Cours d'intégration

3W Academy

CSS 5. La mise en page

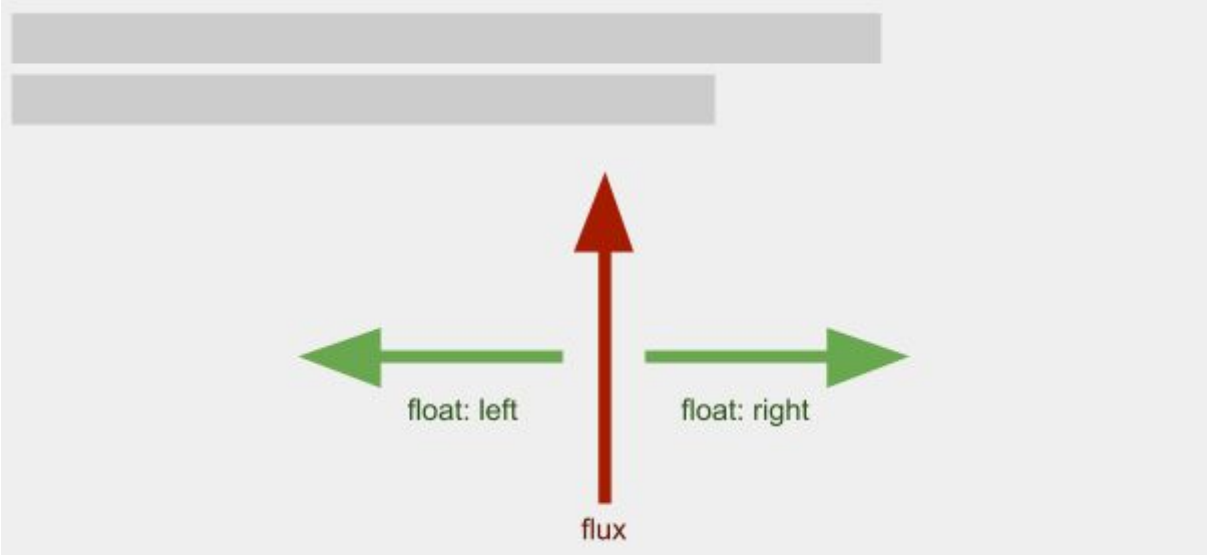
5. La mise en page

5.1. Le flux

Le flux HTML peut se définir comme comme étant le comportement naturel d'affichage des éléments d'une page web. Autrement dit, les éléments se placent les uns à la suite des autres depuis le haut de la page. Il en découle que la position de chaque élément dépend de celui d'avant.

Le flux agit comme une gravité inversée. On peut le schématiser par une flèche du bas vers le haut.

On peut faire sortir un élément du flux par la gauche ou par la droite à l'aide de la propriété float, par exemple pour faire en sorte que le texte entoure une image.



Dans le CSS

```
img.left {
    float: left;
    margin-right: 1em;
}
```

Dans le document HTML

```
<p>
    <img class="left" src="images/1.jpg" alt="Float left example">
```

```
    Lorem ipsum ...
</p>
```

Un élément flottant n'est pas pris en compte dans le calcul de la hauteur de son parent, il faut parfois utiliser un élément additionnel avec la propriété clear ou un artifice tel que "clearfix" pour le prendre en compte à posteriori.

On a longtemps utiliser la propriété *float* pour faire de la mise en page tel que juxtaposer des éléments les uns à côté des autres, il est maintenant préférable d'utiliser des méthodes modernes telles que "flex" ou "grid".

5.2. Le modèle de boîte

En CSS le modèle de boîte est la base de la mise en page. Il décrit chaque élément comme une boîte rectangulaire qui possède une série de propriétés liée à la structure en pelure d'oignons de cette boîte. Elle est ainsi décomposée en quatre contours imbriqués: la marge (margin), la bordure (border), le remplissage (padding) et le contenu (content).

5.3. Propriétés de base

Chaque couche de la boîte peut être modifiée avec des propriétés CSS spécifiques:

5.3.1. content

La taille du contenu dépend du texte qu'il contient (sa largeur dépend du parent et sa hauteur est déduit du nombre de ligne de texte) mais peut être modifiée avec les propriétés *width* (largeur) et *height* (hauteur) mais aussi avec leurs déclinaisons minimale et maximale (min-width, max-width, min height et max-height).

Dans le CSS

```
.example {
    width: 50%;
    min-width: 200px;
}
```

5.3.2. padding

Le remplissage permet de mettre de l'espace entre le contenu et la bordure pour améliorer la lisibilité (le texte qui colle à son encadrement ne sera pas très lisible). Il peut être modifié avec la propriété *padding* ou ses déclinaisons en fonction:

- du haut: *padding-top*,
- du bas: *padding-bottom*,
- du côté gauche: *padding-left*,
- du côté droit: *padding-right*.

Dans le CSS

```
.example {
    padding: 1em;
}
```

5.3.3. border

La bordure permet d'afficher une décoration autour de la boîte. Elle peut être modifiée avec la propriété *border* ou ses déclinaisons en fonction:

- du haut: *border-top*,
- du bas: *border-bottom*,

- du côté gauche: *border-left*,
- du côté droit: *border-right*.

La propriété *border* contient trois composantes:
- sa taille en px ou em
- son style: plein (solid), en tirets (dashed), en pointillés (dotted) ...
- sa couleur

Dans le CSS

```
.example {
    border: 1px solid red;
}
```

5.3.4. margin

La marge permet de mettre de l'espace entre la boîte et les autres boîtes adjacentes. Elle peut être modifiée avec la propriété *margin* ou ses déclinaisons en fonction:
- du haut: *margin-top*,
- du bas: *margin-bottom*,
- du côté gauche: *margin-left*,
- du côté droit: *margin-right*.

⚠ **Attention** les marges des boîtes successives se recouvrent, elles ne s'additionnent pas.

Dans le CSS

```
.example {
    margin-bottom: 4em;
}
```

5.4. La propriété display

Certainement la propriété la plus importante de CSS. Elle permet de modifier l'algorithme régissant la disposition de l'élément concerné par rapport aux éléments voisins.

Voici quelques valeurs possibles de la propriété *display*:

block	L'élément prend toute la largeur disponible, sa largeur correspondant à l'espace disponible de son parent. Il y un saut de ligne implicite avant l'élément et un autre après. Les propriétés liées aux modèle de boîte

	s'applique sans restriction. C'est la valeur par défaut pour pour les éléments P, H1, H2, H3 ...
inline	L'élément se comporte comme un mot dans un texte. La largeur de l'élément est celle de son contenu. L'élément se place à la suite des ses voisins de type *inline,* sur la même ligne. Un espace est automatiquement généré entre plusieurs éléments de type *inline*. Les propriétés liées aux modèle de boîte sont partiellement gérés. C'est la valeur par défaut pour les éléments A, STRONG, EM ...
inline-block	L'élément se comporte comme un élément *inline* vis à vis de ses voisins mais se comporte comme un bloc en interne: les propriétés liées aux modèle de boîte s'applique sans restriction en interne.
none	L'élément n'est plus affiché.
table	L'élément se comporte comme une table. C'est la valeur par défaut pour l'élément TABLE
flex	L'élément se comporte comme un bloc mais applique à ses éléments enfant une disposition suivant le modèle "flexbox".
grid	L'élément se comporte comme un bloc mais applique à ses éléments enfant une disposition suivant le modèle "grid".

Tout élément a un une valeur display prédéfini, Il n'y a pas de valeur par défaut globale.

5.5. Le box-sizing

Cette propriété permet de modifier les couches du modèle de boîte pris en compte dans le calcul de la taille (*width*). Par défaut la valeur est *content-box*, la largeur de la boîte est celle de son "content" et ne comprend pas le la largeur du *padding* ou la largeur du *border*, il est possible déclarer la valeur border-box qui à l'inverse, inclut ces largeurs.

Dans le CSS

```
* {
    box-sizing: border-box;
}
```

5.6. Le modèle flexbox

5.6.1. Présentation

Le modèle flexbox permet de faire une mise en page très complexe d'une série d'éléments sans détourner d'autres propriétés CSS non conçue initialement pour cet usage (comme l'utilisation de display table, de display inline-block ou de float) et sans les inconvénients respectives de ces autres méthodes.

On nomme le parent (qui contient la série d'éléments à disposer) le "container" (conteneur en français) et les enfants de ce conteneur les "items". Les items sont représentés en rouge dans les schémas suivants. Certaines propriétés du modèle flexbox s'appliquent au conteneur, d'autres propriétés s'appliquent aux items.

Une simple instruction suffit pour transformer une liste verticale en liste horizontale. Cette propriété (*display: flex*) est donnée au conteneur.
Dans le CSS

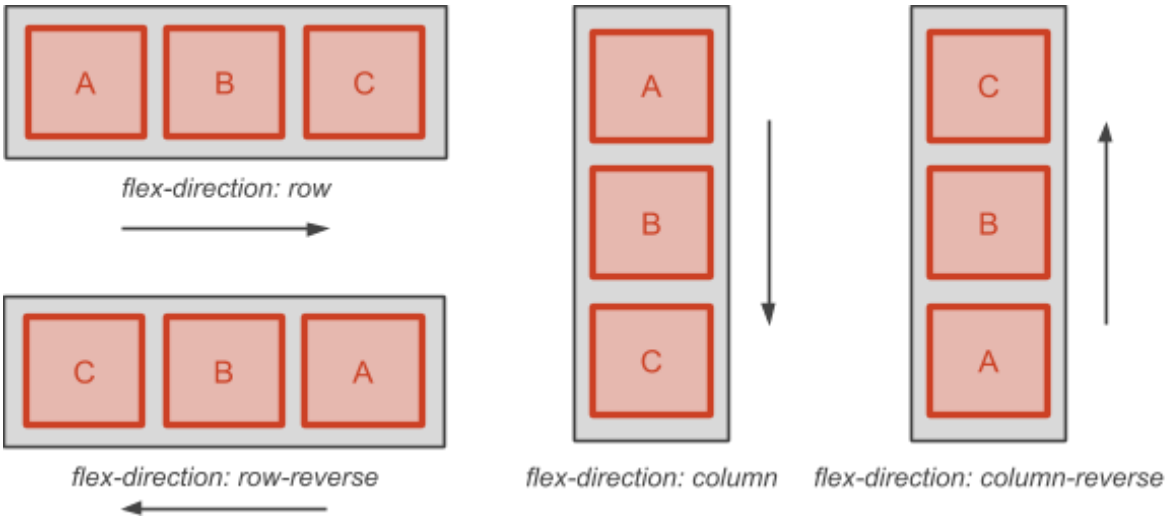
```
.example {
    display: flex;
}
```

Dans le document HTML

```
<ul class="example">
    <li>Lorem ipsum ...</li>
    <li>...</li>
</ul>
```

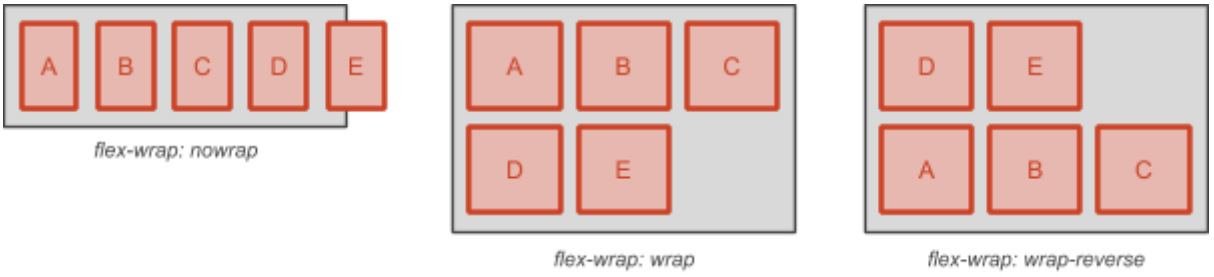
5.6.2. flex-direction (conteneur)

Il est possible de changer l'orientation principale avec la propriété flex-direction:



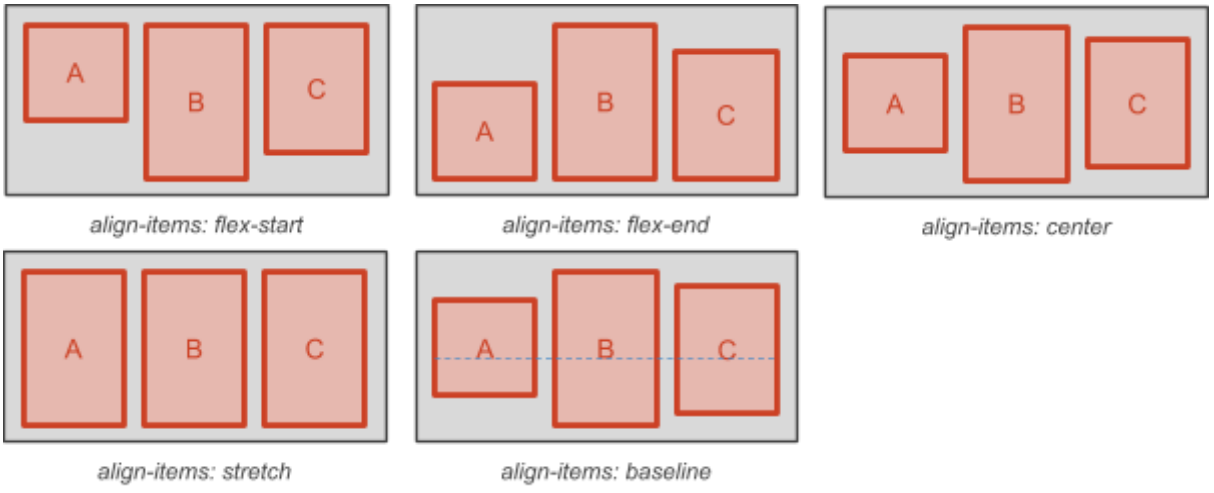
5.6.3. flex-wrap (conteneur)

Il est possible de disposer les items sur plusieurs lignes avec la propriété flex-wrap :



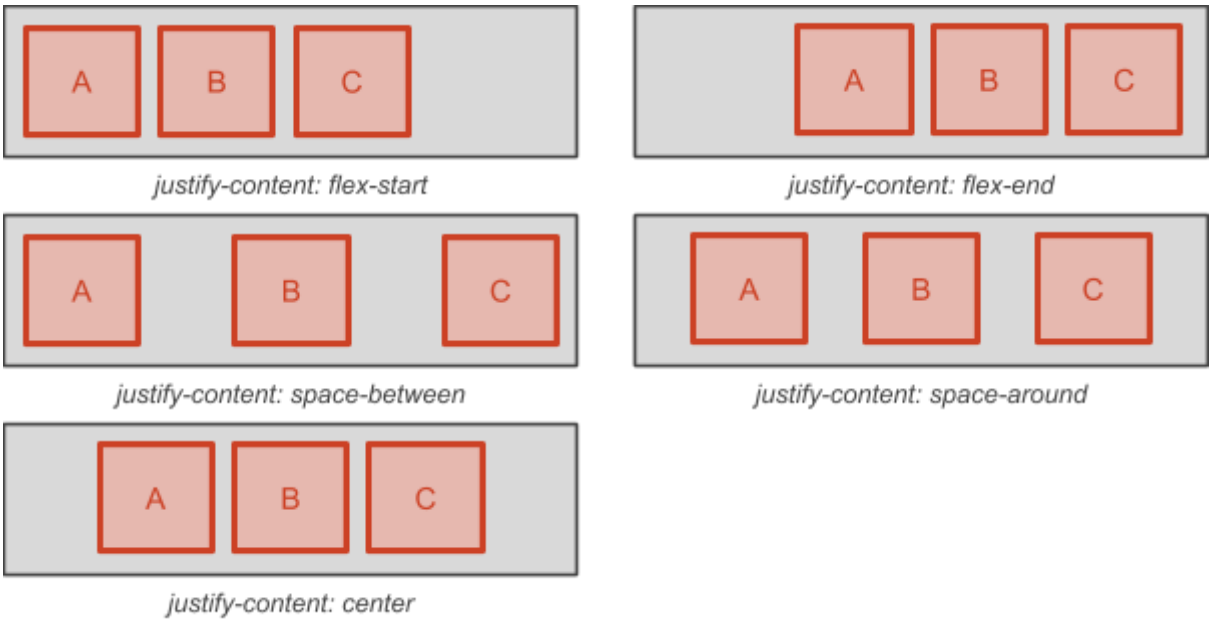
5.6.4. align-items (conteneur)

Il est possible d'ajuster l'alignement vertical des items avec la propriété align-items:



5.6.4. justify-content (conteneur)

Il est possible de répartir les items horizontalement avec la propriété justify-content:



Il est aussi possible de répartir les items verticalement avec la propriété align-content et les mêmes valeurs.

5.6.5. propriétés appliquées aux items

Un certain nombre de propriétés du modèle flexbox s'applique directement aux items:

align-self permet d'aligner un item avec les mêmes valeur que la propriété du conteneur *align-items*.

flex-grow permet de spécifier le poids (grossissant) d'un item par rapport aux autres dans la répartition des tailles des items.

flex-shrink permet de spécifier le poids (amincissant) d'un item par rapport aux autres dans la répartition des tailles des items.

order permet de modifier l'ordre de chacun des items.