Carl Brison

J'apprends facilement le **CSS** avec **GRID & FLEXBOX**



ellipses

Références sciences

J'apprends facilement le CSS avec GRID & FLEXBOX

Carl Brison

ellipses

Collection Références sciences

dirigée par Paul de Laboulaye
paul.delaboulaye@editions-ellipses.fr

Retrouvez tous les livres de la collection et des extraits sur www.editions-ellipses.fr

ISBN 9782340-035294
©Ellipses Édition Marketing S.A., 2019
32, rue Bargue 75740 Paris cedex 15

DANGER
LE PHOTOCOPILLAGE TUE LE LIVRE

Le Code de la propriété intellectuelle n'autorisant, aux termes de l'article L. 122-5.2° et 3°a), d'une part, que les « copies ou reproductions strictement réservées à l'usage privé du copiste et non destinées à une utilisation collective », et d'autre part, que les analyses et les courtes citations dans un but d'exemple et d'illustration, « toute représentation ou reproduction intégrale ou partielle faite sans le consentement de l'auteur ou de ses ayants droit ou ayants cause est illicite » (art. L. 122-4).
Cette représentation ou reproduction, par quelque procédé que ce soit constituerait une contrefaçon sanctionnée par les articles L. 335-2 et suivants du Code de la propriété intellectuelle.

www.editions-ellipses.fr

Avant-propos

Jamais le CSS n'aura été aussi facile pour positionner des éléments !

Je réalise des sites internet depuis le début des années 2000 et à cette époque nous nous servions des tableaux HTML pour gérer la mise en page de nos sites.
Puis de nouveaux supports pour visionner les sites internet sont arrivés sur le marché, tels que les smartphones ou les tablettes. Il a alors fallu penser à un nouveau mode de mise en page afin que les sites internet puissent s'adapter aux nouveaux supports qui les visionnaient. De là, la mise en page par tableaux a été abandonnée au profit d'une mise en page à l'aide du langage CSS.
Cependant, des propriétés CSS telles que ***FLOAT*** n'étaient pas adaptées à la mise en page d'un site internet. Et parfois il pouvait même être extrêmement laborieux de mettre en page un site internet à l'aide des propriétés classiques du CSS.
C'est d'ailleurs pour celà que des frameworks tels que ***bootstrap*** ont vu le jour, pour aider les concepteurs de sites internet à réaliser des mises en pages plus facilement.
Oui mais voilà, ***bootstrap*** n'est pas une technique officielle et le CCS ne peut pas se faire voler la vedette par un framework, d'où la naissance des technologies ***grid*** et ***flexbox*** !
Il s'agit de techniques très simples du CSS pour positionner tous les éléments d'un site internet, exactement comme on le souhaite.
Dans ce livre nous allons apprendre toutes les propriétés liées à ***grid*** et à ***flexbox*** ainsi que toutes leurs valeurs.
Nous étudierons ces technologies séparément puis nous verrons comment les utiliser ensemble pour mettre en page un site web.

A la fin de la lecture de ce livre, vous serez largement opérationnel pour réaliser la mise en page de n'importe quel site internet.

Je vous souhaite une bonne lecture,

Carl Brison.

Table des matières

::: Partie 1

CSS GRID

Chapitre 1.....11
Présentation de *grid*.....11
1.1. Introduction.....11
1.2. Comment est constituée la grille.....11
1.3. Le *display grid* et le *display inline-grid*.....14
1.4. Conclusion de ce chapitre.....15

Chapitre 2.....17
Le conteneur et ses propriétés.....17
2.1. Mise en place du conteneur et des contenus.....17
2.2. Un contenu peut devenir un conteneur 18
2.3. Création de colonnes 20
2.4. Gestion de la hauteur des lignes.....22
2.5. Les gouttières.....28
2.6. Une nouvelle unité de mesure.....30
2.7. La fonction *repeat*.....35

Chapitre 3.....39
Les contenus et leurs propriétés.....39
3.1. Les lignes de grille verticales.....39
3.2. Les lignes de grille horizontales.....42
3.3. Le mot-clé *span*.....45
3.4. Propriétés raccourcies.....47

Chapitre 4.....51
Autres propriétés.....51
4.1. Changer le sens d'affichage.....51
4.2. Créer une colonne virtuelle.....53
4.3. Créer une ligne virtuelle.....56
4.4. Application aux balises HTML5.....57
4.5. Définition des zones.....62
4.6. La fonction *minmax*.....68
4.7. La propriété *order*.....70

Chapitre 5.....75
Déplacement des contenus.....75
5.1. Présentation.....75
5.2. Alignement de la grille sur l'axe horizontal.....77
5.3. Alignement de la grille sur l'axe vertical.....81
5.4. Alignement de tous les contenus.....87
5.5. Alignement d'un contenu.....96

Chapitre 6.....101
Création d'une maquette d'un site responsive.....101
6.1. Présentation du travail.....101
6.2. Mise en place des bases du travail.....101
6.3. CSS côté smartphone.....105
6.4. CSS côté ordinateur.....108
6.5. Conclusion.....110

::: Partie 2

FLEXBOX

Chapitre 7.....115
Le display *flex*.....115
7.1. Mise en place de nos documents de base.....115
7.2. Déclarer du *flex* dans notre code CSS 117
7.3. Deux possibilités *flex* ou *inline-flex* 118
7.4. La largeur des contenus 118
7.5. Modifions le style par défaut.....119

Chapitre 8.....121
La différence entre *flex* et *inline-flex*.....121
8.1. La propriété *display*.....121
8.2. Plusieurs conteneurs *flex*.....122
8.3. Plusieurs conteneurs *inline-flex*.....124
8.4. Conclusion de ce chapitre.....125

Chapitre 9.....127
Définir la direction des contenus.....127
9.1. Une propriété CSS liée à la direction.....127
9.2. Diriger nos contenus en ligne.....127
9.3. Diriger nos contenus en colonne.....129
9.4. Diriger nos contenus en ligne inversée.....130
9.5. Diriger nos contenus en colonne inversée.....131
9.6. Conclusion.....132

Chapitre 10.....133
Le retour à la ligne.....133
10.1. Une propriété CSS liée au retour à la ligne.....133
10.2. Empêcher le retour à la ligne des contenus.....133
10.3. Autoriser le retour à la ligne des contenus.....135
10.4. Autoriser le retour à la ligne inversé des contenus.....136
10.5. Une propriété réunissant deux propriétés.....137
10.6. Conclusion.....137

Chapitre 11.....139
L'axe principal et l'axe secondaire.....139
11.1. La notion d'axe.....139
11.2. Alignement sur l'axe principal.....140
11.3. Alignement horizontal.....141
11.4. Alignement vertical.....143
11.5. Récapitulatif de ce que nous savons.....150
11.6. Alignement sur l'axe secondaire.....151
11.7. Inversion de l'axe principal.....156
11.8. Alignement d'un contenu particulier.....160
11.9. Alignement de plusieurs lignes ou colonnes.....167
11.10. Conclusion.....175

Chapitre 12.....177
Manipulation des contenus.....177
12.1. Gérer les ordres d'affichage.....177
12.2. Augmenter la largeur d'un contenu.....181
12.3. Diminuer ou définir la largeur d'un contenu.....185
12.4. La super propriété *flex* 189
12.5. Conclusion.....190

Chapitre 13.....191
Création d'une maquette.....191
13.1. Présentation du travail.....191
13.2. Première partie.....192
13.3. Deuxième partie.....193
13.4. Conclusion.....198

::: Partie 3
GRID & FLEXBOX ENSEMBLE

Chapitre 14.....201
Utiliser *grid* & *flexbox* ensemble.....201
14.1. Le principe de base.....201
14.2. Le HTML 5.....202
14.3. Les principes de base de *grid*.....206
14.4. Le principe des *media queries*.....210
14.5. Le *viewport*.....211
14.6. Mise en place du responsive.....212
14.7. Les principes de base de *flexbox*.....215

Index lexical.....221

Partie 1

CSS-GRID

Chapitre 1

Présentation de grid

1.1. Introduction

Tout d'abord, pour accéder à la technologie ***grid***, nous devons utiliser la propriété CSS ***display*** à qui nous donnons la valeur ***grid***. Ceci aura pour conséquence de créer une grille virtuelle afin de nous faciliter la mise en page des élements HTML. Voici le code CSS :

```
display: grid;
```

1.2. Comment est constituée la grille

La grille est constituée de lignes et de colonnes, elles-mêmes séparées par des lignes de grille. Voici à quoi ressemble la grille :

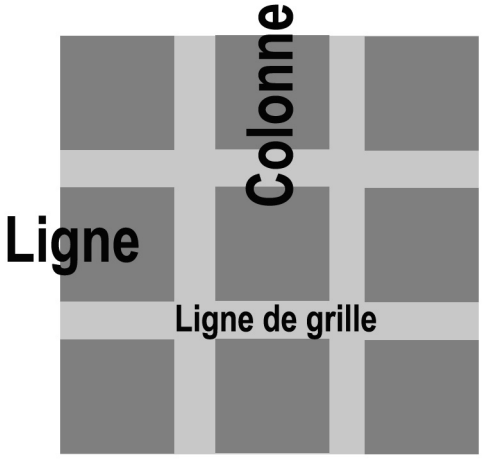


Figure 1-1 : Grille CSS

La grille est constituée de plusieurs lignes et de plusieurs colonnes. Il peut y avoir autant de lignes et de colonnes que l'on souhaite
La grille est également constituée de lignes de grille. Les lignes de grille sont les lignes qui vont séparer les différentes lignes et colonnes qui constituent la grille. La ligne de grille est également présente tout autour de la grille.
En étudiant la figure ci-dessus *(figure 1-1)*, nous pouvons nous apercevoir que celle-ci est constituée de 3 lignes.

Voici les 3 lignes de notre grille :

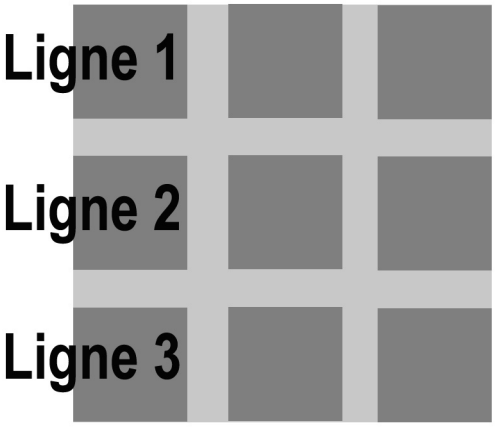


Figure 1-2 : Les lignes

De même qu'en étudiant la figure 1-1, nous pouvons nous apercevoir que celle-ci est constituée de 3 colonnes.

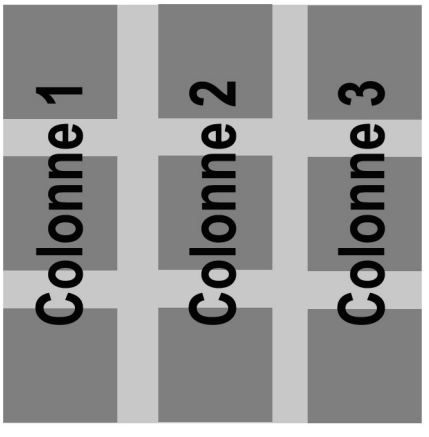


Figure 1-3 : Les colonnes

Quant aux lignes de grille, nous avons sur la figure 1-1, 4 lignes de grille horizontale et 4 lignes de grille verticale.

Voici les 4 lignes de grille horizontale :

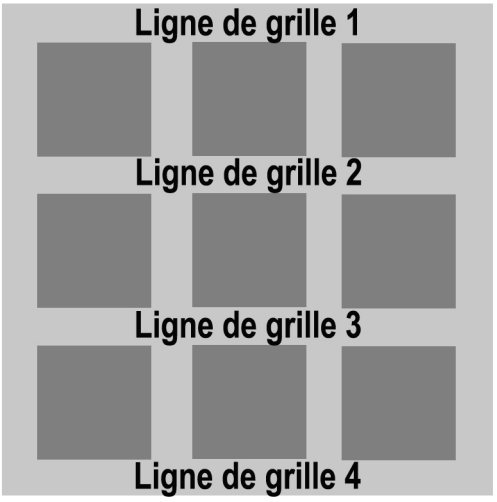


Figure 1-4 : Lignes de grille horizontale

Voici les 4 lignes de grille verticale :

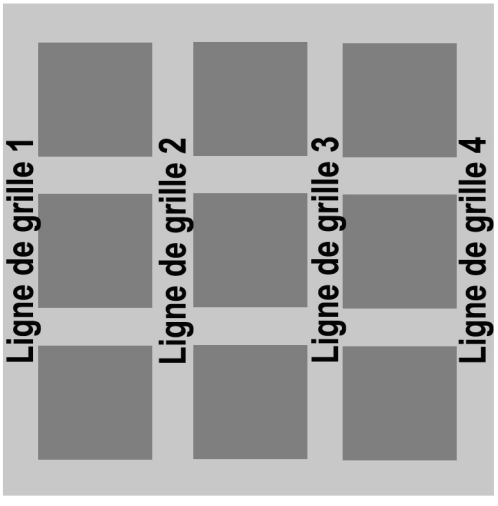


Figure 1-5 : Lignes de grille verticale

1.3. Le *display grid* et le *display inline-grid*

Comme nous l'avons vu au début de ce chapitre, pour accéder à la technologie ***grid*** nous devons nous servir de la propriété CSS ***display***.

Nous allons créer deux boîtes ***div*** dans un fichier HTML à qui nous allons donner un identifiant. Nous écrirons le mot ***Grille 1*** dans la première boîte et le mot ***Grille 2*** dans la seconde boîte.

```
<div id="grille1">Grille 1</div>
<div id="grille2">Grille 2</div>
```

Nous allons ensuite donner une couleur de fond à ces boîtes ***div*** dans un fichier CSS externe.

```
#grille1 {
        background-color: #f00;
}
#grille2 {
        background-color: #00f;
}
```

Voici le résultat que nous allons obtenir dans un navigateur :



Figure 1-6 : Mise en place de 2 boîtes div

Sans surprise, nous constatons que les deux boîtes ***div*** sont empilées l'une sur l'autre car par défaut elles ont un comportement de type ***bloc***.
Nous allons maintenant ajouter la propriété ***display*** à nos deux identifiants à qui nous donnerons la valeur ***grid***.

```
#grille1 {
        background-color: #f00;
        display: grid;
}
#grille2 {
        background-color: #00f;
        display: grid;
}
```

Voyons à présent ce que cela va nous donner comme résultat dans un navigateur :



Figure 1-7 : Résultat du display grid

Le constat que nous faisons ici est que le fait d'avoir défini un ***display: grid*** aux deux boîtes ***div*** n'a absolument pas modifié le résultat dans un navigateur.
Nous pouvons donc conclure que le fait de définir un ***display: grid*** à une boîte ***div*** donne à cette boîte ***div*** un comportement de type ***bloc***. Il n'est donc pas possible de positionner ces deux boîtes ***div*** l'une à côté de l'autre.
Comment faire pour modifier cet état si malgré tout on souhaite positionner ces deux boîtes ***div*** l'une à côté de l'autre ? Il suffit tout simplement de modifier la valeur du ***display*** en la passant de ***grid*** à ***inline-grid***.
Vérifions cela en modifiant notre code CSS :

```
#grille1 {
        background-color: #f00;
        display: inline-grid;
}
#grille2 {
        background-color: #00f;
        display: inline-grid;
}
```

Voici le résultat obtenu dans un navigateur



Figure 1-8 : Résultat du display inline-grid

Nos deux boîtes ***div*** viennent de perdre leur comportement de type ***bloc*** pour adopter celui du type ***inline***. Elles ont pour largeur ce qu'elles contiennent.

1.4. Conclusion de ce chapitre

Pour accéder aux grilles, nous devons utiliser la propriété CSS ***display***. Si nous souhaitons que la boîte qui va contenir cette propriété CSS soit de type ***bloc***, alors nous donnerons la valeur ***grid***. Si nous souhaitons que la boîte qui va contenir cette propriété CSS soit de type ***inline***, alors nous donnerons la valeur ***inline-grid***.
Ce qu'il faut également retenir est que la boîte qui va contenir le ***display: grid*** deviendra alors un ***conteneur***. C'est-à-dire qu'elle contiendra une grille.

Chapitre 2

Le conteneur et ses propriétés

2.1. Mise en place du conteneur et des contenus

Il est très important de bien comprendre ce qu'est un conteneur et ce que sont les contenus. Le conteneur est une boîte qui va contenir d'autres boîtes, c'est-à-dire des contenus. C'est au conteneur que l'on donne la propriété ***display: grid***.
Prenons tout de suite un exemple en créant des boîtes ***div*** dans un document HTML. Nous donnerons un identifiant à la boîte qui sera le conteneur et nous donnerons une classe aux boîtes qui seront les contenus.

```
<div id="grille">
        <div class="un">un</div>
        <div class="deux">deux</div>
        <div class="trois">trois</div>
        <div class="quatre">quatre</div>
        <div class="cinq">cinq</div>
        <div class="six">six</div>
</div>
```

Ici le conteneur a pour identifiant le mot ***grille*** et les contenus ont pour classe les mots ***un***, ***deux***, ***trois*** jusqu'à ***six***.
Le conteneur renferme les six boîtes ***div***. C'est à lui que nous allons donner la technologie ***grid*** au travers d'une feuille de style. Quant aux contenus, nous leur donnerons à chacun une couleur différente.

```
#grille {
        display: grid;
}
.un {
        background-color: #fcf;
}
.deux {
        background-color: #f9f;
}
.trois {
        background-color: #f6f;
}
```

```
.quatre {
        background-color: #fc9;
}
.cinq {
        background-color: #fc6;
}
.six {
        background-color: #f96;
}
```

Voici le résultat que nous obtenons dans un navigateur :

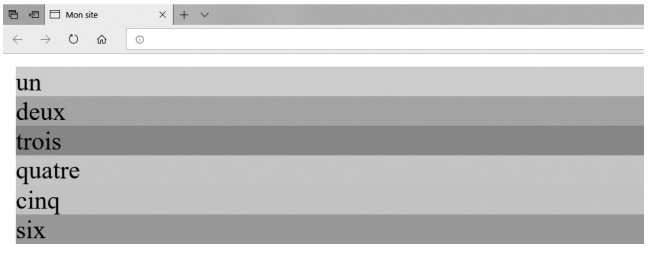


Figure 2-1 : Conteneur et contenus

Nous pouvons observer que le conteneur ainsi que les contenus ont par défaut un comportement de type ***bloc*** et ce malgré le fait que l'on a donné la propriété ***display*** au conteneur.
Par conséquent, le fait de donner la propriété ***display: grid*** à un conteneur ne modifie en rien le comportement par défaut des boîtes ***div***. En revanche, cela nous permet d'entrer dans la technologie ***grid*** et d'accéder ainsi à des nouvelles propriétés CSS qui vont nous permettre de créer une grille virtuelle et de positionner les contenus selon cette grille.

2.2. Un contenu peut devenir un conteneur

Dans la seconde partie de ce livre, nous étudierons la technologie ***flexbox***. Il s'agit là-aussi d'une technologie CSS de positionnement d'éléments. Cependant la technologie ***flexbox*** et la technologie ***grid*** ne sont pas concurrentes mais complémentaires.
Bien que nous n'ayons pas encore étudié la technologie ***flexbox***, nous allons faire une petite parenthèse en nous servant de ***flexbox*** pour transformer les contenus en conteneurs.

Pour cela, nous allons donner une classe supplémentaire à nos contenus et nous appellerons cette nouvelle classe ***flexbox***.

```
<div id="grille">
        <div class="un flexbox">un</div>
        <div class="deux flexbox">deux</div>
        <div class="trois flexbox">trois</div>
        <div class="quatre flexbox">quatre</div>
        <div class="cinq flexbox">cinq</div>
        <div class="six flexbox">six</div>
</div>
```

Dans notre feuille de style, nous allons ajouter cette nouvelle classe et lui donner la propriété ***display***, mais cette fois ce ne sera pas la valeur ***grid*** mais la valeur ***flex***. Sans entrer dans le détail de la technologie ***flexbox*** que nous étudierons dans la deuxième partie de ce livre, le fait de donner la propriété ***display: flex*** à la classe ***flexbox*** fait que tous les éléments HTML qui possèdent cette classe deviendront des conteneurs.
A cette classe ***flexbox***, nous allons ordonner que les éléments HTML soient centrés, qu'ils aient une taille de 30 pixels, qu'ils soient écrits en gras, qu'ils aient un padding de 20 pixels et que la boîte ***div*** qui possède cette classe ait une bordure solide de 1 pixel.

```
.flexbox {
        display: flex;
        justify-content: center;
        padding: 20px;
        font-size: 30px;
        font-weight: bold;
        border: 1px solid #333;
}
```

Nous reviendrons très largement sur la technologie ***flexbox*** à la deuxième partie de ce livre. Comprenez simplement ici que la propriété ***justify-content*** va nous permettre de centrer un texte sur la largeur du nouveau conteneur.

Voici le résultat que nous obtenons dans un navigateur :

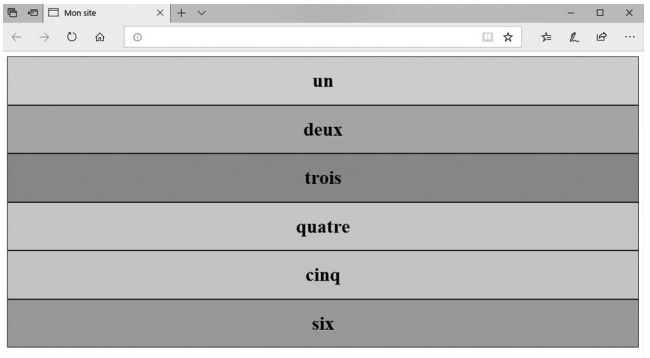


Figure 2-2 : Les contenus deviennent des conteneurs

De ce fait, nous nous retrouvons avec un conteneur qui possède six contenus qui sont eux-mêmes des conteneurs.

2.3. Création de colonnes

Revenons à notre boîte ***div*** à qui nous avons donné pour identifiant le nom ***grille***. Au niveau CSS, nous en avons fait un conteneur puisque nous lui avons donné la propriété ***display*** à qui nous avons donné la valeur ***grid***.
De ce fait, nous pouvons maintenant accéder à la technologie ***grid*** pour ce conteneur. Donc nous allons pouvoir mettre en place une grille CSS.
Une grille est naturellement constituée de deux choses : des **lignes** et des **colonnes**. Ici nous allons voir comment créer des colonnes grâce à une nouvelle propriété CSS. Cette nouvelle propriété se nomme ***grid-template-columns***. Elle attend des valeurs pour chaque colonne créée. Par exemple si on veut créer deux colonnes, on écrira deux valeurs. Chaque valeur correspondra à la largeur de la colonne ainsi créée.
Si on souhaite créer deux colonnes de 200 pixels chacune, nous allons alors ajouter cette nouvelle propriété à l'identifiant ***grille*** en lui donnant pour valeur 200px et 200px.

```
#grille {
        display: grid;
        grid-template-columns: 200px 200px;
}
```

Ceci aura pour conséquence de détruire le comportement de type ***bloc*** des contenus.