



Cursus Front CSS AVANCÉES

M2I Formation 2022

Glodie Tshimini



PLAN

Glodie Tshimini : contact@tshimini.fr

PLAN

- I. Sélecteurs CSS
- II. Unités
- III. Mise en page
- IV. Responsive Design
- V. Transformations CSS
- VI. Transitions et animations
- VII. Préprocesseur SASS
- VIII. Optimisations et bibliothèques CSS

EXERCICE

Glodie Tshimini : contact@tshimini.fr



Exercise 1

- ▶ 0-exercices/exercice1.md

RAPPELS

Glodie Tshimini : contact@tshimini.fr



Sélecteurs CSS

EXERCICE

Glodie Tshimini : contact@tshimini.fr



Exercise 2

- ▶ `0-exercices/exercice2.md`



Déclaration

```
Sélecteur {  
    propriété : valeur;  
}
```

Le **sélecteur** est l'élément ciblé

Quelques sélecteurs **CSS**

Sélecteurs	Exemple(s)
Sélecteur d'élément ou balise HTML	<code>p { color : red; }</code> <code>h1 { text-align : center; }</code>
Sélecteur de class	<code>.underline { text-transform : underline; }</code> <code>.bold { font-weight : bold; }</code>
Sélecteur d'un identifiant (ID)	<code>#id { border : 1 px solid black; }</code> <code>#product-5 { text-align : justify; }</code>
Sélecteurs universels	<code>* { margin : 10px auto ; }</code>

Combinateurs

Sélecteur	Exemple(s)
Sélection des enfants directs	<code>div > p { background-color : orange ; }</code>
Sélecteur des enfants à n'importe quel niveau	<code>div p { display : none ; }</code> <code>div span { color : green ; }</code>
Voisin direct	<code>div + a { cursor : pointer ; }</code>

- ▶ [Documentation W3C sur les sélecteurs](#)
- ▶ [Test W3C sur les sélecteurs](#)

Quelques propriétés

	Propriété	Exemples
Couleurs	color background-color	color : red rgb(255, 99, 71) #F10606; background-color : gray #808080;
Police	font-family	font-family: Impact, Verdana, "Arial Black";
Taille texte	font-size	font-size : 15px small 1,5em;
Styles de police	font-style font-weight ...	font-style : italic; font-weight : bold; text-decoration : underline;
Alignement	text-align	text-align: center right left justify ;
Marges	margin padding	margin : 5px ; margin-top : 10px ; margin-bottom : 30px; padding : 10px ; padding-left : 15px ;

Quelques propriétés

	Propriété	Exemple(s)
Bordure	border	border : 1px solid black ;
Bordures arrondies	border-radius	border-radius : 10px ;
Soulignements et autres décorations	text-decoration	text-decoration : underline line-through overline none ;
Ombres block	box-shadow	box-shadow : 6px 6px 0px black ;
Hauteur, largeur	height, width	height : 400px; width : 300px ;

[Source image christian Lisangola](#)

Pseudo-classe

Ajouter un style spécifique en fonction de l'état dans lequel se trouve l'élément ciblé ou en tenant compte des autres éléments présents dans le document HTML

Pseudo-classe	
:hover	Etat de survol de l'élément
:active	Etat lors du clique sur l'élément
:visited	Etat après avoir cliqué sur l'élément
:checked	Etant lorsque l'élément ciblé est coché
:only-child	Enfant unique d'un élément parent

Pseudo-elements

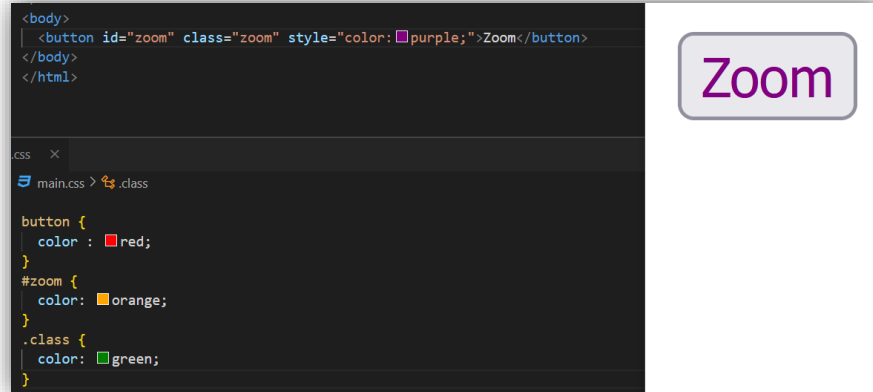
Styliser une partie de l'élément sélectionné

Pseudo-element	
::first-line	Styliser la première lettre de l'élément ciblé.
::before	Ajouter du contenu avant l'élément cible à l'aide la propriété content et de sa valeur (première enfant de la cible)
::after	Ajouter du contenu après l'élément cible à l'aide la propriété content et de sa valeur (dernière enfant de la cible)

Propriété	Valeur	
content	«Contenu entre guillemets »	Associé aux pseudo-elements ::before et ::after dont la valeur est généré par le CSS et affiché par le navigateur

Spécificités

- ▶ Lorsque plusieurs règles (propriétés et valeurs) sont appliquées à un élément avec des sélecteurs différents qui ciblent ce même élément, le navigateur applique en premier les règles les plus spécifiques.
- ▶ Propriétés depuis le style *inline* > Propriétés depuis l'*ID* > propriétés depuis la *Class*, *pseudo-class* et *attributs* > propriétés depuis le *tag HTML* (élément) et les *pseudoéléments*

The image shows a code editor with two panels. The top panel displays HTML code:

```
<body>
| <button id="zoom" class="zoom" style="color: purple;">Zoom</button>
| </body>
| </html>
```

 The bottom panel displays CSS code:

```
main.css > .class
button {
  color: red;
}
#zoom {
  color: orange;
}
.class {
  color: green;
}
```

 To the right of the code editor is a visual preview of a button with a purple border and the text 'Zoom' in purple font.



Unités

Numériques et **absolues**

Unité num. et absolue	
cm	Réservé aux feuilles de style dédiée à l'impression
mm	Réservé aux feuilles de style dédiée à l'impression
in	Réservé aux feuilles de style dédiée à l'impression
px	Unité de référence pour les écrans, un écran est défini par sa résolution en pixels

Numériques et relatives

Unités num. et relatives	
%	Dimension est proportionnelle à la dimension du conteneur (parent) à utiliser principalement pour définir une largeur (width)
em	Relative par rapport à la taille de la police de l'élément ou de l'élément parent
rem	Relative par rapport à la taille de la police de l'élément racine html ou body
vh	Vh pour viewport height (hauteur visible) 1 vh = 1 % de la hauteur visible
vw	Vw pour viewport width (largeur visible) 15 vw = 15 % de la largeur visible

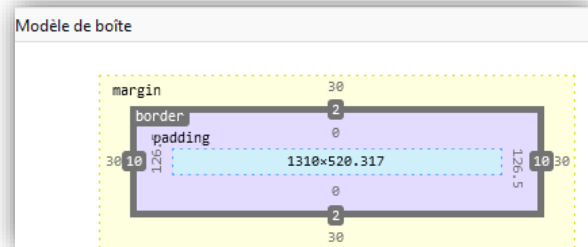
Unités pour les couleurs	Signification
Nom de la couleur en anglais	Valeur en langage naturel, rapidement limité par rapport à toutes les possibilités (~16 millions)
rgb(0,255,10)	Triplet des nombres compris entre 0 et 255 (1 octet) qui représente les couleurs rouge, vert et bleu
#74dd8a	Notation en base 16 permettant d'économiser le nombre d'octets utilisés pour coder une couleur
#dab	Notation raccourcie de l'hexadécimal en 3 octets lorsque les lettres sont identiques #dab = #ddaabb
hsl(10%, 20%, 30%)	Triplet des nombres entre 0 et 360, exprimés en % pour déterminer la teinte, la luminosité et la saturation
rgba(15, 20, 35, .9)	Identique au rgb sauf qu'on tient compte de la transparence dont la valeur est comprise entre 0 et 1



Mise en page

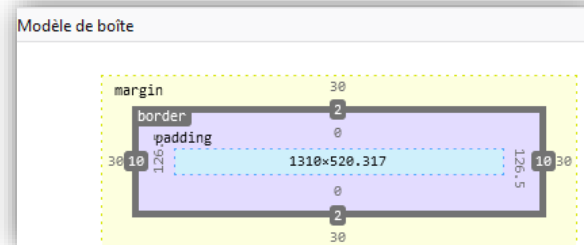
BOX MODEL

- ▶ Block
 - ▶ Occupe toute la largeur disponible
 - ▶ Élément suivant se positionne en bas (retour à la ligne)
 - ▶ Propriétés *width* & *height* modifiables
- ▶ Inline
 - ▶ Occupe la place nécessaire qu'il lui faut
 - ▶ Élément suivant se positionne à la suite
 - ▶ Propriétés *width* et *height* non modifiables
- ▶ Inline-block
 - ▶ Se comporte comme un *inline*
 - ▶ Propriétés *width* & *height* modifiables



BOX SIZING

- ▶ Définit la façon dont la hauteur et la largeur totale d'un élément sont calculées à l'aide de la propriété *box-sizing*
- ▶ Valeur par défaut *content-box*
 - ▶ A la largeur (*width*) et à la hauteur (*height*) de base de l'élément, viennent s'ajouter les marges intérieures (*padding*), les marges extérieures (*margin*) et la bordure (*border*)
- ▶ Valeur *border-box*
 - ▶ Prend en compte le *padding*, *margin* et *border* dans le calcul du *width* et *height*
 - ▶ Valeurs du *width* et *height* incluent le *margin*, *padding* et *border*



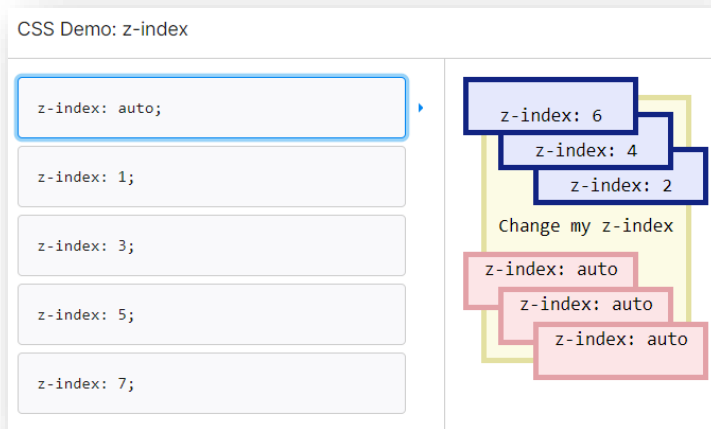


Positionnement

- Le positionnement permet de **gérer la mise en page** des éléments à l'aide de la propriété CSS **position**.
- La propriété *position* peut prendre 5 valeurs :
 - **(static)**: Positionnement *statique* des éléments (valeur par défaut)
 - **relative** : Positionnement ajusté par rapport à la position initiale de l'élément
 - **absolute** : Positionnement ajusté par rapport au parent le plus proche (on peut placer les éléments les uns sur les autres)
 - **fixed** : Semblable à *absolute*, cependant l'élément est toujours visible sur la fenêtre en scrollant (navigation sur la page)
 - **Sticky** : Positionnement *relative* puis se transforme en *fixed* en scrollant

Positionnement

- ▶ La propriété *position* est complétée par les propriétés **top**, **left**, **bottom** et **right** pour positionner l'élément à l'endroit souhaité. Les valeurs sont numériques avec une unité absolue ou relative. Ils peuvent être négatives .
- ▶ La propriété **z-index** permet de gérer l'affichage des éléments lorsqu'ils se chevauchent.



EXERCICE

Glodie Tshimini : contact@tshimini.fr

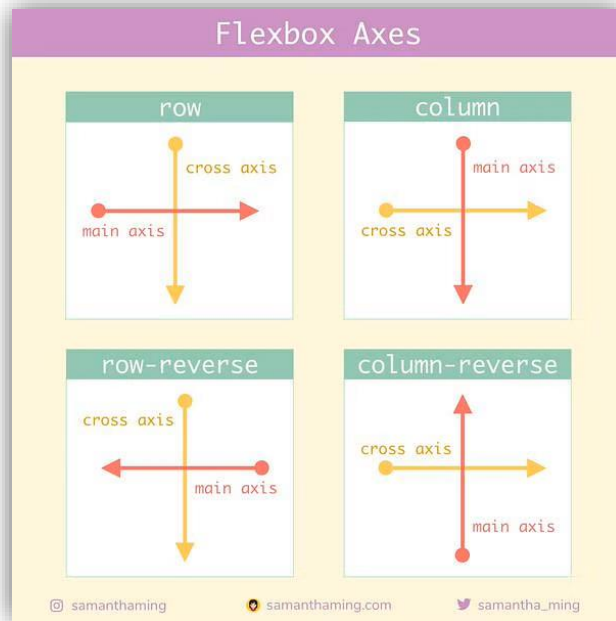


Exercise 3

- ▶ 0-exercices/exercice3.md

Synthèse Flexbox Layout

- ▶ Boîtes flexibles
- ▶ Un **container** (l'élément parent)
 - ▶ Rendu flexible grâce à la propriété **display : flex**
 - ▶ Les éléments vont se positionner de manière flexible à l'intérieur selon une direction (1 dimension)
- ▶ 2 axes
 - ▶ Axe principal ou main axis
 - ▶ Axe secondaire ou cross axis ou axe transversal



Flexbox Cheat Sheet

@simonpaix

Parent properties

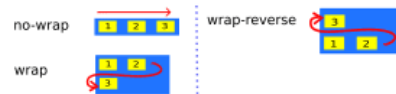
display: enables flex context for all direct children.

```
.container{
  display: flex; // or inline-flex
}
```

flex-direction: sets the main-axis.



flex-wrap: allows the items to wrap as needed.



Children properties

order: changes the order of flex items.

```
.item {
  order: 3 // the default is 0
}
```



flex-grow: allows item to grow using remaining space.

```
.item-1 { flex-grow: 0 } .item-1 { flex-grow: 1 }
//default
```

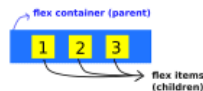
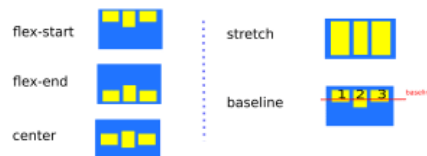


Tip: If all items have flex-grow: 1, the remaining space is distributed equally.

justify-content: defines alignment along the main axis.

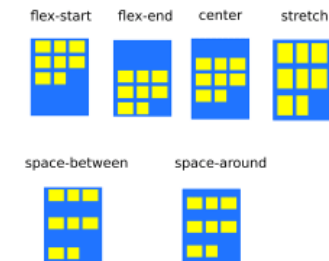


align-items: defines alignment along the cross axis.



LearnPine

align-content: aligns multiple lines, like justify-content does with individual items.



flex-shrink: defines the ability for a flex item to shrink.

```
.one { flex-shrink: 1; }
.two { flex-shrink: 2; }
.three { flex-shrink: 3; }
.four { flex-shrink: 4; }
```

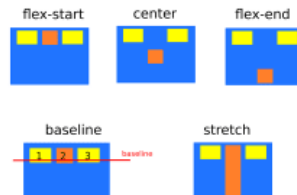


Tip: Defaults to 1. The highest the value the more it shrinks compared to siblings.

flex-basis: sets the default size of a flex item. It accepts:

- specific values : pixels, rm, %
- auto : defaults to width or height property
- content : automatic sizing, based on its content
- global values : inherit, initial, unset

align-self: overrides default alignment (or the one specified by align-items) for a specific item.



EXERCICE

Glodie Tshimini : contact@tshimini.fr

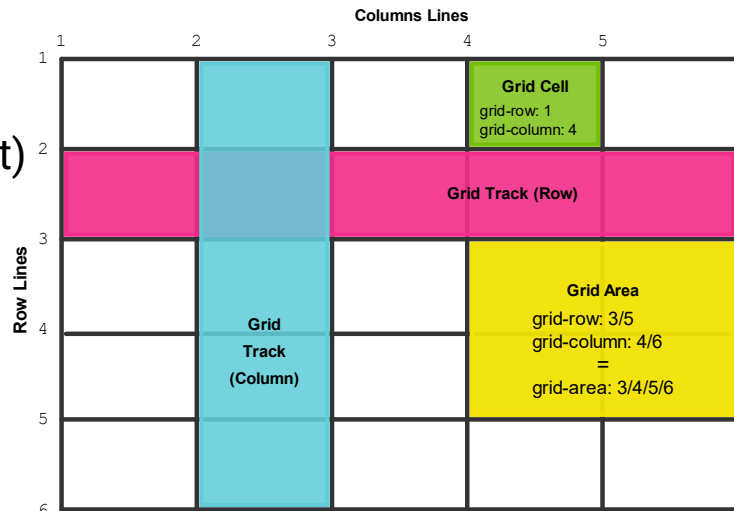


Exercise 4

- ▶ 0-exercices/exercice4.md

Synthèse CSS Grid Layout

- ▶ Grille
- ▶ Un **container** (l'élément parent)
 - ▶ Transformer en grille grâce à la propriété ***display : grid***
 - ▶ Les éléments vont se positionner selon 2 dimensions (colonne et ligne).
- ▶ Unité ***fr unit*** (fraction unit) correspond à une fraction de l'espace disponible dans le conteneur de la grille
([Source Developer Mozilla](#))



[Source image Kustavo](#)

Grid Cheat Sheet



@simonpaix

Parent properties

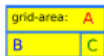
display: enables grid context for all direct children.
`.container{ display: grid; } // or inline-grid`

grid-template: defines the rows and columns of the grid. Set track size values and line-names(optional).

`grid-template-columns: 10px 30px auto 20%;`
`grid-template-rows: repeat(3, 20px);`

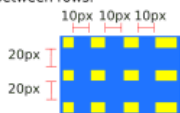
grid-template-areas:

`"A A A A"`
`B B B C"`



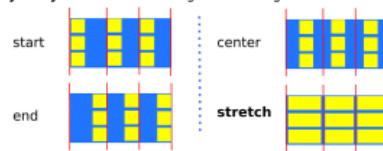
grid-gap: sets the size of the grid lines, the gutters between columns and between rows.

`column-gap: 10px;`
`row-gap: 20px;`

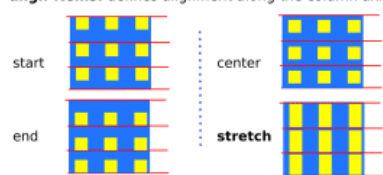


tip: default values are in bold

justify-items: defines alignment along the row axis.



align-items: defines alignment along the column axis.



grid-auto-flow: defines how to automatically place grid items that aren't explicitly placed.



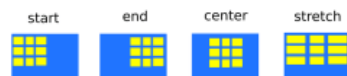
grid container (parent)



grid items (children)



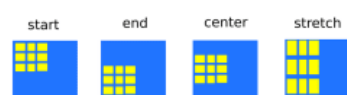
justify-content: justifies all grid content on row axis if total grid size is smaller than container.



space-between **space-around** **space-evenly**



align-content: justifies all grid content on column axis if total grid size is smaller than container.

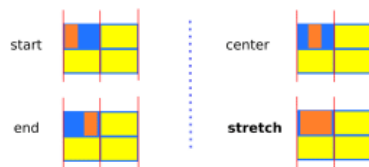


space-between **space-around** **space-evenly**

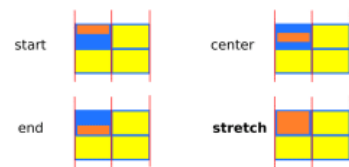


Children properties

justify-self: aligns an item inside a single cell along the row axis.



align-self: aligns an item inside a single cell along the column axis.



grid-column: determines the item's location based on a start and an end column lines (or a span).

`grid-column-start: 2;`

`grid-column-end: 4;`

`// or grid-column: 2 / 4`



grid-row: same but for the row location.

`grid-row-start: 1;`

`grid-row-end: 3;`

`// or grid-row: 1 / span 2`



EXERCICE

Glodie Tshimini : contact@tshimini.fr



Exercices 5

- ▶ 0-exercices/exercice5.md



IV.

Responsive Design



Responsive web design

- ▶ La responsive web design est une méthode qui permet de créer une page Web qui s'adapte automatiquement par rapport à la taille du support d'affichage.
- ▶ Cette méthode est possible grâce à la mise en place des **medias queries** dans le style CSS.
- ▶ 1 seul code HTML et des règles CSS différentes selon les tailles d'écran

Mise en place des **media queries**

```
@media screen and (min-width: 480px) {  
    /* Votre code CSS */  
}  
  
@media screen and (min-width: 768px) {  
    /* Votre code CSS */  
}  
  
@media screen and (min-width: 992px) {  
    /* Votre code CSS */  
}  
  
@media screen and (min-width: 1200px) {  
    /* Votre code CSS */  
}
```

[Source image christian Lisangola](#)

Autres opérateurs et caractéristiques

Op./caract. cf. documentation	
all	Caractéristique pour désigner tout type de media (screen, print, speech)
only	Caractéristique pour indiquer les règles à appliquer uniquement pour un type de media spécifique
min-width width max-width	Largeur minimal (au moins) ou fixe ou maximal (jusqu'à)
orientation	Orientation du périphérique, portrait ou landscape (paysage)
and	Opérateur logique permettant de combiner les caractéristiques
not	Opérateur logique permettant de spécifier ce qu'on ne veut pas (l'opposé)

Où placer son code ?

- ▶ Directement dans le fichier CSS principal ou spécifique de son projet idéalement à la fin, pour que les dernières propriétés écrasent les précédentes (cascade) qui ont le même poids en termes de spécificité du sélecteur
- ▶ Dans un fichier CSS à part qui sera ajouté dans le *header* avec un attribut

```
4 <head>
5   <meta charset="UTF-8">
6   <meta http-equiv="X-UA-Compatible" content="IE=edge">
7   <meta name="viewport" content="width=device-width, initial-scale=1.0">
8   <link rel="stylesheet" href="main.css">
9   <link rel="stylesheet" href="small-screen.css" media="screen and (max-width: 600px)">
10  <title>Document</title>
11 </head>
```



Tailles des écrans

Les tailles des écrans peuvent varier en fonction des librairies CSS utilisées plus tard. Ici, les tailles sont celles définies par le site

W3schools

Tailles	
Maximum 600px de large	Smartphone, montre
Minimum 600px de large	Orientation portrait tablette et phablette (entre smartphone et tablette)
Minimum 768px de large	Orientation paysage tablette
Minimum 992px de large	Ordinateurs portable et de bureau
Minimum 1200px de large	Écrans de télévision, ordinateurs portables, ordinateurs de bureau très large



Avantages

- ▶ Développement rapide
- ▶ Cout faible
- ▶ Bon pour le référencement naturel (SEO) des moteurs de recherche
- ▶ Transition automatique et fluide de l'interface utilisateur en fonction de la taille de l'écran
- ▶ Couverture de toutes les tailles d'écran très facile et aisé à l'aide des media queries

EXERCICE

Glodie Tshimini : contact@tshimini.fr



Exercise 6

- ▶ 0-exercices/exercice6.md



VII.

Préprocesseur Sass



Qu'est-ce que le **SASS** ?

- ▶ *Syntactically Awesome Style Sheets*
- ▶ Un préprocesseur ou superset du langage CSS
 - ▶ Ajout des nouvelles fonctionnalités qui viennent compléter les généralement les lacunes du langage
- ▶ Lacunes
 - ▶ Langage verbeux (beaucoup de code, difficile à maintenir)
 - ▶ Organisationnelle
- ▶ Les nouvelles fonctionnalités
 - ▶ Variables
 - ▶ Conditions
 - ▶ Imbrications
 - ▶ Boucles
 - ▶ Plus facile à maintenir pour les développeurs



Installation

- ▶ Depuis les [fichiers](#) mis à disposition par le site officiel selon son OS
- ▶ Depuis le gestionnaire de paquet de Node NPM
 - ▶ Installez d'abord [NODE JS](#)
 - ▶ A l'aide du gestionnaire de paquet npm
 - ▶ **npm install sass**
- ▶ Depuis le gestionnaire de paquet de Windows chocolatey
 - ▶ **choco install sass**
- ▶ Depuis le gestionnaire de paquet de Mac Os ou Linux
 - ▶ **brew install sass/sass/sass**



Comment travailler avec Sass ?

- ▶ Créez des fichiers qui ont comme extension .scss
- ▶ Compilez votre code .scss en css
 - ▶ `./dart_css/sass.dat src/main.scss dest/main.css`
- ▶ Après la compilation, notre fichier CSS est créé et un fichier `.map.css` également. Ce dernier fait la correspondance entre le fichier d'origine .scss et le fichier généré .css
 - ▶ `.map.css` est utile uniquement durant la phase de développement.
- ▶ Ajoutez vos fichiers .css compilé dans votre fichier HTML comme d'habitude



Les options de la ligne de commande

- ▶ Compilez à chaque fois qu'une modification est faite
 - ▶ `./dart_css/sass.dat src/main.scss dest/main.css --watch`
- ▶ Compilez avec compression du code pour la production
 - ▶ `./dart_css/sass.dat main.scss main.css --style=compressed`
- ▶ Voir toutes les options
 - ▶ `./dart_css/sass.dat --help`

EXERCICE

Glodie Tshimini : contact@tshimini.fr



Exercice9

- ▶ 0-exercices/exercice9.md



Les commentaires

- ▶ 2 possibilités
 - ▶ `/* commentaire compilé et visible au niveau du fichier finale CSS*/`
 - ▶ `// commentaire non visible lors de la compilation`



Variables

- ▶ Les variables sont désormais natives sur le CSS
 - ▶ `:root { --info-bg-color : bluelight; }`
 - ▶ `p { background-color : var(--info-bg-color); }`
- ▶ Les variables en SCSS
 - ▶ Offrent la possibilité d'effectuer des opérations
 - ▶ Mathématiques (+, -, *, etc.) sur les valeurs numériques
 - ▶ Interpolation (remplacement)
Pour utiliser la valeur d'une variable en tant que sélecteur ou tout simplement récupérer cette valeur pour un autre cas d'usage, il suffit utiliser la syntaxe `#{$var}`

Listes

- ▶ Suite des variables séparées par une virgule
 - ▶ Elles peuvent être parcourues par les boucles que l'on verra plus tard dans ce cours
 - ▶ Elles sont principalement utilisées pour générer des sélecteurs ou propriétés de manière dynamique

```
$cities : paris, nantes, lyon, marseille;
```

```
$seasons :  
  summer  green,  
  spring  orange,  
  autumn  brown,  
  winter  white;
```



Debugger en SCSS

- ▶ Pour debugger ses variables, conditions, fonctions que l'on verra plus tard, il suffit d'utiliser `@debug $myVar`
- ▶ La valeur sera affichée dans la console lors de la compilation du code en CSS

```
33 $seasons :
34   summer green,
35   spring orange,
36   autumn brown,
37   winter white;
38
39 @debug $seasons
```

TERMINAL PROBLEMS 8 OUTPUT COMMENTS

▼ TERMINAL

```
Glodie@Glodie MINGW64 /e/formations/coderbase/m2i/poe-front-28112022/5-css (main)
$ ./dart-sass/sass.bat demo/test.scss demo/test.css
demo\test.scss:39 Debug: summer green, spring orange, autumn brown, winter white
```




Maps

- ▶ Similaire aux listes
- ▶ Ensemble de clés valeurs avec une notation plus stricte
- ▶ Possible de récupérer les clés ou les valeurs à l'aide la fonction ***map_get(\$maps, key, key, ...)***

```
$breakpoints: (  
  xs: (  
    width: 600px  
  ),  
  sm: (  
    width: 768px  
  ),  
  md: (  
    width: 992px  
  ),  
  lg : (  
    width: 1200px  
  )  
);  
  
@debug map-get($breakpoints, xs);  
@debug map-get($breakpoints, xs, width);
```

EXERCICE

Glodie Tshimini : contact@tshimini.fr



Exercice10

- ▶ 0-exercices/exercice10.md



Imbrication des sélecteurs CSS

- ▶ Aussi appelé Nesting en anglais
- ▶ Permet à partir d'un sélecteur parent, d'imbriquer plusieurs sélecteurs enfants en suivant la hiérarchie dans le document HTML
- ▶ Pour garder une bonne lisibilité, ne pas dépasser 3 niveaux d'imbrications

Exemples d'imbrication

```
table {  
  color: ■ #cdf;  
  thead {  
    font-weight: 700;  
  }  
  tbody {  
    font-weight: 500;  
    td {  
      text-align: center;  
    }  
  }  
}  
  
a {  
  text-decoration: none;  
  &:hover {  
    text-decoration: solid;  
    color: ■ #800080;  
  }  
}
```

```
1 table {  
2   color: ■ #cdf;  
3 }  
4 table thead {  
5   font-weight: 700;  
6 }  
7 table tbody {  
8   font-weight: 500;  
9 }  
10 table tbody td {  
11   text-align: center;  
12 }  
13  
14 a {  
15   text-decoration: none;  
16 }  
17 a:hover {  
18   text-decoration: solid;  
19   color: ■ #800080;  
20 }
```

EXERCICE

Glodie Tshimini : contact@tshimini.fr



Exercice 11

- ▶ 0-exercices/exercice11.md



Mixins

- ▶ Permet de regrouper plusieurs propriétés (bloc de code CSS) au sein d'une structure appelée *mixin*
 - ▶ Evite la répétition du code notamment dans le cas des propriétés qui nécessitent des préfixes des navigateurs pour des questions de compatibilités
 - ▶ Internet Explorer, Edge : *-ms-*
 - ▶ Google, Safari et nouvelles versions d'Opera : *-webkit-*
 - ▶ Firefox : *-moz-*
 - ▶ Anciennes versions d'Opera : *-o-*



Comment créer des Mixins ?

- ▶ Créez un seul fichier unique nommé ***mixins.scss***
- ▶ Ecrivez vos mixins à l'intérieur de ce fichier, on utilise ***@mixin*** pour déclarer une mixin
 - ▶ ***@mixin myMixin {...}***
 - ▶ ***@mixin myMixinWithParams (\$a)***
 - ▶ ***@mixin myMixinDefaultValue(\$a : 14px)***
- ▶ Importez le fichier ***mixins.scss*** dans votre fichier principal ***.scss***
 - ▶ ***@import 'libs/mixins.scss'***
- ▶ Utilisez la mixin dans un sélecteur à l'aide du mot-clé ***@include***
 - ▶

```
p {  
    @include myMixin;  
    @include myMixinWithParams(#ddd);  
    @include myMixinDefaultValue(20px);  
}
```

Exemple mixin

```

.container {
  @include example(■#800080);
}

1 @mixin example($color: ■red) {
2   display: grid;
3   transition: all .5s;
4   user-select: none;
5   background: linear-gradient(to bottom, ■white, $color);
6   display: -ms-grid;
7   display: grid;
8   -webkit-transition: all .5s;
9   -o-transition: all .5s;
10  transition: all .5s;
11  -webkit-user-select: none;
12  -moz-user-select: none;
13  -ms-user-select: none;
14  user-select: none;
15  background: -webkit-gradient(linear, left top, left botto
16  background: -o-linear-gradient(top, ■white, $color);
17  background: linear-gradient(to bottom, ■white, $color);
18  }

32 .container {
33   display: grid;
34   transition: all 0.5s;
35   user-select: none;
36   background: linear-gradient(
37     to bottom, ■white, ■#800080
38   );
39   display: -ms-grid;
40   display: grid;
41   -webkit-transition: all 0.5s;
42   -o-transition: all 0.5s;
43   transition: all 0.5s;
44   -webkit-user-select: none;
45   -moz-user-select: none;
46   -ms-user-select: none;
47   user-select: none;
48   background: -webkit-gradient(linear, left top, left botto
49   background: -o-linear-gradient(top, ■white, $color);
50   background: linear-gradient(to bottom, ■white, $color);
51 }

```

EXERCICE

Glodie Tshimini : contact@tshimini.fr



Exercice 12

- ▶ 0-exercices/exercice12.md



Fonctions de base

- ▶ *darken(\$red, 20)* : assombri la couleur définie par la variable \$red de 20%
- ▶ *lighten(\$green, 30)* : éclairci la couleur définie par la variable \$green de 30%
- ▶ *rgba(\$hexa, .5)* : converti la couleur hexadécimale en rgba avec une opacité de .5
- ▶ [Liste des fonctions natives](#)



Comment créer ses propres fonctions ?

Comme en algorithmie ou depuis un langage de programmation

1. Nommer la fonction (signature)
 - Peut prendre 0, 1 ou plusieurs arguments (paramètres)
2. Effectuez vos traitements
3. Renvoyez le résultat final

```
@function toEm($pixel, $base : 16) {  
  @return calc(1em * $pixel / $base);  
}
```

```
48 p {  
49   font-size: toEm(64);  
50 }
```

```
50  
51 p {  
52   font-size: 4em;  
53 }
```



Conditions

- ▶ Fonctionnement habituel de l'algorithmie ou de n'importe quel langage de programmation
- ▶ Applique certaines règles en fonction d'une condition
- ▶ **@if (condition) {**
} @else {
}
- ▶ Les opérateurs
 - ▶ De comparaison : ==, != , <, >=, <=, >
 - ▶ Logique : and, or et not

Exemple condition

```
$light-background: #f2ece4;
$light-text: #036;
$dark-background: #6b717f;
$dark-text: #d2e1dd;

@mixin theme-colors($light-theme: true) {
  @if $light-theme {
    background-color: $light-background;
    color: $light-text;
  } @else {
    background-color: $dark-background;
    color: $dark-text;
  }
}

.banner {
  @include theme-colors($light-theme: true);
  body.dark & {
    @include theme-colors($light-theme: false);
  }
}
```

```
54
55 .banner {
56   background-color: #f2ece4;
57   color: #036;
58 }
59 body.dark .banner {
60   background-color: #6b717f;
61   color: #d2e1dd;
62 }
63
64 /*# sourceMappingURL=test.css.map */
65
```


La boucle @for

- Pour parcourir des listes, maps

```
@for $i from 1 to 6 { // exclut 6
  .m-#{ $i } {
    margin: 0 ($i * .25rem);
  }
}
```

```
74 .m-1 {
75   margin: 0 0.25rem;
76 }
77
78 .m-2 {
79   margin: 0 0.5rem;
80 }
81
82 .m-3 {
83   margin: 0 0.75rem;
84 }
85
86 .m-4 {
87   margin: 0 1rem;
88 }
89
90 .m-5 {
91   margin: 0 1.25rem;
92 }
```

```
@for $i from 1 through 6 { // inclut 6
  h#{ $i } {
    font-size: calc(3rem / $i );
  }
}
```

```
94 h1 {
95   font-size: 3rem;
96 }
97
98 h2 {
99   font-size: 1.5rem;
100 }
101
102 h3 {
103   font-size: 1rem;
104 }
105
106 h4 {
107   font-size: 0.75rem;
108 }
109
110 h5 {
111   font-size: 0.6rem;
112 }
113
114 h6 {
115   font-size: 0.5rem;
116 }
```

La boucle @each

- Pour également parcourir des listes et maps

```
@each $city in $cities {  
  ##{$city}::after {  
    content: "#{$city}";  
  }  
}
```

```
118 #paris::after {  
119   content: "paris";  
120 }  
121  
122 #nantes::after {  
123   content: "nantes";  
124 }  
125  
126 #lyon::after {  
127   content: "lyon";  
128 }  
129  
130 #marseille::after {  
131   content: "marseille";  
132 }
```

EXERCICE

Glodie Tshimini : contact@tshimini.fr



Exercice 13

- ▶ 0-exercices/exercice13.md

TO BE CONTINUED.

Glodie Tshimini: contact@tshimini.fr