



# Introduction to Drupal 8 Theme Development

# Introduction to Drupal 8 Theme Development



Jonathan Daggerhart

Technology Coordinator, Educational Partners International  
No. 2, Drupal Camp Asheville 2016



Drupal.org: daggerhart

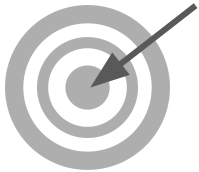
Twitter: @daggerhart





# Introduction to Drupal 8 Theme Development

## What we will cover



### Part 1

- Anatomy of theme file system
  - Creating your first theme
  - Twig templates, and where to find them
  - Twig syntax
- 

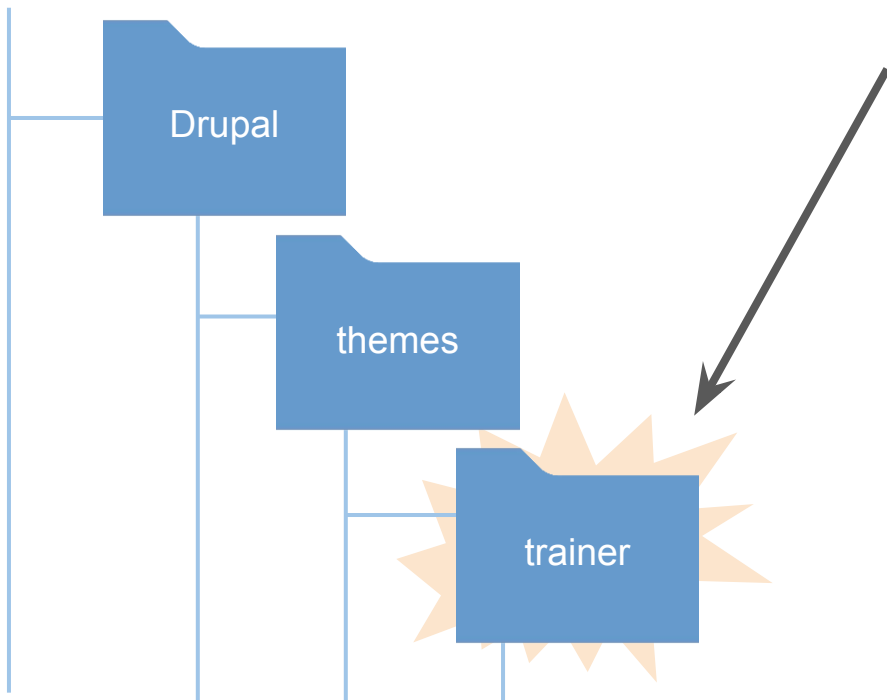
### Part 2









- Working with template suggestions
- Modifying content with `hook_preprocess()`
- Render arrays
- Custom templates with `hook_theme()`



# Introduction to Drupal 8 Theme Development

## Anatomy of a Theme named “trainer”



-  `trainer.info.yml`
-  `trainer.libraries.yml`
-  `trainer.theme`
-  `templates/`
-  `css/`
-  ○ `my-style.css`
-  `js/`
-  ○ `my-js.js`



# Introduction to Drupal 8 Theme Development

## Theme info file (`trainer.info.yml`)



`trainer.info.yml`

A theme's info file tells Drupal about the theme and its configuration. It uses the YAML file format of key-value pairs and lists of KV pairs.

- Required Properties

- `name: 'Trainer'`
- `description: 'Theme example for DCAV1'`
- `version: 8.x-0.1`
- `type: theme`
- `core: 8.x`

- Recommended Properties

- `base theme: stable`



# Introduction to Drupal 8 Theme Development

## `trainer.info.yml` continued...



`trainer.info.yml`

- **Regions** - containers for blocks (content)
  - `regions:`
    - `navigation: 'Navigation'`
    - `header: 'Top Bar'`
    - `footer: 'Footer'`
    - `content: 'Content'`
    - `sidebar_first: 'Primary Sidebar'`
- **Libraries** - assets or resources required by the theme
  - `libraries:`
    - `'trainer/my-custom-assets'`



# Introduction to Drupal 8 Theme Development

## Libraries (`trainer.libraries.yml`)



`trainer.libraries.yml`

### Name of asset bundle

### Version

### CSS by type (theme|module)

*Note: file path is key and object is value*

### JS

*Note: file path is key and object is value*

### Dependencies

*Name of asset bundles required by this asset bundle*

```
my-custom-assets:  
  version: 1.x  
  css:  
    theme:  
      css/my-style.css: {}  
  js:  
    js/my-js.js: {}  
  dependencies:  
    - core/jquery
```



# Introduction to Drupal 8 Theme Development

## **my-style.css -and- my-js.js**

Our new theme needs some style and interactivity. What better for style than red borders around everything, and how about a popup box for interaction!?



my-style.css

- **trainer/css/my-style.css** -

```
div { border: 1px solid red; }
```



my-js.js

- **trainer/js/my-js.js** -

```
(function($){  
    alert('hello world - jQuery version ' + $.fn.jquery);  
})(jQuery);
```





# Introduction to Drupal 8 Theme Development

## Child Theme



[trainer.info.yml](http://trainer.info.yml)

Creating a new theme as a “child theme” means that your theme inherits the configuration of another theme. This allows a developer to leverage another theme’s infrastructure in hopes of shorter theme development time and greater theme stability.

**How to make a child theme of ‘classy’** - `trainer.info.yml`

```
base theme: classy
```

**How to make a new base theme** - `trainer.info.yml`

```
base theme: false
```



# Introduction to Drupal 8 Theme Development

## Templates & Twig



`<*.html.twig`

Templates are snippets of code that represent an HTML component of your Drupal site. Drupal aspires to keep all HTML in templates. In Drupal 8, the Twig templating engine is used. Twig filenames end in “.html.twig”

**Example:** some core templates

- **page.html.twig** - contains generic content wrapper HTML, such as `<header>`, `<footer>`, `<main>`, `<section id="primary-sidebar">`
- **html.html.twig** - contains outer-most HTML sent to the browser, such as the `<html>`, `<head>`, and `<body>` tags
- **node.html.twig** - contains generic “node” entity HTML wrappings, such as `<article>`



# Introduction to Drupal 8 Theme Development

## Overriding templates

To override another Drupal template, copy the original template to your theme's `templates` folder.



core module  
template file

copy



custom theme  
template file

**Copy template from core module that provides it--**



core/modules/system/templates



page.html.twig

**To your theme's `templates` folder--**



themes/trainer/templates



page.html.twig



# Introduction to Drupal 8 Theme Development

## Overriding templates - Child theme edition

To override a base theme's template, copy the template from the that theme's `templates` folder to your theme's `templates` folder. **If the file doesn't exist in base theme, look in the module for it.**



base theme template file



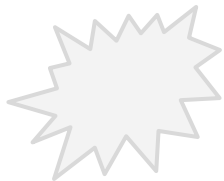
custom theme template file

**Copy template from base theme if it is provided--**

```
core/themes/classy/templates/  
Page.html.twig
```

**To your theme's templates folder--**

```
themes/trainer/templates  
page.html.twig
```



# Introduction to Drupal 8 Theme Development

## Practice: Override the core twig templates in your theme

Drupal provides many templates out of the box. Some examples of core templates are:

- `html.html.twig`
- `page.html.twig`
- `region.html.twig`
- `node.html.twig`
- `block.html.twig`
- `field.html.twig`

Using what you've learned, find these templates, and add them to your theme. **When in doubt, look in the “system” module.**



# Introduction to Drupal 8 Theme Development

## Template suggestions

Drupal's template discovery process is based on an array of "template suggestions". This array contains a list of possible template file names, and can be modified by modules and themes.

Example: Drupal core `page` template suggestions

```
[ "page__", "page__front" ]
```

Example: Drupal core `node` template suggestions

```
[ "node__full", "node__article", "node__article__full",  
"node__5", "node__5__full" ]
```





# Introduction to Drupal 8 Theme Development

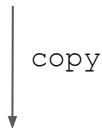
## Using a Template Suggestion

Page Template Suggestions on Front page

```
"page__",  
"page__front"
```



page.html.twig



page--front.html.twig

Copy template and rename it to new suggestion, replacing underscores with dashes--

### Result:

- themes/trainer/templates
  - page.html.twig
  - page--front.html.twig

Now we have a separate `page` template for the site's homepage!



# Introduction to Drupal 8 Theme Development

## Working with Twig - Syntax

In Twig there are two kinds of delimiters: `{% ... %}` and `{{ ... }}`. The first one is used to execute statements such as for-loops, the latter prints the result of an expression to the template. <http://twig.sensiolabs.org/>

<b>Output variable values</b>	<code>{{ my_variable }}</code>	Hello world
<b>Output nested values</b>	<code>{{ name.first }}</code> <i>or</i> <code>{{ name['first'] }}</code>	Jonathan
<b>Statements with {% ... %}</b>	<code>{% if name.first == 'Jonathan' %}</code> Hi <code>{{ name.first }}</code> <code>{% endif %}</code>	Hi Jonathan
<b>Filter values w/ pipe  </b>	<code>{{ name.first lower }}</code>	jonathan
<b>Code Comments</b>	<code>{# this is ignored by twig #}</code>	





# Introduction to Drupal 8 Theme Development

## Twig Template Example

Comments

Output

Statements

```
<main role="main">
  <a id="main-content" tabindex="-1"></a>{# link is in html.html.twig #}

  <div class="layout-content">
    {{ page.content }}
  </div>{# /.layout-content #}

  {% if page.sidebar_first %}
    <aside class="layout-sidebar-first" role="complementary">
      {{ page.sidebar_first }}
    </aside>
  {% endif %}

  {% if page.sidebar_second %}
    <aside class="layout-sidebar-second" role="complementary">
      {{ page.sidebar_second }}
    </aside>
  {% endif %}
</main>
```



# Introduction to Drupal 8 Theme Development

## More about Twig

<b>Variable Assignment</b>	<pre>{% set my_string = 'The quick brown...' %} {% set my_list = [ 'three', 2, 'one' ] %} {% set my_object = { 'name': 'Bobby' } %}</pre>
<b>Loops</b>	<pre>{% for item in my_list %}   {{ item }} {% endfor %}</pre>
<b>Functions</b>	<pre>{{ link("My Link", "http://google.com") }}</pre>
<b>Join &amp; Split filters</b>	<pre>{{ my_list join(',') }} {% set new_list = "five,six,seven" split(',') %}</pre>



# Introduction to Drupal 8 Theme Development

## Twig Resources

### Twig

- Documentation - <http://twig.sensiolabs.org/documentation>
- Template Designers Syntax - <http://twig.sensiolabs.org/doc/templates.html>

### Twig & Drupal

- Twig in Drupal 8 - <https://www.drupal.org/theme-guide/8/twig>
- Twig Best Practices - <https://www.drupal.org/node/1920746>
- Drupal Twig Functions - <https://www.drupal.org/node/2486991>



# Introduction to Drupal 8 Theme Development

## Theme `.theme` file (`trainer.theme`)



`trainer.theme`

A theme's "dot theme" file (`trainer.theme`) is an entry point for PHP developers to hook into Drupal and make most customizations. Place custom PHP in this file. For example: hooks.

**Most often a theme file and its dependencies will focus on modifying the output of the Drupal site as HTML.**

That is the whole purpose of a theme after all; to stylize the output of a system, as opposed to governing the system with logic.



# Introduction to Drupal 8 Theme Development

## Theme `.theme` file continued...

Using the `trainer.theme` file, let's hook into Drupal and begin making modifications to the system! We'll do the following things:

- Add new template suggestion for unpublished nodes using `hook_template_suggestions_HOOK_alter()`.  
Then implement the new suggestion.
- Preprocess the page template and alter its contents by using `hook_preprocess_HOOK()`.  
Create a new value that the Twig template can output.



# Introduction to Drupal 8 Theme Development

## hook\_template\_suggestions\_HOOK\_alter

Provide a template suggestion if the node is not published.

```
function trainer_theme_suggestions_node_alter(&$suggestions, $variables){  
  // node object is available within $variables array  
  $node = $variables['elements']['#node'];  
  
  if ( !$node->isPublished() ) {  
    $suggestions[] = 'node__unpublished';  
  }  
}
```



# Introduction to Drupal 8 Theme Development

Using new template suggestion: `node--unpublished.html.twig`

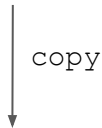
Copy the basic node template and rename it to new suggestion, replacing underscores with dashes, and leaving the file extension “.html.twig”--

## Result:

- `themes/trainer/templates`
  - `node.html.twig`
  - `node--unpublished.html.twig`



`node.html.twig`



`node--unpublished.html.twig`

Now we have a separate `node` template for unpublished nodes! This isn't the most practical example, but you should now have an idea of easy it can be to add a new template suggestion when you have the right hook:

```
hook_template_suggestions_HOOK_alter
```



# Introduction to Drupal 8 Theme Development

## hook\_preprocess\_HOOK

Alter variables available to the template before they are sent to Twig

Q: What is this?

A: That's a "render array"

```
function trainer_preprocess_page( &$variables ) {  
  $variables['example_thing'] = [  
    '#markup' => 'This is my custom theme thing. Deal with it.',  
  ];  
}  
  
function trainer_preprocess_html( &$variables ){  
  // provide a variable to twig template named "is_front"  
  $variables['is_front'] = \Drupal::service('path.matcher')->isFrontPage();  
}
```





# Introduction to Drupal 8 Theme Development

## Render Arrays

"Render Arrays" or "Renderable Arrays" are the foundational components of a Drupal page. A render array is a hierarchical associative array containing data to be rendered and properties describing how the data should be rendered. A render array is often returned by a function to specify markup to be sent to the web browser or other services

### Most simple example:

```
$variables['example_thing'] = [  
  '#markup' => 'This is my custom theme thing. Deal with it.',  
];
```



# Introduction to Drupal 8 Theme Development

## Render Arrays Example: item list

Create an HTML unordered bullet list of strings:

# #

Properties of the element are prefixed with the number sign, whereas values are not.

```
$variables['my_list'] = [ // key is twig var name
  '#theme' => 'item_list', // template
  '#list_type' => 'ul', // template property
  '#items' => [ // template property
    'first item',
    'second item',
    'third item',
  ],
];
```



# Introduction to Drupal 8 Theme Development

## Render Array Examples in Action

PHP - trainer.theme

```
function trainer_preprocess_page( &$variables ) {  
  $variables['example_thing'] = [  
    '#markup' => 'This is my custom theme thing. De  
  ];  
  $variables['my_list'] = [  
    '#theme' => 'item_list',  
    '#list_type' => 'ul',  
    '#items' => [  
      'first item',  
      'second item',  
      'third item',  
    ],  
  ],  
};
```

Twig - page.html.php

```
HERE---  
  
{{ example_thing }}  
{{ my_list }}
```

Rendered Output

HERE--- This is my custom theme thing. De

- first item
- second item
- third item



# Introduction to Drupal 8 Theme Development

## Render Array Resources

- **Drupal 8 Render Arrays API** - General overview of render arrays  
<https://www.drupal.org/developing/api/8/render/arrays>
- **Drupal 7 Form API** - Though the API is technically for Drupal 7, these properties also apply to Drupal 8.  
[https://api.drupal.org/api/drupal/developer%21topics%21forms\\_api\\_reference.html/7.x](https://api.drupal.org/api/drupal/developer%21topics%21forms_api_reference.html/7.x)
- **Render Array Snippets** - Oldie but a goodie  
<https://chacadwa.com/blog/2012/12/20/drupal-7-render-array-snippets>
- **Drupal 7 Examples module** - render\_examples submodule  
<https://www.drupal.org/project/examples>



# Introduction to Drupal 8 Theme Development

## Hook\_theme - Providing new templates

All templates within Drupal have been provided by some module or theme's implementation of `hook_theme`. Providing your own new templates is a great way to further understand render arrays, and how Drupal works fundamentally.

### **Hook**

#### **Summary:**

return an array of new components

```
function hook_theme($existing, $type, $theme, $path){
  return [
    'my_new_component' => [
      '#template' => 'my_new_component_template_name'
      '#variables' => [ 'some_var' => 'default value' ]
    ]
  ];
}
```



# Introduction to Drupal 8 Theme Development

## Hook\_theme - About Node Author

This example provides a new template to Drupal that will serve as an “about the author” component.

```
function trainer_theme(){  
  return [  
    'about_node_author' => [  
      'template' => 'node_author',  
      'variables' => [  
        'first_name' => '',  
        'last_name' => '',  
        'picture' => [],  
      ],  
    ],  
  ];  
}
```

**Key** is the component's machine safe name

**Template** is the template file name without file extension.  
ie, `node_author.html.twig`

**Variables** is an array of possible component variables and their default values.



# Introduction to Drupal 8 Theme Development

## Hook\_theme - new Twig Template

Next, we want to create a new Twig template for this component

```
function trainer_theme(){  
  return [  
    'about_node_author' => [  
      'template' => 'node_author',  
      'variables' => [  
        'first_name' => '',  
        'last_name' => '',  
        'picture' => [],  
      ],  
    ],  
  ];  
}
```

trainer/templates/node\_author.html.twig

```
<div>  
  <h3>About the author</h3>  
  <div> {{ picture }} </div>  
  <h4>{{ first_name }} {{ last_name }}</h4>  
</div>
```



# Introduction to Drupal 8 Theme Development

## Hook\_theme - In Action

Finally, we'll use `hook_preprocess_node` to add our new element to all node pages. In `trainer.theme` --

```
function trainer_theme(){
  return [
    'about_node_author' => [
      'template' => 'node_author',
      'variables' => [
        'first_name' => '',
        'last_name' => '',
        'picture' => [],
      ],
    ],
  ];
}
```

```
function trainer_preprocess_node( &$variables ) {
  $author = $variables['elements']['#node']->getOwner();

  $variables['node_author_details'] = [
    '#theme' => 'about_node_author',
    '#first_name' => $author->get('field_first_name')->value,
    '#last_name' => $author->get('field_last_name')->value,
    '#picture' => [
      '#theme' => 'image_style',
      '#style_name' => 'thumbnail',
      '#uri' => $author->get('user_picture')->entity->getFileUri(),
    ],
  ];
}
```





# Introduction to Drupal 8 Theme Development

## Hook\_theme - Results

trainer.theme

```
function trainer_preprocess_node( &$variables ) {  
  $author = $variables['elements']['#node']->getOwner();  
  
  $variables['node_author_details'] = [  
    '#theme' => 'about_node_author',  
    '#first_name' => $author->get('field_first_name')->value,  
    '#last_name' => $author->get('field_last_name')->value,  
    '#picture' => [  
      '#theme' => 'image_style',  
      '#style_name' => 'thumbnail',  
      '#uri' => $author->get('user_picture')->entity->getFileUri(),  
    ],  
  ];  
}
```

node.html.twig

```
<div{{ content_attributes }}>  
  {{ content }}  
</div>  
  
{{ node_author_details }}
```

node\_author.html.twig

```
<div>  
  <h3>About the author</h3>  
  <div> {{ picture }} </div>  
  <h4>{{ first_name }} {{ last_name }}</h4>  
</div>
```

About the author



Jonathan Daggerhart