Possibilités principales sont :

1. Distribution des éléments horizontale ou verticale, avec passage à la ligne autorisé ou non,
2. Alignements et centrages horizontaux et verticaux, justifiés, répartis,
3. Réorganisation des éléments indépendamment de l'ordre du flux (DOM),
4. Gestion des espaces disponibles (fluidité).

Ses enfants deviennent alors automatiquement (inutile de leur déclarer quoi que ce soit) des éléments de type "flex-item" :

Un conteneur et ses éléments

Le conteneur est une balise HTML, et les éléments sont d'autres balises HTML à l'intérieur :

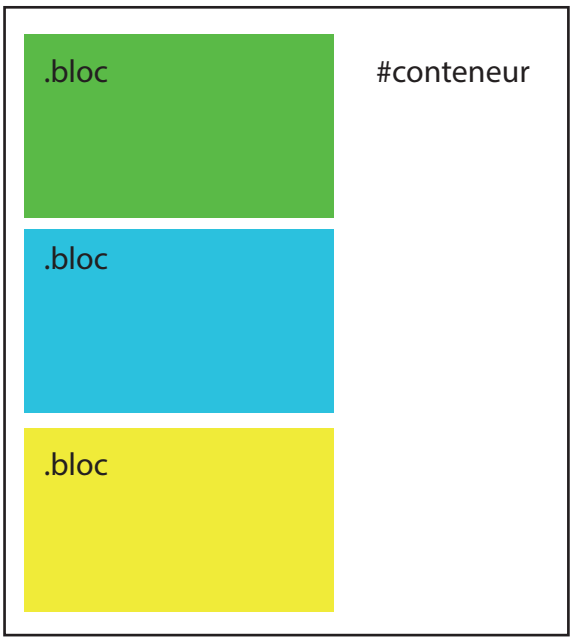
en HTML

```
<div id="conteneur">
   <div class="bloc">Elément 1</div>
   <div class="bloc">Elément 2</div>
   <div class="bloc>Elément 3</div>
</div>
```

=

Resultat

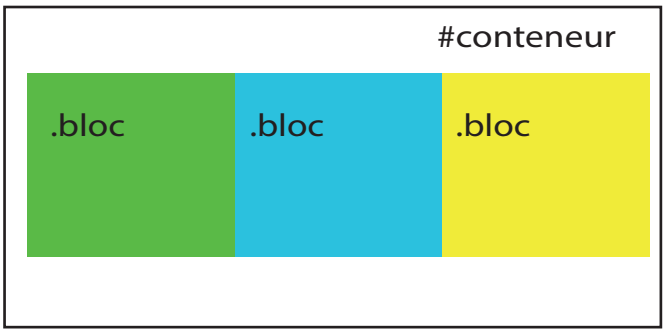
Vos blocs vont se mettre les uns en-dessous des autres



EN CSS

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

=



... alors les blocs se placent par défaut côte à côte.

Direction Avec flex-direction

Flexbox nous permet d'agencer ces éléments dans le sens que l'on veut. Avec flex-direction , on peut les positionner verticalement ou encore les inverser. Il peut prendre les valeurs suivantes :

row : organisés sur une ligne (par défaut)

column : organisés sur une colonne

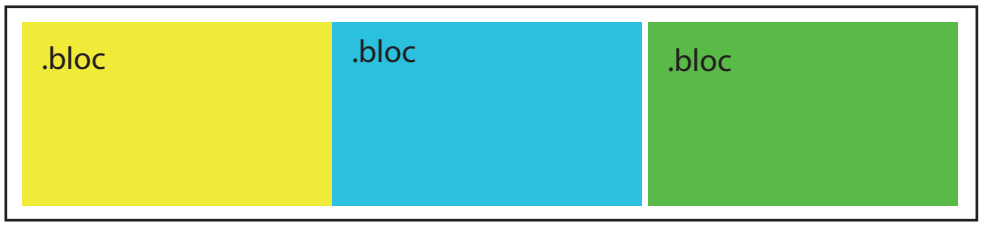
row-reverse : organisés sur une ligne, mais en ordre inversé

column-reverse : organisés sur une colonne, mais en ordre inversé

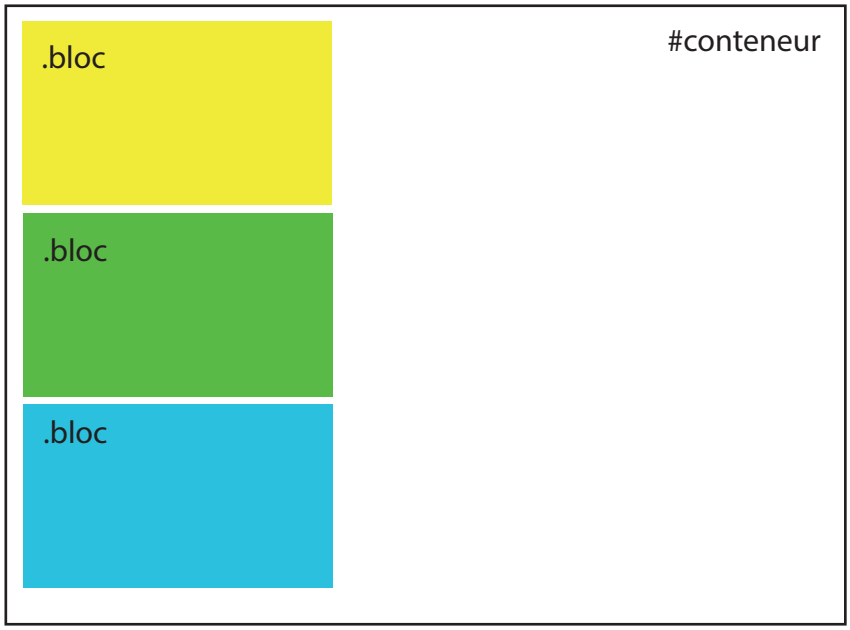
```
#conteneur {
  display: flex;
  flex-direction: column-reverse;
  }
```

Voici l'effet que prennent les différentes valeurs sur une même illustration :

flex-direction: row;



flex-direction: column;



Distribution **flex-wrap**

Par défaut, les blocs essaient de rester sur la même ligne (ce qui peut provoquer des bugs).
Si vous voulez, vous pouvez demander à ce que les blocs aillent à la ligne lorsqu'ils n'ont plus la place avec flex-wrap qui peut prendre ces valeurs :

nowrap : pas de retour à la ligne (par défaut)

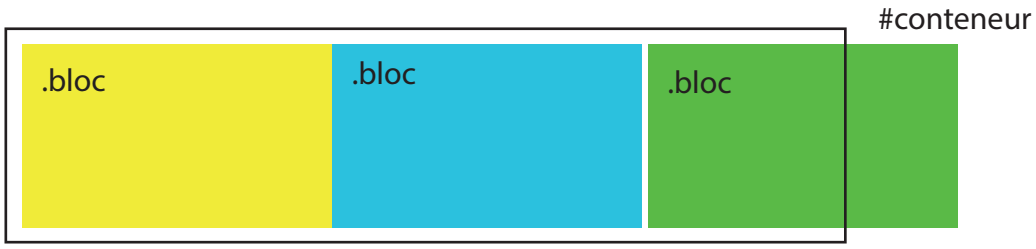
wrap : les éléments vont à la ligne lorsqu'il n'y a plus la place

wrap-reverse : les éléments vont à la ligne lorsqu'il n'y a plus la place en sens inverse

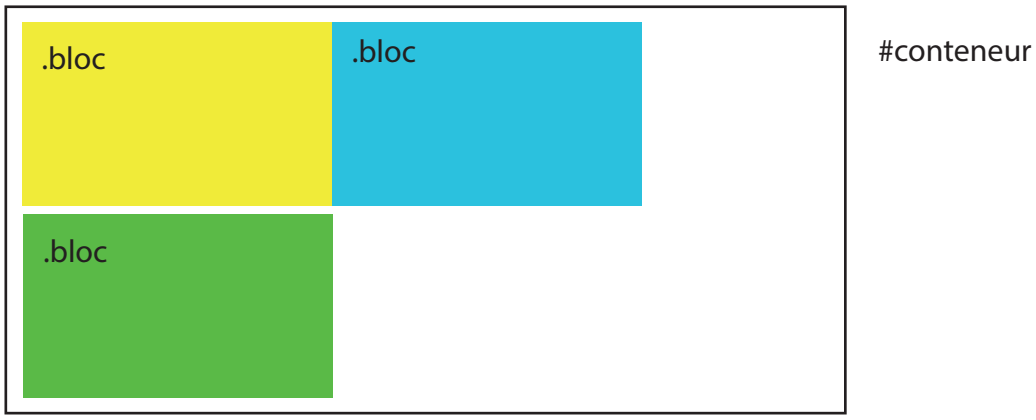
```
#conteneur {

display: flex;
flex-wrap: wrap

}
```

Voici l'effet que prennent les différentes valeurs sur une même illustration :

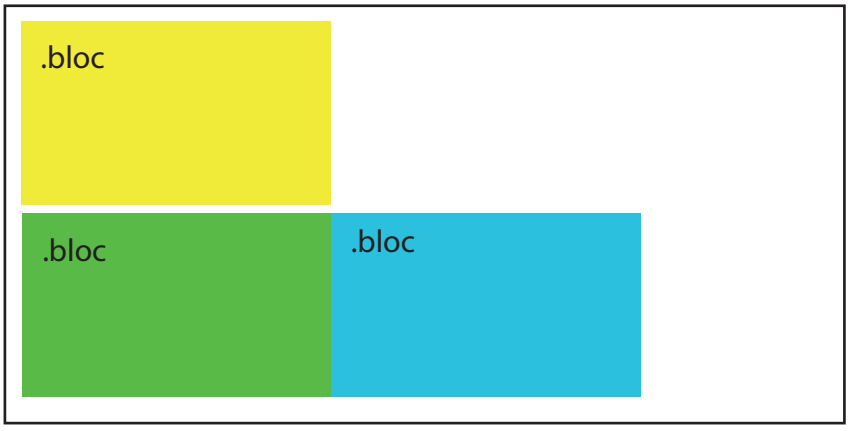
flex-wrap: nowrap



flex-wrap: wrap



flex-wrap: wrap-reserve



Aligner sur l'axe principal **justify-content**

Pour changer leur alignement, on va utiliser justify-content , qui peut prendre ces valeurs :

flex-start : alignés au début (par défaut)

flex-end : alignés à la fin

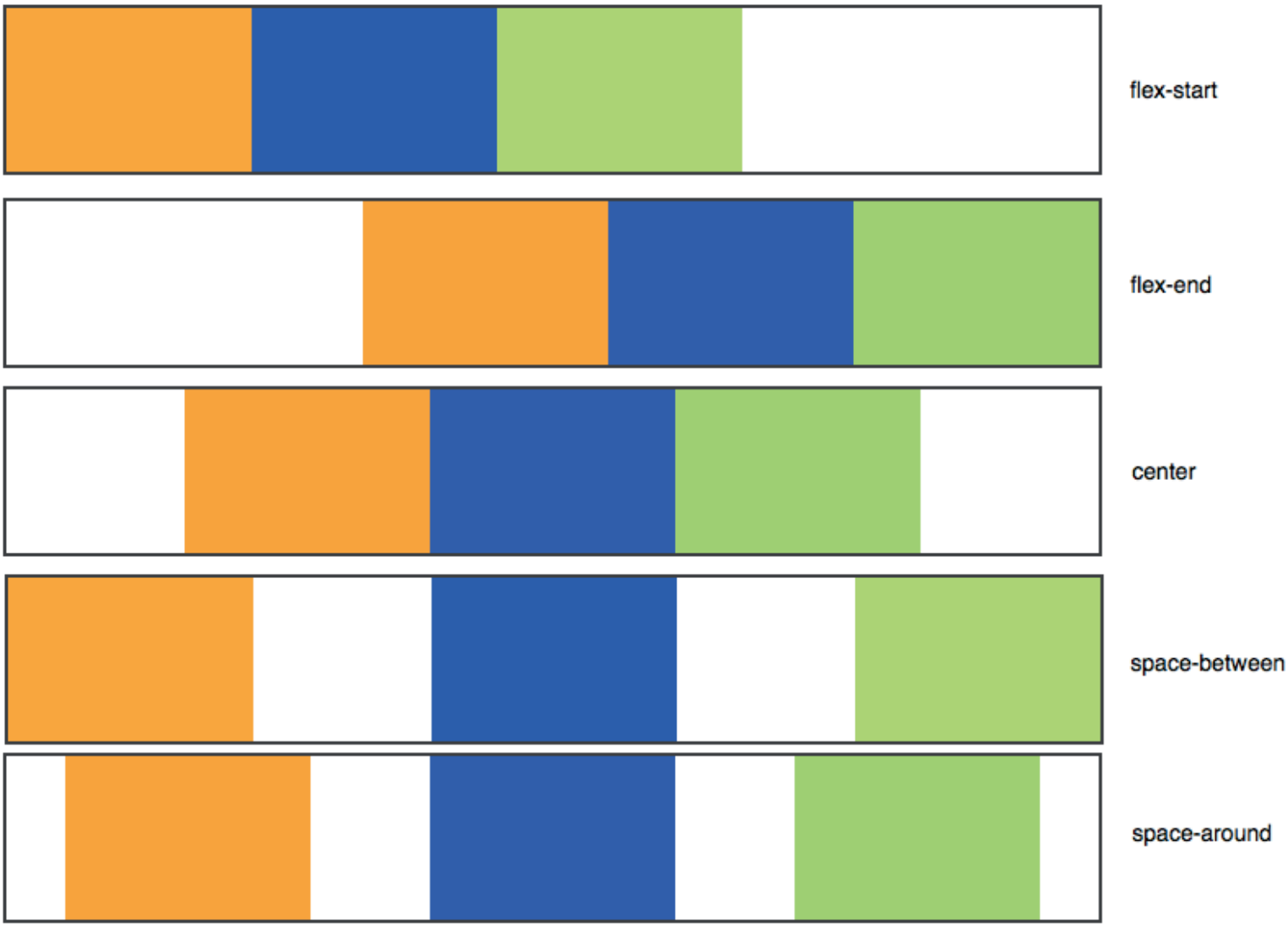
center : alignés au centre

space-between : les éléments sont étirés sur tout l'axe (il y a de l'espace entre eux)

space-around : idem, les éléments sont étirés sur tout l'axe, mais ils laissent aussi de l'espace sur les extrémité

```
#conteneur {
 display: flex;
 justify-content: space-around;
 }
```

Voici l'effet que prennent les différentes valeurs sur une même illustration :



cela fonctionne aussi si vos éléments s ont dans une direction verticale.

Aligner sur l'axe secondaire **align-items**

Avec align-items , nous pouvons changer leur alignement sur l'axe secondaire. Il peut prendre ces valeurs :

stretch : les éléments sont étirés sur tout l'axe (valeur par défaut)

flex-start : alignés au début

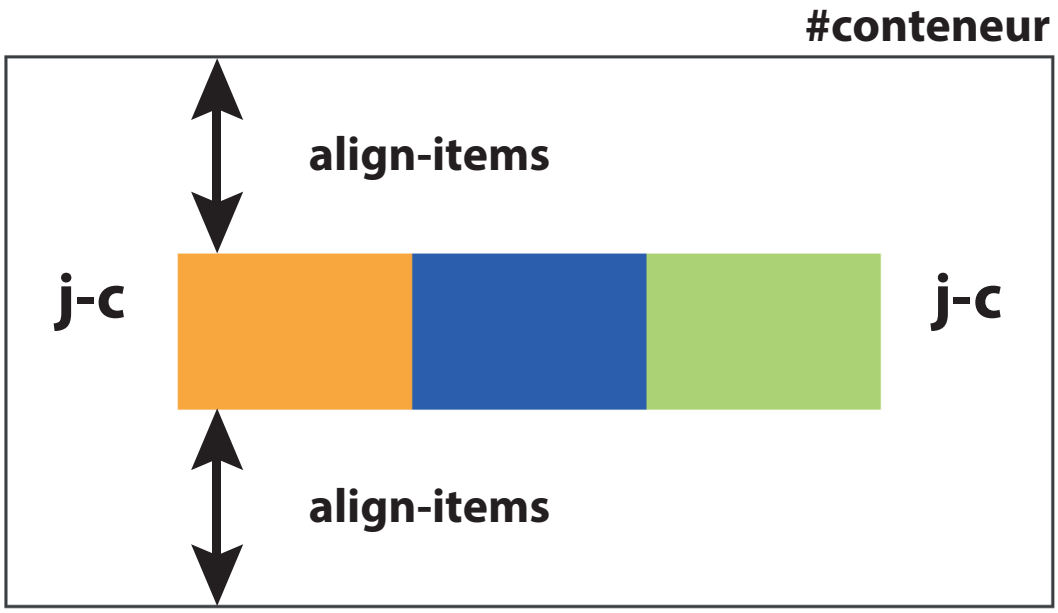
flex-end : alignés à la fin

center : alignés au centre

baseline : alignés sur la ligne de base (semblable à flex-start)

```
#conteneur {
 display: flex;
 justify-content: center;
 align-items: center
 }
```

Voici l'effet que prennent les différentes valeurs sur une même illustration :



Un alignement sur l'axe secondaire avec align-items nous permet de centrer complètement l'élément dans le conteneur !

Aligner un seul élément **avec align-self :**

Il est possible de faire une exception pour un seul des éléments sur l'axe secondaire avec align-self :

```
#conteneur

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


.bloc:nth-child(2) /* On prend le deuxième bloc */

{	background-color: blue;
	align-self: flex-end; /* Seul ce bloc sera aligné à la fin */
}
```

Plusieurs lignes **avec align-content .**

Si vous avez plusieurs lignes dans votre Flexbox, vous pouvez choisir comment celles-ci seront réparties avec align-content .

Cette propriété n'a aucun effet s'il n'y a qu'une seule ligne dans la Flexbox.

Prenons donc un cas de figure où nous avons plusieurs lignes. Je vais rajouter des éléments :

```
<div id="conteneur">

<div class="element"></div>
<div class="element"></div>
<div class="element"></div>

 <div class="element"></div>
<div class="element"></div>
<div class="element"></div>

<div class="element"></div>
<div class="element"></div>
<div class="element"></div>
</div>
```

J'autorise mes éléments à aller à la ligne avec flex-wrap :

Jusque-là, rien de vraiment nouveau.
Voyons voir comment les lignes se répartissent différemment avec la nouvelle propriété align-content que je voulais vous présenter. Elle peut prendre ces valeurs :

flex-start : les éléments sont placés au début
flex-end : les éléments sont placés à la fin
center : les éléments sont placés au centre
space-between : les éléments sont séparés avec de l'espace entre eux
space-around : idem, mais il y a aussi de l'espace au début et à la fin
stretch (par défaut) : les éléments s'étirent pour occuper tout l'espace

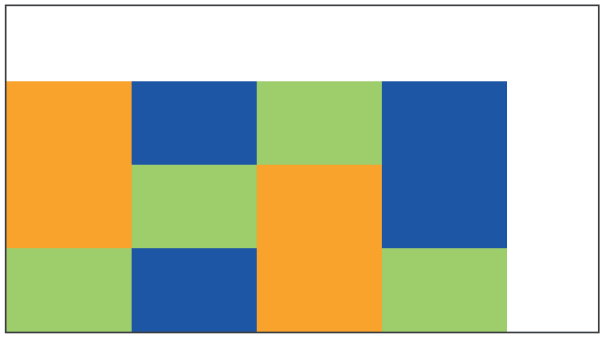
Voici ce que donnent les différentes valeurs suite page 8:

Plusieurs lignes **avec align-content .**

Voici ce que donnent les différentes valeurs suite de la page 7

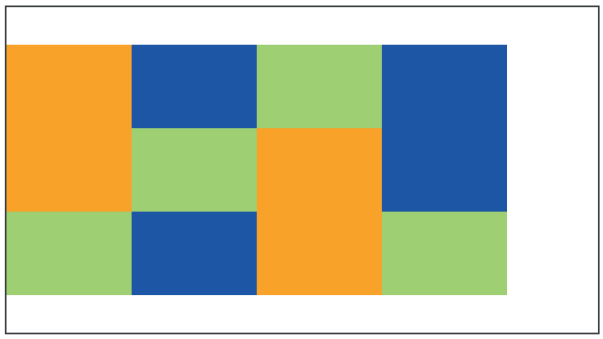
flex-start

space-between



flex-end

space-around



center

stretch