Preview

This plugin supports Preview and has been designed to make the WordPress admin preview experience as seamless as possible.

How Preview works

When configured properly by a developer, preview should function almost identically to how it does in regular WordPress. The admin updates content and presses “preview” and the preview template opens and displays the preview.

Setting up Preview

You can find our tutorial on setting up WPGatsby here. Part-way down the page there are instructions you can follow on setting up Preview.

Debugging the build process of Previews

If you enable the plugin option `options.debug.preview` by setting it to `true`, you will see additional logging through the Preview build process with information such as the contents of the webhook body that was sent to Gatsby, the preview node data, and the list of preview actions that were pulled from WordPress. See the plugin options documentation for more info.

Gutenberg and ACF

Note that if you use these two together, you cannot preview ACF data. This is a core WordPress Gutenberg issue. Follow https://github.com/WordPress/gutenberg/issues/16006 for more information. If you use ACF and would like to preview data changes, use the Classic Editor plugin for now.

Common gatsby-node.js problems that break previews

1. When querying for nodes, the previewed node is filtered out

Make sure that when `process.env.NODE_ENV === "development"` you aren’t filtering your `gatsby-node.js` node queries. For example, the following will break Previews:

```
exports.createPages = async ({ graphql }) => {
  const graphqlResult = await graphql(/* GraphQL */ `
    query {
      allWpPost(filter: { status: { eq: "publish" } }) {
        edges {
          node {
            id
            uri
          }
        }
      }
    }
  `)
}
```

Filtering by a specific category can also break previews since adding categories to posts is not previewable in WordPress:

```
exports.createPages = async ({ graphql }) => {
  const graphqlResult = await graphql(/* GraphQL */ `
    query {
      allWpPost(
        filter: {
          categories: { nodes: { elemMatch: { name: { eq: "Blog" } } } }
        }
      ) {
        edges {
          node {
            id
            uri
          }
        }
      }
    }
  `)
}
```

2. Using createPagesStatefully

The correct Node API for creating pages is `createPages`, using `createPagesStatefully` will prevent previews from working.

Built in Preview plugin options preset

In order to speed up previews, there are some built in default plugin options for when your Gatsby site is in Preview mode. This preset disables static asset transformations in html fields and limits the number of nodes initially fetched during a cold build. You can disable this preset by passing `null` to the preset option. Any options you've set yourself will override the preset options.

```
{
    resolve: `gatsby-source-wordpress`,
    options: {
        url: `https://your-site.com/graphql`,
        presets: null
    }
}
```

The preset (as found in src/models/gatsby-api.ts) is:

```
{
  // this is an internal name
  presetName: `PREVIEW_OPTIMIZATION`,

  // these are the conditions the preset will be added under
  useIf: () => inDevelopPreview || inPreviewRunner,

  // these options will be merged into the global default options and your options will be
  options: {
    html: {
      useGatsbyImage: false,
      createStaticFiles: false,
    },
    type: {
      __all: {
        //   all nodes are limited to 50 in cold builds
        limit: 50,
      },
      Comment: {
        //   all comments are excluded
        limit: 0,
      },
      //   there's no limit to the following three types in cold builds
      Menu: {
        limit: null,
      },
      MenuItem: {
        limit: null,
      },
      User: {
        limit: null,
      },
    },
  },
}
```

How Preview works behind the scenes

When the WP “preview” button is pressed, a JWT is generated (with an expiry time of 1 hour) and POST’ed to the Gatsby Preview instance webhook. The Preview instance then uses this short-lived JWT to request a list of pending previews for all users. Gatsby starts processing each pending preview. At the same time, WordPress automatically opens the WP preview template which has been overridden by WPGatsby to redirect to Gatsby Cloud's Content Sync service. This service handles the loading/error states and will redirect the user to the right page when it's been built.

Preview Security Considerations

In order to support multiple users previewing simultaneously (and/or content updates happening at the same time as previews), WPGatsby needs to generate a user-agnostic JWT token and post that to Gatsby to pull all pending previews for any variety of users. This JWT can authenticate as any user. What this means is your Gatsby instance is being trusted at the same level as WP core code or WP plugins/themes. You should make sure that only trusted individuals or teams have code-level access to your WP server code/hosting and your Gatsby code/hosting. You should always use SSL for your WP host and your Gatsby Preview instance (on Gatsby Cloud this is taken care of for you). This JWT is only accessible when POST'ed to your Gatsby Preview instance during previews. The settings for where this JWT is POST'ed to can only be configured by administrators (via the WPGatsby settings page) or by anyone with code-level access to your WP instance.

Additionally Gatsby Preview has no conception of user access roles. This means anyone with frontend, GraphiQL, or code level access to your Preview instance can see any currently existing Gatsby Previews that have been processed.

:point_left: Back to Features