



# An overview of front-end component integration methods in Drupal 8

**Brian Perry**  
Lead Front End Developer  
**bounteous**



Slides and example repo: <http://bit.ly/component-int>



# Abide by our code of Conduct

All attendees, speakers, sponsors, and volunteers at our conference are required to agree with the following code of conduct.

If you need to report an incident, ask one of the volunteers in the registration area to contact us or call 321-396-2340.



Jordana Fung



Mike Anello



# BRIAN PERRY

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



**d.o: brianperry**

**twitter: bricomedy**

**github: backlineint**

**nintendo: wabrian**

**brianperryinteractive.com**

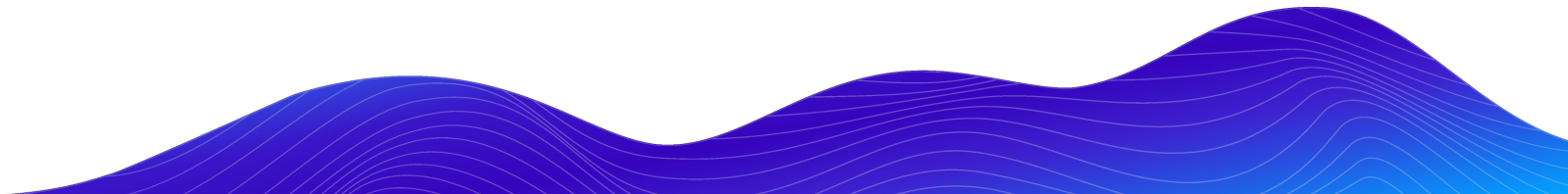
# bounteous



**VERSION 0.8.5**

This talk is **coming in hot...**

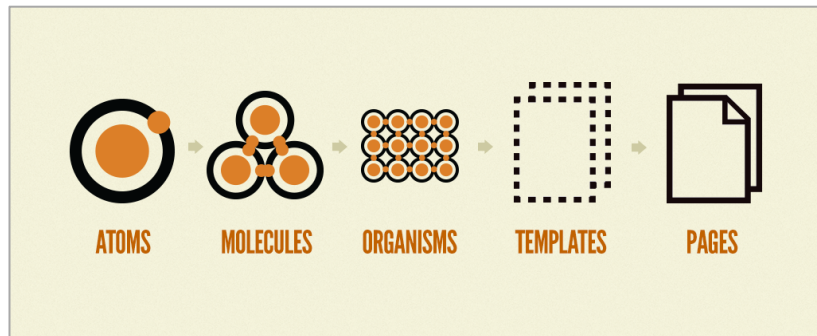
**COMPONENTS!**



# COMPONENT BASED THEMING

## What is it?

- Creating modular and re-usable elements
- Building a design system, not a series of pages
- Can use a pattern library for documentation and prototyping
  - Tools like Pattern Lab and Storybook

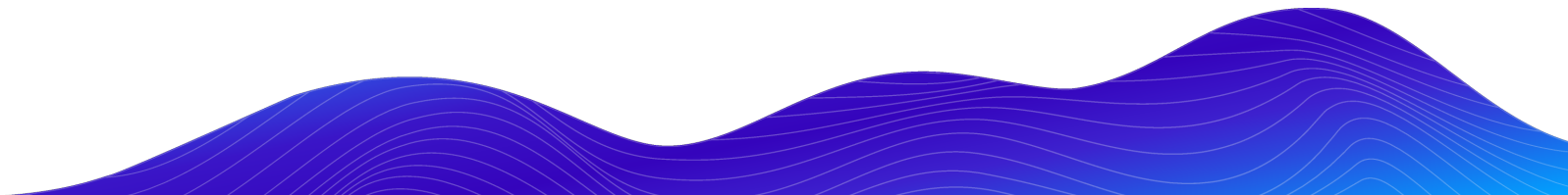


# COMPONENT BASED THEMING

## Why Take a component-based approach?

- Efficient re-use
  - Write once, use everywhere.
  - Within a single project and even across projects (beyond Drupal even)
- Well isolated chunks of code
- Decoupling front and back end development
  - Theming doesn't have to come last
- Living Style Guide / Pattern Library
  - Simplifies coordination between designers and developers / developers and developers.
  - Rapid prototyping
  - Design system source of truth

# OUR EXAMPLE COMPONENT



## Containers

### Container.is-centered

Good morning. Thou hast had a good night's sleep, I hope.

### Container.is-dark

Good morning. Thou hast had a good night's sleep, I hope.

Good morning. Thou hast had a good night's sleep, I hope.

Good morning. Thou hast had a good night's sleep, I hope.



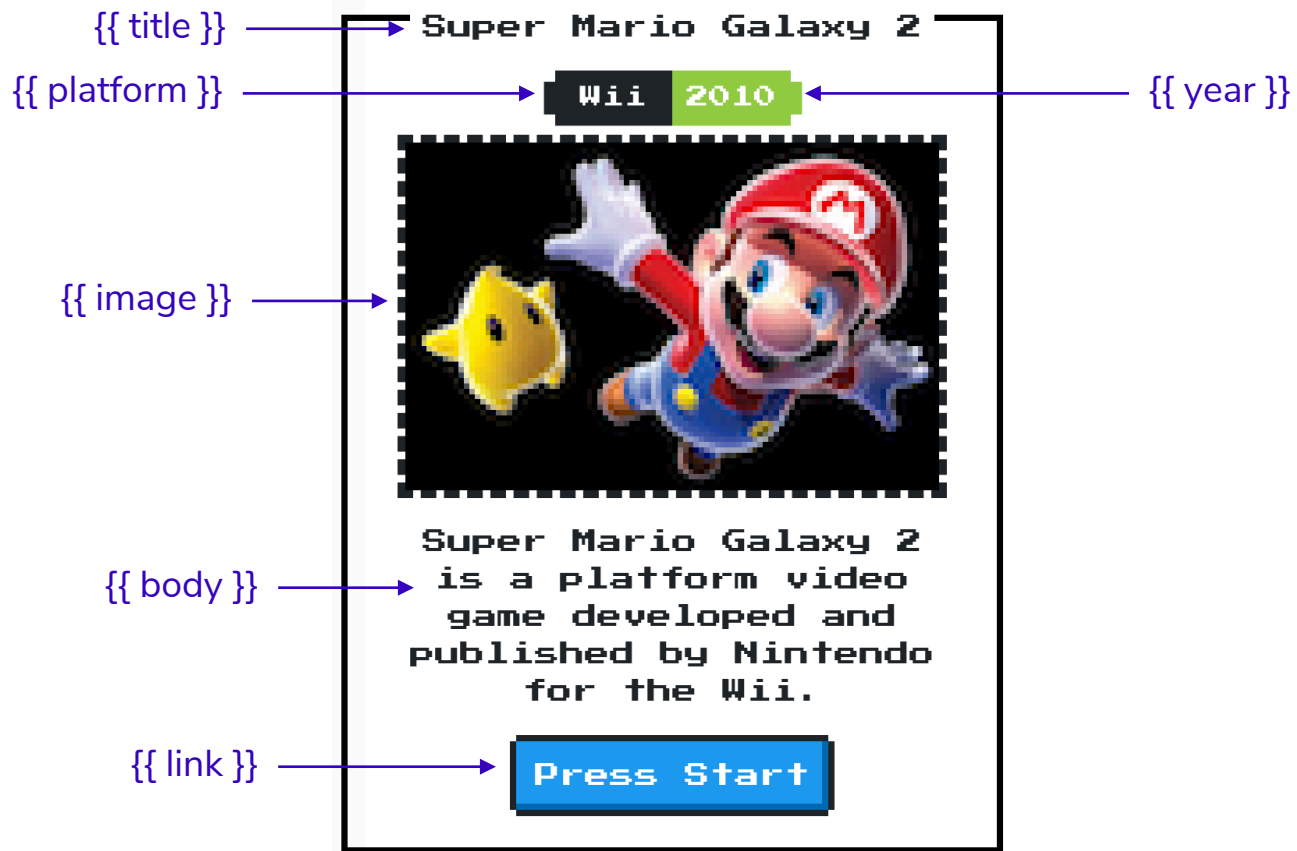
## Super Mario Galaxy 2

Wii 2010



Super Mario Galaxy 2 is a platform video game developed and published by Nintendo for the Wii.

Press Start



```

c-container.twig web/themes/custom/nes/nes-components/source/_patterns/03-components/container/c-container
<div class="nes-container with-title is-centered">
  {% if title %}
    <p class="title">{{ title }}</p>
  {% endif %}
  {% if platform or year %}
    <div class="nes-badge is-splited">
      {% if platform %}<span class="is-dark">{{ platform }}</span>{% endif %}
      {% if year %}<span class="is-success">{{ year }}</span>{% endif %}
    </div>
  {% endif %}
  {% if image %}
    {{ image }}
  {% endif %}
  {% if body %}
    {{ body }}
  {% endif %}
  {% if link %}
    <a class="nes-btn is-primary" href={{ link }}>Press Start</a>
  {% endif %}
</div>

```

## Super Mario Galaxy 2

Wii 2010



Super Mario Galaxy 2  
is a platform video  
game developed and  
published by Nintendo  
for the Wii.

Press Start

```
! c-container.yml web/themes/custom/nes/nes-components/source/_patterns/03-components/c
title: Container
platform: Wii
year: 2010
image: >
  
body: <p>Super Mario Galaxy 2 is a platform video game developed and
link: '#'
```

## Super Mario Galaxy 2

Wii 2010



Super Mario Galaxy 2  
is a platform video  
game developed and  
published by Nintendo  
for the Wii.

Press Start

# GRID LAYOUT



Component Integration Sandbox

## Games

### Mega Man 3

NES 1990



Mega Man 3 is an action-platform video game developed and published by Capcom.

Press Start

### Metroid Prime

GC 2002



Metroid Prime is the fifth main installment in the Metroid series, and the first Metroid game played from the first-person perspective.

Press Start

### Super Mario Galaxy 2

Wii 2010

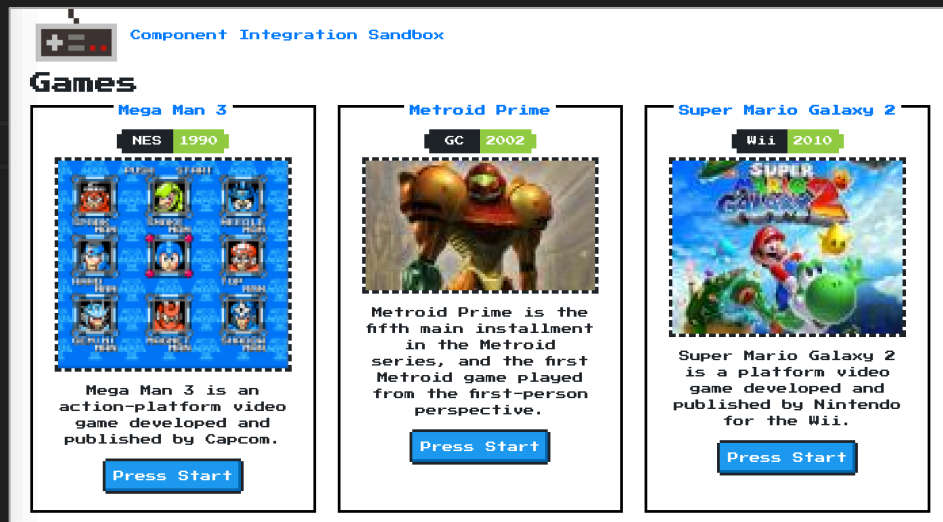


Super Mario Galaxy 2 is a platform video game developed and published by Nintendo for the Wii.

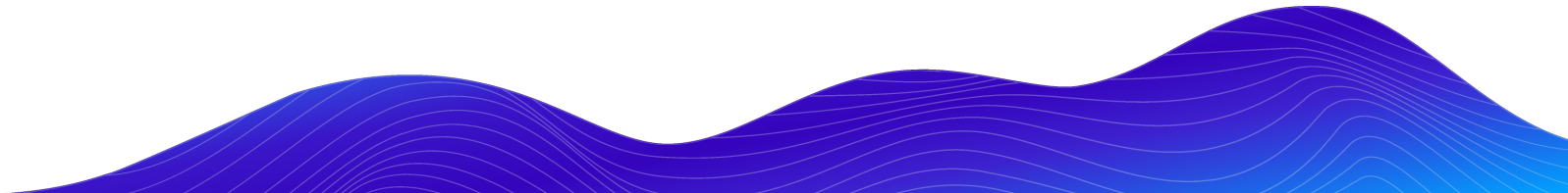
Press Start

```
# l-grid.css web/themes/custom/nes/nes-components/source/css/l-grid.css/...
```

```
.l-grid {  
  display: grid;  
  grid-column-gap: 2rem;  
  grid-row-gap: 2rem;  
}  
  
@media screen and (min-width: 768px) {  
  .l-grid {  
    grid-template-columns: repeat(2, 1fr);  
  }  
}  
  
@media screen and (min-width: 1024px) {  
  .l-grid {  
    grid-template-columns: repeat(3, 1fr);  
  }  
}
```



# COMPONENTS IN DRUPAL



# WHERE DO MY COMPONENTS LIVE?

For the sake of this talk...

## Standard Drupal Components

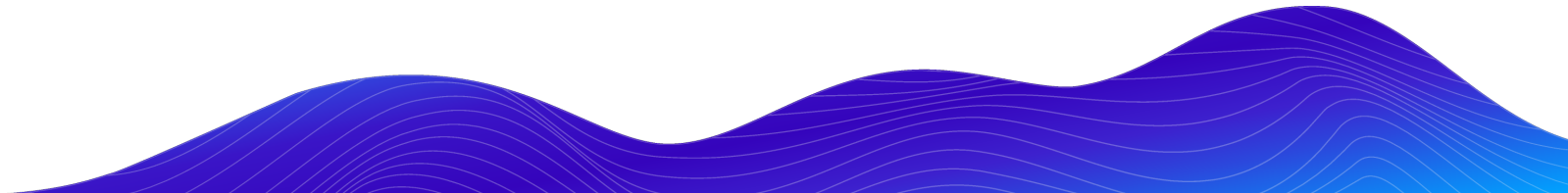
- Live in the default template directory
- May not require any additional effort to get data to display

## Integrated Drupal Components

- Live somewhere other than the default templates directory
- Require some additional effort to get data to display
- For this talk, I don't really care how your integrated components get into your theme.
  - Could live in your theme
  - Could be external dependency

# STANDARD DRUPAL COMPONENTS

Building components that live in the traditional templates directory

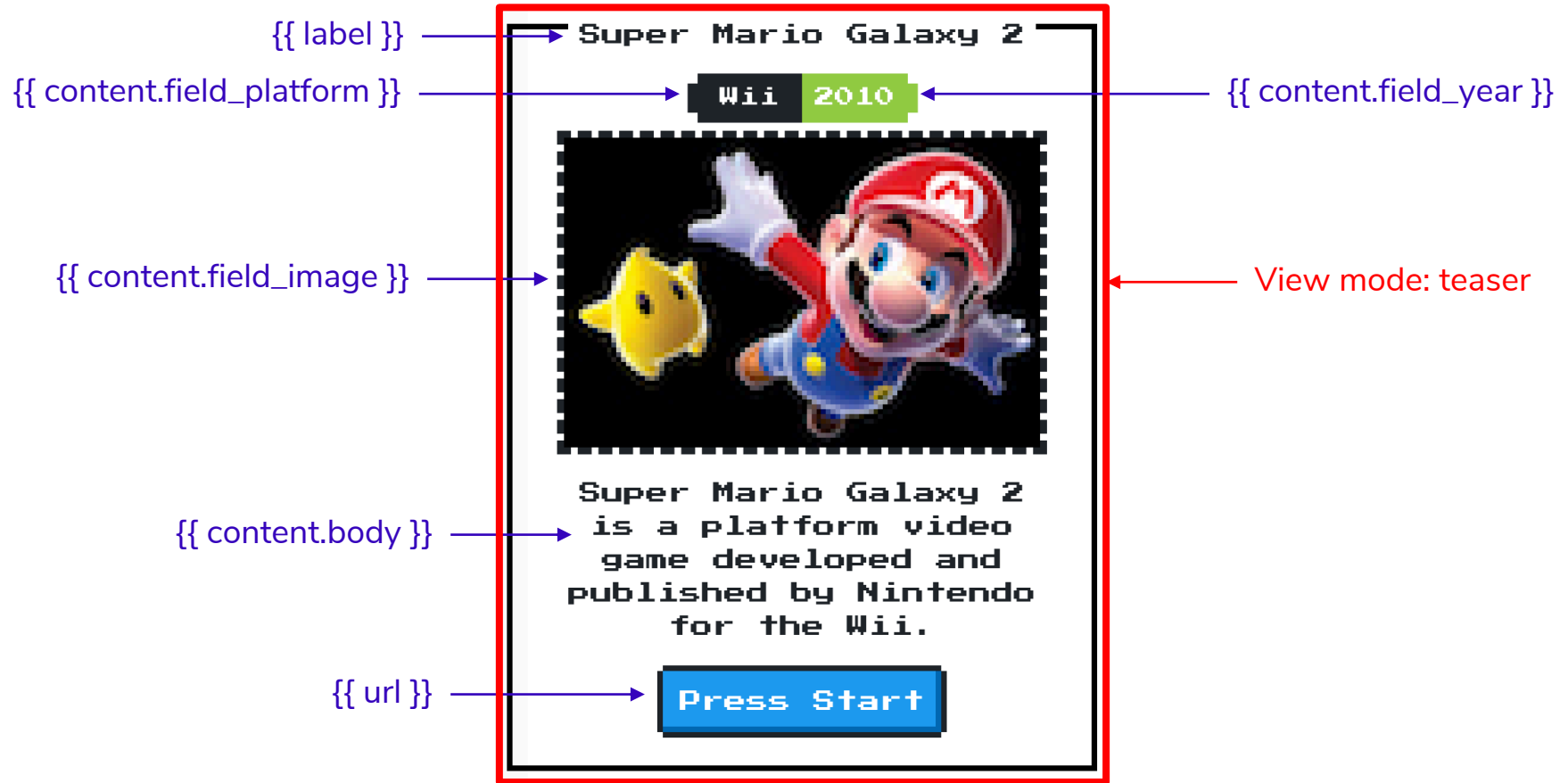


# STANDARD DRUPAL COMPONENTS

May be right for your team or project. No shame necessary.

- Build with Drupal (and only Drupal) in mind.
- Take advantage of things that can be re-used in Drupal
  - Display modes
  - Blocks
  - Paragraphs
  - Layouts
- Lose out on rapid prototyping advantages.

# STANDARD DRUPAL COMPONENT





```
<article{{ attributes.addClass(classes) }}>
```

```
  {{ title_prefix }}
```

```
  {% if label and not page %}
```

```
    <h2{{ title_attributes.addClass('title') }}>
```

```
      <a href="{{ url }}" rel="bookmark">{{ label }}</a>
```

```
    </h2>
```

```
  {% endif %}
```

```
  {{ title_suffix }}
```

```
<div{{ content_attributes.addClass('node__content') }}>
```

```
  {% if content.field_platform or content.field_year %}
```

```
    <div class="nes-badge is-splited">
```

```
      {% if content.field_platform %}<span class="is-dark">{{ content.field_platfo
```

```
      {% if content.field_year %}<span class="is-success">{{ content.field_year }}
```

```
    </div>
```

```
  {% endif %}
```


```
  {{ content|without('field_platform', 'field_year') }}
```

```
  <a class="nes-btn is-primary" href="{{ url }}" rel="bookmark">Press Start</a>
```

```
</div>
```

```
</article>
```


# GRID LAYOUT

 Component Integration Sandbox

## Games

### Mega Man 3

NES 1990




Mega Man 3 is an action-platform video game developed and published by Capcom.

Press Start

### Metroid Prime

GC 2002




Metroid Prime is the fifth main installment in the Metroid series, and the first Metroid game played from the first-person perspective.

Press Start

### Super Mario Galaxy 2

Wii 2010



Super Mario Galaxy 2 is a platform video game developed and published by Nintendo for the Wii.

Press Start

View: Games

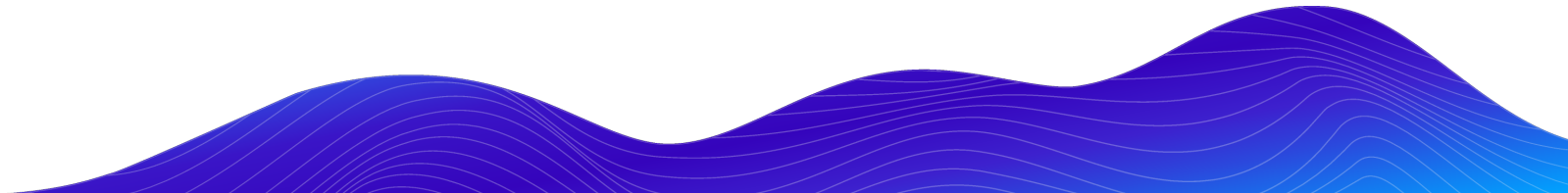
```

#}
<div class="l-grid">
  {% if title %}
    <h3>{{ title }}</h3>
  {% endif %}
  {% for row in rows %}
    {%
      set row_classes = [
        default_row_class ? 'views-row',
      ]
    %}
    <div{{ row.attributes.addClass(row_classes) }}>
      {{- row.content -}}
    </div>
  {% endfor %}
</div>

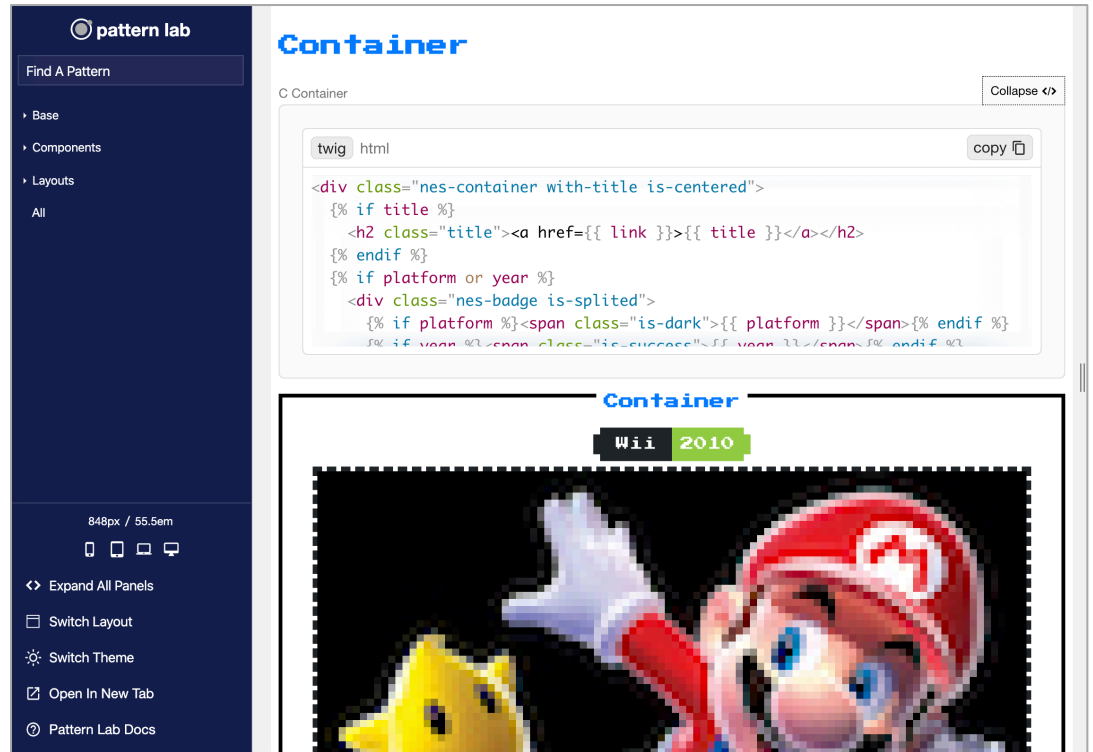
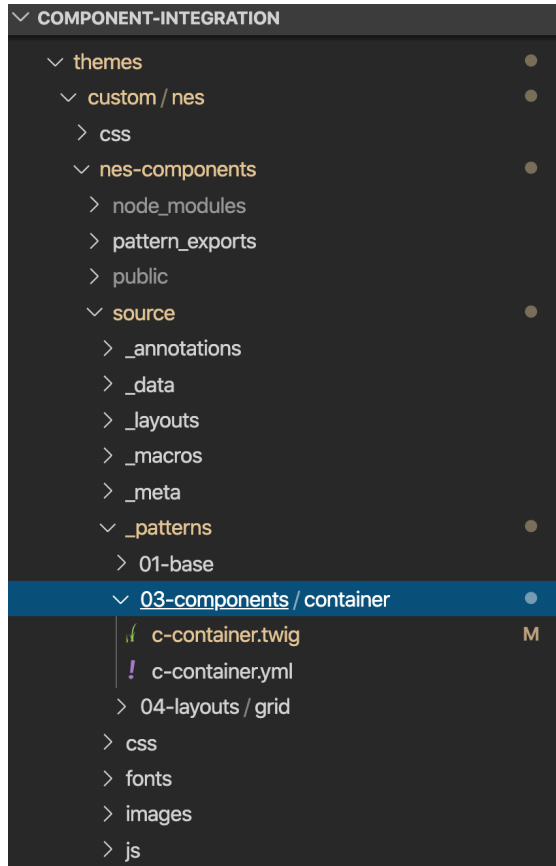
```

# INTEGRATED DRUPAL COMPONENTS

Building components that live outside of the traditional templates directory



# COMPONENT LIBRARY / PATTERN LAB



# COMPONENTS MODULE

Creates Twig namespaces to access templates in non-standard locations

! *nes.info.yml* *web/themes/custom/nes/nes.info.yml*

```
component-libraries:
  components:
    paths:
      - nes-components/source/_patterns/03-components
```

# INTEGRATION APPROACHES

Primarily fall into two categories

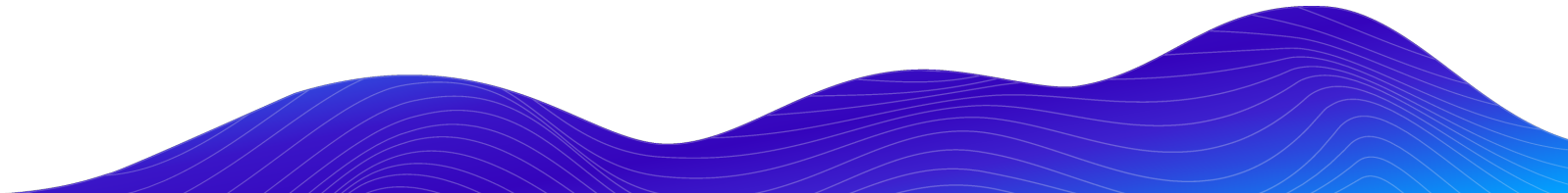
## Mapping Data In Code

- Includes:
  - Mapping in Twig templates
  - Preprocessing
  - Themes and starter kits
- More likely to get out of sync with Drupal UI
- More likely to break things like caching, Drupal functionality, etc.

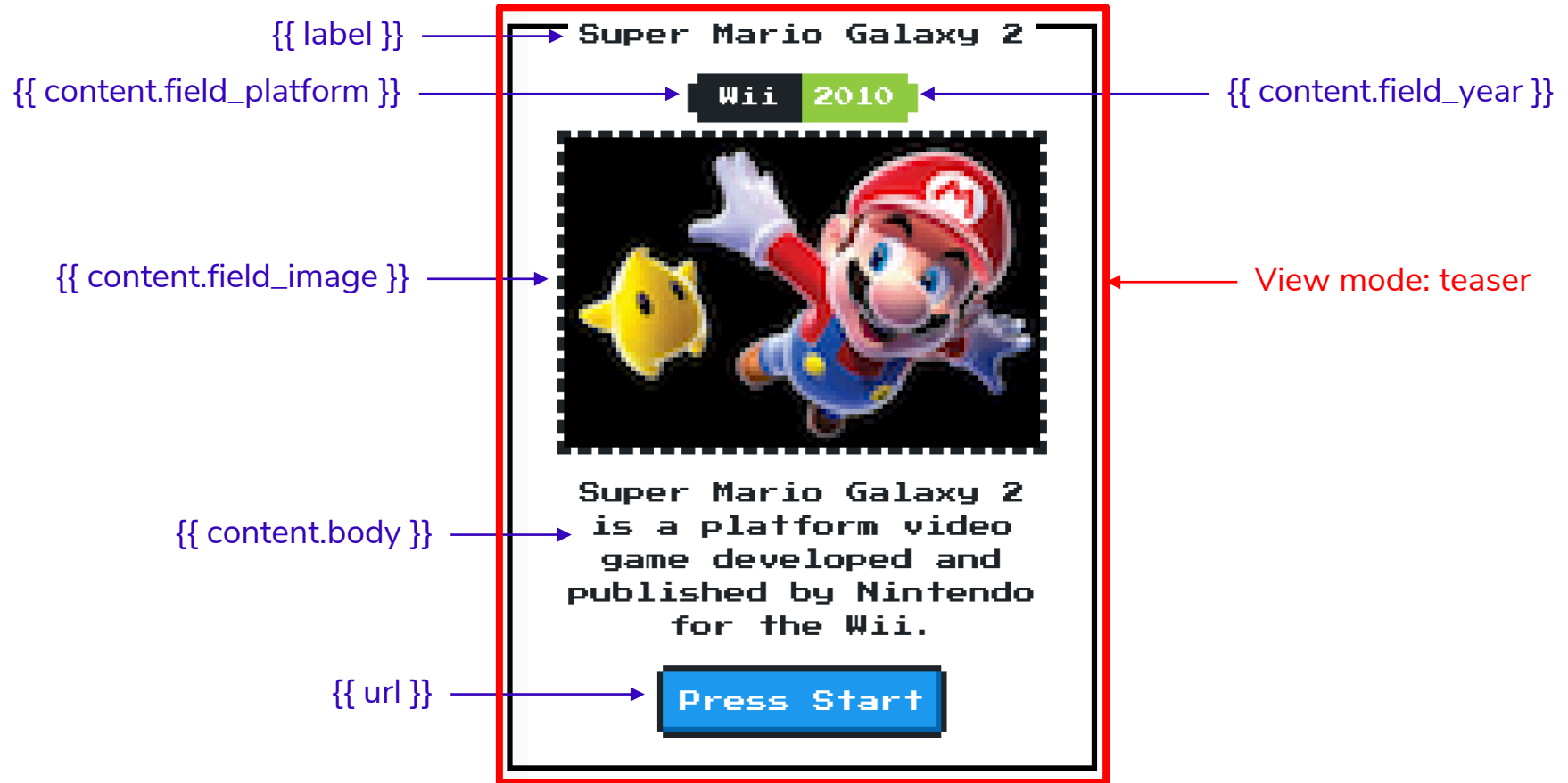
## Mapping Data In Admin UI

- Includes:
  - UI Patterns
  - Layouts
- Less likely to disrupt Drupal functionality
- Potentially not as flexible

# MAPPING DATA IN CODE



# INTEGRATING IN CODE



# MAPPING IN TWIG PRESENTER TEMPLATE

Drupal template references template in component library

node--game--teaser.html.twig web/themes/custom/nes/templates/node--game--teaser.html.twig

```
<article{{ attributes.addClass(classes) }}>
  {% include "@components/container/c-container.twig"
    with {
      'title': label,
      'link': url,
      'platform': content.field_platform,
      'year': content.field_year,
      'image': content.field_image,
      'body': content.body
    }
  %}
</article>
```

# SIDE NOTE: FIELDS VS FIELD ELEMENTS

```
1  {{ attach_library('foundation_patterns/marketing-site-content-section') }}
2
3  {% include "@marketing/content-section/marketing-site-content-section.twig"
4    with {
5      "image": content.field_fpc_image,
6      "header": content.field_fpc_header,
7      "subheader": content.field_fpc_subheader,
8      "button": content.field_fpc_link
9    }
10  %}
```

Drupal\Core\Template\Attribute (1)

contents Available methods (17) Iterator contents (1)

protected storage -> array (1)

'class' => Drupal\Core\Template\AttributeArray (2)

contents Available methods (10) Static class properties


protected value -> array (2)

- string (13) "node\_\_content"
- string (8) "clearfix"
- protected name -> string (5) "class"

```
{% include "@molecules/card/card.twig"
with {
  "img_src": file_url(content.field_image.entity.fileuri),
  "img_alt": content.field_image.0['#item'].alt,
  "header": label,
  "copy": content.body
}
%}
```

# DATA MAPPING IN PREPROCESS

Map in preprocess...

 nes.theme web/themes/custom/nest/nest.theme

```
/**
 * Implements hook_preprocess_node().
 */
function nes_preprocess_node__game(array &$variables) {
  $variables['title'] = $variables['label'];
  $variables['link'] = $variables['url'];
  $variables['platform'] = $variables['content']['field_platform'];
  $variables['year'] = $variables['content']['field_year'];
  $variables['image'] = $variables['content']['field_image'];
  $variables['body'] = $variables['content']['body'];
}
```

# DATA MAPPING IN PREPROCESS

...and use simpler include in Twig presenter template

```
✓ node--game--teaser.html.twig web/themes/custom/nes/templates/node--game--teaser.html.twig
{
  {
    {%
      set classes = [
        'node',
        'node--type-' ~ node.bundle|clean_class,
        node.isPromoted() ? 'node--promoted',
        node.isSticky() ? 'node--sticky',
        not node.isPublished() ? 'node--unpublished',
        view_mode ? 'node--view-mode-' ~ view_mode|clean_class,
      ]
    %}
    {{ attach_library('classy/node') }}
    <article{{ attributes.addClass(classes) }}>
    {
      {% include "@components/container/c-container.twig" %}
    }
    </article>
  }
}
```

# HELPER MODULES

## Simplify Twig Mapping

### Twig Field Value

- Get Partial data from field render arrays
  - field\_label
  - field\_value
  - field\_raw
  - field\_target\_entity
- Map just the data you want
- May require additional caching considerations...

### Twig Tweak

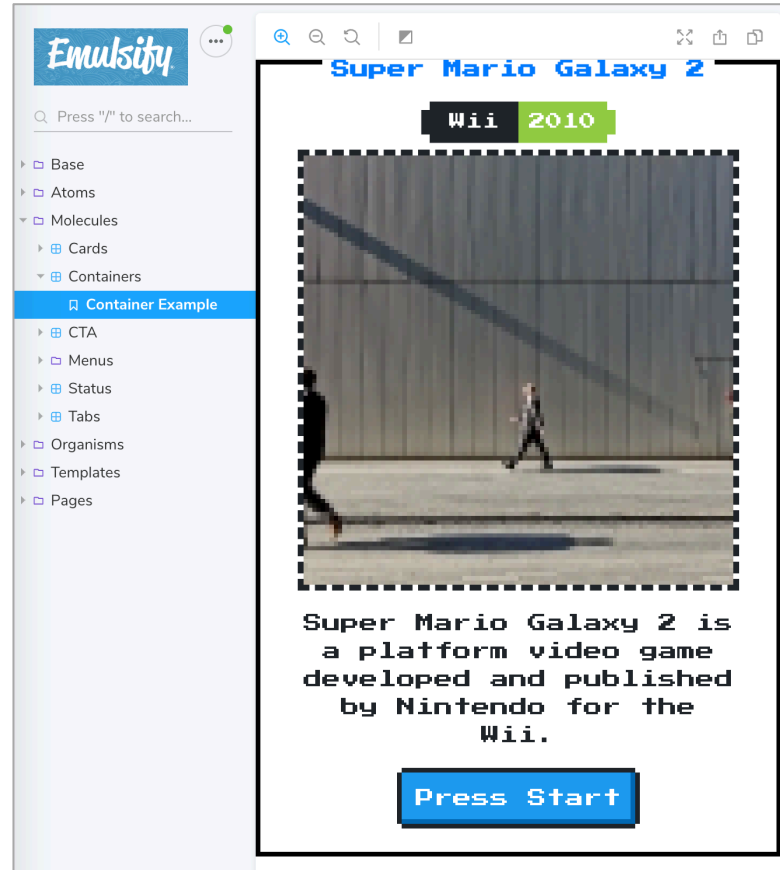
- Helpful twig functions and filters
  - Render views, blocks, regions, fields, entities and so on.
  - Render image with specific image style
  - Extract tokens from context

# STARTER KITS AND THEMES

- Simplify set up and provide default tooling
- Some provide default components and helper functions
- Various levels of opinionated
- Examples:
  - Emulsify
  - Gesso
  - Shila
  - Particle

# EMULSIFY DESIGN SYSTEM EXAMPLE

- Same presenter templates
- Different component location
- Different component library tool



JS container.stories.js web/themes/custom/nes\_emulsify/components/02-molecules/container/container.stories.js/...



```
import React from 'react';

import container from './c-container.twig';

import './nes.min.css';
import './fonts.css';

import containerData from './c-container.yml';

/**
 * Storybook Definition.
 */
export default { title: 'Molecules/Containers' };

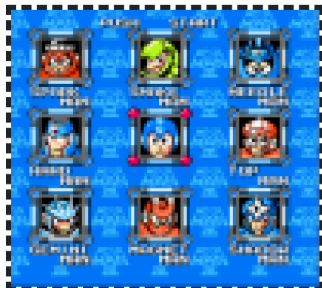
export const containerExample = () =>
  <div dangerouslySetInnerHTML={{ __html: container(containerData) }} />;
```



## Games

### Mega Man 3

NES 1990



Mega Man 3 is an action-platform video game developed and published by Capcom.

Press Start

### Metroid Prime

GC 2002



Metroid Prime is the fifth main installment in the Metroid series, and the first Metroid game played from the first-person perspective.

Press Start

### Super Mario Galaxy 2

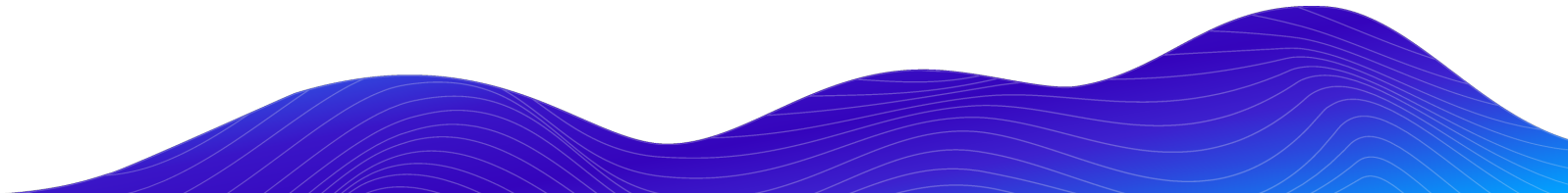
Wii 2010



Super Mario Galaxy 2 is a platform video game developed and published by Nintendo for the Wii.

Press Start

# MAPPING DATA IN THE ADMIN UI



# UI PATTERNS MODULE

Define and manage components in a way that Drupal understands

- Define UI Patterns as Drupal Plugins
- Use defined patterns with component friendly modules
  - Views, field groups, layout builder, display suite, paragraphs (requires field layout or display suite)
- Configure data mappings in the UI
- Optional Pattern Library page exposed in Drupal
- Also allows Drupal to:
  - Preprocess patterns
  - Render patterns programmatically



```
container:
  label: Container
  description: A container component from NES.css
  fields:
    title:
      type: text
      label: Title
      description: Game title.
      preview: Super Mario Galaxy 2
    platform:
      type: text
      label: Platform
      description: Console or platform
      preview: Wii
    year:
      type: text
      label: Year
      description: Year of release
      preview: 2010
    image:
      type: image
      label: Image
      description: Box art
      preview: Super Mario Galaxy 2 is a platform video game developed and publis
    link:
      type: url
      label: Link
      description: link to node
      preview: '#'
```

[illegible]

# Pattern library

## Available patterns

[. Container](#)

### Container

A container component from NES.css

Field	Label	Type	Description
title	Title	text	Game title.
platform	Platform	text	Console or platform
year	Year	text	Year of release
image	Image	image	
body	body	text	Body text
link	Link	url	link to node

### Preview

[Super Mario Galaxy 2](#)

Wii 2010



# UI PATTERNS VIEWS

## Games (Content) ☆

[Home](#) » [Administration](#) » [Structure](#) » [Views](#)

### Displays

Page

+ Add

Edit view name/description ▼

Display name: [Page](#)

View Page ▼

TITLE

Title: [Games](#)

FORMAT

Format: [Unformatted list](#) | [Settings](#)

Show: [Pattern](#) | [Settings](#)

FIELDS

Content: [Title](#)

Content: [Body](#)

Content: [Image](#)

Content: [Link to Content](#)

Content: [Platform](#)

Content: [Year](#)

PAGE SETTINGS

Path: [/games](#)

Menu: [No menu](#)

Access: [Permission](#) | [View published content](#)

HEADER

Add

FOOTER

Add

NO RESULTS BEHAVIOR

Add

PAGER

Use pager: [Mini](#) | [Mini pager, 10 items](#)

More link: [No](#)

► ADVANCED







bounteous

47

- ☐ Provide default field wrapper elements
- If not checked, fields that are not configured to customize their HTML elements will get no wrappers at all for their field, label and fieldset. You can use this to quickly reduce the amount of markup the view provides by default, at the cost of making it more difficult to apply custom CSS.
- ☒ Hide empty fields
- Do not display fields, labels or markup for fields that are empty.

Pattern \*

Container ▾

SOURCE	PLUGIN	DESTINATION
 Content: Title	Views row	Title ▾
 Content: Body	Views row	body ▾
 Content: Image	Views row	Image ▾
 Content: Link to Content	Views row	Link ▾
 Content: Platform	Views row	Platform ▾
 Content: Year	Views row	Year ▾

Apply

Cancel



# LAYOUTS



# LAYOUTS AND LAYOUT BUILDER

! nes.layouts.yml web/themes/custom/nes/nes.layouts.yml

```
container:
  label: 'Container'
  category: 'NES'
  template: templates/container-layout
  default_region: body
  regions:
    title:
      label: Title
    platform:
      label: Platform
    year:
      label: Year
    image:
      label: Image
    body:
      label: Body
    link:
      label: Link
```

container-layout.html.twig web/themes/custom/nes/templates/container-layout.html.twig

```
{% include "@components/container/c-container.twig"
with {
  'title': content.title,
  'link': content.link,
  'platform': content.platform,
  'year': content.year,
  'image': content.image,
  'body': content.body
}
%}
```

# LAYOUTS AND LAYOUT BUILDER

```
c-container.twig web/themes/custom/nes/nes-components/source/_patterns/03-components/container/c-contain
<div {{ attributes.addClass('container') }}>
  <div class="nes-container with-title is-centered">
    {% if title %}
      <div {{ region_attributes.title.addClass('title-region') }}>
        <h2 class="title"><a href={{ link }}>{{ title }}</a></h2>
      </div>
    {% endif %}
    {% if platform or year %}
      <div class="nes-badge is-splited">
        {% if platform %}<span class="is-dark">{{ platform }}</span>{% endif %}
        {% if year %}<span class="is-success">{{ year }}</span>{% endif %}
      </div>
    {% endif %}
    {% if image %}
      <div {{ region_attributes.image.addClass('image-region') }}>
        {{ image }}
      </div>
    {% endif %}
    {% if body %}
      <div {{ region_attributes.body.addClass('body-region') }}>
        {{ body }}
      </div>
    {% endif %}
    {% if link %}
      <a class="nes-btn is-primary" href={{ link }}>Press Start</a>
    {% endif %}
  </div>
</div>
```

# LAYOUTS AND LAYOUT BUILDER

## Add 'container' section for teaser layout

Configure Container


**Title** field

+Add block

**Body** field

+Add block

+Add +Add block

 Choose a layout for this section



## One column



## Two column



## Three column



## Four column

# Container

# LAYOUTS AND LAYOUT BUILDER

## Challenges:

- For full layout builder functionality, need to retain container and region attributes
- Need to be able to addClass and attributes functions in Pattern Library
- Some components might not be practical for visual field mapping
- Layout Builder UI introduces additional markup that may conflict with your component

# COMPONENT DEFINITION APPROACHES

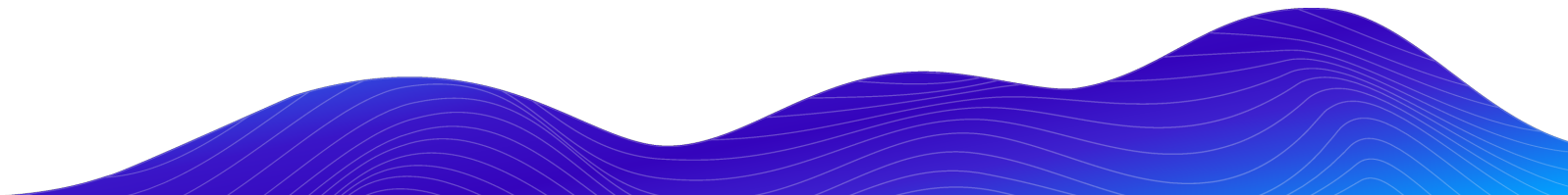
## Manual Definition

- Define component in code so that Drupal becomes aware of it.
- Likely requires some amount of duplication between Drupal and component library

## Automatic Discovery

- Drupal module automatically discovers components from component library and makes them available to Drupal.
- Emerging/experimental concept.
- Expects a particular convention and thus won't work with all component libraries.

# **AUTOMATIC DISCOVERY**



# UI PATTERNS PATTERN LAB


Automatically create UI Patterns from your pattern library... really.

- End result same as previous UI Patterns Example
- No redundant ui\_patterns.yml file necessary
- Some limitations
  - Requires yaml or json file with pattern data
  - Requires specific approach to nested components.

## 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.

UI

UI

UI

PatternLab

UI Patterns

# LAYOUTS FROM PATTERN LAB

Automatically create Drupal Layouts from your pattern library... really.

- No explicitly defined layout necessary
- Use with Layout Builder
- Can select default field wrappers or only content (similar to UI Patterns)
- Early - many limitations
  - Need to work around Pattern Lab 3 not working with html.twig extension
  - Open issues with Layout Builder drag and drop.

## Layouts from Pattern Lab

[View](#) [Edit](#) [Version control](#) [View history](#) [Maintainers](#) [Automated testing](#)

By [brianperry](#) on 8 December 2019, updated 8 December 2019



This project is not covered by Drupal's [security advisory policy](#).

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

### Supporting organizations:

[Bounteous](#)

### Project information

Module categories: [Content Display](#)



This project is not covered by the [security advisory policy](#).

**Use at your own risk!** It may have publicly disclosed vulnerabilities.

### Downloads

Development version: [8.x-1.x-dev](#) updated 13 Dec 2019 at 22:03 UTC

[View all releases](#)

[Add new release](#)

[Administer releases](#)

[Report as spam](#)

# PATTERNKIT

Combines aspects of manual definition and automatic discovery

- Requires creating schema definition file (which has potential applications outside of Drupal)
- Automatically derives blocks from pattern library components
- Supports a specific set of field types
- Token support in D7 but not yet D8

## Patternkit

[View](#) [Version control](#) [View history](#) [Automated testing](#)

By [cyb.tachyon](#) on 30 January 2018, updated 7 February 2020

This project is not covered by Drupal's [security advisory policy](#).

**Patternkit loads your templates/patterns into Drupal as blocks\*, where you can add them to your pages and layouts. The fields are read from a similarly-named JSON Schema file next to the twig template.**

This allows Drupal 8 Core's Layout Builder or anything that uses blocks to function as a full custom page builder app.

\*Panes in Drupal 7.

This module will parse a pattern library (local or through REST endpoints) to generate a list of blocks that can be drag/dropped into layouts. It currently supports directories full of Twig templates, and support for additional pattern types is planned. All you need to do is add a "patterns" section to your *{theme-or-module}.libraries.yml*, and a .json file written with JSON Schema (Draft 4) to power the editor.

**patternkit\_example.libraries.yml**

```
patternkit:  
  version: VERSION
```

# PATTERNKIT PATTERN DISCOVERY

! nes.libraries.yml web/themes/custom/nes/nes.libraries.yml

global-styling:

version: 1.x

css:

theme:

nes-components/source/css/nes.min.css: {}

nes-components/source/css/c-container.css: {}

nes-components/source/css/l-grid.css: {}

css/drupal.css: {}

<https://fonts.googleapis.com/css?family=Press+Start+2P>: { type: external }

patterns:

nes-components/source/\_patterns/03-components/container: {plugin: twig}

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "title": "Container",
  "description": "NES Container",
  "category": "pattern",
  "type": "object",
  "properties": {
    "title": {
      "title": "Title",
      "type": "string"
    },
    "platform": {
      "title": "Platform",
      "type": "string"
    },
    "year": {
      "title": "Year",
      "type": "string"
    },
    "body": {
      "title": "Body",
      "type": "string",
      "format": "textarea",
      "options": {
        "wysiwyg": true
      }
    },
    "link": {
      "title": "Link",
      "type": "string"
    },
    "settings": {
      "title": "Settings",
      "type": "object",
      "properties": {
        "alignment": {
          "title": "Item Alignment",
          "type": "string",
          "enum": [
```

## Configure block ☆

[Home](#) » [Administration](#) » [Structure](#) » [Block layout](#)

**Block description:** [Patternkit] Container

☐ Reusable

**Presentation style**

HTML inline ▼

Container ▼

 Properties

NES Container

Title

Super Mario Galaxy

Platform

Wii

Year

2010

Body

Super Mario Galaxy 2 is a platform video game developed and published by Nintendo for the Wii.

Link

#

Settings ▼

 Properties

Item Alignment

Default



Component Integration Sandbox

# Pattern Kit Example

[View](#)

[Edit](#)

[Delete](#)

[Revisions](#)

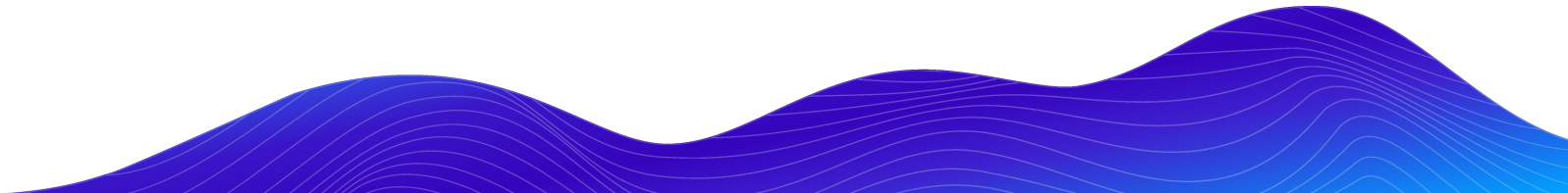
Super Mario Galaxy

Wii 2010

Super Mario Galaxy 2 is a platform video game developed and published by Nintendo for the Wii.

Press Start

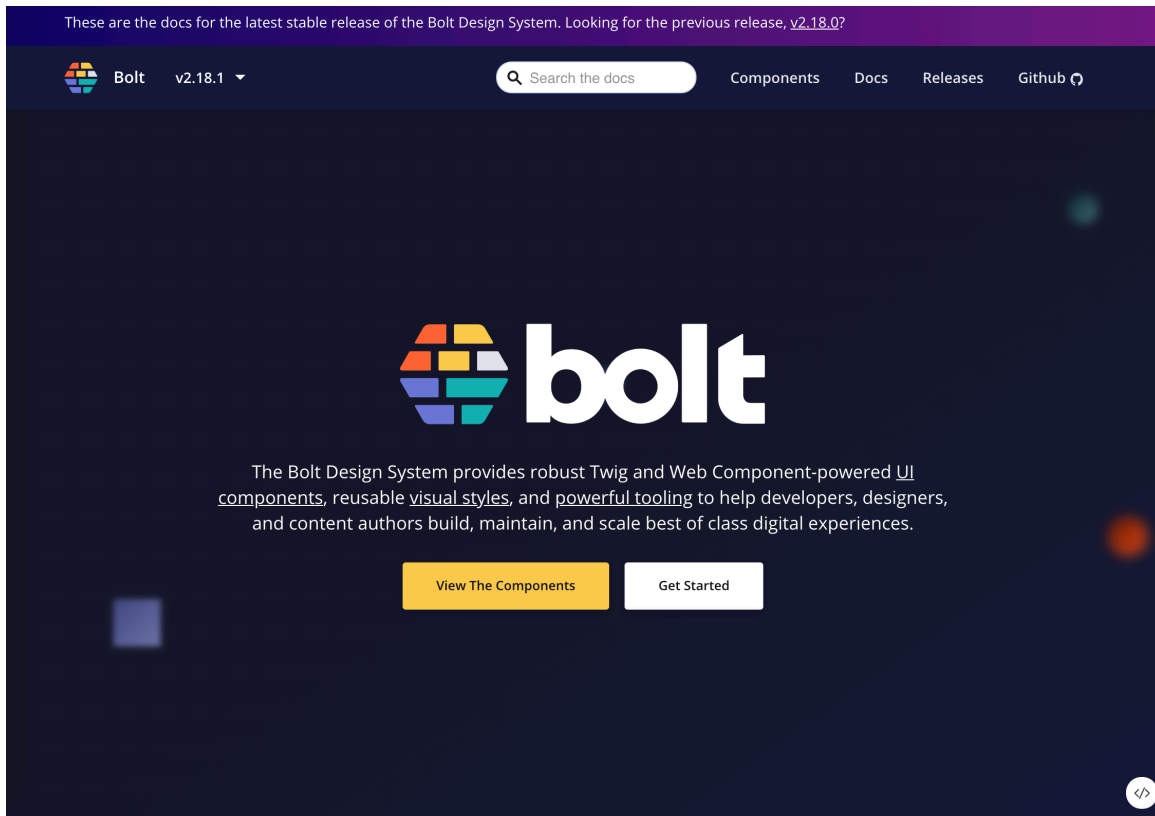
# **PRE-PACKAGED COMPONENT SOLUTIONS**



# BOLT DESIGN SYSTEM

Ready to use web-components

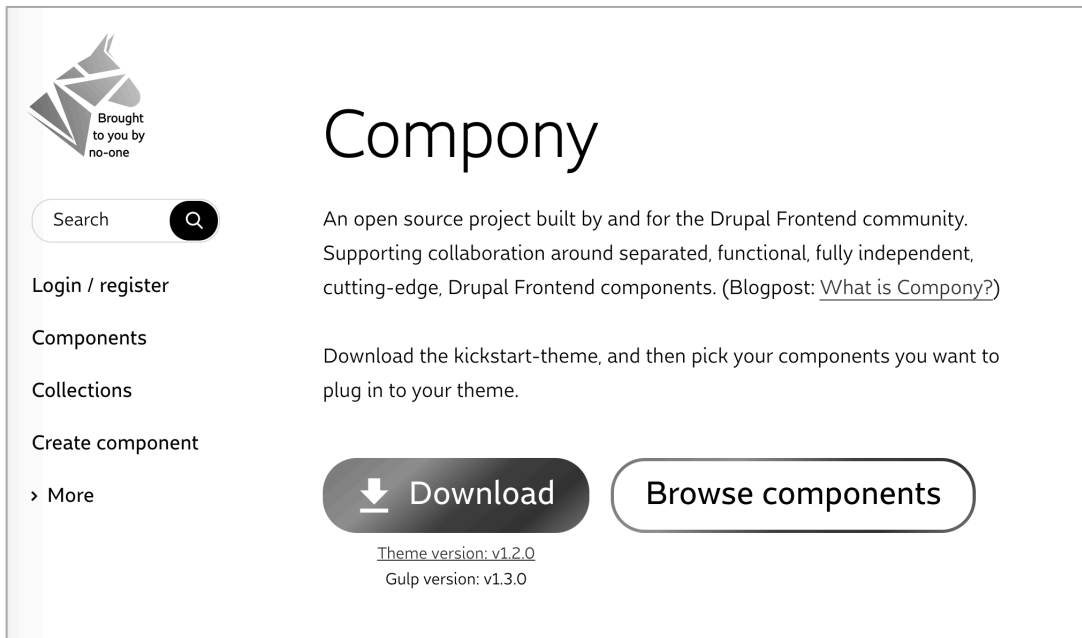
- Full design system
- Selectively require components.



# COMPONY

## Component distribution system

- Combines a theme, Gulp workflow and components.
- Download existing components or create your own
- Not Composer / NPM driven



The screenshot shows the Compony website. On the left is a sidebar with a logo that says 'Brought to you by no-one'. The sidebar contains links: 'Search' (with a magnifying glass icon), 'Login / register', 'Components', 'Collections', 'Create component', and '> More'. The main content area has the title 'Compony' in a large font. Below the title is a paragraph: 'An open source project built by and for the Drupal Frontend community. Supporting collaboration around separated, functional, fully independent, cutting-edge, Drupal Frontend components. (Blogpost: [What is Compony?](#))'. Below this is another paragraph: 'Download the kickstart-theme, and then pick your components you want to plug in to your theme.' At the bottom of the main area are two buttons: 'Download' (with a download icon) and 'Browse components'. Below the 'Download' button, it says 'Theme version: v1.2.0' and 'Gulp version: v1.3.0'.

# SINGLE FILE COMPONENTS

## Drupal components with Vue style syntax

- Use like any template
- Automatically generates library definitions
- Derive Blocks and Layouts with Annotations
- Provides component library
- Doesn't really solve integration problem
- Does help with distribution and re-use.

```
<?php

namespace Drupal\sfc_test\Plugin\SingleFileComponent;

use Drupal\sfc\ComponentBase;

/**
 * Contains an example single file component.
 *
 * @SingleFileComponent(
 *   id = "say_hello",
 * )
 */
class SayHello extends ComponentBase {

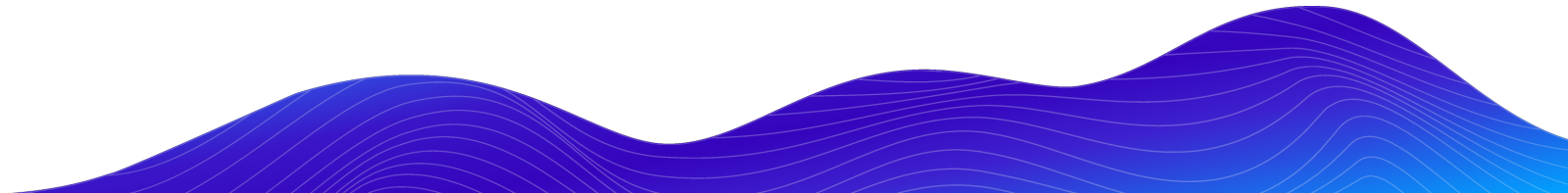
    const TEMPLATE = <<<TWIG
    <p class="say-hello">Hello {{ name }}!</p>
    TWIG;

    const CSS = <<<CSS
    .say-hello {
        color: pink;
    }
    CSS;

    /**
     * {@inheritdoc}
     */
    public function prepareContext(array &$context) {
        if (!isset($context['name'])) {
            $context['name'] = \Drupal::currentUser()->getDisplayName();
        }
    }
}
```

# COMPONENT WORKFLOW

Present and Future



# CURRENT APPROACH

## Pushing to represent components in Drupal UI

- Define component mapping in Drupal UI where possible
- Build components compatible with Layout Builder
  - Iterating on Layouts From Pattern Lab module.
- Falling back on Twig mapping approach when necessary
- Following Drupal's lead in some cases
  - Images
  - Navigation

# DREAM WORKFLOW

Basically React (or insert the name of your favorite JS framework here)

- Build fully packaged distributable components
- Easily install them
  - `npm install cool-component / composer require drupal/cool-component`
- Import them in code
  - `import 'CoolComponent' from 'cool-component'; / {% include '@components/cool-component.twig' %}`
- Use them as I see fit
  - `<CoolComponent />`

# HOW DO WE GET THERE?

We seem to have a lot of the pieces

- Continue to improve UI based component configuration process in Drupal
  - With specific focus on Layout Builder.
- Make it easier to package, distribute and use individual components
- Continue to evolve approaches allowing Drupal to automatically discover components
- Keep building amazing looking component based sites using Drupal

Thanks to the many **Drupal**  
**component ecosystem**  
**contributors!**



# Contribution Day

Sunday, February 23, 2020


1:15pm - 5:00pm



First-time contributor workshop • Mentored contribution • General contribution

#DrupalContributions

<https://www.fldrupal.camp/conference/contribution-day>

# WOULD LOVE YOUR FEEDBACK

**DrupalCon**  
MINNEAPOLIS 2020

DIAMOND SPONSORS  
 

My account Log out

Next event  
**MINNEAPOLIS 2020**  
MAY 18 - 22, 2020

Why DrupalCon? Program Registration Plan Your Trip Sponsor About

Register now

View Edit Evaluations

## An overview of Drupal 8 front-end component integration methods

**brianperry**

Since the release of Drupal 8, great strides have been made to develop a component-based theming workflow that takes advantage of the best that Twig has to offer and also plays nice with component libraries and design systems. Gone are the days of redundant styles and markup, making way for the efficiencies found when Drupal and tools like Pattern Lab or Storybook can share the exact same code. That said, handling the mapping of data between Drupal and your component library can still be quite complicated and difficult to coordinate on larger cross-functional teams.

This session will provide an overview of methods that can be used to provide data from Drupal to a front-end component that lives outside of the traditional Drupal templates directory, including:

- Mapping data via preprocessing
- Mapping data in twig templates
  - Helper modules including **Component Libraries**, **Twig Tweak**, and **Twig Field Value**
- Popular component-based themes and starter kits

**Track**  
Development & Coding

**Tags**  
Drupal 8  
front-end development  
theming

**Experience Level**  
Intermediate

**ADD TO MY SCHEDULE**

# Q&A

**Brian Perry**

Lead Front End Developer

**Email:** [brian.perry@bounteous.com](mailto:brian.perry@bounteous.com)

