MÉMO • HTML/CSS

La base

Le html utilise des balises pour délimiter ou créer des éléments, l'écriture est la suivante :

```
<balise>...</balise> (pour les balises par paire)
ou <balise /> ou <balise> (pour les autres)

<balise attribut="valeur">...</balise>
<balise attribut="valeur" />
➧ pas de guillemets typographiques “” « »
➧ on n’invente pas ses propres balises
```

<> html5

Un document HTML5 se structure ainsi :

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <title>Le titre de la page</title>
</head>
<body>
  <!-- … -->
</body>
</html>
(!+tab avec Emmet)
```

<> les balises de titrage

```
h1,h2,h3,h4,h5,h6 (pas de h7, h8...)
```

<> les premières balises à connaitre

```
p,ul,ol,li,a,strong,em,br,img,div,span,nav,article,
header,footer
```

faire un lien

```
<a href="http://exemple.fr">texte du lien</a>
```

insérer une image

```
<img src="image.jpg" alt="texte alternatif">
➧ src pour source (attention à la coquille « scr »)
<a href="http://exemple.fr"><img src="image.jpg"
alt="texte alternatif"></a><!-- image dans un lien-->
```

insérer une feuille de style

```
<link rel="stylesheet" href="style.css">
(link:css+tab avec Emmet)
```

la meta viewport (responsive)

Pour que le site s'adapte correctement à son écran.

```
<meta name="viewport" content="width=device-width,
initial-scale=1.0">
```

la class et l’id

L'id est **unique** dans la page HTML, la class est réutilisable.
Un seul id par élément, une ou plusieurs class par élément.

```
<div id="post-12" class="post sticky"> (1 id et 2 class)
  <p class="date">...</p>
</div>
```

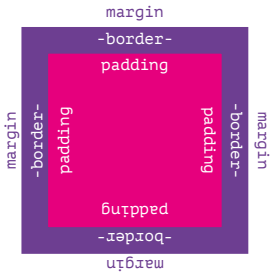
le CSS

```
sélecteur-a,
sélecteur-b,
sélecteur-père > enfants,
sélecteur-ancestre héritier{
  propriété-1 : valeur;
  propriété-2 : valeur;
}
```

propriétés à connaître

```
display, font-family, font-size, font-weight,
line-height, width, height, margin, padding, border,
color, background, text-transform, text-decoration,
position
```

propriétés de l’élément




```
.exemple{
  margin: 10px 12px 14px 18px;
  border: 1px solid #f60;
  padding: 20px;
}
```

margin/padding
1 valeur : les 4 marges identiques
2 : haut = bas & droite = gauche
3 : haut, droite = gauche & bas
4 : haut, droite, bas et gauche

mise en page & display

```
(HTML avec Emmet : .parent>(.item*3)+tab)
.item{
  display:block;
}
```

item
item
item

```
.item{
  display:inline-block;
}
```

item item item

Positionnement avec flex

```
.parent{
  display:flex;
  justify-content:space-between;
}
```

item item item

```
.parent{
  display:flex;
}
```

item item item*
*avec margin-left:auto

```
.parent{
  display:flex;
  flex-wrap:wrap;
}
```

item item item item item item item item item item item item item item item item

```
.parent{
  display:flex;
  flex-direction:column;
}
```

item
item
item

pratiques courantes

box-sizing + flex

Recommandation pour le calcul simplifié de la taille des blocs.

```
html {
  box-sizing: border-box;
}
*,*::before,*::after {
  box-sizing: inherit; min-width: 0; min-height: 0;
}
```

max-width

Empêche toutes les images de dépasser du navigateur

```
img{
  max-width: 100%; height: auto;
}
```

reset rapide

Pour un petit projet ou une démo.

```
*{
  margin: 0; padding: 0;
}
```

➧ Pour un projet plus important « normalize.css ou reboot.css ».

mobile-first

Pensez vos projets pour mobile dès le début, pour la maquette, la navigation, le tactile et codez-le d'abord dans sa version mobile !.

```
.article p{
  color: #333; /* régle pour tous les écrans */
}

@media (min-width:640px){
  .article p{
     columns: 2; /* pour exemple */
  }
}

@media (min-width:960px){
  .article p{
     columns: 3; /* pour exemple */
  }
}
```

Margin : auto

Quand un élément a une taille définie (width), margin : auto permet de centrer celui-ci en horizontal et même en vertical si son père est en display:flex.

```
.wrap{
  width: 960px;
  margin: 0 auto;
}
```

Grille avec flex + calc

```
.parent{
  display: flex; flex-wrap: wrap;
  justify-content: space-between;
}
.item-33{
 width: calc(33.33% - 16x); /* 2 gouttières de 24px */
}                  /* 16 = 2 × 24 × 0.33… [1] */
.item-50{
  width: calc(50% - 12px); /* 1 gouttière de 24px */
}                /* 12 = 1 × 24 × 0.50 [1] */
.item-100{
  width: 100%; /* 0 gouttière */
}
```

[1] (24 × nb de gouttière × %)

➧ espaces avant et après le signe - dans calc()

Grille avec Grid

```
.parent{
  display: grid; /* LA grille */
  grid-template-columns: 1fr 1fr 1fr; /* 3 colonnes */
  grid-gap; 24px; /* gouttières pour Safari */
  gap: 24px; /* idem mais pour tous */
}

.item-a{
  grid-column: 1 / 3; /* place sur deux colonnes */
  grid-row: 2 / 5; /* de la 2e ligne jusqu'à la 5e */
}
```

Grid propose de nombreuses possibilités et donc un vocabulaire assez long pour répondre à certaines problématiques. Si Grid ne s'adapte pas à votre projet et que vous ne voulez pas adapter celui-ci à Grid, il reste d'autres alternatives.

Noms de class fréquentes

Même si les noms sont libres, bien nommer ses class apporte un gain de lisibilité et de temps, voici quelques exemples pour vous inspirer.

`.wrap (pour emballer la page), .header, .navbar`
`ou .header-nav, .pagination, .is-hidden, .home, .page,`
`.post, .post-title, .footer`

En privilégiant les class dans le fichier CSS, on verra souvent des class avec le nom de la balise comme <header class="header">.

Raccourcis clavier

Pomme ⌘ + Shift ⇧ + r (rafraîchir la page sans cache)
Pomme ⌘ + Alt ⌥ + i (ouvrir l'inspecteur)
ctrl + Tab ⇥ (bascule entre les onglets)
Pomme ⌘ + Tab ⇥ (bascule entre applications)

les outils

VS Code
Atom
Extension Emmet
Extension Beautify
Extension Autoprefixer

Sites indispensables

https://developer.mozilla.org/fr/Apprendre
https://www.alsacreations.com/
https://www.grafikart.fr/
https://openclassrooms.com

https://css-tricks.com/
https://stackoverflow.com/
https://codepen.io/

https://cssgrid.io/
https://labs.jensimmons.com/
https://www.youtube.com/layoutland
https://gridbyexample.com/