2 CDUI CSS

Patrick VINCENT
pvincent@erasme.org

Sylvain PHILIP
contact@sphilip.com

Les feuilles de style

-CSS 3-

Présentation des feuilles de style

Définition

- **C**ascading **S**tyle **S**heets = Feuilles de Style en Cascade
  - Langage utilisé pour décrire la présentation d'un document structuré en HTML
  - Le standrard est dicté par le World Wide Web Consortium (www.w3c.org), les navigateurs essayent de le respecter
  - Versions : CSS 2 (>1998) et CSS3 (en cours)
- Séparation du fond de la forme
  - Les pages HTML décrivent la structure et le contenu
  - Les CSS décrivent la présentation pour l'affichage sur un média donné (screen, imprimante...)

Par la pratique

- **CSSZenGarden**
  - 1 page HTML <-> n feuilles de styles
    - http://www.csszengarden.com/tr/francais/
    - http://www.mezzoblue.com/zengarden/alldesigns/official/
- **Désactiver les styles via FireFox**
  - Affichage > Style de la page > Aucun style
    - http://fr.wikipedia.org/wiki/Accueil
    - http://fr.yahoo.com/
- **Chaque balise HTML a un rendu par défaut dans le navigateur**
    - le style CSS relatif à la balise vient **surcharger** ce rendu

Pourquoi les CSS ?

- Les CSS assure la cohérence graphique d'un site.
- La modification de la charte graphique d'un site web est facilitée par l'utilisation des CSS : seule la feuille de style est modifiée.
- Le code HTML de la page est allégé, il y a donc une diminution du poids et par conséquent du temps de chargement.

Implémentation des styles

- **Dans les balises**
  - à éviter car on perds l'aspect global du style

```
<body>
   <p style="/* Mes styles */">texte</p>
</body>
```

- **Dans la page**
  - pour des maquettes et des feuilles simples à tester

```
<head>
   <style type="text/css">
      /* Mes styles */
   </style>
</head>
```

Implémentation des styles (2)

- **Dans une feuille de style extérieure**
  - solution à retenir pour externaliser et mutualiser la feuille

```
<head>
   <link rel="stylesheet" href="style.css">
</head>
```

OU

```
<head>
   <style type="text/css">
      @import url(style.css);
   </style>
</head>
```

Implémentation des styles (3)

- possibilité de spécifier le **média** dédié au style
  - **all, screen**, **print**, projection, braille, embossed, aural, speech...
  - seuls pour l'instant les medias d'affichage fonctionnent correctement

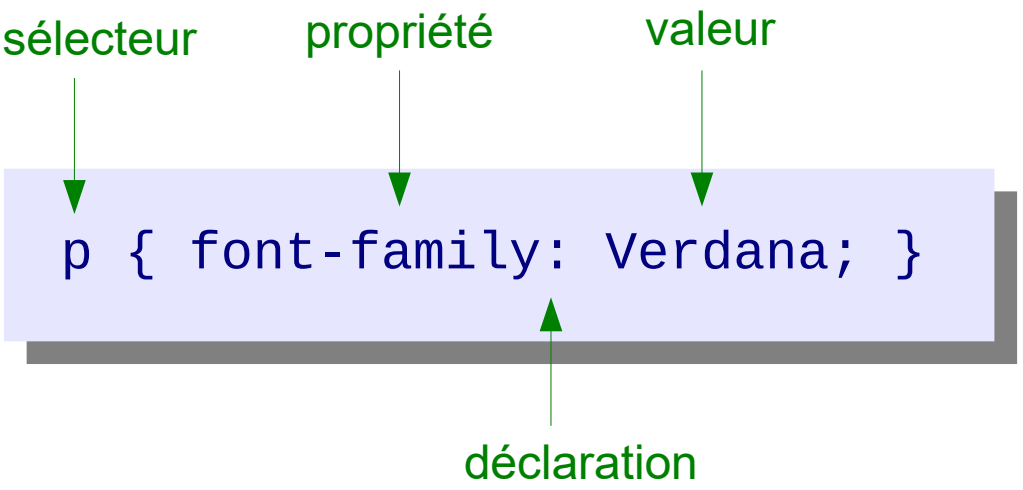
```
<head>
    <style media="screen">
        /* Mon style1 */
    </style>
    <style media="screen">
        /* Mon style2 */
    </style>
    <style media="print">
        /* Style impression */
    </style>
</head>
```

page13.html

Syntaxe

Déclaration des styles

```
balise
{
  propriete: valeur;
  propriete: valeur;
  propriete: valeur;
}
```



- balise : nom de la balise dont on modifie l'apparence
  - , <b>, <h1>
- propriété : catégories d'effet de style
  - color, font-size, font-family
- valeur : valeur associée à la propriété
  - red, 10px, Verdana

Syntaxe

- On peut ajouter des **commentaires /*...*/**
- Une déclaration ouverte doit toujours être fermée **{...}**
- Une définition de valeur doit se terminer par **;**
  On peut enchaîner plusieurs définitions de valeurs au sein d'une même déclaration.
- Les espaces et les tabulations ne sont pas interprétés.

```
balise
{
   /* Première déclaration */
   propriete: valeur;
}
```

Premiers exemples

```
body
{
    font-family: sans-serif;
    color: #646c55;
    font-size: 16px;
    margin: 0;
    padding: 0;
}
```

```
div, header, section, article, footer, n
{
    padding: 10px;
    margin: 10px;
    border: black solid 1px;
}
```

```
h1
{
    font-size: 28px;
    letter-spacing: 1px;
    margin: 0;
    text-align: center;
}
```

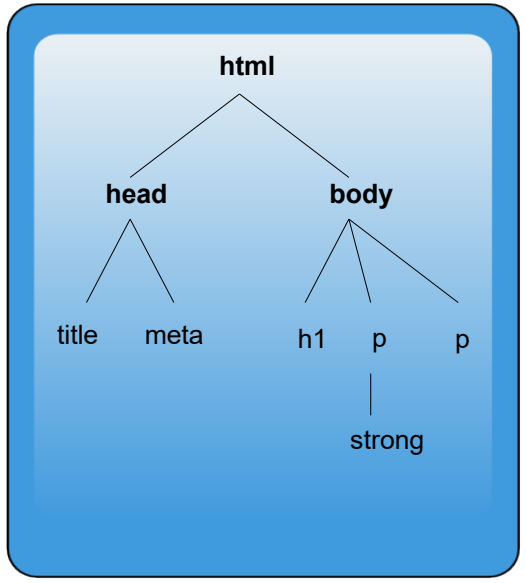
```
a {
    color: #565e49;
}

a:hover {
    color: #646c55 ;
}
```

Feuilles en cascade

- Les styles sont appliquées successivement dans l'ordre suivant (par ordre croissant d'importance) :
  - 1. Styles extérieurs
  - 2. Styles de la page
  - 3. Style dans la balise elle-même
  - En cas de surcharge, le dernier l'emporte

- Les propriétés de chaque balise sont héritées des éléments parents dans l'arbre du document

transmission des propriétés



Couleurs

- Une couleur peut être définie par
  - son nom (si il existe) : green
  - son code hexadécimal long : #008080
  - son code hexadécimal court : #FFF
  - son code RGB : rgb(0,0,255) ou rgb(10%,10%,30%)
  - transparent
- Les 16 couleurs de base du Web (compatibles W3C)

Unités de valeur

- La taille et la position des éléments doivent comporter une unité :
  - font-size : 12px;
  - width: 90%;
  - margin: 0;
- Rapports
  - **%** (pourcentage) : la taille / position de l'élément est calculée de façon relative par rapport à son parent
  - **em** : 1em = 100%, 1.2em = 120%, etc.
  - **rem** : 1rem = 100%, 1.2rem = 120%, etc.
- Mesures
  - **px** : pixels, plus petite unité de l'écran
  - **pt** : points, plus petite unité de l'imprimerie

TP

projet4.html

- A partir de votre dernière version de *projet4.html*
  - Créer une feuille de styles **style.css** et placer-la dans le dossier css
  - Ajouter la feuille de styles à la page dans le <head> : <link rel="stylesheet" href="css/style.css">
  - ajouter les styles vus en exemple pour les éléments
    - body
    - a et a:hover
    - h1
    - div
  - tester le résultat dans le navigateur à chaque étape

Les sélecteurs

bal remplace n'importe quelle balise

*** { }** **sélecteur universel**

- désigne tous les éléments

bal { } **sélecteur de balise**

- désigne toutes les balises <*bal*> de la page

.nom_classe { } **sélecteur de classe**

- désigne les éléments ayant l'attribut *class="nom_classe"*

bal.nom_classe { } **sélecteur de classe**

- désigne toutes les balises *bal* ayant l'attribut *class="nom_classe"*

Les sélecteurs (2)

bal remplace n'importe quelle balise

#nom_id { } **sélecteur d'identification**

- désigne l'élément ayant reçu id=*"nom_id"*

bal#nom_id { } **sélecteur d'identification**

- désigne la balise <*bal*> ayant reçu id=*"nom_id"*

bal1 bal2 bal3 { } **hiérarchie**

- désigne les éléments <*bal3*> contenus dans une balise <*bal2*> lui-même contenu dans une balise <*bal1*>

bal1, bal2, bal3 { } **sélecteur collectif**

- s'appliques aux balises <bal1>, <bal2> et <bal3>

Les sélecteurs (3)

bal remplace n'importe quelle balise

bal1 + bal2 { } **sélecteur adjacent**

- désigne les <*bal2*> directement placés derrière un <*bal1*>

bal1 > bal2 { } **sélecteur d'enfants**

- désigne les <*bal2*> directement placés dans un <*bal1*>
-

bal1[attr="value"] ***sélecteur d'attributs***

- *désigne les <bal1> ayant l'attribut attr="value"*

Pseudo classes et éléments

Note : les balises sont citées à titre d'exemple

p:first-letter|first-line|after { } **pseudo-élément**

- désigne un sous ensemble de l'élément <*p*>
  - first-letter : première lettre
  - first-line : première ligne
  - after : ajout après l'élément

img:active|hover|focus|visited { } **pseudo-classes dyn.**

- désigne l'élément <*img*> dans un état en temps réel
  - hover : l'élément est survolé
  - active : souris pressée et non relâchée
  - focus : l'élément est activé (un formulaire par exemple)
  - visited : lien visité (<a> uniquement)

Comment écrire les sélecteurs qui s'appliquent au texte en vert ?

```
<strong>c'est important</strong>
```

```
<div id="principal">
    Mon texte
</div>
```

```
<a href="#" class="liens">
    cliquer ici
</a>
```

```
<div class="home">
    <strong>c'est important</strong>
</div>
```

Comment écrire les sélecteurs qui s'appliquent au texte en vert ?

```
<strong>c'est important</strong>
```

strong { ... }

```
<div id="principal">
    Mon texte
</div>
```

#principal { ... }

```
<a href="#" class="liens">
    cliquer ici
</a>
```

a, a:hover { ... }

ou

a.liens { ... }

a.liens:hover { ... }

```
<div class="home">
    <strong>c'est important</strong>
</div>
```

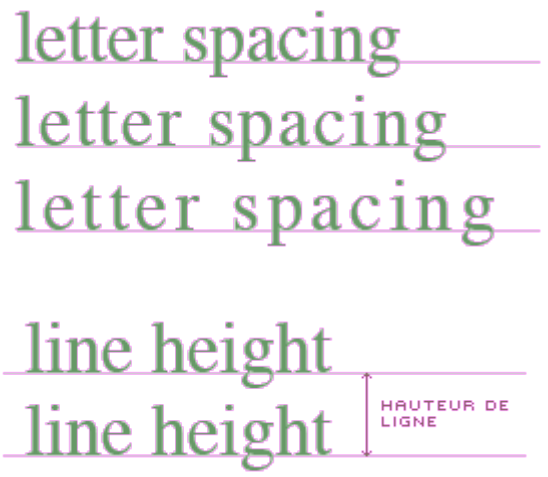
div.home strong { ... }

Mise en forme

Typographie

- Polices
  - Elles ne sont pas toutes présentes sur tous les ordinateurs
  - Elles se déclarent dans un ordre de préférence
  - On termine par les polices standard : sans-serif, serif, monospace, cursive.

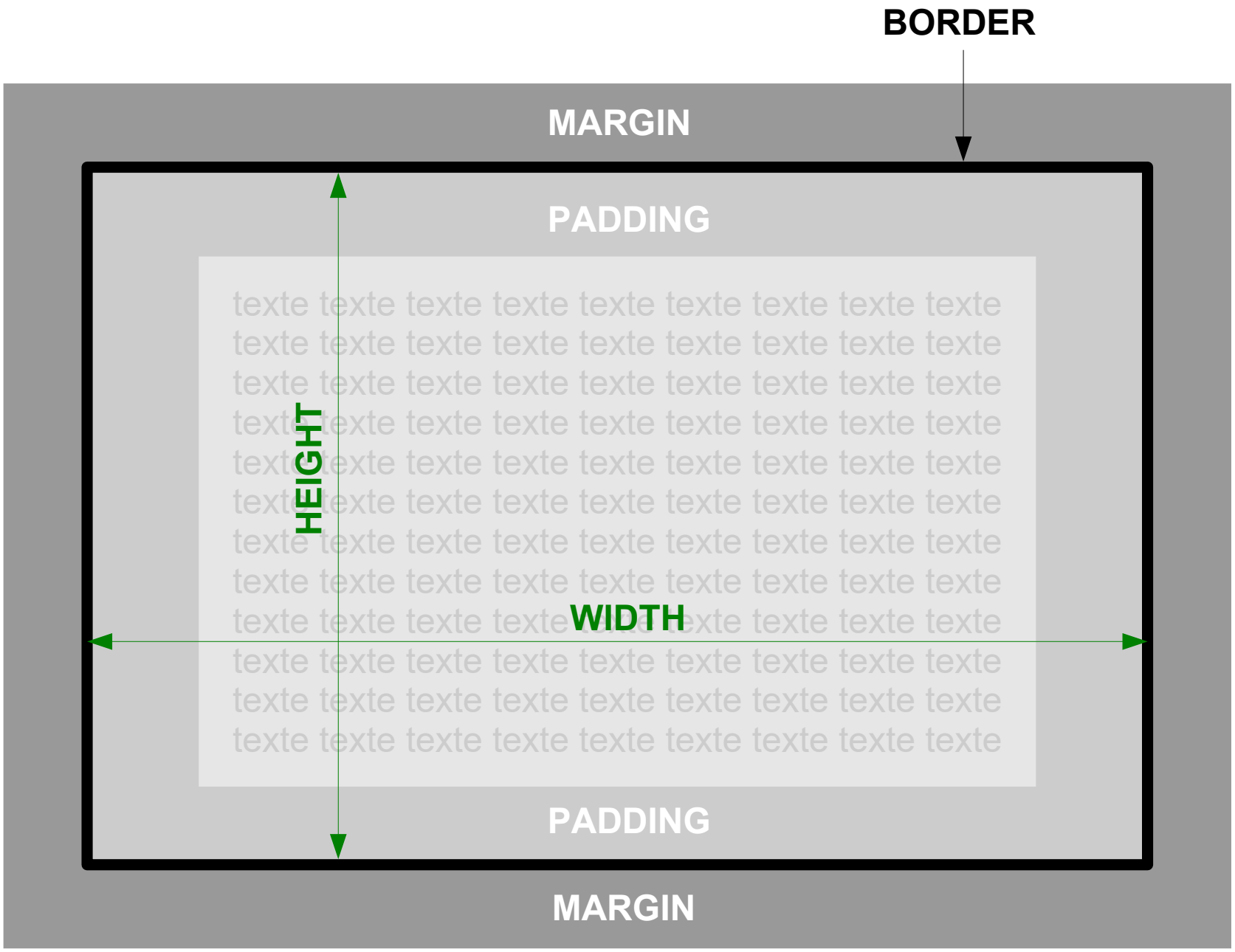
- Les espacements
  - Peuvent être fixés par les attributs line-height et letter-spacing



Typographie

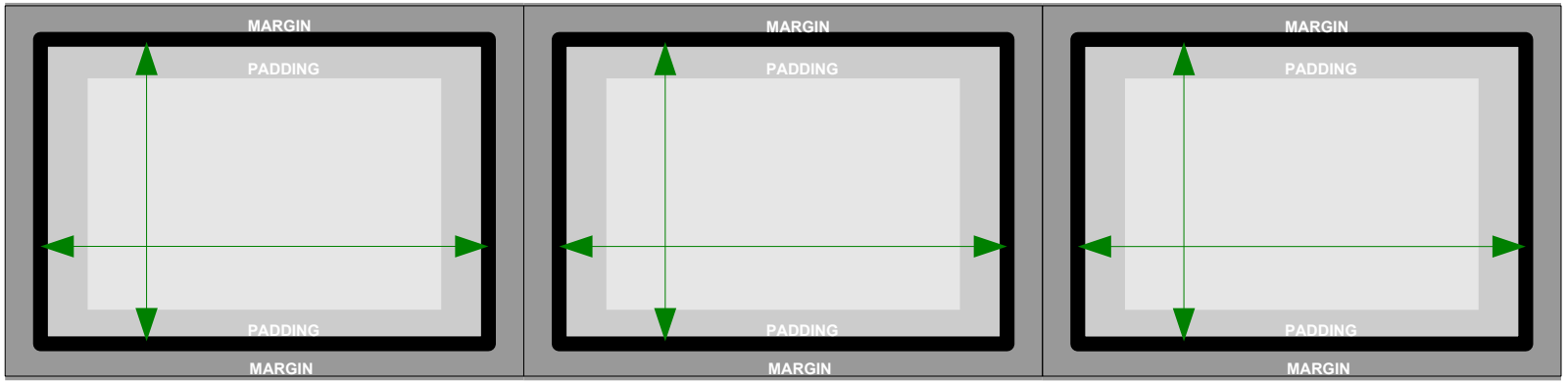
Action	Propriété	Valeur	Description
Choix de la police	**font-family :**	**"Trebuchet MS", arial, sans-serif;**	applique les polices par ordre décroissant ; mettre les noms avec espace entre ""
Taille de la police	**font-size :**	**10px; 1.2em; 90%;**	préférer em et % pour hériter des valeurs des éléments parents
Couleur	**color :**	**silver; #0033FF; transparent;**	
Orientation de la police	**font-style :**	**normal; italic; oblique;**	
Epaisseur	**font-weight :**	**bold; bolder; normal;**	
Soulignement	**text-decoration :**	**none; underline; overline; line-through;**	
Mise en majuscule	**text-transform :**	**none; capitalize; uppercase; lowercase;**	capitalize: première lettre uniquement.
Alignement du texte	**text-align :**	**left; right; center; justify;**	
Hauteur de ligne	**line-height :**	**normal; 1.2em; 90%; 10px;**	
Espacement entre les lettres	**letter-spacing :**	**normal; -2px;**	
Espacement entre les mots	**word-spacing :**	**normal; 3px;**	
Raccourci	**font :**	**font-style, font-variant, font-weight, fontsize/line-height, font-family**	p { font: bold 12px/24px verdana }

Présentation des blocs



Compléments

- Plusieurs blocs : les margins s'ajoutent



- Quelles est la taille d'un bloc total (boîte) ?
  - Selon le standard : **boîte = margin+border+width**
  - Pour IE : **boîte = width**
  - -> Privilégier les padding aux margin...

Présentation des blocs

- Tout élément en bloc possède 4 attributs :
  - taille propre : *width* et *height*
  - bordure : *border*
  - marge intérieure : *padding*
  - marge extérieure : *margin*

Action	Propriété	Valeur	Notes
Largeur	**width :**	**30%; 250px; auto;**	auto calcule automatiquement la taille actuelle (image par exemple)
Hauteur	**height :**	**30%; 250px; auto;**	

Marges et espacements

Description	Propriété	Exemple
Marge supérieure	**margin-top :**	**5px;**
Marge de droite	**margin-right :**	**0.5em;**
Marge inférieure	**margin-bottom :**	**2pt;**
Marge de gauche	**margin-left :**	**0;**
Raccourci pour les marges	**margin :**	**-5px 0.5em 2pt 0; auto;** (alignement centré du bloc)
Espace intérieur entre l'élément et la bordure supérieure	**padding-top :**	**3px;**
Espace intérieur entre l'élément et la bordure droite	**padding-right :**	**0.25em;**
Espace intérieur entre l'élément et la bordure inférieure	**padding-bottom :**	**0;**
Espace intérieur entre l'élément et la bordure gauche	**padding-left :**	**2pt;**
Raccourci vers l'ensemble des propriétés d'espace intérieur	**padding :**	**3px 0.25em 0 2pt;**

Arrières plan

Image d'arrière-plan	**background-image :**	**url(http://url);**	
Répétition de l'arrière-plan	**background-repeat :**	**repeat; repeat-x; repeat-y; no-repeat;**	
Spécifie si l'image reste fixe avec les déplacements de l'écran	**background-attachment :**	**scroll; fixed;**	
Position de l'image par rapport au coin supérieur gauche	**background-position :**	**top; middle; bottom; left; center; right;**	possibilité d'indiquer des valeurs en pixels
Taille de l'arrière-plan	**background-size :**	**cover; contain; px; %;**	
Raccourci global vers les propriétés des AP	**background :**	**#000000 url(test.jpg) no-repeat scroll center top;**	

```
body
{
    background: transparent url(img/presentation-bg.jpg)
    no-repeat scroll center top;
}
```

Bordures

Description	Propriété	Valeur
Epaisseur de la bordure	**border[-top -left -bottom -right]-width :**	**10px; 2em;**
Epaisseur de la bordure	**border-width :**	**10px 15px 15px 10px;** (HDBG)
Couleur de la bordure	**border[-top -left -bottom -right]-color :**	**#RRGGBB;**
Style de la bordure	**border[-top -left -bottom -right]-style :**	***solid;*** *(pleine)* ***dashed;*** *(en tirets)* ***dotted;*** *(en pointillés)* ***double;*** *(double)* ***ridge;*** ***inset;*** ***outset;*** *(en 3D)*
Effet arrondi	**border-radius :**	**10px; 2em;** **10px 10px 10px 10px;** (HDBG)
Raccourci global les propriétés de bordure	**border :**	border: 1px 0 0 2px dotted green;

```
body
{
    border: 1px 0 0 2px dotted green;
}
```

TP

style.css

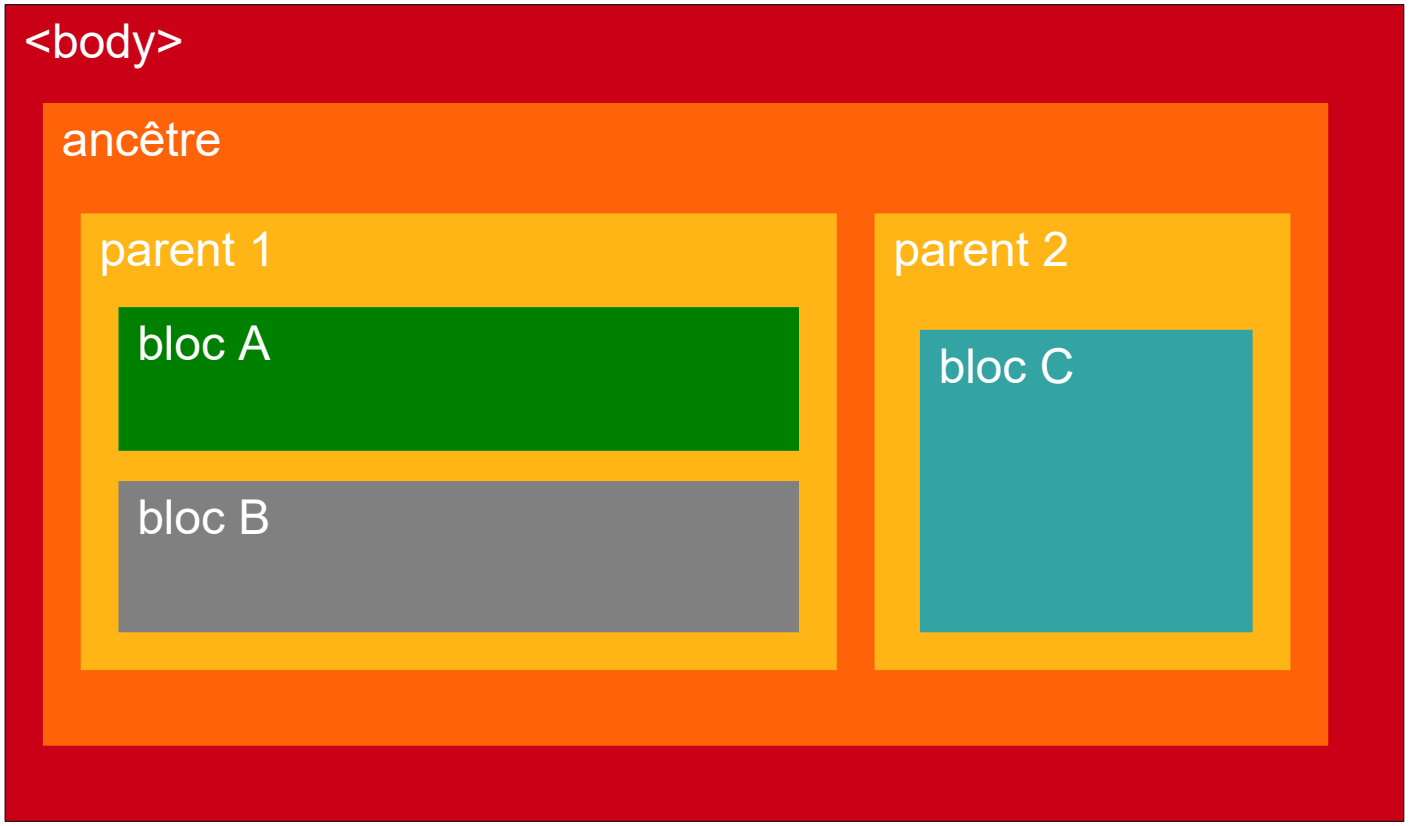
- Modifier style.css
  - Ajouter Arial et Helvetica comme polices par défaut du <body>
  - Ajouter les fonds de couleurs pour les blocs **contact** et **footer**
  - Ajouter l'image de fond pour le bloc **banniere-presentation**

Positionnement

Propriétés de position

Description	Propriété	Valeur
Propriétés	**display :**	***none;*** *(n'affiche pas le bloc)* ***block;*** *(force la propriété bloc)* ***inline;*** *(force la propriété en-ligne)* ***list-item;*** *(force la propriété en-ligne)* ***table;*** *(force la propriété tableau)*
Affichage	**visibility :**	**visible; hidden;** (cache l'élément mais réserve un espace pour son affichage)
Position	**position :**	**static;** (default - dans le flux) **relative;** (décalage - dans le flux) **absolute;** (fixe - hors du flux) **fixed;** (fixe - hors du flux – noscroll)
Distance au parent	**top : / right : / bottom : / left :**	**-10px; 30%; 2em;** (pas pour static)
Profondeur d'affichage (calque)	**z-index :**	**auto; 1000;**
Positionnement flottant	**float :**	**left; right; none;** (default)
Efface le flottement	**clear :**	**left; right; both; none;** (default)

Hiérarchie des éléments



- A et B sont frères
- A, B et C ont le même ancêtre
- Tous les éléments sont contenus dans <body>

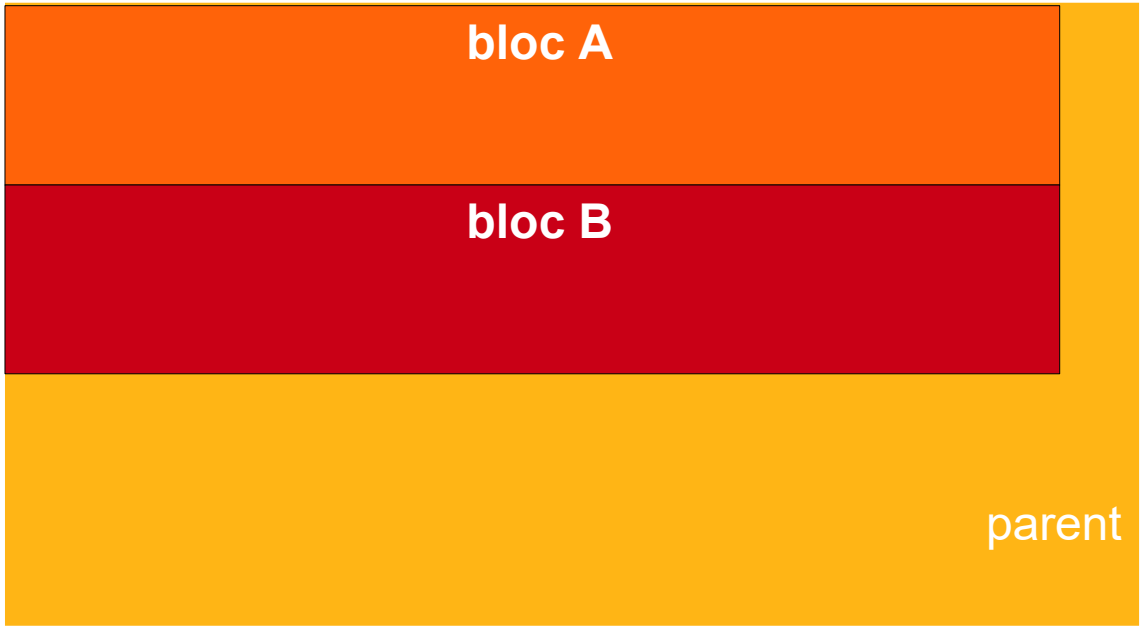
- De quelle nature sont bloc A et bloc B ? (bloc ou en ligne)
- Parent 1 et Parent 2 ?

Le flux de document

position par defaut

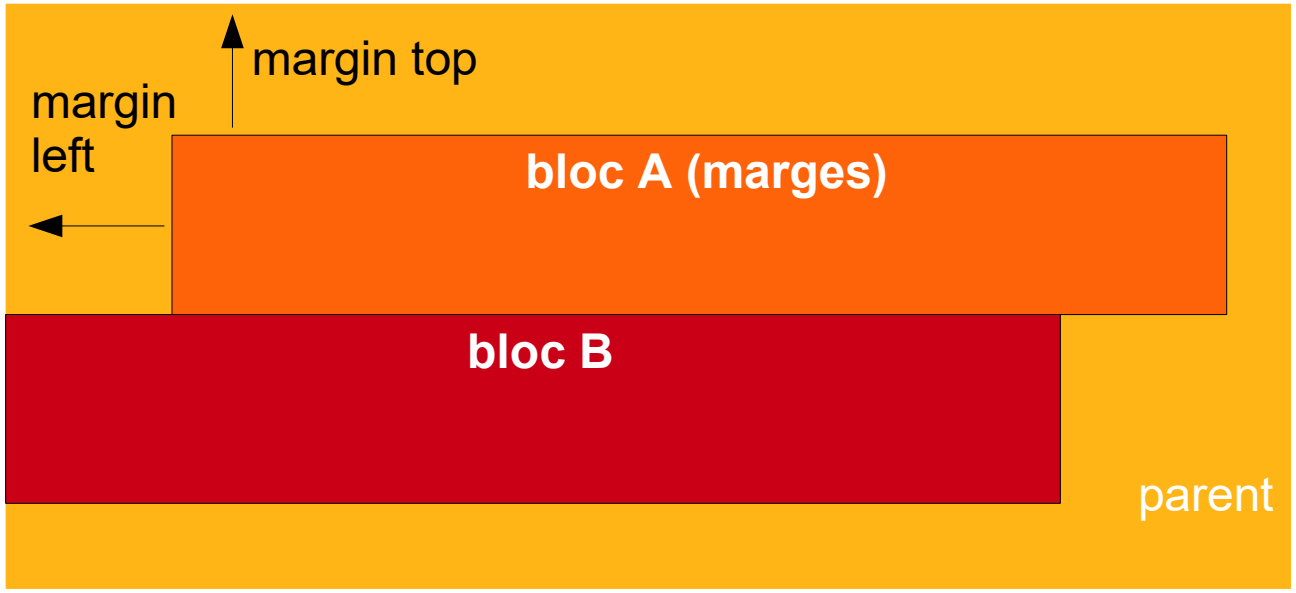
position : relative;

- Par défaut les éléments se placent en haut à gauche de leur parent et suivent le flux de document :
  - 1. **l'ordre** dans lequel ils sont inscrits dans le code HTML
  - 2. **leur nature** (bloc ou en-ligne)
    - bloc : en dessous du précédent
    - en-ligne : à côté du précédent



Le flux de document (2)

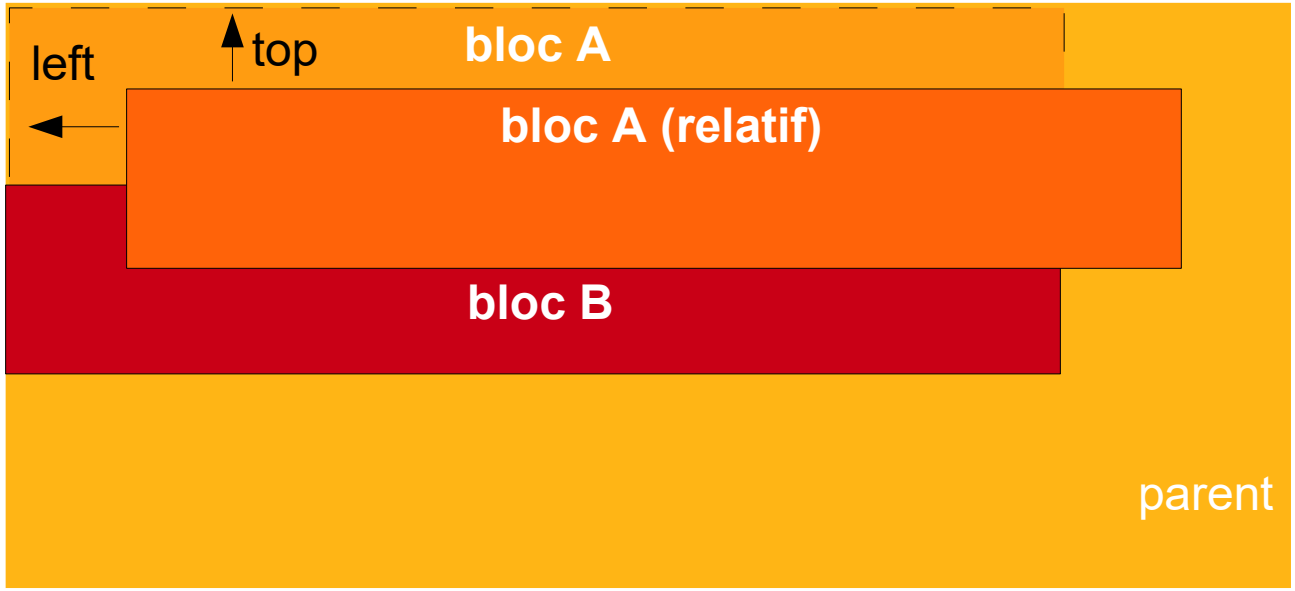
- 3. **leurs marges** internes (padding) et externes (margin)



note : il s'agit bien des marges externes (margin) du bloc et non des marges internes (padding) du parent

Le flux de document (3)

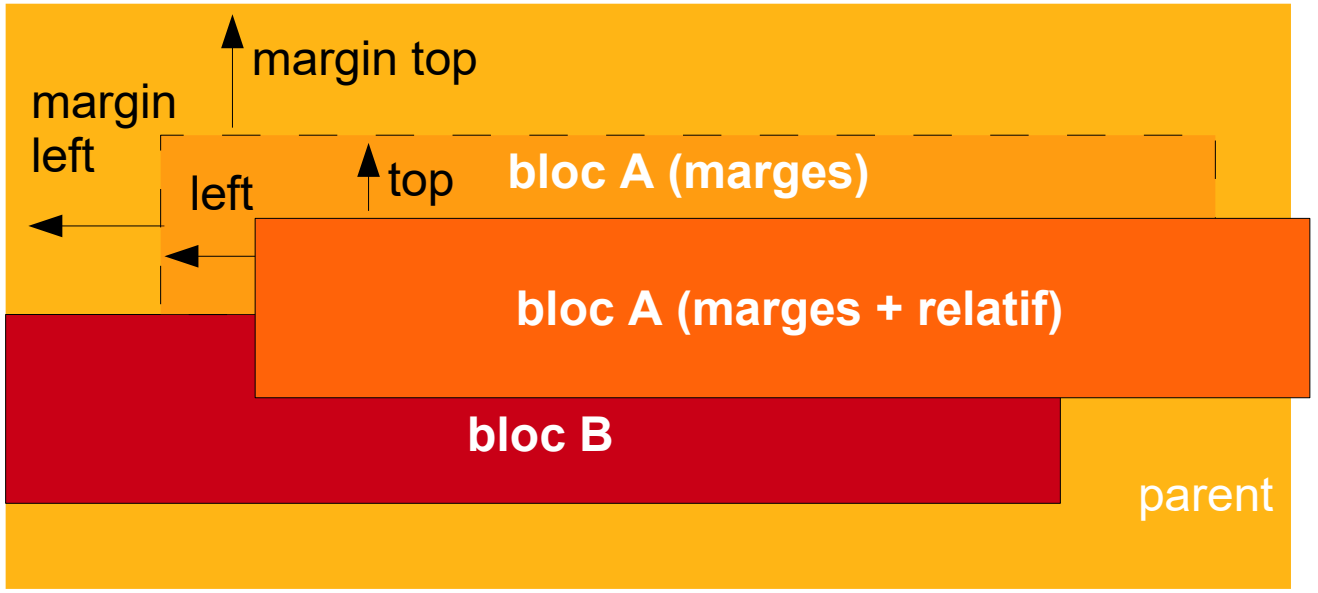
- 4. **les décalages** (top, right, left, bottom) lorsque les blocs sont en position relative



  - les éléments frères ne sont pas influencés par le décalage : ils se positionnent comme si le bloc A était dans le flux
  - cet exemple n'a pas de marges en plus des décalages
  - le z-index est utile dans de tels cas

Positionnement en flux (exemple)

- **marges + décalage**



- la taille du parent est influencée par celle des enfants si :
  - ils utilisents des marges
  - le parent possède une taille non fixe (%, em, pas de taille)
- les décalages n'ont pas d'influence sur la taille du parent

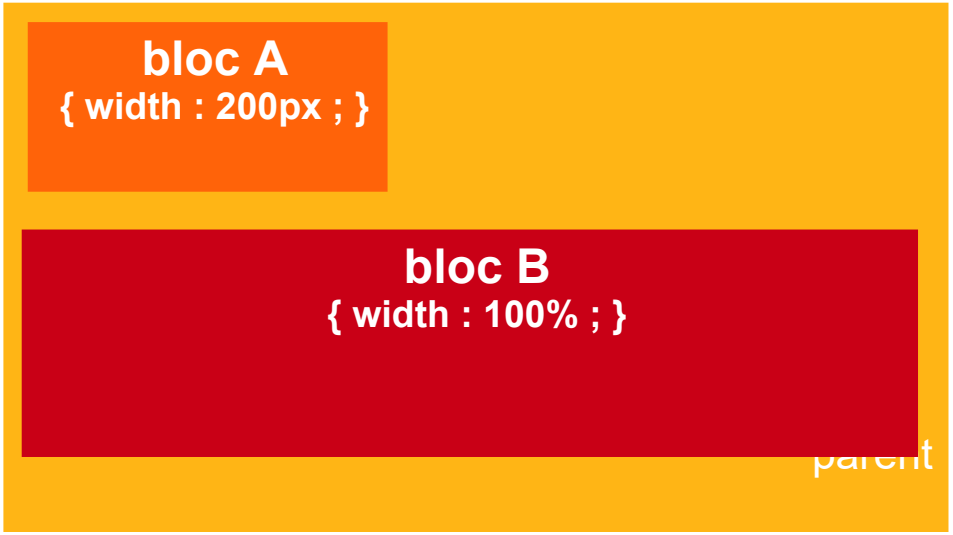
Positionnement flottant

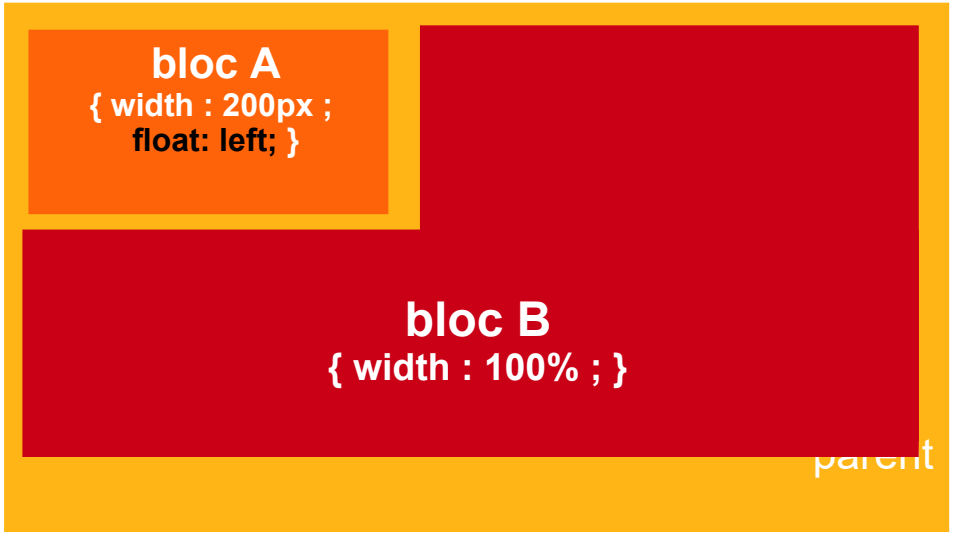
position par defaut

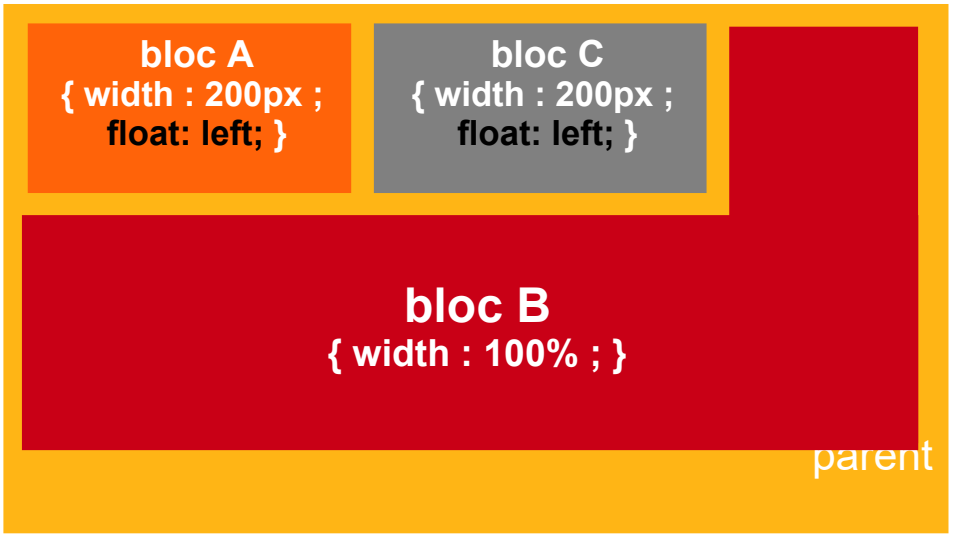
position : relative;

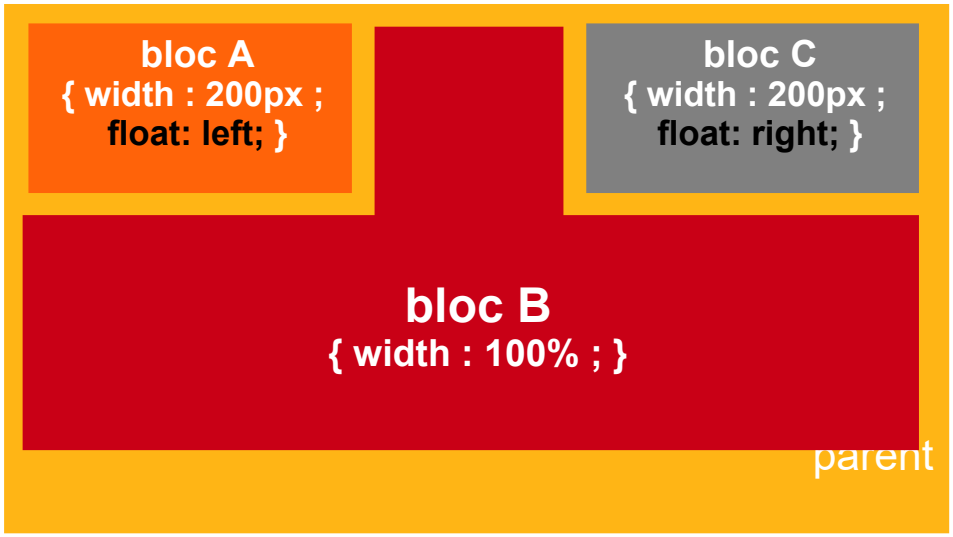
- Propriété **float :** `float: right;` `float: left;`
- Un élément flottant sort du flux normal pour prendre place à gauche ou à droite du bloc qui le contient.
- L'élément qui le suit s'écoulera dans l'espace laissé libre en épousant sa forme.
- Notes
  - L'avantage est de pouvoir placer des blocs côte à côte
  - PIEGE: les flottants sortent du flux normal ce qui impliquent qu'ils n'influencent plus la hauteur du conteneur. Si l'on prend un paragraphe flottant plus gros que les autres, il débordera du conteneur.

Positionnement flottant







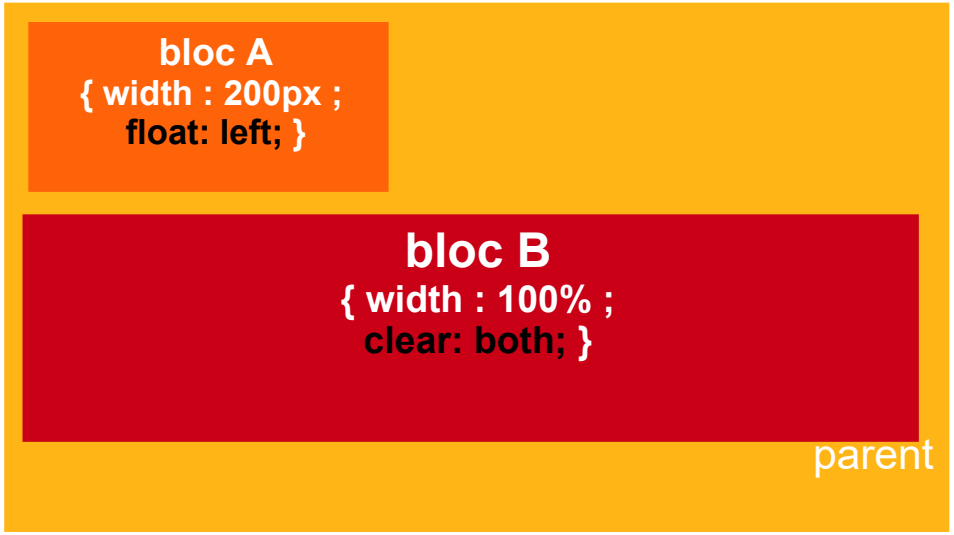


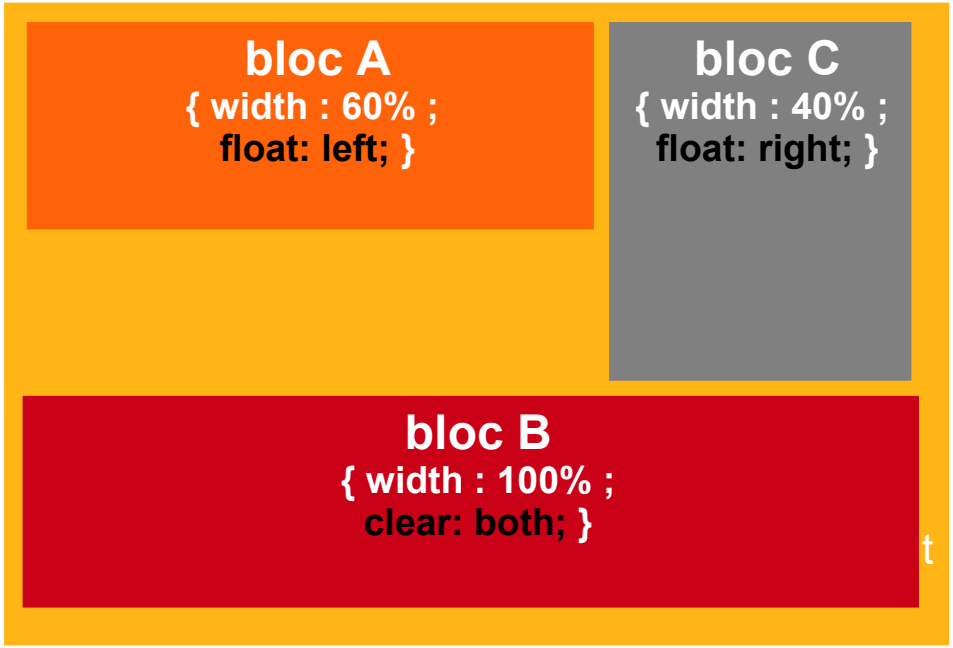
Positionnement flottant

position par defaut

position : relative;

- Propriété **clear :** `clear: both;`
- Empêche un élément du flux de se trouver sur la même ligne qu'une élément flottant.
- Typiquement utilisé pour les pieds de page.

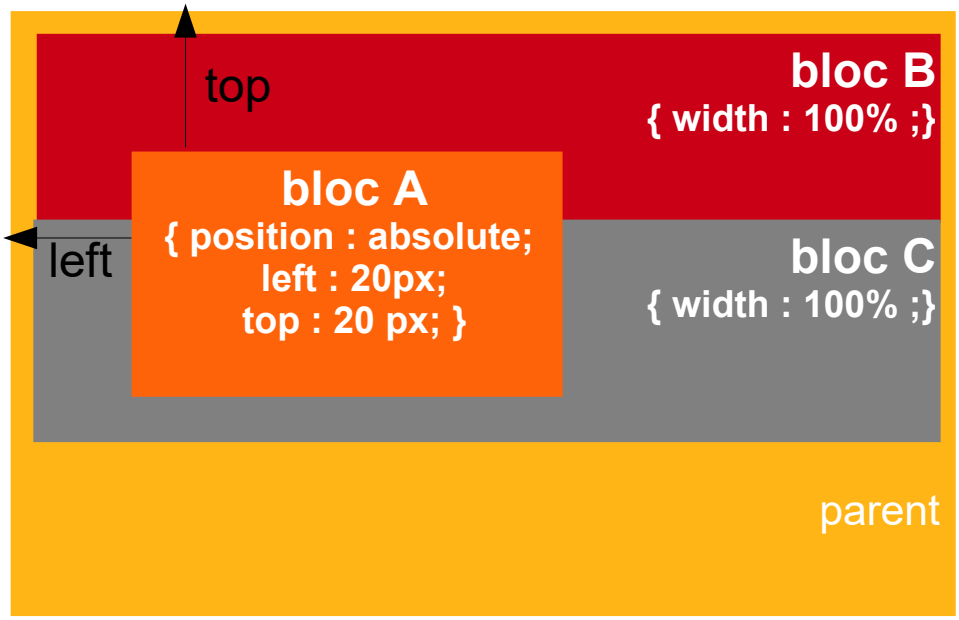




position : absolute;

Positionnement absolu

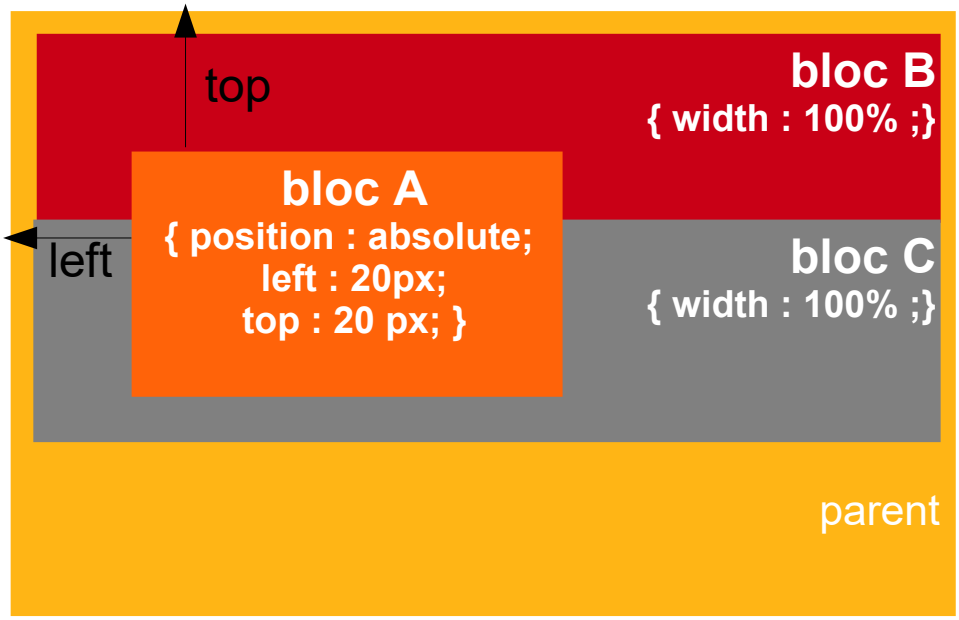
- `position: absolute; top: 2em; left: 0;`
  - l'élément se place par rapport au dernier parent ayant une position absolue. Par défaut : <*body*>
  - l'élément sort du flux et n'affecte pas les autres blocs
  - on utilise les attributs top, left, right, bottom pour le placer
- Pour des éléments indépendants de l'environnent



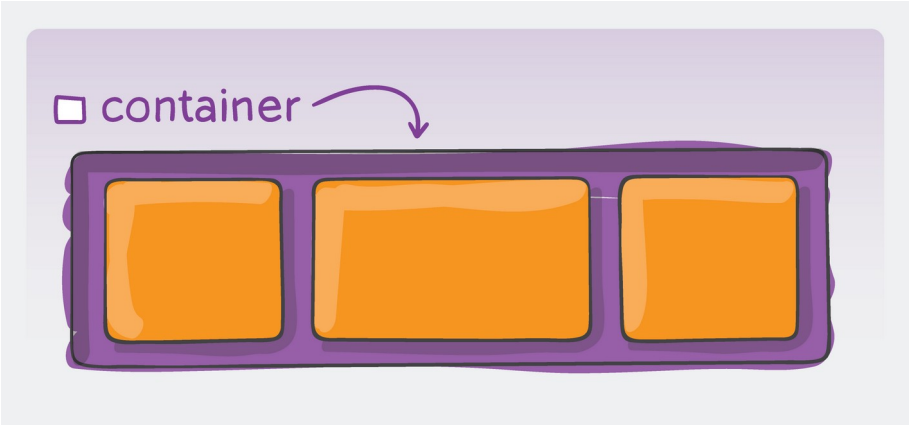
Positionnement fixe

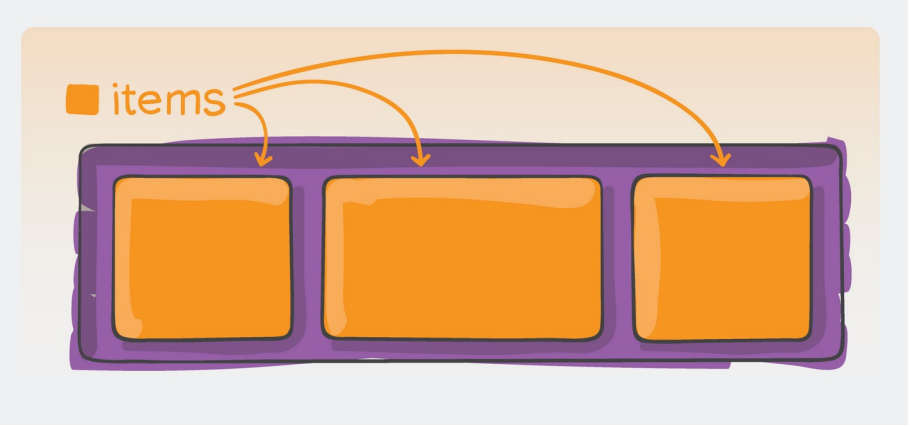
position : fixed;

-
```
position: fixed; top: 2em; left: 0;
```
  - Identique à la position absolue
  - Différence : l'élément ne bouge pas lorsque la page défile avec l'ascenseur
- Pour des effets de style. Ne pas en abuser.



Les boîtes Flex

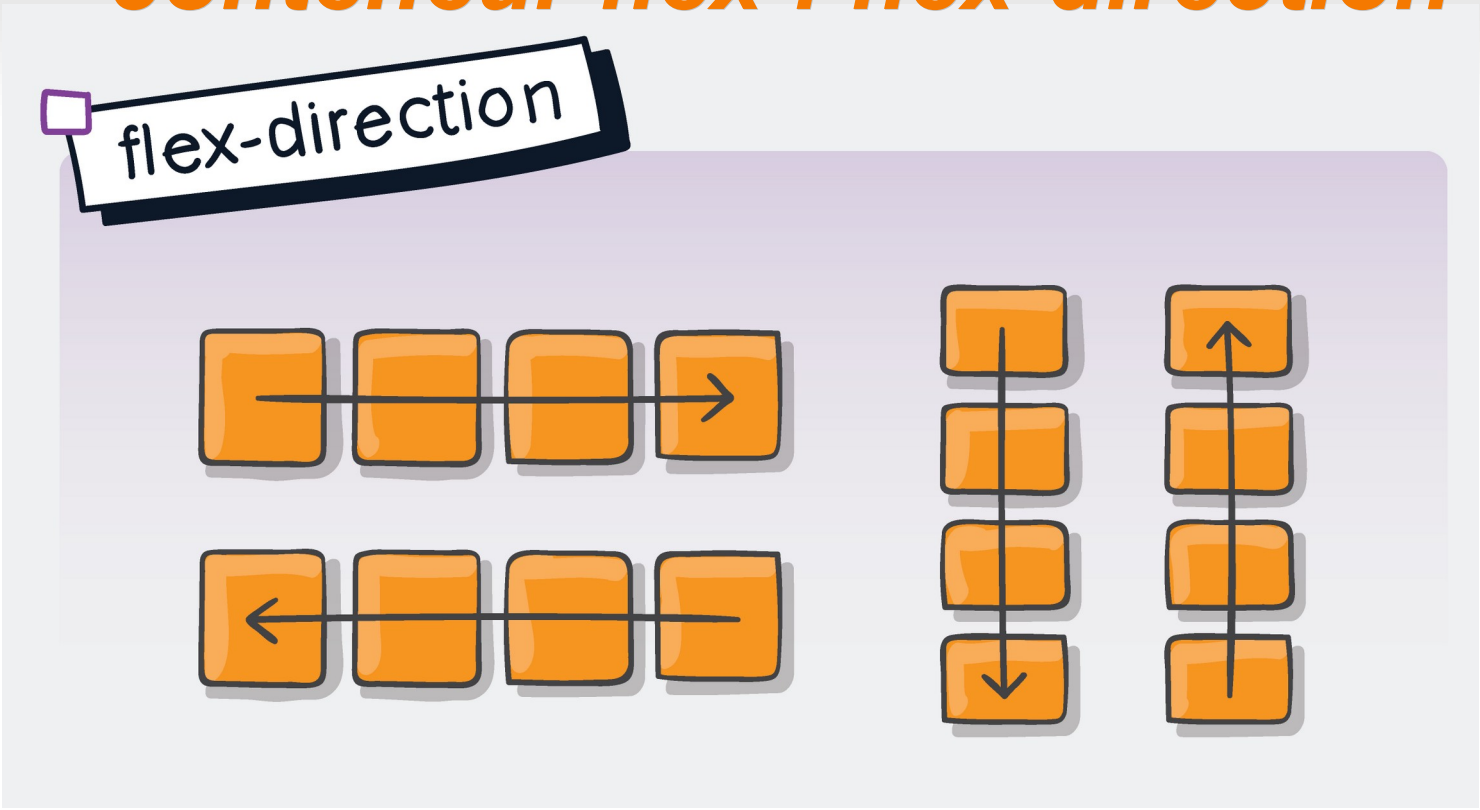




Pour définir un élément en flexbox, utiliser la propriété display : flex.

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

conteneur flex : flex-direction

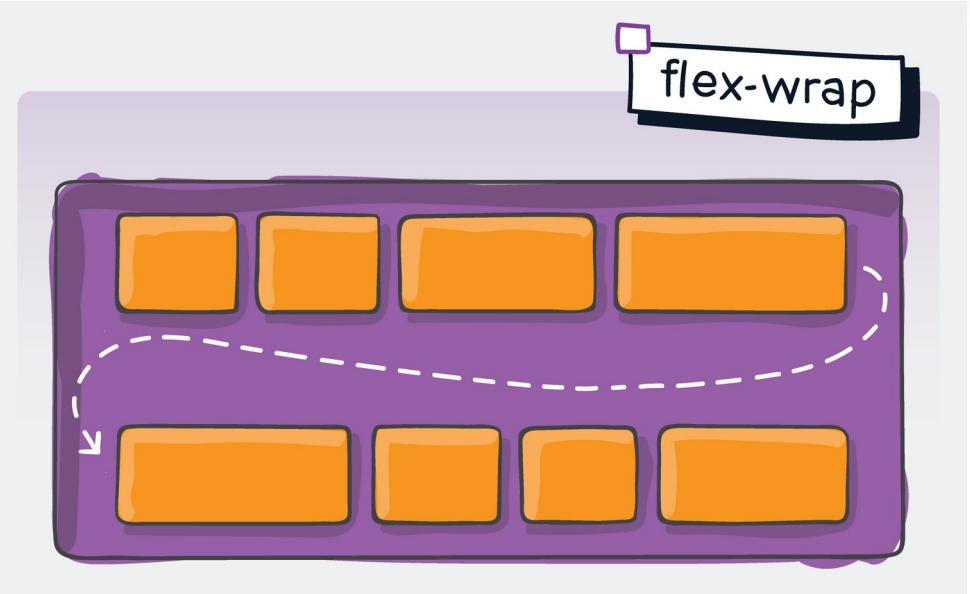


La propriété flex-direction permet de disposer les enfants du conteneur en ligne (**row**) ou en colonne (**column**). par défaut, (si l'on n'utilise pas la propriété), la valeur est **row .**

```
.container {
  flex-direction: row | row-reverse | column | column-reverse;
}
```

Conteneur flex : flex-wrap

Par défaut, les éléments flexibles tentent de tenir sur une seule ligne. Vous pouvez modifier cela et permettre aux éléments de s'enrouler selon les besoins grâce à cette propriété.



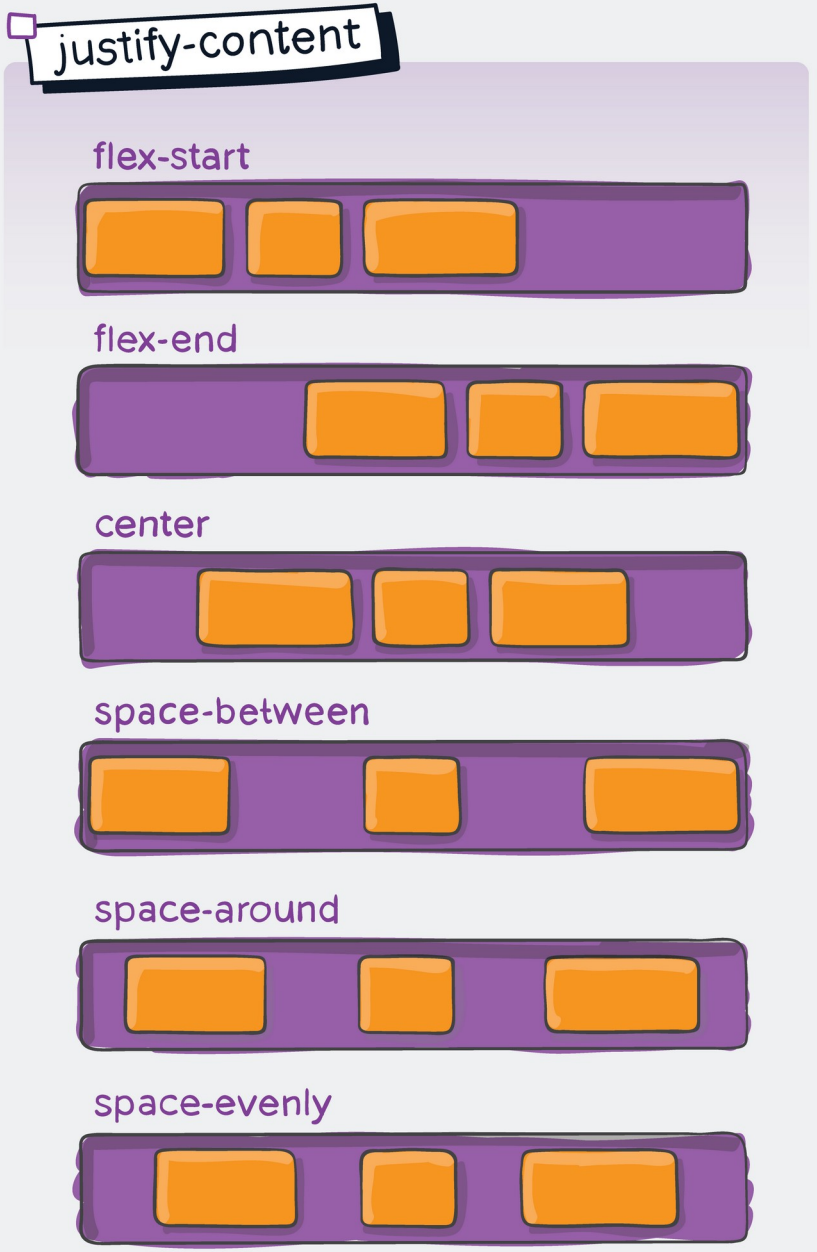

```
.container {
  flex-wrap: nowrap | wrap | wrap-reverse;
}
```

- **nowrap** (par défaut) : tous les éléments flexibles seront sur une seule ligne
- **wrap** : les éléments flexibles s'étendent sur plusieurs lignes, de haut en bas.
- **wrap-reverse** : les éléments flexibles s'étendent sur plusieurs lignes, de bas en haut.

Conteneur flex : justify-content

Cette propriété définit l'alignement des éléments enfants du conteneur flex, le long de l'axe principal.

```
.container {
  justify-content: flex-start | flex-end |
center | space-between | space-around |
space-evenly;
}
```



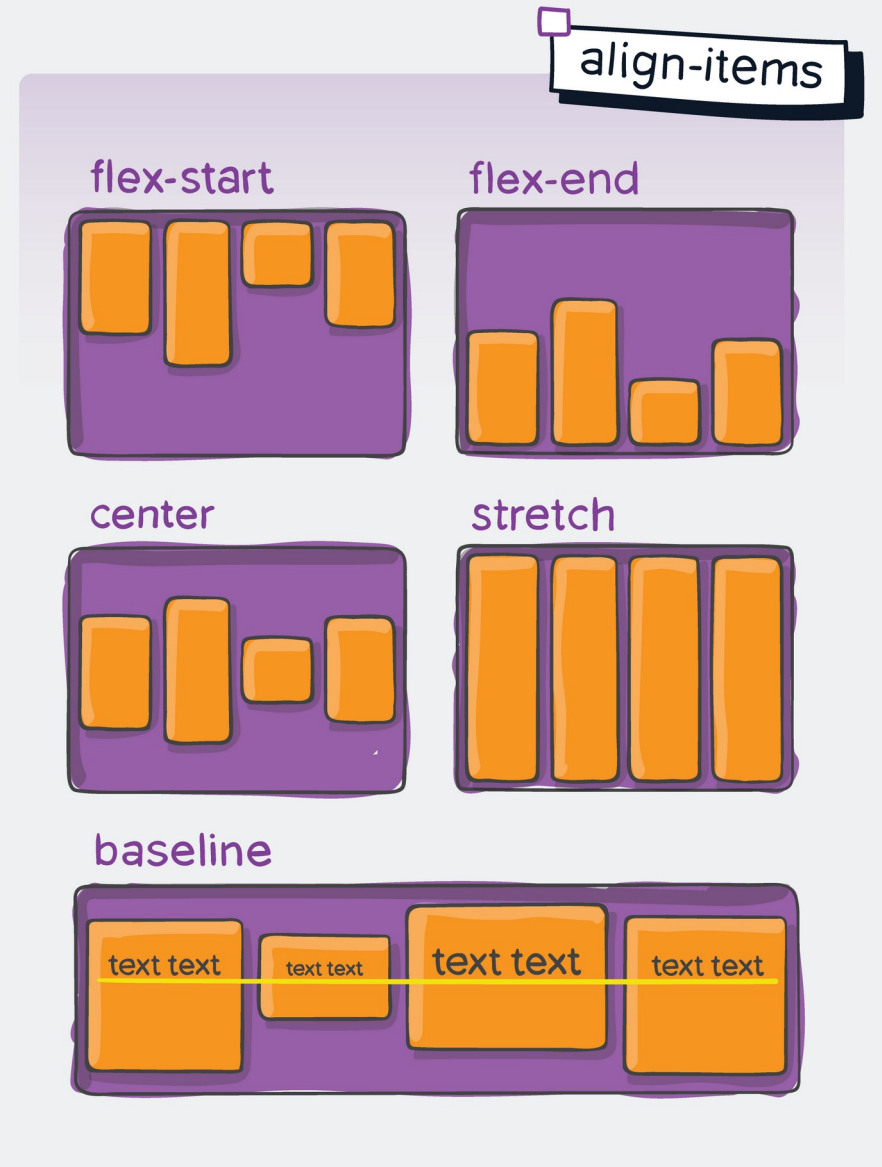
Conteneur flex : justify-content - valeurs

- **flex-start (default)** : les éléments sont disposés au début du conteneur.
- **flex-end** : les éléments sont disposés à la fin du conteneur
- **center** : les éléments sont centrés sur l'axe principal
- **space-between** : les éléments sont répartis uniformément sur l'axe principal ; le premier élément se trouve au début de l'axe, le dernier élément, à la fin de l'axe
- **space-around** : les éléments sont répartis uniformément sur l'axe principal et disposent d'un espace égal autour d'eux.
- **space-evenly** : les éléments sont répartis de manière à ce que l'espace entre deux éléments (et l'espace par rapport aux bords) soit égal.

Conteneur flex : align-items

Ceci définit le comportement par défaut de la disposition des éléments flex le long de l'axe secondaire sur la ligne actuelle. Il s'agit de la version justify-content pour l'axe secondaire (perpendiculaire à l'axe principal).

```
.container {
  align-items: stretch | flex-start | flex-end |
center | baseline;
}
```



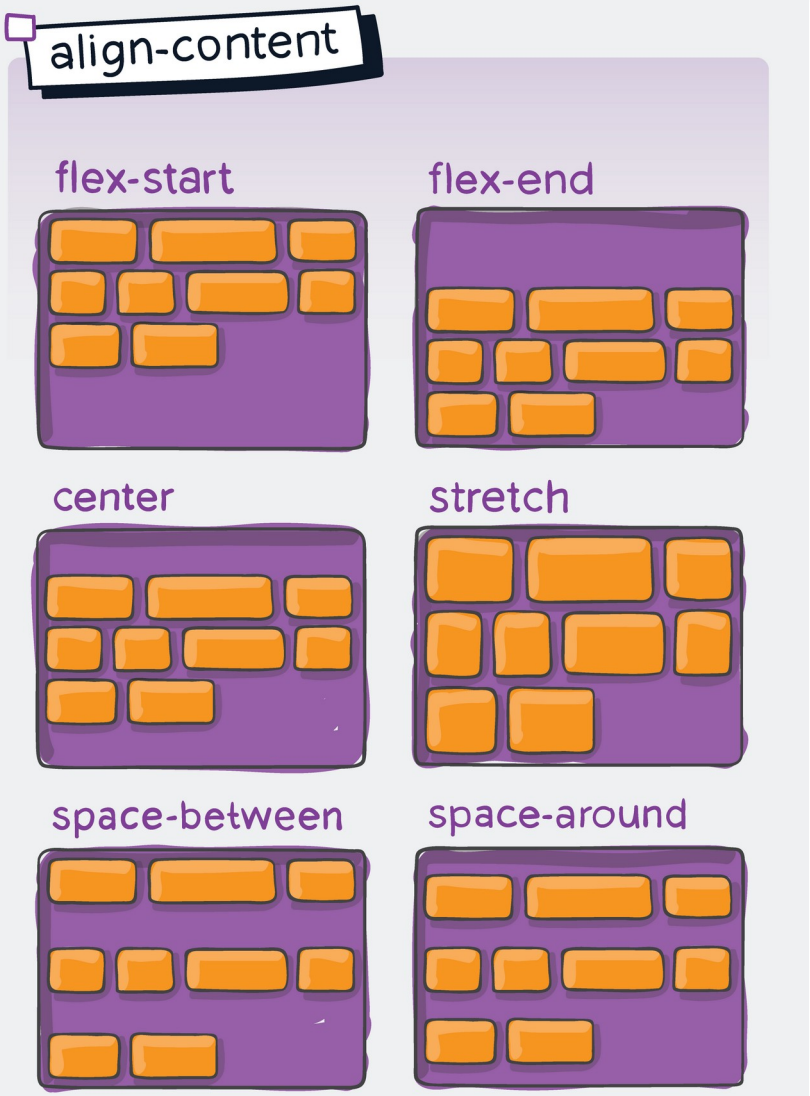
Conteneur flex : align-items - valeurs

- **stretch (défaut)** : les éléments s'étirent pour remplir le conteneur.
- **flex-start** : les éléments sont disposés au début de l'axe secondaire.
- **flex-end** : les éléments sont disposés à la fin de l'axe seconaire
- **center** : les éléments sont centrés sur l'axe secondaire
- **baseline** : les éléments sont répartis sur la ligne de base du texte.

Conteneur flex : align-content

Cette propriété permet d'aligner les éléments d'un conteneur flex lorsqu'il y a de l'espace supplémentaire dans l'axe secondaire, de la même manière que justify-content aligne les éléments individuels dans l'axe principal.

```
.container {
  align-content: flex-start | flex-end |
center | space-between | space-around |
space-evenly | stretch | normal;
}
```



Conteneur flex : align-content - valeurs

- **normal (défaut)** : les éléments sont disposés dans leur position par défaut.
- **flex-start** : les éléments sont disposés au début du conteneur.
- **flex-end** : les éléments sont disposés à la fin du conteneur.
- **center** : les éléments sont centrés sur l'axe secondaire dans le conteneur
- **stretch** : les éléments occupent de façon uniforme tout l'espace sur l'axe secondaire.
- **space-between**: les éléments sont répartis uniformément ; la première ligne se trouve au début du conteneur tandis que la dernière se trouve à la fin.
- **space-around** : les élémens sont répartis de manière uniforme avec un espace égal autour de chaque ligne
- **space-evenly** : les éléments sont répartis de manière uniforme avec un espace égal autour d'eux

Flexbox - références

Pour aller plus loin dans la compréhension et l'utilisation des flexbox :

- https://www.youtube.com/watch?v=DPHgIIdrmFk
- https://css-tricks.com/snippets/css/a-guide-to-flexbox/
- Doc mozilla : Concept des flexbox
- https://flexboxfroggy.com/#fr

Les media queries

Les media queries

- Les media queries permettent d’appliquer des styles différents en fonction des caractéristiques de l'appareil utilisé pour afficher la page web. Cela inclut notamment :
  - La taille de l'écran
  - La résolution
  - L'orientation (portrait ou paysage)
- Les media queries permettent ainsi d'adapter le design et la mise en page d'un site web en fonction du support utilisé, qu'il s'agisse d'un ordinateur de bureau, d'une tablette ou d'un smartphone. C’est ce que l’on appelle couramment le **responsive design**.

Syntaxe des media queries

- Utiliser la règle CSS : **@media (requête) { … }**
- Par exemple :

```
@media (min-width : 1200px) {
        #entete {
                display : flex ;
        }
}
```

Ici, l'élément #entête aura sa propriété *display* à *flex* uniquement si la largeur de la zone d'affichage (viewport) est supérieure ou égale à 1200px .

- Il est possible de spécifier le type de média : **@media screen (requête) { … }**
- Types de médias possibles :
  - **all** (par défaut si non spécifié) : Tous les médias
  - **print** : imprimantes
  - **screen** : écrans
  - **speech** : Synthèse vocale

Caractéristiques média (media features)

Voici la liste des caractéristiques les plus couramment utilisées :

Nom	Description	Exemple
width	La largeur de la zone d'affichage (viewport)	@media (max-width:500px) { … }
height	La hauteur de la zone d'affichage	@media (min-height:900px) { … }
aspect-ratio	Le rapport largeur/hauteur de la zone d'affichage	@media (aspect-ratio: 16 / 9) {...}
orientation	L'orientation la zone d'affichage	@media (orientation : portrait) {...}
resolution	La densité de pixel pour l'appareil d'affichage	@media (min-resolution: 72dpi) {}
monochrome	L'affichage en noir & blanc.	@media (monochrome) { … } @media (monochrome : 0) { … }

Voir toutes les caractéristiques détaillées

Intégration HTML

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

- Cette balise est à placer dans la partie <head> du document HTML :
- Elle est utilisé pour définir la manière dont un site web est affiché sur les appareils mobiles.
- La balise **<meta>** est utilisée pour fournir des informations supplémentaires sur la page web. Dans ce cas, l'attribut ***name*** est défini sur **"viewport"**, ce qui indique que nous allons spécifier les paramètres de la zone d'affichage.
- L'attribut ***content*** définit les valeurs de ces paramètres. Dans cet exemple, **"width=device-width"** signifie que la largeur de la zone d'affichage sera égale à la largeur de l'appareil utilisé pour visualiser la page. Cela permet au site de s'adapter à différentes tailles d'écran.
- L'option **"initial-scale=1"** indique que le niveau de zoom initial sera de 1. Cela signifie que le contenu sera affiché à une taille normale sans zoom.
- **En résumé, ce code HTML permet d'optimiser l'affichage d'un site web sur les appareils mobiles en ajustant automatiquement la taille de l'affichage et en évitant le zoom initial.**

CSS GRID

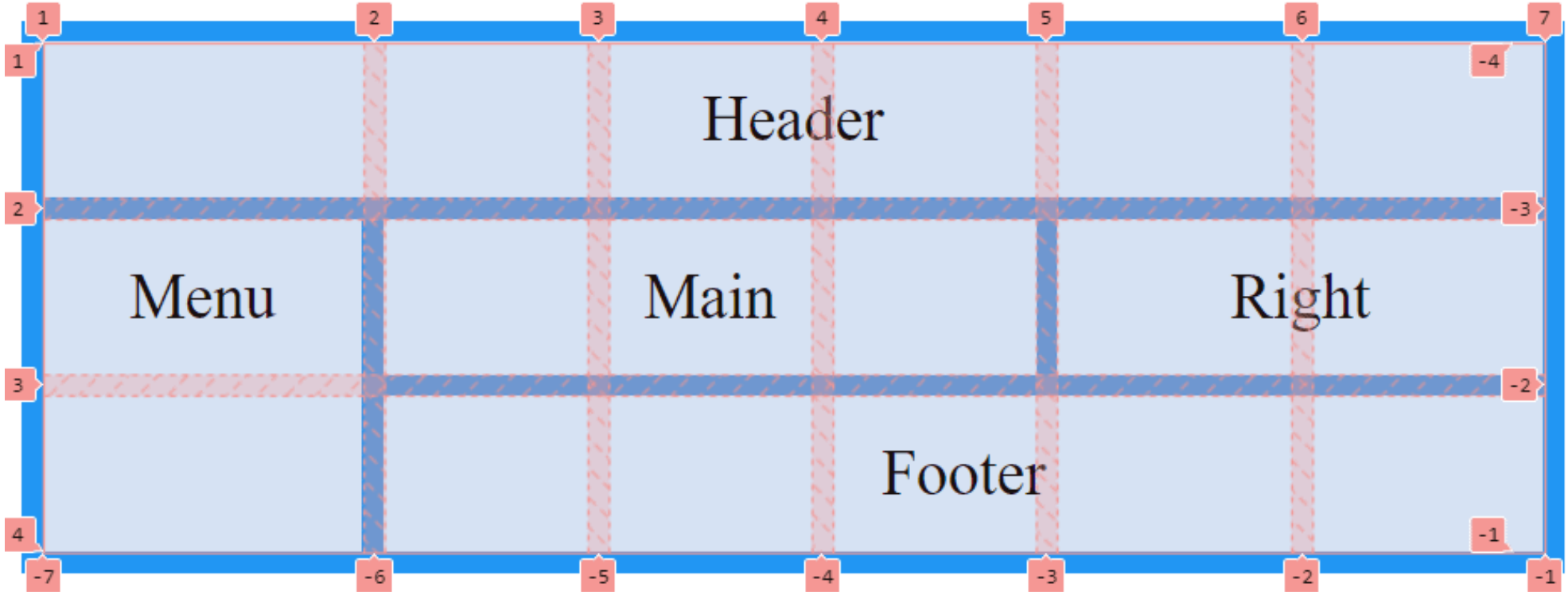
- CSS grid permet de gérer la mise en page en deux dimensions contrairement aux boîtes flex qui n'en gèrent qu'une (flex-direction : row / column)
- CSS grid est une spécification CSS 3. C'est donc un framework natif (pas besoin d'installer Bootstrap ou autres framework CSS).
- Comme pour les boîtes flex nous avons les notions de conteneur et d'éléments enfant (*grid-item*).
- Pour déclarer une grille, on applique la propriété **display: grid** sur le conteneur.

CSS GRID ITEM

- Un grid Item est un enfant **direct** d'un Grid Container. Les petits-enfants ne participent pas au modèle de grille.
- Il n'est pas nécessaire d'appliquer une propriété CSS pour transformer un élément en Grid Item. Il suffit que le parent direct est la propriété **display : grid**.

```
<section class="grid-container">
    <div>Salade</div>
    <div>Tomate</div>
    <div>Oignon</div>
    <div>Bière</div>
</section>
```

CSS Grid : Modèle de grille



CSS Grid : Propriétés du conteneur

- **grid-template-columns**
  Cette propriété définie le nombre et la largeur des colonnes de la grille
- **grid-template-rows**
  Cette propriété spécifie ne nombre ainsi que la hauteur des rangés de la grille.
- **grid-template-areas**
  Cette propriétés permet d'associer un nom personnalisé aux différentes zone de la grille.

CSS GRID : Propriétés des Grid Items

- **grid-row-start** : N° de la ligne horizontale de départ.
- **grid-row-end** : N° de la ligne horizontale de fin.
- **grid-column-start** : N° de la ligne verticale de départ.
- **grid-column-end**: N° de la ligne verticale de fin.
- **grid-row** : raccourci grid-row-start / grid-row-end
- **grid-column** : raccourci grid-column-start / grid-colum-end
- **grid-area** : raccourci **grid-row / grid-column.** Peut-être aussi utilisé pour spécifier une zone personnalisé, définie avec *grid-template-area* sur le conteneur.

Grid CSS : Ressources

- https://css-tricks.com/snippets/css/complete-guide-grid/
- CSS grid layout sur Mozilla
- https://cssgridgarden.com/#fr

Compléments

Curseur

- Détermine l'apparence du curseur au survol
- Valeurs
  - **default** : Curseur par défaut
  - **pointer** : Main
  - **crosshair** : Viseur
  - **help** : Point d'interrogation
  - **wait** : Attente
  - **text** : Texte
  - **move** : Objet déplaçable
  - **...-resize** : Changer la taille en fonction de la direction

```
élément
{
    cursor : apparence;
}
```

Listes

Action	Propriété	Valeur
Type de puces et de numérotation	**list-style-type :**	**decimal, upper-roman, lower-latin, disc, circle, square ou none**
Permet de personnaliser les puces avec une image	**list-style-image :**	**url(image.png);**
Spécifie le retrait des puces	**list-style-position :**	**inside; outside;**
Raccourci global vers les propriétés des listes	**list-style :**	**type position url();**

- Des menus avec des listes
  - http://css.alsacreations.com/Galeries-de-menus-en-CSS
  - http://www.alsacreations.com/livre/?/Exemples/exdouze
  - http://www.cssplay.co.uk/menus/visitedmenu.html

TP

projet8.html

```html
<nav id="menu">
  <ul>
    <li><a href="#presentation">À propos</a></li>
    <li><a href="#services">Services</a></li>
    <li><a href="#contact">Contact</a></li>
    <li><a class="bouton" href="#">Devis</a></li>
  </ul>
</nav>
```

- Créer votre menu horizontal à l'aide des styles suivants :

```css
nav ul {
    margin-top: 10px;
    margin-bottom: 10px;
    padding: 0;
    list-style: none;
}
nav a {
    text-decoration: none;
}
#menu ul > li {
    margin-bottom: 10px;
}
#menu ul > li > a {
    font-size: 16px;
    font-weight: bold;
}
#footer nav > ul > li {
    margin-bottom: 15px;
}
#footer nav > ul > li a {
    text-decoration: none;
}
@media (min-width: 800px) {
    #menu {
        flex-grow: 2;
    }
    #menu ul {
        text-align: right;
    }
    #menu ul > li {
        display: inline-block;
        margin-left: 20px;
    }
    #menu .bouton {
        margin-left: 30px;
    }
}
```

Outils de validation

Les outils de validation sont des partenaires précieux :

- Outil de validation pour HTML:
  - https://validator.w3.org/
- Outil de validation pour CSS:
  - https://jigsaw.w3.org/css-validator/
- Outils de validation pour l'accessibilité d'un document:
  - Wave : https://wave.webaim.org/
  - https://www.accessibilitychecker.org/

Notion de template

- Template = canevas = modèle de mise en forme
- Un template est composé de deux éléments
  - Une page de structure et de données en **HTML**
  - Une feuille de style de présentation en **CSS**

- La création de template est l'étape préalable à la création des thèmes Wordpress
  - -> il ne restera qu'à remplacer les données statiques par des données dynamiques grâces au PHP

Pour en savoir plus sur les CSS

- https://www.alsacreations.com/tutoriels/
- http://css.mammouthland.net/
- https://openweb.eu.org/css
- https://developer.mozilla.org/fr/docs/Web/CSS
- https://css-tricks.com/