

# Intégration Web

MMI 1 – CM#2 S2

Danielo **JEAN-LOUIS**  
Michele **LINARDI**

# Mise en page - Historique

- Au début, les développeurs utilisaient des tableaux (<table>) pour faire la mise en page
  - **Ne jamais, jamais utiliser ceci !**
  - A utiliser uniquement :
    - Pour afficher des données tabulaires
    - Faire des e-mails

## Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/float>
- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/table>
- <https://www.alsacreations.com/article/lire/1209-display-inline-block.html>
- <https://putaindecode.io/podcasts/s02e04-css-layout> – podcast en français

# Mise en page - Historique

- Ensuite la propriété "float"
  - **A éviter. Crée plein d'effets de bord**
  - ~~Adapté pour certaines mise en page~~

## Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/float>
- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/table>
- <https://www.alsacreations.com/article/lire/1209-display-inline-block.html>
- <https://putaindecode.io/podcasts/s02e04-css-layout> – podcast en français

# Mise en page - Historique

- Puis la valeur "inline-block" pour la propriété "display"
  - Très limité dans l'usage
- Et enfin le CSS3 apporta **flexbox** (vu en S1)
  - Et **grid** qui est le sujet du jour

## Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/float>
- <https://developer.mozilla.org/fr/docs/Web/HTML/Element/table>
- <https://www.alsacreations.com/article/lire/1209-display-inline-block.html>
- <https://putaindecode.io/podcasts/s02e04-css-layout> – podcast en français

# CSS Grid

- Nouveauté CSS 3
- Petit frère de flexbox
- Permet l'affichage en grille
  - Contrevient aux limites de flexbox

## Source(s) :

- [https://developer.mozilla.org/fr/docs/Web/CSS/CSS\\_grid\\_layout](https://developer.mozilla.org/fr/docs/Web/CSS/CSS_grid_layout)

# CSS Grid

- Gestion du placement des éléments sur deux dimensions
  - Flexbox ne gère qu'une dimension
- Géré par tous les navigateurs modernes
  - PC et terminaux mobiles

## Source(s) :

- [https://developer.mozilla.org/fr/docs/Web/CSS/CSS\\_grid\\_layout](https://developer.mozilla.org/fr/docs/Web/CSS/CSS_grid_layout)

# CSS Grid

- Possibilité de faire une grille dans une autre
- Permet de faire des grilles inégales
  - Lignes et colonnes de tailles différentes
  - Ajoute du dynamisme à la mise en page du site

## Source(s) :

- [https://developer.mozilla.org/fr/docs/Web/CSS/CSS\\_grid\\_layout](https://developer.mozilla.org/fr/docs/Web/CSS/CSS_grid_layout)



# CSS Grid

- Comme pour flexbox, c'est le parent qui définit ce type d'affichage grâce à la propriété CSS "display"

A code editor window with a pink border and a dark purple background. The title bar shows three colored circles (red, yellow, green) and a blue icon with a white 'E' followed by the text 'css-grid'. The code is written in a light-colored font with syntax highlighting: the selector '.mon-selecteur' is in orange, the property 'display' is in blue, the value 'grid' is in orange, and the comment '// [...]' is in light gray. The code is enclosed in curly braces.

```
.mon-selecteur {  
  display: grid;  
  // [...]  
}
```

Exemple de code

Source(s) :

- [https://developer.mozilla.org/fr/docs/Web/CSS/CSS\\_grid\\_layout](https://developer.mozilla.org/fr/docs/Web/CSS/CSS_grid_layout)

## CSS Grid – Propriétés (liste non exhaustive)

- `grid-template-rows` : définit les lignes
- `grid-template-columns` : définit les colonnes
- `grid-gap` : espacement des gouttières
- `align-items`, `justify-items` : comme dans flexbox

## grid-template-rows

- Définit les lignes de la grille
  - Nombre infini (comme pour les colonnes)
- Les enfants occupent naturellement les lignes (en fonction des colonnes)

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-template-rows>

# grid-template-rows



css-grid

```
.mon-selecteur {  
  display: grid;  
  grid-template-rows: 250px 400px;  
}
```

Ici on définit deux lignes, une de 250px et une autre de 400px

## grid-template-rows

- S'il y a plus d'éléments que de lignes définies, les éléments en plus auront une hauteur égale à leur contenu

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-template-rows>

**Essayons avec plus de lignes**

# grid-template-rows



css-grid

```
.mon-selecteur {  
  display: grid;  
  grid-template-rows: 250px 400px 400px 400px 400px 250px;  
}
```

Maintenant, nous avons six colonnes mais...

**...on se répète**



## Fonction repeat()

- Ne fonctionne qu'avec les propriétés “grid-template-columns” et “grid-template-rows”
- Permet de répéter une valeur de grille
- Accepte deux arguments :
  - Nombre de répétitions (entier positif)
  - Valeurs à répéter, séparée par un espace

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/repeat>

# Fonction repeat()



css-grid

```
.mon-selecteur {  
  display: grid;  
  grid-template-rows: 250px repeat(4, 400px) 250px;  
}
```

Même code qu'auparavant mais sans répétition de code

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/repeat>

## grid-template-columns

- Définit les colonnes de la grille
  - Nombre infini (comme pour les lignes)
-

## grid-template-columns

- Les enfants occupent naturellement la grille
  - Ex : trois colonnes → **Premier** élément, **première** colonne. **Deuxième** élément, **deuxième** colonne. **Troisième** élément, **première** colonne, etc.

# grid-template-rows



css-grid

```
.mon-selecteur {  
  display: grid;  
  grid-template-columns: 250px 400px;  
}
```

Ici on définit deux colonnes, une de 250px et une autre de 400px

## Unité fr

- Représente une fraction de l'espace restant
  - $1\text{fr} = 100\%$  de l'espace restant
- Accepte une valeur décimale positive
- Prend en compte les gouttières dans le calcul
  - Contrairement à l'unité %
- A préférer à l'unité %

Source(s) :

- [https://developer.mozilla.org/fr/docs/Web/CSS/CSS\\_grid\\_layout/Basic\\_concepts\\_of\\_grid\\_layout#lunit%C3%A9\\_fr](https://developer.mozilla.org/fr/docs/Web/CSS/CSS_grid_layout/Basic_concepts_of_grid_layout#lunit%C3%A9_fr)

# Unité fr



css-grid

```
.mon-selecteur {  
  display: grid;  
  grid-template-columns: repeat(4, 1fr);  
}
```

Ici on définit quatre colonnes de tailles égales

Source(s) :

- [https://developer.mozilla.org/fr/docs/Web/CSS/CSS\\_grid\\_layout/Basic\\_concepts\\_of\\_grid\\_layout#lunit%C3%A9\\_fr](https://developer.mozilla.org/fr/docs/Web/CSS/CSS_grid_layout/Basic_concepts_of_grid_layout#lunit%C3%A9_fr)

# Livecode

<https://download-directory.github.io/?url=https%3A%2F%2Fgithub.com%2FDanYellow%2Fcours%2Ftree%2Fmain%2Fintegration-web-s2%2Fcours-magistraux%2Fnumero-2%2Fressources>



# Placer les éléments

- Utilisation des propriétés `grid-{column, row}-{start, end}`
  - Définissent le début et jusqu'où l'élément s'étire
  - Valeur par défaut "auto"
    - Le navigateur place tout seul l'élément

## Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-column-start>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-column-end>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-row-start>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-row-end>

# Placer les éléments

- L'ordre des balises dans le HTML n'est plus pris en compte, c'est vous qui placez les balises comme bon vous semble

## Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-column-start>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-column-end>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-row-start>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-row-end>

# Placer les éléments



css-grid

```
.mon-selecteur {  
  display: grid;  
  grid-template-columns: 250px 400px;  
}
```

Reprenons notre exemple

## Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-column-start>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-column-end>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-row-start>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-row-end>

# Placer les éléments



Voilà ce que génère notre navigateur

## Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-column-start>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-column-end>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-row-start>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-row-end>

# Placer les éléments



Voilà que notre navigateur interprète. Il crée des “points” et étire les éléments en fonction des propriétés CSS

## Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-column-start>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-column-end>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-row-start>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-row-end>

# Placer les éléments



Notre élément en bleu s'étire de 1 à 2, il est possible de changer ceci

## Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-column-start>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-column-end>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-row-start>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-row-end>

# Placer les éléments

- Le comptage commence à 1 et finit au nombre de colonnes / lignes + 1
  - Exemple : Trois colonnes = 1 à 4
- On peut compter
  - De gauche à droite : nombres positifs
  - De droite à gauche : nombres négatifs

## Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-column-start>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-column-end>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-row-start>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-row-end>

# Placer les éléments



css-grid

```
.mon-selecteur {  
  grid-column-start: 1;  
  grid-column-end: 3;  
}
```

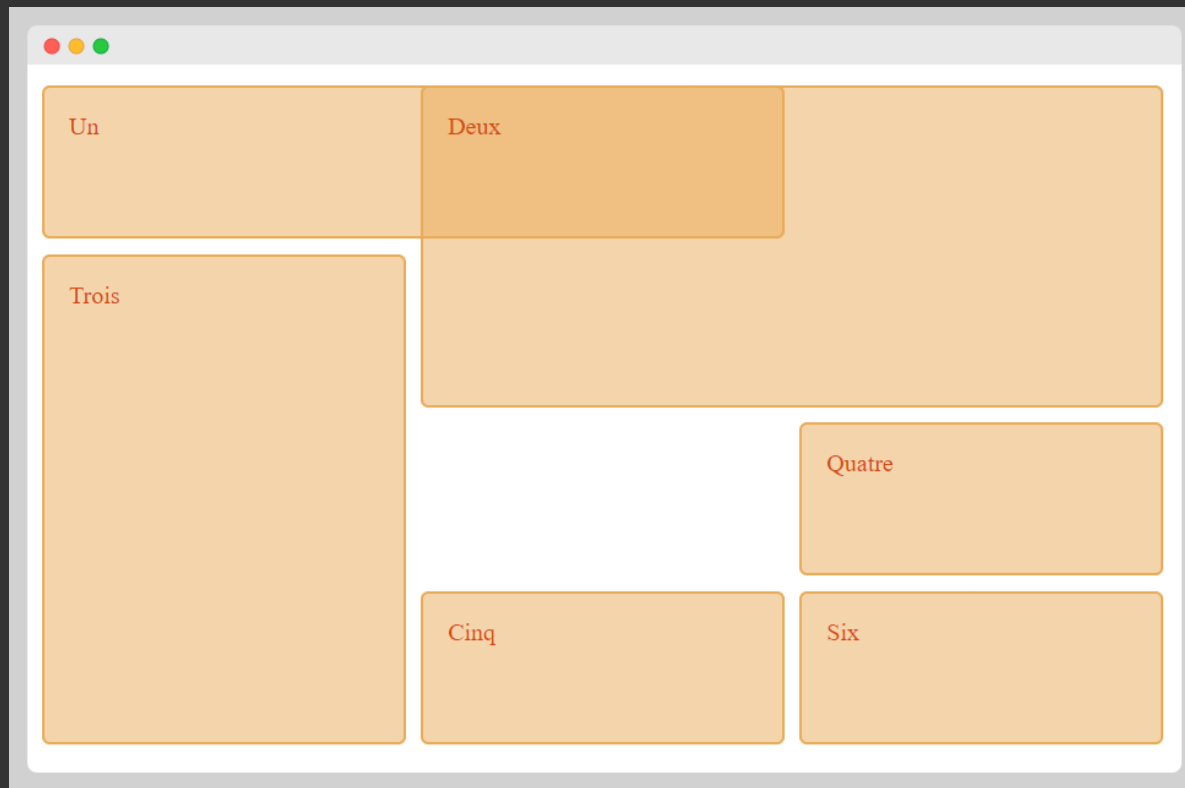
Notre élément occupera deux colonnes grâce à ce code (le principe est le même avec les lignes et la propriété “grid-row-start” et “grid-column-end”)

## Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-column-start>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-column-end>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-row-start>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-row-end>



# Placer les éléments

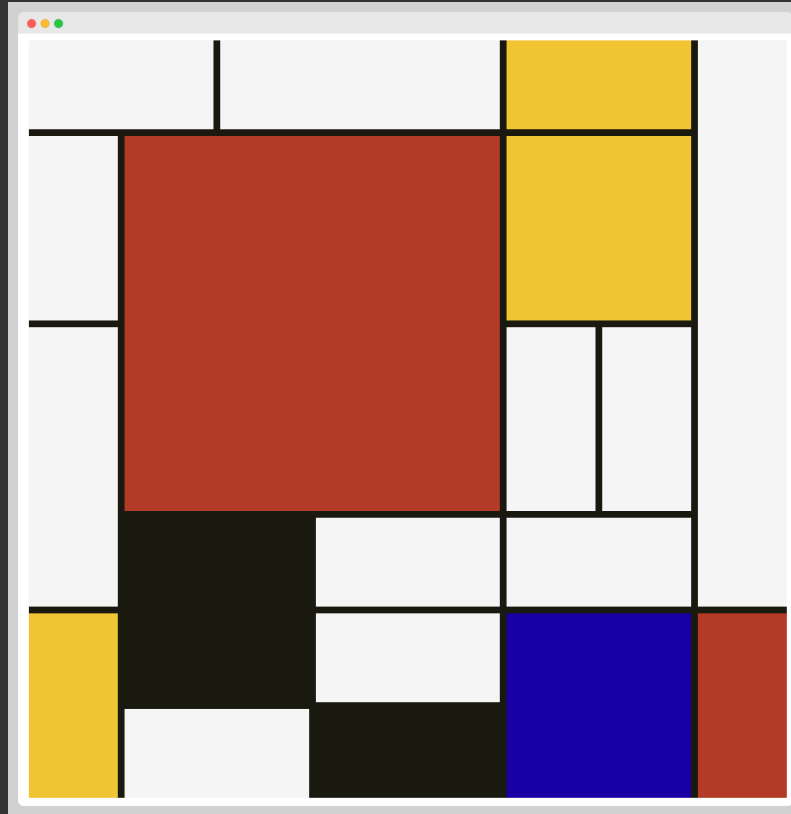


On peut faire des mises en page plus complexe grâce à ces propriétés CSS – Crédit mdn

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-column-start>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-column-end>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-row-start>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-row-end>

# Placer les éléments



Et même plus (code dans la ressource)

Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-column-start>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-column-end>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-row-start>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-row-end>

# Navigateur

- Chrome et Firefox proposent des aides visuelles pour afficher la grille du conteneur

# Navigateur

The screenshot displays a web browser window with a CSS Grid layout. The layout consists of four items arranged in a 2x2 grid. Item 1 (top-left) is a pink box labeled 'item 1'. Item 2 (top-right) is a green box labeled 'item 2'. Item 3 (bottom-left) is a purple box labeled 'item 3'. Item 4 (bottom-right) is a red box labeled 'item 4'. Each item has a small red box in its top-left corner containing a number and a sign, representing its grid coordinates: Item 1 (1, 1), Item 2 (2, 1), Item 3 (1, 2), and Item 4 (2, 2). Below the browser window, the developer tools are open, showing the 'Elements' panel on the left and the 'Styles' panel on the right. The 'Elements' panel shows the HTML structure of the grid, with the 'div.grid' element selected. The 'Styles' panel shows the CSS rules for the grid, including the grid-template-columns and grid-template-rows properties.

```
<!DOCTYPE html>
<html>
  <head>
    </head>
  <body>
    <div class="grid">
      <div class="row1-column1">item 1</div>
      <div class="row1-column2">item 2</div>
      <div class="row2-column1">item 3</div>
      <div class="row2-column2">item 4</div>
    </div>
  </body>
</html>
```

Styles panel:

```
.grid {
  -ms-grid-columns: 100px 100px;
  grid-definition-columns: 100px 100px;
  grid-template-columns: 100px 100px;
  -ms-grid-rows: 50px 50px;
  grid-definition-rows: 50px 50px;
  grid-template-rows: 50px 50px;
}

.grid {
  display: -ms-grid;
  display: grid;
  background-color: #DDD;
}

div {
  display: block;
  unicode-bidi: isolate;
}
```

Après avoir cliqué sur “grid”, notre navigateur affiche notre grille

# Fonctionnalités supplémentaires

- Possibilité de faire une mise en page en “brique”
  - Propriété “masonry” pas encore gérée par les navigateur (2024)
- Possibilité de nommer les cellules (propriété “grid-template-areas”)
- ...

## Source(s) :

- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-template-areas>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-row>
- <https://developer.mozilla.org/fr/docs/Web/CSS/grid-column>

# Exemples de mise en page avec CSS Grid

- <https://gridbyexample.com/examples/>
- <https://igalia.github.io/css-grid-layout/>
- <https://grid.layoutit.com/>
  - Générateur de code pour CSS Grid

**Questions ?**

