

# Drupal 8 / Theming Quickstart

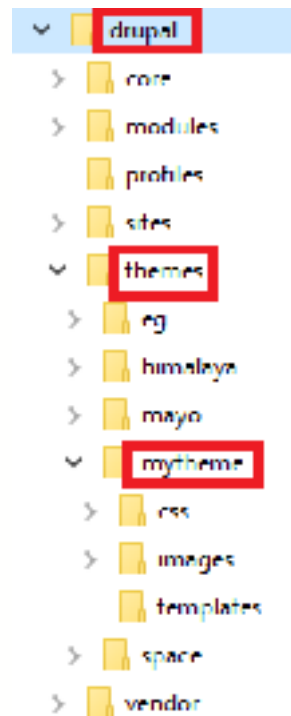


# Introduction to themes in Drupal 8

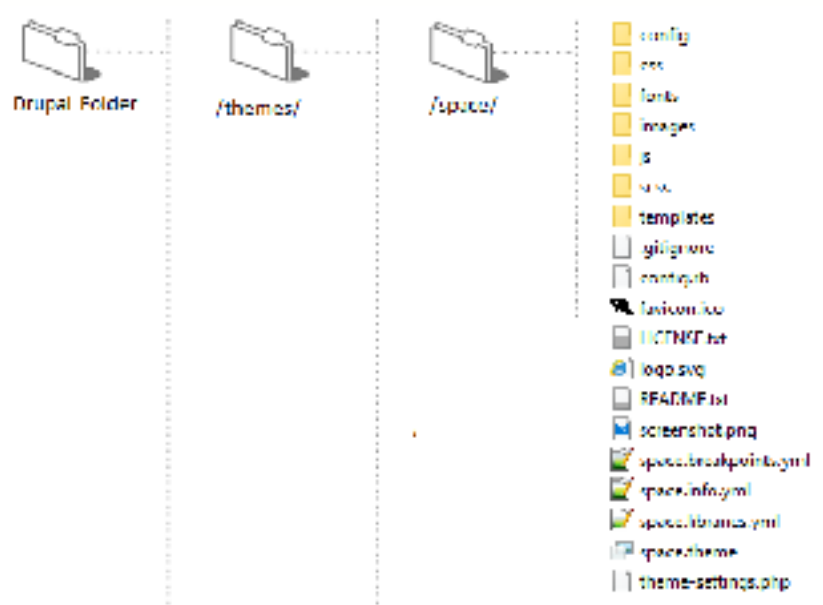
- » New theme layer based in Twig (used in other CMSs)
- » 2 new core base themes: Stable & Classy
  - » Both contain all the templates Drupal puts out from core
  - » 'Stable' markup will not change between major releases
  - » 'Classy' contains BEM-style class structure and logic
- » Base your new theme on one or the other depending on needs
- » If no base theme declared, it's Stable by default

# Theme folder in Drupal hierarchy

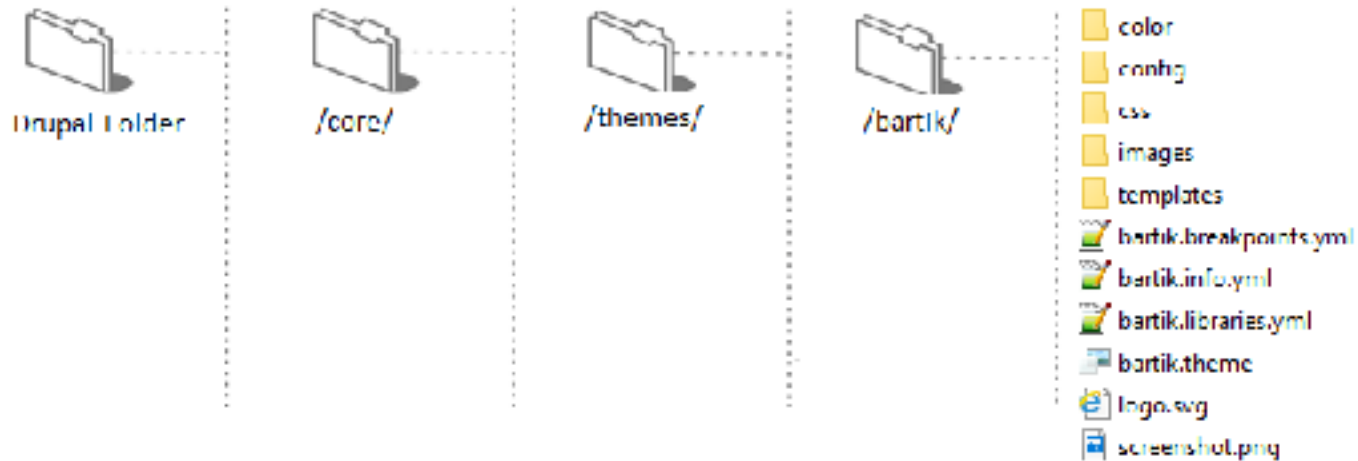
- » Location in file hierarchy:
  - Drupal core folder/themes/



# Custom theme file organization



# Core theme file organization



# Types of themes

<b>Core themes</b>	Drupal core comes with a few themes. These are suitable for very basic sites.
<b>Contributed Standard themes</b>	Free themes that have been contributed back to the Drupal Community. <a href="http://drupal.org/project/themes">http://drupal.org/project/themes</a>
<b>Contributed Starter/Base themes</b>	Base or Starter themes are contributed themes designed to be used as a starting point for a custom subtheme.
<b>Custom themes</b>	Most sites require a custom look and feel. These are often created as subthemes of a <i>starter</i> or <i>base theme</i> .
<b>Contributed Administration themes</b>	Themes that are displayed only in administration sections of a site

# Finding & evaluating contributed themes

- » Contributed Theme Considerations
  - Two types of themes, visitor-facing, and admin-facing
  - Themes may or may not resize for mobile devices.
  - Base themes may not be complete enough for site builders new to Drupal
- » [https://www.drupal.org/project/project\\_theme](https://www.drupal.org/project/project_theme)

# Project page

- » Who maintains this?
- » What are current issues?
- » Similar projects?
- » Documentation?
- » Download links

The screenshot shows the SourceForge project page for 'Drupal'. The page is divided into several sections, with red boxes highlighting specific areas of interest:

- Project Information:** This section provides details about the project, including its name, version, and a brief description. It also includes a table of download links for different operating systems and architectures.
- Maintenance for Query Countdown:** This section lists the maintainers and contributors for the project, along with their contact information and roles.
- Issues for Query Countdown:** This section displays a list of open issues, including their titles, status, and the date they were created.
- Download Links:** This section contains a table of download links for the project, organized by operating system and architecture.

Operating System	Architecture	Download Link
Windows	x86	<a href="#">Download</a>
Windows	x64	<a href="#">Download</a>
Linux	x86	<a href="#">Download</a>
Linux	x64	<a href="#">Download</a>
Mac OS X	x86	<a href="#">Download</a>
Mac OS X	x64	<a href="#">Download</a>



# Downloads and versions

## Downloads

### Recommended releases

Version	Download	Date
7.x-2.9	<a href="#">tar.gz</a> (184.45 KB)   <a href="#">zip</a> (223.92 KB)	2015-Mar-16
6.x-1.5	<a href="#">tar.gz</a> (91.22 KB)   <a href="#">zip</a> (119.68 KB)	2012-Aug-03

### Other releases

Version	Download	Date
8.x-3.0-unstable-4	<a href="#">tar.gz</a> (79.48 KB)   <a href="#">zip</a> (175.55 KB)	2015-May-07

### Development releases

Version	Download	Date
8.x-3.x-dev	<a href="#">tar.gz</a> (88.52 KB)   <a href="#">zip</a> (198.57 KB)	2015-Jul-24
7.x-2.x-dev	<a href="#">tar.gz</a> (184.44 KB)   <a href="#">zip</a> (223.96 KB)	2015-Feb-18
6.x-1.x-dev	<a href="#">tar.gz</a> (91.24 KB)   <a href="#">zip</a> (119.7 KB)	2013-Oct-01

[View all releases](#)

# Project information

## Project Information

Maintenance status: **Actively maintained**  
Development status: **Under active development**  
Reported installs: **189,136** sites currently report  
Downloads: 1,990,655  
Automated tests: Enabled  
Last modified: December 2, 2014

### Maintainers for Rules

**klausl** – 186 commits  
last: 5 days ago, first: 5 years ago  
**fago** – 946 commits  
last: 1 month ago, first: 7 years ago  
[View all committers](#)  
[View commits](#)

### Issues for Rules

To avoid duplicates, please search before submitting a new issue.  
[Advanced search](#)

#### All issues

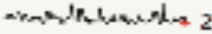
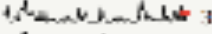
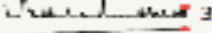
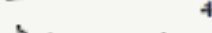

1369 open, 3432 total

#### Bug report

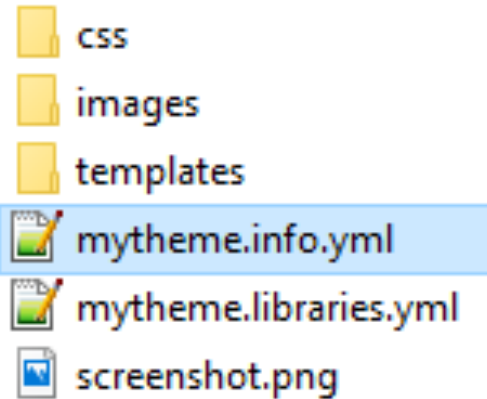
402 open, 1137 total

[Subscribe via e-mail](#)

#### Statistics

New issues  2  
Response rate  38 %  
1st response  35 hours  
Open bugs  402  
Participants  12  
2 year graphs, updates weekly

# The contents of a simple theme



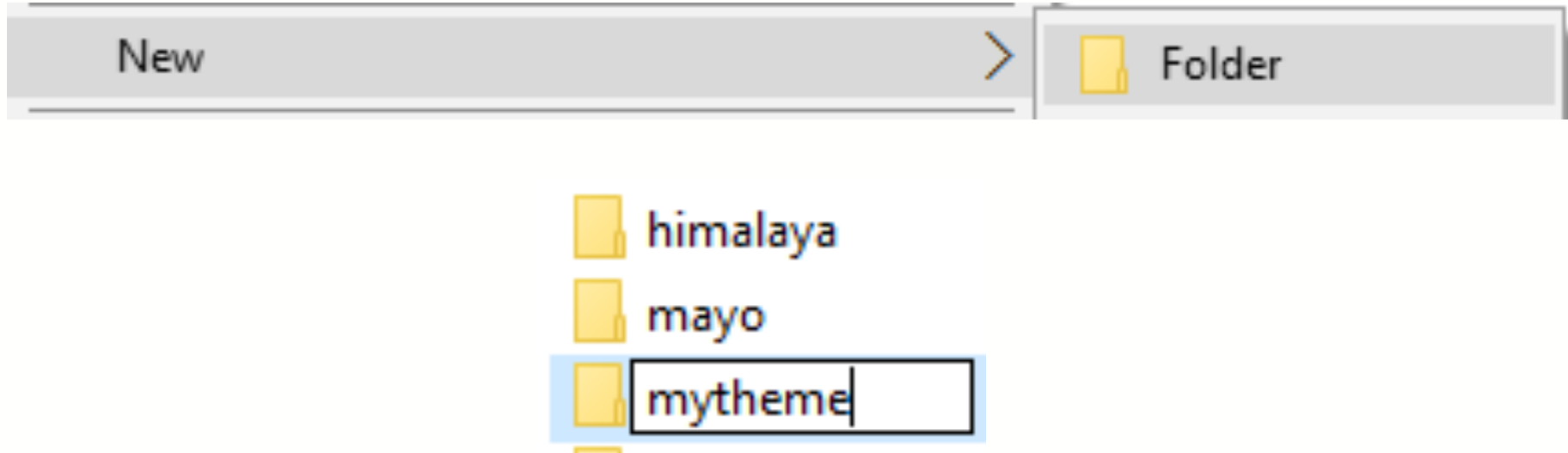
# Let's make a custom theme

- » You'll need your text editor to create new files
- » Project names are important
  - They're the “key” that connect all components of the project
  - Must be unique
  - Conventions
    - all lower case beginning with a letter
    - no spaces, dashes or punctuation
    - can contain underscores and numbers if not leading with one

# .info.yml file

- » A **.info.yml** file is the only required file of a theme
- » It makes declarations to Drupal, specifying:
  - the theme name label
  - type of project (module or theme)
  - base theme
  - core compatibility
  - regions and feature overrides
  - location of CSS and other needed files
- » Optional values not specified will use Drupal's default

# Create your folder



# Make the **.info.yml** file

## » Make a Simple Theme

- Create a mytheme folder in the /themes/ directory
- Start a **mytheme.info.yml** file in your text editor, save to your project folder

# Contents of **mytheme.info.yml**

```
name: My Great Theme  
type: theme  
description: This is my first theme project  
core: 8.x
```



# Exercise: Add a logo and screenshot

- » Copy screen shot and logo from course asset folder
- » Declare in **mytheme.info.yml**
- » Visit **Manage > Structure > Block layout** and ensure the Site Branding block is in the Header region
- » Configure to turn off Site name and Site slogan



**THIS IS MY LOGO**

# Updated contents of **mytheme.info**

```
name: My Great Theme
type: theme
description: This is my first theme project
core: 8.x
screenshot: mytheme.png
```

# Regions

- » The areas you assign blocks to
- » Drupal has default regions that it uses if you do not declare ANY custom regions in **.info.yml**
- » Declare ONE custom region, you need to declare all
- » Three files help you customize regions:
  - Declare in the **.info.yml**
  - Render **page.html.twig** template
  - Style with CSS

# Default regions

- » sidebar\_first: Items for the first sidebar.
- » sidebar\_second: Items for the second sidebar.
- » content: The main content of the current page.
- » header: Items for the header region.
- » primary\_menu: Items for the primary menu region.
- » secondary\_menu: Items for the secondary menu region.
- » footer: Items for the footer region.
- » highlighted: Items for the highlighted content region.
- » help: Dynamic help text, mostly for admin pages.
- » breadcrumb: Items for the breadcrumb region.

# Hidden regions

- » Two hidden regions
  - page\_top
  - page\_bottom
- » Not available to for blocks, but modules and Drupal system messages might utilize them

# Notes about regions

- » Don't forget: your theme has **all** default regions—*until you define your first region!*
- » The regions you define are displayed in **Manage > Structure > Block layout**
- » This **Block layout** list will match the labels you assign and the order you used in your **.info.yml**

# Exercise: Adding regions

- » Open **mytheme.info.yml**.
- » Add the code to the right.
- » Save and clear the cache.
- » Go to the Block Layout page.
- » You'll only see the three visible regions you declared.
- » Click **Demonstrate block regions**.  
*Notice it reflects your regions!*

```
regions:  
  header: Header  
  content: Content  
  footer: Footer  
  page_top: Page top  
  page_bottom: Page bottom
```

# Exercise: Add `.libraries.yml` file

- » Create a `mytheme.libraries.yml` text file
- » Indent with 2 spaces at a time—do not use tabs

```
1 global-css:
2   version: VERSION
3   css:
4     theme:
5       css/elements.css: {}
6       css/layout.css: {}
7       css/print.css: { media: print }
```



# Exercise: Edit `.info.yml`

» Add this to `mytheme.info.yml` (spacing is important!):

```
6 libraries:  
7   - mytheme/global-css
```

# Exercise: Clear the cache!

- » After updating **.info.yml** files and **.libraries.yml** files, always clear your cache
  - **Manage > Configuration > Development > Performance > Clear all caches**

# Using remote assets

- » Content Delivery Networks
- » Remote CSS
- » Webfonts

# Using JavaScript assets

- » You can also define JavaScript assets for your theme
- » Make a container for them in **.libraries.yml**
- » Syntax is similar to CSS. Example:

```
picturefill:  
  remote: https://github.com/scottjehl/picturefill  
  version: "3.0.1"  
  license:  
    name: MIT  
    url: https://github.com/scottjehl/picturefill/blob/3.0.1/LICENSE  
    gpl-compatible: true  
  js:  
    assets/vendor/picturefill/picturefill.min.js: { weight: -10, minified: true }
```

# Declaring dependencies

- » Drupal 8 provides assets like jQuery and normalize.css
  - asset libraries available for your theme
  - does not load scripts by default
  - you selectively loads libraries
- » Core asset location: /core/assets/vendor
- » Syntax for loading core assets: - core/assetname
- » Syntax for other installed assets: - project/asset

```
global-js:  
dependencies:  
  - core/jquery
```

# Exercise: Adding a web font

- » Open **mytheme.libraries.yml**
- » Add the following after the end of global-css:

```
8  pacifico:
9  license:
10     name: SIL Open Font License 1.1
11     url: http://scripts.sil.org/cms/scripts/page.php?site_id=nrsi&id=OFL
12  css:
13     theme:
14         //fonts.googleapis.com/css?family=Pacifico: { type: external }
15         css/styles.css: {}
16
```

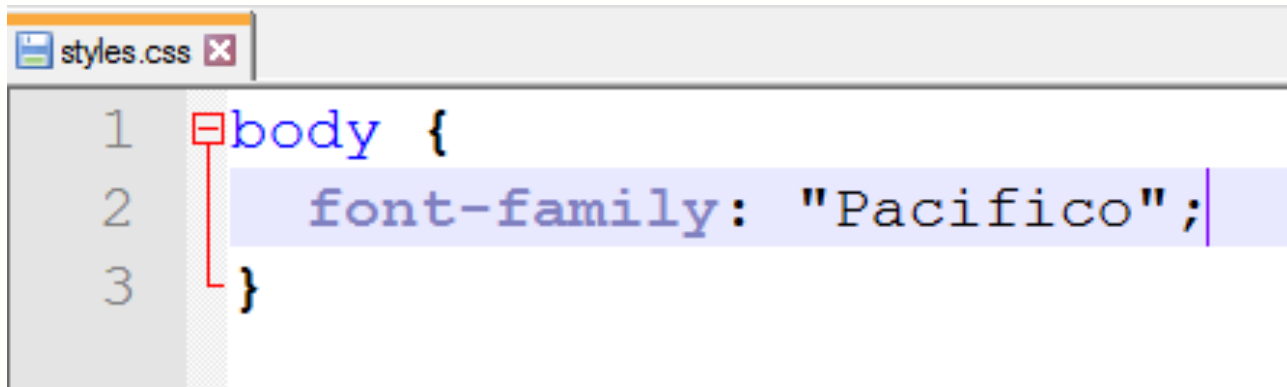
# Exercise: Updating `.info.yml`

- » Open `mytheme.info.yml`
- » Update the list of libraries to look like this:

```
1  name: My Great Theme
2  type: theme
3  description: This my first theme project
4  core: 8.x
5  screenshot: mytheme.png
6  libraries:
7    - mytheme/global-css
8    - mytheme/pacifico
```

# Exercise: Adding styles.css

- » Create **styles.css** in the /css/ folder
- » Add the following CSS:

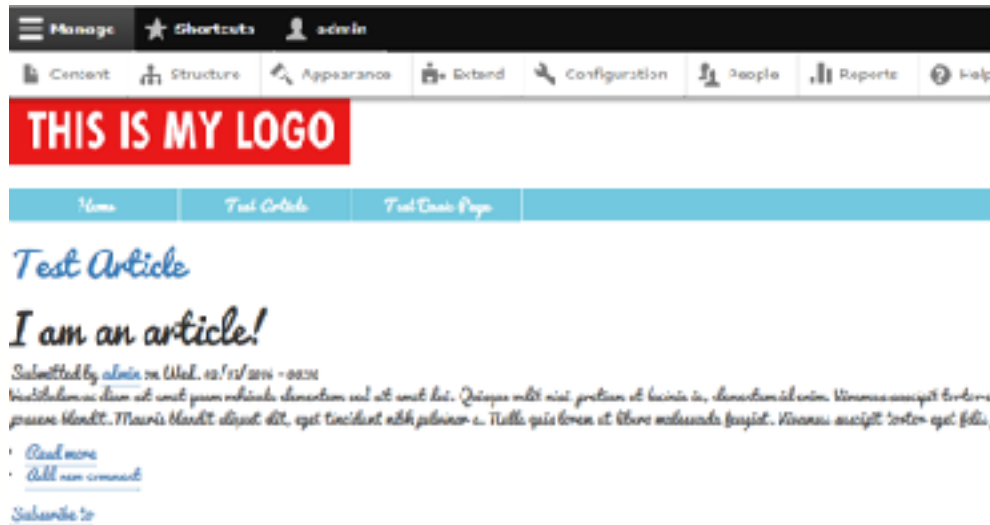
A screenshot of a code editor window titled 'styles.css'. The editor shows three lines of CSS code. Line 1: 'body {' (the word 'body' is blue). Line 2: 'font-family: "Pacifico";' (the text is highlighted in light blue). Line 3: '}' (the closing brace). A red bracket on the left side of the code block connects the opening and closing braces. A vertical purple cursor is at the end of line 2.

```
1 body {  
2   font-family: "Pacifico";  
3 }
```



# Exercise: Result

- » Clear the cache
- » Visit your page!



# Advanced **.info.yml** options

## » libraries-extend

```
libraries-extend:  
  core/drupal.user:  
    - mytheme/user1  
    - mytheme/user2
```

## » libraries-override

```
libraries-override:  
  † Replace an entire library.  
  core/drupal.collapse: mytheme/collapse
```

## » stylesheet-remove

```
stylesheet remove:  
  - core/assets/vendor/normalize-css/normalize.css
```

# Adding breakpoints

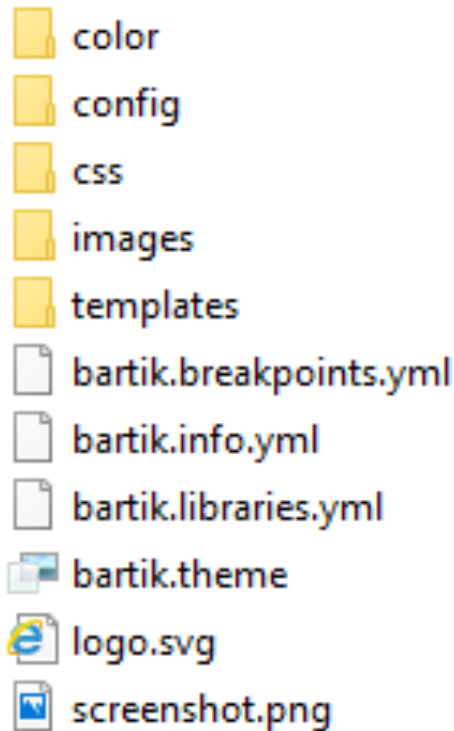
- » Used in responsive design
- » Consist of a label and a media query
- » Media queries encode the breakpoints, allow themer to implement different ways of displaying content
- » Breakpoints are defined in a **.breakpoints.yml** file
- » Breakpoint name has syntax of themename.descriptor
- » Example: bartik.mobile: or bartik.narrow:

# Exercise: Adding breakpoints

```
mytheme.mobile:
  label: mobile
  mediaQuery: ''
  weight: 0
  multipliers:
  | 1x
mytheme.narrow:
  label: narrow
  mediaQuery: 'all and (min width: 360px) and (max width: 860px)'
  weight: 1
  multipliers:
  | 1x
mytheme.wide:
  label: wide
  mediaQuery: 'all and (min-width: 861px)'
  weight: 2
  multipliers:
  | - 1x
```

# Anatomy of a theme

- » Drupal themes are made up of many files – most commonly:
  - The **.info.yml** file
  - The **.libraries.yml** file
  - Image, CSS and JS files
  - Twig Template Files (html.twig)
  - A **thename.theme** file



# screenshot.png and logo.svg

## » screenshot.png

- allows Drupal admins to preview what your theme will look like, when viewing your theme on the Appearance page

## » logo.svg

- the site logo that displays with your theme
- .svg format is required and allows scalability

# The .theme file

- » The .theme file is used to store theme-specific PHP functions, preprocess functions, and hooks
- » This was called template.php in previous versions of Drupal, but now has a syntax of **themenametheme**

```
1 1/100
2
3 /**
4  * @file
5  * Functions to support theming in the Bartik theme.
6  */
7
8 use Drupal\Core\Template\Attribute;
9
10 /**
11  * Implements hook_preprocess_HOOK() for HTML documents templates.
12  *
13  * * Adds body classes if certain regions have content.
14  */
15 function bartik_preprocess_html(&$variables) {
16   // Add information about the number of columns.
17   if ($page = $variables['page']) { $sidebar_class = $page['sidebar_output']; }
18   $variables['attributes']['class'][] = "layout-{$sidebar_class}";
19 }
20
21 elseif ($page['variables']['page']['sidebar_first']) {
22   $variables['attributes']['class'][] = "layout-{$sidebar_class}";
23   $variables['attributes']['class'][] = "layout-{$sidebar_class}-first";
24 }
25 elseif ($page['variables']['page']['sidebar_output']) {
26   $variables['attributes']['class'][] = "layout-{$sidebar_class}";
27   $variables['attributes']['class'][] = "layout-{$sidebar_class}-output";
28 }
29
30 elseif ($variables['attributes']['class']) {
31   $variables['attributes']['class'][] = "layout-{$sidebar_class}";
32 }
33
34 if ($page['variables']['page']['content_top']) {
35   $variables['attributes']['class'][] = "has-content-top";
36 }
37 }
```

# Coding standards for Drupal: PHP

- » Use an indent of 2 spaces, with no tabs.
- » Lines should have no trailing whitespace.
- » Files should be formatted with Unix line endings (“\n”)
- » Don’t use Windows line endings (“\r\n”)
- » Lines should not be longer than 80 chars (generally)
- » <http://drupal.org/coding-standards>



# Coding standards for Drupal: Twig

- » Use a space after an opening delimiter, and before a closing delimiter
  - Examples:
    - `{{ foo }}`
    - `{% if bar %}{% endif %}`
- » Put one space before and after operators
  - Examples:
    - `{{ foo == 1 }}`
    - `{{ true ? true : false }}`
- » Put one space after the use of `:` or `,` in arrays or hashes
- » Do not put a space between open and closing parentheses in expressions
  - Example:
    - `{{ 1 + (2 * 3) }}`
- » Do not put a space between string delimiters
  - Example:
    - `{{ 'foo' }}`
    - `{{ "foo" }}`

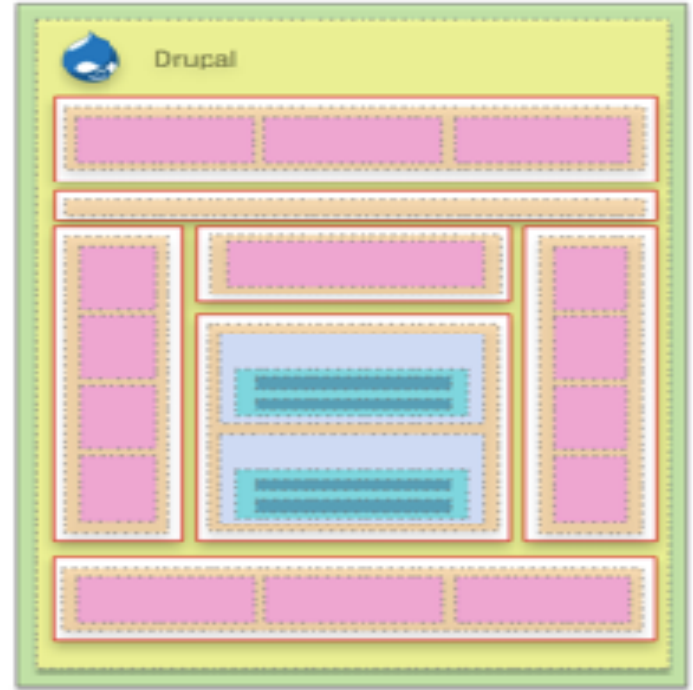


# Coding standards for Drupal: .yaml

- » The .yaml files do not accept tab characters, and will throw an error if they are present
- » Indents should be two spaces long, per indentation
- » When encountering errors, go back to an admin page and then visit: **Reports > Recent log messages** for help troubleshooting

# Introduction to Twig template files

- » Files end in **.html.twig**
- » Supply the markup that renders what is seen in browsers
- » Contain mix of HTML 5 syntax and Twig syntax



# Drupal 8 template hierarchy

- » Templates render from most specific to least specific
- » field.html.twig >
  - node.html.twig >
    - region.html.twig >
      - page.html.twig >
        - » html.html.twig

# Modules and .html.twig template files

- » It's discouraged to create markup in module PHP files
- » Most module markup can be found in .html.twig templates
- » <https://www.drupal.org/node/2640110>

# Template overrides

- » To override an existing template, do three things:
  - Locate the existing template you wish to override
  - Make a copy of it
    - This is important, you don't want to edit core templates!
  - Place the copy in your theme's /templates/ folder
- » Can be general—overriding **page.html.twig** everywhere
- » Can be specific—overriding **page.html.twig** on node 44

...but which template do you override?

# Exercise: Turn on Twig debugging

```
<!-- THEME DEBUG -->
<!-- THEME HOOK: 'node' -->
<!-- FILE NAME SUGGESTIONS:
  * node--65--full.html.twig
  * node--65.html.twig
  * node--article--full.html.twig
  * node--article.html.twig
  * node--full.html.twig
  x node.html.twig
-->
<!-- BEGIN OUTPUT from 'core/themes/bartik/templates/node.html.twig' -->
▼<article data-history-node-id="65" data-quickedit-entity-id="node/65" role="article"
class="contextual-region node node--type-article node--promoted node--view-mode-full
clearfix" about="/node/65" typeof="schema:Article" data-quickedit-entity-instance-id="0">
```



# Drupal 8 CSS best practices

## » SMACSS categorization

- Base
- Layout
- Component
- State
- Theme



## » Minimum Files

- base.css
  - layout.css
  - components.css
- » Drupal aggregates CSS files, so multiples won't hinder speed on load.

How would you  
do that in Drupal?

# Real-life design considerations

- » Working in small groups or pairs, choose an inspiring Drupal site from [drupalshowcase.com](https://drupalshowcase.com)
- » Then, select a specific page or section to analyze
- » Define the content types, blocks, regions, and work on a wireframe drawing
- » Also identify if there's a base theme you might want to start out with if you were to create a new theme!

Thank you