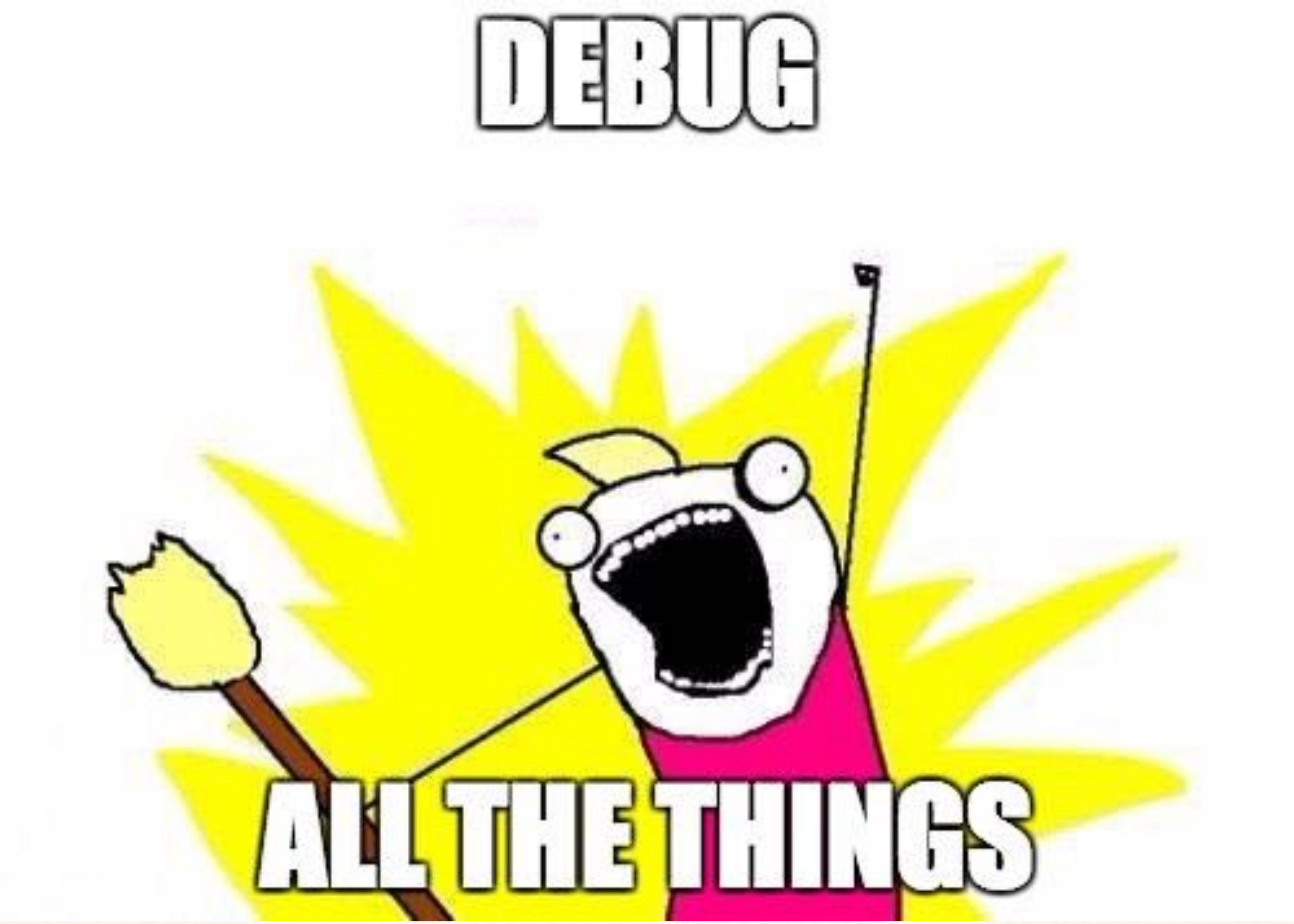
Intro to Drupal 8 Theming - Covering (and debugging) all the bases of the new D8 theming system with Twig

Lee Walker

Code Journeymen

Otherwise Know as….



Principal Engineer

Lee Walker

@drupalnooga

lee@codej.us

www.codejourneymen.com   www.codej.us

https://www.drupal.org/u/mr_scumbag

Organizer of Drupal Camp Chattanooga.

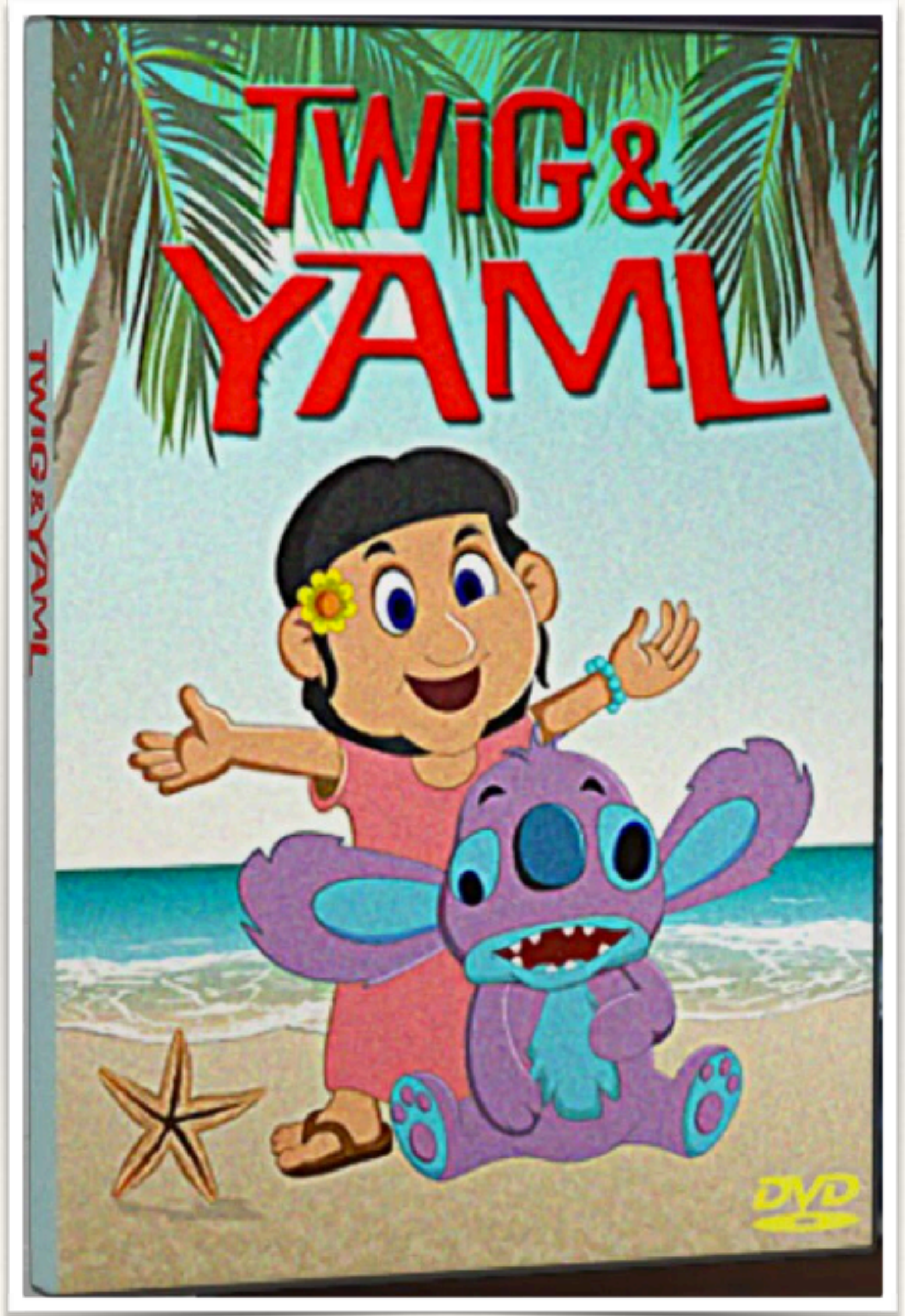
What are trying to accomplish?

- Create a theme from scratch using Bootstrap V3.
  - Not a sub theme - we want to see how it all fits together.
- Adding CSS & Javascript libraries.
- Twig templates - where to find them.
- Twig templates - what to call them - including custom.
- Twig templates - adding your own custom data.
- YAML what files used in theming.
- Debugging the whole darn thing….. <- This is really important

Is this what you see?

Twig & YAML

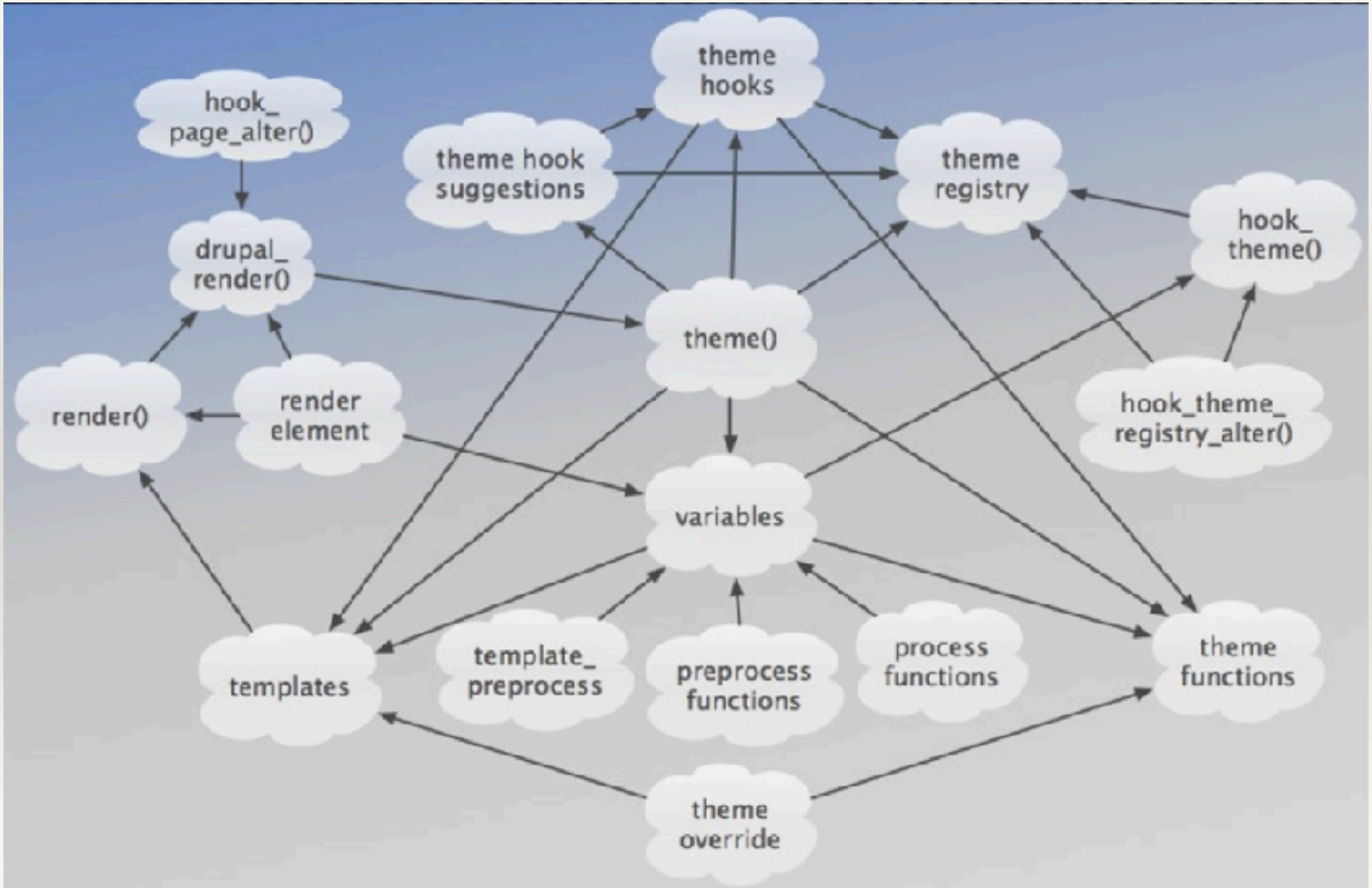
(Straight to DVD)



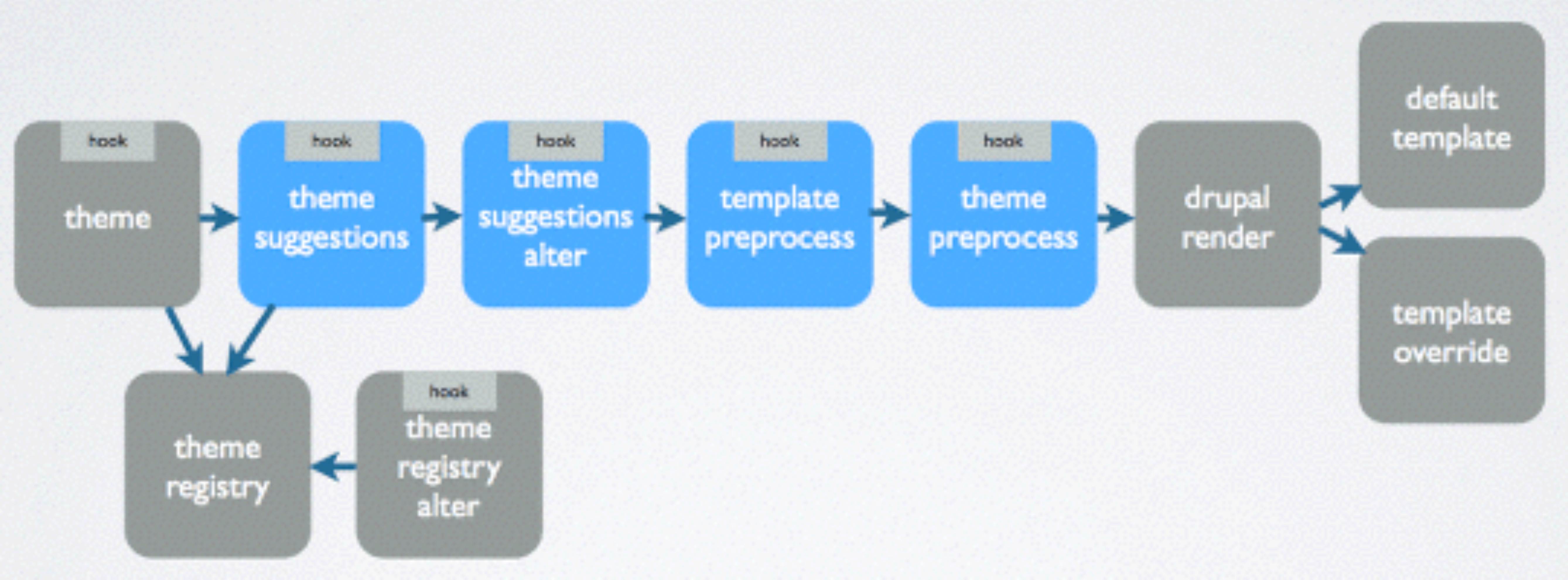
Where To Start?

- What twig templates to use? Create new or override? How do we name our templates?
- How do we preprocess our template data - ie add extra things into our templates?
- How do we use our own template name suggestions?
- Printing/showing debug twig variables
- Breakpoints in Twig?

Why is D8 theming better?



Why is D8 theming better?

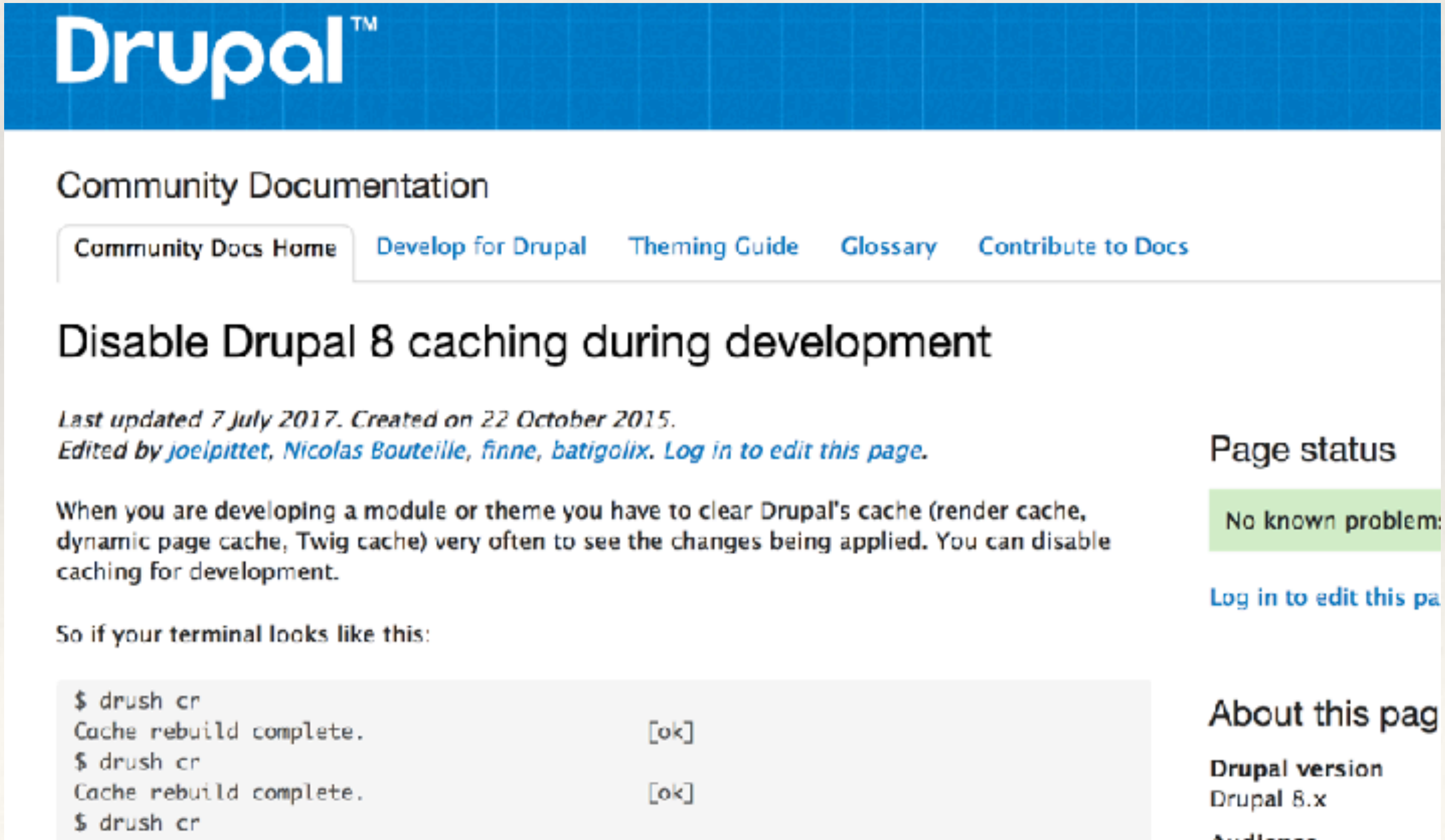


Where To Start?

- Turn on all the debugging you can for development.
- Disable CSS/JS aggregation
- Disable all caching - D8 all on by default
    - $ drush cr… $drush cr…. $ drush cr — no more of this
- Web profiler from Devel module

Debugging ON & Caching OFF

- Definitive guide now at https://www.drupal.org/node/2598914
- (Just turning on twig debugging isn't enough)



Community Documentation

Disable Drupal 8 caching during development

Last updated 7 July 2017. Created on 22 October 2015.
Edited by joelpittet, Nicolas Bouteille, finne, batigolix. Log in to edit this page.

When you are developing a module or theme you have to clear Drupal's cache (render cache, dynamic page cache, Twig cache) very often to see the changes being applied. You can disable caching for development.

So if your terminal looks like this:

```
$ drush cr
Cache rebuild complete.                         [ok]
$ drush cr
Cache rebuild complete.                         [ok]
$ drush cr
```

Page status

No known problem

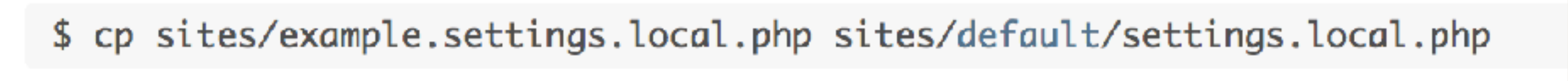
Log in to edit this pa

About this pag

Drupal version
Drupal 8.x

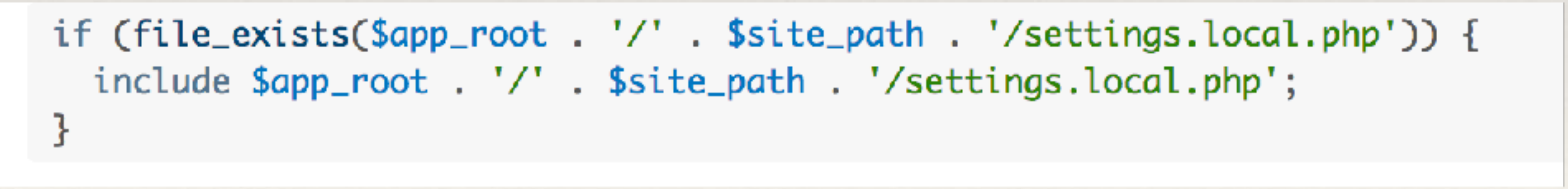
where to get settings.local.php

A default one is in core



```
$ cp sites/example.settings.local.php sites/default/settings.local.php
```

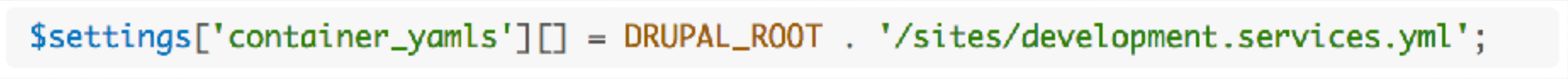
Edit **settings.php** to load your **settings.local.php** by uncommenting these lines..



```
if (file_exists($app_root . '/' . $site_path . '/settings.local.php')) {
  include $app_root . '/' . $site_path . '/settings.local.php';
}
```

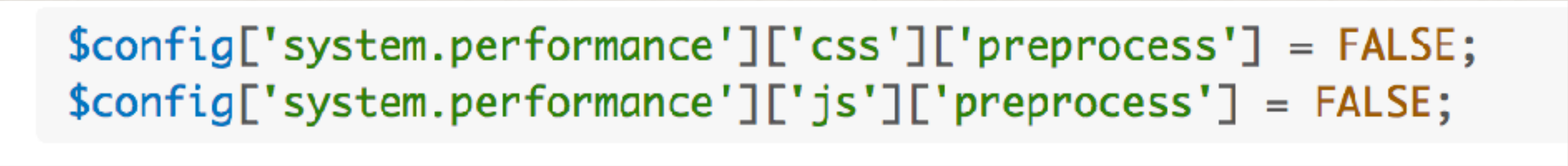
settings.local.php

Uses development services file. (Already configured)



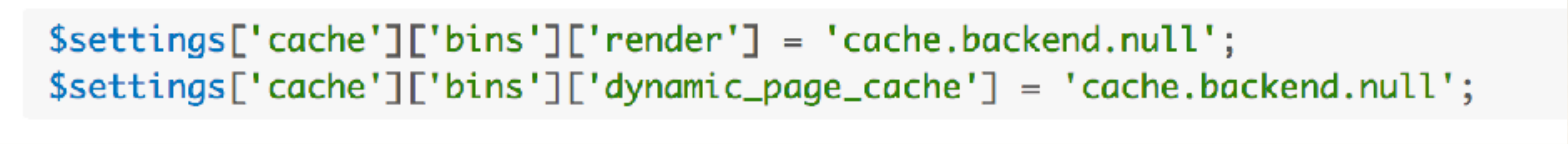
```
$settings['container_yamls'][] = DRUPAL_ROOT . '/sites/development.services.yml';
```

Turn off js/css aggregation. (Already configured)



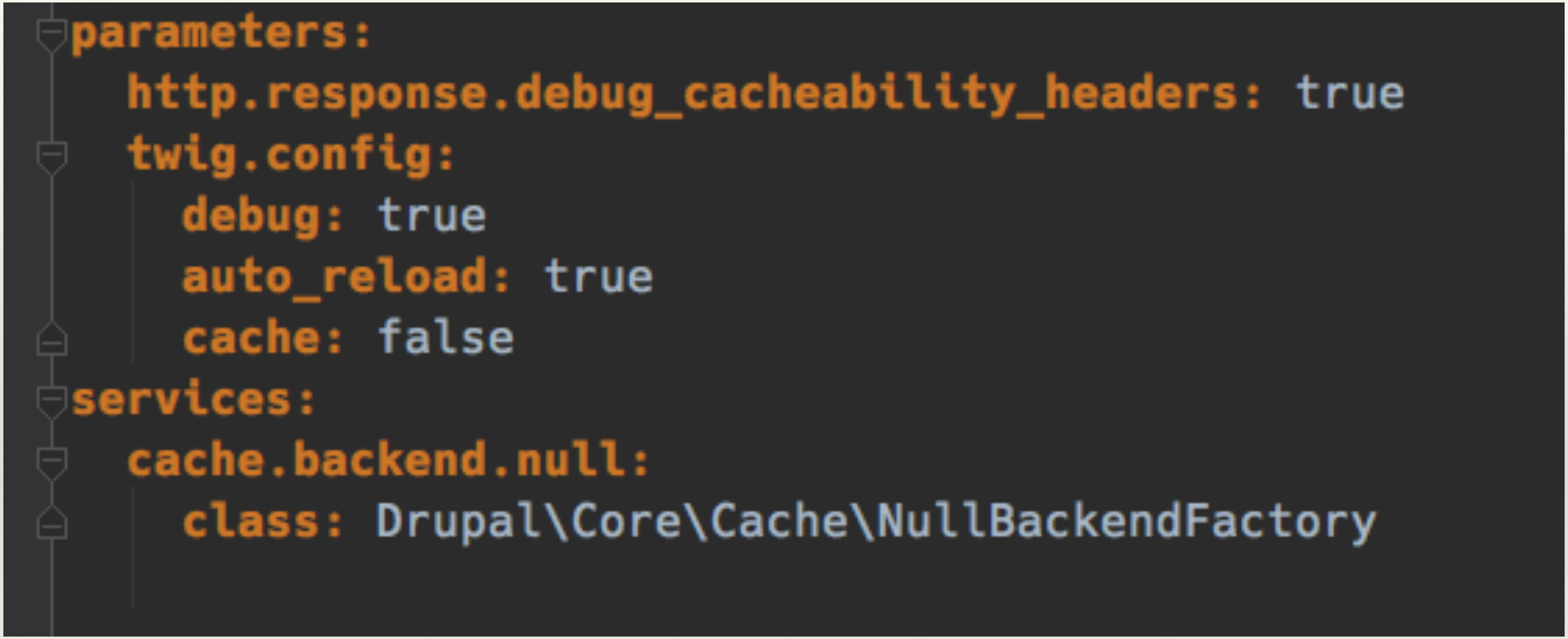
```
$config['system.performance']['css']['preprocess'] = FALSE;
$config['system.performance']['js']['preprocess'] = FALSE;
```

Turn off Drupal 8 caching. (Will need to uncomment these lines)



```
$settings['cache']['bins']['render'] = 'cache.backend.null';
$settings['cache']['bins']['dynamic_page_cache'] = 'cache.backend.null';
```

development.services.yml

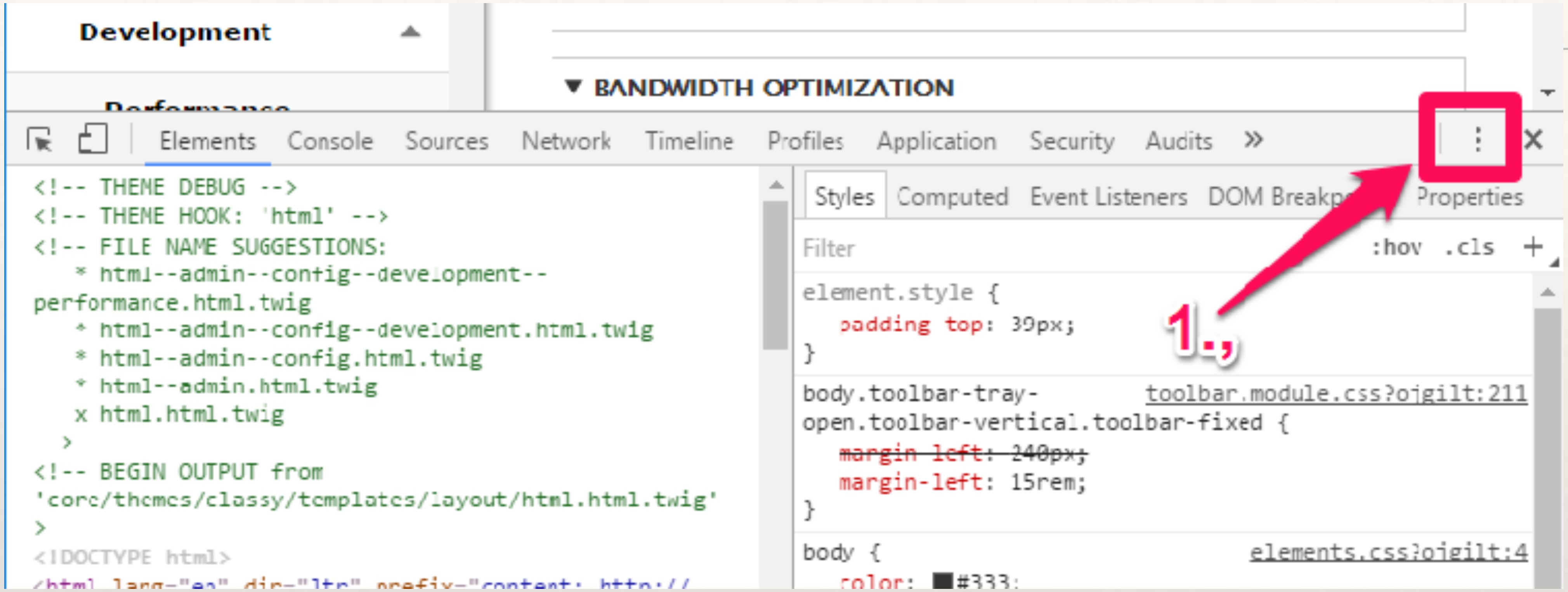


```yaml
parameters:
  http.response.debug_cacheability_headers: true
  twig.config:
    debug: true
    auto_reload: true
    cache: false
services:
  cache.backend.null:
    class: Drupal\Core\Cache\NullBackendFactory
```

Disable Caching in your browser

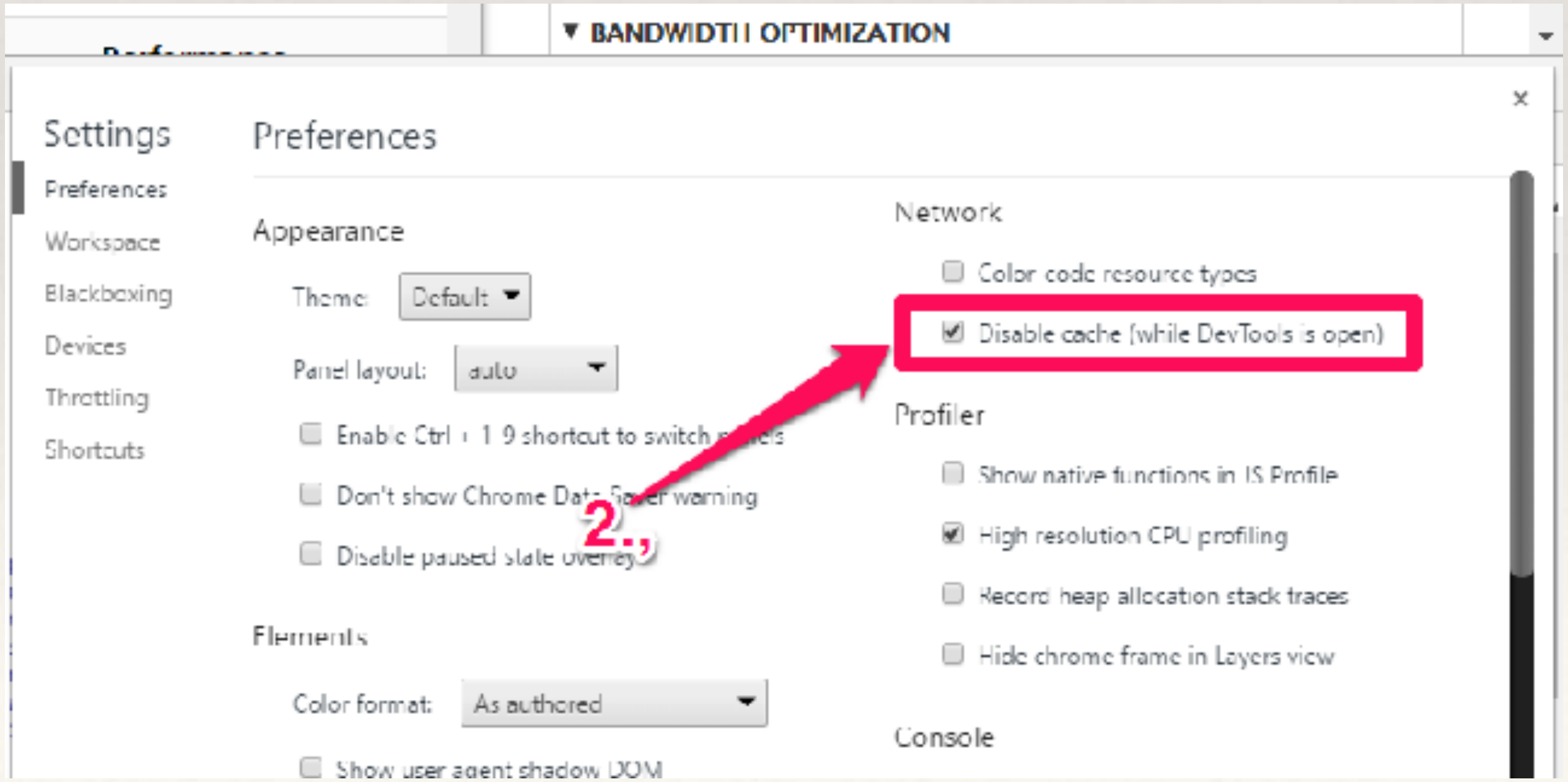
- Why does a reload still not show my CSS/JS changes after I have disabled all Drupal caching.
  Your browser still wants to cache.
- So, after all that, your browser could still be caching….

Disable Chrome Caching



```
<!-- THEME DEBUG -->
<!-- THEME HOOK: 'html' -->
<!-- FILE NAME SUGGESTIONS:
   * html--admin--config--development--
performance.html.twig
   * html--admin--config--development.html.twig
   * html--admin--config.html.twig
   * html--admin.html.twig
   x html.html.twig
  >
<!-- BEGIN OUTPUT from
'core/themes/classy/templates/layout/html.html.twig'
>
<!DOCTYPE html>
<html lang="en" dir="ltr" prefix="content: http://
```

```
element.style {
   padding top: 39px;
}
body.toolbar-tray-          toolbar.module.css?ojgilt:211
open.toolbar-vertical.toolbar-fixed {
   ~~margin-left: 240px;~~
   margin-left: 15rem;
}
body {                           elements.css?ojgilt:4
   color: ■#333;
```



Preferences

Appearance

Theme: Default

Panel layout: auto

□ Enable Ctrl + 1-9 shortcut to switch panels

□ Don't show Chrome Data Saver warning

□ Disable paused state overlay

Elements

Color format: As authored

□ Show user agent shadow DOM

Network

□ Color code resource types

☑ Disable cache (while DevTools is open)

Profiler

□ Show native functions in JS Profile

☑ High resolution CPU profiling

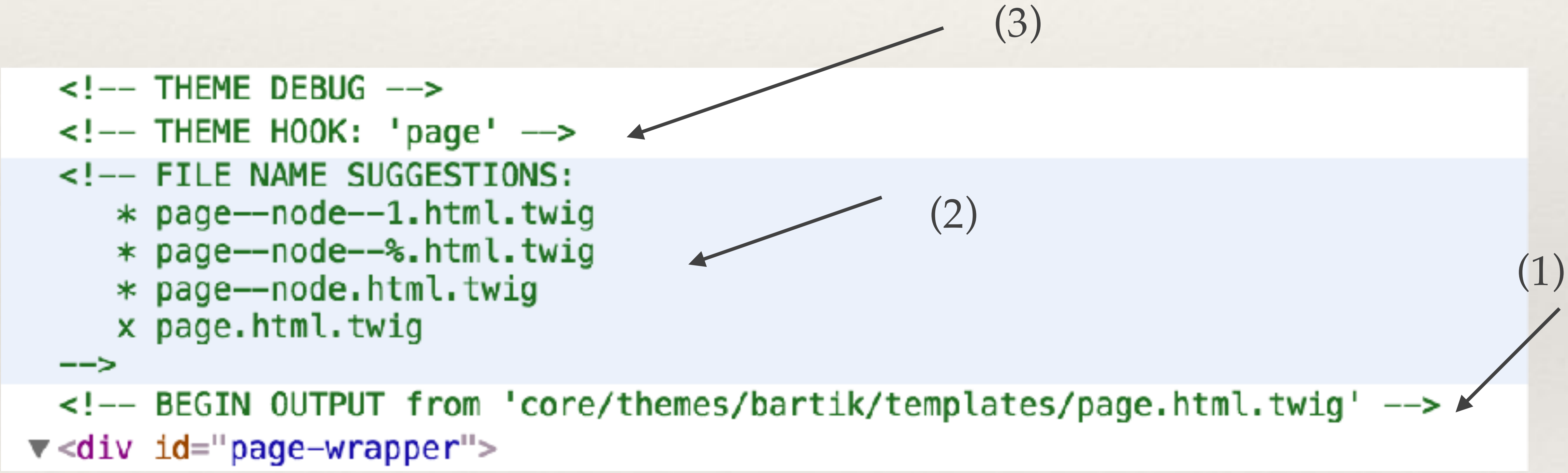
□ Record heap allocation stack traces

□ Hide chrome frame in Layers view

Console

Ensure debugging is on

Reload your front page and check for twig debug information

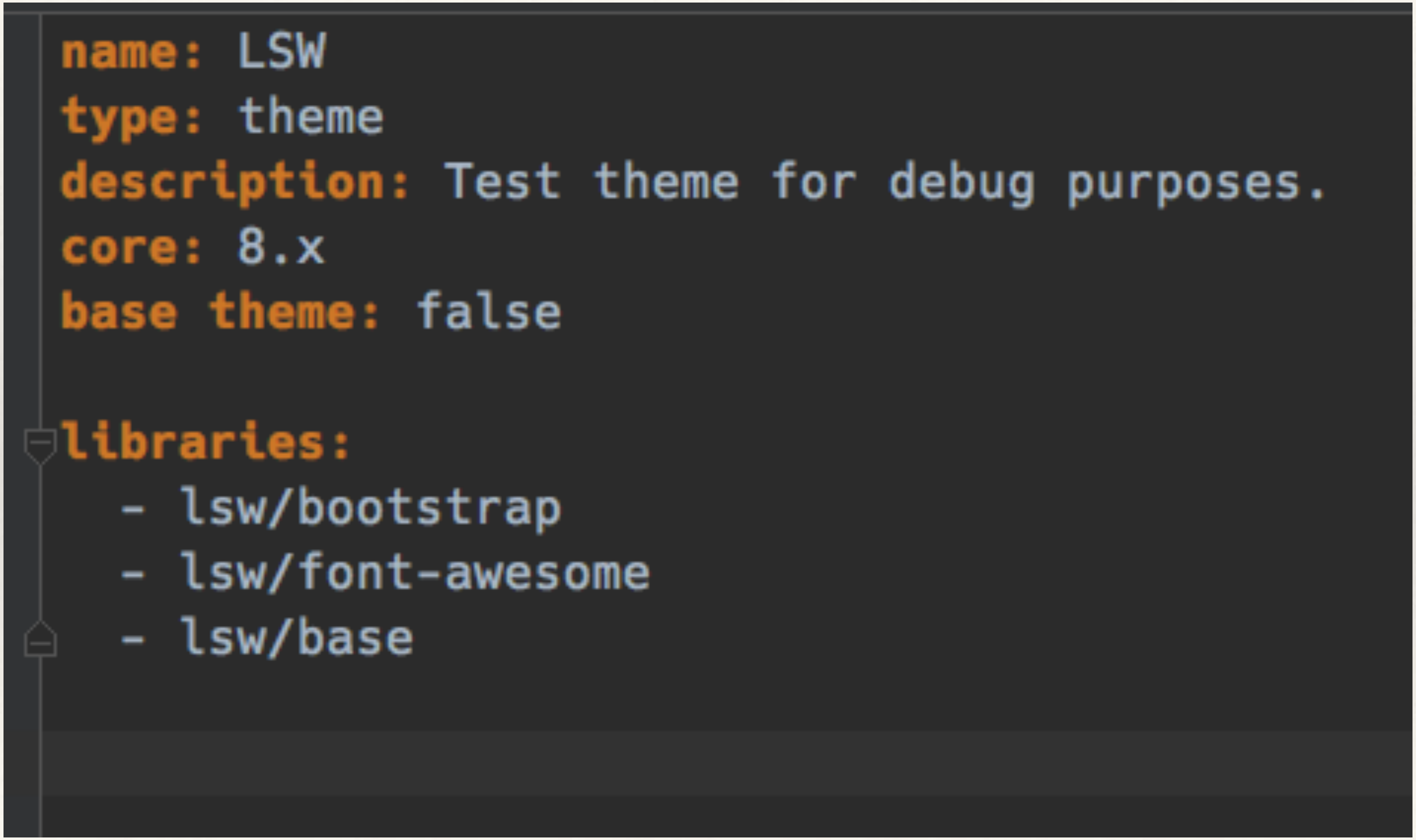


```
<!-- THEME DEBUG -->
<!-- THEME HOOK: 'page' -->
<!-- FILE NAME SUGGESTIONS:
   * page--node--1.html.twig
   * page--node--%.html.twig
   * page--node.html.twig
   x page.html.twig
-->
<!-- BEGIN OUTPUT from 'core/themes/bartik/templates/page.html.twig' -->
▼<div id="page-wrapper">
```

Now lets get started with a theme

- Theme will be called ‘LSW’
- ‘lsw.info.yml’ definition file.
- Add bootstrap 3 as a library. In a file called ‘lsw.libraries.yml’
- Add our own CSS and JS files - as a library too. In libraries.yml
- Override a template.
- Use a theme preprocess function to add new variables.
- Define new regions & Twig templates to go with them.

lsw.info.yml



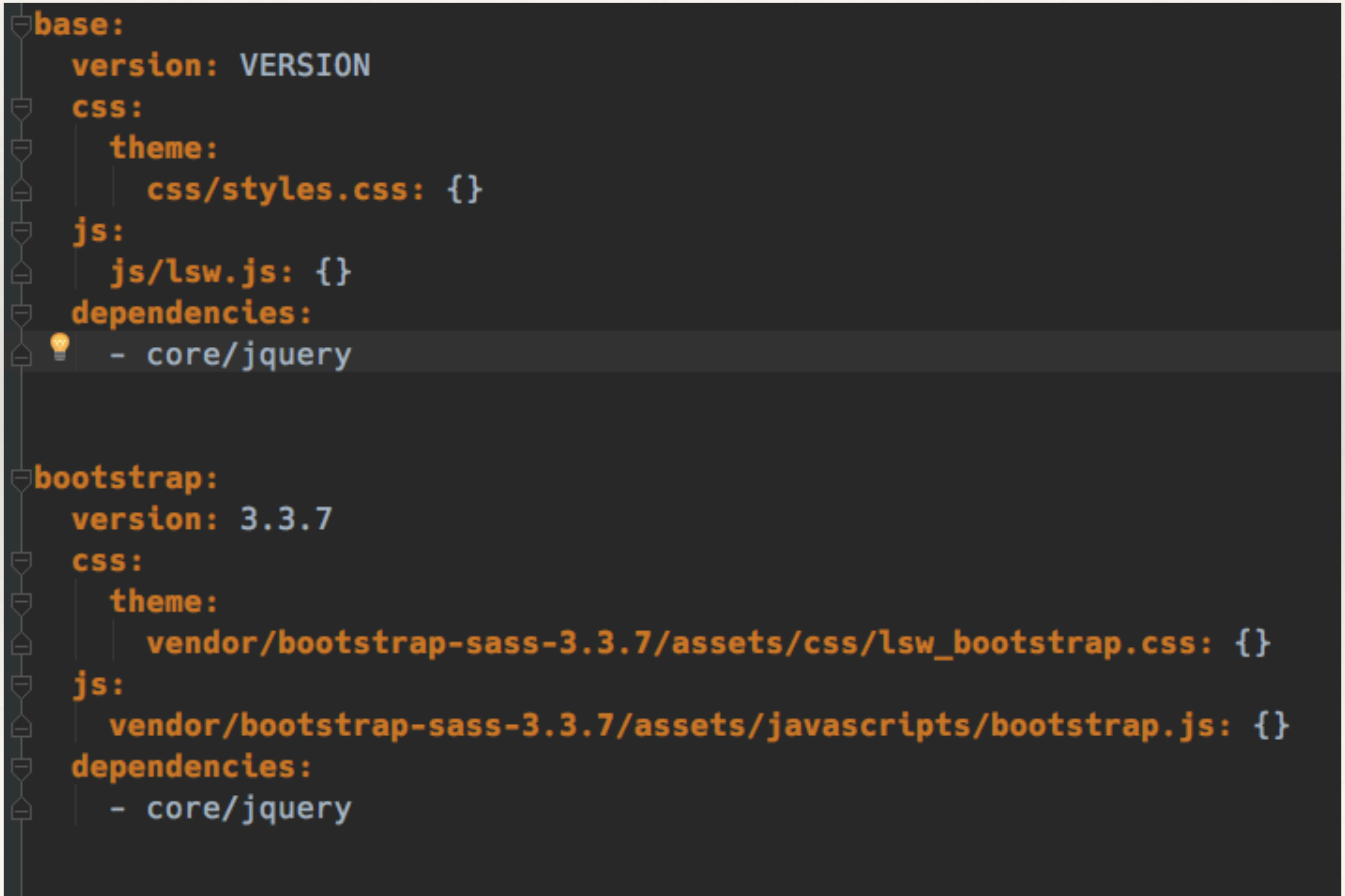
```yaml
name: LSW
type: theme
description: Test theme for debug purposes.
core: 8.x
base theme: false

libraries:
  - lsw/bootstrap
  - lsw/font-awesome
  - lsw/base
```

Base Theme

- base theme: false
    - Means it gets Stable as a base theme. Even though we said no base theme. Stable provides the default theme ‘pieces’ (templates, javascript, css etc) that Drupal will always use. (Sometimes you get Classy templates, because Stable doesn’t supply it)

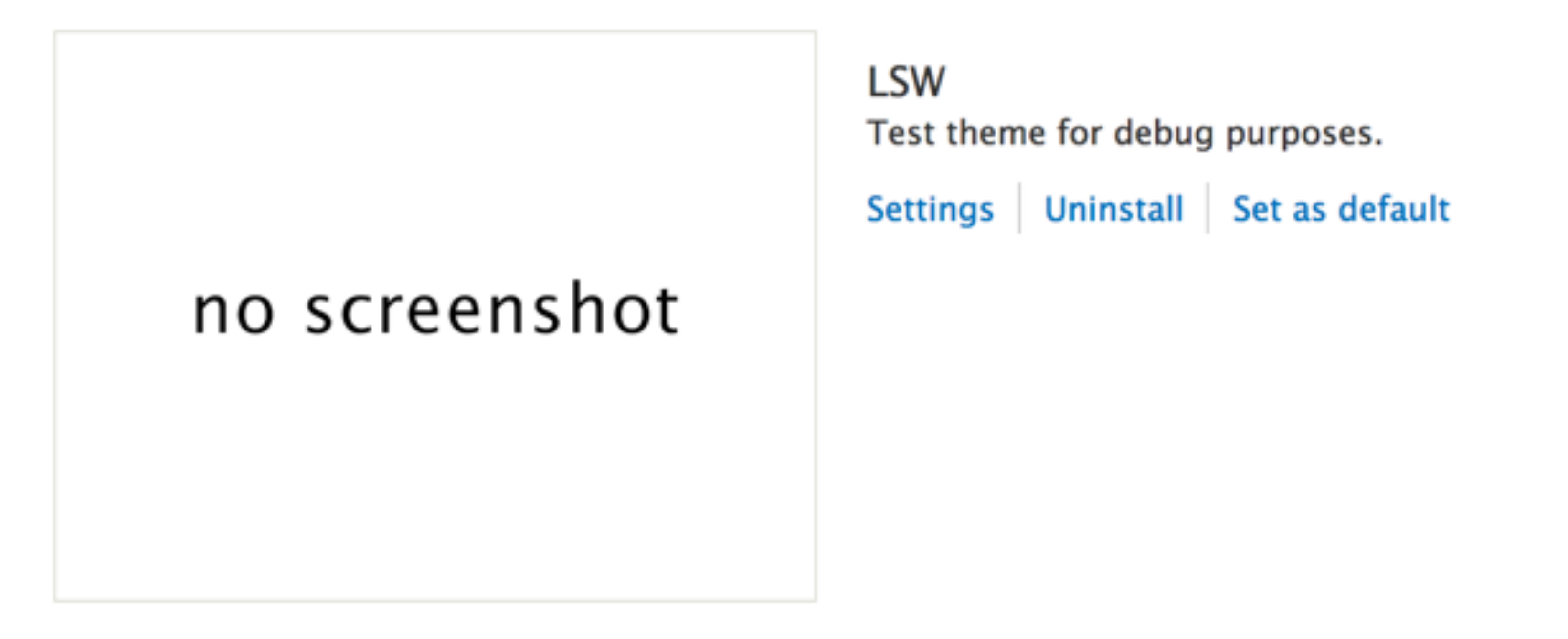
Adding libraries - lsw.libraries.yml

- How Javascript and CSS is added in D8 themes



```yaml
base:
  version: VERSION
  css:
    theme:
      css/styles.css: {}
  js:
    js/lsw.js: {}
  dependencies:
    - core/jquery


bootstrap:
  version: 3.3.7
  css:
    theme:
      vendor/bootstrap-sass-3.3.7/assets/css/lsw_bootstrap.css: {}
  js:
    vendor/bootstrap-sass-3.3.7/assets/javascripts/bootstrap.js: {}
  dependencies:
    - core/jquery
```

We can go ahead and turn this theme on now.



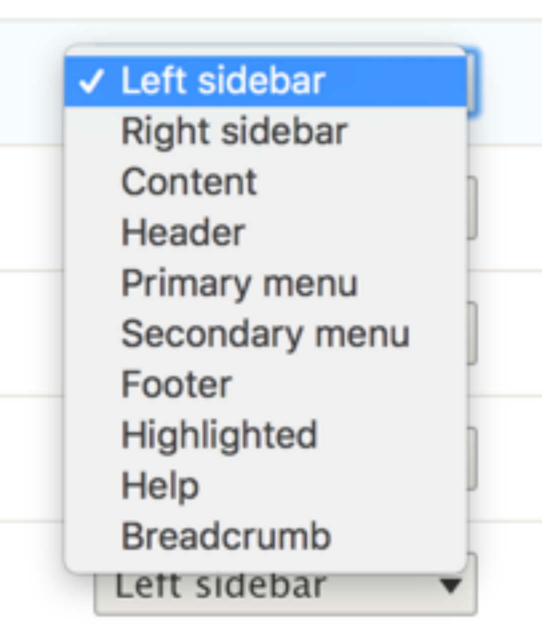
no screenshot

LSW

Test theme for debug purposes.

We have these odd regions now…

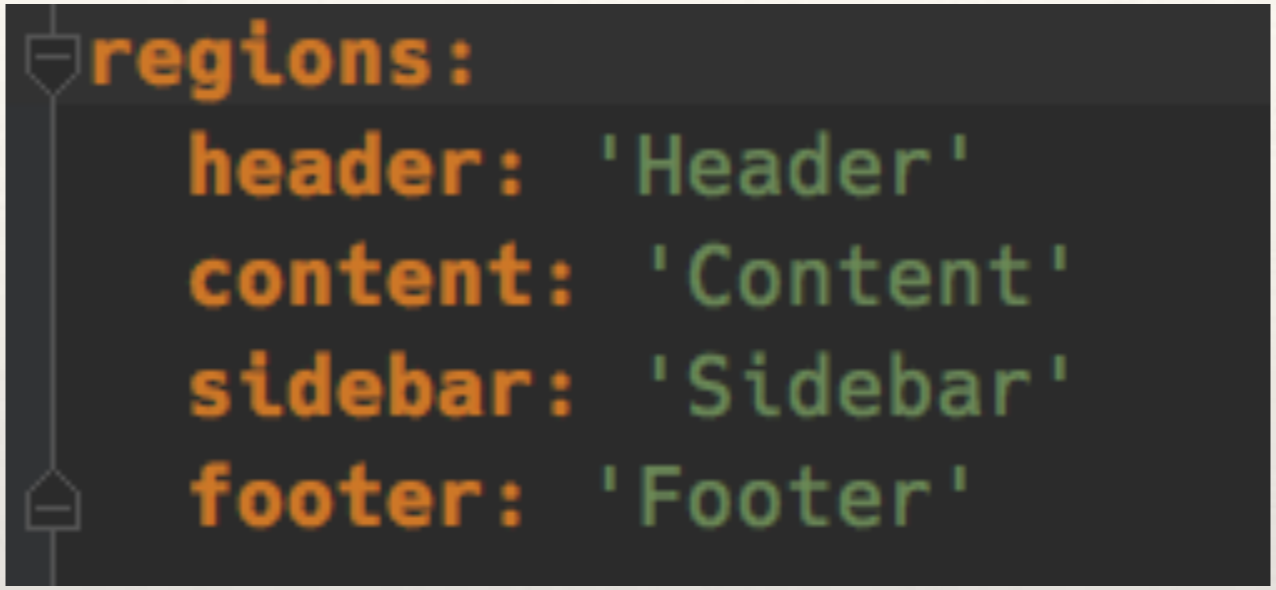
- Because we haven’t defined any - we get the ones from Stable.



- Add our regions to lsw.info.yml - then add region templates.

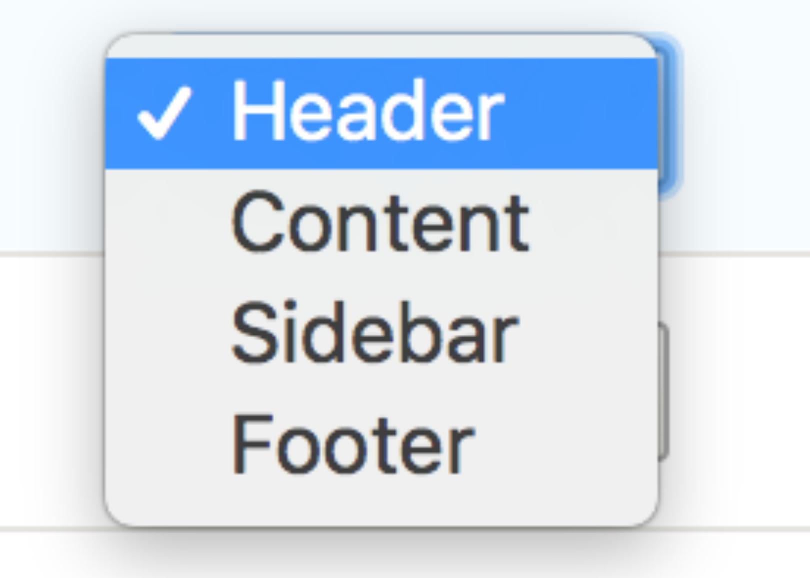
Regions in lsw.info.yml

Just define 4 regions

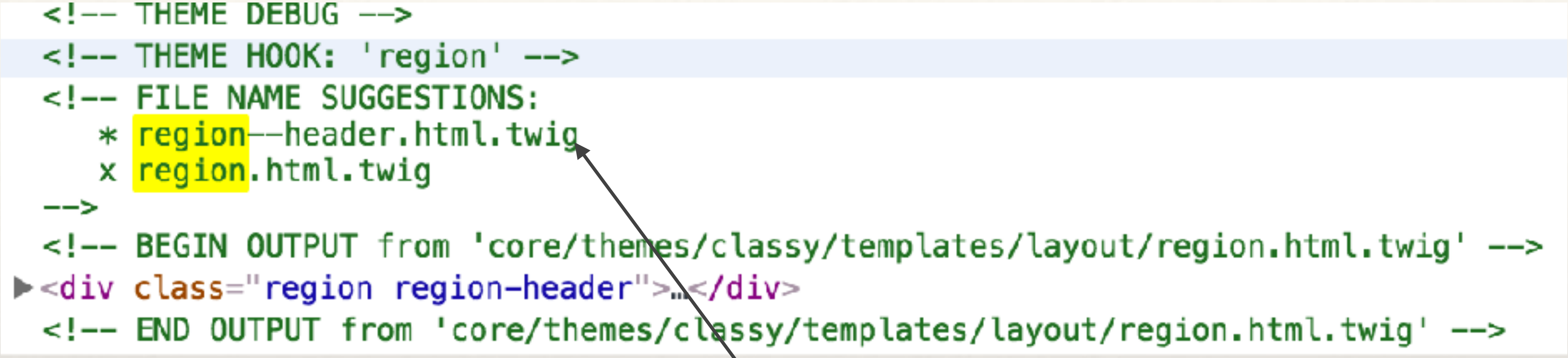


```
regions:
  header: 'Header'
  content: 'Content'
  sidebar: 'Sidebar'
  footer: 'Footer'
```

After a $drush cr, we'll see block layout selections now reflect our desires.

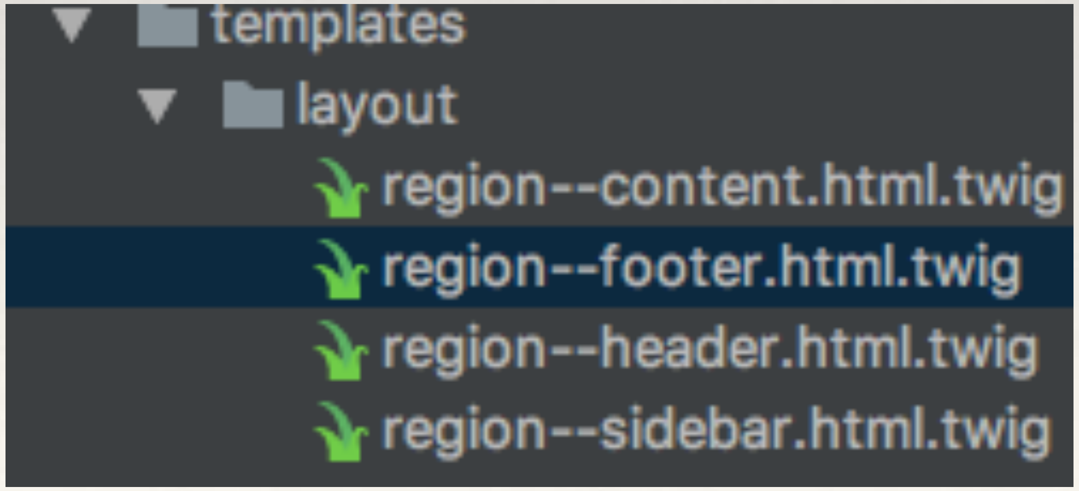


Grab region templates from base and modify (if needed)



```
<!-- THEME DEBUG -->
<!-- THEME HOOK: 'region' -->
<!-- FILE NAME SUGGESTIONS:
   * region--header.html.twig
   x region.html.twig
-->
<!-- BEGIN OUTPUT from 'core/themes/classy/templates/layout/region.html.twig' -->
▶<div class="region region-header">…</div>
<!-- END OUTPUT from 'core/themes/classy/templates/layout/region.html.twig' -->
```

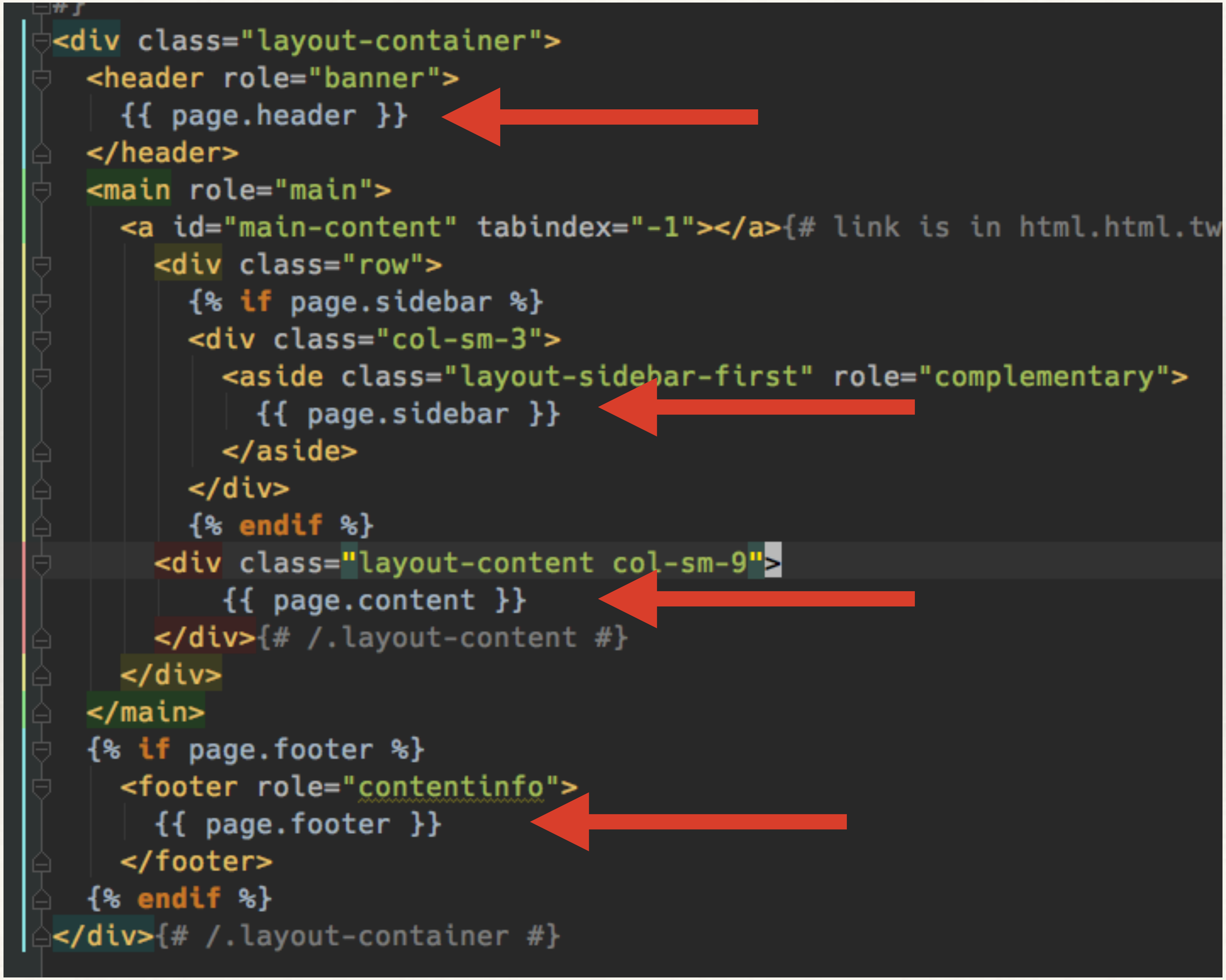
Copy the classy **region.html.twig** to the suggestion for each region.



```
▼ templates
    ▼ layout
        region--content.html.twig
        region--footer.html.twig
        region--header.html.twig
        region--sidebar.html.twig
```

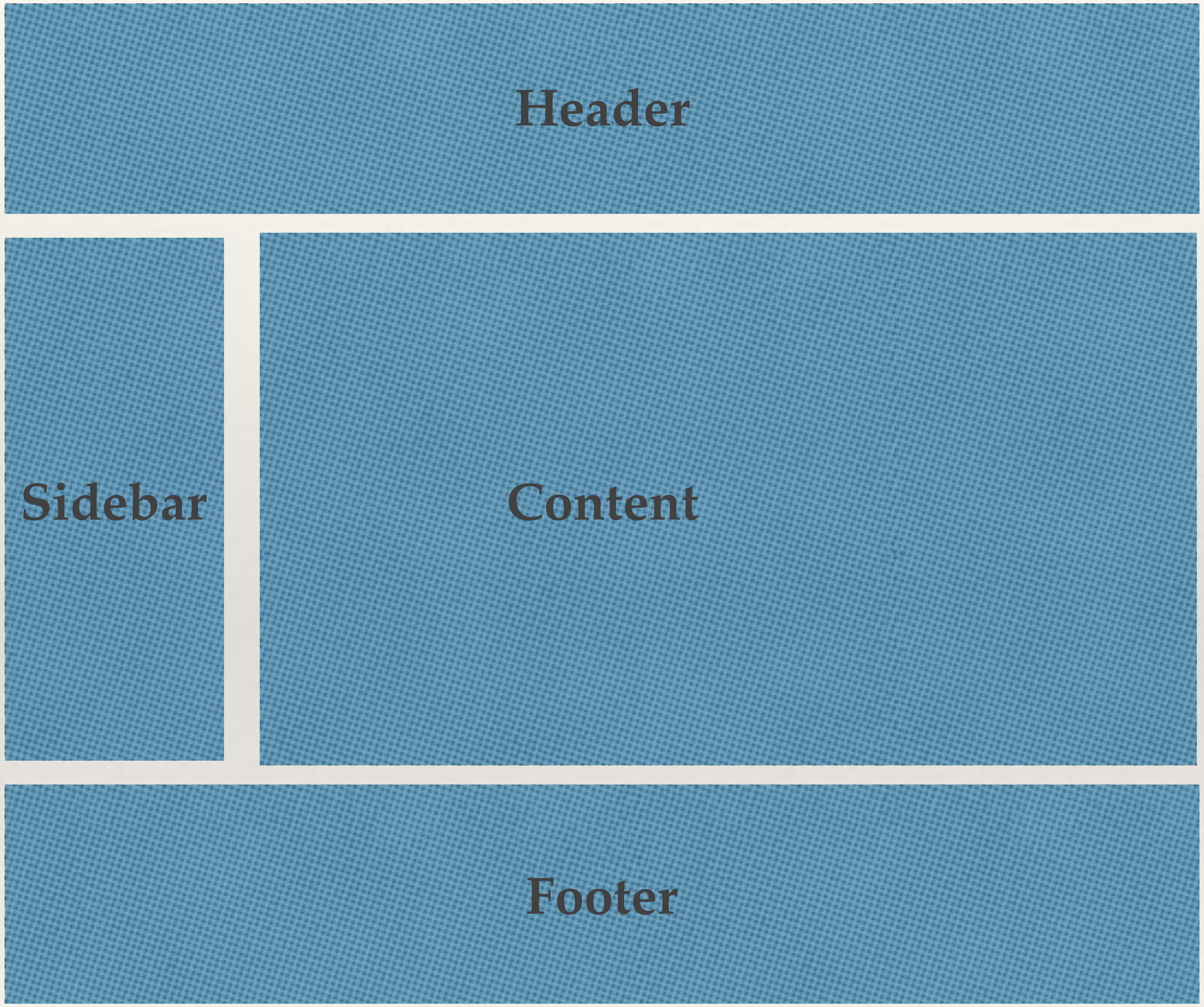
Must define regions in page.html.twig

- Define them in lsw.info.yml
- Must add them to page.html.twig



```
#}
<div class="layout-container">
  <header role="banner">
    {{ page.header }}
  </header>
  <main role="main">
    <a id="main-content" tabindex="-1"></a>{# link is in html.html.tw
      <div class="row">
        {% if page.sidebar %}
        <div class="col-sm-3">
          <aside class="layout-sidebar-first" role="complementary">
            {{ page.sidebar }}
          </aside>
        </div>
        {% endif %}
      <div class="layout-content col-sm-9">
          {{ page.content }}
      </div>{# /.layout-content #}
    </div>
  </main>
  {% if page.footer %}
    <footer role="contentinfo">
      {{ page.footer }}
    </footer>
  {% endif %}
</div>{# /.layout-container #}
```

What our regions will define



Header

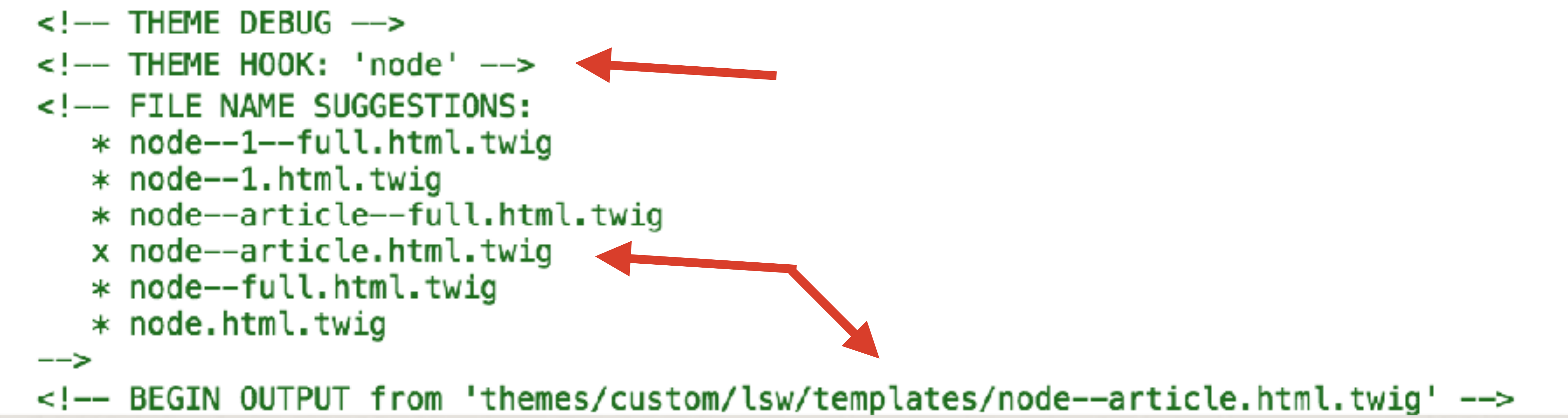
Sidebar

Content

Footer

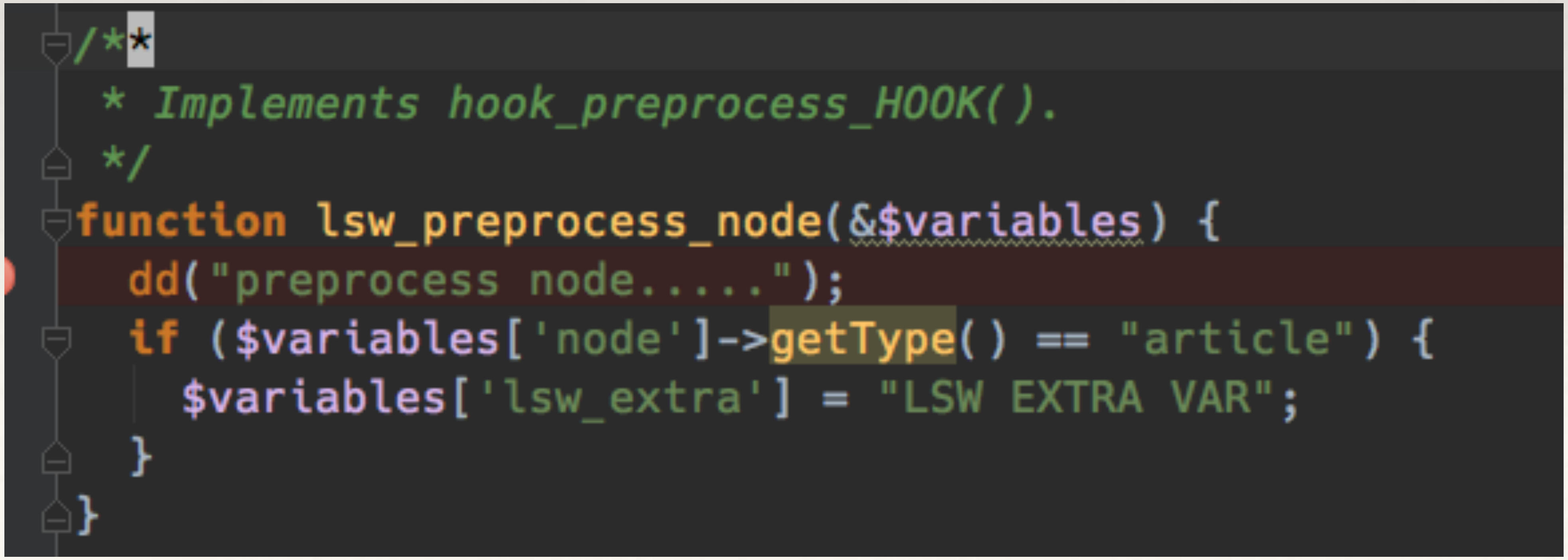
This is set up in page.html.twig

How add custom variables to templates



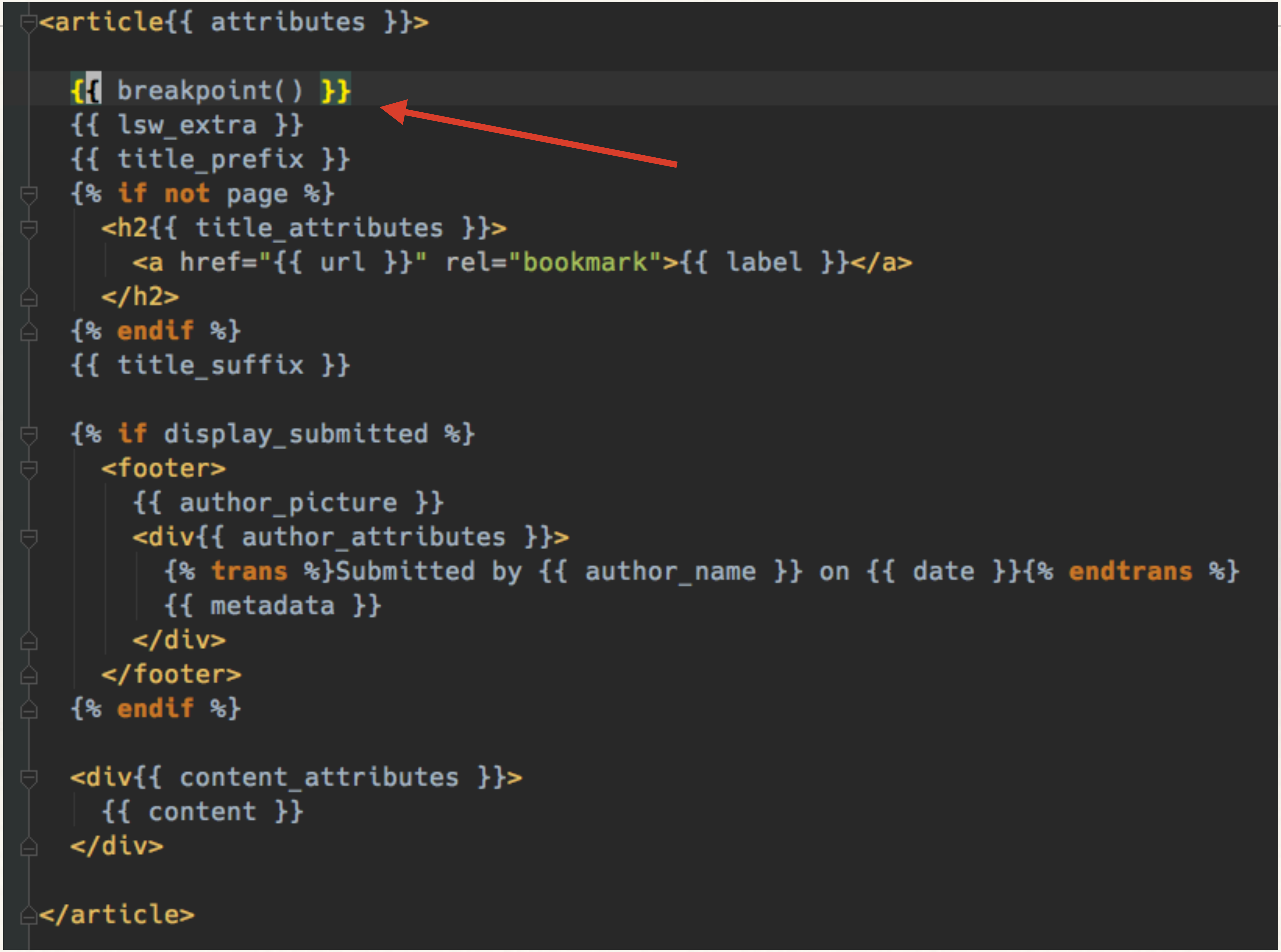
```
<!-- THEME DEBUG -->
<!-- THEME HOOK: 'node' -->
<!-- FILE NAME SUGGESTIONS:
   * node--1--full.html.twig
   * node--1.html.twig
   * node--article--full.html.twig
   x node--article.html.twig
   * node--full.html.twig
   * node.html.twig
-->
<!-- BEGIN OUTPUT from 'themes/custom/lsw/templates/node--article.html.twig' -->
```

hook_preprocess_node() in lsw.theme



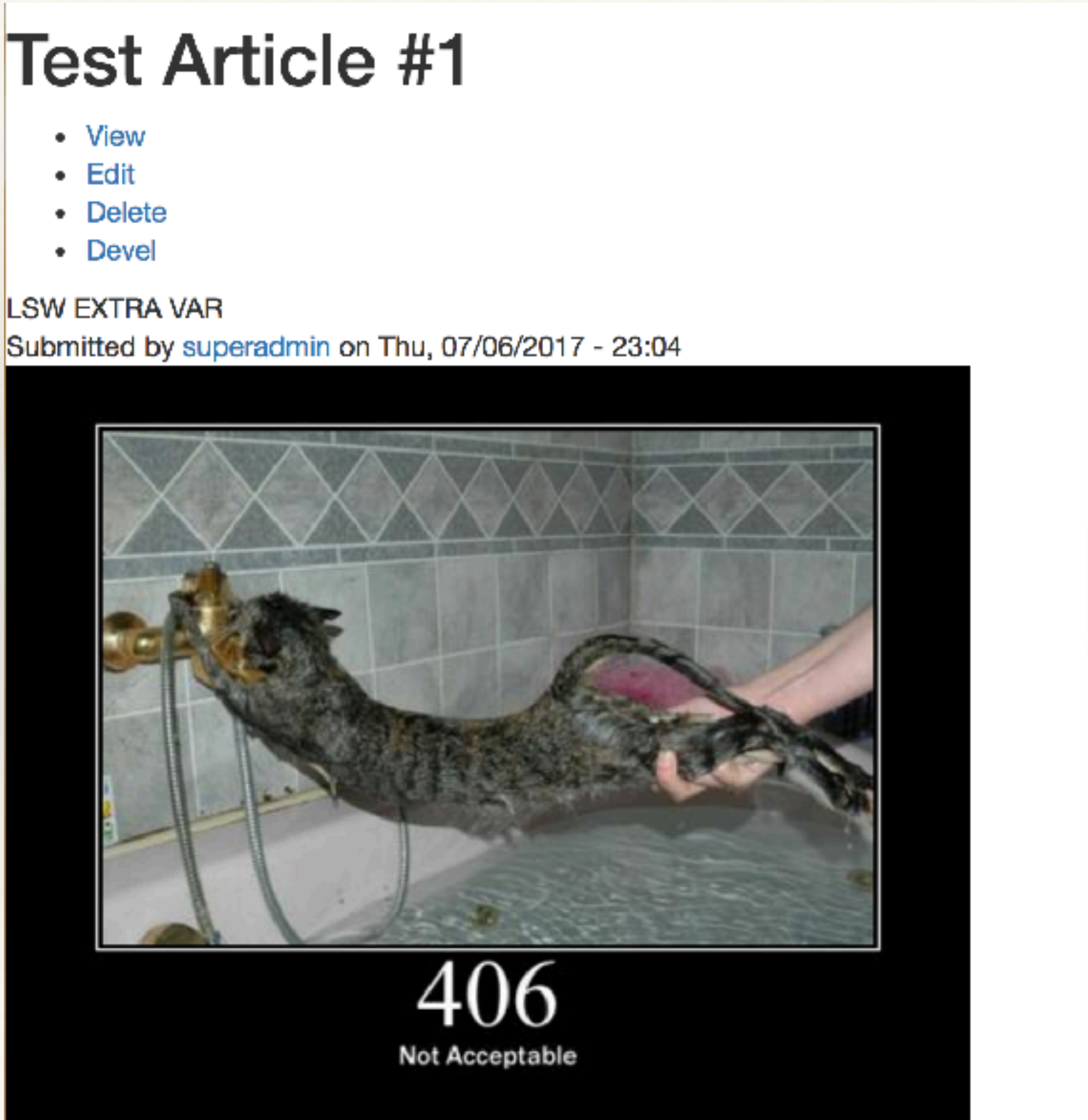
```
/**
 * Implements hook_preprocess_HOOK().
 */
function lsw_preprocess_node(&$variables) {
  dd("preprocess node.....");
  if ($variables['node']->getType() == "article") {
    $variables['lsw_extra'] = "LSW EXTRA VAR";
  }
}
```

Our custom node-article.html.twig



```
<article{{ attributes }}>

  {{ breakpoint() }}
  {{ lsw_extra }}
  {{ title_prefix }}
  {% if not page %}
    <h2{{ title_attributes }}>
      <a href="{{ url }}" rel="bookmark">{{ label }}</a>
    </h2>
  {% endif %}
  {{ title_suffix }}

  {% if display_submitted %}
    <footer>
      {{ author_picture }}
      <div{{ author_attributes }}>
        {% trans %}Submitted by {{ author_name }} on {{ date }}{% endtrans %}
        {{ metadata }}
      </div>
    </footer>
  {% endif %}

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

</article>
```

Viewing the article



Test Article #1

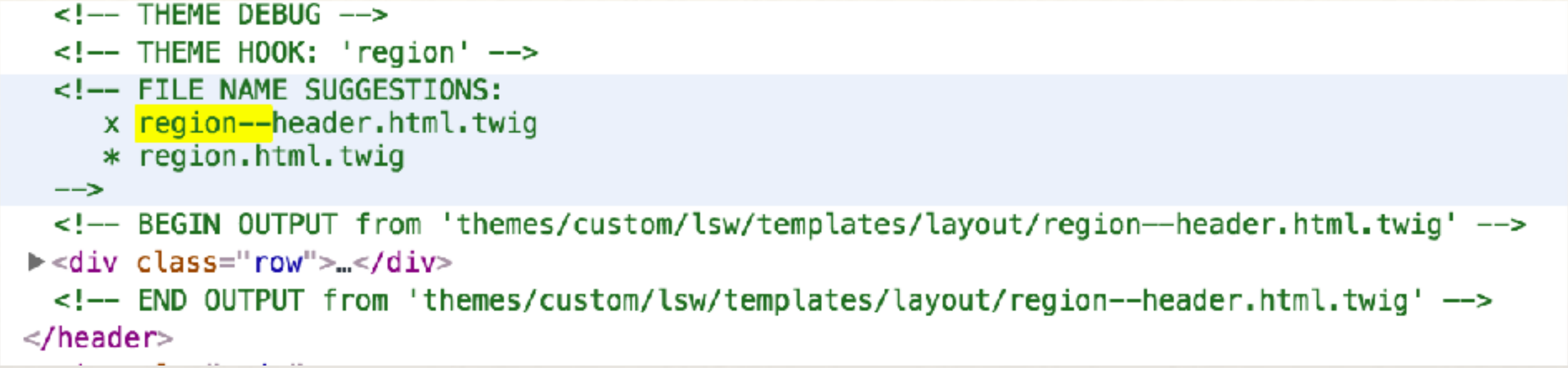
- View
- Edit
- Delete
- Devel

LSW EXTRA VAR
Submitted by superadmin on Thu, 07/06/2017 - 23:04

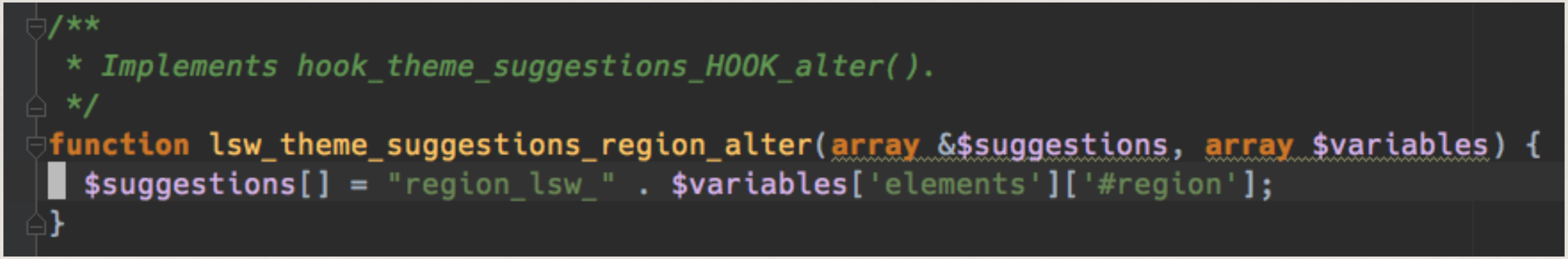
How to add theme suggestions

- Use `hook_theme_suggestions_HOOK_alter()`
- Views doesn’t show (many) suggestions, so add your own custom with
  - hook_theme_suggestions_views_view_alter()
  - $suggestions[] = ‘views_view__’ . $your_info;
- Or for blocks :-
  - hook_theme_suggestions_block_alter()

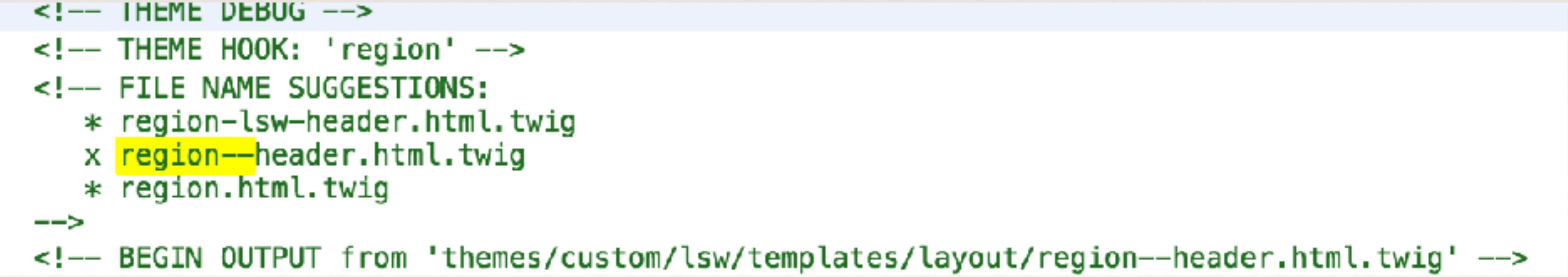
Adding suggestions for region.html.twig



```
  <!-- THEME DEBUG -->
  <!-- THEME HOOK: 'region' -->
  <!-- FILE NAME SUGGESTIONS:
     x region--header.html.twig
     * region.html.twig
  -->
  <!-- BEGIN OUTPUT from 'themes/custom/lsw/templates/layout/region--header.html.twig' -->
▶<div class="row">…</div>
  <!-- END OUTPUT from 'themes/custom/lsw/templates/layout/region--header.html.twig' -->
</header>
```



```
/**
 * Implements hook_theme_suggestions_HOOK_alter().
 */
function lsw_theme_suggestions_region_alter(array &$suggestions, array $variables) {
  $suggestions[] = "region_lsw_" . $variables['elements']['#region'];
}
```



```
<!-- THEME DEBUG -->
<!-- THEME HOOK: 'region' -->
<!-- FILE NAME SUGGESTIONS:
   * region-lsw-header.html.twig
   x region--header.html.twig
   * region.html.twig
-->
<!-- BEGIN OUTPUT from 'themes/custom/lsw/templates/layout/region--header.html.twig' -->
```

What devel/debug modules do we use?

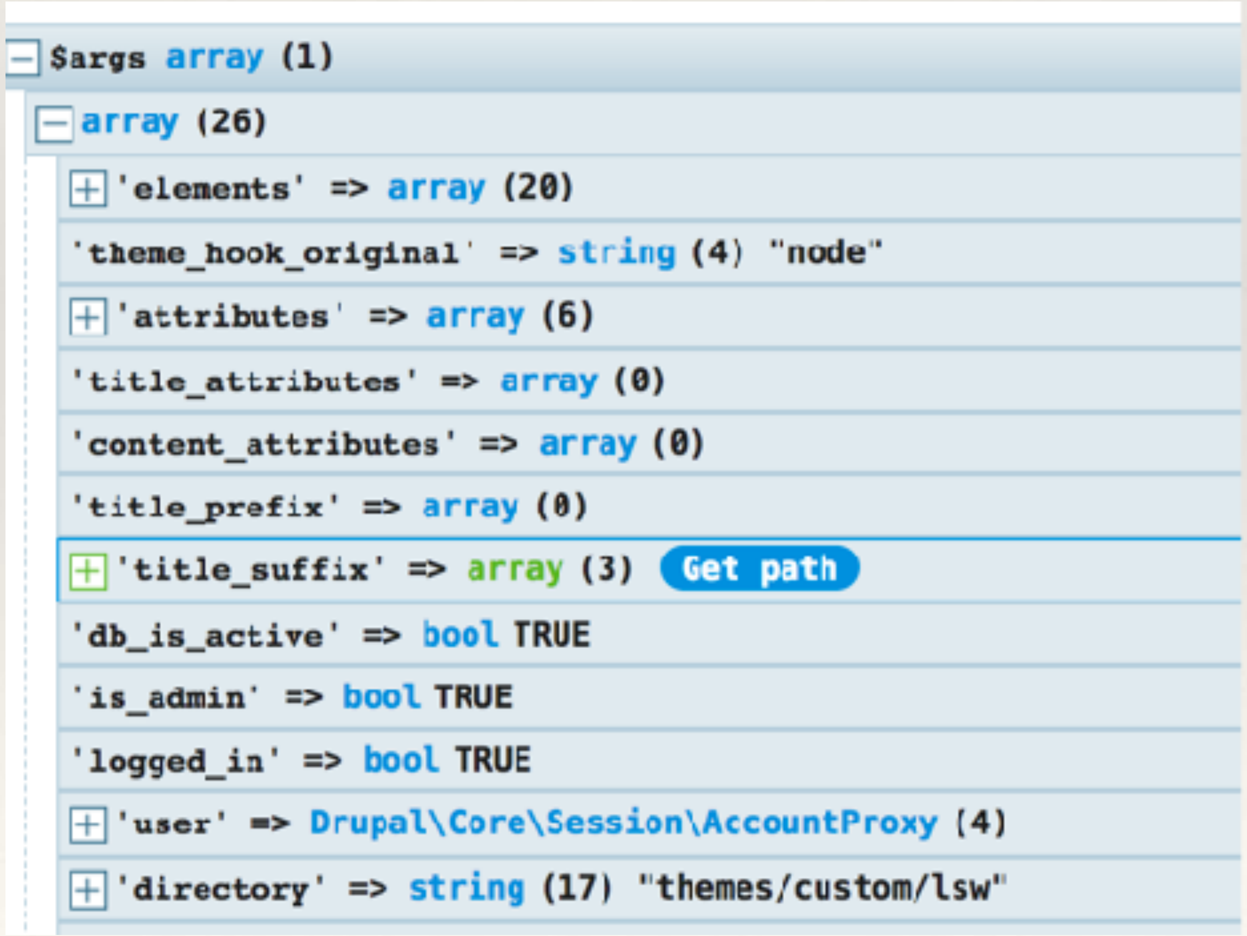
- devel - gives us dsm(), dd()
- twig_xdebug - breakpoints in twig files.
- kint - gives us kint()
  - search_kint - search all the kint….
- webprofiler
- XDebug inside PHPStorm is awesome.

devel and kint

Using the devel module allows you to use `dd()` to send to the TEMP/ drupal_debug.txt loging file. (TEMP is usually /tmp).

Also can use dsm() to spit the values to the page, but thats **everything** - indented, but not folded. So tends to be very large - useful only for individual vars and small objects.

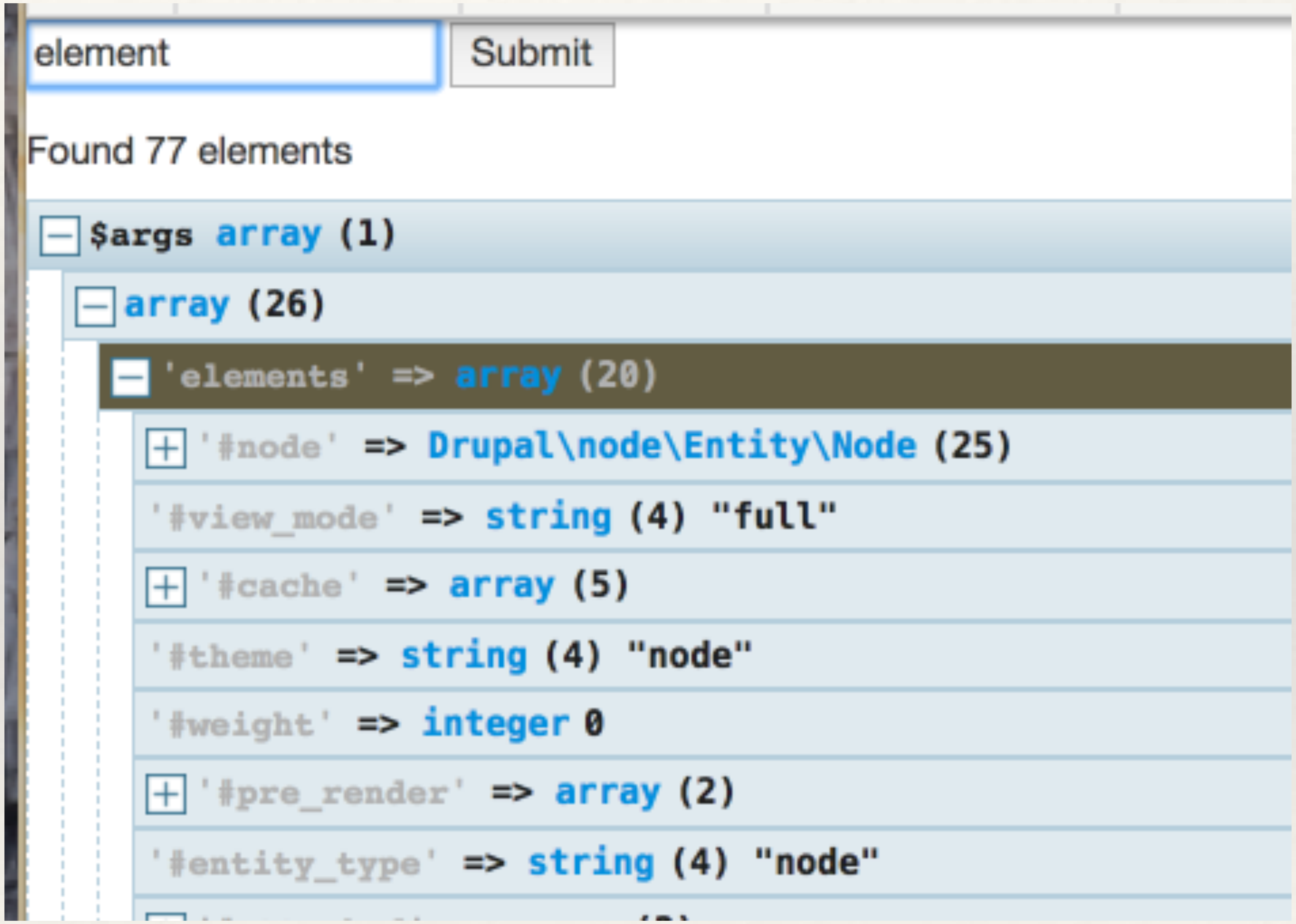
kint($var)
dumps out the information to the page.
Be careful clicking the first [+] it'll open **EVERYTHING**.
Kint is very slow, but very useful.



```
[−]$args array (1)
  [−]array (26)
    [+]'elements' => array (20)
       'theme_hook_original' => string (4) "node"
    [+]'attributes' => array (6)
       'title_attributes' => array (0)
       'content_attributes' => array (0)
       'title_prefix' => array (0)
    [+]'title_suffix' => array (3) Get path
       'db_is_active' => bool TRUE
       'is_admin' => bool TRUE
       'logged_in' => bool TRUE
    [+]'user' => Drupal\Core\Session\AccountProxy (4)
    [+]'directory' => string (17) "themes/custom/lsw"
```

Search Kint

Having the Search Kint module enabled also shows a search box to search though the dump for anything that matches.

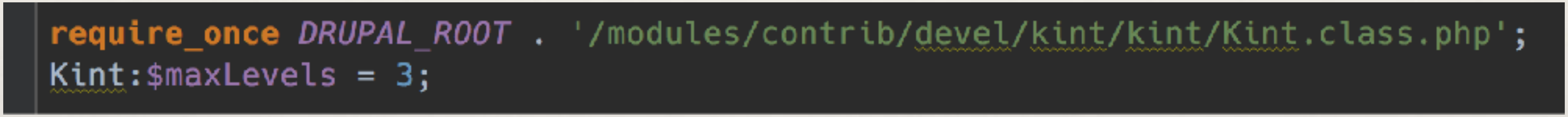


element

Found 77 elements

```
$args array (1)
  array (26)
    'elements' => array (20)
      '#node' => Drupal\node\Entity\Node (25)
      '#view_mode' => string (4) "full"
      '#cache' => array (5)
      '#theme' => string (4) "node"
      '#weight' => integer 0
      '#pre_render' => array (2)
      '#entity_type' => string (4) "node"
```

Reduce Kint levels



```php
require_once DRUPAL_ROOT . '/modules/contrib/devel/kint/kint/Kint.class.php';
Kint:$maxLevels = 3;
```

Just Use PHPStorm

$89/year individual.
Free if contributing to open source

Debugging Twig Templates

- twig_xdebug contrib module
  - https://www.drupal.org/project/twig_xdebug
  - set a breakpoint in your template with {{ breakpoint() }}
  - Then can inspect your theme variables with xdebug.
    - $context[‘values’]

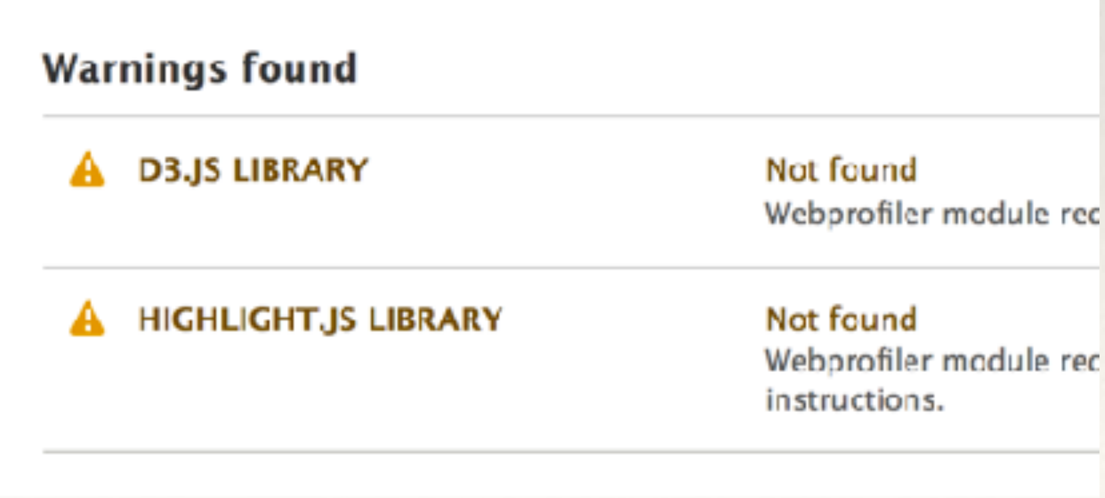
How to install twig_xdebug

```
$ composer require --dev drupal/twig_xdebug
```

Will install module and required libraries.

Web Profiler - for Theming?

- Not really needed for theme dev, but useful to know.
- Devel module
  - Webprofile sub module
  - Helps to install 2 libraries - check status report
    - For SQL formatting - highlight.js
    - For timeline graphics - d3.js

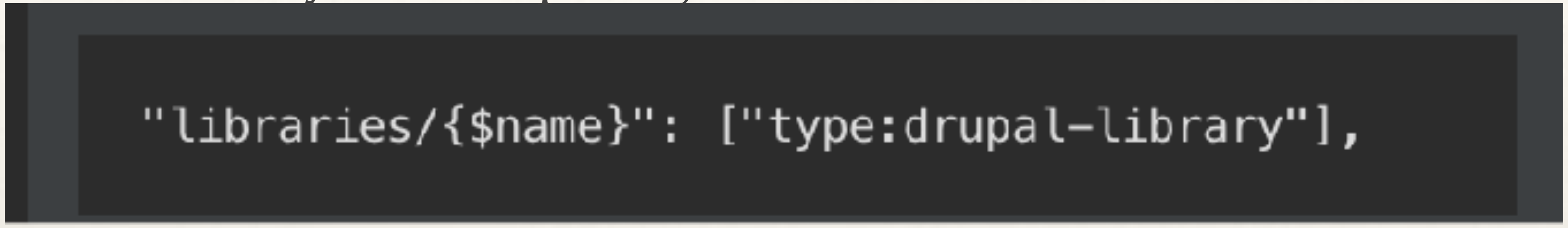


Warnings found

⚠	D3.JS LIBRARY	Not found Webprofiler module rec
⚠	HIGHLIGHT.JS LIBRARY	Not found Webprofiler module rec instructions.

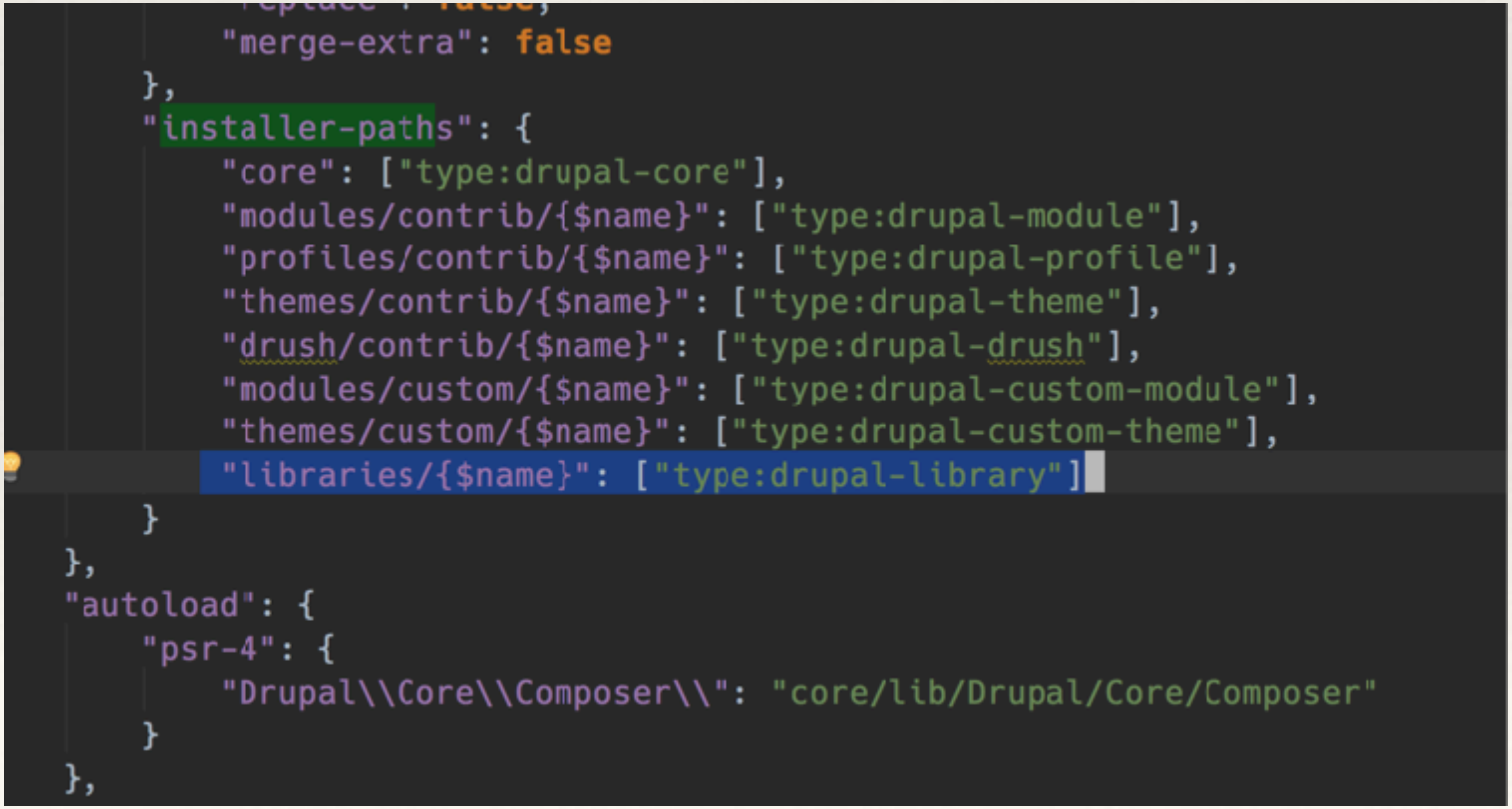
Web Profiler - install extra libraries

Add this to your composer.json



```
"libraries/{$name}": ["type:drupal-library"],
```

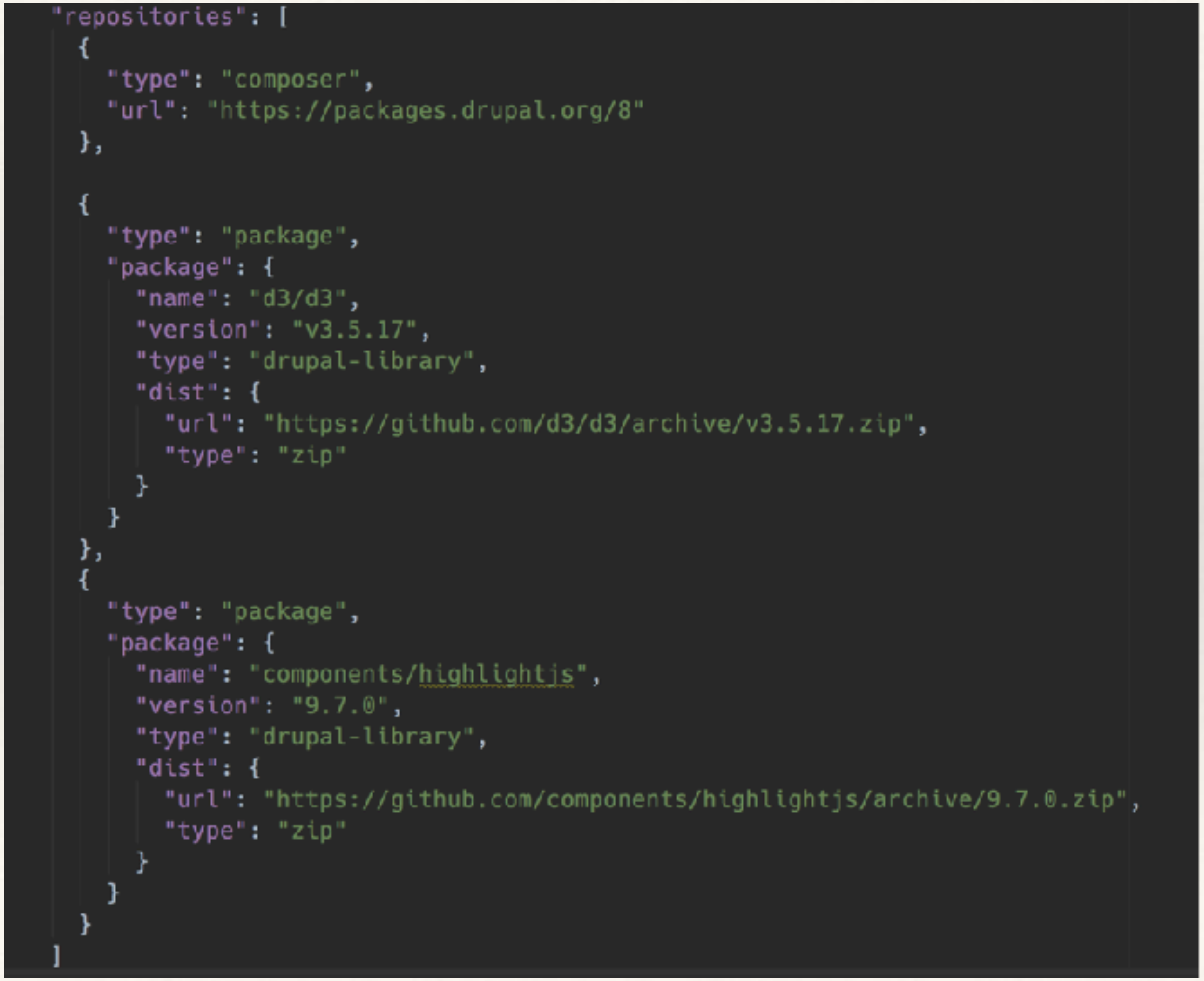
So it will look like this



```
            "replace": false,
            "merge-extra": false
        },
        "installer-paths": {
            "core": ["type:drupal-core"],
            "modules/contrib/{$name}": ["type:drupal-module"],
            "profiles/contrib/{$name}": ["type:drupal-profile"],
            "themes/contrib/{$name}": ["type:drupal-theme"],
            "drush/contrib/{$name}": ["type:drupal-drush"],
            "modules/custom/{$name}": ["type:drupal-custom-module"],
            "themes/custom/{$name}": ["type:drupal-custom-theme"],
            "libraries/{$name}": ["type:drupal-library"]
        }
    },
    "autoload": {
        "psr-4": {
            "Drupal\\Core\\Composer\\": "core/lib/Drupal/Core/Composer"
        }
    },
```

Web profiler libraries

- edit composer.json

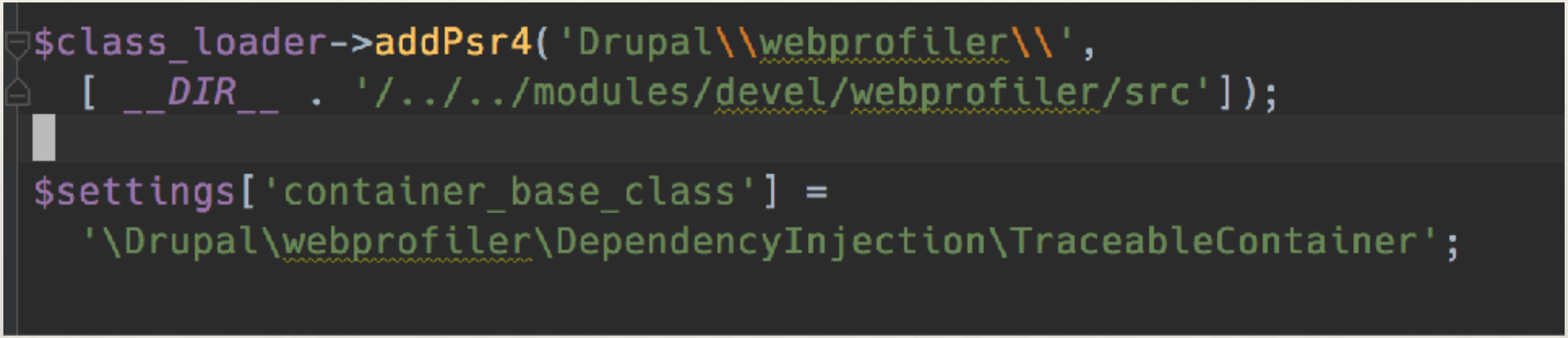


```
"repositories": [
  {
    "type": "composer",
    "url": "https://packages.drupal.org/8"
  },

  {
    "type": "package",
    "package": {
      "name": "d3/d3",
      "version": "v3.5.17",
      "type": "drupal-library",
      "dist": {
        "url": "https://github.com/d3/d3/archive/v3.5.17.zip",
        "type": "zip"
      }
    }
  },
  {
    "type": "package",
    "package": {
      "name": "components/highlightjs",
      "version": "9.7.0",
      "type": "drupal-library",
      "dist": {
        "url": "https://github.com/components/highlightjs/archive/9.7.0.zip",
        "type": "zip"
      }
    }
  }
]
```

Installing Web profiler libraries

- composer require --prefer-dist composer/installers
- composer require --prefer-dist d3/d3:3.5.* components/highlightjs:9.7.*

Also need to add to your settings.local.php



```php
$class_loader->addPsr4('Drupal\\webprofiler\\',
  [ __DIR__ . '/../../modules/devel/webprofiler/src']);

$settings['container_base_class'] =
  '\Drupal\webprofiler\DependencyInjection\TraceableContainer';
```

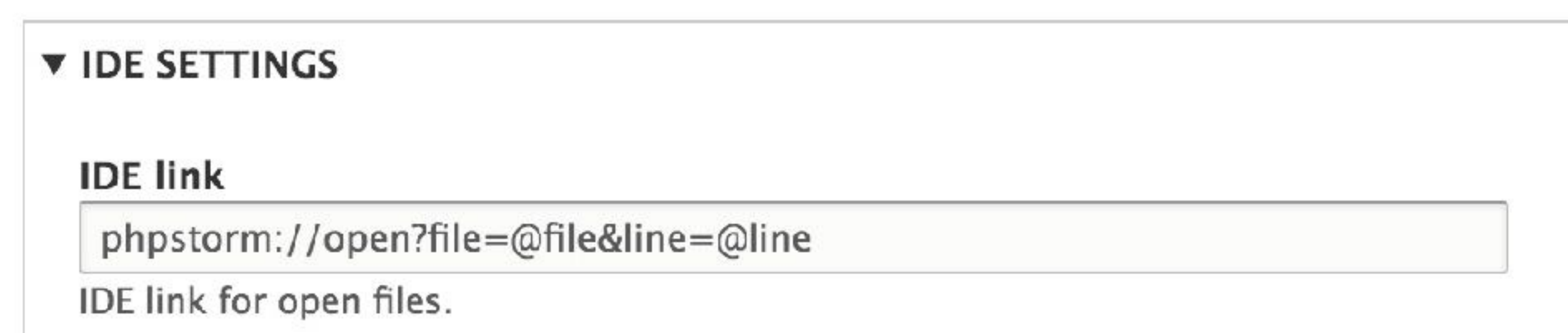
- After this timeline works

Load into PHPStorm from Webprofiler

Let's configure Web Profiler to open PhpStorm.

Go to the "Webprofiler settings" page (admin/config/development/devel/webprofiler) or click on the "Configure Webprofiler" link by hovering over the Drupal icon on the left.

Click on the "IDE SETTINGS" field-set and add
phpstorm://open?file=@file&line=@line
into "IDE link".



▼ IDE SETTINGS

IDE link

phpstorm://open?file=@file&line=@line

IDE link for open files.

Thank You - Useful links

Web Profiler setup
https://www.webwash.net/debug-site-performance-using-web-profiler-in-drupal-8

Twig XDebug
https://www.drupal.org/project/twig_xdebug

Disable caching and Twig debug for development
https://www.drupal.org/node/2598914

Drupal 8 Theming with Twig - Packt Pub - Chaz Chumley
https://www.packtpub.com/web-development/drupal-8-theming-twig

Lee Walker - lee@codej.us - @DrupalNooga - www.codejourneymen.com
drupal.org : mr_scumbag