugin-sharp` powers the connections between Sharp and Gatsby Plugins
- `gatsby-transformer-sharp` allows you to create multiples images of the right sizes and resolutions with a query

If the final image is of a fixed size, optimization relies on having multiple resolutions of the image. If it is responsive, meaning it stretches to fill a container or page, optimization relies on having different sizes of the same image. See the Gatsby Image documentation for more information.

You can also use arguments in your query to specify exact, minimum, and maximum dimensions. See the Gatsby Image documentation for complete options.

This example is for an image gallery where images stretch when the page is resized. It uses the `fluid` method and the fluid fragment to grab the right data to use in `gatsby-image` component and arguments to set the maximum width as 400px and maximum height as 250px.

```
export const query = graphql`
  query {
    fileName: file(relativePath: { eq: "images/myimage.jpg" }) {
      childImageSharp {
        fluid(maxWidth: 400, maxHeight: 250) {
          ...GatsbyImageSharpFluid
        }
      }
    }
  }
`
```

Optimizing images with gatsby-plugin-image

`gatsby-image` is a plugin that automatically creates React components for optimized images that:

> - *Loads the optimal size of image for each device size and screen resolution*
> - *Holds the image position while loading so your page doesn't jump around as images load*
> - *Uses the "blur-up" effect i.e. it loads a tiny version of the image to show while the full image is loading*
> - *Alternatively provides a "traced placeholder" SVG of the image*

> - *Lazy loads images, which reduces bandwidth and speeds the initial load time*
> - *Uses WebP images, if browser supports the format*

Here is an image component that uses the query from the previous example:

```
<Img fluid={data.fileName.childImageSharp.fluid} alt="" />
```

Using fragments to standardize formatting

What if you have a bunch of images and you want them all to use the same formatting?

A custom fragment is an easy way to standardize formatting and re-use it on multiple images:

```
export const squareImage = graphql`
  fragment squareImage on File {
    childImageSharp {
      fluid(maxWidth: 200, maxHeight: 200) {
        ...GatsbyImageSharpFluid
      }
    }
  }
`
```

The fragment can then be referenced in the image query:

```
export const query = graphql`
  query {
    image1: file(relativePath: { eq: "images/image1.jpg" }) {
      ...squareImage
    }

    image2: file(relativePath: { eq: "images/image2.jpg" }) {
      ...squareImage
    }

    image3: file(relativePath: { eq: "images/image3.jpg" }) {
      ...squareImage
    }
  }
`
```

Additional resources

- Gatsby Image API docs
- Using gatsby-image with Gatsby, free egghead.io playlist
- gatsby-image plugin README file
- gatsby-plugin-sharp README file
- gatsby-transformer-sharp README file