Drupal 8 is the most advanced Drupal ever built and it will surely will be a game changer among other CMSes out there. Part of its strength comes from adopting the principles from other technologies like Symfony, YAML, Twig, and Backbone.js.

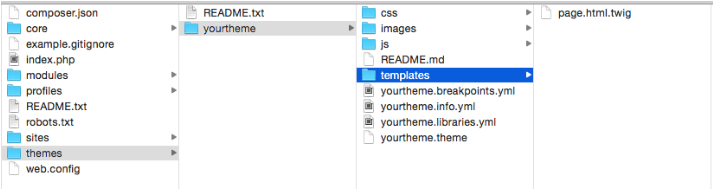
The theming has undergone many changes since Drupal 7, especially the deep integration with Twig – a popular templating engine for PHP, which opens the door to more robust, scalable and secure themes.

*Note that since Drupal 8 is still under active development, some of the details below may change prior to its release. Still, since Drupal 8 is now feature-frozen, hopefully most info should remain relevant.*

These are several things to note about the changes for theming in Drupal 8, and we will discuss each of these further in the next section:

* **a. *yourtheme.info.yml***  
  The use of *\*.info.yml* file as a replacement for *\*.info* file is definitely giving more options and flexibility on the theme configuration.
* **b. *yourtheme.libraries.yml***  
  This a configuration file to define assets for your theme – notably for css and javascript files.
* **c. *yourtheme.breakpoints.yml***  
  This a configuration file to setup breakpoints for your responsive custom theme.
* **d. *yourtheme.theme***  
  This file contains all of the preprocess functions, similar to the *template.php*file in Drupal 7.
* **e. *Twig template file (\*.html.twig)***  
  The usage of Twig as mentioned above gives more robustness on the front-end. The*new* syntax is cleaner and addresses a lot of common security pitfalls. Usually all the*\*.html.twig* files are put under the *templates*sub-folder.

Here is a screenshot for the basic folder structure for the theme:

**[](http://x-team.com/wp-content/uploads/2015/03/Screen-Shot-2015-03-24-at-15.32.55.png)**

***a. yourtheme.info.yml***

This is the first file that you need to set up for your theme. This file is equivalent to the *\*.info*file in Drupal 7, providing meta-data about your theme.

Consider this yaml code below:

|  |  |
| --- | --- |
|  | name: YourTheme |
|  | description: 'A Cool theme by you.' |
|  | type: theme |
|  | package: Custom |
|  | base theme: classy |
|  | core: 8.x |
|  | screenshot: yourtheme.png |
|  | libraries: |
|  | - yourtheme/global-styling |
|  | regions: |
|  | header: ‘Header’ |
|  | primary\_menu: 'Primary menu' |
|  | secondary\_menu: 'Secondary menu' |
|  | help: Help |
|  | page\_top: 'Page top' |
|  | page\_bottom: 'Page bottom' |
|  | featured\_top: 'Featured top' |
|  | breadcrumb: Breadcrumb |
|  | content: Content |
|  | sidebar\_first: 'Sidebar first' |
|  | sidebar\_second: 'Sidebar second' |
|  | featured\_bottom\_first: 'Featured bottom first' |
|  | featured\_bottom\_second: 'Featured bottom second' |
|  | featured\_bottom\_third: 'Featured bottom third' |
|  | footer\_first: 'Footer first' |
|  | footer\_second: 'Footer second' |
|  | footer\_third: 'Footer third' |
|  | footer\_fourth: 'Footer fourth' |
|  | footer\_fifth: 'Footer fifth’ |

[**view raw**](https://gist.github.com/ardnet/410bf79e6268674a48f0/raw/816f64bc3a371c40577741c2a3c1ccd11ace8eff/info.yml)[**info.yml**](https://gist.github.com/ardnet/410bf79e6268674a48f0#file-info-yml) hosted with  by **[GitHub](https://github.com/)**

These are the basic *keys*along with *values* that need to be provided:

* name: YourTheme  
  Required. The human readable name will appear on the Appearance page, where you can activate your theme.
* description: 'A Cool theme by you.'  
  Required. The description is also displayed on the Appearance page.
* type: theme  
  Required. The type key indicates the type of extension, e.g. module, theme or profile. For themes this should always be set to “theme”.
* package: Custom  
  The package key allows you to group themes together on the Appearance page.
* base theme: classy  
  The theme can inherit the resources from another theme by defining it as a base theme. **classy** in this case is the name of the base theme.
* core: 8.x  
  Required. The core key specifies the version of Drupal core that your theme is compatible with.
* screenshot: yourtheme.png  
  With the screenshot key you define a screenshot that is shown on the Appearance page. If you do not define this key then Drupal will look for a file named ‘screenshot.png’ in the theme folder to display.
* libraries:

|  |  |
| --- | --- |
|  | libraries: |
|  | - yourtheme/global-styling |

* [**view raw**](https://gist.github.com/ardnet/410bf79e6268674a48f0/raw/816f64bc3a371c40577741c2a3c1ccd11ace8eff/libraries.yml)[**libraries.yml**](https://gist.github.com/ardnet/410bf79e6268674a48f0#file-libraries-yml) hosted with  by **[GitHub](https://github.com/)**
* The libraries key can be used to add asset libraries — which can contain both CSS and JS assets — to all pages where the theme is active. From the example above,**yourtheme** is the name of the custom theme, followed by / (slash) and **global-styling** is the name of value to reference the library that we setup in the file of**yourtheme.libraries.yml** (will discuss further below).
* regions:https://gist.github.com/ardnet/410bf79e6268674a48f0#file-regions-ymlBasically a container for the block that can be added.Above we see some basic*key: ’value’* being defined. Taken from the example above: header is the region system name, and *‘Header’* is the name of the region that will display in the admin page.

**WARNING:**YAML has significant whitespace, so remember to check the INDENTATION, otherwise an error will occur.

***b. yourtheme.libraries.yml***

This file contain references to the assets – notably is the CSS and JavaScript files that needs to be loaded from your theme.

This is the basic setup of what could be contained in the *\*.libraries.yml* file.

|  |  |
| --- | --- |
|  | global-styling: |
|  | css: |
|  | theme: |
|  | css/styles.css: {} |
|  | css/layout.css: {} |
|  | css/print.css: { media: print } |
|  |  |
|  | js: |
|  | js/yourtheme.js: {} |
|  | js/custom-library.js: {} |
|  |  |
|  | dependencies: |
|  | - core/jquery |
|  | - core/drupal |
|  |  |
|  | custom-stuff: |
|  | css: |
|  | theme: |
|  | css/print.css: {} |
|  | css/another-style.css: {} |
|  |  |
|  | js: |
|  | js/yourtheme.js: {} |
|  | js/haha.js: {} |

[**view raw**](https://gist.github.com/ardnet/410bf79e6268674a48f0/raw/816f64bc3a371c40577741c2a3c1ccd11ace8eff/global-styling.yml)[**global-styling.yml**](https://gist.github.com/ardnet/410bf79e6268674a48f0#file-global-styling-yml) hosted with  by **[GitHub](https://github.com/)**

The *global-styling* and *custom-stuff* here can act as a container for the defined assets. And you’ve noticed that we have 2 containers of libraries here.

When you include the libraries in the ***\*.info.yml*** (ie: ***yourtheme.info.yml***) file, please note that the naming should be in this format:

|  |  |
| --- | --- |
|  | libraries: |
|  | - yourtheme/global-styling |
|  | - yourtheme/custom-stuff |

[**view raw**](https://gist.github.com/ardnet/410bf79e6268674a48f0/raw/816f64bc3a371c40577741c2a3c1ccd11ace8eff/libraries2.yml)[**libraries2.yml**](https://gist.github.com/ardnet/410bf79e6268674a48f0#file-libraries2-yml) hosted with  by **[GitHub](https://github.com/)**

where *yourtheme* is the name of your custom theme followed by / (slash), and *global-styling*and *custom-stuff* is the name of the libraries that you want to include (see **a.*yourtheme.info.yml*** section above).

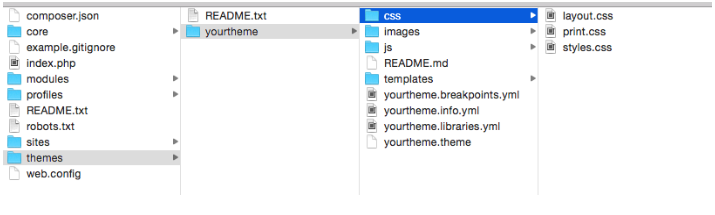
You can define as many containers as you need.

Each library’s container follows a similar pattern, as explained below:

|  |  |
| --- | --- |
|  | css: |
|  | theme: |
|  | css/styles.css: {} |
|  | css/layout.css: {} |
|  | css/print.css: { media: print } |

[**view raw**](https://gist.github.com/ardnet/410bf79e6268674a48f0/raw/816f64bc3a371c40577741c2a3c1ccd11ace8eff/libraries-css.yml)[**libraries-css.yml**](https://gist.github.com/ardnet/410bf79e6268674a48f0#file-libraries-css-yml) hosted with  by **[GitHub](https://github.com/)**

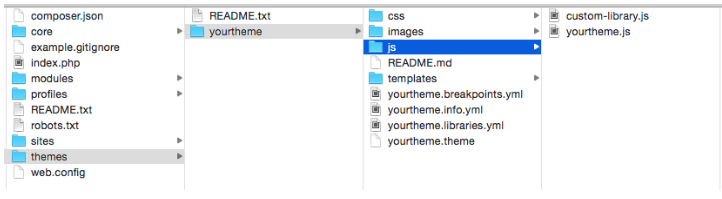
Above is a basic setup where we defined 3 css files, all of them located in the sub-folder called *css* under the *yourtheme* (see screenshot below for reference). The last one is set specifically to one type of media which is *print*.

[](http://x-team.com/wp-content/uploads/2015/03/Screen-Shot-2015-03-26-at-16.20.14.png)

|  |  |
| --- | --- |
|  | js: |
|  | js/yourtheme.js: {} |
|  | js/custom-library.js: {} |

[**view raw**](https://gist.github.com/ardnet/410bf79e6268674a48f0/raw/816f64bc3a371c40577741c2a3c1ccd11ace8eff/libraries-js.yml)[**libraries-js.yml**](https://gist.github.com/ardnet/410bf79e6268674a48f0#file-libraries-js-yml) hosted with  by **[GitHub](https://github.com/)**

JavaScript files are set up similarly to css files. Here we defined 2 javascript files located in the sub-folder of *js* under *yourtheme* (see screenshot below for reference).

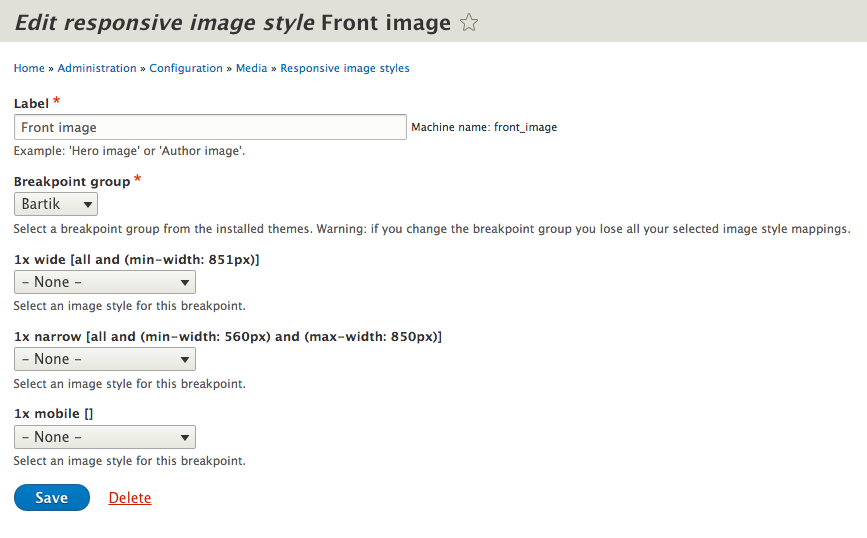
[](http://x-team.com/wp-content/uploads/2015/03/Screen-Shot-2015-03-26-at-16.23.43.png)

***c.yourtheme.breakpoints.yml***

*Breakpoints* are basically the point where a website will adjust the theme or layout based on the browser widths by using [*media query* in css](http://www.w3schools.com/cssref/css3_pr_mediaquery.asp).

The purpose of this configuration file is to define the *breakpoints* for your custom theme, so other (contributed/custom) modules or/and theme that has features and/or functionality related with the *breakpoints* will be able to access the configuration and use it accordingly.

For example, with the [*Responsive image*](https://www.drupal.org/project/responsive_images) module, the module uses the breakpoints that are defined in the custom theme. Below is a screenshot of the admin page of the *Responsive image* module where you can specify *Image styles* based on the defined breakpoints in the custom theme.

[](http://x-team.com/wp-content/uploads/2015/04/Screen-Shot-2015-04-03-at-23.18.25.png)

In order to setup the breakpoints configuration, you need to create a file called *yourtheme.breakpoints.yml*, with basic code as shown below:

|  |  |
| --- | --- |
|  | yourtheme.mobile: |
|  | label: mobile |
|  | mediaQuery: '(min-width: 0px)' |
|  | weight: 2 |
|  | multipliers: |
|  | - 1x |
|  | yourtheme.narrow: |
|  | label: narrow |
|  | mediaQuery: 'all and (min-width: 560px) and (max-width: 860px)' |
|  | weight: 1 |
|  | multipliers: |
|  | - 1x |
|  | yourtheme.wide: |
|  | label: wide |
|  | mediaQuery: 'all and (min-width: 851px)' |
|  | weight: 0 |
|  | multipliers: |
|  | - 1x |

[**view raw**](https://gist.github.com/ardnet/410bf79e6268674a48f0/raw/816f64bc3a371c40577741c2a3c1ccd11ace8eff/breakpoints.yml)[**breakpoints.yml**](https://gist.github.com/ardnet/410bf79e6268674a48f0#file-breakpoints-yml) hosted with  by **[GitHub](https://github.com/)**

Each section represents one breakpoint, which consists of a unique machine name (ie:*yourtheme.mobile*,*yourtheme.narrow*,*yourtheme.wide*), with properties explained below:

* label  
  A human readable label for the breakpoint.
* mediaQuery  
  Media query [proper syntax](http://www.w3schools.com/cssref/css3_pr_mediaquery.asp).
* weight  
  Order for the breakpoints to take place.
* multipliers  
  Supported pixel resolution multipliers.

For more info about breakpoints configuration for a custom theme, please refer to this[page](https://www.drupal.org/documentation/modules/breakpoint).

***d. yourtheme.theme***

This file contains the *hookpreprocess* to alter or make modification to the theme. This file serves the same purpose as *template.php*in Drupal 7.

Below is a simple example of code on how to add custom assets library to our custom theme.

**function** **yourtheme\_preprocess\_page**(&$variables) {

$variables['#attached']['library'] = **array**('yourtheme/global-styling', 'yourtheme/custom-stuff');

}

As you can see we use *hook\_preprocess\_page()*, and we add *global-styling* and *custom-stuff* via the *#attached library* array that comes from *$variables*.

***e. Twig template file (\*.html.twig)***

Another significant change introduced in Drupal 8 is the use of the popular PHP templating engine Twig, to replace the PHPTemplate-based theme system in Drupal 7.

For example, if you used to have *page.tpl.php* in Drupal 7, it has now been replaced by*page.html.twig* in Drupal 8.

For the sake of easy management, Twig template files are put under the *templates* sub-directory from your custom theme.

Here is a list of possible Twig template files that are available in Drupal 8 by default.

**HTML (template)**

* *html.html.twig*

**Page template**

* *page.html.twig*
* *page–node.html.twig*
* *page–node–<nid>.html.twig,***ie:** *page–node–1.html.twig*
* *page–node–edit.html.twig*

**Regions**

* *region.html.twig*
* *region–<region-name>.html.twig,***ie:** *region–right-sidebar.html.twig*

**Blocks**

* *block–<module-name>.html.twig,***ie:** *block–cust-mod.html.twig*
* *block–<module-name>–<delta>.html.twig,***ie:** *block–cust-mod–1.html.twig*

**Nodes**

* *node.html.twig*
* *node–<content-type>.html.twig,***ie:** *node–article.html.twig*
* *node–<nid>.html.twig,***ie:** *node–<nid>.html.twig*

**Taxonomy terms**

* *taxonomy-term.html.twig*
* *taxonomy-term–<vocab-machine-name>.html.twig*
* *node–<nid>.html.twig,***ie:** *node–<nid>.html.twig,***ie:***taxonomy-term–ingredients.html.twig*
* *taxonomy-term–<tid>.html.twig,***ie:***taxonomy-term–1.html.twig*

**Fields**

* *field–<field-type>.html.twig,***ie:***field–long-text.html.twig*
* *field–<field-name>.html.twig,***ie:***field–phone-number.html.twig*

**Comments**

* *comment.html.twig*
* *comment-wrapper.html.twig*

**Forums**

For forum containers:

* *forums–containers.html.twig*
* *forums–<forum-id>.html.twig*
* *forums–containers–<forum-id>.html.twig*

For forum topics:

* *forums–topics.html.twig*
* *forums–<forum-id>.html.twig*
* *forums–containers–<forum-id>.html.twig*

**Maintenance page**

* *maintenance-page.html.twig*

**Search result**

* *search-result.html.twig*

In terms of coding, Twig template files in Drupal 8 are not much different to PHPTemplate files in Drupal 7. In both cases you’re essentially writing HTML. The difference is how you define variables and define conditional logic within it.

Here are some examples:

**Docblock**  
**PHPTemplate**

<?php

/\*\*

\* **@file**

\* File description

\*/

?>

**Twig**

{#

/\*\*

\* **@file**

\* File description

\*/

#}

**Variables**  
Printing a variable:  
**PHPTemplate:** <div class="content"><?php print $content; ?></div>  
**Twig:** <div class="content">{{ content }}</div>

Printing a hash key item:  
**PHPTemplate:** <?php print $item['#item']['alt']; ?>  
**Twig:** {{ item['#item'].alt }}

Assigning a variable:  
**PHPTemplate:** <?php $custom\_var = $content->comments; ?>  
**Twig:** {% set custom\_var = content.comments %}

Assigning an array:  
**PHPTemplate:** <?php $args = array('!author' => $author, '!date' => $created); ?>  
**Twig:** {% set args = {'!author': author, '!date': created} %}

**Conditionals:**  
**PHPTemplate:** <?php if ($content->comments): endif; ?>  
**Twig:** {% if content.comments %} {% endif %}

**PHPTemplate:** <?php if (!empty($content->comments)): endif; ?>  
**Twig:** {% if content.comments is not empty %} {% endif %}

**PHPTemplate:** <?php if (isset($content->comments)): endif; ?>  
**Twig:** {% if content.comments is defined %} {% endif %}

**PHPTemplate:** <?php if ($count > 0): endif; ?>  
**Twig:** {% if count > 0 %} {% endif %}

**Control structures:**  
**PHPTemplate:** <?php foreach ($users as $user) {} ?>  
**Twig:** {% for user in users %} {% endfor %}

**Filters:**  
Check\_plain:  
**PHPTemplate:** <?php print check\_plain($title); ?>  
**Twig:** {{ title|striptags }}

Translate:  
**PHPTemplate:** <?php print t('Home'); ?>  
**Twig:** {{ 'Home'|t }}

Translate with substitutions:  
**PHPTemplate:** <?php print t('Welcome, @username', array('@username' => $user->name)); ?>  
**Twig:** {{ 'Welcome, @username'|t({ '@username': user.name }) }}

For more examples please refer to the [Twig homepage.](http://twig.sensiolabs.org/)

And that wraps it up! That should be enough to know about the basic changes to theming in Drupal 8, and enough to get you started on your own exploration and experimentation.