1 Travailler sur le thème « classic » de Prestashop

1.1 Introduction

Le dernier Prestashop 1.7.1 est fourni avec le thème « classic ».

Je n'ai pas réussi à utiliser webpack… pour le moment.

Du coup j'ai entrepris d'utiliser Gulp seulement il n'y a pas de ressource là-dessus.

Seulement j'ai été confronté à des problèmes de version puisqu'il utilise le récent (pas le dernier) Bootstrap 4 alpha.4.

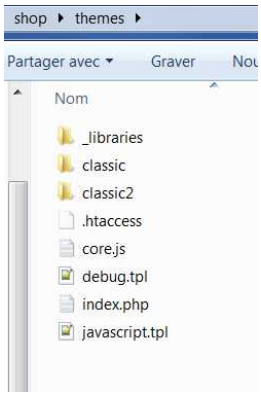
Je vais donc expliquer comment faire.

1.2 Dupliquer le thème

Dans le répertoire de thème de Prestashop, dupliquer le thème.

Modifier \config\theme.yml en conséquence.

Ex : classic2



Maintenant, tout va se passer dans le répertoire suivant :

\themes\classic2_dev

1.3 Installation du pré-processeur

Installation du 11/07/2017 - J'ai réussi

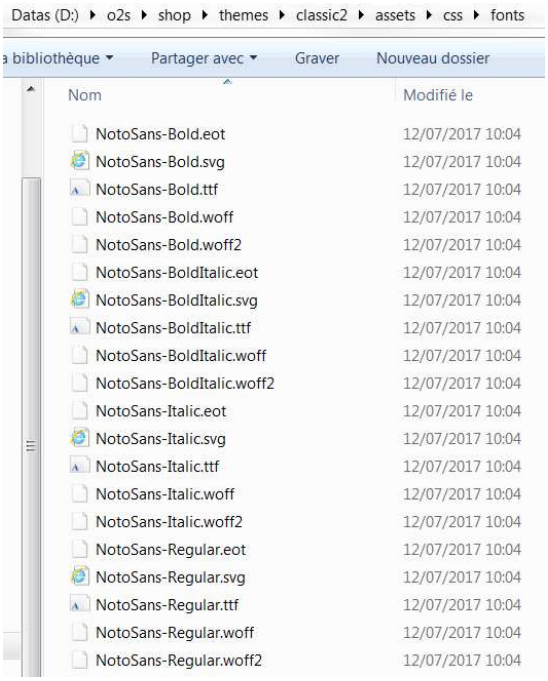
- $ npm init
  → Je fais mon fichier package.json
- Créer un fichier vide README.md
  → Inutile mais ça enlève le message d'erreur
- $ npm install -g gulp
  → Créé d'un répertoire nodemodule
- $ npm install --save-dev gulp
  → Installation de la dépendence
- $ npm install --save-dev gulp-sass
  → Installation de la dépendence
- $ npm install --save bourbon
  → installation de bourbon
- $ npm install bootstrap@4.0.0-alpha.4 –save
  → Attention, le thème utilise en fait bootstrap 4.0.0-alpha.4 et non bootstrap 3 ni alpha.6
- $ npm install --save bootstrap-touchspin
  → installation de bootstrap-touchspin
- Modification de theme.scss (theme classic de prestashop 1.7)
  → Attention, ne pas mettre .scss
  @import "../node_modules/bourbon/app/assets/stylesheets/_bourbon";
  @import "../node_modules/bootstrap/scss/bootstrap";
  @import "../node_modules/bootstrap-touchspin/dist/jquery.bootstrap-touchspin.min.css";
- Ajouter en haut de theme.scss : $output-bourbon-deprecation-warnings: false;
  → Cela va désactiver les messages d'erreurs de touchspin

1.4 Polices de caractères

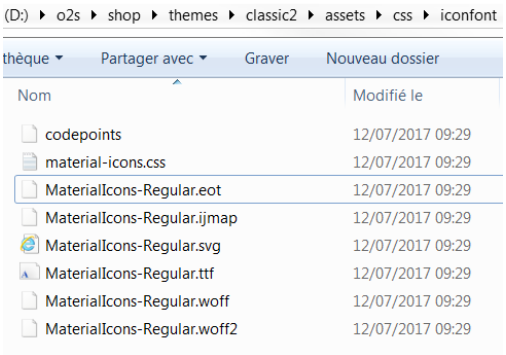
- $ npm install material-design-icons
  → Installation des fonts icones

- npm install notosans-fontface –save
  → Installation des fonts icones

- Dans le répertoire \assets\css créer un répertoire fonts et un répertoire iconfont
  → css\fonts
  → css\iconfonts
- Modifier les chemins dans **_fonts.scss**
  → Exemple : url(~notosans-fontface/fonts/NotoSans-Regular.eot) devient url(fonts/NotoSans-Regular.eot)
  → Exemple : url(~material-design-icons/iconfont/MaterialIcons-Regular.eot) devient url(iconfont/MaterialIcons-Regular.eot)
- Copier les fonts NotoSans dans le répertoire \assets\css\fonts



- Copier les fonts MaterialIcons dans le répertoire \assets\css\iconfont



1.5 Élaboration du Gulpfiles.js

- Création du fichier gulpfiles.js
  → Attention aux chemins
  → **ULTRA IMPORTANT :** Ne pas compiler tout le contenu du répertoire CSS mais simplement le fichier theme.css
- $ npm install --save-dev gulp-cssmin
  → minifier le CSS
- $ npm install gulp-clean-css --save-dev
  → minifier le CSS (je crois que le précédent gulp-cssmin fait pareil)
- $ npm install gulp-rename --save-dev
  → renommer les fichiers
- Créer un répertoire preprod dans _dev
  → _dev\preprod
  → C'est pour créer les fichiers avec gulp
- Créer un répertoire preprod-min dans _dev
  → _dev\preprod-min
  → C'est pour créer les fichiers minifiés avant de les copier en "prod"



Gulfiles.js

```
var gulp = require('gulp');
var sass = require('gulp-sass');
var cleanCSS = require('gulp-clean-css');
var rename = require("gulp-rename");

gulp.task('traitement-sass', function() {
                gulp.src('css/*.scss')
                                .pipe(sass().on('error', sass.logError))
                                .pipe(gulp.dest('preprod/'))
});

// Minify CSS
gulp.task('minify-css', function() {
   return gulp.src('preprod/*.css')
      .pipe(cleanCSS({ compatibility: 'ie8' }))
      .pipe(gulp.dest('preprod-min'))
});

// Copy CSS files from preprod to prod (\assets\css)
gulp.task('cssfiles', function() {
   return gulp.src(['preprod-min/**/*'])
      .pipe(gulp.dest('../assets/css'))
})

// Copy all dependencies from node_modules - Commande à part mais je peux le mettre dans "default"
gulp.task('copy', ['cssfiles']);

gulp.task('default', ['traitement-sass', 'minify-css', 'cssfiles'], function() {
   gulp.watch('css/**/*.scss', ['traitement-sass']);
   gulp.watch('preprod/**/*.css', ['minify-css']);
                gulp.watch('preprod-min/**/*.css', ['cssfiles']);
});
```

Maintenant pour bosser :

$ gulp
→ Cela va lancer la tâche « default »

$ gulp traitement-sass
→ Cela va juste compiler le scss en css

1.6 Package.json

package.json

```
{
  "name": "dngps1",
  "version": "1.0.0",
  "private": true,
  "description": "My first prestashop theme",
  "main": "webpack.config.js",
  "scripts": {
    "test": "echo 'hello"'
  },
  "author": "Donny Nguyen",
  "license": "ISC",
  "devDependencies": {
    "gulp": "^3.9.1",
    "gulp-clean-css": "^3.6.0",
    "gulp-cssmin": "^0.2.0",
    "gulp-rename": "^1.2.2",
    "gulp-sass": "^3.1.0"
  },
  "dependencies": {
    "bootstrap": "^4.0.0-alpha.4",
    "bootstrap-touchspin": "^3.1.1",
    "bourbon": "^4.3.4",
    "material-design-icons": "^3.0.1",
    "notosans-fontface": "^1.0.1"
  }
}
```

2 Annexes

2.1 Commentaire global

A la base j'avais plein de messages d'erreurs car dans gulp je compilais les fichiers un par un.
En fait compiler juste theme.scss suffit.

Par exemple :
OK gulp.src('css/theme.scss')
BAD gulp.src('css/**/*.scss')

Beaucoup d'erreurs sont provoquées par le Gulpfile.

2.2 Erreurs commises

- J'ai installé la mauvaise version de Bootstrap alors que le thème « classic » utilise Bootstrap@4
  i.e. $ npm install --save bootstrap@3
  → installation de bootstrap mais installe la version avec LESS
- J'ai installé la mauvaise version de Bootstrap@4, alors que le thème « classic » utilise l'alpha 4
  i.e. $ npm install bootstrap@4.0.0-alpha.6 –save
  → Le thème est bati sur la version alpha.4 et non la alpha.6
- Pas une vraie erreur mais Bootstrap@4 est fourni avec LESS et SASS
  i.e. $ npm install --save bootstrap-sass
  → installation de bootstrap-sass