Aligner avec Flexbox

Placer deux blocs l'un à côté de l'autre a longtemps été le cauchemar du web. Trois lignes de CSS suffisent aujourd'hui.

4TTR Découverte

Les éléments **bloc** s'empilent verticalement : chacun prend toute la largeur et pousse le suivant vers le bas. C'est pratique pour un texte, mais impossible pour une barre de navigation ou trois cartes côte à côte.

Pendant quinze ans, les développeurs ont contourné le problème avec des bricolages. Puis **Flexbox** est arrivé.

🎯 Objectifs

À la fin de ce cours, tu seras capable de :

1. Transformer un conteneur en boîte flexible.
2. Espacer ses enfants avec `gap`.
3. Répartir les éléments horizontalement avec `justify-content`.
4. Les aligner verticalement avec `align-items`.

Le principe : la règle va sur le parent

C'est le point qui déroute au début. On ne dit pas aux éléments de se ranger : on dit au **conteneur** de prendre la main sur ses enfants.

```html
<nav>
  <a href="#">Introduction</a>
  <a href="#">Les virus</a>
  <a href="#">Phishing</a>
</nav>
```

```css
nav {
  display: flex;
  gap: 25px;
}
```

La règle porte sur `nav`, le parent — **pas** sur `nav a`. Dès qu'un conteneur passe en `display: flex`, ses enfants directs se rangent sur une ligne horizontale, et c'est lui qui décide de leur placement.

© 2025 Mr. Lambrechts

> ⚠️ **ATTENTION :** `display: flex` n'agit que sur les **enfants directs**. Un élément situé deux niveaux plus bas n'est pas concerné.

gap : l'espace entre les éléments

```
nav {
  display: flex;
  gap: 25px;
}
```

`gap` fixe l'espace entre les enfants. Une seule ligne, et plus besoin de bricoler des marges sur chaque élément — ni de se demander quoi faire du dernier, qui ne doit pas avoir de marge à droite.

> 💡 **ASTUCE :** `gap` ne s'applique qu'**entre** les éléments, jamais sur les bords du conteneur. Pour de l'air autour, c'est le `padding` du conteneur.

justify-content : répartir sur la ligne

Une fois le conteneur en `flex`, `justify-content` décide de la répartition horizontale.

Valeur	Effet
`flex-start`	Tout à gauche — c'est le comportement par défaut
`center`	Groupés au centre
`flex-end`	Tout à droite
`space-between`	Le premier à gauche, le dernier à droite, l'espace réparti entre eux

```
nav {
  display: flex;
  gap: 25px;
```

```
    justify-content: center;
    background-color: SteelBlue;
    padding: 12px;
  }
```

`space-between` est celui qu'on utilise pour un en-tête de site : le logo à gauche, le menu à droite, et rien à calculer.

Chacune de ces mises en page demandait autrefois des heures de tâtonnement. Aujourd'hui, c'est un mot à changer.

align-items : l'autre direction

`justify-content` place les éléments sur la ligne. `align-items` les place **en travers** de la ligne, c'est-à-dire verticalement.

```
nav {
  display: flex;
  align-items: center;
}
```

C'est ce qui permet à un logo haut de 40 pixels et à un menu haut de 20 pixels d'être alignés sur leur milieu plutôt que sur leur bord supérieur.

Valeur	Effet
`stretch`	Les enfants s'étirent sur toute la hauteur — par défaut
`center`	Alignés sur leur milieu
`flex-start`	Alignés en haut
`flex-end`	Alignés en bas

> 🧠 **À RETENIR :** `justify-content` agit dans le sens de la ligne, `align-items` en travers. Les deux se combinent, et `align-items: center` est de loin la valeur la plus utilisée.

Passer à la ligne

Trois cartes tiennent côte à côte sur un ordinateur, pas sur un téléphone. Par défaut, Flexbox les comprime plutôt que de les faire descendre. Une ligne corrige ça :

© 2025 Mr. Lambrechts

```css
.cartes {
  display: flex;
  gap: 20px;
  flex-wrap: wrap;
}
```

`flex-wrap: wrap` autorise le passage à la ligne quand la place manque. Les cartes se réorganisent d'elles-mêmes selon la largeur de l'écran.

🧪 Vérifications

1. Sur quel élément écrit-on `display: flex` : le conteneur ou les enfants ?
2. Quelle est la différence entre `gap` et le `padding` du conteneur ?
3. Tu veux le logo à gauche et le menu à droite d'une même barre. Quelle valeur de `justify-content` ?
4. À quoi sert `align-items: center` ?

✍️ À retenir

- `display: flex` se pose sur le **parent**, et n'agit que sur ses enfants directs.
- `gap` espace les enfants entre eux, sans toucher aux bords.
- `justify-content` répartit **dans le sens de la ligne** : `center`, `space-between`…
- `align-items` aligne **en travers**, le plus souvent avec `center`.
- `flex-wrap: wrap` autorise le passage à la ligne sur les petits écrans.

Suite

Tu as toutes les pièces. Reste à les assembler en une page complète — et à fabriquer le motif le plus répandu du web : la carte.

© 2025 Mr. Lambrechts