

Philippe GENOUD (LIG-STeamer)  
Philippe.Genoud@imag.fr

**M2CCI – M2 GEOMAS 2024-2025**  
**cours PLAI-TW (Technologies du Web)**

# CSS (2<sup>ème</sup> partie)

## positionnement des éléments

dernière modification : 09/10/2024 11:50



This work is licensed under a [Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License](https://creativecommons.org/licenses/by-nc-sa/4.0/).




- chaque élément HTML considéré comme une **boîte rectangulaire**.
- **flux du document** : ordre dans lequel le navigateur affiche ces boîtes.
  - **flux normal** (par défaut)
    - un élément père est un conteneur
    - un élément fils s'affiche dans son conteneur
    - élément bloc
      - s'affiche en dessous de son frère précédent.
      - occupe toute la largeur disponible dans son conteneur.
    - élément en ligne
      - s'affiche à côté de son frère précédent.
      - retour à la ligne quand il n'y a plus de place dans le conteneur.
  - **flux personnalisé**
    - certaines propriétés CSS permettent de sortir des éléments du flux normal

# Positionnement en flux

- exemple de flux normal

```
<html lang="fr">
  <head>
    <title>Exemple boites</title>
    <meta charset="utf-8" />
  </head>
  <body>
    <header>
      <h1>Elément h1</h1>
    </header>
    <div>
      Cet élément div contient un élément ul, suivi d'un élément p.
      <ul id="menuAccesRapide">
        <li>liste item 1</li>
        <li>liste item 2</li>
      </ul>
      <p>
        Cet élément p contient un élément span qui commence <span> ici et s'arrête là</span>.
        Il contient aussi un <a href="#">élément a (lien hyper texte)</a>
        et un élément img 
        Tout cela pour montrer la notion de boite associée
        à la notion d'élément.
      </p>
    </div>
  </body>
</html>
```

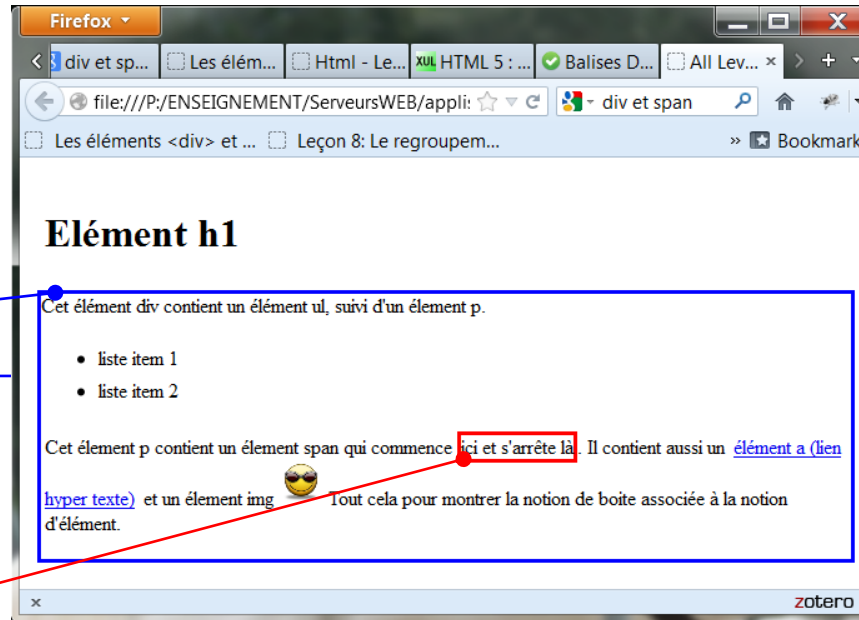
The screenshot shows a Firefox browser window with the following content:

- Header:** Elément h1
- Paragraph:** Cet élément div contient un élément ul, suivi d'un élément p.
- List:**
  - liste item 1
  - liste item 2
- Paragraph:** Cet élément p contient un élément span qui commence ici et s'arrête là. Il contient aussi un [élément a \(lien hyper texte\)](#) et un élément . Tout cela pour montrer la notion de boite associée à la notion d'élément.

# Positionnement en flux

- `<div>` et `<span>` éléments génériques sans information structurante prédéfinie
  - utilisés en association avec des feuilles de style CSS ou du Javascript via les attributs `id`, `class` ou `style`

```
<html lang="fr">
<head>
  <title>Exemple boites</title>
  <meta charset="utf-8" />
</head>
<body>
  <header>
    <h1>Elément h1</h1>
  </header>
  <div>
    Cet élément div contient un élément ul, suivi d'un élément p.
    <ul id="menuAccesRapide">
      <li>liste item 1</li>
      <li>liste item 2</a></li>
    </ul>
    <p>
      Cet élément p contient un élément span qui
      commence <span> ici et s'arrête là</span>
      Il contient aussi un <a href="#">élément a (lien hyper texte)</a>
      et un élément img 
      Tout cela pour montrer la notion de boite associée
      à la notion d'élément.
    </p>
  </div>
</body>
</html>
```



**div** : balise de type bloc

- zone rectangulaire qui ne peut être répartie sur plusieurs lignes
- peut contenir tous les autres éléments de type bloc ou en ligne

**span** : balise de type en ligne (*inline*)

- s'inscrit dans le flux du contenu, peut être répartie sur plusieurs lignes
- peut contenir tous les autres éléments en ligne

```

1 <!DOCTYPE html>
2 <html lang="fr">
3 <head>
4 <title>All Levels</title>
5 <meta charset="UTF-8">
6 <link rel="stylesheet" href="css/styles1.css">
7 </head>
8 <body>
9 <header>
10 <h1>Elément h1</h1>
11 </header>
12 <div>
13 <p>Cet élément div contient un élément ul, suivi d'un élément p.</p>
14 <ul>
15 <li>liste item 1</li>
16 <li>liste item 2</li>
17 </ul>
18 <p>
19 <span>Cet élément p contient un élément span qui
20 commence ici et s'arrête là</span>.
21 Il contient aussi un <a href="#">élément a (lien hyper texte)
22 et un élément img 
23 Tout cela pour montrer la notion de boite associée
24 à la notion d'élément.
25 </p>
26 </div>
27 <div>
28 <p>
29 Ceci est un paragraphe dans un deuxième élément div. Pour avoir
30 du texte un peu logn voilà du Lorem ipsum. Lorem ipsum dolor sit amet,
31 consectetur adipisicing elit. Quas pariatur adipisci dolore in fugit
32 ducimus. Eum reprehenderit iste quae id optio delectus!
33 </p>
34 </div>
35 </body>
36 </html>

```

## Elément h1

Cet élément div contient un élément ul, suivi d'un élément p.

- liste item 1
- liste item 2



Cet élément p contient un élément span qui commence ici et s'arrête là. Il contient aussi un [élément a \(lien hyper texte\)](#) et un élément img

Tout cela pour montrer la notion de boite associée à la notion d'élément.

Ceci est un paragraphe dans un deuxième élément div. Pour avoir du texte un peu logn voilà du Lorem ipsum. Lorem ipsum dolor sit amet, consectetur adipisicing elit. Quas pariatur adipisci dolore incidunt quasi in fugit ducimus. Eum reprehenderit iste quae id optio delectus!

## Elément h1

Cet élément div contient un élément ul, suivi d'un élément p.

- liste item 1
- liste item 2



Cet élément p contient un élément span qui commence ici et s'arrête là. Il contient aussi un [élément a \(lien hyper texte\)](#) et un élément img

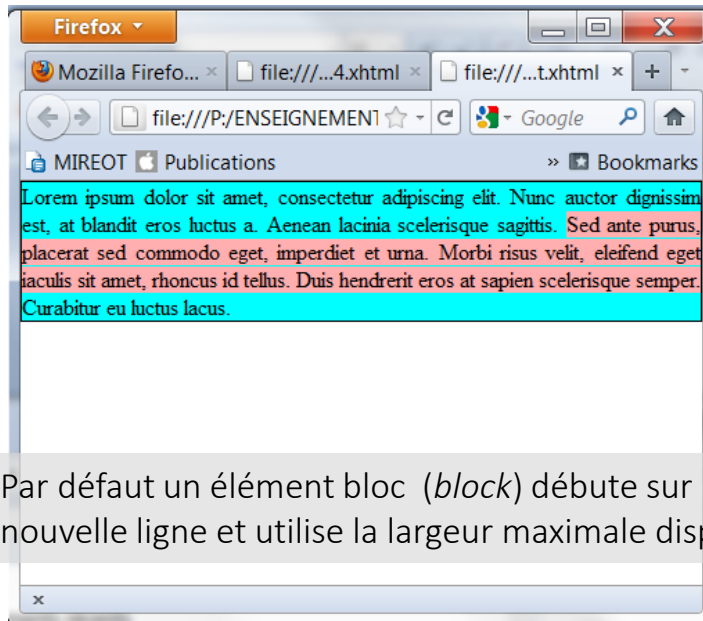
Tout cela pour montrer la notion de boite associée à la notion d'élément.

Ceci est un paragraphe dans un deuxième élément div. Pour avoir du texte un peu logn voilà du Lorem ipsum. Lorem ipsum dolor sit amet, consectetur adipisicing elit. Quas pariatur adipisci dolore incidunt quasi in fugit ducimus. Eum reprehenderit iste quae id optio delectus!

# Positionnement en flux

- positionnement dans le flux en supprimant les marges par défaut

```
<div id ="div1">
  <p id="p1">
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc auctor dignissim est, at blandit eros luctus a.
    Aenean lacinia scelerisque sagittis. <span id="span1">Sed ante purus, placerat sed commodo eget,
    imperdiet et urna. Morbi risus velit, eleifend eget iaculis sit amet, rhoncus id tellus.
    Duis hendrerit eros at sapien scelerisque semper.</span> Curabitur eu luctus lacus.
  </p>
</div>
```



Par défaut un élément bloc (*block*) débute sur une nouvelle ligne et utilise la largeur maximale disponible

Ce comportement est défini par la propriété **display** (valeurs: **inline**, **block**, **none**, **inline-block**...)

```
body {
  text-align: justify;
  padding: 0px;
  margin: 0px;
}
#div1 {
  background: yellow;
}
p {
  border-style: solid;
  border-width: thin;
  padding: 0px;
  margin: 0px;
}
#p1 {
  background: cyan;
}
#span1 {
  background: #ffafaf;
}
```

Par défaut un élément en-ligne (*in line*) ne débute pas sur une nouvelle ligne et utilise la largeur nécessaire

Efface les marges par défaut

```
li {
  display: inline;
}
```

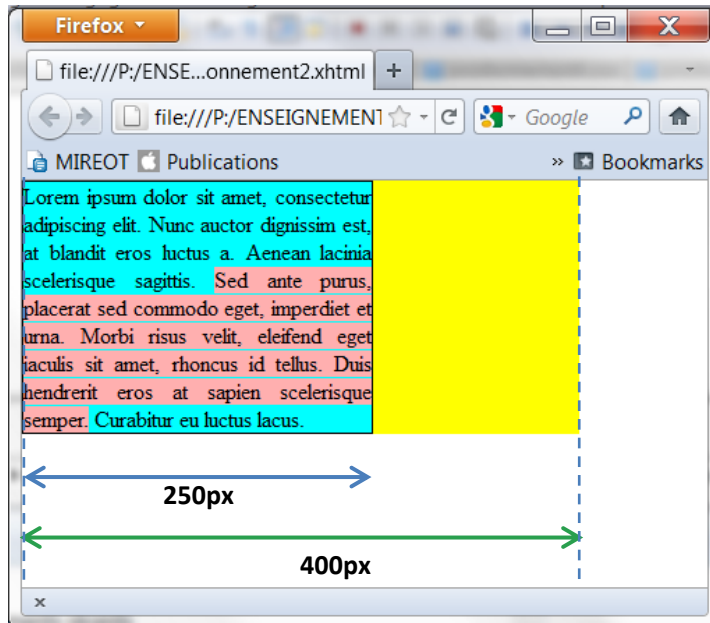
Try it yourself »

# Dimensionnement et marge interne

attribution de dimensions aux éléments `div1` et `p1`

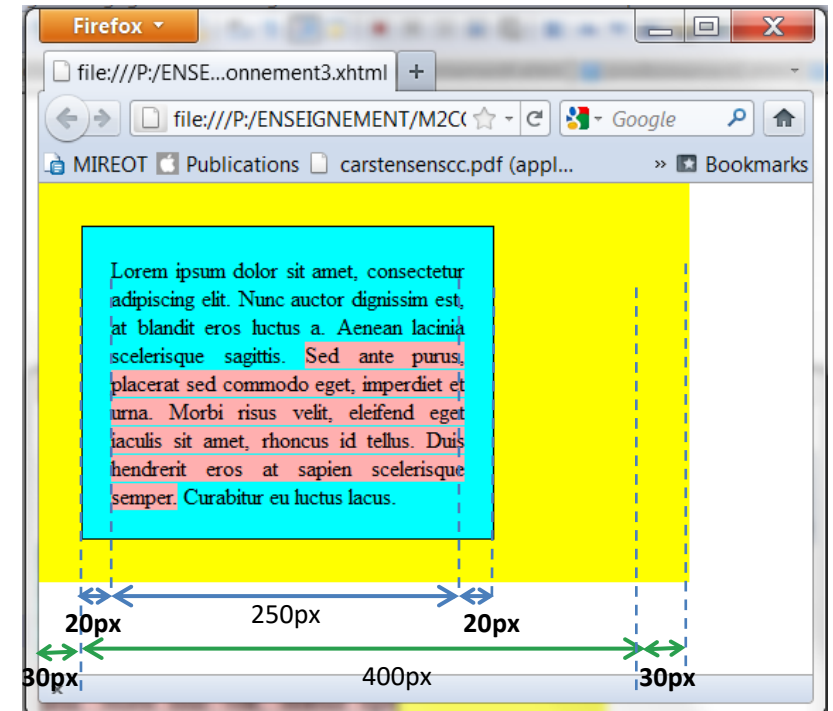
```
#div1 {  
  background: yellow;  
  width: 400px; ←  
}
```

```
#p1 {  
  background: cyan;  
  width: 250px; ←  
}
```



```
#div1 {  
  background: yellow;  
  width: 400px;  
  padding: 30px; ←  
}  
  
#p1 {  
  padding: 20px; ←  
  background: cyan;  
  width: 250px;  
}
```

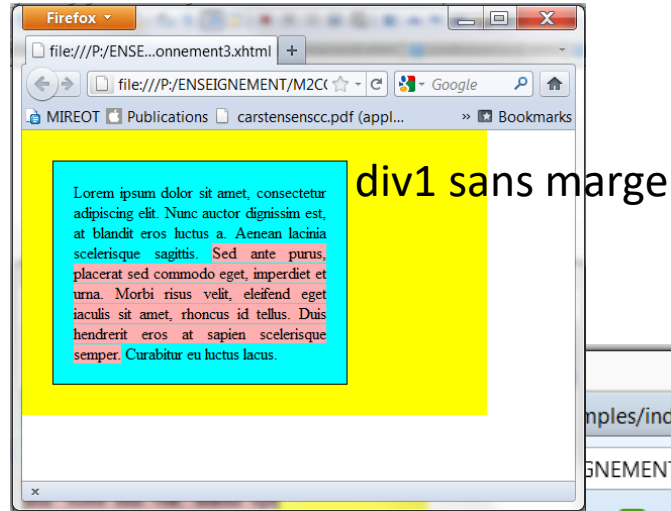
ajout de marges internes (*padding*) aux éléments `div1` et `p1`



# Définition d'une marge externe autour d'un bloc

ajout d'une marge externe (*margin*) a l'élément **div1**

```
#div1 {  
  background-color: yellow;  
  width: 400px;  
  padding: 30px;  
  margin: 20px; ←  
}  
  
#p1 {  
  padding: 20px;  
  background: cyan;  
  width: 250px;  
}
```

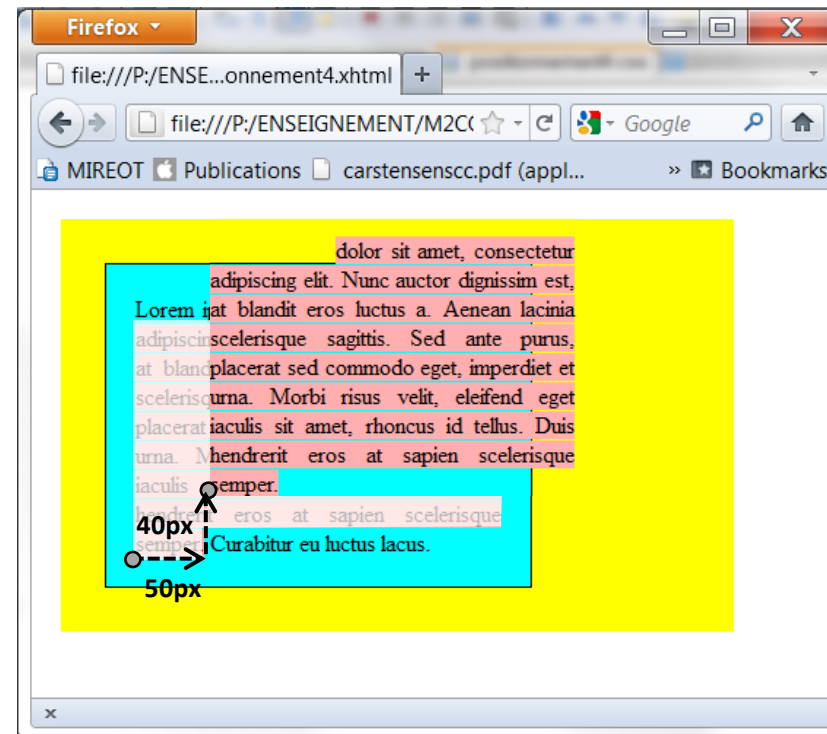




# Positionnement relatif d'un élément

- positionnement relatif
  - l'élément est décalé à l'aide des propriétés **top**, **right**, **left**, **bottom** par rapport à sa position normale dans le flux courant
    - l'élément peut prendre place au dessus de ses éléments frères.
  - n'affecte pas les boites qui l'entourent

```
#span1 {  
  background: #ffafaf;  
  position: relative; ←  
  left: 50px; ←  
  bottom: 40px; ←  
}
```

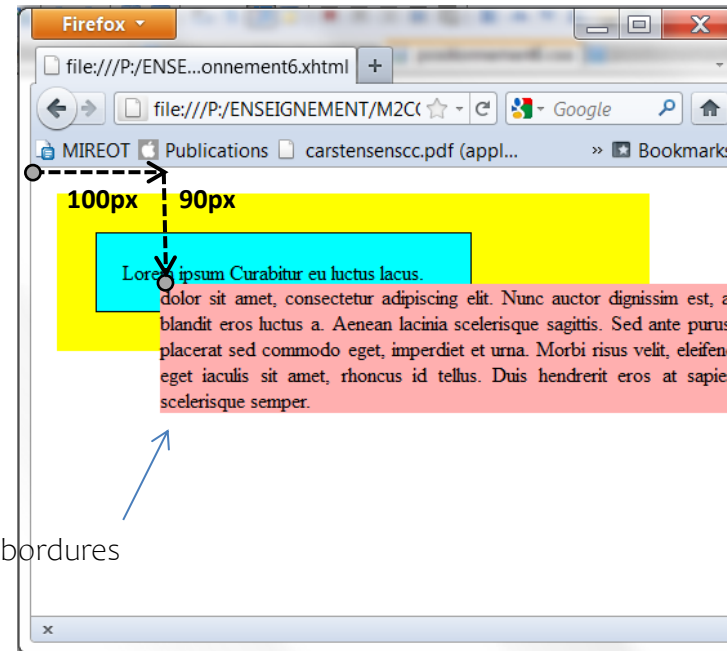


# Positionnement absolu

- positionnement absolu
  - sort un élément du flux
  - l'élément ne participe plus à la position de ses frères
  - positionnement à l'aide des propriétés **top**, **right**, **left**, **bottom**
    - expriment des décalages non plus par rapport à position théorique (positionnement relatif) mais par rapport à la position d'un bloc conteneur de référence
    - boîte conteneur de référence : le premier élément ancêtre positionné (relatif, absolu ou fixed) (élément <body> toujours considéré comme positionné)

```
#span1 {  
  background: #ffafaf;  
  position: absolute; ←  
  left: 100px; ←  
  top: 90px; ←  
}
```

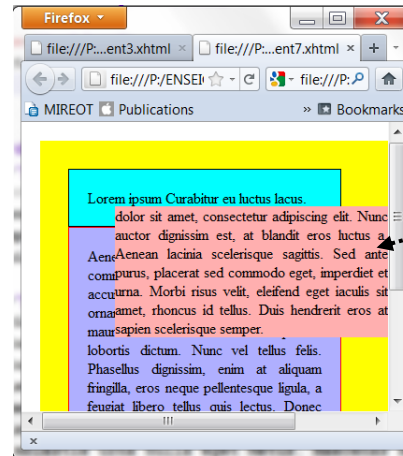
tout élément positionné en absolu est considéré de type bloc  
=> éléments en ligne peuvent ainsi recevoir des dimensions et bordures



# Positionnement fixe

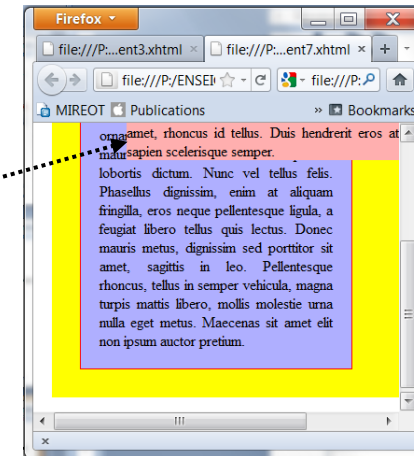
- positionnement fixe
  - cas particulier du positionnement absolu
  - l'élément reste fixe dans la page par rapport à la zone de visualisation (pas de scroll)

```
#span1 {  
  background: #ffafaf;  
  position: absolute;  
  left: 100px;  
  top: 90px;  
}
```

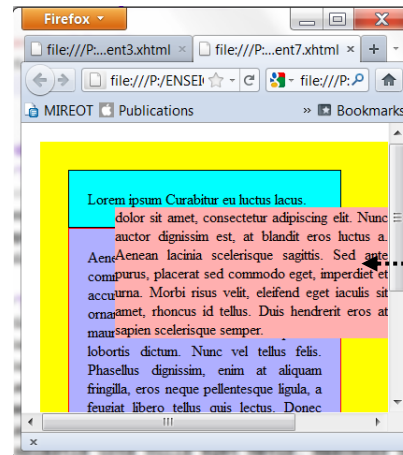


positionnement  
absolu

scrolling en liaison  
avec l'élément de  
référence

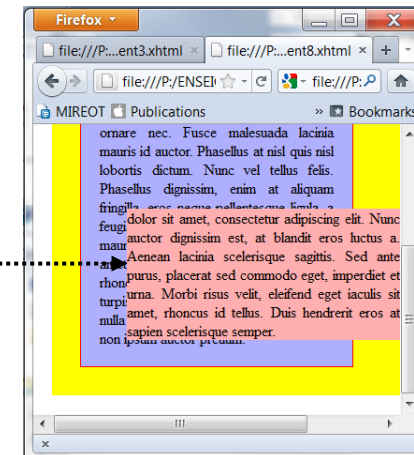


```
#span1 {  
  background: #ffafaf;  
  position: fixed; ←  
  left: 100px;  
  top: 90px;  
}
```



positionnement  
fixe

pas de scrolling

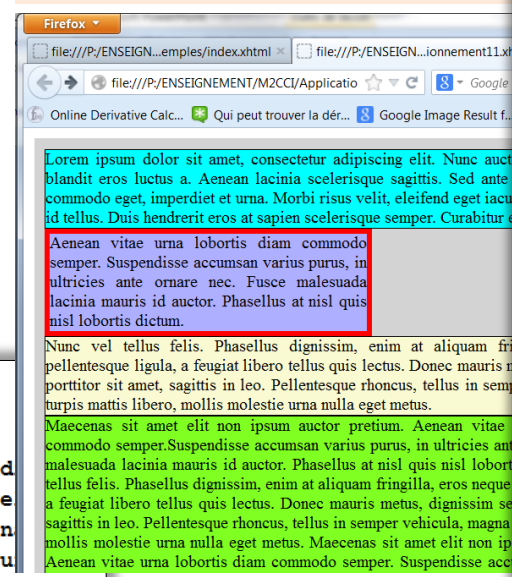


# Positionnement flottant

- positionnement flottant
  1. sort l'élément du flux
  2. l'élément est "poussé" à gauche (`float: left`) ou à droite (`float: right`) de son conteneur.
  3. les éléments qui le suivent dans le conteneur prennent place autour de lui.

```
<body>
  <div id="div1">
    <p id="p1">
      Lorem ipsum <span id="span1">dolor sit amet, consectetur ad
      dignissim est, at blandit eros luctus a. Aenean lacinia sce
      Sed ante purus, placerat sed commodo eget, imperdiet et urn
      Duis hendrerit eros at sapien scelerisque semper.</span> Cu
    </p>
    <p id="p2">
      Aenean vitae urna lobortis diam commo
      in ultricies ante ornare nec. Fusce :
      Phasellus at nisl quis nisl lobortis
    </p>
    <p id="p3">
      Nunc vel tellus felis. Phasellus d:
      neque pellentesque ligula, a feugiat
      sagittis in leo. Pellentesque rhoncu:
      mollis molestie urna nulla eget metu:
    </p>
    <p id="p4">
      Maecenas sit amet elit non ipsum auc:
      in ultricies ante ornare nec. Fusce :
      ....
      mollis molestie urna nulla eget metu:
    </p>
  </div>
</body>
```

positionnement normal (flux)



```
body {
  text-align: justify;
  padding: 0px;
  margin: 0px;
}
p {
  border-style: solid;
  border-width: thin;
  padding: 0px;
  margin: 0px;
}
#div1 {
  background-color: lightgrey;
  padding: 10px;
  margin: 10px;
}
#p1 {
  background: cyan;
}
#p2 {
  background: #afafff;
  border-color: red;
  border-width: 5px;
  width: 300px;
  float: right;
}
#p3 {
  background: #FAFAD2;
}
#p4 {
  background: #7FFF22;
}
```

positionnement flottant (right)



# Positionnement flottant

- possibilité de placer des blocs flottants côte à côte

```
<body>
  <div id="div1">
    <p id="p1"> #p1{
      background: cyan;
      width: 30%;
      float: left;
    }
    <p id="p2"> #p2 {
      background: #afafff;
      width: 30%;
      float: left;
    }
    <p id="p3"> #p3{
      background: #FAFAD2;
      width: 20%;
      float: right;
    }
    <p id="p4"> #p4{
      background: #7FFF22;
      width: 15%;
      float: left;
    }
    <p id="p5"> #p5{
      background: #FF9222;
      width: 15%;
      float: left;
    }
  </div>
  Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc auctor dignissim est, at blandit eros luctus a. Aenean lacinia scelerisque sagittis. Sed ante purus, placerat sed commodo eget, imperdiet et urna. Morbi risus velit, eleifend eget iaculis sit amet, rhoncus id tellus. Duis hendrerit eros at sapien scelerisque semper. Curabitur eu luctus lacus.
  Aenean vitae urna lobortis diam Maecenas sit amet Donec mauris metus, dignissim sed Nunc vel tellus felis.
  commodo semper. Suspendisse elit non ipsum porttitor sit amet, sagittis in leo Phasellus dignissim,
  accumsan varius purus, in ultricies auctor pretium Pellentesque rhoncus, tellus in semper enim at aliquam fringilla,
  ante ornare nec. Fusce malesuada Aenean vitae urna vehicula, magna turpis mattis libero, mollis eros neque pellentesque
  lacinia mauris id auctor. Phasellus at lobortis diam molestie urna nulla eget metus. Maecenas ligula, a feugiat libero
  commodo semper sit amet elit non ipsum auctor pretium tellus quis lectus. Donec mauris metus, dignissim sed porttitor sit amet, sagittis in leo.
  Suspendisse Aenean vitae urna lobortis diam commodum mauris metus, dignissim sed porttitor sit amet, sagittis in leo.
  accumsan varius semper. Suspendisse accumsan varius sed porttitor sit amet, sagittis in leo.
  purus, in ultricies purus, in ultricies ante ornare nec. Fusce sagittis in leo.
  ante ornare nec malesuada lacinia mauris id auctor Pellentesque rhoncus,
  Fusce malesuada Phasellus at nisl quis nisl lobortis dictum tellus in semper
  lacinia mauris id Nunc vel tellus felis. Phasellus dignissim, vehicula, magna turpis
  auctor. Phasellus enim at aliquam fringilla, eros neque mattis libero, mollis
  at nisl quis nisl pellentesque ligula, a feugiat libero tellus molestie urna nulla eget
  lobortis dictum quis lectus. Donec mauris metus, metus.
  Nunc vel tellus dignissim sed porttitor sit amet, sagittis in
  leo. Phasellus leo. Pellentesque rhoncus, tellus in semper vehicula, magna turpis
  dignissim, enim at mattis libero, mollis molestie urna nulla eget metus. Maecenas sit
  aliquam fringilla, amet elit non ipsum auctor pretium. Aenean vitae urna lobortis diam
  eros neque commodo semper. Suspendisse accumsan varius purus, in ultricies
  pellentesque ante ornare nec. Fusce malesuada lacinia mauris id auctor.
  ligula, a feugiat Phasellus at nisl quis nisl lobortis dictum. Nunc vel tellus felis.
  libero tellus quis Phasellus dignissim, enim at aliquam fringilla, eros neque
  pellentesque ligula, a feugiat libero tellus quis lectus. Donec mauris
  metus, dignissim sed porttitor sit amet, sagittis in leo. Pellentesque
  rhoncus, tellus in semper vehicula, magna turpis mattis libero, mollis molestie urna
  non ipsum auctor pretium.
```



# Positionnement flottant

- **clear** permet d'interdire le voisinage avec un élément flottant (à droite, à gauche ou des deux côtés).

```
<body>
  <div id="div1">
    <p id="p1"> #p1{
      background: cyan;
      width: 30%;
      float: left;
    }
    <p id="p2"> #p2 {
      background: #afafff;
      width: 30%;
      float: left;
    }
    <p id="p3"> #p3{
      background: #FAFAD2;
      width: 20%;
      float: right;
    }
    <p id="p4"> #p4{
      background: #7FFF22;
      width: 15%;
      float: left;
    }
    <p id="p5"> #p5{
      background: #FF9222;
      width: 15%;
      float: left;
      clear: right;
    }
  </div>

```

pas d'élément flottant à droite

© UGA-2024 Philippe GENOUD

# Positionnement flottant

- **clear** permet d'interdire le voisinage avec un élément flottant (à droite, à gauche ou des deux côtés).

The image shows a code editor on the left and a browser window on the right. The code editor contains the following HTML and CSS:

```
<body>
  <div id="div1">
    <p id="p1"> #p1{
      background: cyan;
    }
    <p id="p2"> #p2 {
      background: #afafff;
      width:30%;
      float:left;
    }
    <p id="p3"> #p3{
      background: #FAFAD2;
      float: right;
      width:20%;
    }
    <p id="p4"> #p4{
      background: #7FFF22;
      float:left;
      width:15%;
    }
    <p id="p5"> #p5{
      background: #FF9222;
      clear:both;
    }
  </div>
</body>
```

The browser window displays the rendered page. The text is arranged in columns. The first paragraph (p1) is cyan. The second (p2) is light blue and floats left. The third (p3) is light yellow and floats right. The fourth (p4) is light green and floats left. The fifth (p5) is orange and is positioned below the others, with the text 'pas d'élément flottant ni à droite ni à gauche' overlaid on it. The browser window also shows the URL 'http://net/Cours/Cours2 HTML-XHTML-CSS/Exemples/positionnement12-3:' and the Firefox logo.

# Positionnement flottant

- En utilisant les propriétés `float` et `clear` possibilité de définir des grilles de boites remplissant automatiquement la largeur de leur conteneur

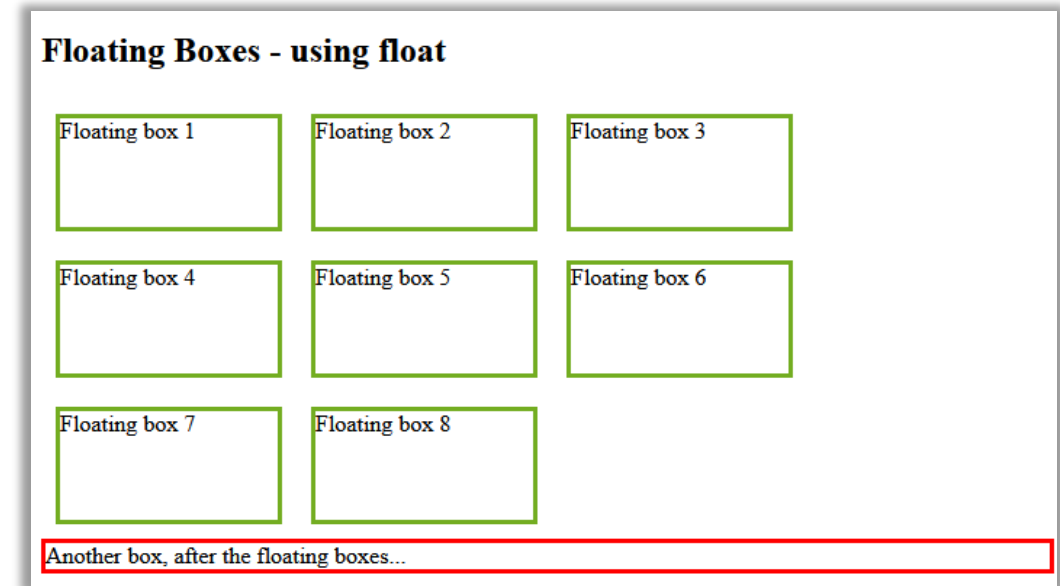
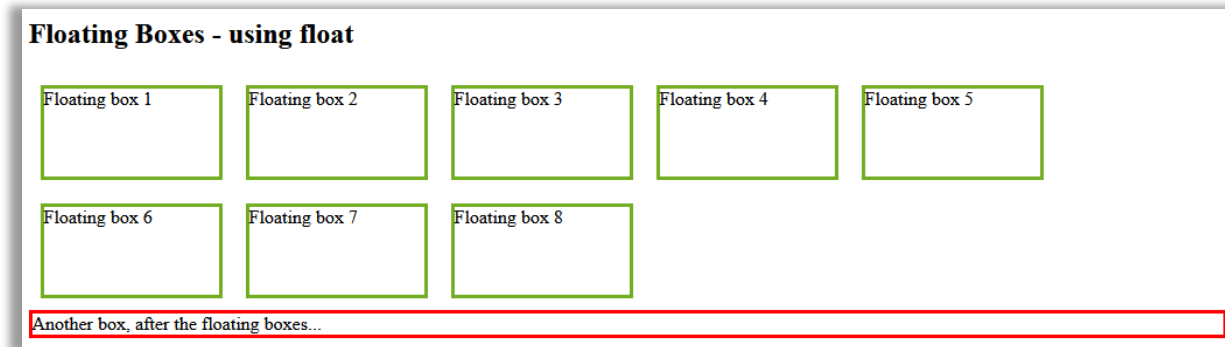
```
<h2>Floating Boxes - using float</h2>
```

```
<div class="floating-box">Floating box 1</div>
<div class="floating-box">Floating box 2</div>
<div class="floating-box">Floating box 3</div>
<div class="floating-box">Floating box 4</div>
<div class="floating-box">Floating box 5</div>
<div class="floating-box">Floating box 6</div>
<div class="floating-box">Floating box 7</div>
<div class="floating-box">Floating box 8</div>
```

```
<div class="after-box">Another box, after the floating boxes...</div>
```

```
.floating-box {
  float: left;
  width: 150px;
  height: 75px;
  margin: 10px;
  border: 3px solid #73AD21;
}

.after-box {
  clear: left;
  border: 3px solid red;
}
```



[https://www.w3schools.com/css/tryit.asp?filename=trycss\\_inline-block\\_old](https://www.w3schools.com/css/tryit.asp?filename=trycss_inline-block_old)



# Positionnement "inline-block"

- Possible d'obtenir facilement le même effet avec la propriété `display:inline-block`.
- Les éléments `inline-block` se comportent comme des éléments en ligne mais peuvent avoir une largeur et hauteur.

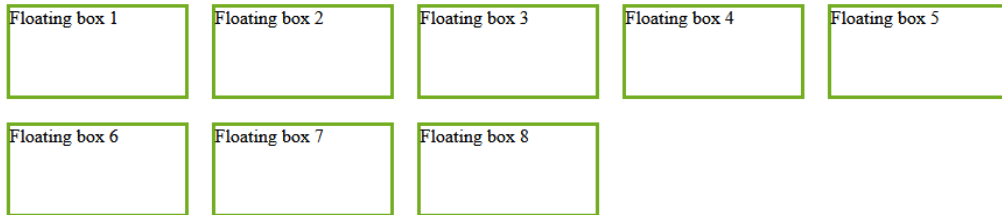
```
<h2>Floating Boxes - using float</h2>
```

```
<div class="floating-box">Floating box 1</div>  
<div class="floating-box">Floating box 2</div>  
<div class="floating-box">Floating box 3</div>  
<div class="floating-box">Floating box 4</div>  
<div class="floating-box">Floating box 5</div>  
<div class="floating-box">Floating box 6</div>  
<div class="floating-box">Floating box 7</div>  
<div class="floating-box">Floating box 8</div>
```

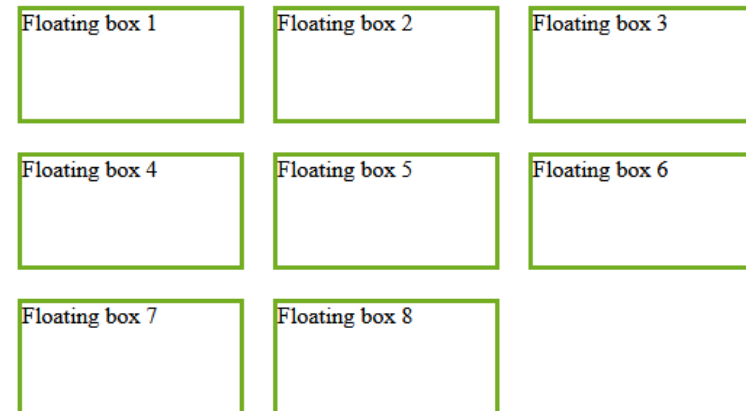
```
<div class="after-box">Another box, after the floating boxes...</div>
```

```
.floating-box {  
float: left; display: inline-block;  
width: 150px;  
height: 75px;  
margin: 10px;  
border: 3px solid #73AD21;  
}  
  
.after-box {  
clear: left;  
border: 3px solid red;  
}
```

## Floating Boxes - using inline-block



## Floating Boxes - using inline-block



[https://www.w3schools.com/css/tryit.asp?filename=trycss\\_inline-block](https://www.w3schools.com/css/tryit.asp?filename=trycss_inline-block)

# Positionnement "inline-block"

```
body {  
  font-family:Arial, Helvetica, sans-serif;  
}  
  
article {  
  background-color: rgb(236, 206, 169);  
  padding-left: 10px;  
  padding-right: 10px;  
  text-align: justify;  
}  
  
p {  
  text-align: justify;  
}  
  
figure {  
  text-align: center;  
}  
  
img {  
  max-width: 100%;  
  height: auto;  
}
```

```
<body>  
  <div class="container">  
    <h1>Tempore excepturi qui molestias.</h1>  
    <p>Lorem ipsum ..</p>  
    <article>  
      <h2>Les Ours</h2>  
      <p>  
        Lorem ipsum ... magni?  
      </p>  
      <figure>  
          
        <figcaption>Un ours en Alaska</figcaption>  
      </figure>  
    </article>  
    <article>  
      <h2>Les Orques</h2>  
      <p>  
        Lorem ...  
      </p>  
      <figure>  
          
        <figcaption>Un orque à Haida Gwaii</figcaption>  
      </figure>  
    </article>  
    <article>  
      <h2>Les Caribous</h2>  
      ...  
    </article>  
    ...  
    <p>  
      Lorem ipsum ...  
    </p>  
  </div>  
</body>
```

positionnement en flux



## Tempore excepturi qui molestias.

Lorem ipsum dolor sit amet consectetur adipisicing elit. Recusandae deleniti odit est suscipit esse, eos molestias, harum inventore corrupti non laborum temporibus quis fugit rerum minima, ea quia ut hic incidunt? Optio qui facilis al Rerum pariatur, laudantium magnam esse molestias ex!

### Les Ours

Lorem ipsum dolor sit amet consectetur adipisicing elit. Ut facere quae eos fugiat sunt? Hic, magni?



Un ours en Alaska

### Les Orques

Lorem ipsum, dolor sit amet consectetur adipisicing elit. Quaerat est sunt voluptas necessitatibus. Aliquam enim architecto quisquam accusantium commodi vitae!

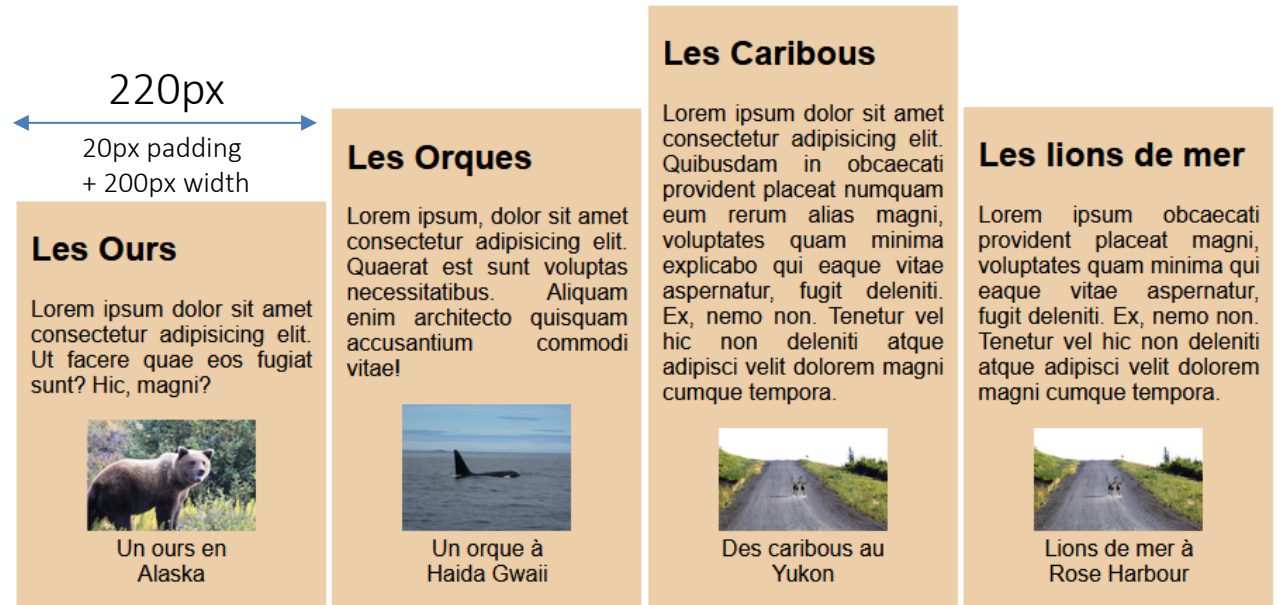


# Positionnement "inline-block"

```
body {  
  font-family: Arial, Helvetica, sans-serif;  
}  
  
p {  
  text-align: justify;  
}  
  
figure {  
  text-align: center;  
}  
  
article {  
  display: inline-block;  
  background-color: #c8e6c9;  
  padding-left: 10px;  
  padding-right: 10px;  
  width: 200px;  
}  
  
img {  
  max-width: 100%;  
  height: auto;  
}
```

## Tempore excepturi qui molestias.

Lorem ipsum dolor sit amet consectetur adipisicing elit. Recusandae deleniti odit est suscipit esse, eos molestias, harum inventore corrupti non laborum temporibus quis fugit rerum minima, ea quia ut hic incidunt? Optio qui facilis al Rerum pariatur, laudantium magnam esse molestias ex!







Lorem ipsum dolor sit amet consectetur adipisicing elit. Maxime similique modi eum velit sed rerum voluptatem debitis hic at nobis temporibus esse illum culpa consequuntur animi explicabo delectus numquam veniam consequatur vero necessitatibus laboriosam, adipisci harum eius. Ex ut in nam aliquam minima id autem.

# Positionnement "inline-block"

```
body {  
  font-family: Arial, Helvetica, sans-serif  
}  
  
p {  
  text-align: justify;  
}  
  
figure {  
  text-align: center;  
}  
  
article {  
  display: inline-block;  
  background-color: #E6B89C; /* rgb(236, 206, 169) */  
  padding-left: 10px;  
  padding-right: 10px;  
  width: 200px;  
  vertical-align: top;  
}  
  
img {  
  max-width: 100%;  
  height: auto;  
}
```

## Tempore excepturi qui molestias.

Lorem ipsum dolor sit amet consectetur adipisicing elit. Recusandae deleniti odit est suscipit esse, eos molestias, harum inventore corrupti non laborum temporibus quis fugit rerum minima, ea quia ut hic incidunt? Optio qui facilis a! Rerum pariatur, laudantium magnam esse molestias ex!

<h3>Les Ours</h3> <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Ut facere quae eos fugiat sunt? Hic, magni?</p>  <p>Un ours en Alaska</p>	<h3>Les Orques</h3> <p>Lorem ipsum, dolor sit amet consectetur adipisicing elit. Quaerat est sunt voluptas necessitatibus. Aliquam enim architecto quisquam accusantium commodi vitae!</p>  <p>Un orque à Haida Gwaii</p>	<h3>Les Caribous</h3> <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Quibusdam in obcaecati provident placeat numquam eum rerum alias magni, voluptates quam minima explicabo qui eaque vitae aspernatur, fugit deleniti. Ex, nemo non. Tenetur vel hic non deleniti atque adipisci velit dolorem magni cumque tempora.</p>  <p>Des caribous au Yukon</p>	<h3>Les lions de mer</h3> <p>Lorem ipsum obcaecati provident placeat magni, voluptates quam minima qui eaque vitae aspernatur, fugit deleniti. Ex, nemo non. Tenetur vel hic non deleniti atque adipisci velit dolorem magni cumque tempora.</p>  <p>Lions de mer à Rose Harbour</p>
--	--	---	---

Lorem ipsum dolor sit amet consectetur adipisicing elit. Maxime similique modi eum velit sed rerum voluptatem debitis hic at nobis temporibus esse illum culpa consequuntur animi explicabo delectus numquam veniam consequatur vero necessitatibus laboriosam, adipisci harum eius. Ex ut in nam aliquam minima id autem.

# Positionnement "inline-block"

- problème des espaces

```
body {  
  font-family:Arial, Helvetica, sans-serif;  
}  
  
p {  
  text-align:justify;  
}  
  
figure {  
  text-align:center;  
}  
  
article {  
  display:inline-block;  
  background-color: rgb(236, 206, 169);  
  padding-left: 10px;  
  padding-right: 10px;  
  width: 200px;  
  vertical-align: top;  
}  
  
img {  
  max-width: 100%;  
  height: auto;  
}
```

```
.container {  
  width:880px;  
  margin: auto;  
  background-color: lightgoldenrodyellow;  
}
```

solution , voir article :

<http://www.alsacreations.com/astuce/lire/1432-display-inline-block-espacesindesirables.html>

## Tempore excepturi qui molestias.

Lorem ipsum dolor sit amet consectetur adipisicing elit. Recusandae deleniti odit est suscipit esse, eos molestias, harum inventore corrupti non laborum temporibus quis fugit rerum minima, ea quia ut hic incidunt? Optio qui facilis al Rerum pariatur, laudantium magnam esse molestias ex!

### Les Ours

Lorem ipsum dolor sit amet consectetur adipisicing elit. Ut facere quae eos fugiat sunt? Hic, magni?



Un ours en Alaska

### Les Orques

Lorem ipsum, dolor sit amet consectetur adipisicing elit. Quaerat est sunt voluptas necessitatibus. Aliquam enim architecto quisquam accusantium commodi vitae!



Un orque à Haida Gwaii

### Les Caribous

Lorem ipsum dolor sit amet consectetur adipisicing elit. Quibusdam in obcaecati provident placeat numquam eum rerum alias magni, voluptates quam minima explicabo qui eaque vitae aspernatur, fugit deleniti. Ex, nemo non. Tenetur vel hic non deleniti atque adipisci velit dolorem magni cumque tempora.



Des caribous au Yukon

### Les lions de mer

Lorem ipsum obcaecati provident placeat magni, voluptates quam minima qui eaque vitae aspernatur, fugit deleniti. Ex, nemo non. Tenetur vel hic non deleniti atque adipisci velit dolorem magni cumque tempora.



Lions de mer à Rose Harbour

Il existe un espace entre les éléments de type inline-block

Lorem ipsum dolor sit amet consectetur adipisicing elit. Maxime similique modi eum velit sed rerum voluptatem debitis hic at nobis temporibus esse illum culpa consequuntur animi explicabo delectus numquam veniam consequatur vero necessitatibus laboriosam, adipisci harum eius. Ex ut in nam aliquam minima id autem.



# CSS

## Positionnement des éléments positionnement flottant

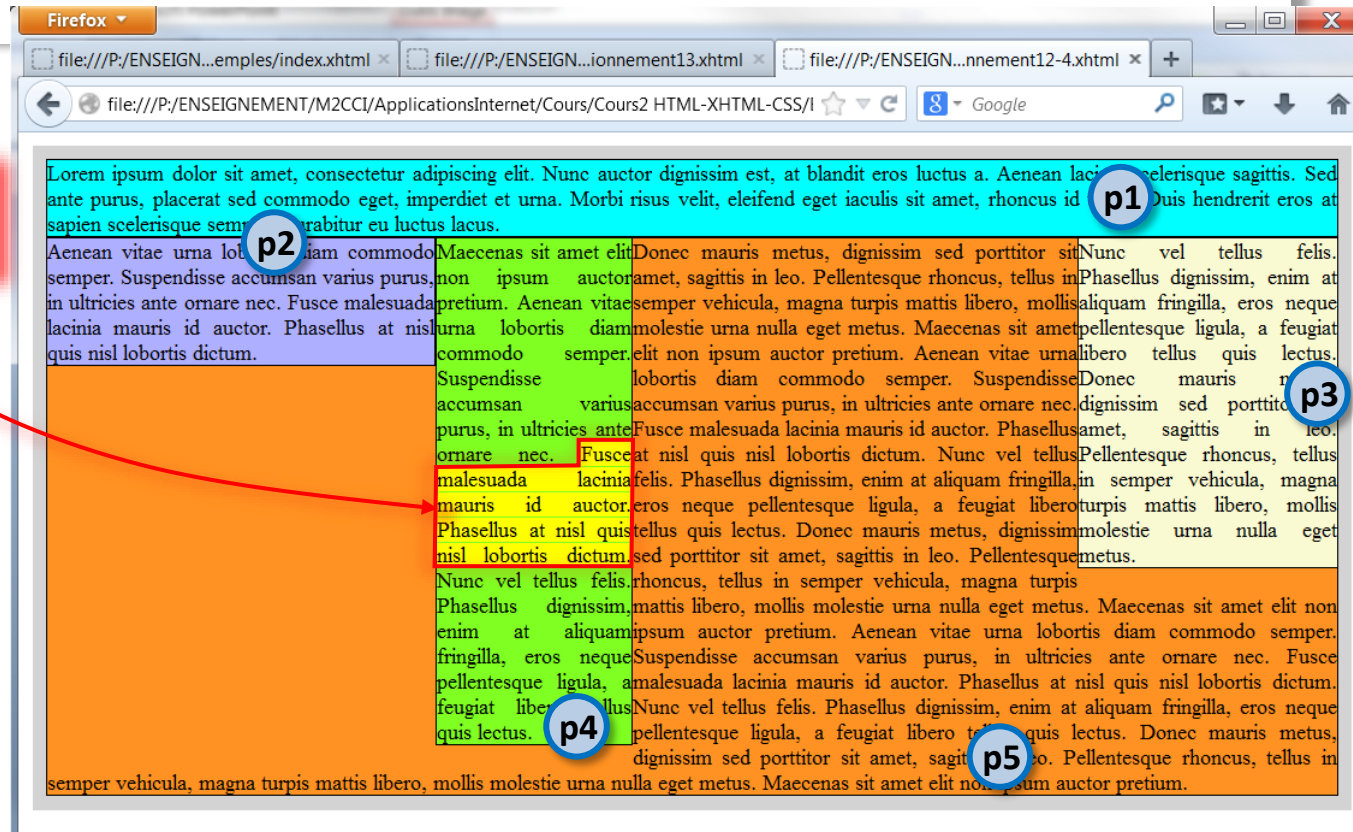
- possibilité de faire flotter des éléments en ligne.

```
<p id="p4">
  Maecenas sit amet elit non ipsum auctor pretium. Aenean vitae urna lobortis diam commodo semper. Sus
  in ultricies ante ornare nec. <span id="span2">Fusce malesuada lacinia mauris id auctor.
  Phasellus at nisl quis nisl lobortis dictum.</span> Nunc vel tellus felis.
  Phasellus dignissim, enim at aliquam fringilla, eros neque pellentesque ligula,
  a feugiat libero tellus quis lectus.
</p>
```

```
#span2 {
  background: yellow;
}
```

+

```
font-size: 50%;
float: left;
width: 20%;
margin: 5px;
padding: 5px;
```



## Positionnement des éléments positionnement flottant

- possibilité de faire flotter des éléments en ligne.

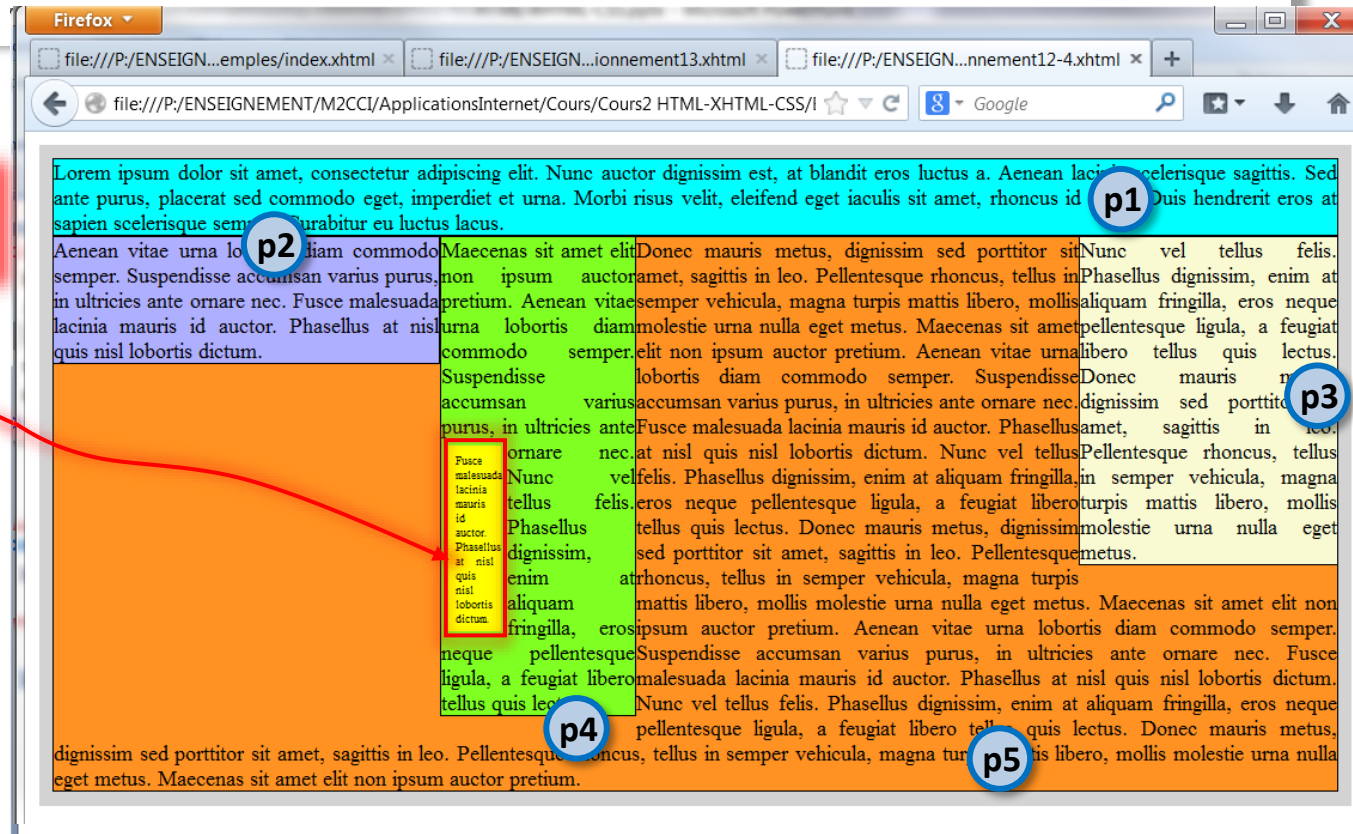
```
<p id="p4">
  Maecenas sit amet elit non ipsum auctor pretium. Aenean vitae urna lobortis diam commodo semper. Sus
  in ultricies ante ornare nec. <span id="span2">Fusce malesuada lacinia mauris id auctor.
  Phasellus at nisl quis nisl lobortis dictum.</span> Nunc vel tellus felis.
  Phasellus dignissim, enim at aliquam fringilla, eros neque pellentesque ligula,
  a feugiat libero tellus quis lectus.
</p>
```

```
#span2 {
  background: yellow;
}

+

font-size: 50%;
float: left;
width: 20%;
margin: 5px;
padding: 5px;
```

- tout élément flottant est considéré de type bloc
- les éléments en ligne peuvent alors recevoir bordures et dimensions

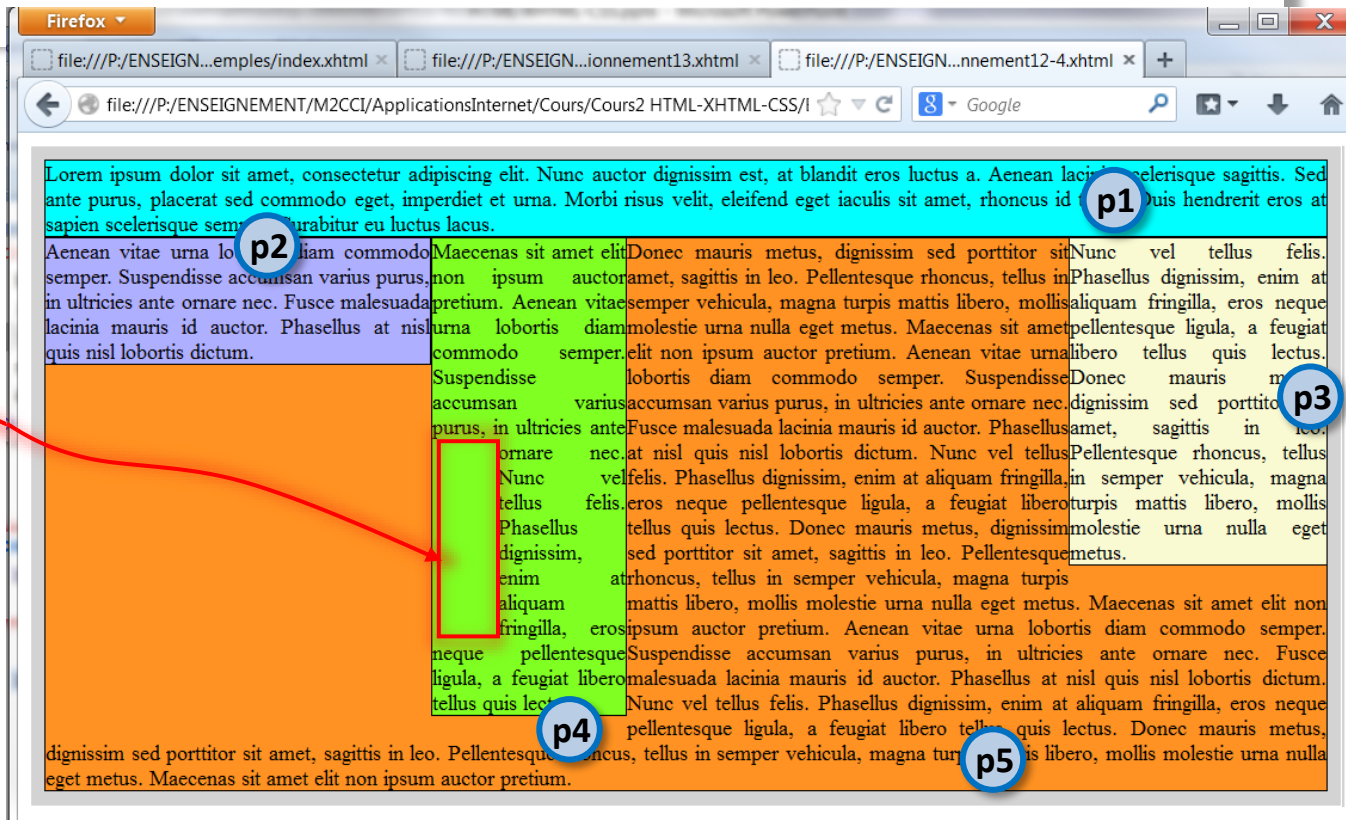


- possibilité de rendre invisible des éléments

```
<p id="p4">
  Maecenas sit amet elit non ipsum auctor pretium. Aenean vitae urna lobortis diam commodo semper. Sus
  in ultricies ante ornare nec. <span id="span2">Fusce malesuada lacinia mauris id auctor.
  Phasellus at nisl quis nisl lobortis dictum.</span> Nunc vel tellus felis.
  Phasellus dignissim, enim at aliquam fringilla, eros neque pellentesque ligula,
  a feugiat libero tellus quis lectus.
</p>
```

```
#span2 {
  background: yellow;
  font-size: 50%;
  float: left;
  width: 20%;
  margin: 5px;
  padding: 5px;
  visibility: hidden;
}
```

- visibility: hidden** masque l'élément mais réserve sa position et ses dimensions. L'élément occupe de l'espace sur la page.
- autres valeurs : **visible**, **collapse** (pour tables), **inherit**



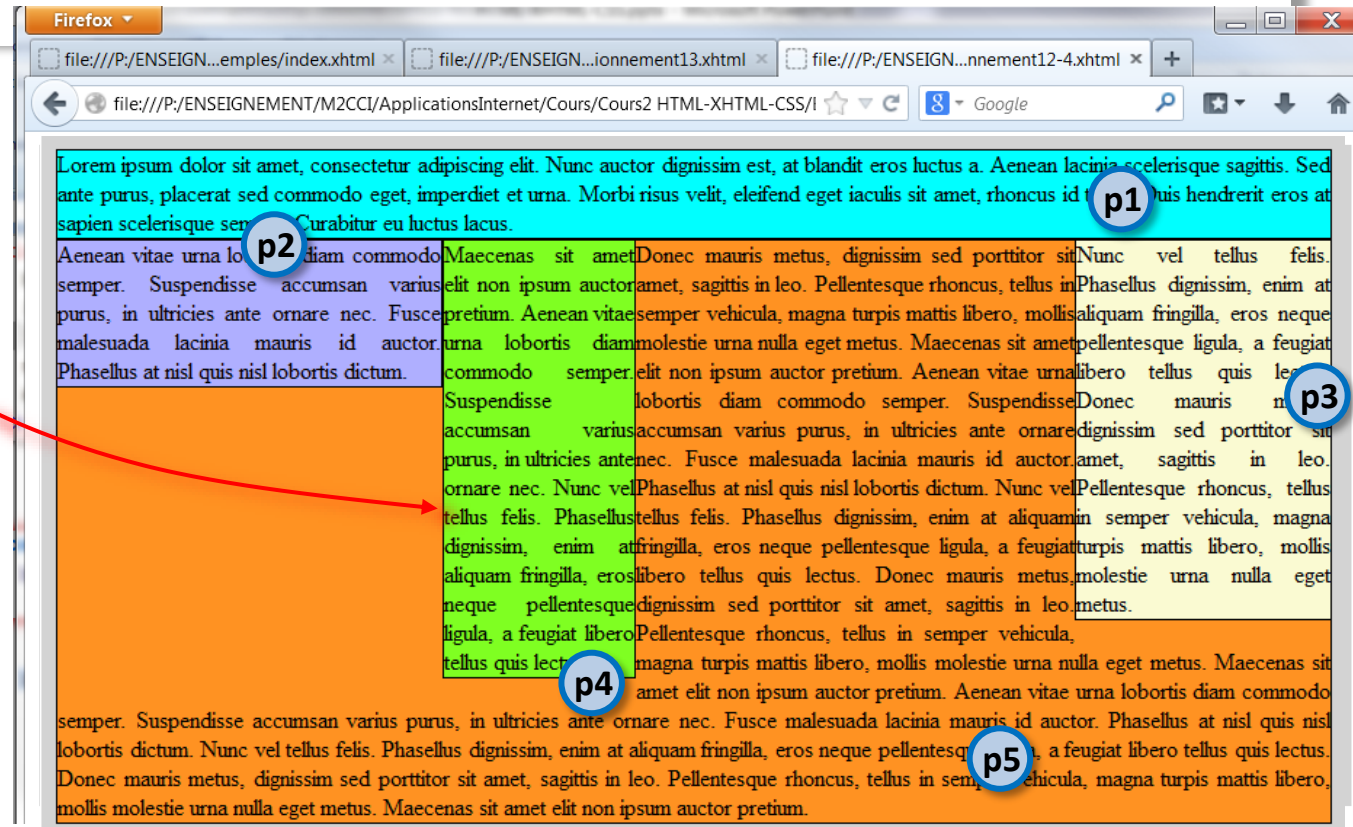


- possibilité de retirer un élément de l'affichage

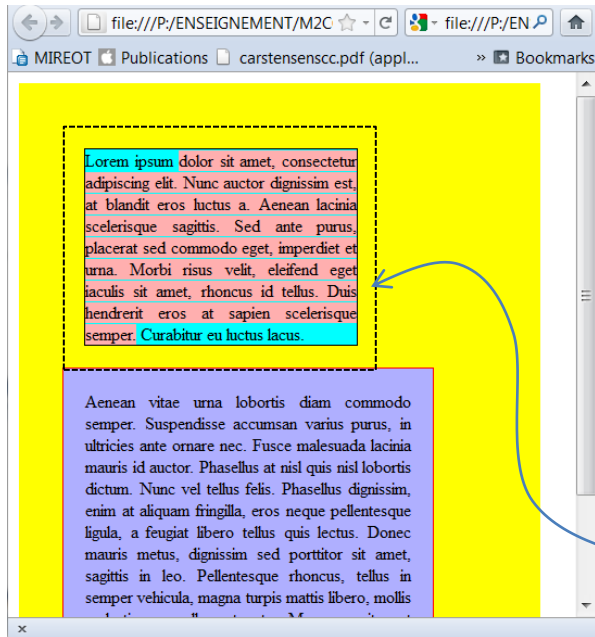
```
<p id="p4">
  Maecenas sit amet elit non ipsum auctor pretium. Aenean vitae urna lobortis diam commodo semper. Sus
  in ultricies ante ornare nec. <span id="span2">Fusce malesuada lacinia mauris id auctor.
  Phasellus at nisl quis nisl lobortis dictum.</span> Nunc vel tellus felis.
  Phasellus dignissim, enim at aliquam fringilla, eros neque pellentesque ligula,
  a feugiat libero tellus quis lectus.
</p>
```

```
#span2 {
  background: yellow;
  font-size: 50%;
  float: left;
  width: 20%;
  margin: 5px;
  padding: 5px;
  display: none;
}
```

- display:none** l'élément n'est plus du tout affiché. Tout se passe comme si il n'existait pas



- positionnement flottant
  - autre exemple.



positionnement en flux

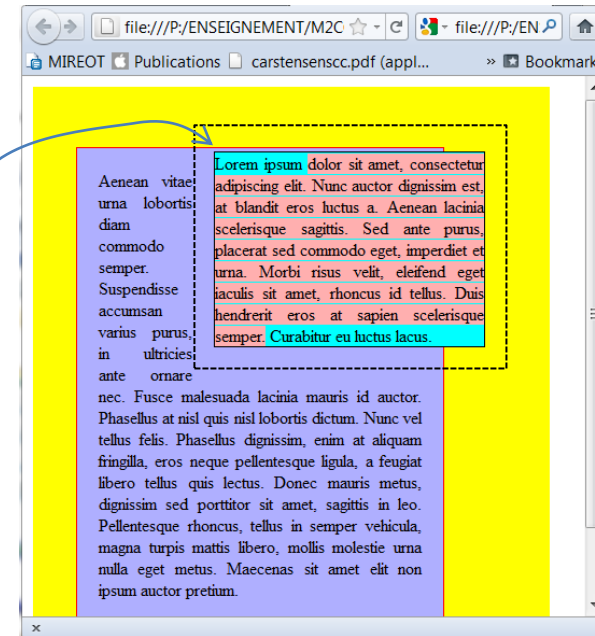
```
p {
  border-style: solid;
  border-width: thin;
  margin: 0px;
}

#div1 {
  background-color: yellow;
  width: 400px;
  padding: 40px;
  margin: 10px;
}

#p1 {
  background: cyan;
  width: 250px;
  margin: 20px;
}

#p2 {
  background: #afafff;
  border-color: red;
  padding: 20px;
  width: 300px;
}
```

`float: right;`



positionnement en flottant

# Positionnement 'Flexbox'

- positionnement flottant complexe à maîtriser
- Boîtes flexibles (flex boxes ou flexs) un nouveau mode de positionnement (layout) des éléments introduits avec CSS3.
  - amélioration par rapport au modèle de bloc
  - comportement plus prévisible pour prise en compte de tailles différentes d'affichage.

**TABLE OF CONTENTS**

- 1 Introduction
  - 1.1 Overview
  - 1.2 Module interactions
- 2 Flex Layout Box Model and Terminology
- 3 Flex Containers: the 'flex' and 'inline-flex' 'display' values
- 4 Flex Items
  - 4.1 Absolutely-Positioned Flex Children
  - 4.2 Flex Item Margins and Paddings
  - 4.3 Flex Item Z-Ordering
  - 4.4 Collapsed Items
  - 4.5 Automatic Minimum Size of Flex Items
- 5 Ordering and Orientation
  - 5.1 Flex Flow Direction: the 'flex-direction' property
  - 5.2 Flex Line Wrapping: the 'flex-wrap' property
  - 5.3 Flex Direction and Wrap: the 'flex-flow' shorthand
  - 5.4 Display Order: the 'order' property
    - 5.4.1 Reordering and Accessibility
- 6 Flex Lines
- 7 Flexibility
  - 7.1 The 'flex' Shorthand
    - 7.1.1 Basic Values of 'flex'
  - 7.2 Components of Flexibility
    - 7.2.1 The 'flex-grow' property
    - 7.2.2 The 'flex-shrink' property

**CSS Flexible Box Layout Module Level 1**  
W3C Candidate Recommendation, 19 November 2018

**This version:**  
<https://www.w3.org/TR/2018/CR-css-flexbox-1-20181119/>

**Latest published version:**  
<https://www.w3.org/TR/css-flexbox-1/>

**Editor's Draft:**  
<https://drafts.csswg.org/css-flexbox-1/>

**Previous Versions:**  
<https://www.w3.org/TR/2018/CR-css-flexbox-1-20181108/>  
<https://www.w3.org/TR/2017/CR-css-flexbox-1-20171019/>  
<https://www.w3.org/TR/2016/CR-css-flexbox-1-20160526/>  
<https://www.w3.org/TR/2016/CR-css-flexbox-1-20160301/>  
<https://www.w3.org/TR/2015/WD-css-flexbox-1-20150514/>  
<https://www.w3.org/TR/2014/WD-css-flexbox-1-20140925/>  
<https://www.w3.org/TR/2014/WD-css-flexbox-1-20140325/>  
<https://www.w3.org/TR/2012/CR-css3-flexbox-20120918/>  
<https://www.w3.org/TR/2012/WD-css3-flexbox-20120612/>  
<https://www.w3.org/TR/2012/WD-css3-flexbox-20120322/>  
<https://www.w3.org/TR/2011/WD-css3-flexbox-20111129/>  
<https://www.w3.org/TR/2011/WD-css3-flexbox-20110322/>  
<https://www.w3.org/TR/2009/WD-css3-flexbox-20090723/>

**Test Suite:**  
[http://test.csswg.org/suites/css-flexbox-1\\_dev/nightly-unstable/](http://test.csswg.org/suites/css-flexbox-1_dev/nightly-unstable/)

**Editors:**  
[Tab Atkins Jr.](#) (Google)  
[Elika J. Etemad / fantasai](#) (Invited Expert)  
[Rossen Atanassov](#) (Microsoft)

**Former Editors:**  
[Alex Mogilevsky](#) (Microsoft Corporation)

<https://www.w3.org/TR/css-flexbox-1/>

une norme récente mais largement anticipée par les navigateurs modernes

# Positionnement 'Flexbox'

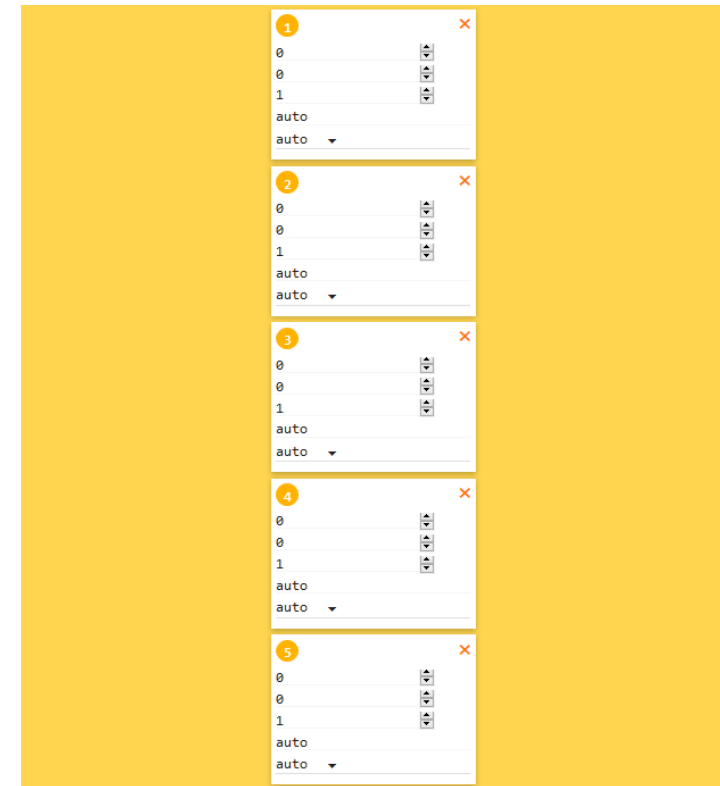
- Modèle de boîte flexible – flex box
- 4 caractéristiques principales
  - **Distribution** des éléments horizontale ou verticale, avec passage à la ligne autorisé ou non,
  - **Alignements** et centrages horizontaux et verticaux, justifiés, répartis,
  - **Réorganisation** des éléments indépendamment de l'ordre du flux (DOM),
  - **Gestion des espaces disponibles** (fluidité).

## Parent Flex Properties – flex container

flex-direction	flex-wrap	justify-content	align-items	align-content
<input type="radio"/> row	<input checked="" type="radio"/> nowrap	<input checked="" type="radio"/> flex-start	<input type="radio"/> stretch	<input checked="" type="radio"/> stretch
<input type="radio"/> row-reverse	<input type="radio"/> wrap	<input type="radio"/> flex-end	<input type="radio"/> flex-start	<input type="radio"/> flex-start
<input checked="" type="radio"/> column	<input type="radio"/> wrap-reverse	<input type="radio"/> center	<input type="radio"/> flex-end	<input type="radio"/> flex-end
<input type="radio"/> column-reverse		<input type="radio"/> space-between	<input checked="" type="radio"/> center	<input type="radio"/> center
		<input type="radio"/> space-around	<input type="radio"/> baseline	<input type="radio"/> space-between
				<input type="radio"/> space-around

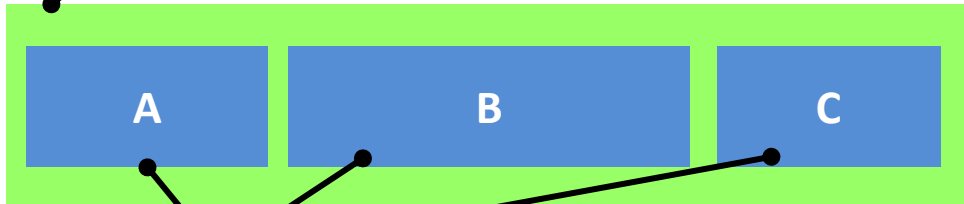
\* The default properties are highlighted

<https://codepen.io/justd/full/yydezN/>



# Positionnement 'Flexbox'

**flex container** : n'importe quel élément HTML doté de la déclaration `display: flex;` ou `display: inline-flex;`



**flex items**: enfants d'un flex container.

le type "flex-item" est implicite, inutile de déclarer quoi que ce soit

Éléments "flex-item" ne sont plus considérés comme des éléments classiques ("bloc" ou "inline") mais sont par défaut distribués horizontalement dans leur conteneur flex.

→ les propriétés de positionnement "classiques" (`display*`, `float` ...) sont sans effet sur les éléments flex-items.

## HTML

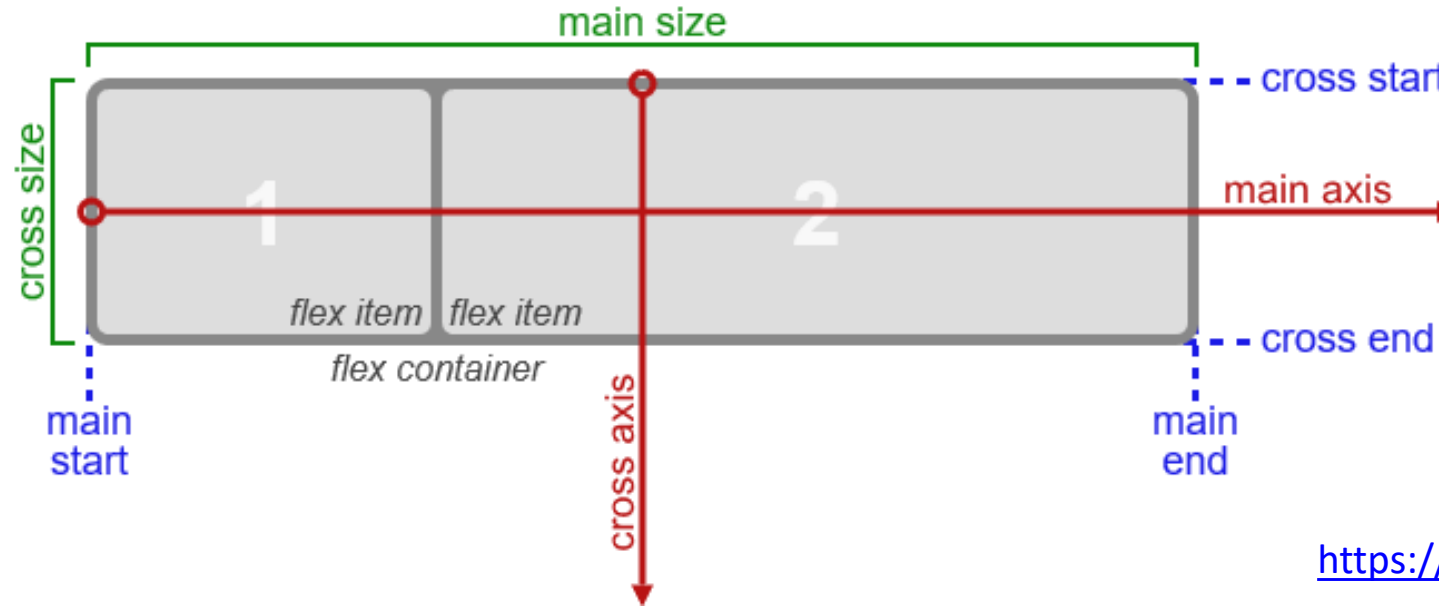
```
<div class="flex-container">
  <div id="div-1">A</div>
  <div id="div-2">B</div>
  <div id="div-3">C</div>
</div>
```

## \* CSS

```
.flex-container {
  display: flex;
  background-color: rgb(153,255,102);
}
.flex-container > div {
  background-color: rgb(85,142,213);
  box-sizing: border-box;
  border: solid red 2px;
  padding: 15px;
  margin: 10px;
  text-align: center;
  color: white;
  font-size: 30px;
}
#div-1, #div-3 {
  width: 25%;
}
#div-2 {
  width: 50%;
}
```

\* mis à part `display: none;`

# Positionnement 'Flexbox'



<https://www.w3.org/TR/css-flexbox-1/>

## main axis main dimension

The **main axis** of a flex container is the primary axis along which [flex items](#) are laid out. It extends in the **main dimension**.

## main-start main-end

The [flex items](#) are placed within the container starting on the **main-start** side and going toward the **main-end** side.

## main size main size property

The width or height of a [flex container](#) or [flex item](#), whichever is in the [main dimension](#), is that box's **main size**. Its **main size property** is thus either its `'width'` or `'height'` property, whichever is in the [main dimension](#). Similarly, its **min** and **max main size properties** are its `'min-width'`/`'max-width'` or `'min-height'`/`'max-height'` properties, whichever is in the [main dimension](#), and determine its **min/max main size**.

## cross axis cross dimension

The axis perpendicular to the [main axis](#) is called the **cross axis**. It extends in the **cross dimension**.

## cross-start cross-end

[Flex lines](#) are filled with items and placed into the container starting on the **cross-start** side of the flex container and going toward the **cross-end** side.

## cross size cross size property

The width or height of a [flex container](#) or [flex item](#), whichever is in the [cross dimension](#), is that box's **cross size**. Its **cross size property** is thus either its `'width'` or `'height'` property, whichever is in the [cross dimension](#). Similarly, its **min** and **max cross size properties** are its `'min-width'`/`'max-width'` or `'min-height'`/`'max-height'` properties, whichever is in the [cross dimension](#), and determine its **min/max cross size**.

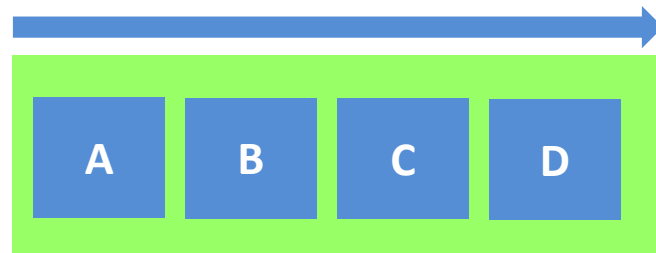
# Positionnement flexbox

- Les propriétés d'un conteneur flex sont :
  - **flex-direction** : définit la direction de l'axe principal (main-axis) selon laquelle les flex-items sont distribués
  - **flex-wrap** :
  - **flex-flow** : raccourci pour spécifier simultanément les propriétés **flex-direction** et **flex-wrap**
  - **justify-content** : définit l'alignement selon l'axe principal
  - **align-items** : définit l'alignement selon l'axe secondaire (perpendiculaire à l'axe principal)
  - **align-content** :

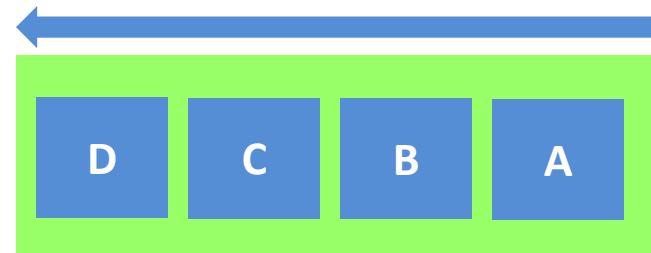
# Positionnement 'Flexbox'

**flex-direction:** (propriété de flex-container)

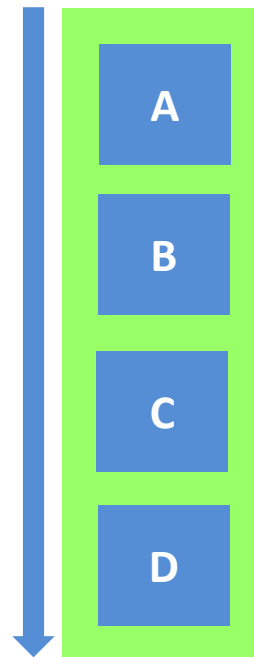
définit la direction (axe principal ou main-axis) selon laquelle les flex-items sont distribués



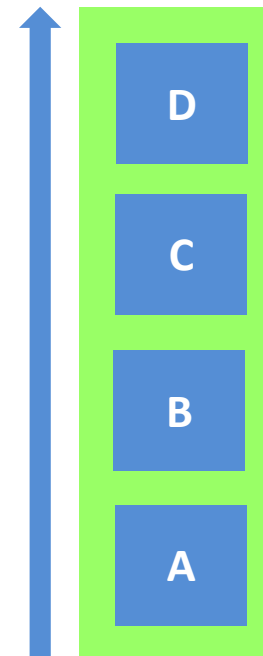
row



row-reverse



column



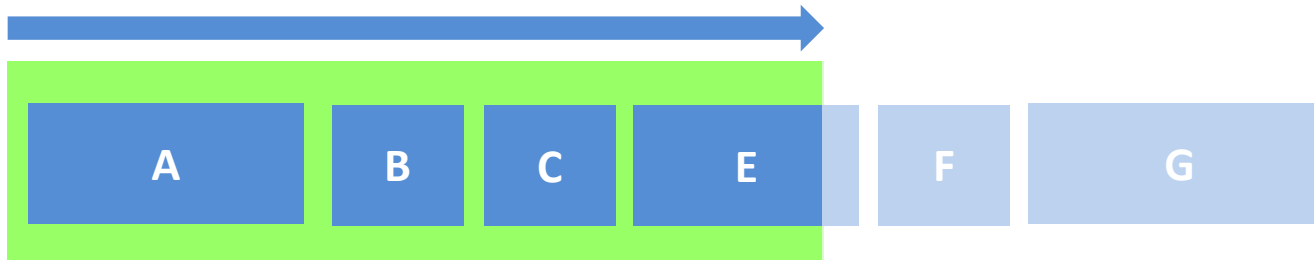
column-reverse



# Positionnement 'Flexbox'

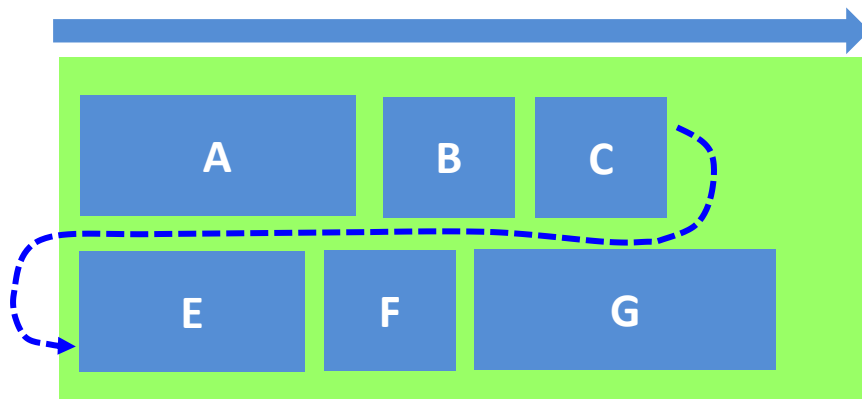
**flex-wrap** (propriété de flex-container)

définit si le contenu sera distribué sur une seule ligne (ou colonne selon l'axe principal) ou sur plusieurs lignes



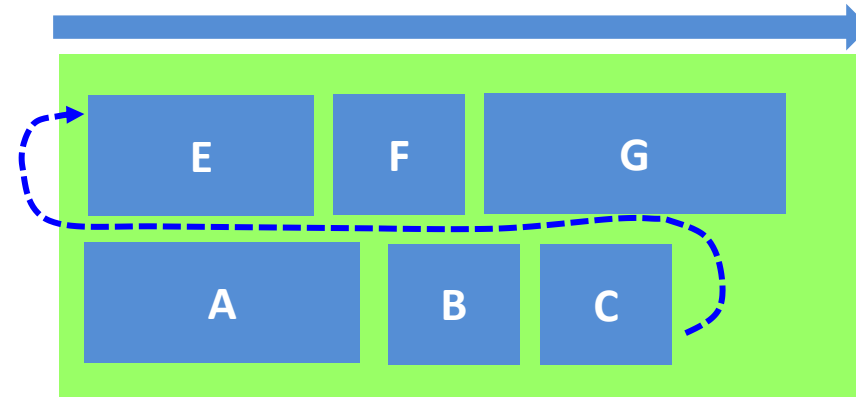
**nowrap**

valeur par défaut : les éléments ne passent pas à la ligne.



**wrap**

les éléments passent à la ligne dans le sens de lecture.



**wrap-reverse**

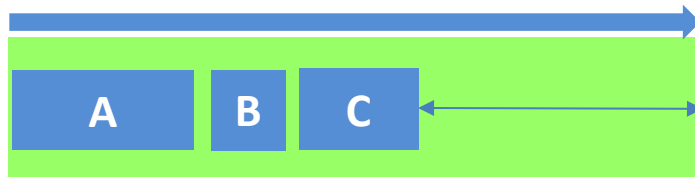
les éléments passent à la ligne dans le sens de lecture inverse.

# Positionnement 'Flexbox'

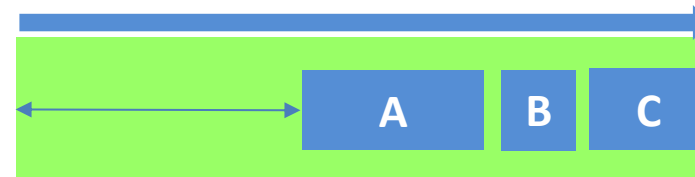
**justify-content** (propriété de flex-container)

définit l'alignement selon l'axe principal

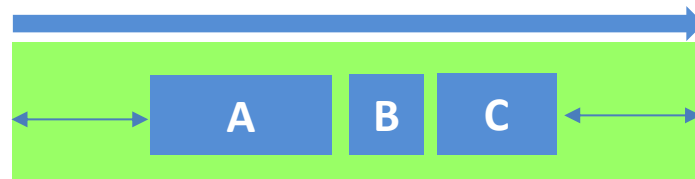
permet de distribuer l'espace restant libre supplémentaire lorsque les éléments d'une ligne sont soit rigides ou si ils sont flexibles ont atteint leur taille maximale. Permet d'exercer également un certain contrôle sur l'alignement des éléments lorsqu'ils débordent la ligne.



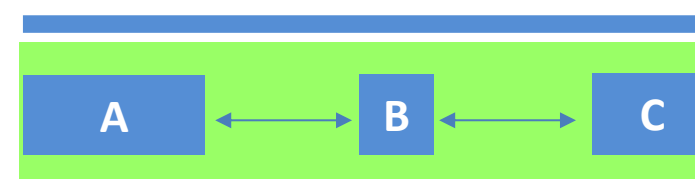
flex-start



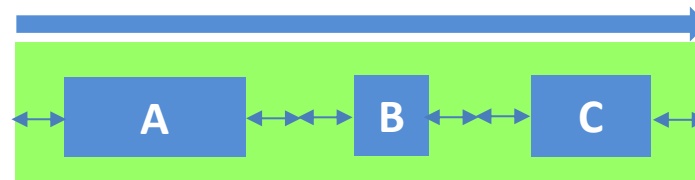
flex-end



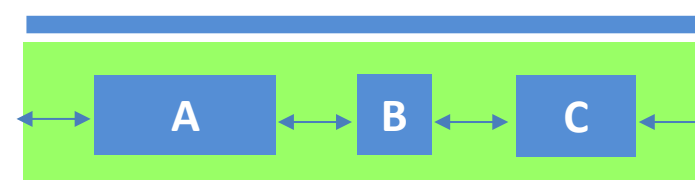
center



space-between



space-around

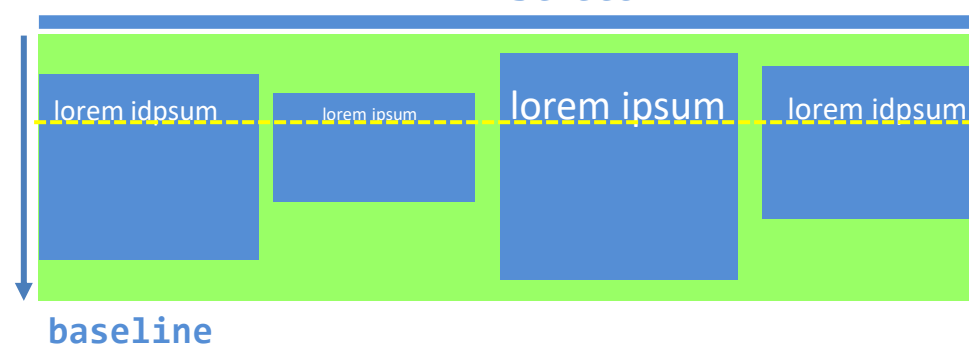
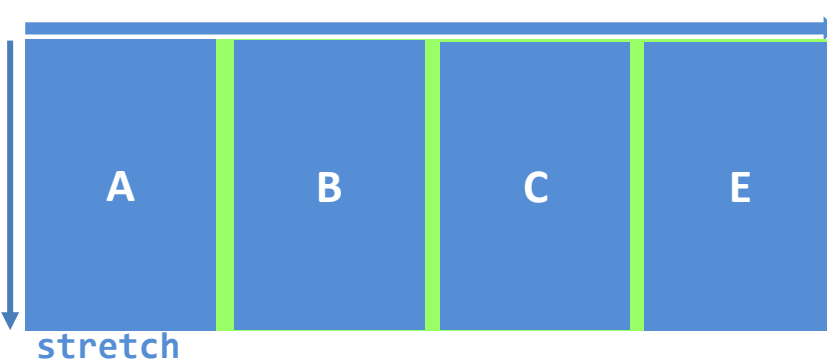
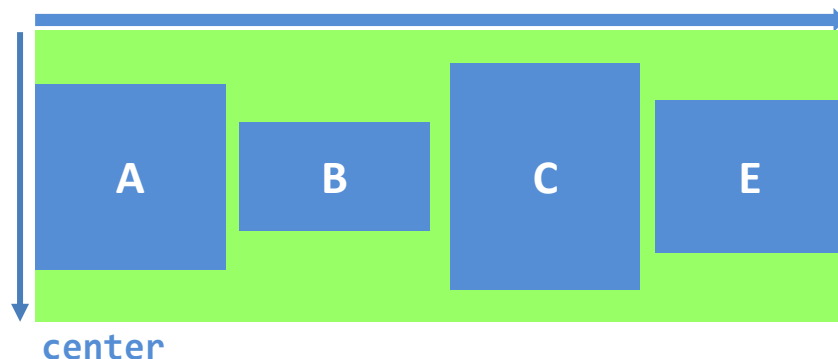
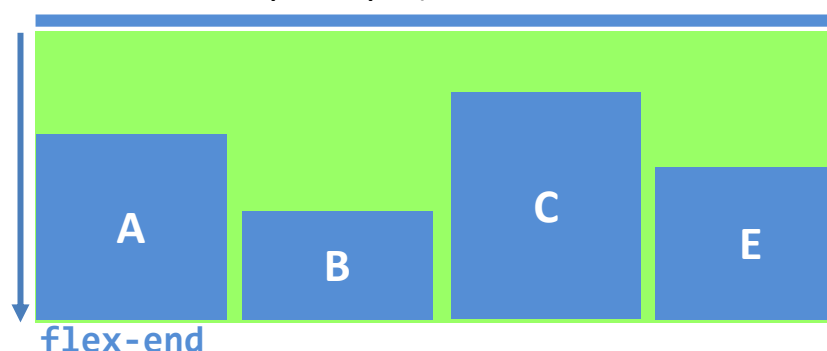
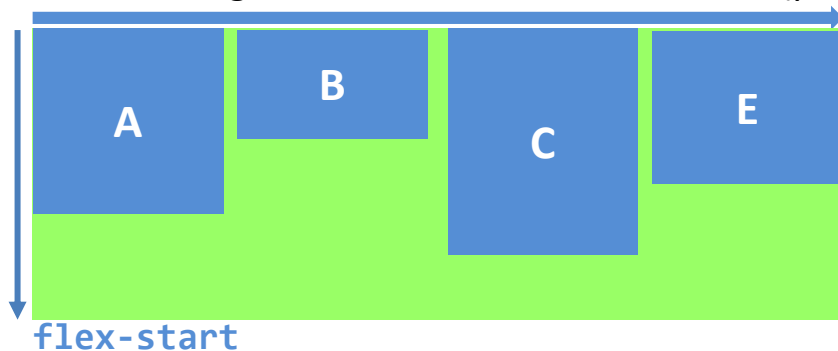


space-evenly

# Positionnement 'Flexbox'

**align-items** (propriété de flex-container)

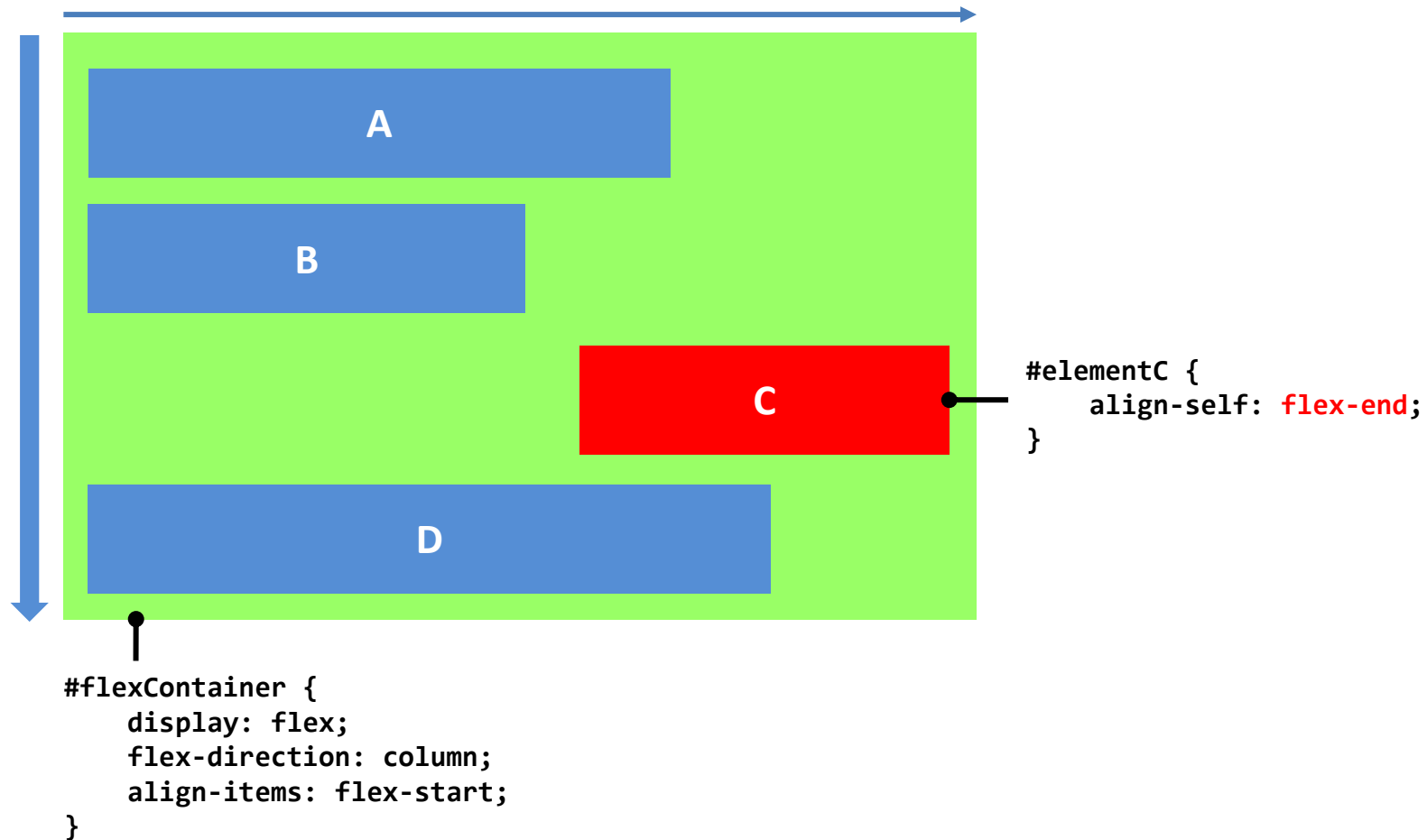
définit l'alignement selon l'axe secondaire (perpendiculaire à l'axe principal)



# Positionnement 'Flexbox'

**align-self** (propriété de flex-item)

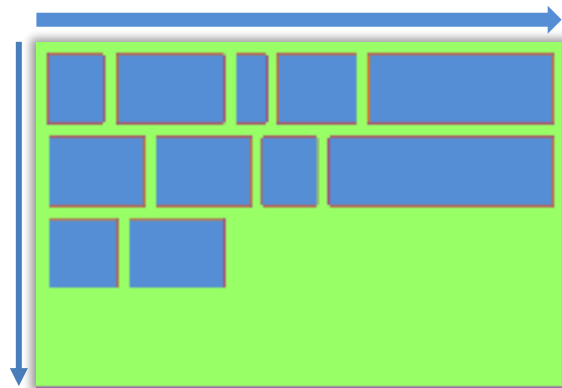
permet de définir un alignement spécifique pour un item particulier  
valeurs de cette propriété identiques à celles de **align-items**



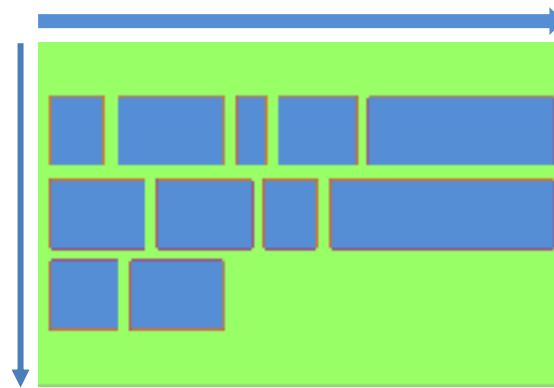
# Positionnement 'Flexbox'

**align-content:** (propriété de flex-container)

permet d'aligner les lignes d'un conteneur flex quand il reste de la place disponible selon l'axe secondaire (similaire à **justify-content** pour l'axe principal)



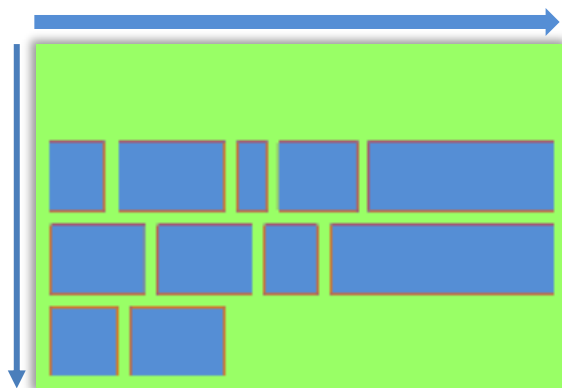
flex-start



center



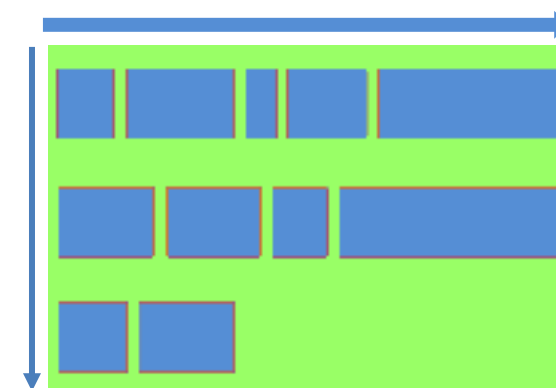
space-between



flex-end



stretch

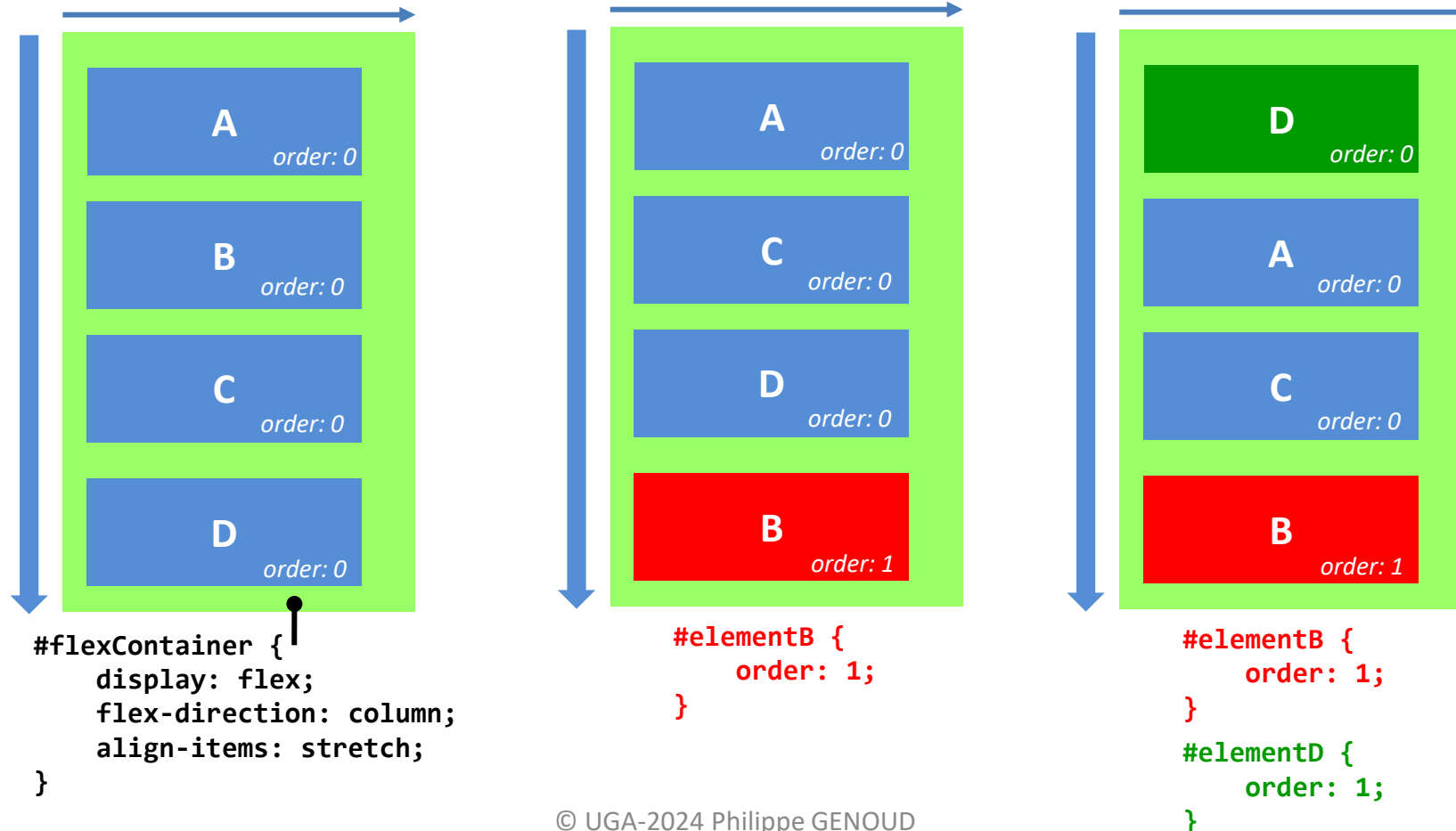


space-around

# Positionnement 'Flexbox'

**order:** (propriété d'un flex item)

- Par défaut, les éléments flex items sont affichés selon leur ordre d'apparition dans le code source de la page. La propriété **order** offre la possibilité de contrôler cet ordre élément par élément.
- La valeur de order (0 par défaut) indique l'ordre dans lequel les éléments sont affichés (de plus petit au plus grand)

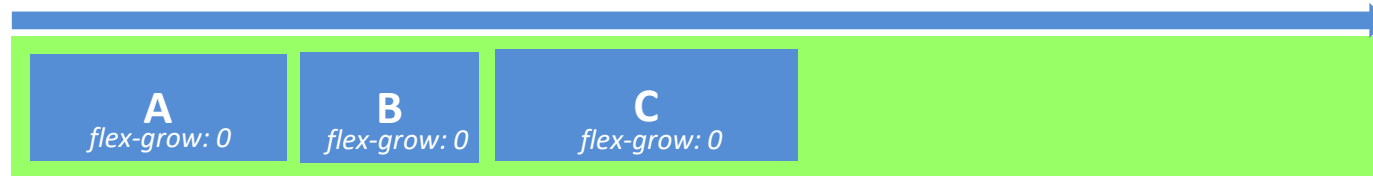


# Positionnement 'Flexbox'

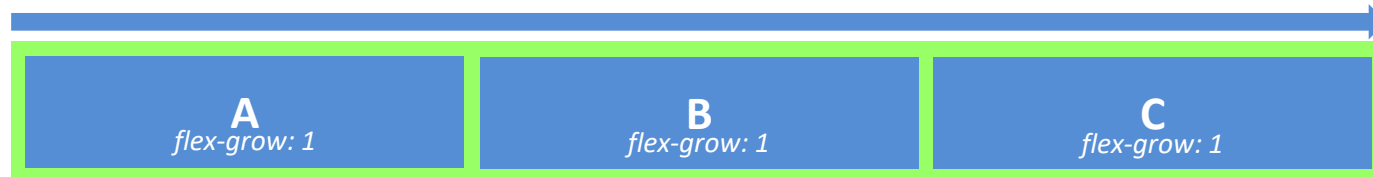
**flex-grow:** (propriété d'un flex item)

- définit la possibilité pour un flex-item de s'étirer ou non dans l'espace restant du conteneur flex.
- par défaut 0, pas d'étirement
- si valeur > 0, étirement (la proportion de l'espace restant attribué aux éléments étirables sera proportionnelle à la valeur de leur attribut flex-grow)

```
#flexContainer {  
  display: flex;  
}
```



```
#flexContainer {  
  display: flex;  
}  
#A, #B, #C {  
  flex-grow: 1;  
}
```



*tous les éléments flex-item ont un flex-grow égal à 1, l'espace restant est réparti de manière égale entre tous les items*

```
#flexContainer {  
  display: flex;  
}  
#A, #B {  
  flex-grow: 1;  
}  
#B {  
  flex-grow: 2;  
}
```



*l'élément B a un flex-grow de 2, l'espace restant qui lui sera attribué sera (si possible) 2 fois plus grand que celui attribué aux flex-items ayant un valeur de flex-grow égale à 1.*

# Positionnement 'Flexbox'

**flex-shrink:** (propriété d'un flex item)

- définit la possibilité pour un flex-item de se contracter si nécessaire.
- par défaut 1, pas d'étirement

**flex-shrink:** (propriété d'un flex item)

- définit la possibilité pour un flex-item de se contracter si nécessaire.
- par défaut 1, pas d'étirement





# CSS Grid

**TABLE OF CONTENTS**

- 1 Introduction**
  - 1.1 Background and Motivation
    - 1.1.1 Adapting Layouts to Available Space
    - 1.1.2 Source-Order Independence
  - 1.2 Value Definitions
- 2 Overview**
  - 2.1 Declaring the Grid
  - 2.2 Placing Items
  - 2.3 Sizing the Grid
- 3 Grid Layout Concepts and Terminology**
  - 3.1 Grid Lines
  - 3.2 Grid Tracks and Cells
  - 3.3 Grid Areas
- 4 Reordering and Accessibility**
- 5 Grid Containers**
  - 5.1 Establishing Grid Containers: the 'grid' and 'inline-grid' 'display' values
  - 5.2 Sizing Grid Containers
  - 5.3 Scrollable Grid Overflow
  - 5.4 Limiting Large Grids
- 6 Grid Items**
  - 6.1 Grid Item Display
  - 6.2 Grid Item Sizing
  - 6.3 Reordered Grid Items: the 'order' property
  - 6.4 Grid Item Margins and Paddings
  - 6.5 Z-axis Ordering: the 'z-index' property
  - 6.6 Automatic Minimum Size of Grid Items
- 7 Refining the...**

**CSS Grid Layout Module Level 1**  
W3C Candidate Recommendation Draft, 18 December 2020

**This version:**  
<https://www.w3.org/TR/2020/CRD-css-grid-1-20201218/>

**Latest published version:**  
<https://www.w3.org/TR/css-grid-1/>

**Editor's Draft:**  
<https://drafts.csswg.org/css-grid/>

**Previous Versions:**  
<https://www.w3.org/TR/2020/CRD-css-grid-1-20201021/>  
<https://www.w3.org/TR/2020/CR-css-grid-1-20200818/>  
<https://www.w3.org/TR/2017/CR-css-grid-1-20171214/>  
<https://www.w3.org/TR/2017/CR-css-grid-1-20170509/>  
<https://www.w3.org/TR/2017/CR-css-grid-1-20170209/>  
<https://www.w3.org/TR/2016/CR-css-grid-1-20160929/>  
<https://www.w3.org/TR/2016/WD-css-grid-1-20160519/>  
<https://www.w3.org/TR/2015/WD-css-grid-1-20150917/>  
<https://www.w3.org/TR/2015/WD-css-grid-1-20150806/>  
<https://www.w3.org/TR/2015/WD-css-grid-1-20150317/>  
<https://www.w3.org/TR/2014/WD-css-grid-1-20140513/>  
<https://www.w3.org/TR/2014/WD-css-grid-1-20140123/>  
<https://www.w3.org/TR/2013/WD-css3-grid-layout-20130910/>  
<https://www.w3.org/TR/2013/WD-css3-grid-layout-20130402/>  
<https://www.w3.org/TR/2012/WD-css3-grid-layout-20121106/>  
<https://www.w3.org/TR/2012/WD-css3-grid-layout-20120322/>  
<https://www.w3.org/TR/2011/WD-css3-grid-layout-20110407/>

**Implementation Report:**  
<https://wpt.fyi/results/css/css-grid>

**Test Suite:**  
[http://test.csswg.org/suites/css-grid-1\\_dev/nightly-unstable/](http://test.csswg.org/suites/css-grid-1_dev/nightly-unstable/)

**Issue Tracking:**  
CSSWG Issues Repository  
Disposition of Comments  
Issue Tracking

<https://www.w3.org/TR/css-grid-1/>

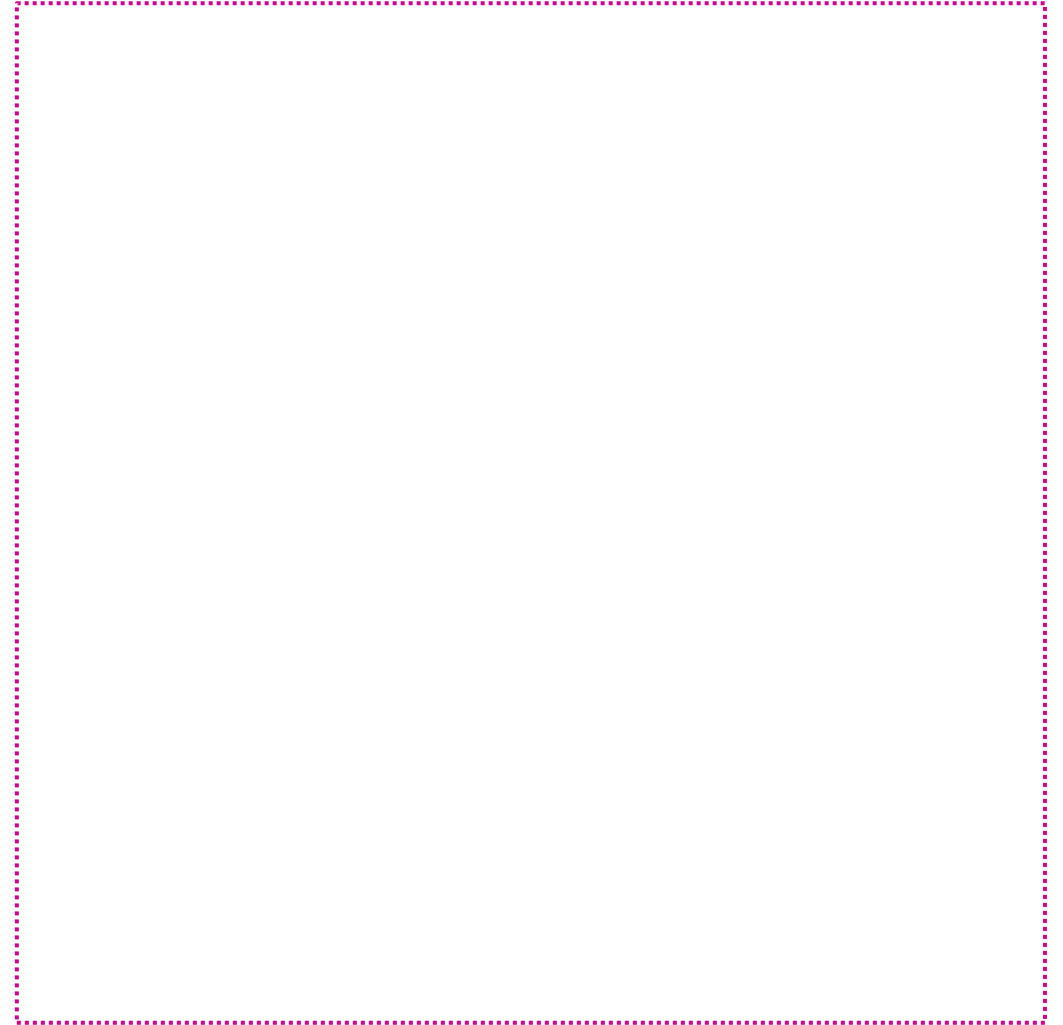
# CSS Grid

- Terminologie

## Grid **container**

- Élément qui accueille la grille
- Peut être n'importe quel type d'élément :
  - body, div, section, footer, header ....

`<div>`



# CSS Grid

- Terminologie  
Grid **Lines**

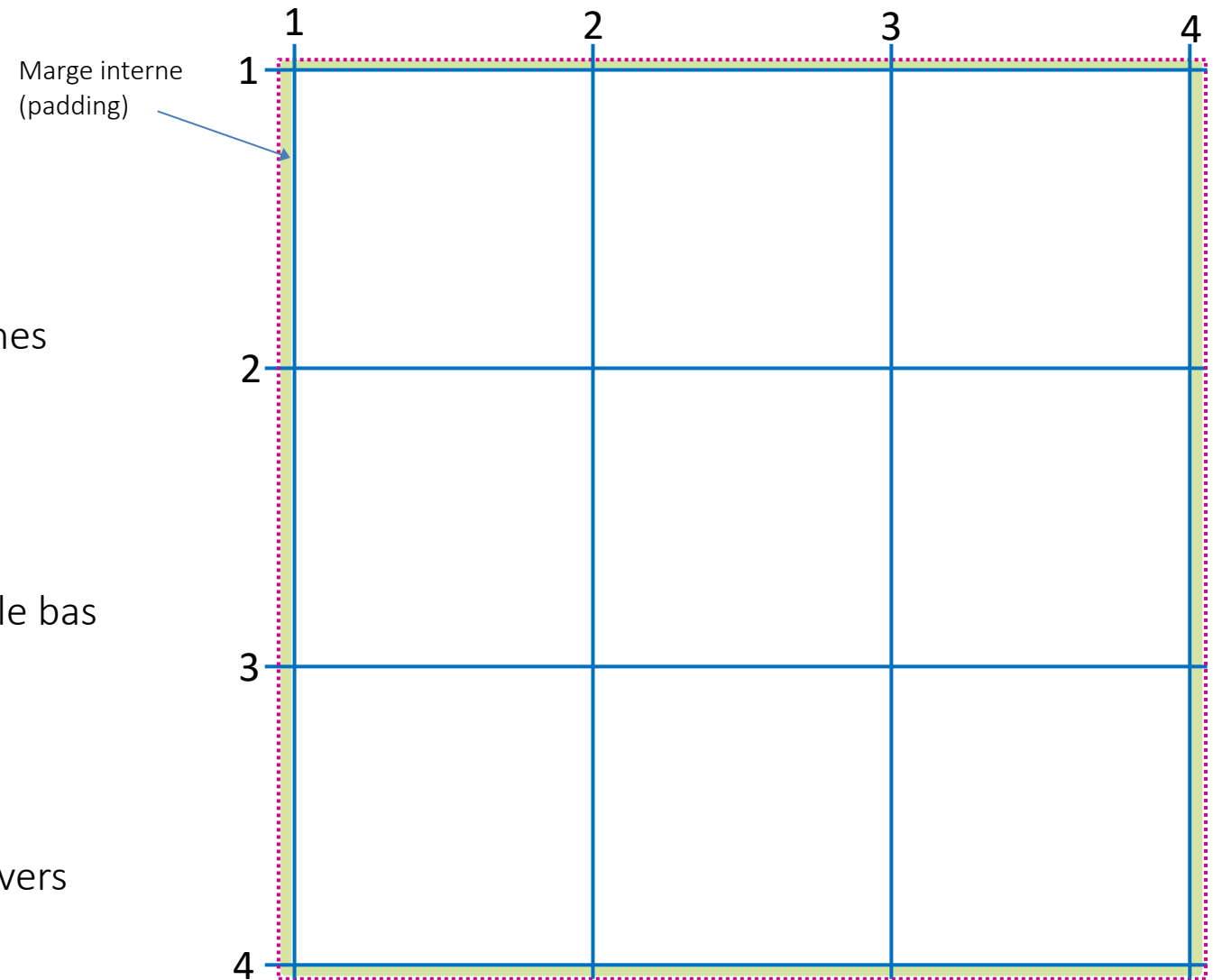
- Divisent le grid container pour créer une structure de grille régulière en lignes et colonnes

## row **Lines**

- Lignes horizontales
- Numérotées de 1 à  $n_h$  en partant du haut vers le bas

## column **Lines**

- Lignes verticales
- Numérotées de 1 à  $n_v$  en partant de la gauche vers la droite



La grille peut être quelconque (carrée ou rectangulaire) :  $n_h \geq 2$  et  $n_v \geq 2$

# CSS Grid

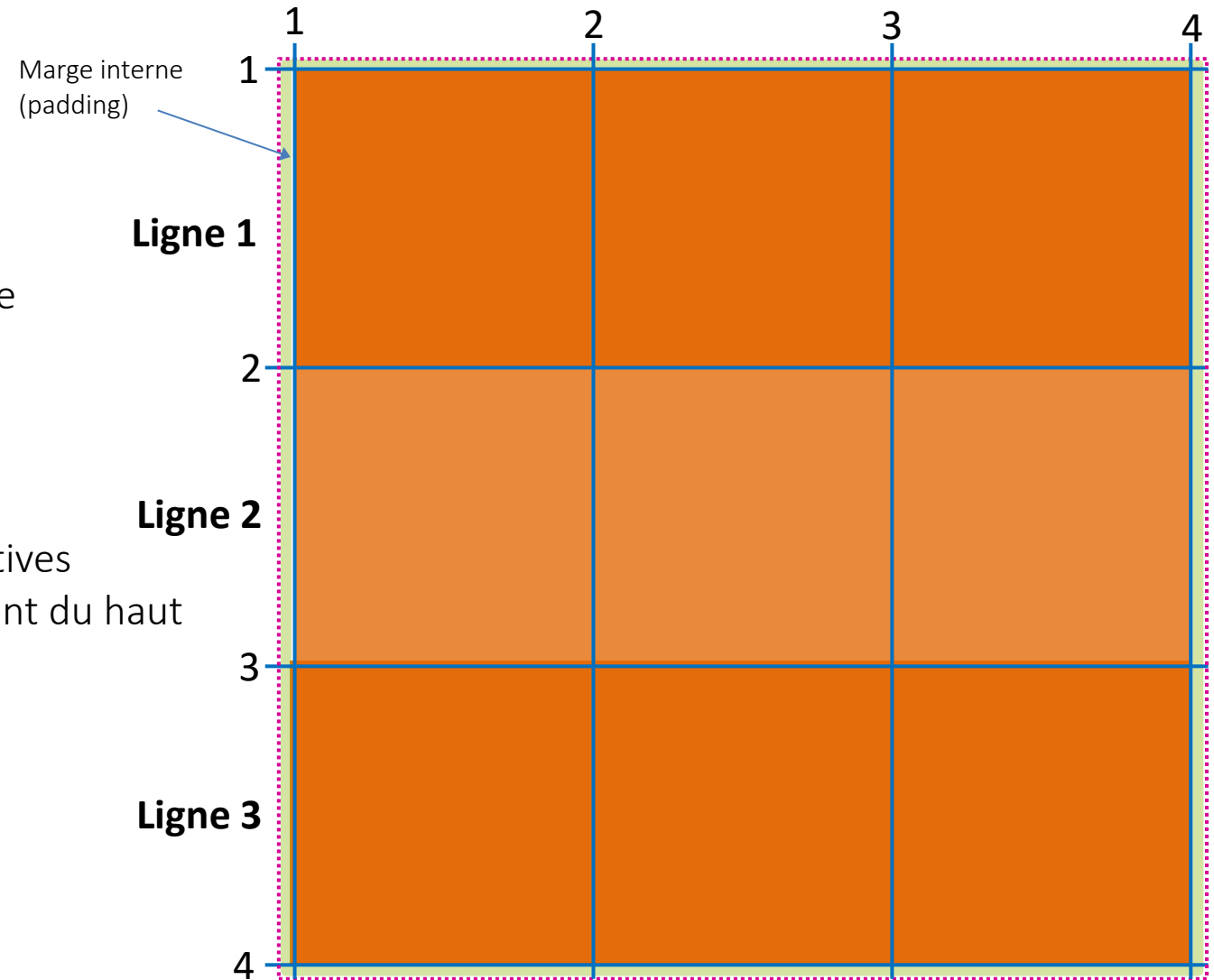
- Terminologie

## Grid Tracks

- Région entre deux lignes adjacentes de la grille

## Rows (lignes)

- Région entre deux lignes horizontales consécutives
- Les lignes sont numérotées de 1 à  $n_{h-1}$  en partant du haut



# CSS Grid

- Terminologie

## Grid Tracks

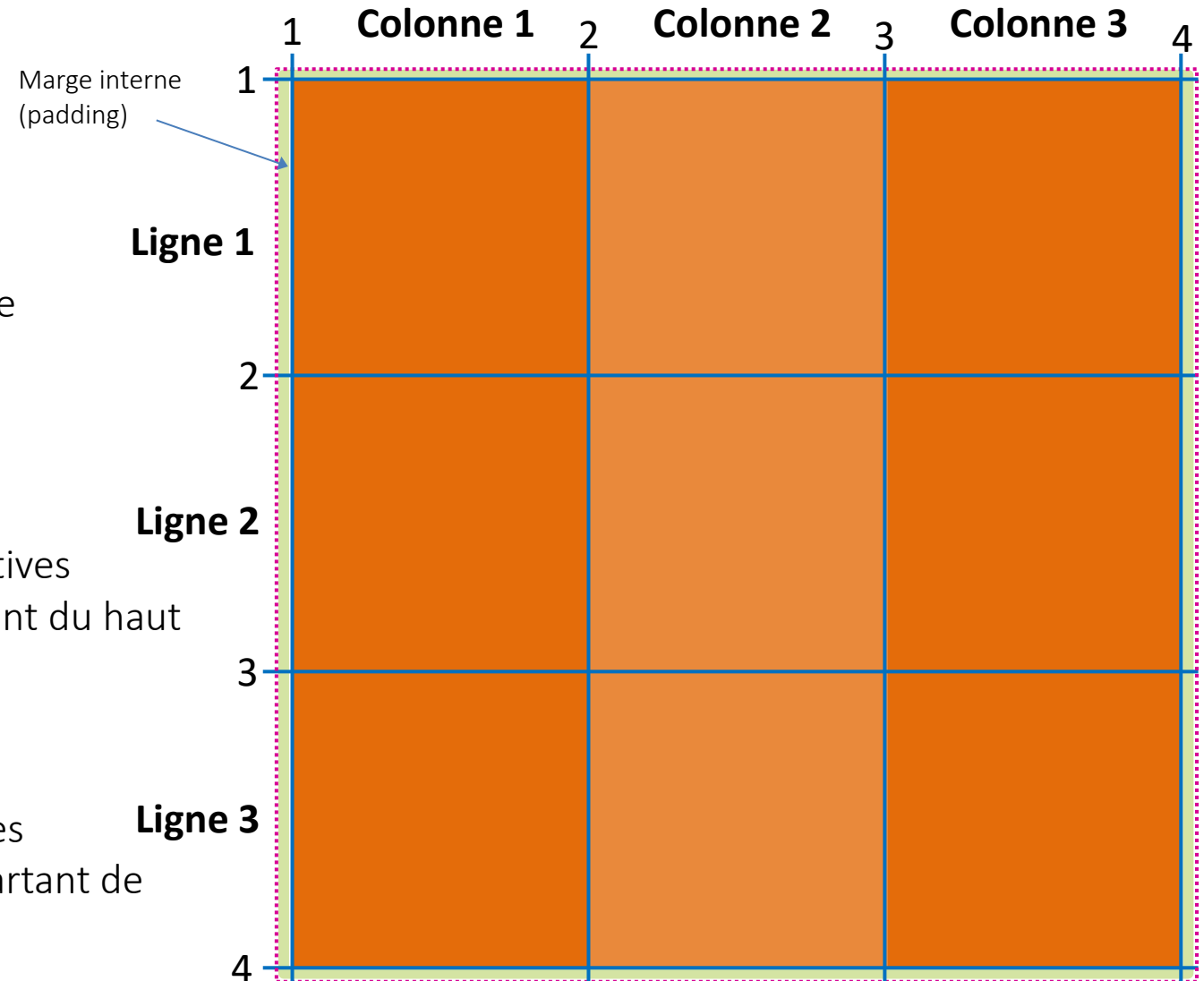
- Région entre deux lignes adjacentes de la grille

## Rows (lignes)

- Région entre deux lignes horizontales consécutives
- Les lignes sont numérotées de 1 à  $n_{h-1}$  en partant du haut

## Columns (colonnes)

- Région entre deux lignes verticales consécutives
- Les colonnes sont numérotées de 1 à  $n_{v-1}$  en partant de la gauche

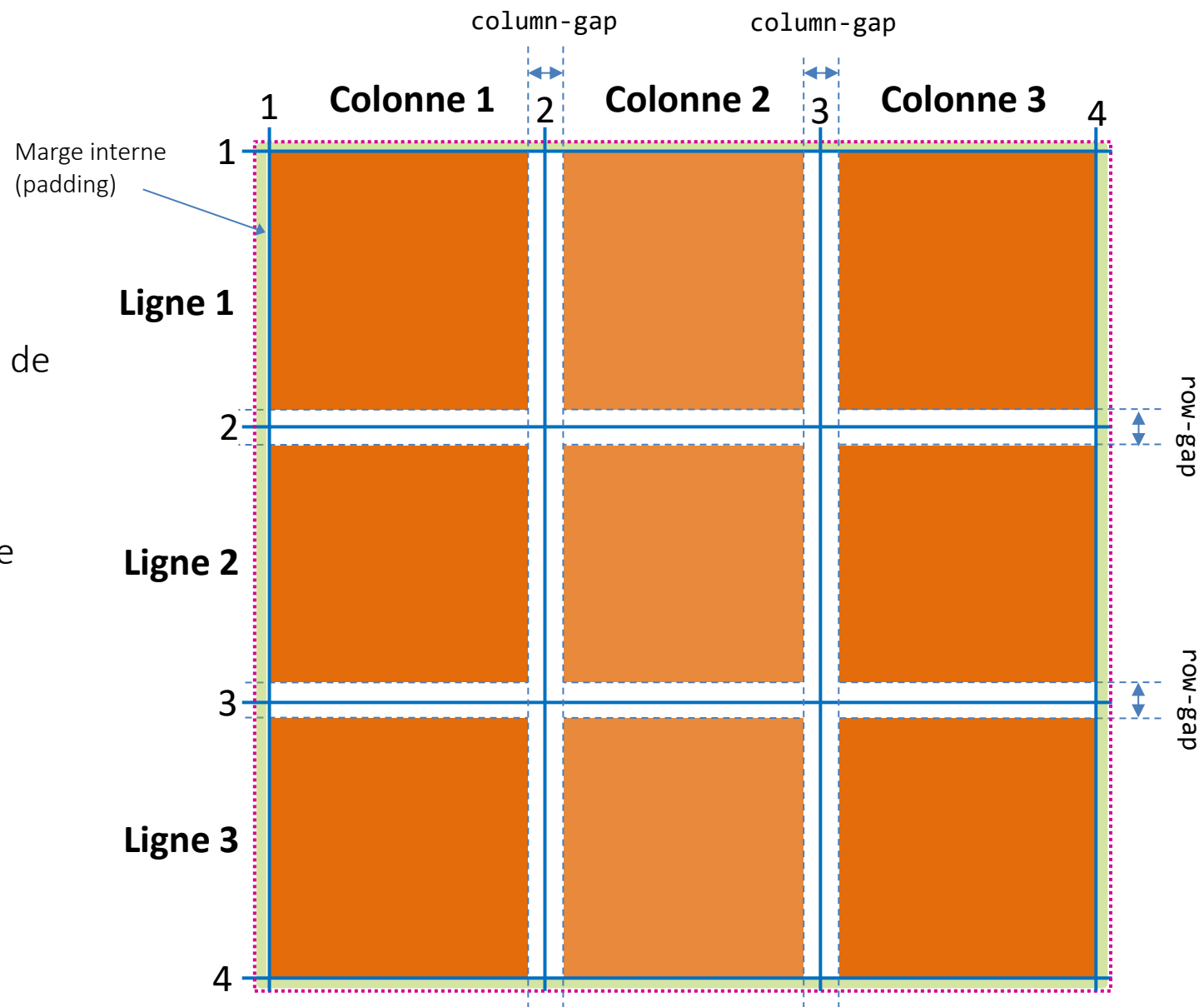


# CSS Grid

- Terminologie

## Grid Gaps

- Espace entre les lignes et colonnes adjacentes de la grille
- **column-gap** : espace entre deux colonnes (identique pour toutes les colonnes)
- **row-gap** : espace entre deux lignes (identique pour toutes les lignes)
- **column-gap** et **row-gap** peuvent avoir des valeurs différentes

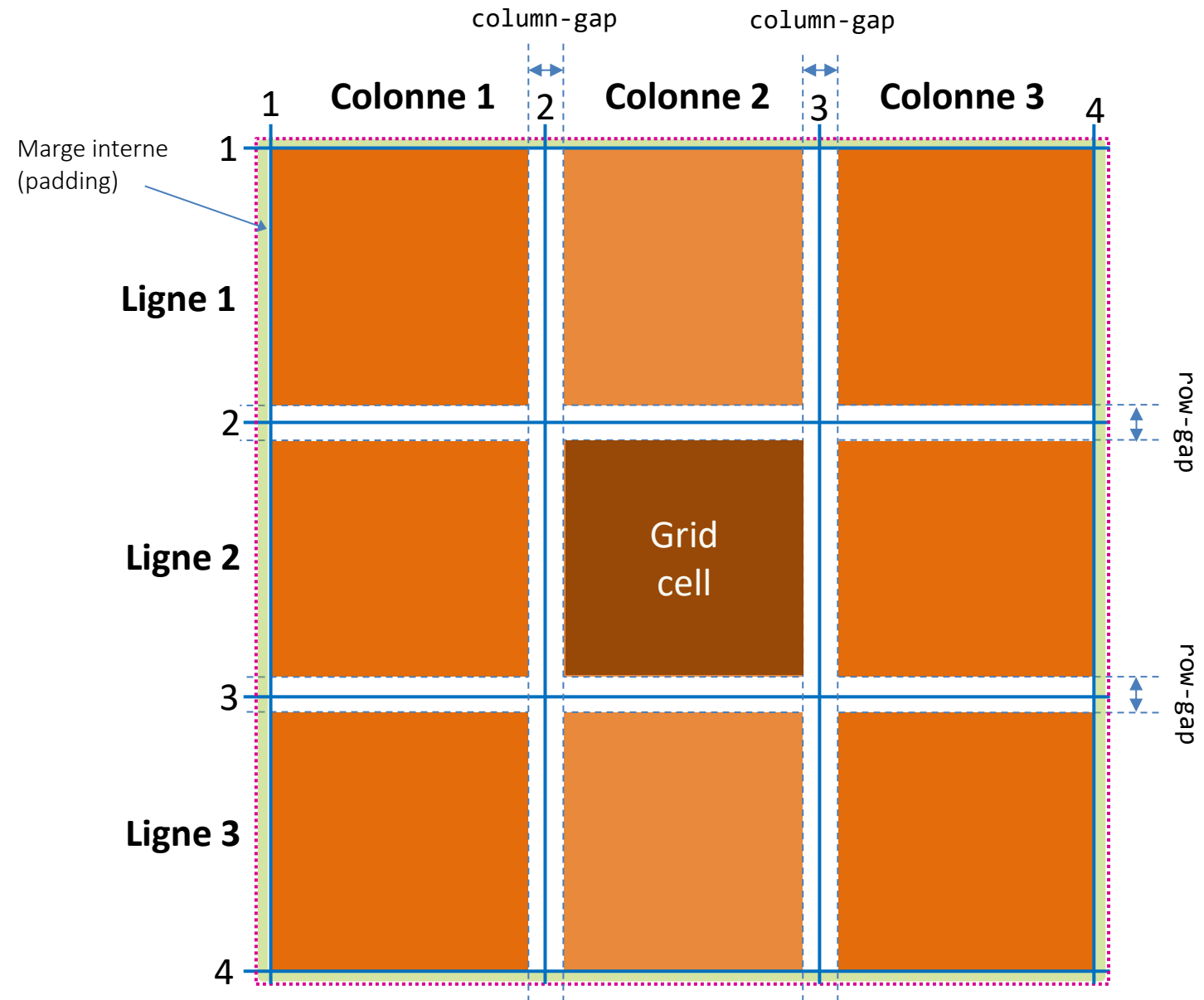


# CSS Grid

- Terminologie

## Grid Cells

- Intersection d'un ligne et d'une colonne
- L'unité de base de la grille
- Le nombre de cellules est  $n_{h-1} * n_{v-1}$

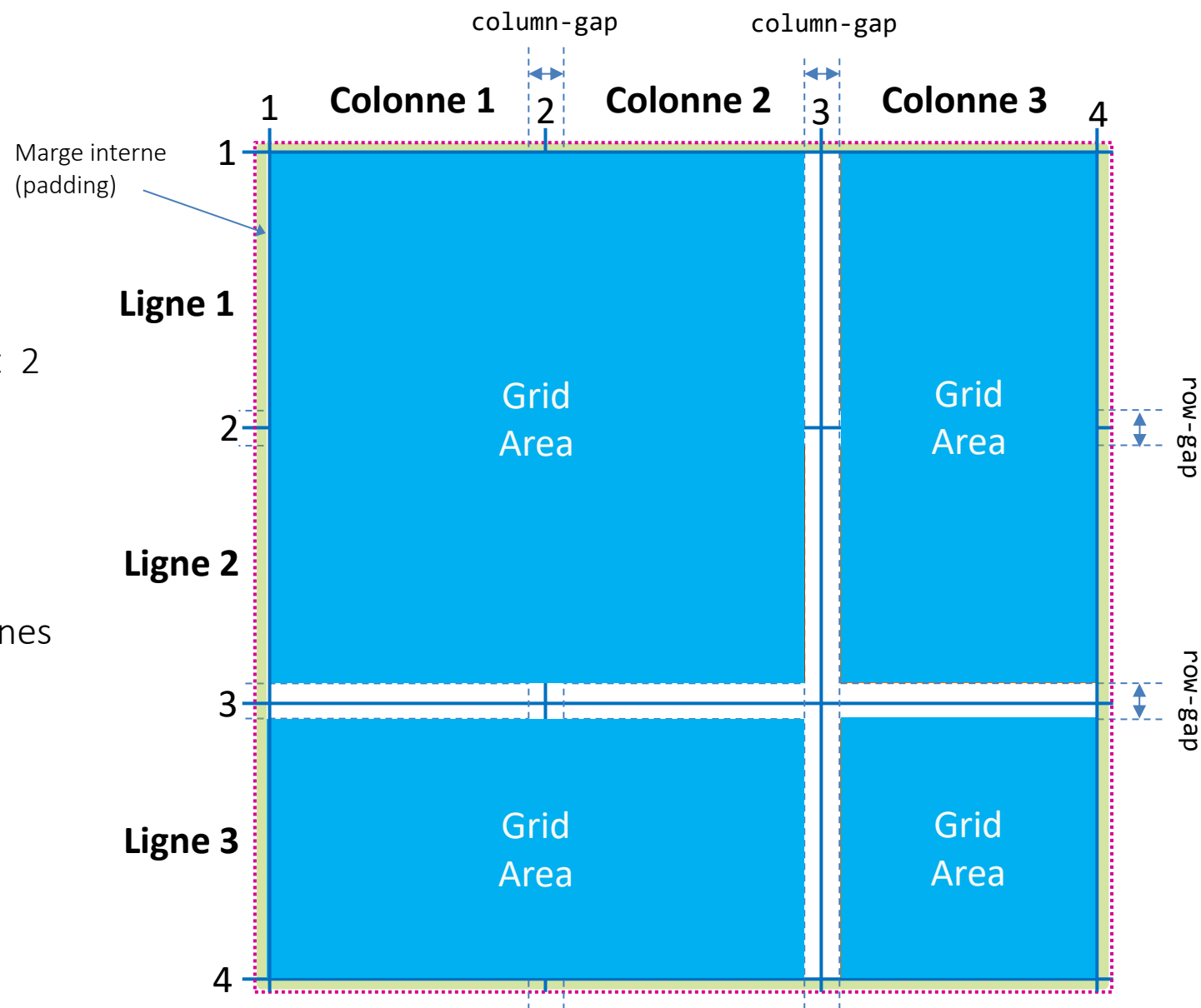


# CSS Grid

- Terminologie

## Grid Area

- La zone définie par 4 grid lines (2 rows lines et 2 column lines)
- C'est **toujours** une zone rectangulaire
  
- Identifiée par les numéros des lignes et colonnes qui la délimitent





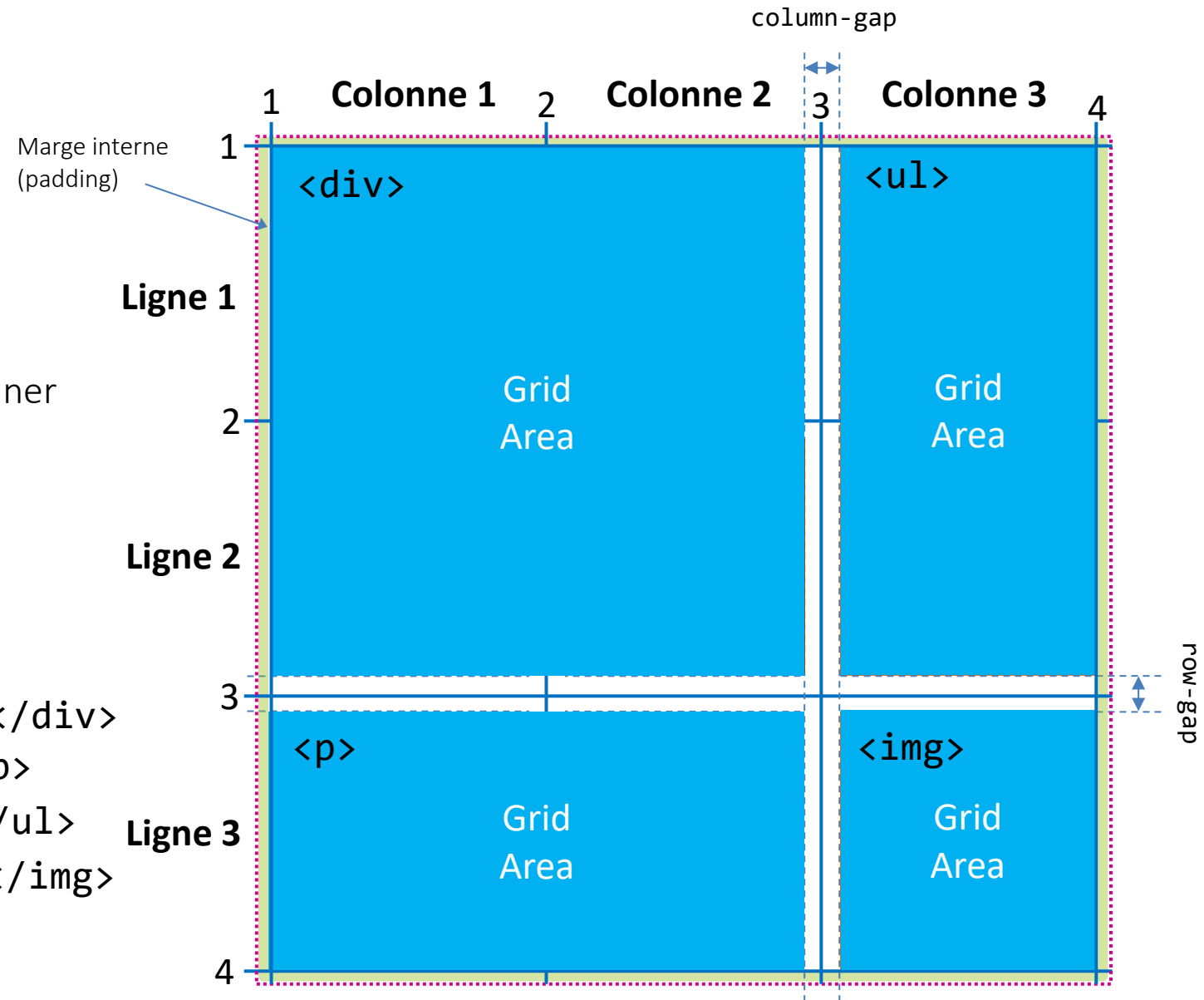
# CSS Grid

- Terminologie

## Grid items

- Éléments **descendants directs** du grid container
- Associés à une zone de la grille (grid area)
- Peuvent être de n'importe quel type :
  - p, ul, div, a, img ...

```
<div class="grid-container">  
  <div class="grid-item1"> ... </div>  
  <p class="grid-item2"> ... </p>  
  <ul class="grid-item3"> ... </ul>  
  <img class="grid-item4"> ... </img>  
</div>
```



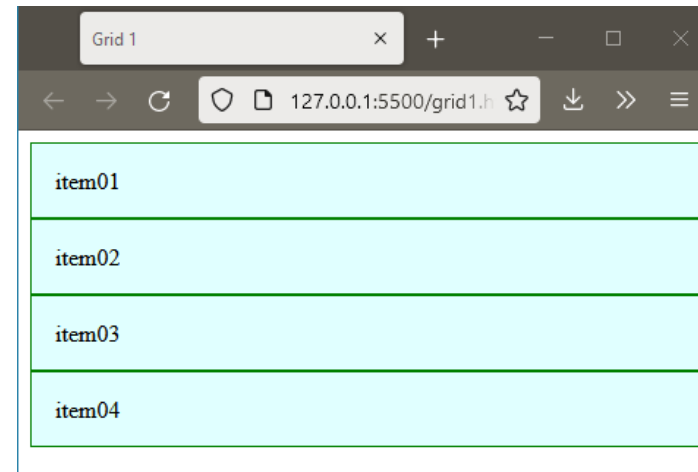
# CSS Grid

- Créer un conteneur grid
  - Possibilité d'utiliser n'importe quel type d'élément HTML (`body`, `div`, `p`, `ul`...) en lui associant la propriété `display` avec l'une des deux valeurs `grid` ou `inline-grid`

```
#my-grid {  
  display: grid;  
}  
  
/* style pour démo : pas nécessaire pour les grilles */  
.item {  
  border: solid 1px green;  
  background-color: lightcyan;  
  padding: 1rem;  
}
```

```
<div id="my-grid">  
  <div class="item">item01</div>  
  <div class="item">item02</div>  
  <div class="item">item03</div>  
  <div class="item">item04</div>  
</div>
```

Tout type d'élément ajouté à la grille est considéré comme un *grid item*



Par défaut le conteneur grille se comporte comme un flow layout normal (ici `div` les un en dessous des autres)

Pour les positionner il faut définir les lignes et colonnes de la grille explicitement

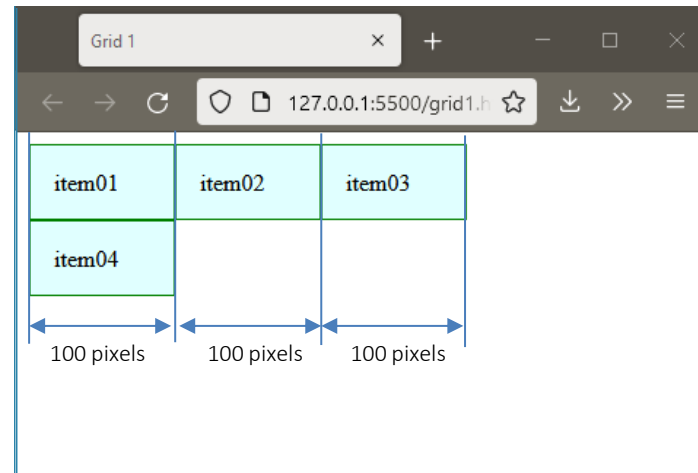
Nb: le nom de classe `item` est arbitraire (il peut être ce que vous voulez et n'est pas imposé par le layout en grille)

# CSS Grid

- Définir les lignes et les colonnes
  - Effectué à l'aide des propriétés `grid-template-rows` et `grid-template-columns` qui permettent de spécifier le nombre de lignes (colonnes) avec éventuellement leur hauteur (largeur)

```
#my-grid {  
  display: grid;  
  grid-template-columns: 100px 100px 100px;  
}  
  
/* style pour démo : pas nécessaire pour les grilles */  
.item {  
  border: solid 1px green;  
  background-color: lightcyan;  
  padding: 1rem;  
}
```

```
<div id="my-grid">  
  <div class="item">item01</div>  
  <div class="item">item02</div>  
  <div class="item">item03</div>  
  <div class="item">item04</div>  
</div>
```



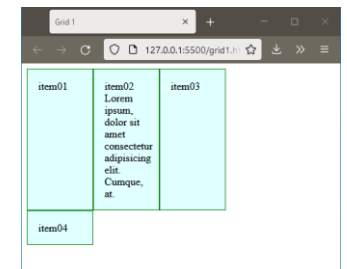
Grille avec 3 colonnes d'une largeur de 100 pixels

Les 3 premiers items sont placés successivement dans chacune des colonnes

Pour le 4<sup>ème</sup> item la grille utilise un algorithme d'auto placement qui le positionne automatiquement dans une nouvelle ligne

Ici les élément d'une ligne de la grille ont une hauteur calculée automatiquement (qui dépend du contenu)

```
<div id="my-grid">  
  <div class="item">item01</div>  
  <div class="item">item02<br> Lorem ipsum, dolor sit amet  
  consectetur adipiscing elit. Cumque, at.</div>  
  <div class="item">item03</div>  
  <div class="item">item04</div>  
</div>
```

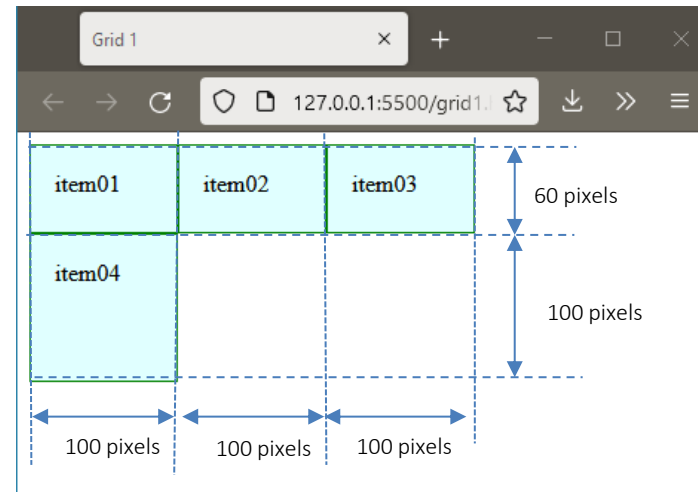


# CSS Grid

- Définir les lignes et les colonnes
  - Effectué à l'aide des propriétés css `grid-template-rows` et `grid-template-columns` qui permettent de spécifier le nombre de lignes (colonnes) avec éventuellement leur hauteur (largeur)

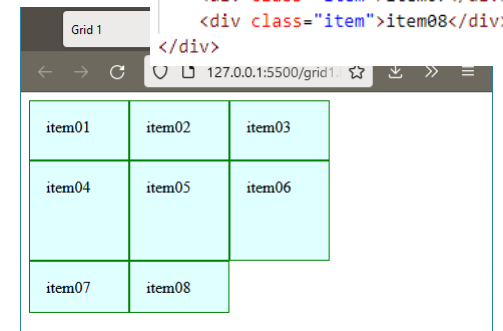
```
#my-grid {  
  display: grid;  
  grid-template-columns: 100px 100px 100px;  
  grid-template-rows : 60px 100px  
}  
  
/* style pour démo : pas nécessaire pour les grilles */  
.item {  
  border: solid 1px green;  
  background-color: lightcyan;  
  padding: 1rem;  
}
```

```
<div id="my-grid">  
  <div class="item">item01</div>  
  <div class="item">item02</div>  
  <div class="item">item03</div>  
  <div class="item">item04</div>  
</div>
```



Grille avec 3 colonnes d'une largeur de 100 pixels et deux lignes de hauteur respective 60 et 100 pixels

```
<div id="my-grid">  
  <div class="item">item01</div>  
  <div class="item">item02</div>  
  <div class="item">item03</div>  
  <div class="item">item04</div>  
  <div class="item">item05</div>  
  <div class="item">item06</div>  
  <div class="item">item07</div>  
  <div class="item">item08</div>  
</div>
```



Si le container contient plus d'items que la grille définit de cellules l'algorithme d'auto placement les positionne automatiquement dans de nouvelle ligne dont la hauteur sera calculée automatiquement en fonction du contenu des items qui la constituent

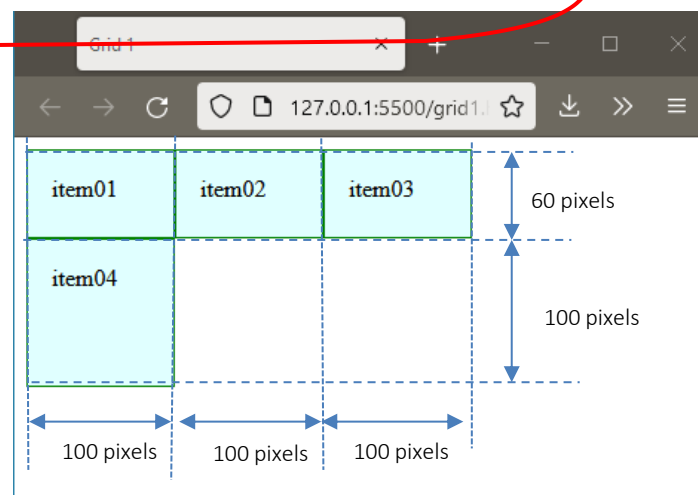
# CSS Grid

- Définir les lignes et les colonnes
  - `grid-template : rows / columns` raccourci des propriétés `grid-template-rows` et `grid-template-columns`

```
#my-grid {  
  display: grid;  
  grid-template-columns : 100px 100px 100px;  
  grid-template-rows : 60px 100px;  
}  
  
/* style pour démo : pas nécessaire pour les grilles */  
.item {  
  border: solid 1px green;  
  background-color: lightcyan;  
  padding: 1rem;  
}
```

```
<div id="my-grid">  
  <div class="item">item01</div>  
  <div class="item">item02</div>  
  <div class="item">item03</div>  
  <div class="item">item04</div>  
</div>
```

`grid-template : 60px 100px / 100px 100px 100px;`



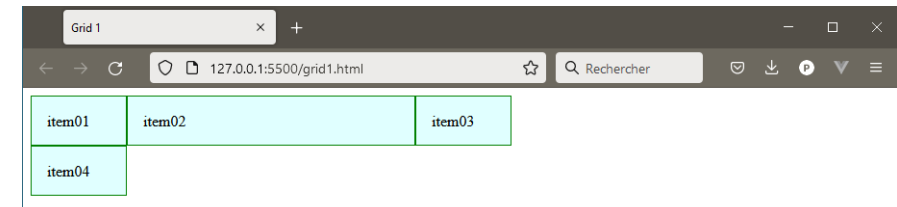
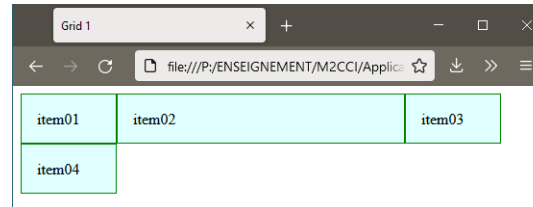
# CSS Grid

- Unités pour définir les largeurs de colonnes et hauteurs de lignes

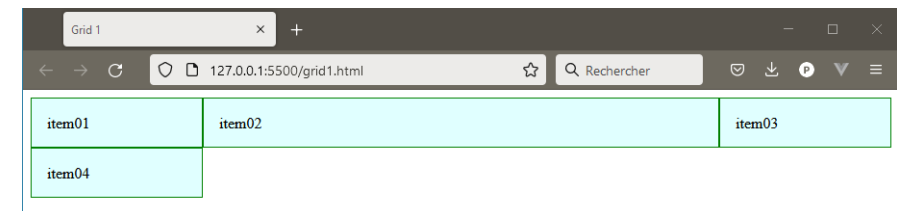
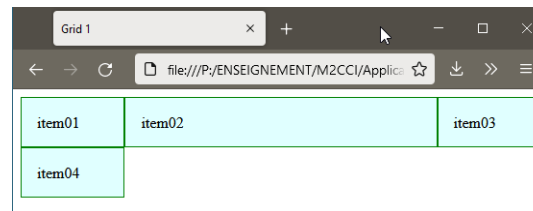
- Colonnes :

- Taille fixe : `px`
- Taille relative : `%` pourcentage de la largeur du container
- Fraction : `fr` fraction de la largeur disponible
- ...

```
grid-template-columns : 100px 300px 100px;
```



```
grid-template-columns : 20% 60% 20%;
```



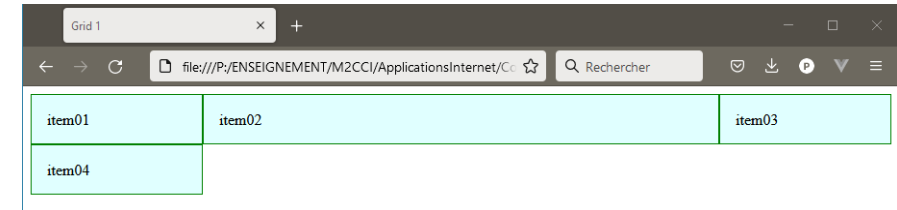
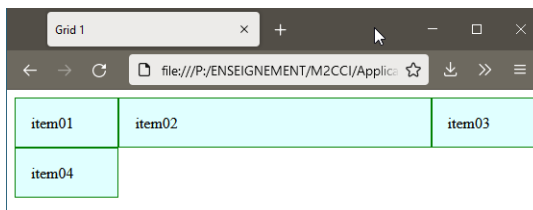
```
grid-template-columns : 2fr 6fr 2fr;
```

Le navigateur calcule le nombre total de fractions : 10

1<sup>ère</sup> colonne occupe 2 fractions sur 10

2<sup>ème</sup> colonne occupe 6 fractions sur 10

3<sup>ème</sup> colonne occupe 2 fractions sur 10



# Responsive Web Design : Media queries

- Nécessité d'adapter l'affichage des pages à différents modes de consultation



- Avec CSS2 et HTML4 il était déjà possible de spécifier un média de destination pour l'application d'une feuille de style

```
<!doctype html>
<head>
  ...
  <link rel="stylesheet" media="screen" href="screen.css" type="text/css" />
  <link rel="stylesheet" media="print" href="print.css" type="text/css" />
</head>
```

*Attribut précisant le contexte dans lequel les styles doivent être appliqués*

Media queries CSS3 permettent de définir un panel de critères plus précis en fonction de facteurs liés au dispositif d'affichage : Largeur, hauteur, orientation, résolution...

- **media query** : expression booléenne définissant un ensemble de conditions à réunir pour l'application de styles

```
@media screen and (min-width: 600px) and (max-width: 1024px) {  
  .bloc {  
    display:block;  styles appliqués si l'expression  
    clear:both;    media est vraie  
  }  
  img {  
    float left;  
  }  
}
```

```
<link rel="stylesheet" media="screen and (max-width: 640px)"  
      href="smallscreen.css" type="text/css" />
```

Opérateurs logiques:  
**and, or, not, only**

Critères possibles:  
préfixés par **min-** ou  
**max-** lorsqu'ils  
acceptent des valeurs  
numériques

**color** support de la couleur (bits/pixel)  
**color-index** périphérique utilisant une table de couleurs indexées  
**device-aspect-ratio** ratio du périphérique de sortie (par exemple 16/9)  
**aspect-ratio** ratio de la zone d'affichage  
**device-height** dimension en hauteur du périphérique  
**device-width** dimension en largeur du périphérique  
**grid** périphérique bitmap ou grille (ex : lcd)  
**height** dimension en hauteur de la zone d'affichage  
**monochrome** périphérique monochrome ou niveaux de gris (bits/pixel)  
**orientation** orientation du périphérique (**portrait** ou **landscape**)  
**resolution** résolution du périphérique (en dpi, dppx, ou dpcm)  
**scan** type de balayage des téléviseurs (**progressive** ou **interlace**)  
**width** dimension en largeur de la zone d'affichage