ADAM BERGSTEIN & BRIAN PERRY

CAN WE FIGURE THIS DRUPAL COMPONENT THING OUT ALREADY?

HELLO THERE

I'M NERDSTEIN

Hook 42

HELLO THERE

I’M BRIAN PERRY (and I lack a strong personal brand)

HS2 SOLUTIONS

OUTLINE OF THE TALK

INTRODUCTION

KEY CONCEPTS

IMPLEMENTATION IDEAS

EMERGING CONCEPTS

SECTION 1:

INTRODUCTION

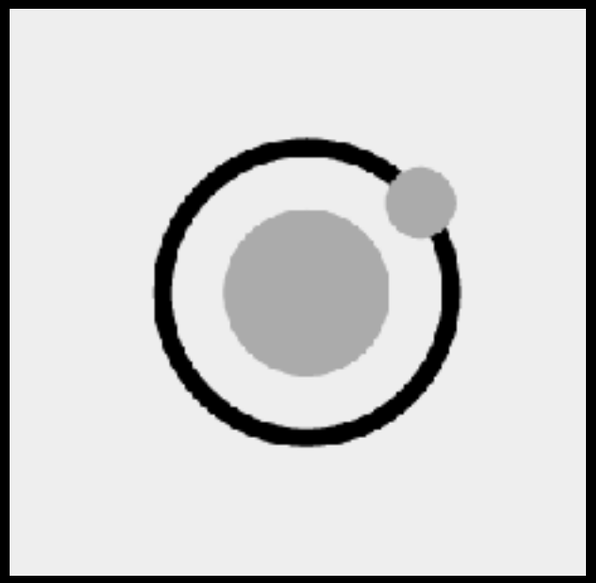
A multi year conversation between Adam and Brian continues right now…

Over time our opinions on best practices in component based development have continued to align

…mostly.

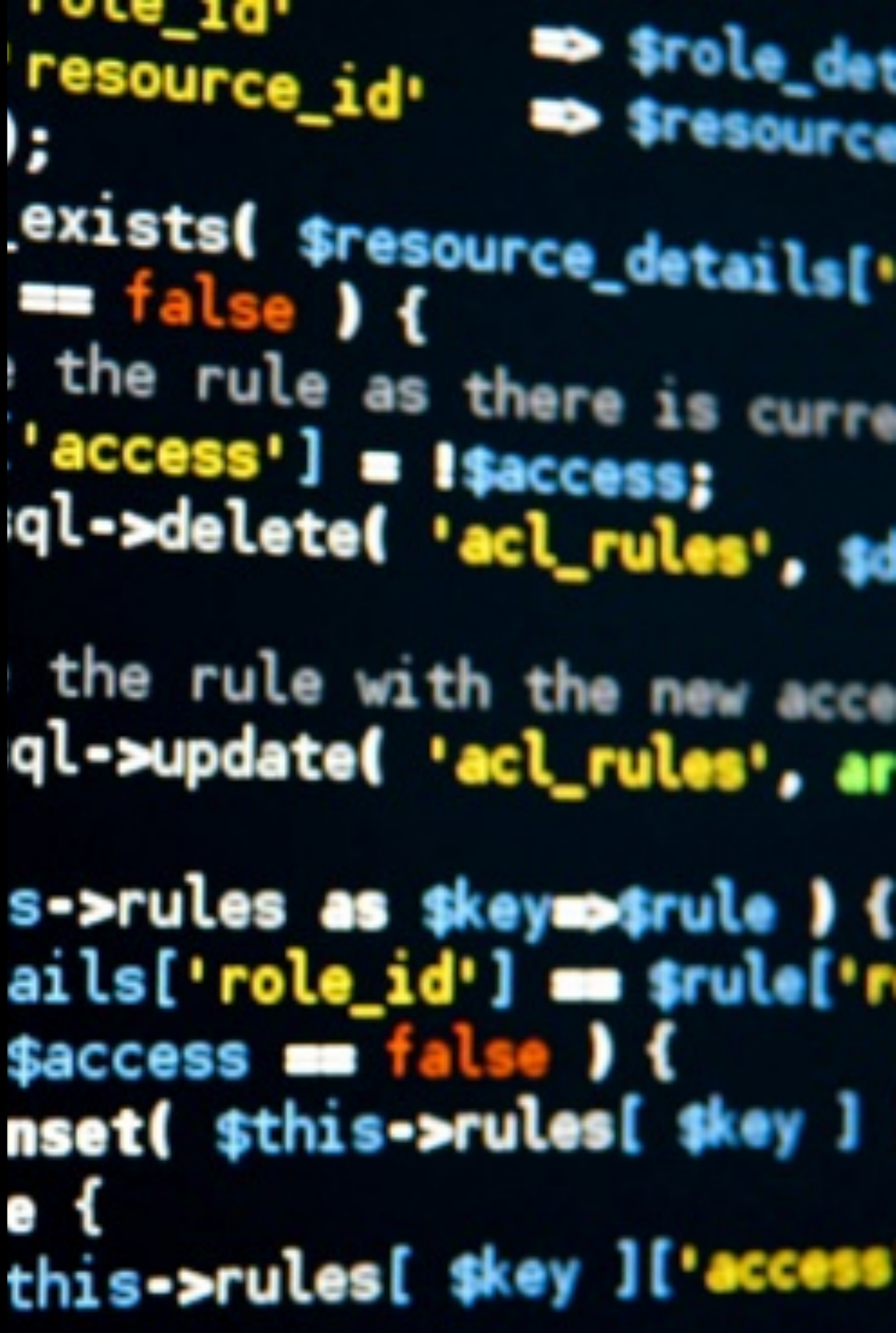
DESIGN

IMPLEMENTATION



EXAMPLES IN THE WILD

- Pattern Lab Drupal Starter Kit
- Emulsify Drupal theme
- Shila Drupal theme
- Several other solutions



LIMITATIONS

- Tightly coupled with Drupal limits reusability
  - Outside of Drupal
  - Secondary Drupal sites
- Technology bias (SASS, LESS, etc)
- Poor documentation on “how” and “why” problems were being solved

NERDSTEIN

Exploring simplicity in Drupal design components

On 02 Oct, 2017 By admin

Component-based architectures have become both a popular and fairly crowded space in the Drupal community. For over a year, I have followed the progress of some tools created by those leveraging Pattern Lab as a component based design library. I can't claim to know the full breadth of problems these individuals encountered, many of which are experienced technologists in our space. But, every solution I have seen has been complex and demonstrates some architectural red flags. I wanted to take a fresh look. I paid a designer to redesign my blog for a migration to Drupal 8. This presented the perfect opportunity to try this out. Consider the following post a simple approach you might be able to use.

I set out to explore this space independently based on my intuition that this seemed overly complex and didn't need to be. I wanted an approach that was both a simple integration and completely deccupled, as I believe any design library should be capable of being reused throughout any enterprise of digital properties. This means not packaging Pattern Lab with a specific Drupal theme. This separation also helps with a separation of duties and defined collaboration between a creative team and a technical team. I respect and wanted to preserve the conventional model of a visual experience driving the technology. Architecturally, I wanted the creative artifacts to be reused directly by Drupal (as much as possible) to afford ongoing parity with proposed design changes. The rest of this blog post articulates my approach and findings.

First, let's start with Pattern Lab. While I received an HTML prototype from the designer, I actually believe Pattern Lab eliminates the need for conventional prototyping. The design library, specifically the "pages" aspect of atomic design, effectively replaces a conventional prototype. While my exercise was converting what the designer turned over to me, I believe it to be perfectly reasonable for Pattern Lab to assume this responsibility both as a prctotyping tool and a living design library routinely updated as one.

PROBLEMS

- Create a fully decoupled solution
- Shared technology baseline between the design system and the content management system
- The packaging and delivery of design system assets
- The implementation of design system assets in the content management system
- Change management of ongoing changes

PROBLEMS

- Create a fully decoupled solution
- Shared technology baseline between the design system and the content management system
- The packaging and delivery of design system assets
- The implementation of design system assets in the content management system
- Change management of ongoing changes

PROBLEMS

- Create a fully decoupled solution
- Shared technology baseline between the design system and the content management system
- The packaging and delivery of design system assets
- The implementation of design system assets in the content management system
- Change management of ongoing changes

PROBLEMS

- Create a fully decoupled solution
- Shared technology baseline between the design system and the content management system
- The packaging and delivery of design system assets
- The implementation of design system assets in the content management system
- Change management of ongoing changes

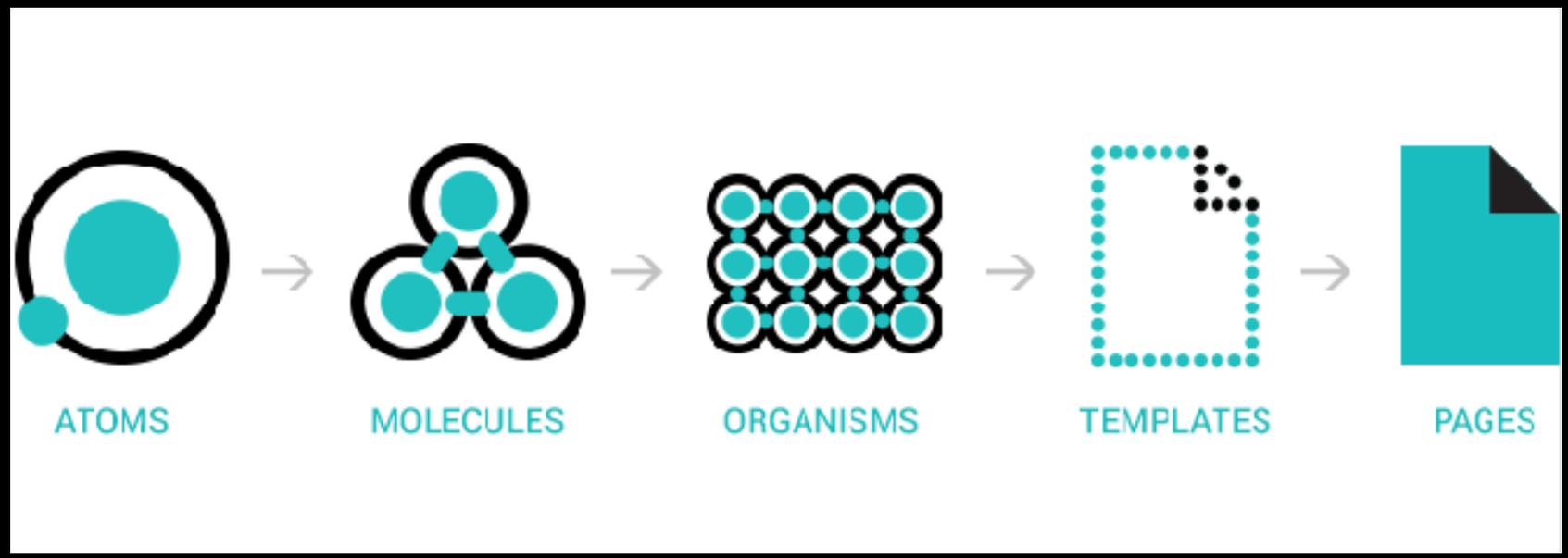
PROBLEMS

- Create a fully decoupled solution
- Shared technology baseline between the design system and the content management system
- The packaging and delivery of design system assets
- The implementation of design system assets in the content management system
- Change management of ongoing changes

SECTION 2:

KEY CONCEPTS

NORMALIZATION / ATOMIC DESIGN



- Design patterns resemble database normalization
- Create patterns in their smallest, atomic form
- Reuse as needed, DRY Principle

“KISS” CONCEPT

- The design system should define the simplest version of the ideal model it is expecting
- This should reduce the need for the design system to perform heavy/complex processing

```
{
  "social_media": [
    {
            "name":"Twitter",
            "message":"Follow us on Twitter",
            "image":"/images/social_media_icons/social_twitter.png",
            "link":"http://www.twitter.com"
    },
    {
            "name":"Facebook",
            "message":"Like us on Facebook",
            "image":"/images/social_media_icons/social_facebook.png",
            "link":"http://www.facebook.com"
    },
    {
            "name":"LinkedIn",
            "message":"Follow us on LinkedIn",
            "image":"/images/social_media_icons/social_linkedin.png",
            "link":"http://www.linkedIn.com"
    }
  ]
}
```

```twig
<ul class="social_media--list">
  {% for social_media__item in social_media %}
    {% include "@atoms/social-icon.twig" with {
      "name":social_media__item.name,
      "message":social_media__item.message,
      "image":social_media__item.image,
      "link":social_media__item.link
    } %}
  {% endfor %}
</ul>
```

LEAST RESPONSIBILITY PRINCIPLE

- The design system owns the ideal model
- The responsibility for processing belongs to the content management system

- Every content management system needs to map and transform the data and architecture into the expected format

```
{% include "@molecules/list-element.twig" with {
"title": fields.title.content,
"link_url": fields.title.url,
"body": fields.body.content,
"tag_list": fields.field_tags.content,
"posted_date": fields.created.content
} %}
```

UNDERSTAND PATTERN VARIATIONS

- Patterns have attributes that are defined in the data
- Data is often thought of as content in the markup
- Data can also represent metadata or configuration
- Example: Red and Blue for the same image

card.twig

```
{# Generic version of card component that can be extended to fill in head,
body and CTA content #}
<div class="m-card {{ modifier }}">
  <div class="m-card__callout" data-equalizer-watch>
  {# Only add an href if it is pointing to a valid path #}
  {% if href %}
    <a class="m-card__detaillink" href="{{ href }}">
  {% else %}
    <a class="m-card__detaillink">
  {% endif %}
      <div class="m-card__head row">
```

card-provider.twig

```
{% embed "@molecules/Cards/card.twig"
  with {
    "modifier": "m-card-provider",
    "href": provider.profile_href,
  }
%}
```

_card-provider.scss

```
// _card_provider.scss - overrides for provider version of card

.m-card-provider {
  margin-bottom: $spacing-xl;
  // Disable default red arrow for component so we can use expanding version.
  .m-card__detaillink::after {
    display: none;
  }
```

ADHERE TO STANDARDS

- Atomic Design
- BEM
- SMACSS
- Drupal / CSS / JS framework standards

SECTION 3:

IMPLEMENTATION IDEAS

DIFFERING APPROACHES

- Brian
  - Focus on ease of component integration
  - Open to help from contrib modules
- Adam
  - Focus on platform agnostic approach
  - Favors functionality in Drupal core

PACKAGING, RELEASING, CHANGE MANAGEMENT

- Create releases of the design system
- Leverage Gulp/Grunt to build one CSS file and multiple JS files
- Leverage Composer to bring in new releases of the design system into CMS
- Map design assets in the theme
- Remediate and launch CMS changes

MANAGING THE DESIGN SYSTEM DEPENDENCY

- In Design System:
    - Add composer.json with “name”:”(orgname)/(repository_name)”
- In Drupal project:
    - Add repositories entry in composer.json referencing design system repo.
    - Configure installer paths to install in docroot/libraries
    - Composer require your design system

composer.json

This configuration file contains list of Composer dependencies

```
{
  "name": "hs2studio/hs2solutionsdesignsystem",
  "description": "HS2 Solutions Design System",
  "type": "design-system"
}
```

(design system composer.json)

composer.json

This configuration file contains list of Composer dependencies Install Updat

```
{
    "name": "hs2studio/drupal-project-docroot",
    "repositories": [
        {
            "url": "git@bitbucket.org:hs2studio/hs2solutionsdesignsystem.git",
            "type": "git"
        }
    ],
    "require": {
        "hs2studio/hs2solutionsdesignsystem": "^0.2.0"
    },
    "extra": {
        "installer-paths": {
            "docroot/libraries/{$name}": ["type:drupal-library", "type:design-system"]
        }
    }
}
```

(relevant portions from project composer.json)

IMPLEMENTING PATTERNS IN THE CMS

- Patterns are not a one-to-one mapping in the CMS
- Flexibility to use any pattern in any way
- Provide well defined approach to handling design system overrides
- Some patterns are not implemented in a CMS
- Pages are often just mockups and representation of pattern usage

MUST USE CORE SPONSORED TOOLS

- Custom block types
- Layout Discovery / Layout builder
- Views
- Content types
- View modes

MUST USE CONTRIBUTED MODULES

- Components module
- Block Type Templates module

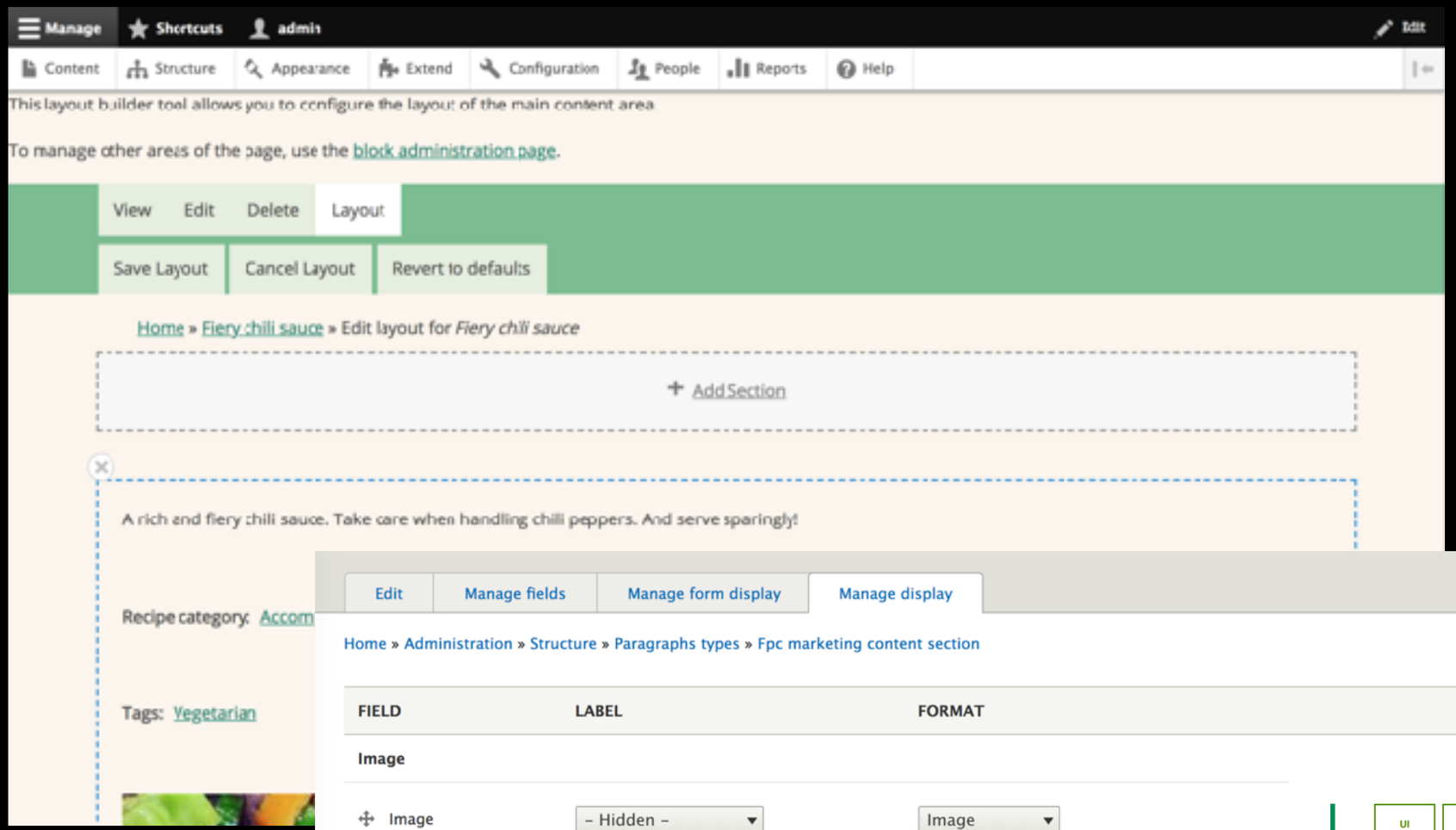
OPTIONAL CONTRIBUTED MODULES

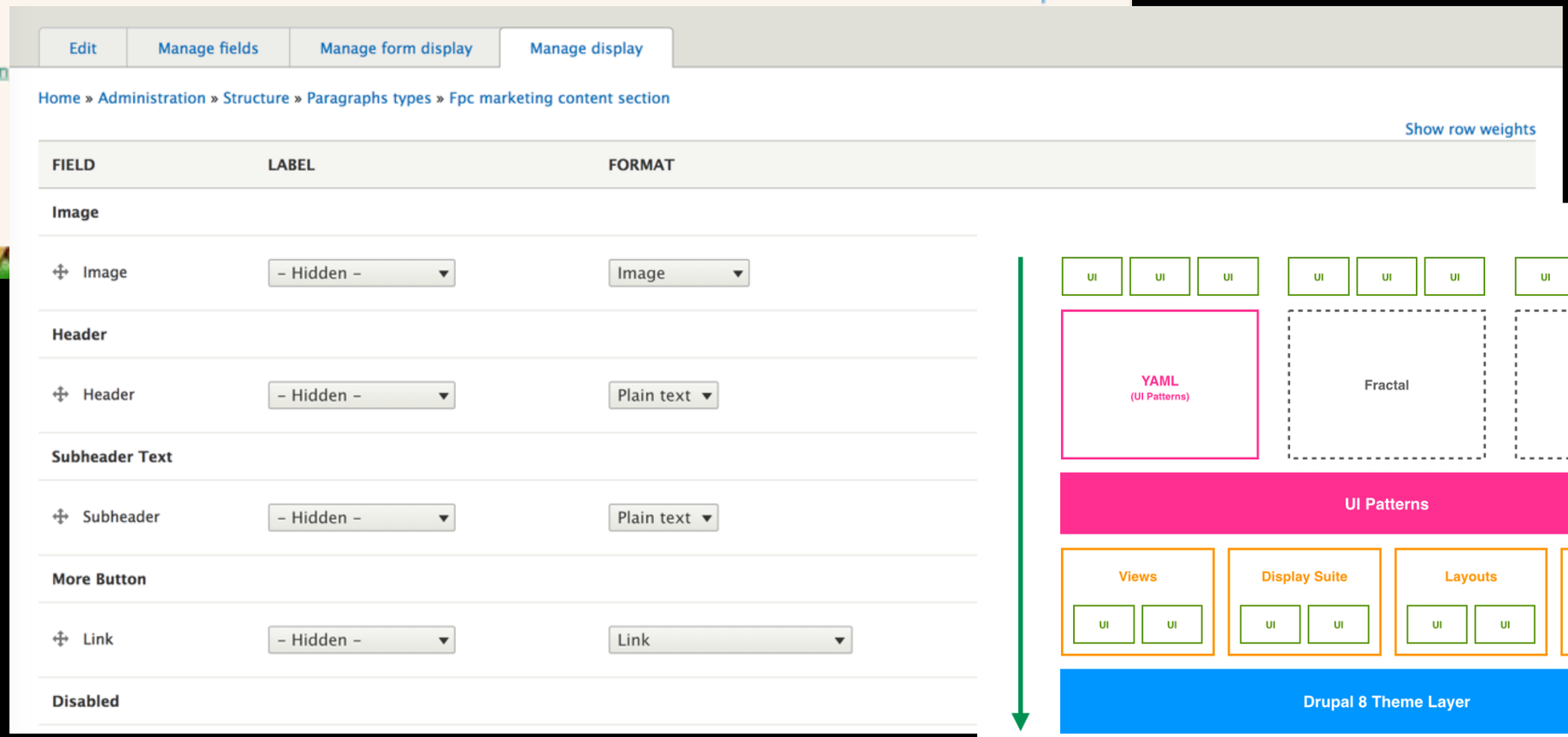
- Paragraphs module
- UI Patterns module
- Display Suite module

SECTION 4:

EMERGING CONCEPTS

DATA MAPPING IN ADMIN UI





AUTOMATIC PATTERN DISCOVERY

UI Patterns + Fractal

View Version control Automated testing

Posted by tonystar on *16 January 2018*, updated *11 May 2018*

This project is not covered by Drupal'

Integration of Fractal patterns as UI Patterns p

Fractal is a tool to help you build and docume
component libraries, and then integrate them
projects. https://fractal.build/

Ui Patterns defines and exposes self-containe
as Drupal plugins and generates a pattern libr
be used as documentation.
https://www.drupal.org/project/ui_patterns

You may also be interested about a Pattern La
https://www.drupal.org/project/ui_patterns_p

Supporting organizations:
WONDROUS

UI Patterns Pattern Lab

View Version control Automated testing

Posted by brianperry on *1 May 2018*, updated *7 May 2018*

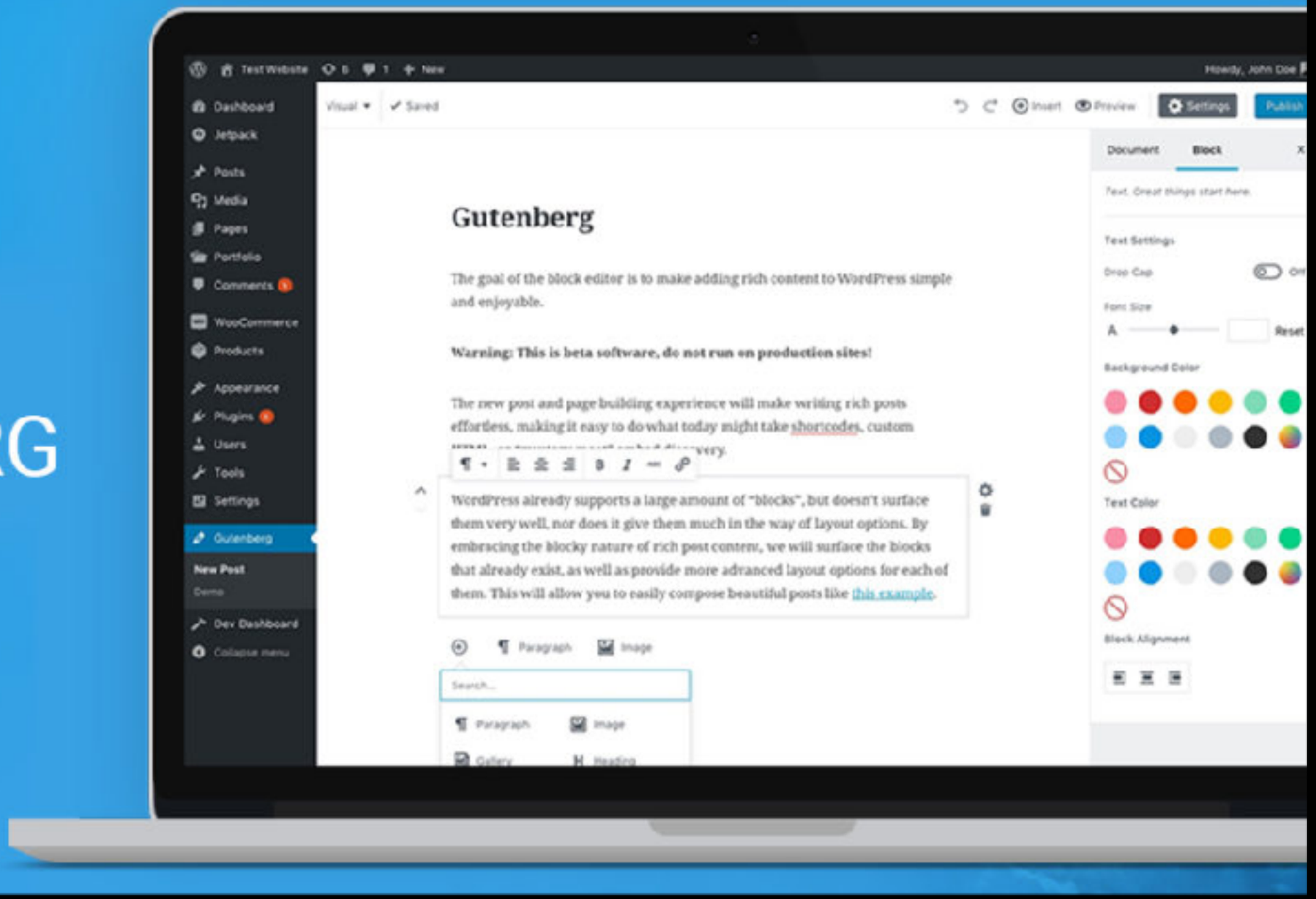
This project is not covered by Drupal's security advisory policy.

The UI Patterns Pattern Lab module automatically discovers patterns defined in a Pattern Lab instance and makes them available to be used in Drupal as UI Patterns.

This module will recognize Pattern Lab patterns in any active module or theme's /templates directory, along with any paths defined as Twig Namespaces in your theme by the Component Libraries module. After enabling this module (which will also enable the dependencies ui_patterns and ui_patterns_library) and clearing your cache, patterns should be visible at /patterns and available to use with any of the UI Patterns integration modules.

This project would not exist without the work of Antonio De Marco who maintains the UI Patterns module and Pierre Dureau who created the UI Patterns Fractal integration that this project is based on.

GUTENBERG


Test Website
Howdy, John Doe
Dashboard
Jetpack
Posts
Media
Pages
Portfolio
Comments
WooCommerce
Products
Appearance
Plugins
Users
Tools
Settings
Gutenberg
New Post
Demo
Dev Dashboard
Collapse menu
Visual
Saved
Insert
Preview
Settings
Publish
Document
Block
Text Settings
Drop Cap
Font Size
Reset
Background Color
Text Color
Block Alignment
Gutenberg
The goal of the block editor is to make adding rich content to WordPress simple and enjoyable.
Warning: This is beta software, do not run on production sites!
The new post and page building experience will make writing rich posts effortless, making it easy to do what today might take shortcodes, custom
WordPress already supports a large amount of "blocks", but doesn't surface them very well, nor does it give them much in the way of layout options. By embracing the blocky nature of rich post content, we will surface the blocks that already exist, as well as provide more advanced layout options for each of them. This will allow you to easily compose beautiful posts like this example.
Paragraph
Image
Search...
Paragraph
Image
Gallery
Heading

THANK YOU!

(Questions?)