

# AN INTRODUCTION TO STATIC SITE GENERATORS FOR DRUPALISTS

**Brian Perry**

March 22, 2019

Slides: <http://bit.ly/d8-ssg>

**bounteous**

# BRIAN PERRY

- Lead Front End Dev at Bounteous
- Rocking the Chicago 'burbs
- Lover of all things components...  
...and Nintendo
- JAMStack curious



**d.o: brianperry**

**twitter: bricomed**

**github: backlineint**

**nintendo: wabrian**

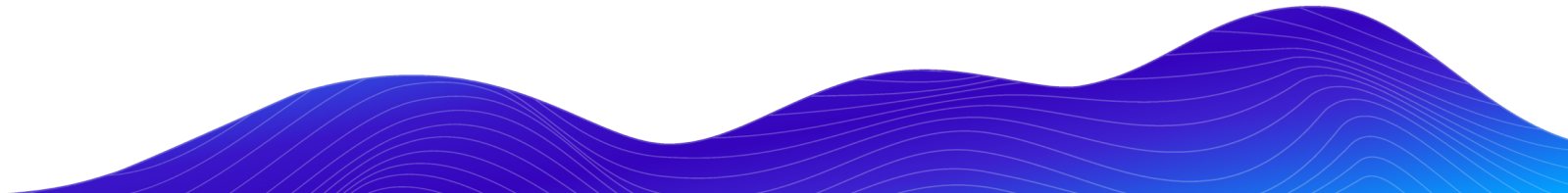
**brianperryinteractive.com**

# bounteous



# STATIC SITE GENERATORS

An Overview





# THIS IS A STATIC SITE





HIGHER. FURTHER. FASTER.  
**CAPTAIN MARVEL!**  
In theaters March 8th



INFO

PLAY THE GAME

MULTIMEDIA

GUESTBOOK

LIMIT  
ONE

GET TICKETS


LIMIT  
ONE

watch trailer



883034968

# THIS IS ALSO A STATIC SITE

 **React**

[Docs](#) [Tutorial](#) [Community](#) [Blog](#)

v16.6.1 [GitHub](#)

# Rendering Elements

Elements are the smallest building blocks of React apps.

An element describes what you want to see on the screen:

```
const element = <h1>Hello, world</h1>;
```

Unlike browser DOM elements, React elements are plain objects, and are cheap to create. React DOM takes care of updating the DOM to match the React elements.

**Note:**

One might confuse elements with a more widely known concept of “components”. We will introduce components in the [next section](#). Elements are what components are “made of”, and we encourage you to read this section before jumping ahead.

INSTALLATION ▾

MAIN CONCEPTS ▲

1. Hello World

2. Introducing JSX

**3. Rendering Elements**

4. Components and Props

5. State and Lifecycle

6. Handling Events

7. Conditional Rendering

8. Lists and Keys

9. Forms

10. Lifting State Up

11. Composition vs Inheritance

12. Thinking In React

ADVANCED GUIDES ▾

API REFERENCE ▾

HOOKS (PROPOSAL) ▾

CONTRIBUTING ▾

EVEN THIS IS A (MOSTLY) STATIC SITE



# SO WHAT IS A STATIC SITE GENERATOR?

- Generates a completely static build (html, css, js) of a site based on raw data
  - Data is often markdown but could be a database, an API, CSVs and so on
  - Typically makes use of a templating engine
- End result doesn't require a monolithic stack, can be hosted anywhere
- Popular projects: Jekyll, Hugo, Hexo, Gatsby, and many, many others.

# ADVANTAGES OF STATIC SITES

- Speed
  - Browsers are pretty great at rendering static html
  - CDNs
- Security
  - Data source only needs to exist at build time
- Hosting
  - Simpler stack, easier to scale
- Developer Experience (depends a bit on the tool)

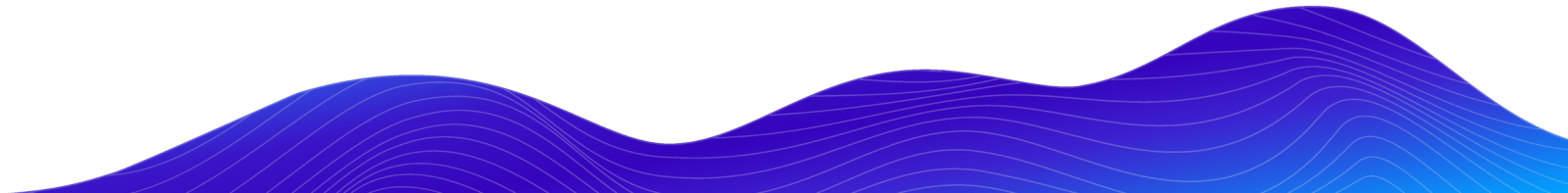
# BUT... I LOVE DRUPAL

Why would you use a static site generator with Drupal?

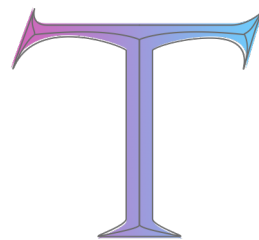
- So much effort to make things static-like
  - Think of the effort that goes into making Drupal sites close to static – varnish, memcache, Drupal caching, etc.
- Think about the next inevitable security update
- Impact on hosting costs
- Drupal's admin UI

# THE STATIC SITE GENERATOR LANDSCAPE

From a Drupal Perspective







# JEKYLL



The granddaddy of modern static site generators.

- Ruby project
- Liquid templating engine
- Markdown (most common), YAML, JSON and CSV data sources
- Content in version control
- Supported natively by Github Pages
- Incremental builds
- Hugely popular

# JEKYLL - DATA

```
1 ---
2 layout: post
3 title: "Welcome to Jekyll!"
4 date: 2018-11-13 17:51:09 -0600
5 categories: jekyll update
6 ---
7 You'll find this post in your `_posts` directory. Go ahead and edit it and
8 re-build the site to see your changes. You can rebuild the site in many
9 different ways, but the most common way is to run `jekyll serve`, which
10 launches a web server and auto-regenerates your site when a file is updated.
11
12 To add new posts, simply add a file in the `_posts` directory that follows the
13 convention `YYYY-MM-DD-name-of-post.ext` and includes the necessary front
14 matter. Take a look at the source for this post to get an idea about how it
15 works.
16
17 Jekyll also offers powerful support for code snippets:
18
19 {% highlight ruby %}
20 def print_hi(name)
21   puts "Hi, #{name}"
22 end
23 print_hi('Tom')
24 #=> prints 'Hi, Tom' to STDOUT.
25 {% endhighlight %}
26
27 Check out the \[Jekyll docs\][jekyll-docs] for more info on how to get the most
28 out of Jekyll. File all bugs/feature requests at \[Jekyll's GitHub
29 repol\][jekyll-gh]. If you have questions, you can ask them on \[Jekyll
```

# JEKYLL - TEMPLATING

```
1      ---
2      layout: default
3      ---
4
5      <div class="home">
6          {%- if page.title -%}
7          <h1 class="page-heading">{{ page.title }}</h1>
8          {%- endif -%}
9
10         {{ content }}
11
12         {%- if site.posts.size > 0 -%}
13         <h2 class="post-list-heading">{{ page.list_title | default: "Posts" }}</h2>
14         <ul class="post-list">
15             {%- for post in site.posts -%}
16             <li>
17                 {%- assign date_format = site.minima.date_format | default: "%b %-d, %Y" -%}
18                 <span class="post-meta">{{ post.date | date: date_format }}</span>
19                 <h3>
20                     <a class="post-link" href="{{ post.url | relative_url }}">
21                         {{ post.title | escape }}
22                     </a>
23                 </h3>
24                 {%- if site.show_excerpts -%}
25                 {{ post.excerpt | slice: 0, 400 }}...
26                 {%- endif -%}
27             </li>
28             {%- endfor -%}
29         </ul>
```

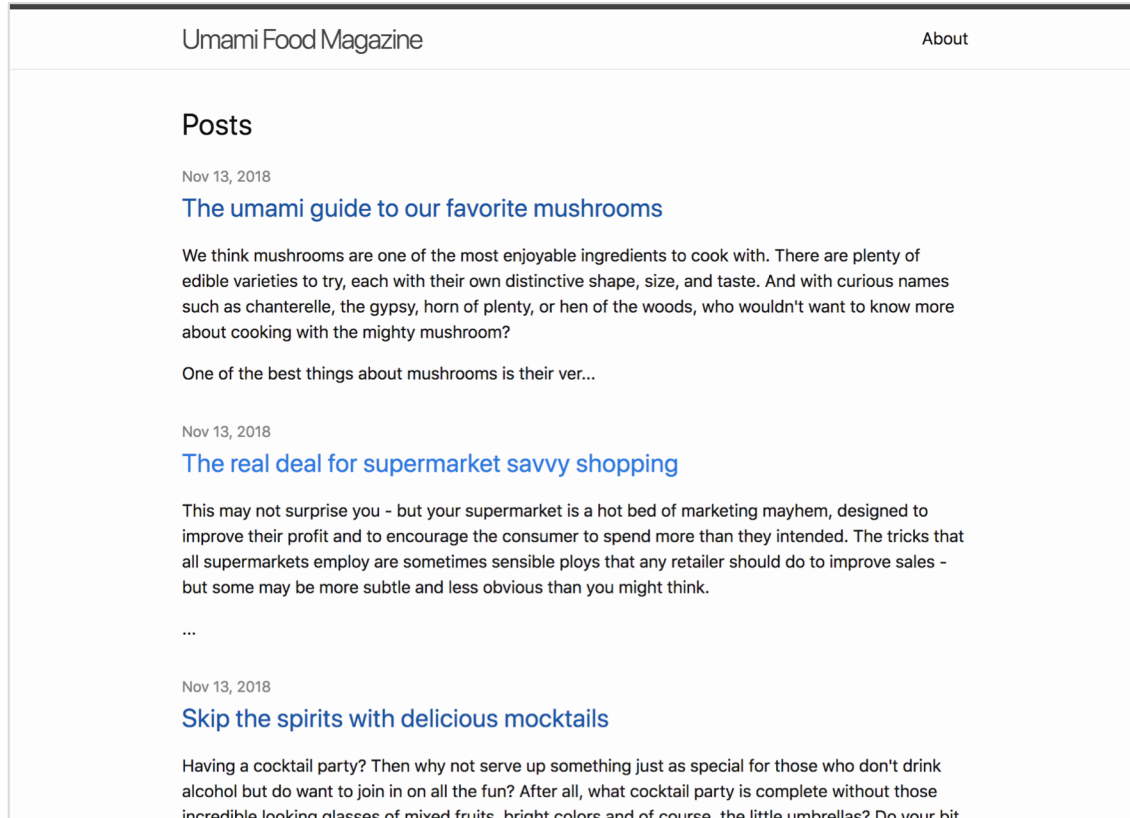
# JEKYLL – WITH DRUPAL

Not a lot of formal options with Drupal 8

- Drupal 6 and 7 importers exist, but not Drupal 8 (that I could find)
- So then what?
  - Export data as flat files
  - Jekyll CSV/RSS importer
  - YAML, JSON or CSV in \_data directory

```
$ ruby -r rubygems -e 'require "jekyll-import";  
  JekyllImport::Importers::CSV.run({  
    "file" => "my_posts.csv"  
  })'
```

# JEKYLL – OUTPUT





For you if:

- You want a simple road tested option
- You want content in version control
- Markdown / flat file data is practical
- You don't hate Ruby



## Other Resources:

- [Jekyll Docs](#)
- [The New Mediagurrent.com:  
Adventures in Decoupled Drupal](#)

## Similar projects:

- [Hexo](#)
- [Hugo](#)
- [Eleventy](#)



# GATSBY



Build blazing fast apps and websites  
with React

- React project
- JSX Templating
- GraphQL to query data
- Thriving plugin ecosystem
- The new(ish) hotness
- Crazy fast. Seriously.
- Not just for static sites

# GATSBY - INSTALLING AND RUNNING

## Create a new site.

```
npx gatsby new gatsby-site
```

SHELL

## Change directories into site folder.

```
cd gatsby-site
```

SHELL

## Start development server.

```
npm run develop
```

SHELL

## Create a production build.

```
npm run build
```

SHELL

## Serve the production build locally.

```
npm run serve
```

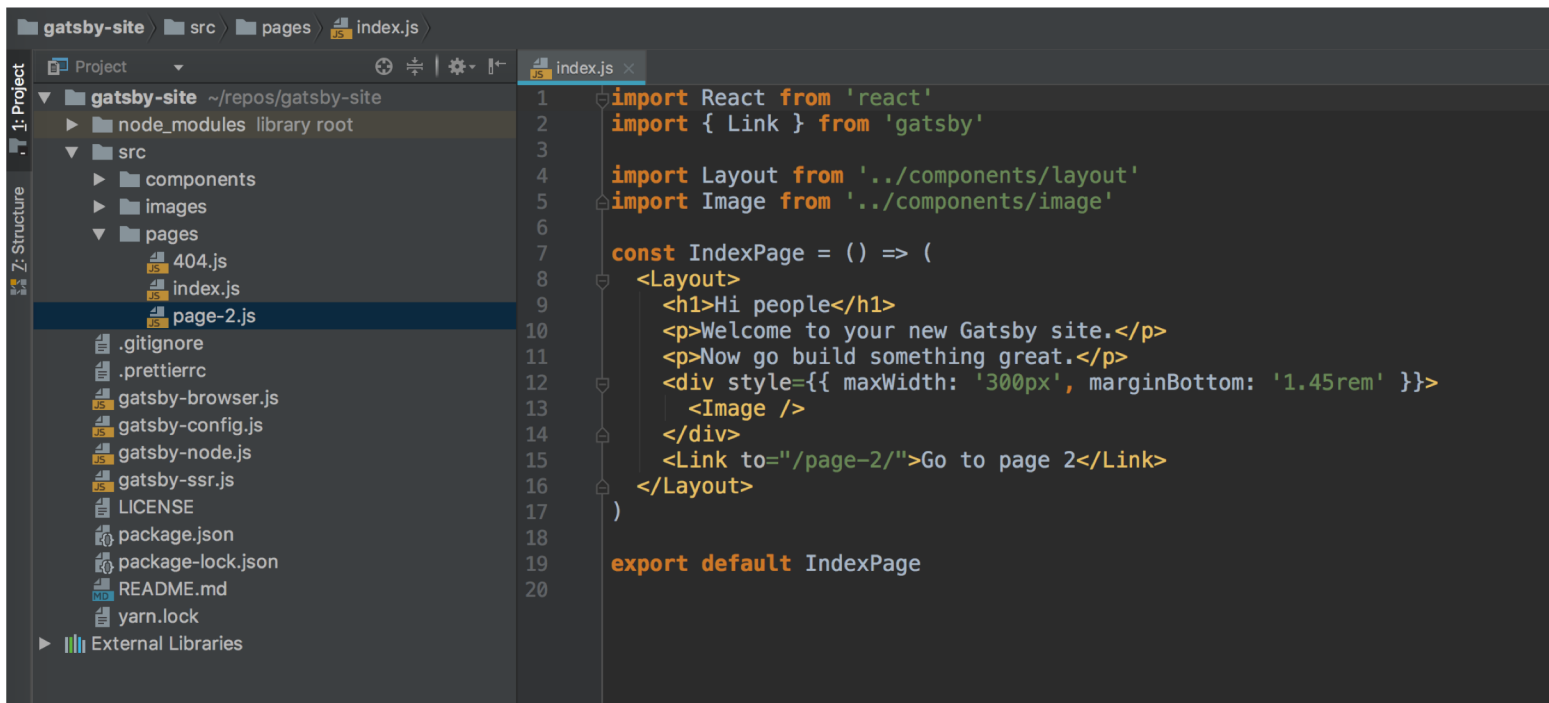
SHELL

Or with a starter:

```
npx gatsby new gatsby-blog  
https://github.com/gatsbyjs/gatsby-starter-blog
```

# GATSBY – DATA

Could be react components in src/pages



The screenshot shows a code editor with a project named 'gatsby-site'. The left sidebar displays the file structure, with the 'pages' directory expanded. The 'index.js' file is selected. The main editor area shows the code for 'index.js', which imports React and Gatsby components, defines an 'IndexPage' component, and exports it as the default.

```
1 import React from 'react'
2 import { Link } from 'gatsby'
3
4 import Layout from '../components/layout'
5 import Image from '../components/image'
6
7 const IndexPage = () => (
8   <Layout>
9     <h1>Hi people</h1>
10    <p>Welcome to your new Gatsby site.</p>
11    <p>Now go build something great.</p>
12    <div style={{ maxWidth: '300px', marginBottom: '1.45rem' }}>
13      <Image />
14    </div>
15    <Link to="/page-2/">Go to page 2</Link>
16  </Layout>
17 )
18
19 export default IndexPage
20
```

# GATSBY – DATA

Could be markdown from filesystem

```
gatsby-config.js
1 module.exports = {
2   siteMetadata: {
3     title: `Pandas Eating Lots`,
4   },
5   plugins: [
6     {
7       resolve: `gatsby-source-filesystem`,
8       options: {
9         name: `src`,
10        path: `${__dirname}/src/`,
11      },
12    },
13    `gatsby-transformer-remark`,
14    `gatsby-plugin-glamor`,
15    {
16      resolve: `gatsby-plugin-typography`,
17      options: {
18        pathToConfigModule: `src/utils/typography`,
19      },
20    },
21  ],
22 };
```

# GATSBY – CONSTRUCT QUERIES VIA GRAPHQL

GraphQL

Prettify

History

< Docs

```
1 {
2   allMarkdownRemark {
3     edges {
4       node {
5         frontmatter {
6           title
7           date
8         }
9         html
10        excerpt
11      }
12    }
13  }
14 }
15
```

```
{
  "data": {
    "allMarkdownRemark": {
      "edges": [
        {
          "node": {
            "frontmatter": {
              "title": "Sweet Pandas Eating Sweets",
              "date": "2017-08-10"
            },
            "html": "<p>Pandas are really sweet.</p>\n<p>Here's a video of a panda eating sweets.</p>\n<iframe width=\n560\nheight=\n315\n src=\n\"https://www.youtube.com/embed/4n0xNbfJLR8\n\" frameborder=\n0\n allowfullscreen>\n</iframe>\n",
            "excerpt": "Pandas are really sweet. Here's a video of a panda eating sweets."
          },
          "node": {
            "frontmatter": {
              "title": "Pandas and Bananas",
              "date": "2017-08-21"
            },
            "html": "<p>Do Pandas eat bananas? Check out this short video that shows that yes! pandas do\nseem to really enjoy bananas!</p>\n<iframe width=\n560\n height=\n315\n src=\n\"https://www.youtube.com/embed/4SZ11r20_bY\n\" frameborder=\n0\n allowfullscreen>\n</iframe>\n",
            "excerpt": "Do Pandas eat bananas? Check out this short video that shows that yes! pandas do\nseem to really enjoy bananas!"
          }
        ]
      }
    }
  }
}
```

QUERY VARIABLES

# GATSBY – 'TEMPLATING'

JS recipes.js x

```
1  import React from "react"
2  import { Link, graphql } from "gatsby"
3
4  import Layout from "../layouts"
5  import Container from "../components/container"
6
7  const AllRecipes = ({ data }) => (
8    <Layout>
9      <Container>
10        <h1>Recipes</h1>
11        <ul>
12          {data.allRecipes.edges.map(({ node }) => (
13            <li>
14              <Link to={node.fields.slug}>{node.title}</Link>
15            </li>
16          ))}
17        </ul>
18      </Container>
19    </Layout>
20  )
21
22  export default AllRecipes
```

```
23
24  export const query = graphql`
25    query {
26      allRecipes(limit: 1000) {
27        edges {
28          node {
29            title
30            fields {
31              slug
32            }
33          }
34        }
35      }
36    }
37  `
```

# GATSBY – WITH DRUPAL



## gatsby-source-drupal

[See starters that use this](#)

Source plugin for pulling data (including images) into Gatsby from Drupal sites.

Pulls data from Drupal 8 sites with the [Drupal JSONAPI module](#) installed.

An example site built with the headless Drupal distro [ContentaCMS](#) is at <https://using-drupal.gatsbyjs.org/>

`apiBase` Option allows changing the API entry point depending on the version of `jsonapi` used by your Drupal instance. The default value is `jsonapi`, which has been used since `jsonapi` version `8.x-1.0-alpha4`.

## Install

```
npm install --save gatsby-source-drupal
```

## How to use

```
// In your gatsby-config.js
plugins: [
  {
    resolve: `gatsby-source-drupal`,
    options: {
      baseUrl: `https://live-contentacms.pantheonsite.io/`,
      apiBase: `api`, // optional, defaults to `jsonapi`
    },
  },
]
```

## How to query

You can query nodes created from Drupal like the following:

```
{
  allArticle {
    edges {
      node {
        title
        internalId
        created(formatString: "DD-MMM-YYYY")
      }
    }
  }
}
```

# GATSBY – OUTPUT

Search by keyword, ingredient, dish

Login


## Umami


Food Magazine

Recipes

Our Recipe Pick


Frankfurter salad with mustard dressing





Main course

Majorcan vegetable bake



Main course

Barbecued beef - Chinese style





For you if:

- You prioritize performance
- You think in React
- You want data source flexibility
- You want to leverage the power of GraphQL
- You want to go beyond static

# GATSBY



Other resources:

- [Gatsby Docs](#)
- Tons of tutorials and talks
- Boina Gatsby [Starter](#) / [Distribution](#)
- [Lando Gatsby Drupal](#)
- [gatsby-remark-drupal](#)



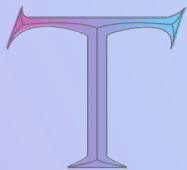
Similar projects:

- [VuePress](#)
- [Gridsome](#)
- [Next.js](#)
- [React Static](#)

You may also enjoy:

- [Quicklink Drupal module](#)

# TOME



A static site generator for Drupal 8

- A Drupal Project!
- Generates a static site that looks like... your Drupal site
- Exports and imports content as json
- Support for incremental builds

# TOME - INSTALLING AND RUNNING

- `composer create-project drupal-tome/tome-project my_site --stability dev --no-interaction --no-install`
- `composer install`
- `drush tome:init`
- `drush runserver 127.0.0.1:8888`
  - Create content, see it written to `/content`
- Re-import based on static content:
  - `drush tome:install`
- Generate a static build:
  - `drush tome:static`

*Can also install in an existing project. See project page.*

# TOME - DATA

```
1  {
2    "uuid": [
3      {
4        "value": "60e94770-2e74-4f13-a2f8-9ae6fd94fe62"
5      }
6    ],
7    "langcode": [
8      {
9        "value": "en"
10     }
11   ],
12   "type": [
13     {
14       "target_id": "article",
15       "target_type": "node_type",
16       "target_uuid": "f9c64e8b-5a0f-4b07-9086-7c5d7f6508a0"
17     }
18   ],
19   "revision_timestamp": [
20     {
21       "value": "2018-06-19T14:33:20+00:00",
22       "format": "Y-m-d\\TH:i:sP"
23     }
24   ],
25   "revision_uid": [
26     {
27       "target_type": "user",
28       "target_uuid": "12d73d3a-832c-487f-bf92-d70d619238c5",
29       "url": "\\user\\1"
30     }
31   ]
32 }
```

# TOME - TEMPLATING

It's Drupal!



# TOME – OUTPUT



[Home](#)

[Articles](#)

[Recipes](#)

## Super easy vegetarian pasta bake

A wholesome pasta bake is the ultimate comfort food. This delicious bake is super quick to prepare and an ideal midweek meal for all the family.

Super easy vegetarian pasta bake







For you if:

- You want to keep using the tools provided by Drupal, but want a static site.
- You want Drupal content under version control
- You want to archive a D8 site
- You want to import flat file Drupal data into another system

# TOME



Additional resources:

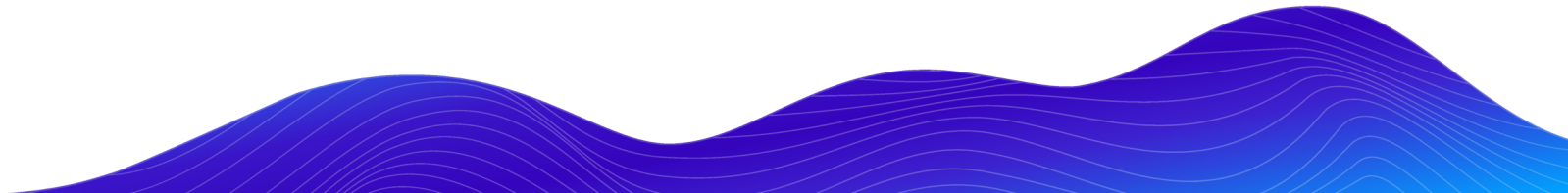
- [Tome Docs](#)
  - Getting started walkthrough
- Try it out on your site

Similar projects:

- [static\\_generator](#)
- [default\\_content](#)
- [yaml\\_content](#)

# AUTOMATING DEPLOYMENTS

“I updated content in Drupal, now what?”



# DEPLOY A BUILD ASSET

backlineint / backlineint.github.io

Unwatch 1

Star 0

Fork 0

<> Code

Issues 0

Pull requests 0

Projects 0

Wiki

Insights

Settings

Deployed to github-pages

d279d0c

 was deployed by backlineint on Oct 16 

Active

View deployment

Activity log

Show: All environments ▾

github-pages

 at d279d0c  
Deployed by backlineint on Oct 16 

Active

View deployment

github-pages

 at f668045  
Deployed by backlineint on Sep 11 

Active

View deployment

github-pages

 at ede8b47  
Deployed by backlineint on Aug 28 

Active

View deployment

github-pages

 at 39a1a42  
Deployed by backlineint on Jul 28 

Active

View deployment

# TRIGGER A BUILD WITH A COMMIT

General

**Build & deploy**

Continuous Deployment

Post processing

Deploy notifications

Domain management

Functions

Identity

Forms

Access control

## Continuous Deployment

Settings for Continuous Deployment from a Git repository

### Deploy settings

Repository:	<a href="https://github.com/backlineint/camp-perry-gatsby">https://github.com/backlineint/camp-perry-gatsby</a>
Build command:	<b>gatsby build</b>
Publish directory:	<b>public/</b>
Production branch:	<b>master</b>
Branch deploys:	<b>Deploy only the production branch and its deploy previews</b>
Public deploy logs:	<b>Logs are public</b>

[Learn more about common configuration directives in the docs →](#)


Edit settings

# TRIGGER A BUILD WITH A WEBHOOK

## Add webhook ☆

[Home](#) » [Administration](#) » [Configuration](#) » [Web services](#) » [Webhook](#)

**Label \***



Machine name: staging\_preview [\[Edit\]](#)


Label for the Webhook.

**Type \***

Outgoing ▾


The webhook type, e.g. incoming or outgoing.

**Payload URL \***



Target URL for your payload.

**Secret**



Secret that the target website gave you.

☒ **Active**

Shows if the webhook is active or not.

**Content Type**

application/json ▾

The Content Type of your webhook.

<input type="checkbox"/> ENTITY TYPE	EVENT
<input type="checkbox"/> User	Create
<input type="checkbox"/> User	Update

# DEPLOY A BUILD WITH TOME

## Deploy to Netlify

[Generate](#)[Download](#)[Preview](#)[Deploy](#)

[Home](#) » [Administration](#) » [Configuration](#)

Submitting this form will deploy the latest static build to Netlify as a draft.

**Deploy title \***

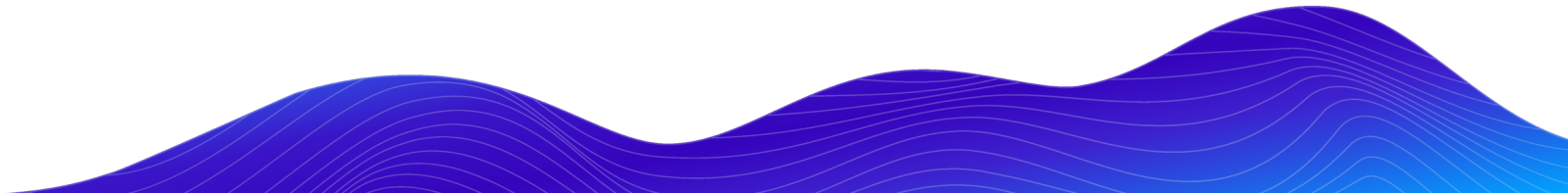
Sent from Tome Netlify

A title to identify this build.

Deploy

# CHALLENGES

“But my site can’t be static...”





# POTENTIAL ROADBLOCKS TO A STATIC BUILD

- Content volume / build times
- Preview experience
- Alternative solutions exist for:
  - Form API
  - Authentication

## CHALLENGE YOUR PERCEPTIONS

Ask yourself:

**“Why can’t this site be static?”**



# CONTRIBUTION DAY

## Saturday 10am to 4pm

You don't have to know code to give back!

New Contributor training 10am to Noon  
with **AmyJune Hinline** of Kanopi Studios

# PLEASE PROVIDE YOUR FEEDBACK!

<https://mid.camp/241>

The top rated sessions will be captioned, courtesy of  
Clarity Partners

**THANK YOU,  
QUESTIONS?**