2018

Build process automation

Kevin O'Brien

kevin@clockwork.com

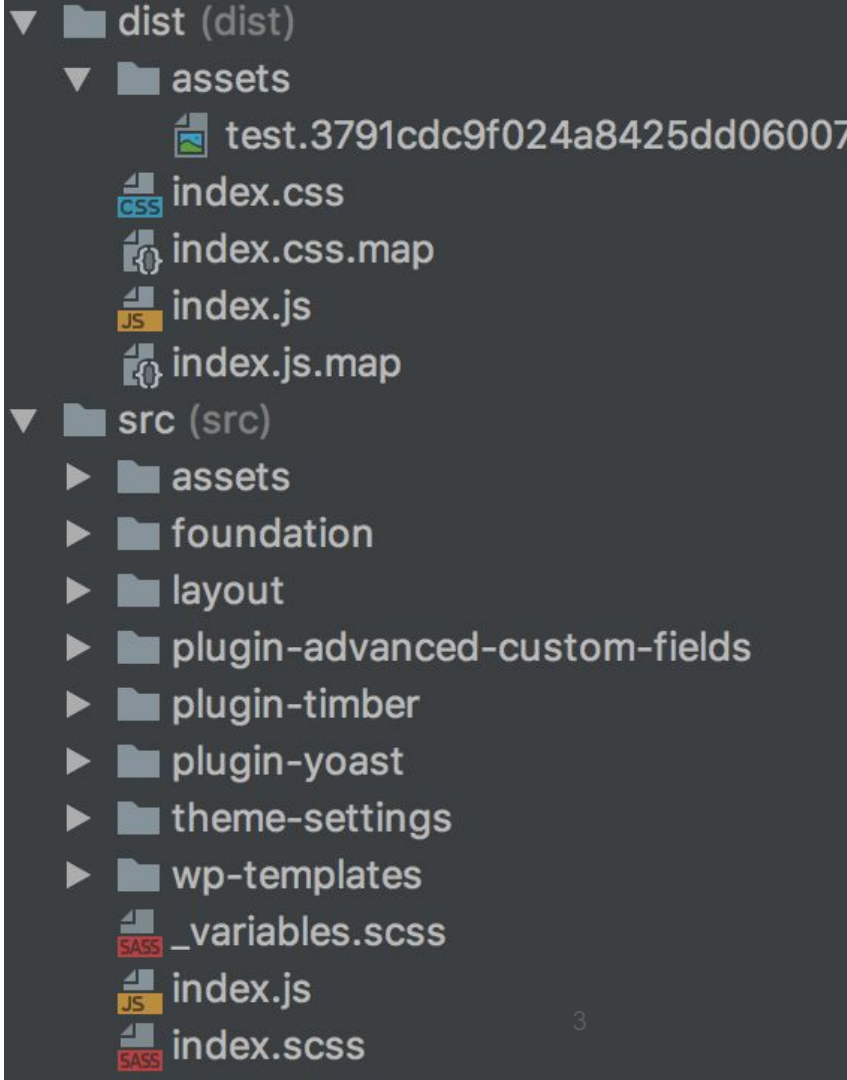
Agenda

- What & why
- Recommended tools
- WordPress set up
- The build script
- Advanced

TL;DR

What is it?

Transform source files (src) to optimized files (dist) automatically.



github.com/keobrien/WC2018

```
jQuery('img[src$=".svg"][typeof="foaf:Image"]').each(function
    const $img = jQuery(this);
    const imgID = $img.attr('id');
    const imgClass = $img.attr('class');
    const imgURL = $img.attr('src');

    jQuery.get(imgURL, function(data) {
        // Get the SVG tag, ignore the rest
        let $svg = jQuery(data).find('svg');

        // Add replaced image's ID to the new SVG
        if(typeof imgID !== 'undefined') {
            $svg = $svg.attr('id', imgID);
        }
        // Add replaced image's classes to the new SVG
        if(typeof imgClass !== 'undefined') {
            $svg = $svg.attr('class', imgClass+' replaced-svg
        }
```

```
ass"),n=e.attr("src");c.get(n,function(n){var l=c
ed-svg")),l=l.removeAttr("xmlns:a"),!l.attr("view
.attr("width")),e.replaceWith(l)},"xml")})}).call
({modifiers:"right, overlay",triggerButton:"#mobi
find(".drilldown").drilldown({event:"click",selec
lldown-container",root:"drilldown-root",sub:"dril
(function(e,t){c(t).find("li.active_trail > a").c
nction(){c('[data-ref="carousel-testimonial"]:not
e:!0,slidesToShow:1,slidesToScroll:1,fade:!0,dots
/button>',responsive:[{breakpoint:992,settings:{a
ict";(function(c){c(document).ready(function(){c(
iveHeight:!0,slidesToShow:2,slidesToScroll:2,infi
/button>',responsive:[{breakpoint:992,settings:{s
nction(c){c(document).ready(function(){c('[data-r
```

TL;DR

Why?!

HUMAN readable **source** code.

MACHINE optimized **production** code.

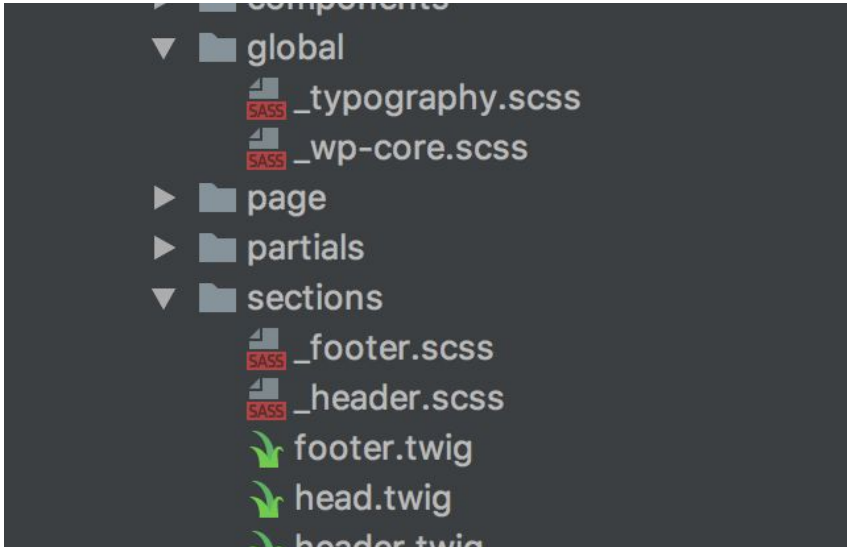
TL;DR

Why?!

- Organization
- Testability
- Developer onboarding
- Catch bugs fast
- Build reports

```
set    Size  Chunks                    Chunk Names
.js  343 kB       0  [emitted]  [big]  main
map  534 kB       0  [emitted]         main
src/index.scss 3.94 kB {0} [built] [failed] [1 error]
 -> factory:0ms building:1275ms = 1275ms
hidden modules

./src/index.scss
ild failed: ModuleNotFoundError: Module not found: Error: Can't resolve '../assets/
ctoryCallback (/Users/kevinobrien/Clockwork/code/wordpress-theme/node_modules/webpa
ctory (/Users/kevinobrien/Clockwork/code/wordpress-theme/node_modules/webpack/lib/N
solver (/Users/kevinobrien/Clockwork/code/wordpress-theme/node_modules/webpack/lib/
yncLib.parallel (/Users/kevinobrien/Clockwork/code/wordpress-theme/node_modules/web
sers/kevinobrien/Clockwork/code/wordpress-theme/node_modules/async/dist/async.js:38
sers/kevinobrien/Clockwork/code/wordpress-theme/node_modules/async/dist/async.js:47
eratorCallback (/Users/kevinobrien/Clockwork/code/wordpress-theme/node_modules/asyn
sers/kevinobrien/Clockwork/code/wordpress-theme/node_modules/async/dist/async.js:95
sers/kevinobrien/Clockwork/code/wordpress-theme/node_modules/async/dist/async.js:38
solvers.normal.resolve (/Users/kevinobrien/Clockwork/code/wordpress-theme/node_modu
Error (/Users/kevinobrien/Clockwork/code/wordpress-theme/node_modules/enhanced-reso
ggingCallbackWrapper (/Users/kevinobrien/Clockwork/code/wordpress-theme/node_module
nAfter (/Users/kevinobrien/Clockwork/code/wordpress-theme/node_modules/enhanced-res
nerCallback (/Users/kevinobrien/Clockwork/code/wordpress-theme/node_modules/enhance
ggingCallbackWrapper (/Users/kevinobrien/Clockwork/code/wordpress-theme/node_module
```



github.com/keobrien/WC2018

```
import authentication from '../authentication';
import screens from '@/screens';
import store from '@/store';
import user from '@/components/user';
import i18n from '@/configured/vue-i18n';
import log from '@/configured/logger';
import offline from '@/offline';
import api, {extendSession, OfflineError} from '@/api';
import constants from '@/constants';

Vue.use(Router);

const router = new Router({
  //mode: 'history',
  routes: [
    ...screens.routes
  ]
});

router.beforeEach(beforeEach);
router.afterEach(afterEach);

export default router;

export {
  isUserLoggedIn,
  isUserLoaded,
  isUserApproved,
}

//////////
```

```
$primary:       #284184 !default;
$secondary:     #714387 !default;

$success:       #308264 !default;
$info:          #26805D !default;
$info-contrast: $white !default;
$warning:       $yellow !default;
$danger:        #af2f38 !default;

$grayscale-100: #f5f9fb !default;
$grayscale-200: #eff5f8 !default;
$grayscale-300: #e3e9ed !default;
$grayscale-400: #5c798a !default;
$grayscale-500: #4d697b !default;
$grayscale-600: #122b3a !default;
```

TL;DR

Why?!

Faster development

Javascript/CoffeeScript & CSS/Sass

- Essential for React, Vue.js, etc. (Gutenberg blocks)
- ES6 & npm modules

Getting started

webpack

npm

Gulp

GRUNT

yarn

Bower

Why WebPack & NPM?

WebPack excels at assets (JavaScript, CSS, Images, etc.)
It was built to understand JavaScript & NPM really well.

Gulp & Grunt are good for bulk files, useful for moving files around or uploading.

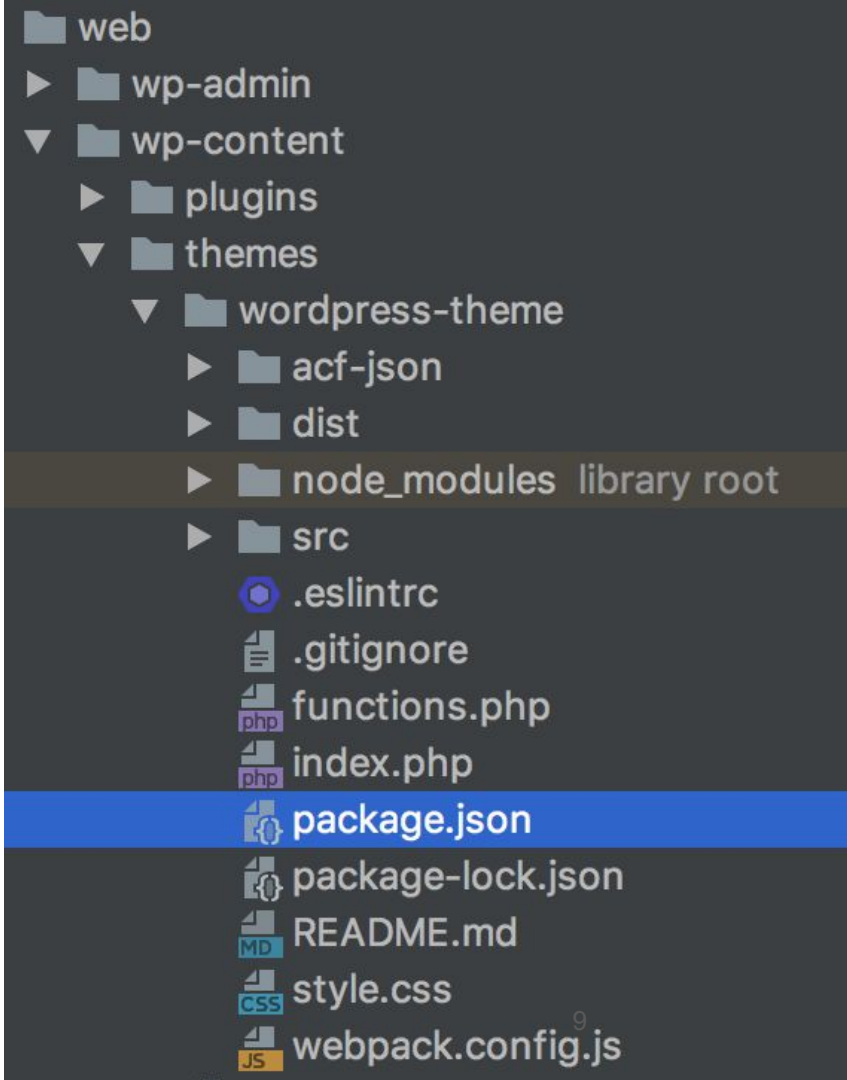
NPM works great with WebPack. Very similar to **Yarn**, I just haven't tried it yet. **Bower** is an older dependency manager and has been all but abandoned.

All of these run using Node.js, make sure you've installed it first (nodejs.org)

Getting started: Files & folders

Package.json is the starting point.

- Source folder (src)
- Destination folder (dist)
- Build config (webpack.config.js)



github.com/keobrien/WC2018

Getting started: package.json

Always use scripts > start as your main script for a consistent starting point.

Separate live site code from build tools with “dependencies” and “devDependencies.”

Tip:
Document the project’s intended node/npm version under “engines.”

Tip:
Store variables in package.json if you don’t need a separate config file.
process.env.npm_package_config_paths_source

github.com/keobrien/WC2018

package.json

```
{
  "name": "test-theme",
  "version": "1.0.0",
  "description": "",
  "author": "",
  "scripts": {
    "start": "./node_modules/.bin/webpack --watch",
    "build-production": "export NODE_ENV=production && ./node_modules/.bin
    "precommit": "npm run vet",
    "lint": "./node_modules/.bin/eslint src/",
    "vet": "npm list --depth=0 && npm run lint"
  },
  "dependencies": {
    "foundation-sites": "^6.4.1",
    "jquery": "^3.2.1"
  },
  "devDependencies": {
    "autoprefixer": "^8.4.1",
    "babel-core": "^6.26.0",
    "babel-loader": "7.1.1",
    "babel-preset-env": "^1.6.1",
    "clean-webpack-plugin": "^0.1.17",
    "css-loader": "^0.28.7",
    "eslint": "^3.12.2",
    "eslint-config-clockwork": "0.0.8",
    "extract-text-webpack-plugin": "^3.0.0",
    "file-loader": "^1.1.5",
    "husky": "^0.14.3",
    "node-sass": "^4.5.3",
    "postcss-loader": "^2.0.8",
    "sass-loader": "^6.0.6",
    "style-loader": "^0.18.2",
    "webpack": "^3.5.6",
    "webpack-notifier": "^1.5.0",
    "yargs": "^8.0.2"
  },
  "config": {
    "paths": {
      "output": "dist",
      "source": "src"
    }
  },
  "engines": {
    "node": "^8.9.3",
    "npm": "^5.5.1"
  }
}
```

Getting started: The essentials

KEEP IT SIMPLE

- Simple is **fast**
  (200 ms - 1 sec is great for watch builds)
- Simple is less maintenance
- Less can go wrong

The classics

- Combine, minify & sourcemap
- SASS CSS
- Auto-prefix CSS
- Cache bust images & fonts

```
"devDependencies": {
  "autoprefixer": "^8.4.1",
  "babel-core": "^6.26.0",
  "babel-loader": "7.1.1",
  "babel-preset-env": "^1.6.1",
  "clean-webpack-plugin": "^0.1.
  "css-loader": "^0.28.7",
  "eslint": "^3.12.2",
  "eslint-config-clockwork": "0.
  "extract-text-webpack-plugin":
  "file-loader": "^1.1.5",
  "husky": "^0.14.3",
  "node-sass": "^4.5.3",
  "postcss-loader": "^2.0.8",
  "sass-loader": "^6.0.6",
  "style-loader": "^0.18.2",
  "webpack": "^3.5.6",
  "webpack-notifier": "^1.5.0",
  "yargs": "^8.0.2"
```

github.com/keobrien/WC2018

Getting Started: Bonus points

Nice additions

- Babel for ES6
- ESLint for code checks
- Husky for Git hooks
- WebPack Notifier for build feedback
- Yargs for adding options to your build scripts

Tip:
Think twice before installing/running ANYTHING with sudo. Instead, install packages locally and/or move your global package location. Always look for another answer when Stack Overflow says "Just run sudo ..."

github.com/keobrien/WC2018

```
"devDependencies": {
  "autoprefixer": "^8.4.1",
  "babel-core": "^6.26.0",
  "babel-loader": "7.1.1",
  "babel-preset-env": "^1.6.1",
  "clean-webpack-plugin": "^0.1.
  "css-loader": "^0.28.7",
  "eslint": "^3.12.2",
  "eslint-config-clockwork": "0.
  "extract-text-webpack-plugin":
  "file-loader": "^1.1.5",
  "husky": "^0.14.3",
  "node-sass": "^4.5.3",
  "postcss-loader": "^2.0.8",
  "sass-loader": "^6.0.6",
  "style-loader": "^0.18.2",
  "webpack": "^3.5.6",
  "webpack-notifier": "^1.5.0",
  "yargs": "^8.0.2"
```

style.css

```
/*
Theme Name: Test_theme
Theme URI:
Author:
Author URI:
Description:
Version: 1.0.0
License: GNU General Public License v2 or later
License URI: LICENSE
Text Domain: test_theme
Tags:
```

index.js

```
import './index.scss';
import $ from 'jquery';
import './foundation';

$(document).foundation();
```

enqueue.php

```
<?php

namespace Theme\Settings\Enqueue;

add_action('wp_enqueue_scripts', __NAMESPACE__ . '\\enqueue_sc

function enqueue_scripts() {
    wp_enqueue_style(
        'style-theme-main',
        get_template_directory_uri() . '/dist/index.css' );

    wp_enqueue_script(
        'style-theme-main',
        get_template_directory_uri() . '/dist/index.js',
        array(), '1.0.0', true );
}
```

Getting started: WordPress

- Only need THEME/style.css for theme setup
- Main JavaScript file (WebPack entry) imports all assets.
- Enqueue only the THEME/dist main files.

Getting Started: Imports

SCSS

- Npm modules with ~/
- Custom files use relative path

JavaScript

- Npm modules by name.
- Top level scss file only, avoids confusing import trees.
- Custom files use relative path
- @ sign is a newer convention in the community to namespace based on author.

index.scss

```scss
// Bootstrap
@import "~bootstrap/scss/functions";
@import "~bootstrap/scss/mixins";
@import "variables"; // Variables Override
@import "~bootstrap/scss/bootstrap"; // Full 4.1
@import "~js-offcanvas/src/js-offcanvas.scss";

// Slick
@import "~slick-carousel/slick/slick";

// Font Awesome
@import "fontawesome";

// Fonts
@import "lib/montserrat/stylesheet";
@import "lib/lora/stylesheet";

// Custom
@import "assets/index.scss";
@import "assets/index";
```

index.js

```js
import 'bootstrap';
import 'slick-carousel';
import "@fortawesome/fontawesome-free/js/all";
import "./lib/mini_line/style.css";
import "./cv";
import "js-offcanvas/dist/_js/js-offcanvas.pkgd";
import "jquery-drilldown";

import "./assets/inline-svg";

import "./assets/_region--offcanvas";
import "./assets/_component--testimonial-carousel";
import "./assets/_component--overflow-carousel";
import "./assets/_component--sponsors";
```

github.com/keobrien/WC2018

index.js

```
import $ from 'jquery';

import { Foundation } from 'foundation-sites/js/foundation.core';
Foundation.addToJquery($);

// Add Foundation Utils to Foundation global namespace for backwards
// compatibility.

import { rtl, GetYoDigits, transitionend } from 'foundation-sites/js
Foundation.rtl = rtl;
Foundation.GetYoDigits = GetYoDigits;
Foundation.transitionend = transitionend;

import { Box } from 'foundation-sites/js/foundation.util.box';
import { onImagesLoaded } from 'foundation-sites/js/foundation.util.
import { Keyboard } from 'foundation-sites/js/foundation.util.keyboa
import { MediaQuery } from 'foundation-sites/js/foundation.util.medi
import { Motion, Move } from 'foundation-sites/js/foundation.util.mo
```

index.scss

Enable File Watcher to compile SCSS to CSS?

```
@include foundation-global-styles;
//@include foundation-grid;
//@include foundation-flex-grid;
@include foundation-xy-grid-classes;
@include foundation-typography;
@include foundation-button;
@include foundation-forms;
@include foundation-range-input;
@include foundation-accordion;
@include foundation-accordion-menu;
@include foundation-badge;
@include foundation-breadcrumbs;
@include foundation-button-group;
@include foundation-callout;
@include foundation-card;
@include foundation-close-button;
@include foundation-menu;
@include foundation-menu-icon;
@include foundation-drilldown-menu;
@include foundation-dropdown;
@include foundation-dropdown-menu;
```

Getting started: JavaScript Modules

There are multiple module formats, WebPack takes care of it for you.

- ES6 modules
- CommonJS modules
- AMD modules
- Module Object and IIFE
- UMD

Brief history of JavaScript Modules

Getting Started: Modules

Everything is a module, even your code.

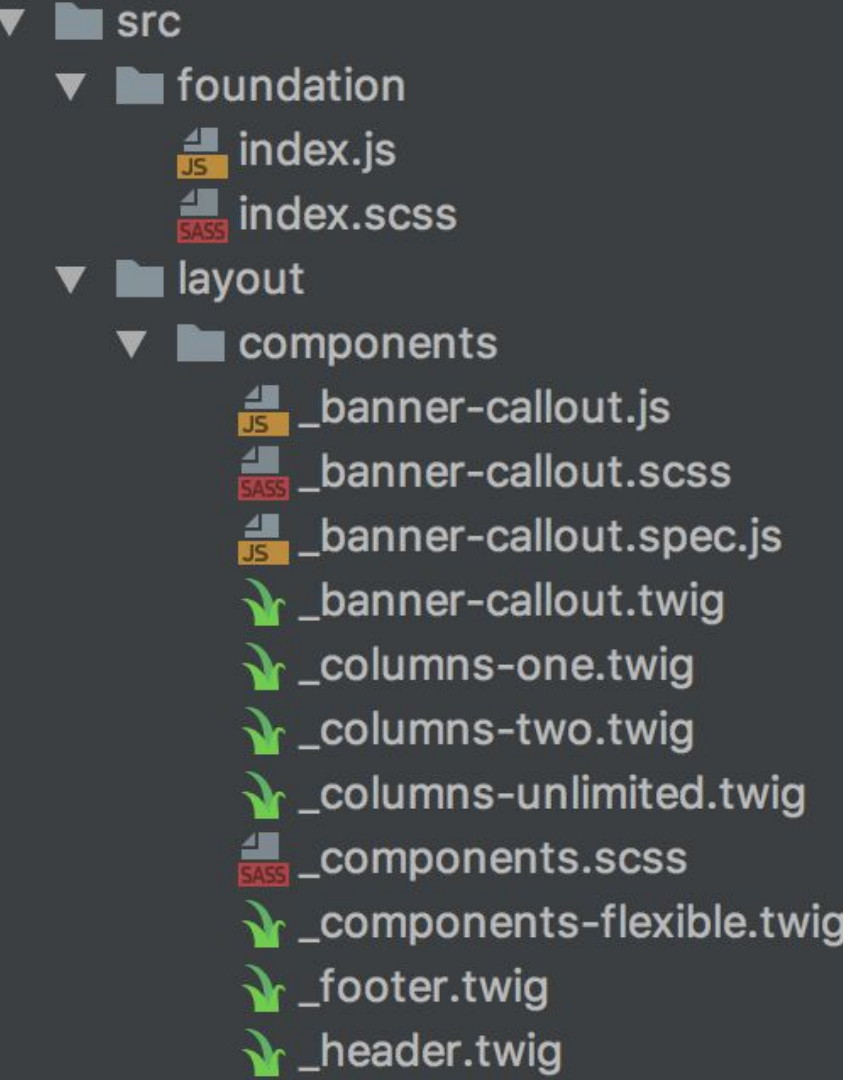
- No globals
- Less to understand per file
- import/export without worrying about format of other modules

```
index.js
import './index.scss';
import $ from 'jquery';
import './foundation';
import 'slick-carousel';

$(document).foundation();

$(document).ready(() => {
    $('[data-ref="events-carousel"]:not(
        accessibility: true,
        adaptiveHeight: true,
        slidesToShow: 2,
        slidesToScroll: 2,
        infinite: false,
        prevArrow: '<button type="button
        responsive: [
            {
                breakpoint: 992,
                settings: {
                    slidesToShow: 1,
                    slidesToScroll: 1,
                }
            }
        ]
    });
});
```

github.com/keobrien/WC2018



Getting started: Group by feature

Splitting up files is easier to read & see relationships.

Tip:
Watch out for circular dependencies. They're a nightmare to debug.

- Try to only include files in the same directory or lower
- Avoid any dependencies in helper scripts or shared services.

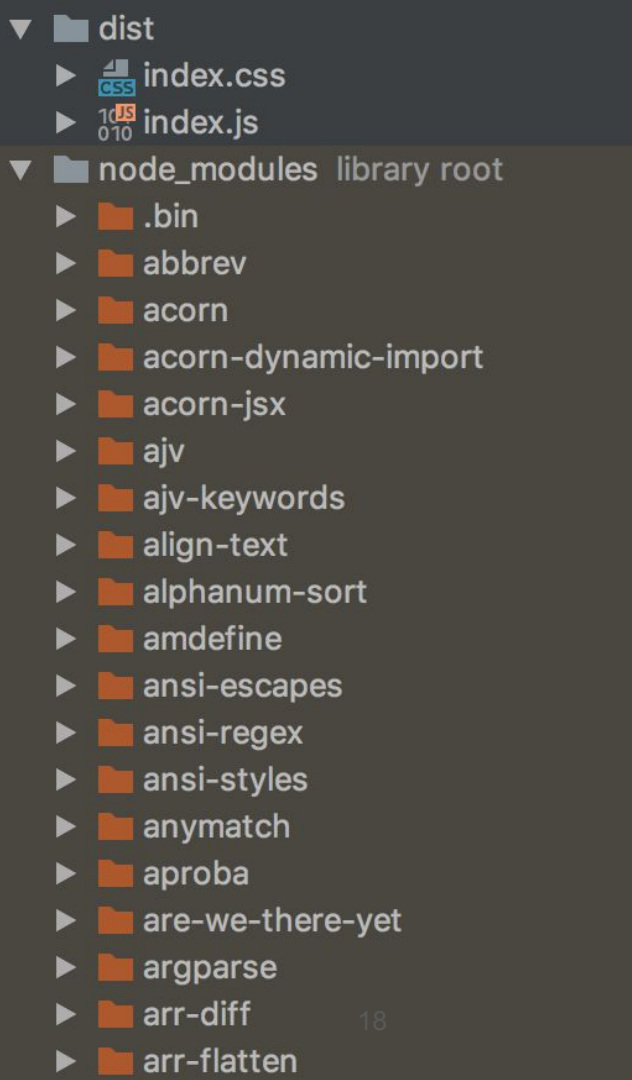
Getting started: Version Control

Exclude dist/ and node_modules/ from version control when possible. (.gitignore)

- Bloats the repository when it can be recreated in minutes
- So many conflicts
- Difficult code reviews

Tip:
Sometimes this is unavoidable because of how a deployment is set up. Resolving conflicts can be done by re-running the build. You should still **NOT** include node modules because WebPack will include any needed code in the built files

github.com/keobrien/WC2018



```
Build  docker  +

> Build setup
> git config --global user.email "$GIT_EMAIL";
> git config --global user.name "$GIT_USER";
> git clone $REPO web
> cd web
> git fetch
> git checkout "$BITBUCKET_BRANCH"
> git pull
> rm -fr wp-content
> cp -r ../wp-content .
> git status
> git add --all .
> git commit --allow-empty -m "Committing build artifacts from BitBu…
> git push origin "$BITBUCKET_BRANCH"
> Build teardown
```

Getting started: Version Control

Separate source from build files with CI/CD

- BitBucket Pipelines
- CircleCI
- GitLab
- Travis CI
- Visual Studio Online

Tip:
When a deployment process requires version control, this is a good way to keep built files out of source code

Getting started: Upload from local

Take the time to set up local development sites

Gulp can help for bulk uploads. Useful when your server does not support Node

Tip:
Lando is great for local sites
https://docs.devwithlando.io/tutorials/wordpress.html

```
const client = require('scp2');

/**
 * Push content to the SFTP server
 **/
gulp.task('push', function() {
    glob(paths.src + '/**/*.*', (err, files) => {
        if (err) {
            console.error('Error pushing', err);
        } else {
            console.info('Found', files.length, 'files');
            async.eachLimit(files, 10, upload('Pushing'));
        }
    });
});


const upload = (msg) => (file, callback = null) => {
  const filePath = file.path ? file.path : file;
  const target = dest(filePath);

  return new Promise((resolve, reject) => {
    client.upload(filePath, target, (e, stat, fd) => {
      if (e) {
        console.error(msg, 'error', target, e);
        if (callback) callback(e);
        return reject(e);
      } else {
        console.info(msg, target, 'success');
        if (callback) callback();
        return resolve();
      }
    });
  });
};

const dest = (localPath) => {
    const root = localPath.startsWith(paths.cwd) ? path.join(paths.cw
    const relative = localPath.replace(root, '');
    return path.join(paths.dest, relative);
};
```

github.com/keobrien/WC2018

Build script

WebPack: Environments

Development

- Tooling for in browser debugging
- Fast incremental compilation
- Useful error messages at run time

Watching for changes

Production

- Small output size
- Fast in browser
- Omitting development only code
- Not exposing source or file paths
- Easy to use output assets

One-time build

WebPack: Config

No config required with WebPack 4

For very basic builds

Without config or provide custom **webpack.config.js**

```
const path = require('path');

module.exports = {
  entry: './src/index.js',
  output: {
    path: path.resolve(__dirname, 'dist'),
    filename: 'bundle.js'
  }
};
```

WebPack: Gutenberg Blocks

create-guten-block

NPM module for zero configuration Gutenberg block build process.

Gutenberg uses React.js and needs a build process.

```
INSIDE: /local_dev_site.tld/wp-content

├── plugin.php
├── package.json
├── README.md
|
├── dist
|   ├── blocks.build.js
|   ├── blocks.editor.build.css
|   └── blocks.style.build.css
|
└── src
    ├── block
    |   ├── block.js
    |   ├── editor.scss
    |   └── style.scss
    |
    ├── blocks.js
    ├── common.scss
    └── init.php
```

github.com/keobrien/WC2018

Dependencies

Entry file(s)

Helpers

```
const argv = require('yargs').argv;
const ExtractTextPlugin = require("extract-text-webpack
const webpack = require('webpack');
const WebpackNotifierPlugin = require('webpack-notifier
const CleanWebpackPlugin = require('clean-webpack-plugi

let config = {
    // https://webpack.github.io/docs/configuration.htm
    // https://docs.npmjs.com/files/package.json#config
    entry : './' + process.env.npm_package_config_paths
    // https://webpack.github.io/docs/configuration.htm
    output: {
        filename: 'index.js',
        // https://docs.npmjs.com/files/package.json#co
        path: __dirname + '/' + process.env.npm_package
    },
    // https://webpack.js.org/configuration/stats/
    stats: {
        // overall build time
        timings: true
    },
    // https://webpack.github.io/docs/configuration.htm
    profile: true,
    // https://webpack.github.io/docs/configuration.htm
    // Example: webpack --watch
    watch : !!argv.watch,
    module: {
        rules: [
            // Javascript processing
            // https://www.npmjs.com/package/babel-load
            // https://babeljs.io/
            // Enables ES6 javascript for all browsers
            {
                test : /\.js$/
```

github.com/keobrien/WC2018

WebPack

Loaders / Modules

- **Test** for files coming through
- **Use** a **loader**
- Configure with **options**

This loader transforms ES6 code to ES5 so all browsers can understand it.

```
// Javascript processing
// https://www.npmjs.com/package/bal
// https://babeljs.io/
// Enables ES6 javascript for all b
{
    test    : /\.js$/,
    exclude: '/node_modules/',
    use : {
        loader: 'babel-loader',
        options: {
            "presets": ["env"],
            "comments": false
        }
    }
},
```

WebPack: SASS

Css-loader

Tells WebPack how to load and read css files

Postcss-loader

Automatically add vendor prefixes

Sass-loader

Tells WebPack how to load and read sass files

```
// CSS processing
// http://sass-lang.com/
{
    test: /\.scss$/,
    // https://www.npmjs.com/package/extract-text-webpack-plugin
    use: ExtractTextPlugin.extract({
        fallback: 'style-loader',
        use: [
            // https://www.npmjs.com/package/css-loader
            {
                loader: 'css-loader',
                options: {
                    // Minimize only if building for production to optimize build times
                    // Example: export NODE_ENV=production && webpack
                    minimize: (process.env.NODE_ENV === 'production'),
                    sourceMap: true
                }
            },
            // Auto-prefix css to automatically add vendor prefixes.
            // https://www.npmjs.com/package/postcss-loader
            {
                loader: 'postcss-loader',
                options: {
                    ident: 'postcss',
                    sourceMap: true,
                    plugins: function(loader)  {
                        return [
                            // https://github.com/postcss/autoprefixer
                            require('autoprefixer')({
                                browsers: ['last 2 versions', 'ie >= 9', 'and_chr >= 2.3']
                            })
                        ];
                    }
                }
            },
            // https://www.npmjs.com/package/sass-loader
            {
                loader: 'sass-loader',
                options: {
                    sourceMap: true
                }
            }
        ]
    })
},
```

github.com/keobrien/WC2018

WebPack: Everything else

Avoid image processing, instead optimize images when adding them to your project so you don't need to wait every time you make a CSS change.

```
// Copy any additional files used in production but not required a specific loader.
// Needed to copy over fonts, images, etc from npm packages.
// https://www.npmjs.com/package/file-loader
{
    test: /\.(eot|ttf|woff|woff2|svg|png|jpg|gif)$/,
    use: [
        {
            loader: 'file-loader',
            options: {
                name: '[name].[hash].[ext]',
                context: './' + process.env.npm_package_config_paths_source
            }
        }
    ]
}
```

WebPack

Plugins

Add any non-file specific tasks

Devtool

Sourcemaps

Tip:
Some sourcemaps are faster than others

Tip:
Use ExtractTextPlugin to avoid javascript injecting CSS

github.com/keobrien/WC2018

```
        },
        // Copy any additional files used in pro
        // Needed to copy over fonts, images, et
        // https://www.npmjs.com/package/file-lo
        {
            test: /\.(eot|ttf|woff|woff2|svg|png
            use: [
                {
                    loader: 'file-loader',
                    options: {
                        name: '[name].[hash].[ex
                        context: './' + process.
                    }
                }
            ]
        }
    ]
},
plugins: [
    // https://www.npmjs.com/package/clean-webpa
    // https://webpack.js.org/guides/output-mana
    new CleanWebpackPlugin([process.env.npm_pack
    // https://www.npmjs.com/package/extract-tex
    // https://webpack.github.io/docs/styleshee
    new ExtractTextPlugin("index.css"),
    // https://www.npmjs.com/package/webpack-not
    new WebpackNotifierPlugin({alwaysNotify: tru
],
// https://webpack.js.org/configuration/devtool/
// Fastest source map that works with text extra
devtool: "#cheap-module-source-map"
```

WebPack

Production Build

Time intensive, more optimizations

Tip:
Use environment variables for triggering production builds.

```
"scripts": {
  "start": "./node_modules/.bin/webpack --watch",
  "build-production": "export NODE_ENV=production &&
  "precommit": "npm run vet",
  "lint": "./node_modules/.bin/eslint src/",
  "vet": "npm list --depth=0 && npm run lint"
},
```

```
// Production override settings.
// See build-prod script in package.json
// export NODE_ENV=production && ./node_modul
if (process.env.NODE_ENV === 'production') {

    config.plugins.push(
        // https://webpack.github.io/docs/lis
        new webpack.optimize.UglifyJsPlugin({
            sourceMap: true,
            compress: {
                warnings: false
            }
        }
    ));

    // https://webpack.js.org/configuration/d
    // Different from dev. This version is sl
    config.devtool = "#source-map";

}

module.exports = config;
```

github.com/keobrien/WC2018

Advanced Use Cases

Advanced: Multiple Node Versions

NVM: Node Version Manager

N: Another Node version manager

Docker: Can install a different version of Node per project

Lando: Can install Node as a service, to run in Docker

Advanced: Two theme styles

Npm concurrently can launch multiple processes

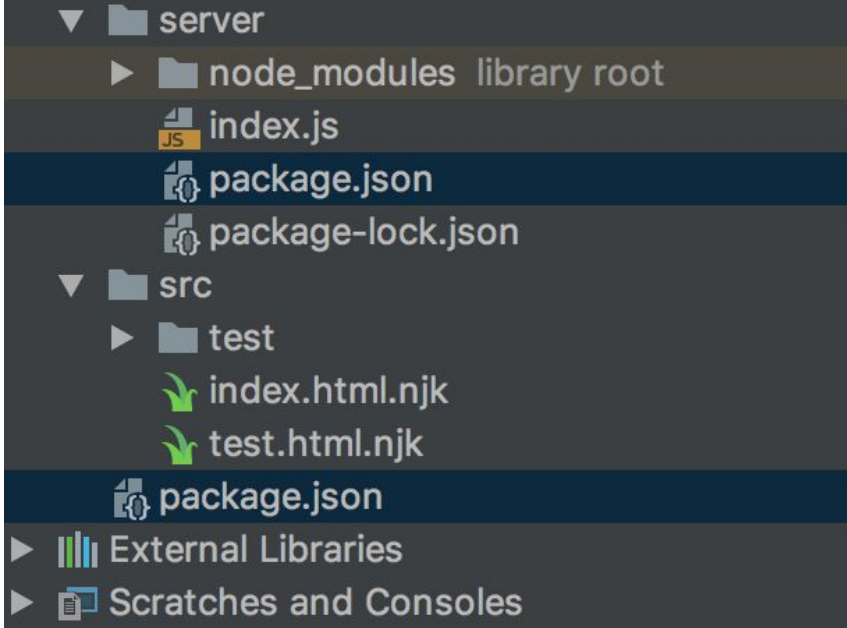
Tip:
Rather than installing global packages, install locally and use node_modules/.bin/...

```
"main": "index.js",
"scripts": {
  "start": "concurrently \"npm:dev-theme1\" \"npm:dev-theme2\"",
  "build-all": "npm run build-theme1 && npm run build-theme2",
  "build-theme1": "export NODE_ENV=production && ./node_modules/.bin/webpack --config ./webpack/theme1.
  "build-theme2": "export NODE_ENV=production && ./node_modules/.bin/webpack --config ./webpack/theme2.
  "dev-theme1": "./node_modules/.bin/webpack --config ./webpack/theme1.config.js --watch",
  "dev-theme2": "./node_modules/.bin/webpack --config ./webpack/theme2.config.js --watch",
  "git-add-working": "git add ../common_2018 ../theme2_2018 ../theme1_2018 . && git reset ../theme1_201
  "git-reset-dist": "git checkout -- ../theme1_2018/dist ../theme2_2018/dist",
  "test": "jest"
},
"author": "",
"license": "ISC",
"dependencies": {
  "@fortawesome/fontawesome-free": "^5.1.0",
```

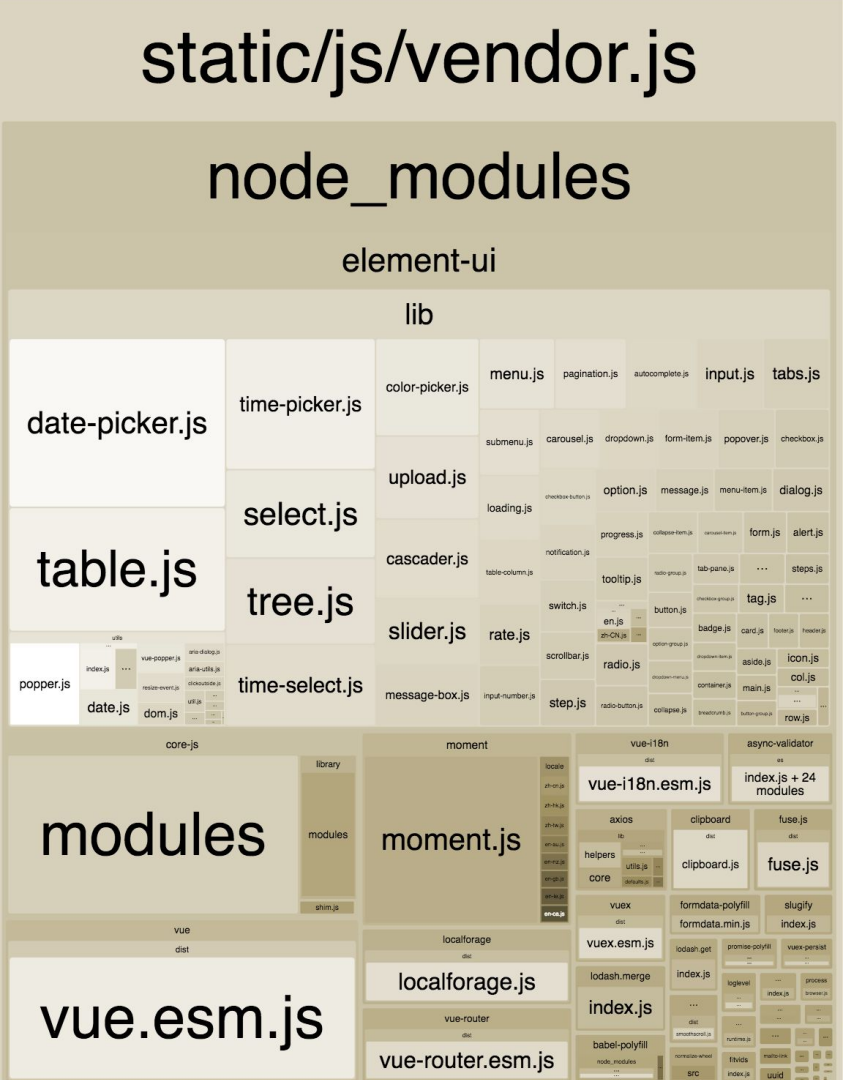
Advanced: Multiple package.json

Splitting projects makes it easier to manage dependencies

Root project needs to manage child folders



github.com/keobrien/WC2018



Advanced: Debugging

Webpack-bundle-analyzer can help debug parts of the app that are too big.

Tip:
Moment.js can often huge builds if additional languages are not excluded.

Advanced: Nuclear option

Inheriting a very old build

- Date the project by looking up release dates of modules and node releases.
- Delete the package.json dependencies, package-lock.json (or npm-shrinkwrap.json)
- Re-install with one giant npm install --save command and let npm work it out

This is a last resort but it has saved me

```
Kevins-MacBook-Pro:wp-build kevinobrien$ npm install --save autoprefixer babel-core
babel-loader babel-preset-env clean-webpack-plugin css-loader eslint eslint-config-c
lockwork extract-text-webpack-plugin file-loader husky node-sass postcss-loader sass
-loader style-loader webpack webpack-notifier yargs
```

Advanced: Errors

Always start with “E…”

```
npm ERR! code EJSONPARSE
npm ERR! Failed to parse json
npm ERR! Unexpected token p in JSON at position 1 while parsing near 'npm install --save
npm ERR! File: /Users/kevinobrien/Clockwork/code/wp-build/web/wp-content/themes/wordpres
npm ERR! Failed to parse package.json data.
npm ERR! package.json must be actual JSON, not just JavaScript.
npm ERR!
npm ERR! Tell the package author to fix their package.json file. JSON.parse

npm ERR! A complete log of this run can be found in:
npm ERR!     /Users/kevinobrien/.npm/_logs/2018-08-22T03_56_39_742Z-debug.log
```

Questions?

joind.in/talk/77937 (kevin@clockwork.com)