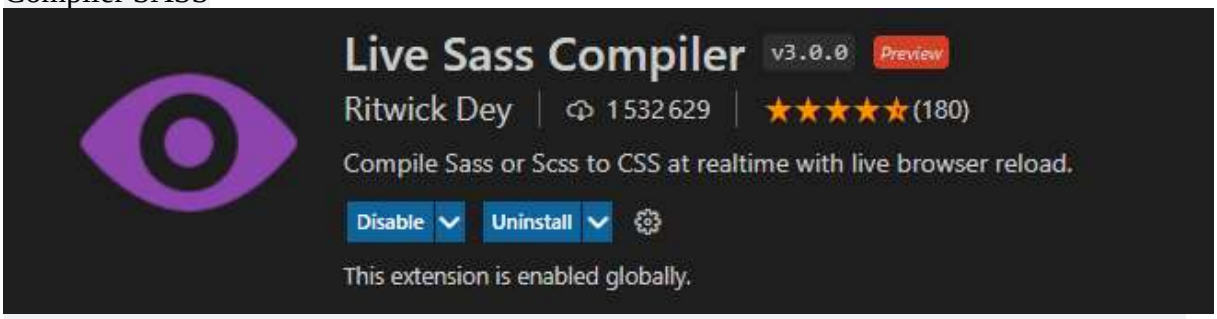
CSS Précis et concis

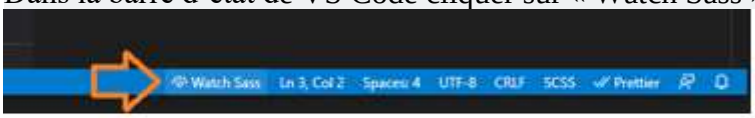
VS Code extensions

IntelliSense CSS
IntelliSense for CSS class names in HTML

Compiler SASS



Dans la barre d'état de VS Code cliquer sur « Watch Sass » pour compiler automatiquement



Compilation SASS

Avec l'extension **Live SASS Compiler**

Ou mieux en installant **sass** (globalement ou localement au projet)
(ajouter un package.json au projet avec « npm init -f »)

```
npm i sass -D
```

Puis ajout d'un script (ou en ligne de commande si sass installé globalement)

```
  "scripts": {
    "sass": "sass --watch sass/style.scss css/style.css"
  },
```

« style » ou nom de la librairie

Bootstrap 5

Display

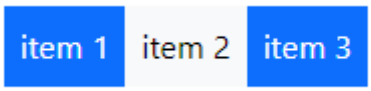
- **« d-inline »** passe en inline
- **« d-inline-block »** passe en inline-block
- **« d-block »** pour passer en block
- **Flexbox** *avec « d-flex »*

Flexbox avec Bootstrap

https://getbootstrap.com/docs/5.0/utilities/flex/

```
<div class="d-flex">
    <div class="p-2 bg-primary text-white">item 1</div>
    <div class="p-2 bg-light text-black">item 2</div>
    <div class="p-2 bg-primary text-white">item 3</div>
```

```
</div>
```



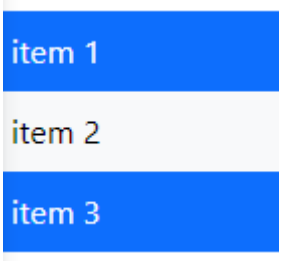
« **justify-content-…** » pour la disposition **horizontale** des children

- « start » met les éléments à gauche
- « end » met les éléments à droite
- « center » éléments centrés
- « around »
- « between »

« **align-items-…** » pour la disposition **verticale** des children

« flex-colum » pour empiler les éléments verticalement. Il faut alors utiliser « align-items » pour définir la disposition horizontalement

```
<div class="d-flex flex-column align-items-start">
    <div class="p-2 bg-primary text-white">item 1</div>
    <div class="p-2 bg-light text-black">item 2</div>
    <div class="p-2 bg-primary text-white">item 3</div>
</div>
```



Navbar

```
.navbar {
    position: relative;
    display: flex;
    flex-wrap: wrap;
    align-items: center;
    justify-content: space-between;
    padding-top: .5rem;
    padding-bottom: .5rem;
}
```

Système de margin et padding

Margin
« m-… » margin tout autour
« mt-… » margin top
« mr-… » margin right
« mb-… » margin bottom

Padding
« p-… » padding tout autour
« pt-… » padding top
« pr-… » padding right
« pb-… » padding bottom

« ml-… » margin left
« mx-… » margin left et right
« my-… » margin top et bottom
« mx-auto » centre le contnu (équivalent à margin : 0 auto)

« pl-… » padding left
« px-… » padding left et right
« py-… » padding top et bottom

Valeurs e 0 à 5

- 0 : 0rem
- 1 : 0.25rem (4px)
- 2 : 0.5rem (8px)
- 3 : 1rem (16px)
- 4 : 1.5rem (24px)

Exemples : « mb-4 » ou « pb-4 »

Couleurs

Background

« bg-… »

- « bg-light »
- « bg-dark »
- « bg-black »
- « bg-primary »
- « bg-secondary »
- « bg-success »
- « bg-danger »
- « bg-warning »

Couleurs de textes

- « text-primary »
- « text-secondary »
- « text-success »
- « text-danger »
- « text-warning »
- « text-info »
- « text-light »
- « text-dark »
- « text-muted »
- « text-white »

Plus « **invisible** » pour **cacher un élément** (« visibility: hidden!important »)

Couleurs et font de base

```
:root {
    --bs-blue: #0d6efd;
    --bs-indigo: #6610f2;
    --bs-purple: #6f42c1;
    --bs-pink: #d63384;
    --bs-red: #dc3545;
    --bs-orange: #fd7e14;
    --bs-yellow: #ffc107;
    --bs-green: #198754;
    --bs-teal: #20c997;
    --bs-cyan: #0dcaf0;
    --bs-white: #fff;
    --bs-gray: #6c757d;
    --bs-gray-dark: #343a40;
    --bs-gray-100: #f8f9fa;
    --bs-gray-200: #e9ecef;
    --bs-gray-300: #dee2e6;
```

```
    --bs-gray-400: #ced4da;
    --bs-gray-500: #adb5bd;
    --bs-gray-600: #6c757d;
    --bs-gray-700: #495057;
    --bs-gray-800: #343a40;
    --bs-gray-900: #212529;
    --bs-primary: #0d6efd;
    --bs-secondary: #6c757d;
    --bs-success: #198754;
    --bs-info: #0dcaf0;
    --bs-warning: #ffc107;
    --bs-danger: #dc3545;
    --bs-light: #f8f9fa;
    --bs-dark: #212529;
    --bs-primary-rgb: 13,110,253;
    --bs-secondary-rgb: 108,117,125;
    --bs-success-rgb: 25,135,84;
    --bs-info-rgb: 13,202,240;
    --bs-warning-rgb: 255,193,7;
    --bs-danger-rgb: 220,53,69;
    --bs-light-rgb: 248,249,250;
    --bs-dark-rgb: 33,37,41;
    --bs-white-rgb: 255,255,255;
    --bs-black-rgb: 0,0,0;
    --bs-body-rgb: 33,37,41;
    --bs-font-sans-serif: system-ui,-apple-
system,"Segoe UI",Roboto,"Helvetica Neue",Arial,"Noto Sans","Liberation Sans",sans-
serif,"Apple Color Emoji","Segoe UI Emoji","Segoe UI Symbol","Noto Color Emoji";
    --bs-font-monospace: SFMono-Regular,Menlo,Monaco,Consolas,"Liberation Mono","Courier New",monospace;
    --bs-gradient: linear-gradient(180deg, rgba(255, 255, 255, 0.15), rgba(255, 255, 255, 0));
    --bs-body-font-family: var(--bs-font-sans-serif);
    --bs-body-font-size: 1rem;
    --bs-body-font-weight: 400;
    --bs-body-line-height: 1.5;
    --bs-body-color: #212529;
    --bs-body-bg: #fff;
}
```

Exemple d'utilisation: pour définir le background des paragraphes

```
<style>
        p {
            background-color: var(--bs-indigo);
        }
</style>
```

Materialize

https://materializecss.com/getting-started.html
Github: https://github.com/Dogfalo/materialize

CDN: Dans la balise "head" (après le title)

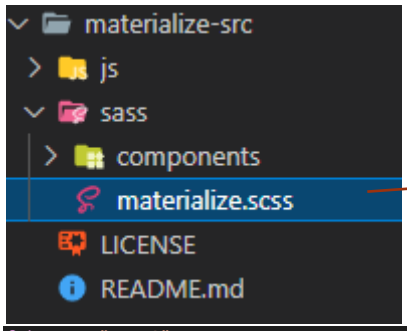
```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Title</title>
        <!--Import Google Icon Font-->
        <link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">
        <!-- Compiled and minified CSS -->
        <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/materialize/1.0.0/css/materialize.min.
css">
    </head>
    <body>

        <!-- Compiled and minified JavaScript -->
        <script src="https://cdnjs.cloudflare.com/ajax/libs/materialize/1.0.0/js/materialize.min.js"></script>
    </body>
</html>
```

Sources: possibilité de télécharger les sources, également au format SASS

NPM

npm install materialize-css@next

Structure

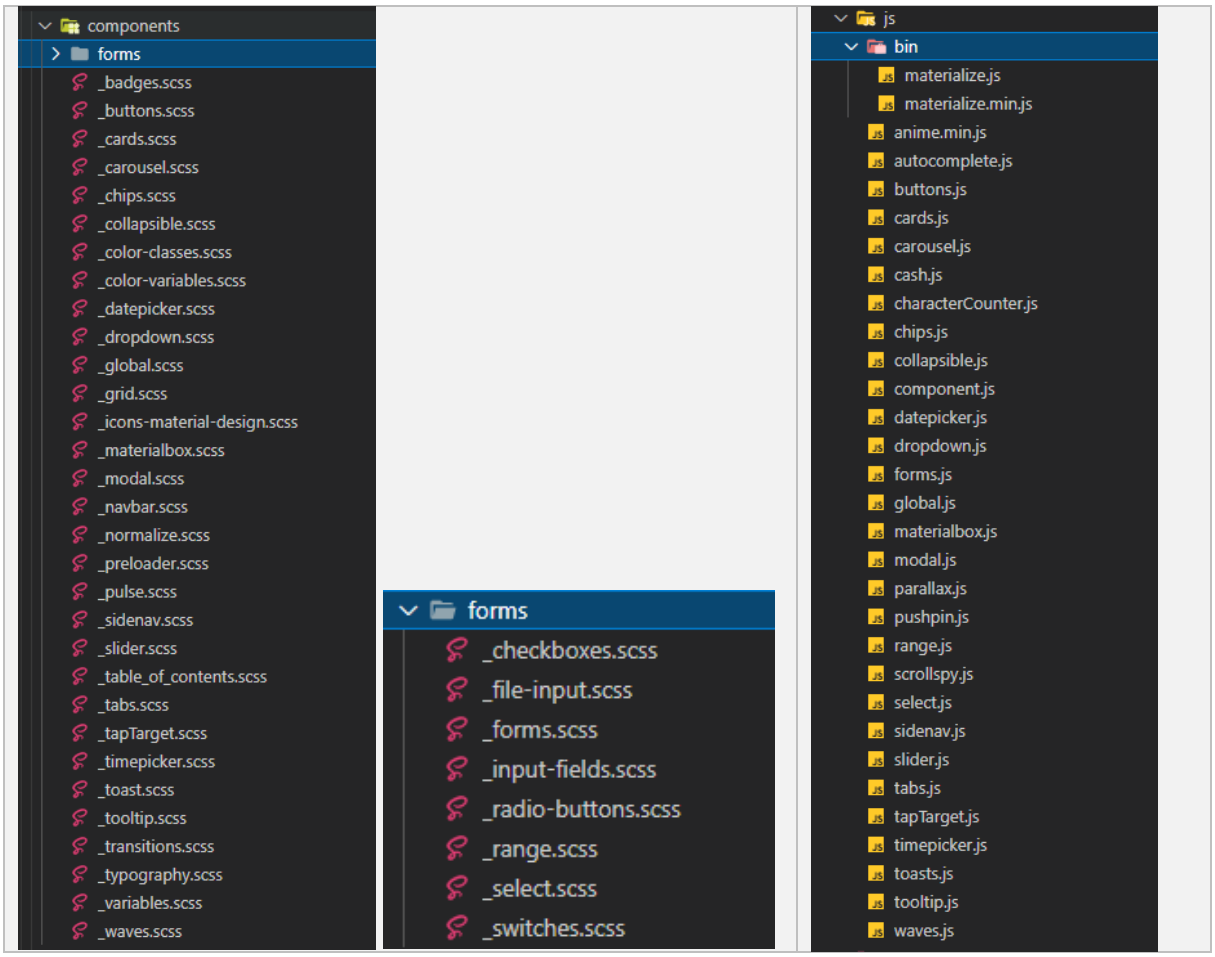


Fichier avec tous les imports

```
@charset "UTF-8";

// Color
@import "components/color-variables";
@import "components/color-classes";

// Variables;
@import "components/variables";

// Reset
@import "components/normalize";

// components
@import "components/global";
@import "components/badges";
@import "components/icons-material-design";
@import "components/grid";
@import "components/navbar";
@import "components/typography";
@import "components/transitions";
@import "components/cards";
@import "components/toast";
@import "components/tabs";
@import "components/tooltip";
@import "components/buttons";
@import "components/dropdown";
@import "components/waves";
@import "components/modal";
@import "components/collapsible";
@import "components/chips";
@import "components/materialbox";
@import "components/forms/forms";
@import "components/table_of_contents";
@import "components/sidenav";
@import "components/preloader";
@import "components/slider";
@import "components/carousel";
@import "components/tapTarget";
@import "components/pulse";
@import "components/datepicker";
@import "components/timepicker";
```

On peut ignorer l'underscore du nom des fichiers « _global.scss » => « global »

components	Js



« **normalize.scss** » utilisée pour le reset css https://necolas.github.io/normalize.css/

Typographie

Fichier "_typography.scss"
Font families utilisées : Segoe UI, Roboto …
Exemple de html

```
html {
  line-height: 1.5;
  font-family: -apple-system, BlinkMacSystemFont, "Segoe UI", Roboto, Oxygen-
Sans, Ubuntu, Cantarell, "Helvetica Neue", sans-serif;
  font-weight: normal;
  color: rgba(0, 0, 0, 0.87);
}

@media only screen and (min-width: 0) {
  html {
    font-size: 14px;
  }
}

@media only screen and (min-width: 992px) {
  html {
    font-size: 14.5px;
  }
}

@media only screen and (min-width: 1200px) {
  html {
    font-size: 15px;
  }
}
```

Titres

un style s'applicque automatiquement aux titres de h1 à h6

```
<h1>Hello World</h1>
<h2>Hello World</h2>
<h3>Hello World</h3>
<h4>Hello World</h4>
<h5>Hello World</h5>
<h6>Hello World</h6>
```

Exemple de CSS pour les titres

```
h1,
h2,
h3,
h4,
h5,
h6 {
  font-weight: 400;
  line-height: 1.3;
}

h1 a,
h2 a,
h3 a,
h4 a,
h5 a,
h6 a {
  font-weight: inherit;
}

h1 {
  font-size: 4.2rem;
  line-height: 110%;
  margin: 2.8rem 0 1.68rem 0;
}

h2 {
  font-size: 3.56rem;
  line-height: 110%;
  margin: 2.37333rem 0 1.424rem 0;
}

h3 {
  font-size: 2.92rem;
  line-height: 110%;
  margin: 1.94667rem 0 1.168rem 0;
}

h4 {
  font-size: 2.28rem;
  line-height: 110%;
  margin: 1.52rem 0 0.912rem 0;
}

h5 {
  font-size: 1.64rem;
  line-height: 110%;
  margin: 1.09333rem 0 0.656rem 0;
}

h6 {
  font-size: 1.15rem;
  line-height: 110%;
  margin: 0.76667rem 0 0.46rem 0;
}
```

blockquote

Citation **blockquote** : style appliqué automatiquement

```
<blockquote>Exemple de citation.</blockquote>
```

> Exemple de citation.

Exemple de CSS pour blockquote (la couleur dépend de la primary-color)

```
blockquote {
  margin: 20px 0;
  padding-left: 1.5rem;
  border-left: 5px solid #ee6e73;
}
```

Paragraphes

« **flow-text** » pour mettre en évidence un paragraphe par exemple

P simple

Lorem ipsum dolor sit amet consectetur adipisicing elit. Incidunt ex, libero molestiae necessitatibus tenetur quae quo accusantium, labore voluptatum hic maxime. A cum, rerum molestias eius assumenda esse iure quidem.

Lorem ipsum dolor sit amet consectetur adipisicing elit. Incidunt ex, libero molestiae necessitatibus tenetur quae quo accusantium, labore voluptatum hic maxime. A cum, rerum molestias eius assumenda esse iure quidem.

Avec « flow-text »

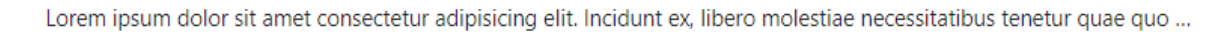
```
<p class="flow-
text">Lorem ipsum dolor sit amet consectetur adipisicing elit. Incidunt ex, libero molestiae necessitatibus tenetur q
uae quo accusantium, labore voluptatum hic maxime. A cum, rerum molestias eius assumenda esse iure quidem.</p>
```

Exemple de CSS

```
@media only screen and (min-width: 360px) {
  .flow-text {
    font-size: 1.2rem;
  }
}
@media only screen and (min-width: 390px) {
  .flow-text {
    font-size: 1.224rem;
  }
}
@media only screen and (min-width: 420px) {
  .flow-text {
    font-size: 1.248rem;
  }
}
@media only screen and (min-width: 450px) {
  .flow-text {
    font-size: 1.272rem;
  }
}
@media only screen and (min-width: 480px) {
  .flow-text {
    font-size: 1.296rem;
  }
}
@media only screen and (min-width: 510px) {
  .flow-text {
    font-size: 1.32rem;
  }
}
@media only screen and (min-width: 540px) {
  .flow-text {
    font-size: 1.344rem;
  }
}
@media only screen and (min-width: 570px) {
  .flow-text {
    font-size: 1.368rem;
  }
}
@media only screen and (min-width: 600px) {
  .flow-text {
    font-size: 1.392rem;
  }
}
@media only screen and (min-width: 630px) {
  .flow-text {
    font-size: 1.416rem;
  }
```

```
}
@media only screen and (min-width: 660px) {
  .flow-text {
    font-size: 1.44rem;
  }
}
@media only screen and (min-width: 690px) {
  .flow-text {
    font-size: 1.464rem;
  }
}
@media only screen and (min-width: 720px) {
  .flow-text {
    font-size: 1.488rem;
  }
}
@media only screen and (min-width: 750px) {
  .flow-text {
    font-size: 1.512rem;
  }
}
@media only screen and (min-width: 780px) {
  .flow-text {
    font-size: 1.536rem;
  }
}
@media only screen and (min-width: 810px) {
  .flow-text {
    font-size: 1.56rem;
  }
}
@media only screen and (min-width: 840px) {
  .flow-text {
    font-size: 1.584rem;
  }
}
@media only screen and (min-width: 870px) {
  .flow-text {
    font-size: 1.608rem;
  }
}
@media only screen and (min-width: 900px) {
  .flow-text {
    font-size: 1.632rem;
  }
}
@media only screen and (min-width: 930px) {
  .flow-text {
    font-size: 1.656rem;
  }
}
@media only screen and (min-width: 960px) {
  .flow-text {
    font-size: 1.68rem;
  }
}
@media only screen and (max-width: 360px) {
  .flow-text {
    font-size: 1.2rem;
  }
}
```

« **truncate** » pour tronquer un texte
Au lieu d'aligner à la ligne le texte est tronqué avec 3 points




```
<p class="truncate">Lorem ipsum dolor sit amet consectetur adipisicing elit. Incidunt ex, libero molestiae necessitat
ibus tenetur quae quo accusantium, labore voluptatum hic maxime. A cum, rerum molestias eius assumenda esse iure quid
em.</p>
```

Exemple de CSS

```
.truncate {
  display: block;
  white-space: nowrap;
  overflow: hidden;
  text-overflow: ellipsis;
}
```

Couleurs

Background color

Palettes de couleurs: https://materializecss.com/color.html

```
<div class="card-panel purple lighten-5">Purple</div>
<div class="card-panel purple lighten-3">Purple</div>
<div class="card-panel purple lighten-1">Purple</div>
<div class="card-panel purple">Purple</div>
<div class="card-panel purple darken-1">Purple</div>
<div class="card-panel purple darken-4">Purple</div>
<div class="card-panel purple accent-1">Purple</div>
<div class="card-panel purple accent-4">Purple</div>
```

On indique la couleur …puis une classe css pour l'éclaircir ou l'accentuer

« lighten-… » 1 à 5 éclaircit de plus en plus
« darken-… » 1 à 4 de plus en plus foncé
« accent-… » 1 à 4

Exemple de variable de couleur scss

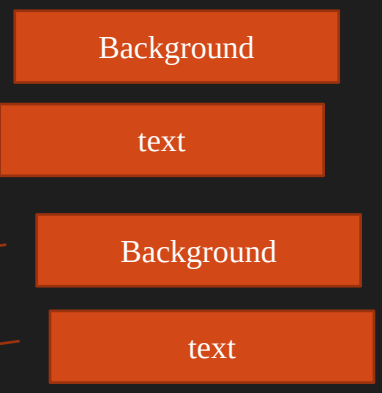
```scss
$purple: (
  "base":       #9c27b0,
  "lighten-5":  #f3e5f5,
  "lighten-4":  #e1bee7,
  "lighten-3":  #ce93d8,
  "lighten-2":  #ba68c8,
  "lighten-1":  #ab47bc,
  "darken-1":   #8e24aa,
  "darken-2":   #7b1fa2,
  "darken-3":   #6a1b9a,
  "darken-4":   #4a148c,
  "accent-1":    #ea80fc,
  "accent-2":    #e040fb,
  "accent-3":    #d500f9,
  "accent-4":    #aa00ff
);
```

Pour générer les différentes classes CSS. On a un variable répertoriant toutes les couleurs définies comme pour « purple » (exemple ci-dessus)

```scss
$colors: (
  "materialize-red": $materialize-red,
  "red": $red,
  "pink": $pink,
  "purple": $purple,
  "deep-purple": $deep-purple,
  "indigo": $indigo,
  "blue": $blue,
  "light-blue": $light-blue,
  "cyan": $cyan,
  "teal": $teal,
  "green": $green,
  "light-green": $light-green,
  "lime": $lime,
  "yellow": $yellow,
  "amber": $amber,
  "orange": $orange,
  "deep-orange": $deep-orange,
  "brown": $brown,
  "blue-grey": $blue-grey,
  "grey": $grey,
  "shades": $shades
) !default;
```

Puis une boucle pour générer les différentes classes css pour les backgrounds et textes

```scss
// Color Classes

@each $color_name, $color in $colors {
  @each $color_type, $color_value in $color {
    @if $color_type == "base" {
      .#{$color_name} {
        background-color: $color_value !important;
      }
      .#{$color_name}-text {
        color: $color_value !important;
      }
    }
    @else if $color_name != "shades" {
      .#{$color_name}.#{$color_type} {
        background-color: $color_value !important;
      }
      .#{$color_name}-text.text-#{$color_type} {
        color: $color_value !important;
      }
    }
  }
}

// Shade classes
@each $color, $color_value in $shades {
  .#{$color} {
    background-color: $color_value !important;
  }
  .#{$color}-text {
    color: $color_value !important;
  }
}
```



Text color

Lorem ipsum dolor sit amet consectetur adipisicing elit.

Lorem ipsum dolor sit amet consectetur adipisicing elit.

Lorem ipsum dolor sit amet consectetur adipisicing elit.

```html
<p class="blue-text text-lighten-3">Lorem ipsum dolor sit amet consectetur adipisicing elit.</p>
<p class="blue-text">Lorem ipsum dolor sit amet consectetur adipisicing elit.</p>
<p class="blue-text text-darken-3">Lorem ipsum dolor sit amet consectetur adipisicing elit.</p>
```

“<color>-text”
“text-lighten-…” 1 à 5
“text-darken-…” 1 à 4

Grid System

https://materializecss.com/grid.html
“_grid.scss”
Container: sert à centrer le contenu

```scss
.container {
  margin: 0 auto;
  max-width: 1280px;
  width: 90%;
}

@media only screen and (min-width : 601px) {
  .container {
    width: 85%;
  }
}
@media only screen and (min-width : 993px) {
  .container {
    width: 70%;
  }
}
```

s.. (1 à 12) nombre de colonnes à occuper. Exemple « s3 » occupe 3 colonnes

exemple de CSS généré (materialize base son système de grille sur float + clear fix + box sizing)

```css
.row {
  margin-left: auto;
  margin-right: auto;
  margin-bottom: 20px;
}
.row:after {
  content: "";
  display: table;
  clear: both;
}
.row .col {
  float: left;
  box-sizing: border-box;
  padding: 0 0.75rem;
  min-height: 1px;
}
.row .col[class*=push-], .row .col[class*=pull-] {
  position: relative;
}
.row .col.s1 {
  width: 8.3333333333%;
  margin-left: auto;
  left: auto;
  right: auto;
}
.row .col.s2 {
  width: 16.6666666667%;
  margin-left: auto;
  left: auto;
  right: auto;
}
```

Selon la taille de l'écran la taille est adaptée

```css
@media only screen and (min-width : 601px) {
  .row .col.m1 {
    width: 8.3333333333%;
    margin-left: auto;
    left: auto;
    right: auto;
  }
  .row .col.m2 {
    width: 16.6666666667%;
    margin-left: auto;
    left: auto;
    right: auto;
  }
```

Screen Sizes

	Mobile Devices <= 600px	Tablet Devices > 600px	Desktop Devices > 992px	Large Desktop Devices > 1200px
Class Prefix	`.s`	`.m`	`.l`	`.xl`
Container Width	90%	85%	70%	70%
Number of Columns	12	12	12	12

Adaptif

s : comportement appliqué quand écran <= 600px et pour toutes les tailles d'écran si c'est la seule classe css définie car c'est la valeur par défaut (ne dépendant pas d'une media query)
m : comportement appliqué quand écran > 600px
l : comportement appliqué quand écran > 992px
xl : comportement appliqué quand écran > 1200px

Exemple de comportement adaptif
1ere div: 2 colonnes <600, 4 colonnes > 600, 6 colonnes > 1200
2ème div: 10 colonnes <600, 8 colonnes > 600, 6 colonnes > 1200

```
<div class="row">
    <div class="col s2 m4 xl6 amber">1</div>
    <div class="col s10 m8 xl6 yellow">2</div>
</div>
```

Push et pull

Push: pousser de x colonnes
Pull: tirer vers la gauche de x colonnes

Exemple cela inverse les 2 div

```
<div class="row">
     <div class="col s6 push-s6 blue">1</div>
      <div class="col s6 pull-s6 orange">2</div>
</div>
```

Cacher

Voir dans les helpers https://materializecss.com/helpers.html
hide
hide-one-med-only
hide-on-med-and-down
etc.

Alignement

Centrer verticalement avec valign-wrapper

```
<div class="green valign-
wrapper" style="height: 300px; width: 100%;">
        <div class="red">Centré Verticalement</div>
</div>
```

Css généré

```
.valign-wrapper {
  display: flex;
  align-items: center;
}
```

Alignement d'elements inline (left-align, center-align, right-align)


```
<div class="container">
     <p class="left-align">Lorem, ipsum dolor sit amet consectetur adipisicing elit.</p>
     <p class="center-align">Lorem, ipsum dolor sit amet consectetur adipisicing elit.</p>
     <p class="right-align">Lorem, ipsum dolor sit amet consectetur adipisicing elit.</p>
</div>
```

Alignement de blocs (left, right, etc.)

Avec les classes left, right (utilise float)

```
<div class="container">
            <div class="right red">Right</div>
            <div class="left blue">Left</div>
</div>
```

CSS généré

```
.left {
  float: left !important;
}

.right {
  float: right !important;
}
```

Pour **centrer** un bloc on utilise le système de grille

```
<div class="row">
        <div class="col s4 offset-s4 purple">Centré</div>
</div>
```

Image et video responsive avec responsive-img et responsive-video

Avec la classe « responsive-img »

```
<div class="container">
          <img src="https://source.unsplash.com/random/800x600" class="responsive-img"/>
</div>
```

Video avec « responsive-video »

```
<video class="responsive-video" controls>
       <source src="./video/movie.mp4" type="video/mp4">
 </video>
```

CSS généré

```
// Images
img.responsive-img,
video.responsive-video {
  max-width: 100%;
```

```
  height: auto;
}
```

Image arrondie avec circle

Avec la classe « circle »

```
 <div class="container">
          <img src="https://source.unsplash.com/random/600x600" class="circle"/>
</div>
```

CSS généré

```
.circle {
  border-radius: 50%;
}
```

Effets

Shadows avec z-depth

Avec « z-depth-… » 1 à 5

```
<div class="col s12 m2">
    <p class="z-depth-1 card-panel">z-depth-1</p>
</div>
<div class="col s12 m2">
    <p class="z-depth-2 card-panel">z-depth-2</p>
</div>
<div class="col s12 m2">
    <p class="z-depth-3 card-panel">z-depth-3</p>
</div>
<div class="col s12 m2">
    <p class="z-depth-4 card-panel">z-depth-4</p>
</div>
<div class="col s12 m2">
    <p class="z-depth-5 card-panel">z-depth-5</p>
</div>
</div>
```

CSS généré

```
.z-depth-1 {
  box-shadow: 0 2px 2px 0 rgba(0, 0, 0, 0.14), 0 3px 1px -2px rgba(0, 0, 0, 0.12), 0 1px 5px 0 rgba(0, 0, 0, 0.2);
}

.z-depth-1-half {
  box-shadow: 0 3px 3px 0 rgba(0, 0, 0, 0.14), 0 1px 7px 0 rgba(0, 0, 0, 0.12), 0 3px 1px -1px rgba(0, 0, 0, 0.2);
}

.z-depth-2 {
  box-shadow: 0 4px 5px 0 rgba(0, 0, 0, 0.14), 0 1px 10px 0 rgba(0, 0, 0, 0.12), 0 2px 4px -1px rgba(0, 0, 0, 0.3);
}

.z-depth-3 {
  box-shadow: 0 8px 17px 2px rgba(0, 0, 0, 0.14), 0 3px 14px 2px rgba(0, 0, 0, 0.12), 0 5px 5px -
3px rgba(0, 0, 0, 0.2);
}

.z-depth-4 {
  box-shadow: 0 16px 24px 2px rgba(0, 0, 0, 0.14), 0 6px 30px 5px rgba(0, 0, 0, 0.12), 0 8px 10px -
7px rgba(0, 0, 0, 0.2);
}

.z-depth-5 {
  box-shadow: 0 24px 38px 3px rgba(0, 0, 0, 0.14), 0 9px 46px 8px rgba(0, 0, 0, 0.12), 0 11px 15px -
7px rgba(0, 0, 0, 0.2);
}
```

Effet au survol avec hoverable

Avec la classe « hoverable »

```
<div class="card-panel hoverable">Effet au survol</div>
```

CSS généré (combinaison de 2 shadows)

```
.hoverable {
  transition: box-shadow 0.25s;
}
.hoverable:hover {
  box-shadow: 0 8px 17px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19);
}
```

Syntaxe

```
/* offset-x | offset-y | color */
box-shadow: 60px -16px teal;

/* offset-x | offset-y | blur-radius | color */
box-shadow: 10px 5px 5px black;

/* offset-x | offset-y | blur-radius | spread-radius | color */
box-shadow: 2px 2px 2px 1px rgba(0, 0, 0, 0.2);

/* inset | offset-x | offset-y | color */
box-shadow: inset 5em 1em gold;

/* Une liste d'ombres, séparées par des virgules */
box-shadow: 3px 3px red, -1em 0 0.4em olive;
```

Pulse

« _pulse.scss »
L'effet se fait en boucle

```
<div class="card-panel red pulse">Pulse</div>
```

CSS généré

```
.pulse {
  overflow: visible;
  position: relative;
}
.pulse::before {
  content: "";
  display: block;
  position: absolute;
  width: 100%;
  height: 100%;
  top: 0;
  left: 0;
  background-color: inherit;
  border-radius: inherit;
  transition: opacity 0.3s, transform 0.3s;
  animation: pulse-animation 1s cubic-bezier(0.24, 0, 0.38, 1) infinite;
  z-index: -1;
}

@keyframes pulse-animation {
  0% {
    opacity: 1;
    transform: scale(1);
  }
  50% {
    opacity: 0;
    transform: scale(1.5);
  }
  100% {
    opacity: 0;
    transform: scale(1.5);
  }
}
```

Note: peut aussi s'appliquer aux buttons

```
<a class="btn-floating btn-large pulse">
                <i class="material-icons">cloud</i>
```

```
            </a>
            <div class="btn pulse">Bouton</div>
```

Appliquer un effet à une cible (js). Exemple fait apparaitre (et disparaitre) un button avec animation de scale

```
<a id="btn-1" href="#!" class="btn-floating btn-large scale-transition">
    <i class="material-icons">add</i>
</a>

<a id="btn-2" href="#!" class="btn-floating btn-large right scale-
transition scale-out">
    <i class="material-icons">add</i>
</a>

</div>
<script src="./js/materialize.min.js"></script>

<script>
M.AutoInit();
const btn1 = document.getElementById("btn-1");
const btn2 = document.getElementById("btn-2");
btn1.onclick = function (ev) {
    btn2.classList.toggle("scale-in");
}
</script>
```

Tableaux

https://materializecss.com/table.html

Un style est appliqué de base

On peut également définir sur table :

- centered pour centrer le tableau
- responsive-table
- etc.

Components

Buttons avec la classe « btn »

https://materializecss.com/buttons.html

Il suffit d'ajouter la classe « btn ». Il y a un effet au survol


```
<button class="btn">Button</button>
```

On peut appliquer une couleur. Exemple cyan

```
<button class="btn cyan">Button</button>
```

Y a un box shadow appliqué

```
.z-depth-1, .card-panel, .btn-floating, .btn, .btn-small, .btn-large {
  box-shadow: 0 2px 2px 0 rgba(0, 0, 0, 0.14), 0 3px 1px -2px rgba(0, 0, 0, 0.12), 0 1px 5px 0 rgba(0, 0, 0, 0.2);
}

.z-depth-1-half, .btn-floating:hover, .btn:hover, .btn-small:hover, .btn-large:hover {
  box-shadow: 0 3px 3px 0 rgba(0, 0, 0, 0.14), 0 1px 7px 0 rgba(0, 0, 0, 0.12), 0 3px 1px -1px rgba(0, 0, 0, 0.2);
}
```

Et le style

```
.btn, .btn-small, .btn-large,
.btn-flat {
  border: none;
  border-radius: 2px;
  display: inline-block;
  height: 36px;
  line-height: 36px;
  padding: 0 16px;
  text-transform: uppercase;
  vertical-align: middle;
  -webkit-tap-highlight-color: transparent;
}

.btn.disabled,
.btn-floating.disabled,
.btn-large.disabled,
.btn-small.disabled,
.btn-flat.disabled,
.btn:disabled,
.btn-floating:disabled,
.btn-large:disabled,
.btn-small:disabled,
.btn-flat:disabled,
.btn[disabled],
.btn-floating[disabled],
.btn-large[disabled],
.btn-small[disabled],
.btn-flat[disabled] {
  pointer-events: none;
  background-color: #DFDFDF !important;
  box-shadow: none;
  color: #9F9F9F !important;
  cursor: default;
}
.btn.disabled:hover,
.btn-floating.disabled:hover,
.btn-large.disabled:hover,
.btn-small.disabled:hover,
.btn-flat.disabled:hover,
.btn:disabled:hover,
.btn-floating:disabled:hover,
.btn-large:disabled:hover,
.btn-small:disabled:hover,
.btn-flat:disabled:hover,
.btn[disabled]:hover,
.btn-floating[disabled]:hover,
.btn-large[disabled]:hover,
.btn-small[disabled]:hover,
.btn-flat[disabled]:hover {
  background-color: #DFDFDF !important;
  color: #9F9F9F !important;
}

.btn,
.btn-floating,
.btn-large,
.btn-small,
.btn-flat {
  font-size: 14px;
```

```
  outline: 0;
}
.btn i,
.btn-floating i,
.btn-large i,
.btn-small i,
.btn-flat i {
  font-size: 1.3rem;
  line-height: inherit;
}

.btn:focus, .btn-small:focus, .btn-large:focus,
.btn-floating:focus {
  background-color: #1d7d74;
}

.btn, .btn-small, .btn-large {
  text-decoration: none;
  color: #fff;
  background-color: #26a69a;
  text-align: center;
  letter-spacing: 0.5px;
  transition: background-color 0.2s ease-out;
  cursor: pointer;
}
.btn:hover, .btn-small:hover, .btn-large:hover {
  background-color: #2bbbad;
}
```

Plusieurs types de buttons

- Raised
- Floating (rond)
- Flat
- Submit
- Small, large

Effet wave au clic

Ajouter les classes « waves-effect » et « waves-light »

```
<button class="btn cyan waves-effect waves-light">Button</button>
```

C'est l'effet disponible sur github http://fian.my.id/Waves (CSS + JS) qui peut demander quelques ajustements

Floating button avec la classe « btn-floating »

```
<a class="red btn-floating">
     <i class="material-icons">add</i>
</a>
```

On peut ajouter l'effet waves avec « waves-effect » et « waves-light » et le rendre plus gros avec « btn-large »

```
 <a class="btn-floating btn-large waves-effect waves-light red">
       <i class="material-icons">add</i>
  </a>
```

Note : pour les icones, ajouter un link vers material icons

```
<link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">
```

Style : c’est un bouton carré (width, height) avec un border-radius de 50%

```
.btn-floating {
  display: inline-block;
  color: #fff;
  position: relative;
  overflow: hidden;
  z-index: 1;
  width: 40px;
  height: 40px;
  line-height: 40px;
  padding: 0;
  background-color: #26a69a;
  border-radius: 50%;
  transition: background-color 0.3s;
  cursor: pointer;
  vertical-align: middle;
}
.btn-floating:hover {
  background-color: #26a69a;
}
.btn-floating:before {
  border-radius: 0;
}
.btn-floating.btn-large {
  width: 56px;
  height: 56px;
  padding: 0;
}
.btn-floating.btn-large.halfway-fab {
  bottom: -28px;
}
.btn-floating.btn-large i {
  line-height: 56px;
}
.btn-floating.btn-small {
  width: 32.4px;
  height: 32.4px;
}
.btn-floating.btn-small.halfway-fab {
  bottom: -16.2px;
}
.btn-floating.btn-small i {
  line-height: 32.4px;
}
.btn-floating.halfway-fab {
  position: absolute;
  right: 24px;
  bottom: -20px;
}
.btn-floating.halfway-fab.left {
  right: auto;
  left: 24px;
}
.btn-floating i {
  width: inherit;
  display: inline-block;
  text-align: center;
  color: #fff;
  font-size: 1.6rem;
  line-height: 40px;
}

button.btn-floating {
  border: none;
}
```

Icones

https://materializecss.com/icons.html

Ajout dans le head

```
<link href="https://fonts.googleapis.com/icon?family=Material+Icons"
rel="stylesheet">
```

Utilisation

```
<i class="material-icons">add</i>
```

Plusieurs tailles disponibles: tiny, small, medium, large

```
<!--
Sizes:
tiny: 1rem
small: 2rem
medium: 4rem
large: 6rem
-->
<i class="large material-icons">insert_chart</i>
```

Cards

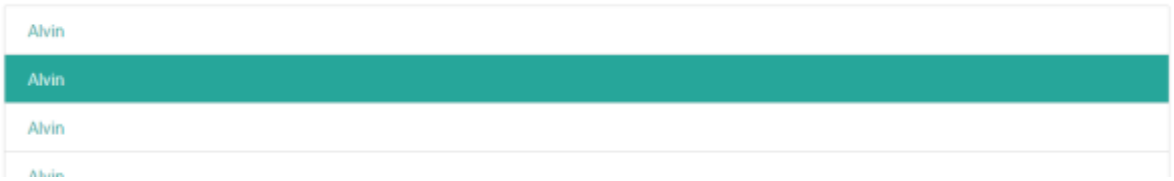
https://materializecss.com/cards.html

Collections

https://materializecss.com/collections.html

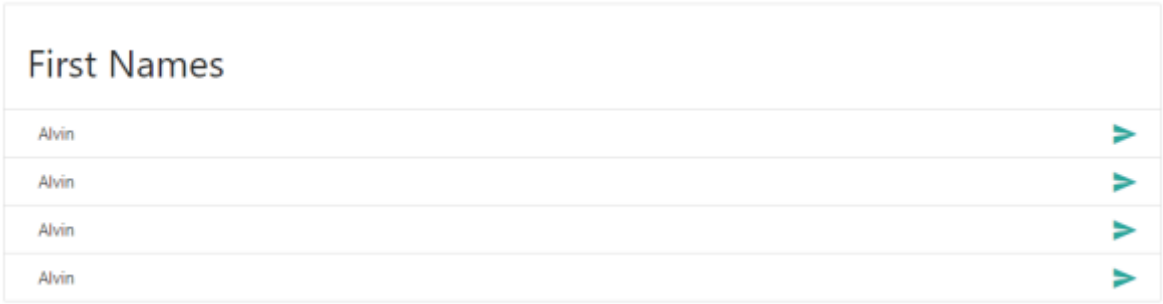
Une liste

```
<div class="collection">
                <a href="#!" class="collection-item">Alvin</a>
                <a href="#!" class="collection-item active">Alvin</a>
                <a href="#!" class="collection-item">Alvin</a>
                <a href="#!" class="collection-item">Alvin</a>
</div>
```



Avec header (titre plus gros) et avec liens à droite pour afficher une page détails par exemple

```
<ul class="collection with-header">
    <li class="collection-header"><h4>First Names</h4></li>
    <li class="collection-item"><div>Alvin<a href="#!" class="secondary-content"><i class="material-
icons">send</i></a></div></li>
    <li class="collection-item"><div>Alvin<a href="#!" class="secondary-content"><i class="material-
icons">send</i></a></div></li>
    <li class="collection-item"><div>Alvin<a href="#!" class="secondary-content"><i class="material-
icons">send</i></a></div></li>
    <li class="collection-item"><div>Alvin<a href="#!" class="secondary-content"><i class="material-
icons">send</i></a></div></li>
</ul>
```



Navbar

https://materializecss.com/navbar.html

Note si on veut que le logo reste toujours à gauche ajouter la classe « left » au logo

```
            <nav>
                <div class="nav-wrapper blue">
                    <a href="#" class="brand-logo">Logo</a>
                    <ul id="nav-mobile" class="right hide-on-med-and-down">
                        <li>
                            <a href="sass.html">Sass</a>
                        </li>
                        <li>
                            <a href="badges.html">Components</a>
                        </li>
                        <li>
                            <a href="collapsible.html">JavaScript</a>
                        </li>
                    </ul>
                </div>
            </nav>
```

Logos et liens inversés avec les classes « left » et « right »

```
<nav>
    <div class="nav-wrapper blue">
        <a href="#" class="brand-logo right">Logo</a>
        <ul id="nav-mobile" class="left hide-on-med-and-down">
            <li>
                <a href="sass.html">Sass</a>
            </li>
            <li>
                <a href="badges.html">Components</a>
            </li>
            <li>
                <a href="collapsible.html">JavaScript</a>
            </li>
        </ul>
    </div>
</nav>
```

Avec button hamburger et menu déroulant sur mobile

```
<nav>
<div class="nav-wrapper">
    <a href="#!" class="brand-logo">Logo</a>
    <a href="#" data-target="mobile-demo" class="sidenav-trigger"><i class="material-icons">menu</i></a>
    <ul class="right hide-on-med-and-down">
    <li><a href="sass.html">Sass</a></li>
    <li><a href="badges.html">Components</a></li>
    <li><a href="collapsible.html">Javascript</a></li>
    <li><a href="mobile.html">Mobile</a></li>
    </ul>
</div>
</nav>

<ul class="sidenav" id="mobile-demo">
<li><a href="sass.html">Sass</a></li>
<li><a href="badges.html">Components</a></li>
<li><a href="collapsible.html">Javascript</a></li>
```

Menu visible sur petit ecran

```
<li><a href="mobile.html">Mobile</a></li>
</ul>
```

Besoin de materialize.js + soit AutoInit

```
<script src="./js/materialize.min.js"></script>
<script>
    M.AutoInit();
</script>
```

Soit

```
<script src="./js/materialize.min.js"></script>
<script>
    document.addEventListener('DOMContentLoaded', function() {
      var elems = document.querySelectorAll('.sidenav');
      var instances = M.Sidenav.init(elems);
  });
</script>
```

Menu caché par défaut

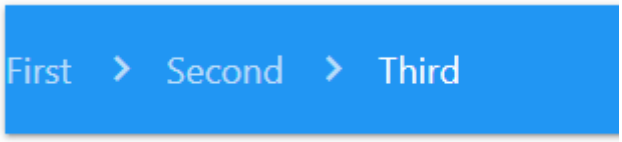
Avec button hamburger

Breadcrumbs

https://materializecss.com/breadcrumbs.html

```
<nav>
    <div class="nav-wrapper blue">
```

```
      <div class="col s12">
        <a href="#!" class="breadcrumb">First</a>
        <a href="#!" class="breadcrumb">Second</a>
        <a href="#!" class="breadcrumb">Third</a>
      </div>
    </div>
  </nav>
```



Barre de progression/ loader

https://materializecss.com/preloader.html


```
<div class="progress">
    <div class="indeterminate"></div>
</div>
```

CSS Généré

```
.progress {
  position: relative;
  height: 4px;
  display: block;
  width: 100%;
  background-color: #acece6;
  border-radius: 2px;
  margin: 0.5rem 0 1rem 0;
  overflow: hidden;
}
.progress .determinate {
  position: absolute;
  top: 0;
  left: 0;
  bottom: 0;
  background-color: #26a69a;
  transition: width 0.3s linear;
}
.progress .indeterminate {
  background-color: #26a69a;
}
.progress .indeterminate:before {
  content: "";
  position: absolute;
  background-color: inherit;
  top: 0;
  left: 0;
  bottom: 0;
  will-change: left, right;
  animation: indeterminate 2.1s cubic-bezier(0.65, 0.815, 0.735, 0.395) infinite;
}
.progress .indeterminate:after {
  content: "";
  position: absolute;
  background-color: inherit;
  top: 0;
  left: 0;
  bottom: 0;
  will-change: left, right;
  animation: indeterminate-short 2.1s cubic-bezier(0.165, 0.84, 0.44, 1) infinite;
```

```
  animation-delay: 1.15s;
}

@keyframes indeterminate {
  0% {
    left: -35%;
    right: 100%;
  }
  60% {
    left: 100%;
    right: -90%;
  }
  100% {
    left: 100%;
    right: -90%;
  }
}
@keyframes indeterminate-short {
  0% {
    left: -200%;
    right: 100%;
  }
  60% {
    left: 107%;
    right: -8%;
  }
  100% {
    left: 107%;
    right: -8%;
  }
}
```

Spinner

```
<div class="preloader-wrapper big active">
    <div class="spinner-layer spinner-blue-only">
        <div class="circle-clipper left">
            <div class="circle"></div>
        </div>
        <div class="gap-patch">
            <div class="circle"></div>
        </div>
        <div class="circle-clipper right">
            <div class="circle"></div>
        </div>
    </div>
</div>
```

Footer

https://materializecss.com/footer.html

Carousel

https://materializecss.com/carousel.html#one!
« _carousel.scss »

```
<div class="carousel">
    <a class="carousel-item" href="#one!"><img src="https://source.unsplash.com/random/250x250"></a>
    <a class="carousel-item" href="#two!"><img src="https://source.unsplash.com/random/250x250"></a>
    <a class="carousel-item" href="#three!"><img src="https://source.unsplash.com/random/250x250"></a>
    <a class="carousel-item" href="#four!"><img src="https://source.unsplash.com/random/250x250"></a>
    <a class="carousel-item" href="#five!"><img src="https://source.unsplash.com/random/250x250"></a>
</div>
```

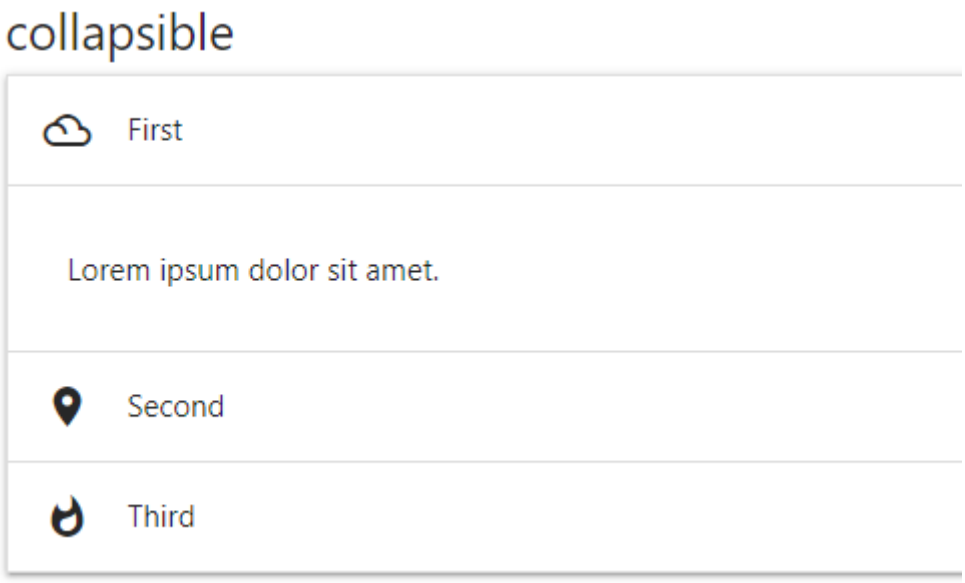
Initialisation js

```
<script src="./js/materialize.min.js"></script>
<script>
    //M.AutoInit();
    // or
    document.addEventListener('DOMContentLoaded', function () {
        var elems = document.querySelectorAll('.carousel');
        var instances = M.Carousel.init(elems);
    });
</script>
```

Collapsible/ accordion

https://materializecss.com/collapsible.html

```
<ul class="collapsible">
    <li>
        <div class="collapsible-header">
            <i class="material-icons">filter_drama</i>First</div>
        <div class="collapsible-body">
            <span>Lorem ipsum dolor sit amet.</span>
        </div>
    </li>
    <li>
        <div class="collapsible-header">
            <i class="material-icons">place</i>Second</div>
        <div class="collapsible-body">
            <span>Lorem ipsum dolor sit amet.</span>
        </div>
    </li>
    <li>
        <div class="collapsible-header">
            <i class="material-icons">whatshot</i>Third</div>
        <div class="collapsible-body">
            <span>Lorem ipsum dolor sit amet.</span>
        </div>
    </li>
</ul>
```



Et dans le javascript

```
M.AutoInit();
```

CSS Généré

```
.collapsible {
  border-top: 1px solid #ddd;
  border-right: 1px solid #ddd;
  border-left: 1px solid #ddd;
  margin: 0.5rem 0 1rem 0;
}

.collapsible-header {
  display: flex;
  cursor: pointer;
  -webkit-tap-highlight-color: transparent;
  line-height: 1.5;
  padding: 1rem;
  background-color: #fff;
  border-bottom: 1px solid #ddd;
}
.collapsible-header:focus {
  outline: 0;
}
.collapsible-header i {
  width: 2rem;
  font-size: 1.6rem;
  display: inline-block;
  text-align: center;
  margin-right: 1rem;
}

.keyboard-focused .collapsible-header:focus {
  background-color: #eee;
}

.collapsible-body {
  display: none;
  border-bottom: 1px solid #ddd;
  box-sizing: border-box;
  padding: 2rem;
}
```

+ besoin JS

Dropdown

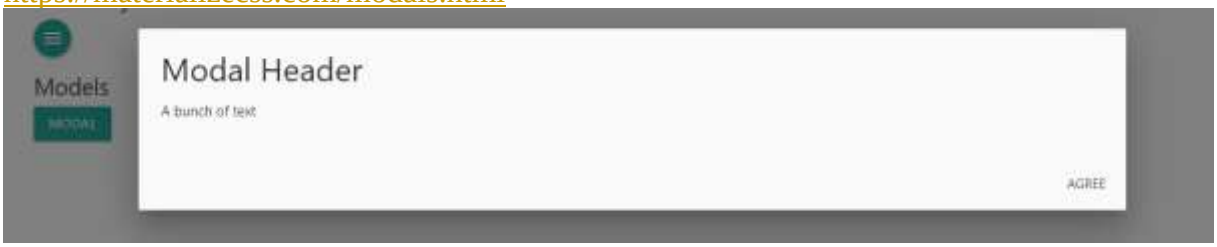
https://materializecss.com/dropdown.html

Feature discovery

https://materializecss.com/feature-discovery.html

Modals

https://materializecss.com/modals.html




```
<!-- Modal Trigger -->
<a class="waves-effect waves-light btn modal-trigger" href="#modal1">Modal</a>

<!-- Modal Structure -->
<div id="modal1" class="modal">
    <div class="modal-content">
        <h4>Modal Header</h4>
        <p>A bunch of text</p>
    </div>
    <div class="modal-footer">
        <a href="#!" class="modal-close waves-effect waves-green btn-
flat">Agree</a>
    </div>
</div>
```

En js

```
M.AutoInit();
```

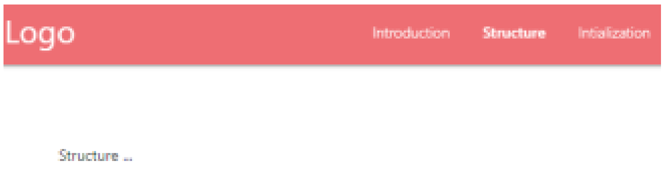
Parallax

https://materializecss.com/parallax.html

Scrolspy

https://materializecss.com/scrollspy.html

Le menu actif est mis à jour selon l'endroit de la page




```
    <div class="navbar-fixed">
            <nav>
                <div class="nav-wrapper">
                    <a href="#" class="brand-logo left">Logo</a>
                    <ul id="nav-mobile" class="right table-of-contents">
                        <li>
                            <a  class="white-text" href="#introduction">Introduction</a>
                        </li>
                        <li>
                            <a class="white-text" href="#structure">Structure</a>
                        </li>
                        <li>
                            <a class="white-text" href="#initialization">Intialization</a>
                        </li>
                    </ul>
                </div>
            </nav>
        </div>

        <div class="container">
            <div class="row">
                <div class="col s12 m9 l10">
                    <div id="introduction" class="section scrollspy">
                        <p>Introduction ....</p><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/
><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/>
                    </div>

                    <div id="structure" class="section scrollspy">
                        <p>Structure ...</p><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br
/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/>
<br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/>
                    </div>

                    <div id="initialization" class="section scrollspy">
                        <p>initialisation ....</p><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><b
r/><br/><br/><br/><br/><br/><br/><br/><br/><br/>
                    </div>
                </div>

            </div</div>

            <script src="./js/materialize.min.js"></script>
            <script>
                M.AutoInit();
            </script>
```

Toasts

https://materializecss.com/toasts.html

```
  <button id="show-toast" class="btn">Show toast</button>
```

Ajout d'un event listener js

```
const btnToast = document.getElementById("show-toast");
            btnToast.onclick = function(){
                M.toast({html: 'I am a toast!'});
}
```

Tooltip

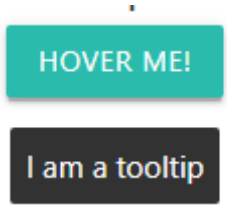
https://materializecss.com/tooltips.html

Exemple apparait au survol

```
<a class="btn tooltipped" data-position="bottom" data-tooltip="I am a tooltip">Hover me!</a>
```

Plus js

```
  M.AutoInit();
```



Waves

https://materializecss.com/waves.html#!

Forms

autocomplete

https://materializecss.com/autocomplete.html

https://materializecss.com/text-inputs.html

```
        <div class="container">
            <h5>Forms</h5>

            <form class="col s12">
                <div class="row">
                    <div class="input-field col s6">
                        <input placeholder="Placeholder" id="first_name" type="text" class="validate">
                        <label for="first_name">First Name</label>
                    </div>
                    <div class="input-field col s6">
                        <input id="last_name" type="text" class="validate">
                        <label for="last_name">Last Name</label>
                    </div>
                </div>
                <div class="row">
                    <div class="input-field col s12">
                        <input disabled value="I am not editable" id="disabled" type="text" class="validate">
                        <label for="disabled">Disabled</label>
                    </div>
                </div>
                <div class="row">
                    <div class="input-field col s12">
                        <input id="password" type="password" class="validate">
                        <label for="password">Password</label>
                    </div>
                </div>
                <div class="row">
                    <div class="input-field col s12">
                        <input id="email" type="email" class="validate">
                        <label for="email">Email</label>
```

```
                </div>
            </div>
            <div class="row">
                <div class="col s12">
                    This is an inline input field:
                    <div class="input-field inline">
                        <input id="email_inline" type="email" class="validate">
                        <label for="email_inline">Email</label>
                        <span class="helper-text" data-error="wrong" data-success="right">Helper text</span>
                    </div>
                </div>
            </div>
        </form>
    </div>
```

Tailwind CSS

A la manière de Boostrap 5, permet d'ajouter des classes CSS sur les éléments html. On peut également créer des custom classes, étendre les classes existantes. Une purge permet d'optimiser le fichier CSS généré en ne conservant que ce qui est utilisé dans les pages.

Installation

Création de package.json

```
npm init -y
```

Installation de tailwind, postcss et autoprefixer (ajout des vendors de chaque navigateur)

```
npm install -D tailwindcss@latest postcss@latest autoprefixer@latest
```

Versions actuelles:

- "tailwindcss": "^2.2.16"
- "postcss": "^8.3.9",
- "autoprefixer": "^10.3.7",

Création de "postcss.config.js"

```
module.exports = {
  plugins: {
    tailwindcss: {},
    autoprefixer: {},
  },
};
```

Création de "tailwind.config.js"

```
npx tailwindcss init
```

```
module.exports = {
  purge: [],
  darkMode: false, // or 'media' or 'class'
  theme: {
    extend: {},
  },
  variants: {
    extend: {},
```

```
  },
  plugins: [],
}
```

Variante avec **tous les styles**

npx tailwindcss init --full

```
const colors = require('tailwindcss/colors')

module.exports = {
  purge: [],
  presets: [],
  darkMode: false, // or 'media' or 'class'
  theme: {
    screens: {
      sm: '640px',
      md: '768px',
      lg: '1024px',
      xl: '1280px',
      '2xl': '1536px',
    },
    colors: {
      transparent: 'transparent',
      current: 'currentColor',

      black: colors.black,
      white: colors.white,
      gray: colors.coolGray,
      red: colors.red,
      yellow: colors.amber,
      green: colors.emerald,
      blue: colors.blue,
      indigo: colors.indigo,
      purple: colors.violet,
      pink: colors.pink,
    },
// etc.
```

Création de la feuille de style de “base”

```
@tailwind base;
@tailwind components;
@tailwind utilities;
```

Astuce : retirer le surlignement de lint dans VS Code

CTRL+MAJ+ P ... Open settings.json …et ajouter
"css.lint.unknownAtRules": "ignore",

Scripts

```
"tailwind": "tailwindcss --postcss -i ./src/input.css -o ./src/style.css",
"tailwind:watch": "tailwindcss --postcss -i ./src/input.css -
o ./src/style.css --watch",
```

Dans la page html

```
<link rel="stylesheet" href="style.css">
```

Purge

Seuls les style sutilisés seront inclus dans la feuille de style générée

```
module.exports = {
  purge: ["./src/**/*.html"],
  darkMode: false, // or 'media' or 'class'
  theme: {
    extend: {},
  },
  variants: {
    extend: {},
  },
  plugins: [],
};
```

Installation de cross-env

npm i cross-env -D

Script en mode production

```
"tailwind:prod": "cross-env NODE_ENV=production tailwindcss --postcss -
i ./src/input.css -o ./src/style.css"
```

React

Un seul fichier de styles global (glabal.css ou input.css par exemple). Ne permet pas d'avoir de fichiers css

"tailwind.config.js"

```
module.exports = {
  purge: ["./src/**/*.{js,jsx,ts,tsx}", "./public/index.html"],
  darkMode: false, // or 'media' or 'class'
  theme: {
    extend: {}
  },
  variants: {
    extend: {}
  },
  plugins: []
};
```

Scripts

```
  "scripts": {
    "prebuild": "rimraf dist && npm run tailwind:prod",
    "build": "webpack --mode=production --progress",
    "start": "webpack serve --mode=development --open --hot --history-api-fallback --port 3000",
    "tailwind": "tailwindcss --postcss -i ./src/input.css -o ./src/index.css --watch",
    "tailwind:prod": "cross-env NODE_ENV=production tailwindcss --postcss -i ./src/input.css -
o ./src/index.css"
  },
```

Dans index.js on importe la feuille de style générée

```
import "./index.css";
```

Avec create react app

https://tailwindcss.com/docs/guides/create-react-app

npm install -D tailwindcss@npm:@tailwindcss/postcss7-compat postcss@^7 autoprefixer@^9
npm install @craco/craco

Replace scripts

```
  "scripts": {
    "start": "craco start",
    "build": "craco build",
    "test": "craco test",
    "eject": "react-scripts eject"
  },
```

Create `craco.config.js`

```
module.exports = {
  style: {
    postcss: {
      plugins: [
        require('tailwindcss'),
        require('autoprefixer'),
      ],
    },
  },
}
```

Create `tailwind.config.js`

npx tailwindcss-cli@latest init

And update the "purge"

```
module.exports = {
  purge: ["./src/**/*.{js,jsx,ts,tsx}", "./public/index.html"],
```

```
  darkMode: false, // or 'media' or 'class'
  theme: {
    extend: {},
  },
  variants: {
    extend: {},
  },
  plugins: [],
};
```

Update `index.css`

```
/* ./src/index.css */
@tailwind base;
@tailwind components;
@tailwind utilities;
```

On peut définir des classes dans index.css et dans les feuilles de styles des components

Avec Next.js

https://tailwindcss.com/docs/guides/nextjs

Création d'un projet à partir d'un exemple, exemple avec tailwind
https://github.com/vercel/next.js/tree/master/examples

npx create-next-app -e with-tailwindcss next-tailwind-sample

Personnalisation

Extend Theme https://tailwindcss.com/docs/theme
Variants https://tailwindcss.com/docs/configuring-variants

Création de custom classe

Dans "input.css"
Exemple avec l'élément input. Au lieu de mettre inline toutes les classes de tailwind on crée une custom classe réutilisable

```
@tailwind base;
@tailwind components;
@tailwind utilities;

input {
  @apply border shadow-inner px-3 py-2 text-gray-700 w-full;
}

input:focus {
  @apply outline-none ring border-blue-300;
}
```

Autre exemple création d'une classe CSS permettant de créer un carré rouge

```
.my-square {
  @apply h-40 w-40 bg-red-500;
}
```

Usage

```
<div class="my-square"></div>
```