Fiche module Flexbox

Au même titre que la propriété display et ses valeurs:inline, inline-block ou block ; la propriété **display avec la valeur flex** va permettre d'agencer des blocs sur une page web.
Si l'objectif est le même : agencer les blocs sur la page, la technique d'utilisation de la balise display : flex est différente et ses applications sont variées.

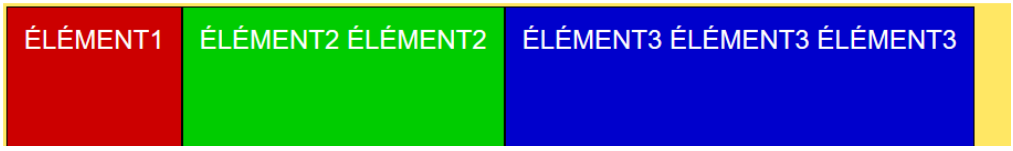
La notion de flexibilité constitue le fondement du module de positionnement Flexbox.

1. Alors qu'on appliquait display : inline-block sur tous les éléments (*enfants*) à positionner à l'intérieur d'un bloc conteneur (*parent*), display :flex s'applique directement sur le bloc conteneur.

```
<div id="flex-container">
        <div class="element">Elément1</div>
        <div class="element">Elément2  Elément2</div>
        <div class="element">Elément3 Elément3 Elément3</div>
</div>
```

```
#flex-container {
        display :flex ;
}
```

Cette propriété permet d'agencer, en mode flexbox, les éléments qui se trouvent à l'intérieur du flex-container. Par défaut tous les éléments se positionnent les uns à côté des autres. Si aucune largeur n'est précisée, la largeur de chaque bloc est celle de son contenu et sa hauteur est celle du bloc conteneur. Tant qu'on ne donne aucune autre instruction, les blocs restent positionnés les uns à côté des autres, quel que soit le contenu.



La propriété **display :flex** va être complétée par d'autres propriétés applicables parent :

- **flex-direction** indique selon quel axe les éléments sont organisés : horizontal (en ligne) ou vertical (en colonnes);
- **flex-wrap** spécifie si l'on autorise ou pas les passages à la ligne ou sur une seconde colonne des éléments lorsque l'espace est insuffisant;
- **justify-content** qui spécifie la façon dont les éléments sont organisés sur l'axe principal, si l'axe principal est horizontal à gauche, à doite, centré ou si l'axe est vertical, en haut, en bas, centré, etc.
- **align-items** qui permet, en tenant compte de l'axe principal, d'agencer les éléments sur l'axe secondaire ;
  Il existe également des propriétés applicables directement à l'élément enfant : **align-self** et **order.**

2. La propriété flex-direction

La **propriété flex-direction** (utilisée avec **display:flex**) permet de positionner les blocs selon un axe principal (main-*axis* ; axe principal horizontal ou axe vertical).

```
#flex-container {
        display :flex ;
        flex-direction :row ; (valeur par défaut)
}
```

Ses valeurs :

flex-direction : row (par défaut), l'axe principal est l'axe horizontal. Les blocs apparaissent les uns à côté des autres.

flex-direction : row-reverse : les éléments à l'intérieur du bloc conteneur apparaissent côte à côte, en ordre inverse

flex-direction : column : l'axe principal est l'axe vertical. Les blocs sont organisés les uns en dessous des autres. Ils occupent toute la largeur de la fenêtre du navigateur.

flex-direction : column-reverse :
Les éléments à l'intérieur du bloc conteneur apparaissent les uns en dessous des autres, en ordre inverse

3. La propriété flex-wrap

La **propriété flex-wrap** utilisée avec les propriétés **display:flex et flex-direction** permet de gérer l'affichage des blocs lorsqu'il n'y a plus d'espace.

3.a. #flex-container {

display :flex ;

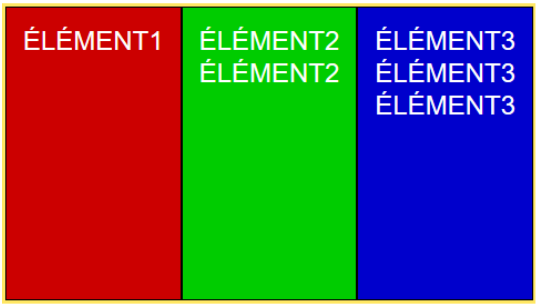
flex-direction :row ; *(valeur par défaut, horizontalement)*

flex-wrap :nowrap ; *(valeur par défaut)*

}

Ses valeurs :

flex-wrap : nowrap (par défaut) : les blocs sont alignés sans retour à la ligne quelle que soit leur largeur d'origine.



flex-wrap : wrap : s'il n'y a plus suffisamment de place horizontalement, les blocs passent à la ligne. La largeur d'origine est prise en compte.

flex-wrap : wrap-reverse : s'il n'y a plus suffisamment de place, les blocs passent à la ligne, les lignes sont organisées en ordre inverse

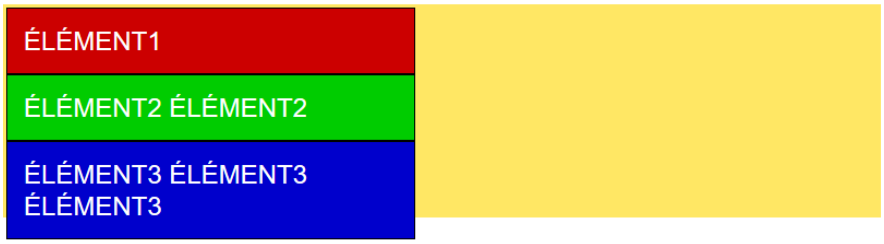
Si l'on veut attribuer une largeur aux éléments, utilisez flex-basis.

3.b. #flex-container {

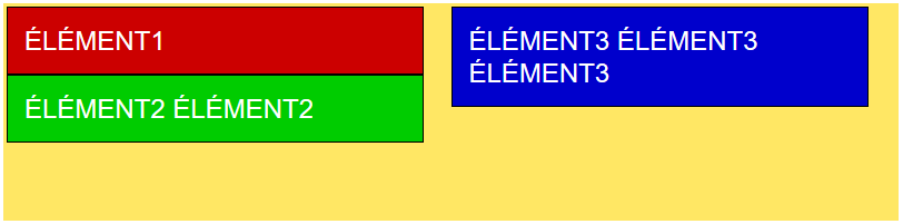
```
    display :flex ;
    flex-direction :column; (verticalement)
    flex-wrap :nowrap ; (valeur par défaut)
}
```

Ses valeurs :

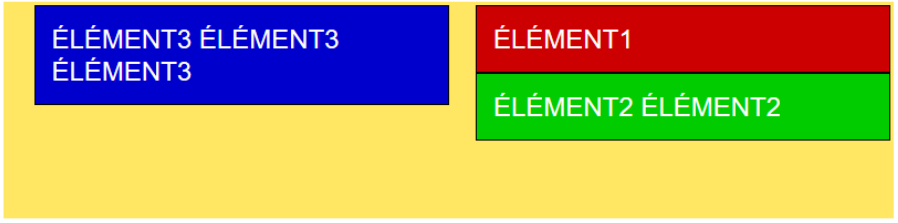
nowrap (par défaut)
Les blocs sont positionnés les uns en dessous des autres sans retour à la ligne quelle que soit leur hauteur d'origine.



wrap :
S'il n'y a plus suffisamment de place verticalement (hauteur du flex-container insuffisante), les blocs passent sur une seconde colonne



wrap-reverse : s'il n'y a plus suffisamment de place, les blocs passent sur une seconde colonne en ordre inverse



Si l'on veut attribuer une hauteur aux éléments, utilisez flex-basis.

2&3. La propriété flex-flow

Une propriété raccourcie : flex-flow permet d'indiquer en une seule déclaration les valeurs de flex-direction et flex-wrap.
Exemple flex-flow : row wrap ;

4. La propriété justify-content

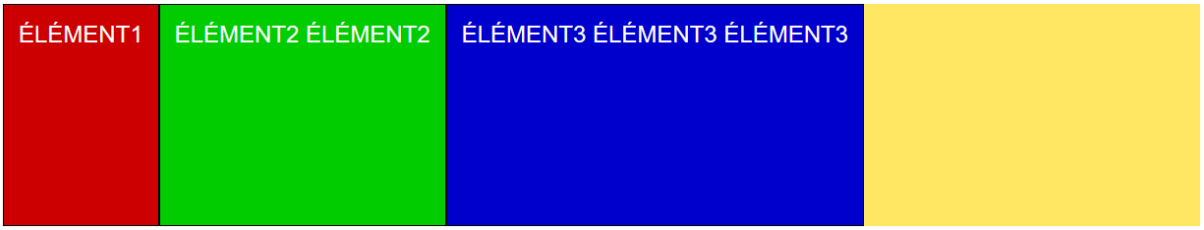
La propriété justify-content permet de spécifier la façon dont les blocs occupent l'espace sur l'axe principal quand l'espace n'est pas totalement comblé

4.a. #flex-container {

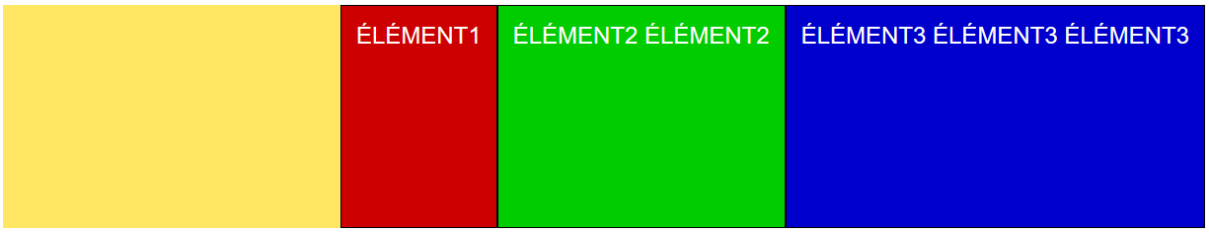
```
display :flex ;
flex-direction :row ; (valeur par défaut)
justify-content :flex-start ; (valeur par défaut)
}
```

Ses valeurs :

flex-start (par défaut). Les éléments sont alignés au début de l'axe horizontal.



flex-end. Les éléments sont alignés à la fin de l'axe horizontal.



center. Les éléments sont centrés sur l'axe horizontal.

space-between. Les éléments occupent tout l'axe horizontal, il y a des espaces entre eux.

space-around. Les éléments occupent tout l'axe horizontal, il y a des espaces entre les éléments et également en début et en fin de ligne.

4.b. #flex-container {

```
display :flex ;
flex-direction :column ;
justify-content :flex-start ; (valeur par défaut)
}
```

Ses valeurs :

flex-start (par défaut)
Les éléments sont alignés au sommet de l'axe vertical.

flex-end.
Les éléments sont alignés en bas de l'axe vertical.

center.
Les éléments sont centrés sur l'axe vertical.

space-between.
Les éléments occupent tout l'axe vertical, il y a des espaces entre eux.

space-around. Les éléments occupent tout l'axe vertical, il y a des espaces entre les éléments et également en début et en fin de ligne.

5. La propriété align-items

La propriété **align-items** permet de positionner les blocs selon un axe secondaire (*cross-axis*). L'axe secondaire est l'axe vertical si l'axe principal est l'axe horizontal (flex-direction :row) et inversement l'axe secondaire sera l'axe horizontal si l'axe principal a été défini par flex-direction :column.

5.a. #flex-container {

```
    display :flex ;
    flex-direction :row ; (valeur par défaut)
    align-items :stretch ; (valeur par défaut)
}
```

Ses valeurs :

stretch (par défaut)

center

5.b. #flex-container {

```
    display :flex ;
    flex-direction :column ;
    align-items :stretch ; (valeur par défaut)
}
```

stretch (par défaut)

center

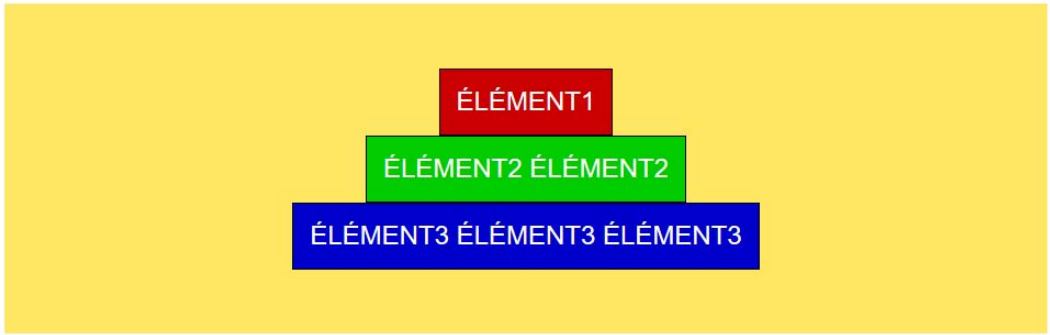
6. Centrer avec les propriétés justify-content:center; et align-items: center ;

6.a. Centrer horizontalement et verticalement avec **justify-content:center; et align-items: center ;** selon l'axe horizontal flex-direction :row ;

```
#flex-container {
    display :flex ;
    flex-direction :row ;
    justify-content:center ;
    align-items :center ;
}
```

6.b. Centrer horizontalement et verticalement avec **justify-content:center; et align-items: center ;** selon l'axe vertical avec flex-direction :column ;

```
#flex-container {
    display :flex ;
    flex-direction :column ;
    justify-content:center
    align-items :center ;
}
```

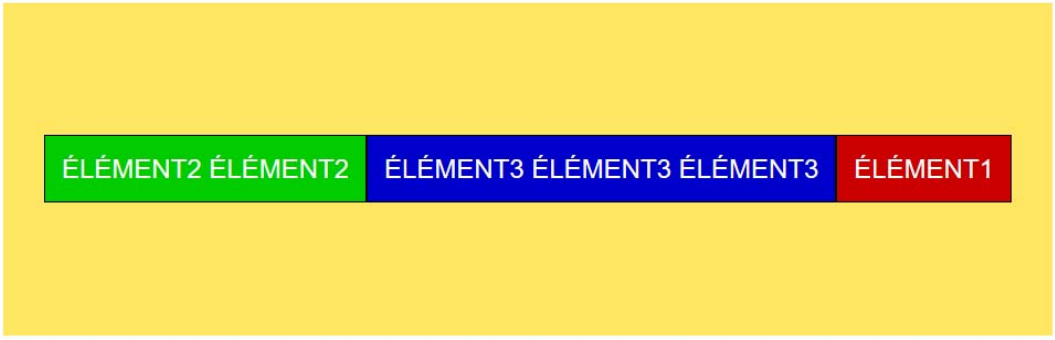


7. La propriété order

Il est possible de modifier l'ordre des éléments avec la propriété order dont la **valeur est un nombre** qui désigne la place de l'élément. Cette propriété s'applique sur l'élément (*enfant*).

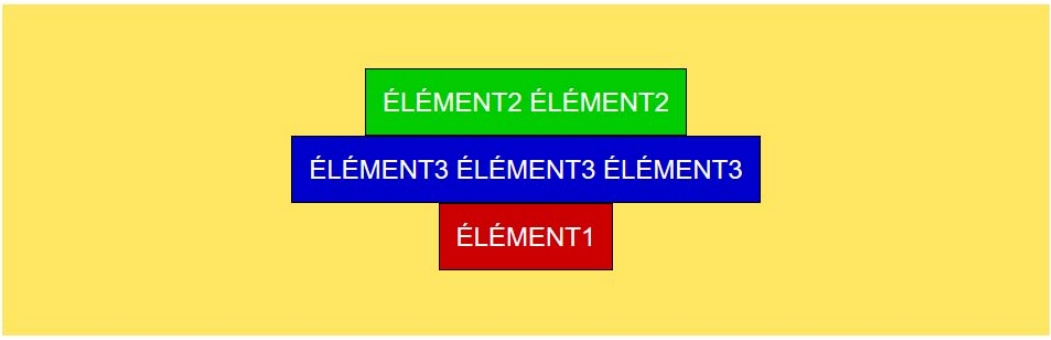
7.a. #flex-container {

```
        display :flex ;
        flex-direction :row ;
        justify-content:center
        align-items :center ;
}
.element :first-of-type {
        order :3 ;
}
.element :nth-of-type(2) {
        order :1 ;
}
.element :last-of-type {
        order :2 ;
}
```



7.b. `#flex-container {`

```
        display :flex ;
        flex-direction :column ;
        justify-content:center
        align-items :center ;
}
.element :first-of-type {
        order :3 ;
}
.element :nth-of-type(2) {
        order :1 ;
}
.element :last-of-type {
        order :2 ;
}
```



8. La propriété align-self

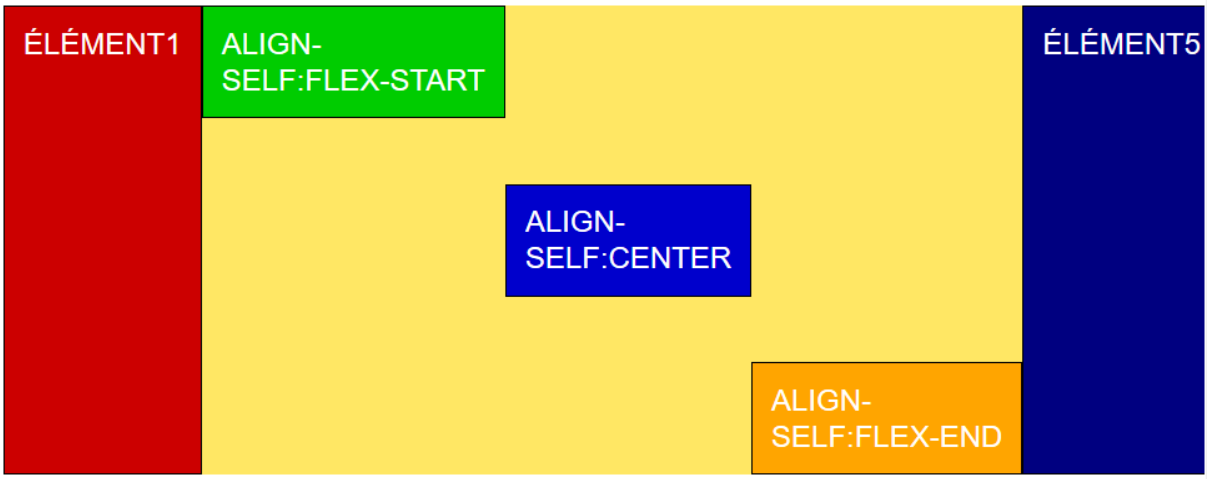
La propriété align-self appliqué sur un élément permet de décaler cet élément par rapport aux autres

Ses valeurs : flex-start et flex-end

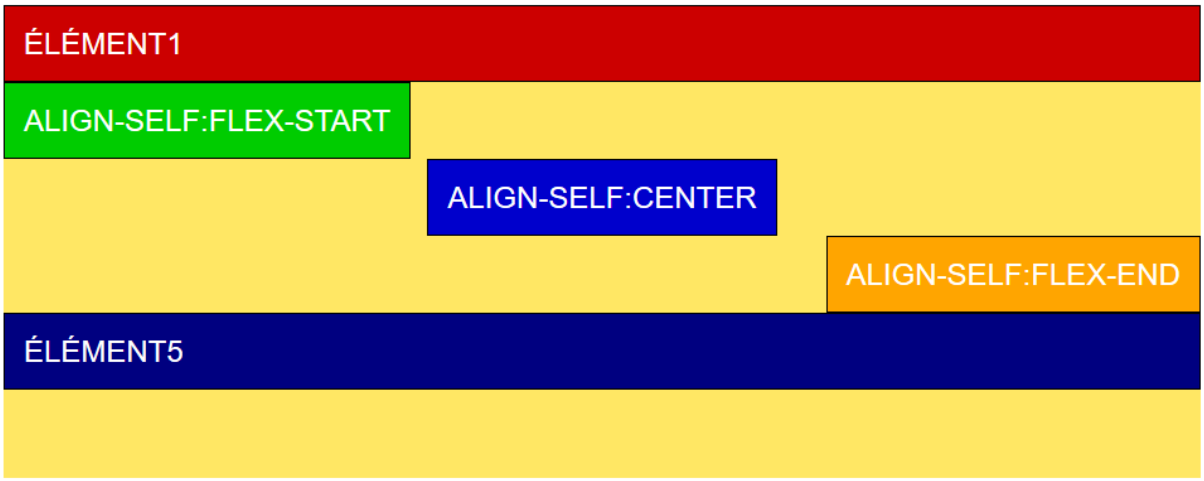
8.a. Centrer horizontalement et verticalement le flex-container avec **justify-content:center; et align-items: center;** selon l'axe horizontal : **flex-direction:row;** et modifier l'alignement d'un bloc avec **align-self**.

Ses valeurs : flex-start, center, flex-end

```
.element:nth-of-type(2) {
        background-color:#0c0; /* vert */
        align-self: flex-start; /* Seul ce bloc sera aligné en haut du flex-container */
                }
.element:nth-of-type(3) {
        background-color:#00c;  /* bleu */
        align-self: center; /* Seul ce bloc sera centré */
                }
.element:nth-of-type(4) {
        background-color:orange; /* orange */
        align-self: flex-end; /* Seul ce bloc sera aligné en bas du flex-container */
                }
```



8.b. Idem selon l'axe vertical avec **flex-direction:column;** et modifier l'alignement d'un bloc avec **align-self.**



9. La propriété flex.

La propriété **flex** est une propriété raccourcie qui définit la capacité d'un élément flexible à modifier ses dimensions afin de remplir l'espace disponible. Les éléments flexibles peuvent être étirés ou réduits pour utiliser un espace proportionnel à leur coefficient de grossissement ou de rétrécissement afin de ne pas dépasser d'un conteneur.

La propriété flex est la propriété raccourcie de trois propriétés :

- **flex-grow :** capacité pour un élément à s'étirer dans l'espace restant (valeur par défaut : 0). C'est sur cette propriété que l'on va travailler pour rendre les éléments flexibles. Il suffira de lui donner une valeur supérieure à zéro pour que l'élément ciblé occupe l'espace restant au sein du flex-container.
- **flex-shrink :** capacité pour un élément à se contracter si nécessaire (valeur par défaut : 1)
- **flex-basis :** taille initiale de l'élément avant que l'espace restant ne soit distribué (valeur par défaut : auto)

Par défaut les éléments occupent la largeur de leur contenu. Plusieurs éléments peuvent être rendus flexibles et se répartir l'espace restant. L'espace disponible est alors tout simplement distribué entre les éléments ciblés.

Pour rendre un élément flexible, il suffit de lui attribuer une valeur de flex-grow (ou flex en raccourci) supérieure à zéro. Cet élément occupera alors l'espace restant au sein du flex-container

#container1. flex:initial (valeur par défaut sur tous les éléments)

#container2. flex:1 / le dernier élément

#container3. flex:1 / les 4e et dernier élément

#container4. flex:1 / tous les éléments

#container5. flex-basis:150px; / tous les éléments

#container6. flex-basis:60px / tous les éléments et flex-grow:1 / élément 2