WEB DESIGN

La propriété «Flexbox»

Au début, les webmasters utilisaient des tableaux HTML pour faire la mise en page.

Puis, CSS est apparu et on a commencé à faire une mise en page à l'aide de la propriété float.

Aujourd'hui, une autre technique existe : Flexbox ! Elle permet toutes les folies (ou presque) et c'est celle que je vous recommande d'utiliser si vous en avez la possibilité, lorsque vous créez un nouveau site.

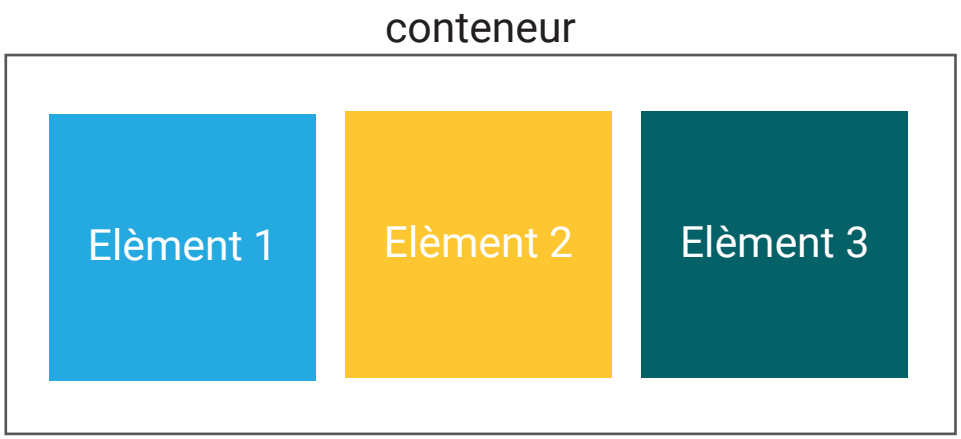
Flexbox est désormais reconnu par tous les navigateurs récents !

Un conteneur, des éléments

Le principe de la mise en page avec Flexbox est simple : vous définissez un conteneur, et à l'intérieur vous placez plusieurs éléments. Imaginez un carton dans lequel vous rangez plusieurs objets : c'est le principe !

Sur une même page web, vous pouvez sans problème avoir plusieurs conteneurs . Ce sera à vous d'en créer autant que nécessaire pour obtenir la mise en page que vous voulez.

Commençons par étudier le fonctionnement d'un conteneur.



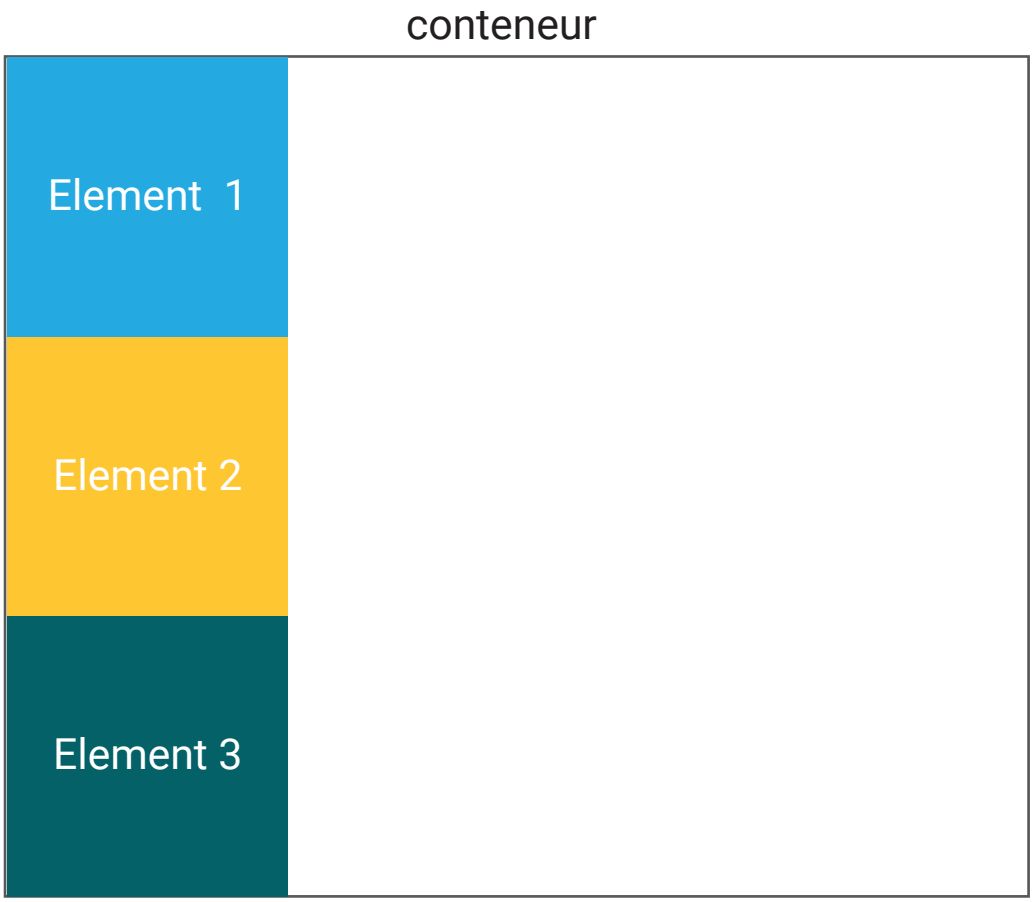
WEB DESIGN

La proriété « Flexbox»

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

```
12 ▼ <body>
13 ▼     <section class="conteneur">
14           <div class="element">Element 1</div>
15           <div class="element">Element 2</div>
16           <div class="element">Element 3</div>
17       </section>
18   </body>
```

Mes éléments vont se mettre les uns en-dessous des autres !!! jusque la c'est le comportement normal n'ayant pas encore travailler la CSS



WEB DESIGN

La propriété « Flexbox»

Flexbox

Découvrons maintenant Flexbox. Si je mets une (une seule !) propriété CSS, tout change. Cette propriété, c'est «flex», et je l'applique au conteneur :

```
49 ▼ section.conteneur {
50       display: flex;
51   }
```

Un coup de flex et les blocs se positionnent côte à côte !

La 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

WEB DESIGN

La propriété « Flexbox»

```
72 ▼ section.conteneur1 {
73       margin: 30px auto;
74       display: flex;
75       flex-direction: column;
```

```
79 ▼ section.conteneur2 {
80       margin: 30px auto;
81       display: flex;
82       flex-direction: column-reverse;
```

WEB DESIGN

La propriété « Flexbox»

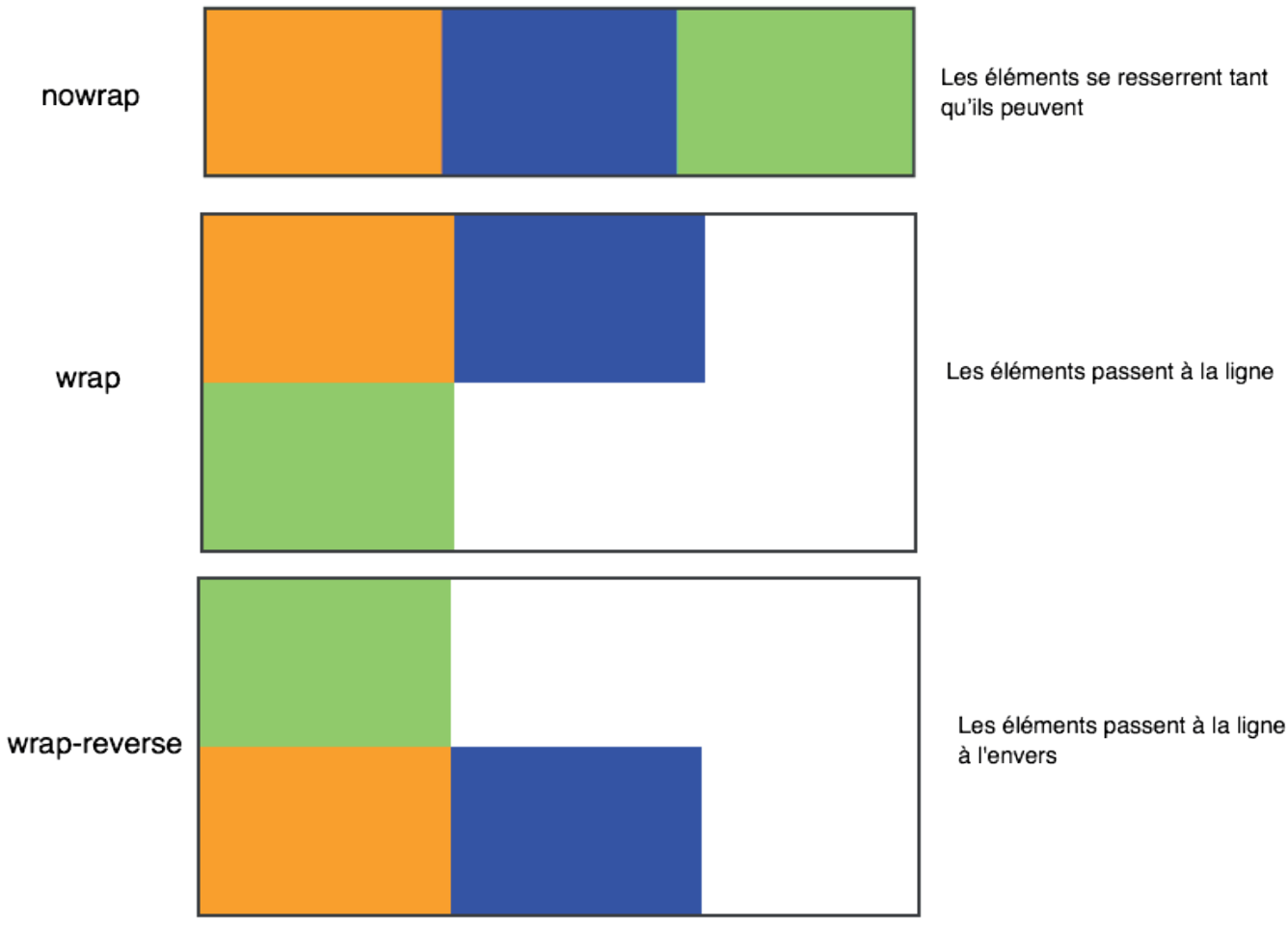
Le retour à la ligne

Par défaut, les blocs essaient de rester sur la même ligne s'ils n'ont pas la place (ce qui peut provoquer des bugs de design parfois). 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

```
99  section.conteneur6 {
100     margin: 30px auto;
101     display: flex;
102     flex-wrap: wrap;
```

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



WEB DESIGN

La propriété « Flexbox»

Alignez-les !

Reprenons. Les éléments sont organisés soit horizontalement (par défaut), soit verticalement. Cela définit ce qu'on appelle l'axe principal. Il y a aussi un axe secondaire (cross axis) :

Si vos éléments sont organisés horizontalement, l'axe secondaire est l'axe vertical.
Si vos éléments sont organisés verticalement, l'axe secondaire est l'axe horizontal.
Pourquoi je vous raconte ça ? Parce que nous allons découvrir comment aligner nos éléments sur l'axe principal et sur l'axe secondaire.

Aligner sur l'axe principal

Pour faire simple, partons sur des éléments organisés horizontalement (c'est le cas par défaut).

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

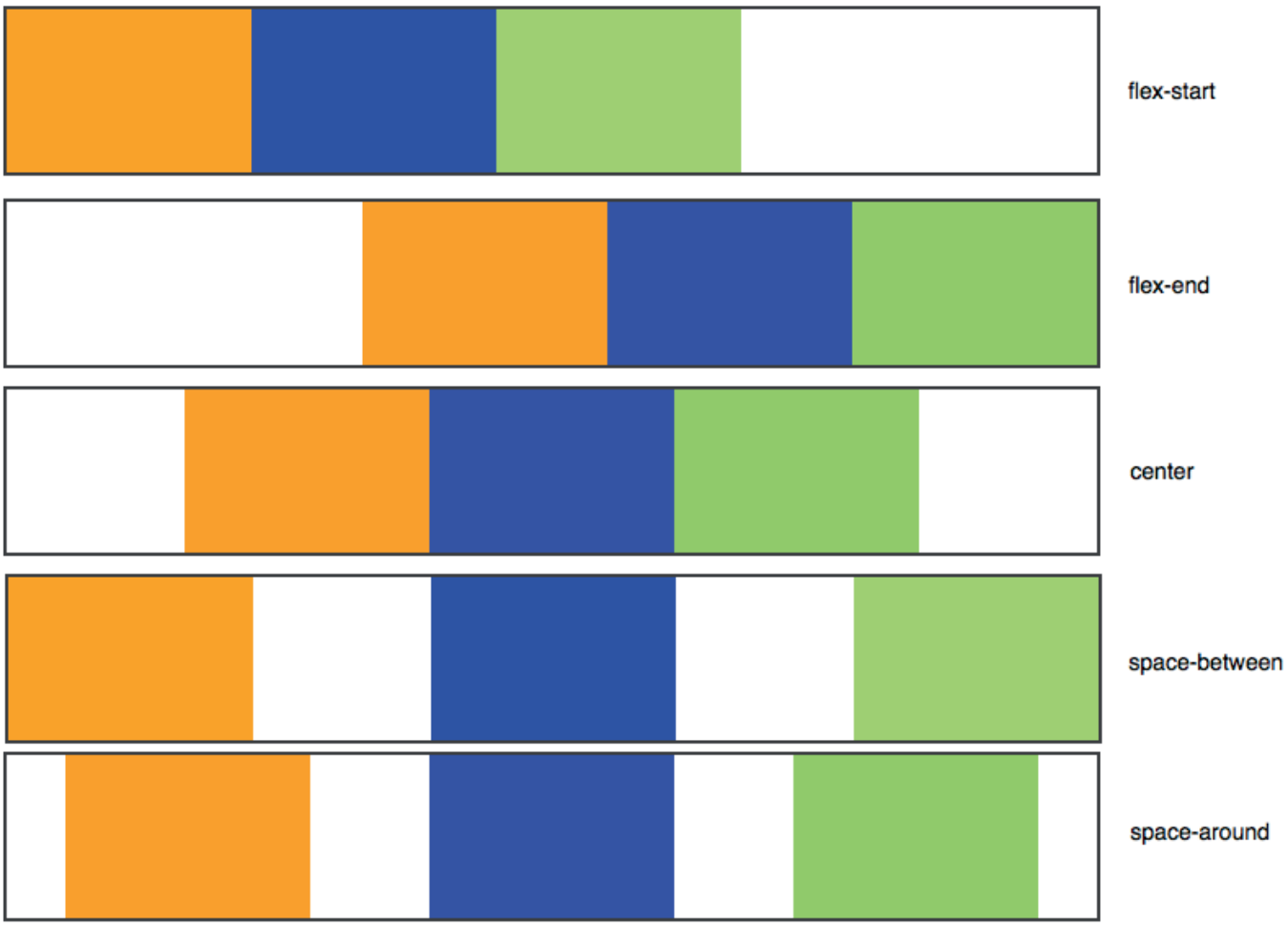
flex-start : alignés au début (par défaut)
flex-end : alignés à la fin
center : alignés au centre
space-between : les éléments sont étirés sur tout l'axe (il y a de l'espace entre eux)
space-around : idem, les éléments sont étirés sur tout l'axe, mais ils laissent aussi de l'espace sur les extrémités

```
65 ▼ section.conteneur {
66       margin: 30px auto;
67       display: flex;
68       justify-content: space-around;
```

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

WEB DESIGN

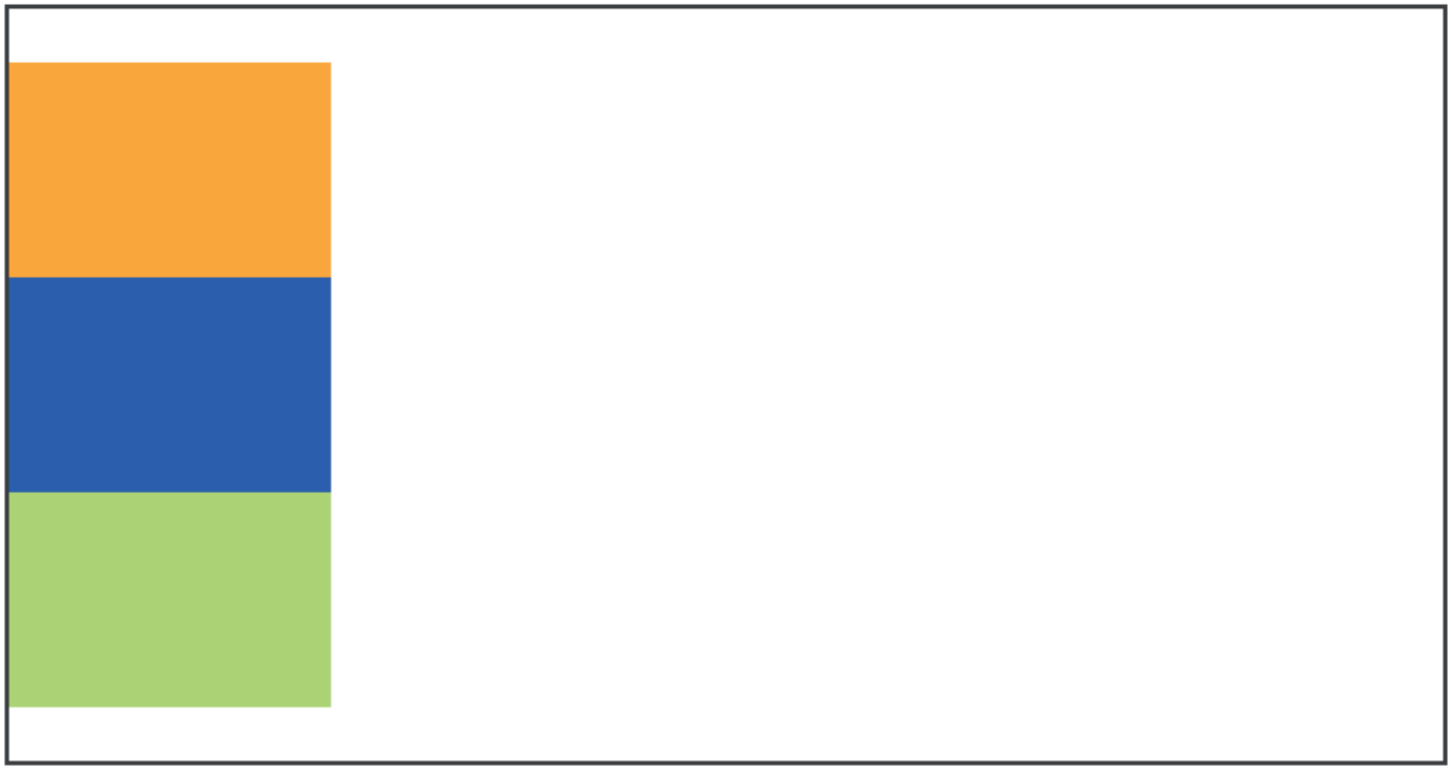
La propriété « Flexbox»



Maintenant, voici ce qu'il faut bien comprendre : ça marche aussi si vos éléments sont dans une direction verticale. Dans ce cas, l'axe vertical devient l'axe principal, et «justify-content» s'applique aussi :

```
section.conteneur {
    display: flex;
    flex-direction: column;
    justify-content: center;
    height: 350px; /* Un peu de hauteur pour que les éléments aient la place de bouger
*/
}
```

WEB DESIGN
La propriété « Flexbox»



Avec une direction verticale (column), le centrage fonctionne de la même façon cette fois en hauteur !

Aligner sur l'axe secondaire

Comme je vous disais, si nos éléments sont placés dans une direction horizontale (ligne), l'axe secondaire est... vertical. Et inversement, si nos éléments sont dans une direction verticale (colonne), l'axe secondaire est horizontal.

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

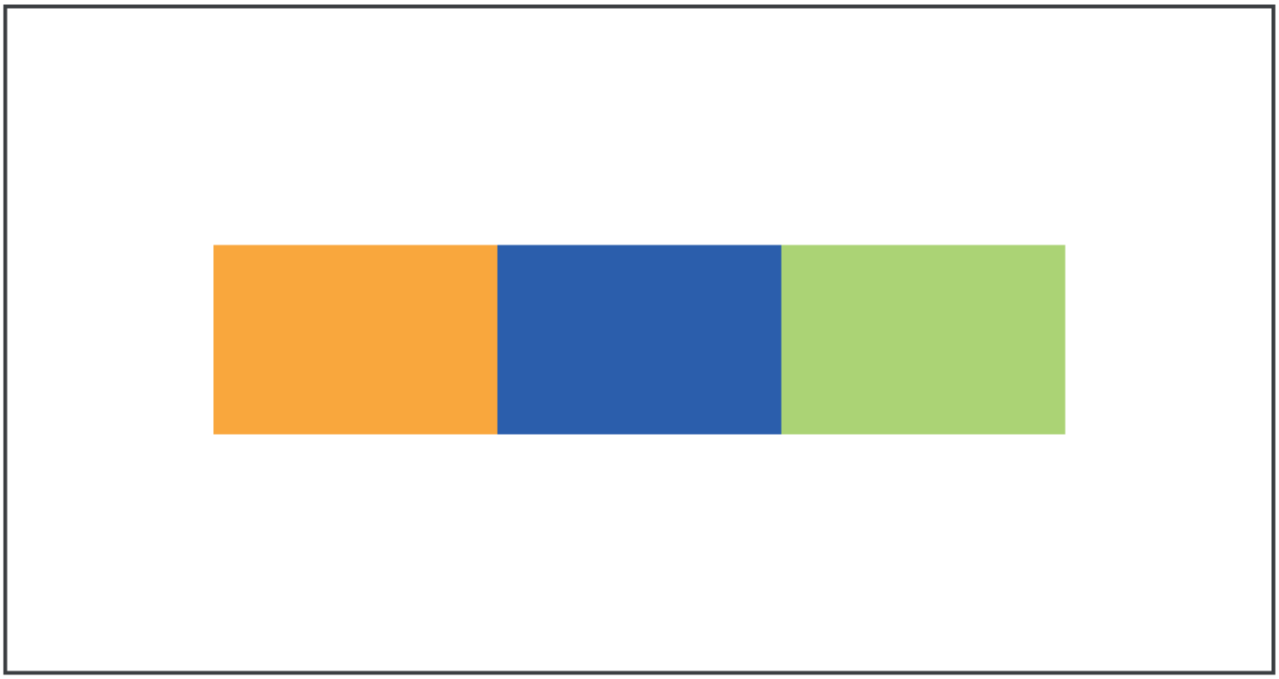
stretch : les éléments sont étirés sur tout l'axe (valeur par défaut)
flex-start : alignés au début
flex-end : alignés à la fin
center : alignés au centre
baseline : alignés sur la ligne de base (semblable à flex-start)

Pour ces exemples, nous allons partir du principe que nos éléments sont dans une direction horizontale (mais n'hésitez pas à tester aussi dans la direction verticale !).

WEB DESIGN

La propriété « Flexbox»

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



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

Info

le centrage vertical et horizontal peut d'ailleurs être obtenu encore plus facilement. Dites que votre conteneur est une flexbox et établissez des marges automatiques sur les éléments à l'intérieur. C'est tout !

```
48 ▼ section.conteneur {
49       display: flex;
50   }
51 ▼ .element {
52       margin: auto;
53   }
```

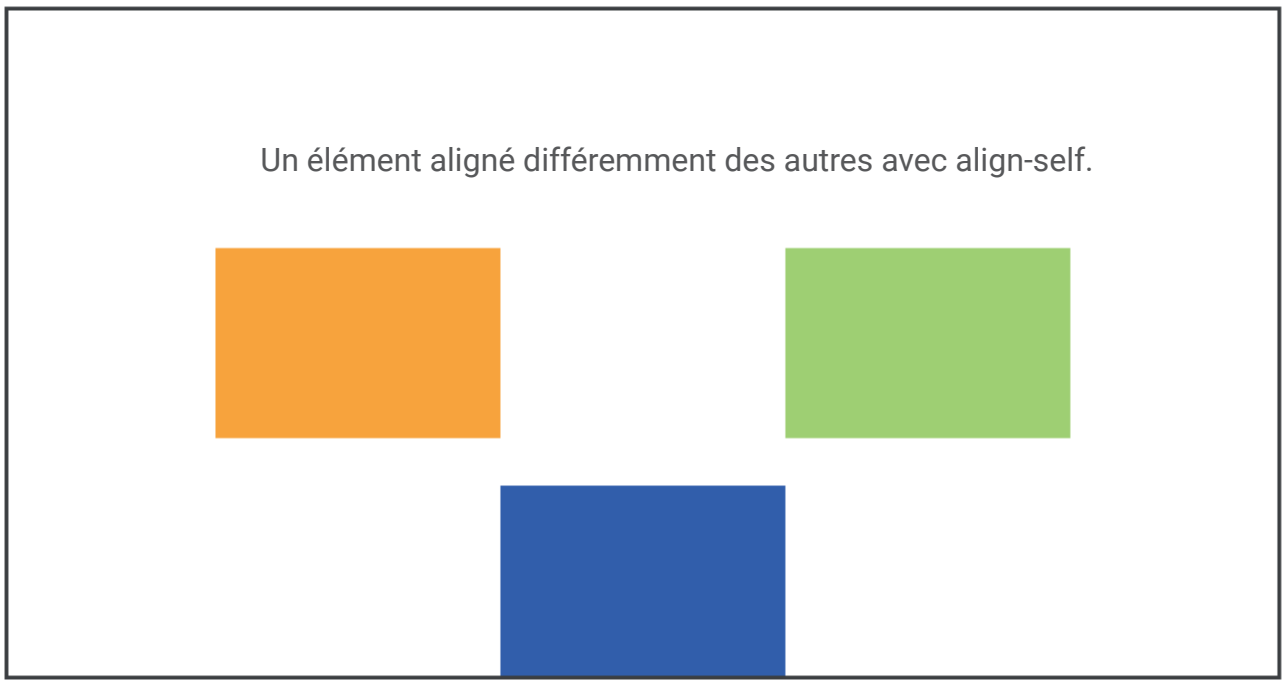
WEB DESIGN

La propriété « Flexbox»

Aligner un seul élément

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

```
section.conteneur {
    display: flex;
    flex-direction: row;
    justify-content: center;
    align-items: center;

}
.element {
    margin: auto;

}

.element:nth-child(1) {
    background-color: cadetblue;
}

.element:nth-child(2) {  /* On prend le deuxième bloc élément */
    background-color: powderblue;
    align-self: flex-end; /* Seul ce bloc sera aligné à la fin */

}
.element:nth-child(3) {
    background-color: deepskyblue;
}
```



WEB DESIGN

La propriété « Flexbox»

Répartir plusieurs lignes

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 :

```
12 ▼        <section class="conteneur">
13              <div class="element"></div>
14              <div class="element"></div>
15              <div class="element"></div>
16              <div class="element"></div>
17              <div class="element"></div>
18              <div class="element"></div>
19              <div class="element"></div>
20              <div class="element"></div>
21              <div class="element"></div>
22          </section>
```

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

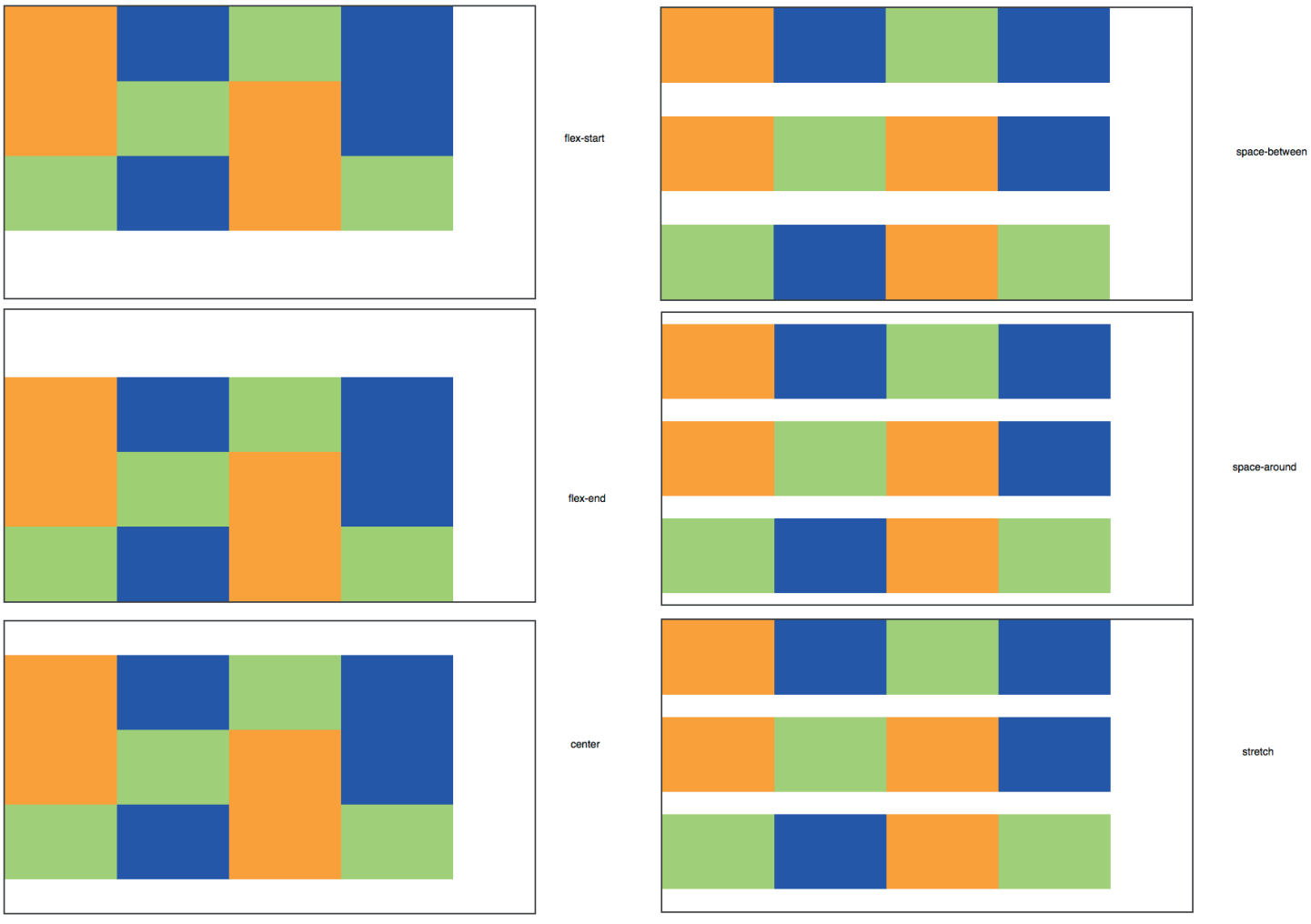
WEB DESIGN

La propriété « Flexbox»

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 :



Les lignes sont placées différemment avec align-content

WEB DESIGN

La propriété « Flexbox»

Rappel à l'ordre

Sans changer le code HTML, nous pouvons modifier l'ordre des éléments en CSS grâce à la propriétéorder. Indiquez simplement un nombre, et les éléments seront triés du plus petit au plus grand nombre.

Reprenons une simple ligne de 3 éléments :

```
section.conteneur {
    display: flex;
}
```

Si je dis que le premier élément sera placé en 3e position, le second en 1ère position et le troisième en 2nde position, l'ordre à l'écran change !

```
}
.element:nth-child(1) {
    background-color: cadetblue;
    order: 3;
}
.element:nth-child(2) {
    background-color: powderblue;
    order: 1;
}
.element:nth-child(3) {
    background-color: deepskyblue;
    order: 2;
}
```

Avec order, nous pouvons réordonner les éléments en CSS

WEB DESIGN

La propriété « Flexbox»

Encore plus flex : faire grossir ou maigrir les éléments

Allez encore une dernière technique, après on passe à la pratique. ;)

Avec la propriété **«flex»** , nous pouvons permettre à un élément de grossir pour occuper tout l'espace restant.

```
52 ▼ .element:nth-child(2) {
53       background-color: powderblue;
54       flex: 1;
55   }
```

Le second élément s'étire pour prendre tout l'espace

Le nombre que vous indiquez à la propriété flex indique dans quelle mesure il peut grossir par rapport aux autres.

```
48 ▼ .element:nth-child(1) {
49       background-color: cadetblue;
50       flex: 2;
51   }
52 ▼ .element:nth-child(2) {
53       background-color: powderblue;
54       flex: 1;
55   }
```

Ici, le premier élément peut grossir 2 fois plus que le second élément :

Le premier élément peut grossir deux fois plus que le second élément

WEB DESIGN

La propriété « Flexbox»

La propriété **«flex»** est en fait une super-propriété qui combine **«flex-grow»** (capacité à grossir), **«flex-shrink»** (capacité à maigrir) et **«flex-basis»** (taille par défaut). J'utilise simplement «flex» comme je vous l'ai montré ici, mais si vous voulez en savoir plus, je vous invite à vous renseigner sur ces autres propriétés.

RESUMER

Il existe plusieurs techniques pour positionner les blocs sur la page. Flexbox est la technique la plus récente et de loin la plus puissante, que je vous recommande d'utiliser.

Le principe de Flexbox est d'avoir un conteneur, avec plusieurs éléments à l'intérieur. Avec «display: flex;» sur le conteneur, les éléments à l'intérieur sont agencés en mode Flexbox (horizontalement par défaut).

Flexbox peut gérer toutes les directions. Avec «flex-direction» , on peut indiquer si les éléments sont agencés horizontalement (par défaut) ou verticalement. Cela définit ce qu'on appelle l'axe principal.

L'alignement des éléments se fait sur l'axe principal avec «justify-content» , et sur l'axe secondaire avec «align-items.»

Avec «flex-wrap», on peut autoriser les éléments à revenir à la ligne s'ils n'ont plus d'espace.

S'il y a plusieurs lignes, on peut indiquer comment les lignes doivent se répartir entre elles avec «align-content».

Chaque élément peut être réagencé en CSS avec «order» (pas besoin de toucher au code HTML !).

Avec la super-propriété «flex», on peut autoriser nos éléments à occuper plus ou moins d'espace restant.