

# gatsby-source-lever

Source plugin for pulling data into [Gatsby](#) from [Lever.co](#).

## Wish list

- ☒ gatsby source plugin for Lever.co
- ☒ tests
- ☐ example site

## Install

```
npm install gatsby-source-lever
```

## How to use

```
// In your gatsby-config.js
module.exports = {
  plugins: [
    {
      resolve: "gatsby-source-lever",
      options: {
        // Your Lever site instance name.
        site: "lever",
        // Set verboseOutput to true to display a verbose output on `npm run
develop` or `npm run build`
        // It can help you debug specific API Endpoints problems
        verboseOutput: false,
      },
    },
  ],
}
```

## GraphQL Query to get all jobs

```
allLever {
  edges {
    node {
      id
      lever_id
      createdAt
      text
      hostedUrl
      applyUrl
      categories {
        commitment
        location
        team
      }
    }
  }
}
```

```

    description
    descriptionPlain
    lists {
      text
      content
    }
    additional
    additionalPlain
  }
}
}

```

## Site's `gatsby-node.js` example

If you wish to create Gatsby Pages for each Lever.co jobs, you can modify your `gatsby-node.js` .

```

const _ = require(`lodash`)
const Promise = require(`bluebird`)
const path = require(`path`)
const { slash } = require(`gatsby-core-utils`)

exports.createPages = ({ graphql, actions }) => {
  const { createPage } = actions
  return new Promise((resolve, reject) => {
    // The "graphql" function allows us to run arbitrary
    // queries against the local WordPress graphql schema. Think of
    // it like the site has a built-in database constructed
    // from the fetched data that you can run queries against.

    // ==== PAGES (LEVER) ====
    graphql(
      `
      {
        allLever {
          edges {
            node {
              id
            }
          }
        }
      }
    `
    )
    .then(result => {
      if (result.errors) {
        console.log(result.errors)
        reject(result.errors)
      }

      // Create Lever pages.
      const pageTemplate = path.resolve("../src/templates/page.js")

```

```

// We want to create a detailed page for each
// lever node. We'll just use the ID for the slug.
_.each(result.data.allLever.edges, edge => {
  // Gatsby uses Redux to manage its internal state.
  // Plugins and sites can use functions like "createPage"
  // to interact with Gatsby.
  createPage({
    // Each page is required to have a `path` as well
    // as a template component. The `context` is
    // optional but is often necessary so the template
    // can query data specific to each page.
    path: `/${edge.node.id}/`,
    component: slash(pageTemplate),
    context: {
      id: edge.node.id,
    },
  })
})
// ==== END PAGES ====

// resolve() must be called at the end so Gatsby knows that we're done add
pages.
.then(resolve())
})
}

```