BACK TO BASICS DRUPAL 8 THEMING

PRESENTED BY BRIAN PERRY

June 28, 2018

Slides: http://bit.ly/basics-d4d

BRIAN PERRY

Hi!

- Interactive Developer at HS2 Solutions
- Rocking the Chicago suburbs
- Lover of all things components
- ... and Nintendo

d.o: brianperry
Twitter: bricomedy
Github: backlineint
Nintendo: wabrian
brianperryinteractive.com

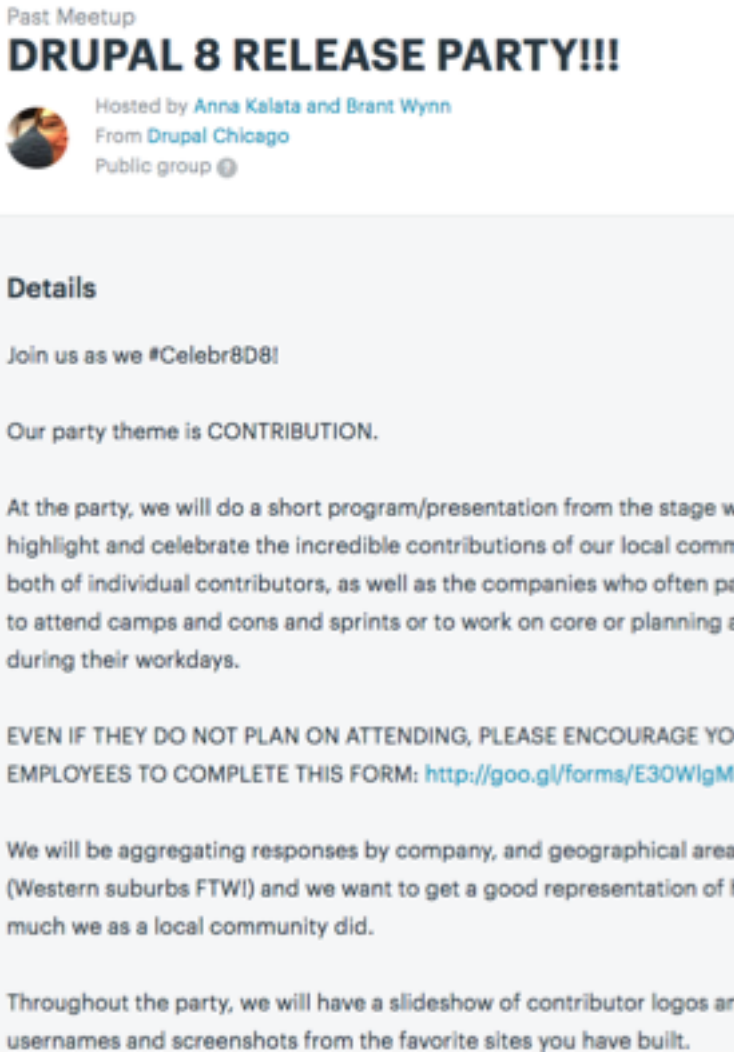
HS2 SOLUTIONS

NOVEMBER 19, 2015

NOV 19, 2015 - DRUPAL 8 IS RELEASED

“Welp, guess I need to learn Twig…”

Those were simpler times.



MODERN FRONT END IS AMAZING

- Automation
  - NPM, Yarn, Gulp, Grunt, Webpack, etc.
  - SASS, PostCSS
- CSS Grid
- Design systems and pattern libraries
- Component libraries - Bootstrap, Foundation
- The Continued Rise of JS
  - ES6
  - Frameworks – React, Vue, Angular
  - Ever growing module ecosystem

MODERN FRONT END IS A DUMPSTER FIRE

- Automation
  - NPM, Yarn, Gulp, Grunt, Webpack, etc.
  - SASS, PostCSS
- CSS Grid
- Design systems and pattern libraries
- Component libraries - Bootstrap, Foundation
- The Continued Rise of JS
  - ES6
  - Frameworks – React, Vue, Angular
  - Ever growing module ecosystem

SINCE NOVEMBER 19, 2015 I HAVE…

- Become comfortable with Twig
- Incorporated Pattern Lab Into workflow
  - Mapped data to components via Twig
  - Mapped data to components via UI Patterns Module
- Managed Drupal sites with Composer
- Decoupled
  - Progressively decoupled Drupal/Angular site
  - Fully Decoupled Drupal/React sites (SPA/Static)
- Learned (some) ES6
- Dabbled with CSS Grid

“

While creating new processes, it might help to find a balance between things that break new ground, but maybe also might simplify the workflow.

'ANONYMOUS'

PERSON FROM MY PEER REVIEW WHO WAS PROBABLY 100% RIGHT

BACK TO BASICS

*Ba*sic /basick/ adjective*

1. forming an essential foundation or starting point; fundamental.

BASIC != EASY

OPO (ONE PERSON'S OPINION) WARNING

All. Of. The. Things.

DEPENDENCY MANAGEMENT

AUTOMATION

CSS PREPROCESSING

CSS LAYOUT

JAVASCRIPT

TWIG

COMPONENTS

SITE BUILDING

LOCAL DEV ENVIRONMENTS

DEPENDENCY MANAGEMENT

DO

- ✓ Manage Drupal Dependencies with Composer
- ✓ Manage JS Dependencies with NPM

DON'T

- Use Bower (mostly a reminder for me)
- Worry about the hot new thing (but feel free to use Yarn if team/project prefers it)

NEXT STEPS

- Get REALLY comfortable with managing Composer and NPM dependencies

Dependency hell is a real thing, but ignoring it won't make it go away.

DEPENDENCY MANAGEMENT

- Where does all of this stuff go?
  - Composer dependencies – root of Drupal project (see drupal-composer/drupal-project for example)
  - Node dependencies – root of your theme

DEPENDENCY MANAGEMENT

AUTOMATION

DO

- ✓ Use NPM to run scripts
- ✓ Use Gulp as a task runner (my personal preference)

DON'T

- ○ Get caught up in a battle over Gulp / Grunt / Webpack. If it works better for your team, use it.

It is worth the initial effort to let the robots do the work for you.

AUTOMATION

CSS PROCESSING

DO

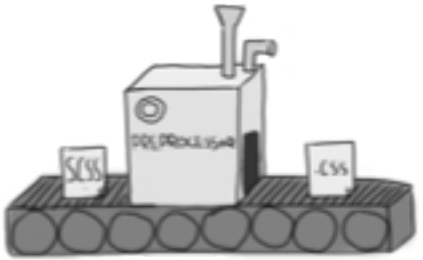
- ✓ Preprocess with SASS & compile to a single CSS file
- ✓ Use SASS variables
- ✓ Post process with Autoprefixer

DON'T

- Nest like a madman
- Create individual css files per partial (yet)

NEXT STEPS

- Follow BEM class naming conventions



When used responsibly a little bit of CSS processing can go a long way.

CSS PREPROCESSING

PACKAGE.JSON

```
{
  "name": "back_to_basics",
  "version": "1.0.0",
  "scripts": {
    "start": "gulp",
    "sass": "gulp sass",
    "sass:watch": "gulp sass:watch"
  },
  "author": "Brian Perry",
  "private": true,
  "devDependencies": {
    "gulp": "^3.9.1",
    "gulp-autoprefixer": "^5.0.0",
    "gulp-sass": "^4.0.1",
    "gulp-sass-glob": "^1.0.9"
  }
}
```

GULPFILE.JS

```
'use strict';

var gulp = require('gulp');
var sass = require('gulp-sass');
var sassGlob = require('gulp-sass-glob');
var autoprefixer = require('gulp-autoprefixer');

gulp.task('default', ['sass:watch']);

gulp.task('sass', function () {
  return gulp
    .src('./sass/back_to_basics.scss')
    .pipe(sassGlob())
    .pipe(sass())
    .pipe(autoprefixer({
      browsers: ['last 2 versions'],
      cascade: false
    }))
    .pipe(gulp.dest('./css'));
});

gulp.task('sass:watch', function () {
  gulp.watch('./sass/**/*.scss', ['sass']);
});
```

CSS PREPROCESSING – NEXT STEPS - BEM

```scss
// _provider-card-head.scss - styles for head section of provider card

.o-provider-card-head { // Block: .o-provider-card-head
  &__photo { // Element: .o-provider-card-head__photo
    border-radius: 50%;
    width: rem-calc(100);
    height: rem-calc(100);
    object-fit: cover;
    @include breakpoint(large) {
      width: rem-calc(70);
      height: rem-calc(70);
    }
    &--placeholder { // Modifier: .o-provider-card-head__photo--placeholder
      position: relative;
      display: inline-block;
      width: rem-calc(100);
      height: rem-calc(100);
      padding: rem-calc(50) 0;
      @include breakpoint(large) {
        width: rem-calc(70);
        height: rem-calc(70);
        padding: rem-calc(35) 0;
      }
      border-radius: 50%;
      background-color: $c-cent-accent5;
      font-size: rem-calc(18px);
      line-height: .25;
      text-align: center;
      color: $c-cent-white;
    }
```

.o-provider-card-head

CSS PREPROCESSING

CSS LAYOUT

DO

- ✓ Use CSS Grid if project browser requirements will allow it.

DON'T

- ○ Default to using frameworks like bootstrap/foundation if you can use CSS Grid

NEXT STEPS

- Build layouts you've never thought possible

Build with CSS Grid today for a better tomorrow.

CSS LAYOUT

JAVASCRIPT

DO

- ✓ Properly include your JS as libraries
- ✓ Use Drupal behaviors

DON'T

- Just copy and paste some JS code from Stack Overflow and add it to a global library.
- Default to using jQuery for everything

NEXT STEPS

- Use babel to compile your JS (wait, what?)
- Start becoming familiar with ES6 (and beyond) syntax

The time to modernize your approach to JavaScript is now.

JAVASCRIPT – DRUPAL BEHAVIORS

```
(function ($, Drupal) {
  Drupal.behaviors.ourTeamApplyHeight = {
    attach: function (context, settings) {
      // Equalize our team cards
      $('.o-our-team-cards').once().each(function() {
        // Don't try to equalize something with a height of 0 - foundation will
        // throw an error.
        if ($(this).height() > 0) {
          new Foundation.Equalizer($(this)).applyHeight();
        }
      });

      // Equalize our team cards when an accordion is expanded.
      $('.o-accordion').once().click(function() {
        new Foundation.Equalizer($(this).find('.o-our-team-cards')).applyHeight();
      });

    }
  };
})(jQuery, Drupal);
```

JAVASCRIPT – NEXT STEPS – COMPILE JS

- Temporary Solution:
  - Repurpose package.json from /core in your theme
    - You will need to merge your own scripts / dependencies
  - Copy /scripts/js from /core into your theme
  - Run *npm install to add dependencies*
  - You can now run scripts to compile *.es6.js files:
    - npm run build:js
    - npm run watch:js
  - Bonus - you get Core's linting rules as well.
- Issue to support a less copy and paste solution: https://www.drupal.org/project/drupal/issues/2957390

JAVASCRIPT

TWIG

DO

- ✓ Get to know all of the great things Twig can do (tags, filters, functions, etc)
- ✓ Follow Drupal Twig conventions in Drupal templates
- ✓ Get great at debugging

DON'T

- ○ Get too carried away with logic and Twig magic in templates

Twig can do a bunch of cool stuff, but at the end of the day it's just the markup that we know and love.

TWIG

- Twig Recipes – Making Drupal 8 Render the Markup You Want
  - In the next session slot - 4:30 – 144

TWIG

COMPONENTS

DO

- ✓ Use template suggestions responsibly
- ✓ Twig Include/Extend/Embed for reuse

DON'T

- ○ Overcomplicate your use of Twig Include / Extend
- ○ Fail the 'time to markup test'

NEXT STEPS

- • Use a Pattern Library / Style Guide
- • Integrate with an external Pattern Library

Start by thinking in components, and grow into using a full pattern library driven workflow.

COMPONENTS – INCLUDE/EXTEND/EMBED

paragraph--marketing-content-section.html.twig

```
{{ attach_library('foundation_patterns/marketing-site-content-section') }}

{% include "@marketing/content-section/marketing-site-content-section.twig"
  with {
    "image": content.field_fpc_image,
    "header": content.field_fpc_header,
    "subheader": content.field_fpc_subheader,
    "button": content.field_fpc_link
  }
%}
```

COMPONENTS

SITE BUILDING

DO

- ✓ Use Responsive images
- ✓ Use Drupal Layouts (Layout API / Layout Discovery)

DON'T

- ○ Make 'manage display' in the admin UI a liar

NEXT STEPS

- Get ready for Layout Builder

Embrace layouts now to prepare for Drupal's bright layout future.

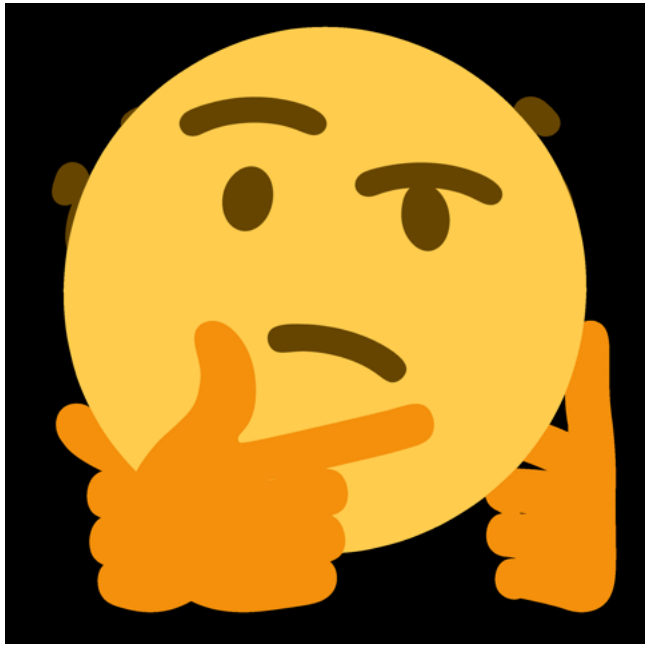
SITE BUILDING - LAYOUTS

- Registering Layouts

  mytheme.layouts.yml

```yaml
one_column:
  label: 'One column'
  category: 'My Layouts'
  template: templates/one-column
  default_region: main
  regions:
    main:
      label: Main content
two_column:
  label: 'Two column'
  category: 'My Layouts'
  template: templates/two-column
  default_region: main
  regions:
    main:
      label: Main content
    sidebar:
      label: Sidebar
```

SITE BUILDING



BONUS: LOCAL DEV ENVIRONMENT

DO

- ✓ Use a local dev environment consistent with the rest of your team

DON'T

- ○ Sink crazy amounts of time into maintaining your VM

NEXT STEPS

- Experiment with Docker VMs (my preference is Lando)
- Run all front end tooling in containers

VM configuration can help eliminate versioning issues and make front end tools more easily available.

BONUS: LOCAL DEV ENVIRONMENT

WAIT A MINUTE…

Is Drupal not the hard part of Drupal theming?

IN REVIEW

DO

- ✓ Create beautiful, responsive, fast, accessible Drupal sites

DON'T

- ○ Let the Bartik stereotype live on (be more Umami)

NEXT STEPS

- Share your work and contribute back

D8 Front End FTW!

THANKS!

Brian Perry
Interactive Developer
HS2 Solutions, Inc.

brian.perry@hs2solutions.com

@bricomedy
d.o: brianperry

HOW COULD THERE POSSIBLY BE MORE QUESTIONS?