Formation Développeur Expert Java

CSS3

- Création d'IOcean en 2001
- 2 Implantations :
  - Siège à Montpellier
  - Agence Paris
- 2 Activités :
  - ESN (SSII)
  - Editeur
- En 2016, 30 personnes pour 2,2 M€

Votre formateur

IOCEAN

Les projets informatiques dont les clients parlent avec plaisir

Sommaire

CSS3

Histoire

Le langage CSS (pour Cascading Style Sheets) permet de décrire la présentation d'un document pour le mettre en forme.

Chaque version de CSS est une extension de la version précédente. Ainsi, la version CSS1 est un sous-ensemble de CSS2 qui est lui-même un sous-ensemble de CSS3.

Les différentes versions sont :

En 1996, version officielle CSS1

En 1998, version CSS2 qui est un échec puis une version CSS2.1 à partir de 2001

En 2007, version CSS3

Définition

Depuis l'arrivée de HTML5 et CSS3, le rôle des deux langages est bien distincts :

HTML permet de définir le contenu et sa structure sémantique : tel contenu est le menu, tel autre contenu définit un article dans la page, …

CSS permet de définir comment les contenus vont s'afficher dans la page : la police de caractère, les couleurs, si le menu s'affiche en haut ou à gauche, …

Il est très important de conserver cette structuration dans une page HTML afin de pouvoir profiter de la puissance de chacun des langages.

De plus, certains navigateurs ne gèrent pas ou partiellement certaines propriétés CSS.
C'est particulièrement le cas pour les anciens navigateurs (IE par exemple). Il est donc nécessaire de tester le rendu attendu sur les navigateurs cibles.

Définition

La syntaxe pour écrire un code CSS est la suivante :

```
sélecteur {
        propriété: valeur;
}
```

Avec :

sélecteur : Permet de préciser l'élément ou les éléments HTML dont on veut changer l'apparence

propriété : Permet de préciser quelle propriété de l'apparence on veut modifier (la couleur, la taille, …)

valeur : Permet de préciser quelle valeur on veut donner à l'apparence (la couleur, la taille, …)

Exemple pour mettre la couleur grise en fond de page :

```
body {
        background-color: grey;
}
```

Définition

Pour mettre du CSS dans une page HTML, vous avez 3 possibilités :

- En chargeant un fichier.css externe à la page
- Dans une balise ***<style> </style>*** qui se trouve dans le fichier HTML
- Directement dans les balises HTML en utilisant l'attribut style

Définition

La syntaxe pour le chargement d'un fichier .css externe est la suivante :

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

L'attribut ***href*** permet de définir l'URL où charger le fichier CSS.

Le contenu du fichier CSS pourra être le suivant

```
body {
        background-color: grey;
}
```

L'avantage de cette syntaxe est de pouvoir partager un fichier CSS avec plusieurs fichiers HTML. Ainsi, si une valeur change dans le CSS, elle sera changée pour tous les fichiers HTML qui l'utilisent.

Définition

La syntaxe pour définir du CSS directement dans le fichier HTML est la suivante :

```
<style>
        body {
                background-color: grey;
        }
</style>
```

Ce code se trouve généralement dans la balise ***<head> </head>*** du fichier HTML mais il peut être n'importe où.

Il permet de définir un style particulier pour une page donnée.

Définition

Pour définir du CSS directement dans une balise, il faut utiliser l'attribut ***style*** :

*<body **style**="background-color: grey;">*

Ce code permet de définir un style particulier pour une balise particulière.

Il est recommandé de ne pas trop l'utiliser afin de faciliter la maintenance et l'évolutivité d'un site Web.

Définition

Pour mettre des commentaires en CSS, il faut entourer le commentaire avec « /* » et « */ »

Exemple :

```
/* Ceci est un commentaire */


/*
    Ceci est un commentaire
     sur plusieurs lignes
*/
```

Les sélecteurs

Les sélecteurs permettent de définir sur quelles balises s'appliquent le code CSS.

Les sélecteurs simples sont les suivants :

balise : s'applique à toutes les balises du nom « élément ».

```
body {
   …
}
```

#identifiant : s'applique à la balise qui a un attribut « id=identifiant ».

```
#logo {
   …
}
```

.classe : s'applique à toutes les balises qui ont un attribut « class=classe ».

```
.error {
   …
}
```

Les sélecteurs

Il est possible d'appliquer les mêmes règles à plusieurs sélecteurs.

Pour cela, on les sépare par une « , ».

Exemples :

```
/* toutes les balises h6 et p */
h6, p {
  …
}

/* toutes les balises h1, celles qui ont la class titre et la balise id="entete" */
h1, .titre, #entete {
  …
}
```

Les sélecteurs

Il est possible de combiner les sélecteurs.

Pour faire un ET LOGIQUE, on accole les sélecteurs.

```
/* toutes les balises h6 qui ont la classe titre */
h6.titre {
  …
}
```

Pour les sélecteurs descendants d'un autre (inclus dans celui-ci), on utilise un espace.

```
/* toutes les balises p qui sont dans une balise qui a la class titre */
.titre p {
  …
}
```

Les sélecteurs

Pour les sélecteurs fils d'un autre (directement inclus dans celui-ci), on utilise « > ».

Exemples :

```
/* toutes les balises p qui sont des enfants directs d'une balise qui a la class titre */
.titre > p {
  …
}
```

Pour les sélecteurs suivants (juste après un autre), on utilise « + ». Exemples :

```
/* toutes les balises p situées juste après une balise qui a la class titre */
.titre + p {
  …
}
```

Les sélecteurs

Il est possible de tester des attributs d'un sélecteur :

élément[attr] : Permet de cibler un élément qui possède un attribut attr.

élément[attr="valeur"] : Permet de cibler un élément qui possède un attribut attr et dont la valeur de cet attribut est exactement valeur.

élément[attr~="valeur"] : Permet de cibler un élément qui possède un attribut attr et dont la valeur de cet attribut une liste de mots séparés par des blancs dont un vaut exactement valeur.

élément[attr|="valeur"] : Permet de cibler un élément qui possède un attribut attr et dont la valeur de cet attribut est exactement valeur ou dont la valeur commence par valeur suivi immédiatement d'un tiret (U+002D). Cela peut notamment être utilisé pour effectuer des correspondances avec des codes de langues.

élément[attr^="valeur"] : Permet de cibler un élément qui possède un attribut attr et dont la première valeur commence par valeur.

élément[attr$="valeur"] : Permet de cibler un élément qui possède un attribut attr et dont la dernière valeur termine par valeur.

élément[attr*="valeur"] : Permet de cibler un élément qui possède un attribut attr et dont la valeur contient au moins une occurrence d'une chaîne de caractères qui contient valeur.

Les sélecteurs

Une **pseudo-classe** est un mot-clé ajouté à un sélecteur afin d'indiquer l'état spécifique dans lequel l'élément doit être pour être ciblé par la déclaration.

La pseudo-classe ***:hover***, par exemple, permettra d'appliquer une mise en forme spécifique lorsque l'utilisateur survole l'élément ciblé par le sélecteur (changer la couleur d'un bouton par exemple).

Les différentes pseudo-classe sont :

:active	:first	:indeterminate	:nth-last-child()	:right
:any	:first-child	:in-range	:nth-last-of-type()	:root
:any-link	:first-of-type	:invalid	:nth-of-type()	:scope
:checked	:fullscreen	:lang()	:only-child	:target
:default	:focus	:last-child	:only-of-type	:valid
:dir()	:focus-visible	:last-of-type	:optional	:visited
:disabled	:host	:left	:out-of-range	
:defined	:host()	:link	:read-only	
:empty	:host-context()	:not()	:read-write	
:enabled	:hover	:nth-child()	:required	

Les sélecteurs

Un **pseudo-élément** est un mot-clé ajouté à un sélecteur qui permet de mettre en forme certaines parties de l'élément ciblé par la règle.

Ainsi, le pseudo-élément ***::first-line*** permettra de ne cibler que la première ligne d'un élément visée par le sélecteur.

Les différents pseudo-élément sont :

::after	::backdrop
::before	::placeholder
::cue	::marker
::first-letter	::slotted
::first-line	::spelling-error
::selection	::grammar-error

Cascade et priorité

Pourquoi parle-t-on de feuille de style en cascade ?

Dans la pratique, plusieurs règles CSS s'appliquent en cascade et selon un ordre bien précis.

Nous pouvons donc retrouver la cascade de styles suivante :

1. Les styles du navigateur
2. Les styles définis par l'utilisateur
3. Les styles externes au document
4. Les styles internes au document
5. Les styles appliqués explicitement aux balises (via l'attribut style)

Chaque niveau surcharge le précédent et est donc « prioritaire ».

Le mot clé « **!important** » permet de préciser qu'une règle est prioritaire sur toutes les autres sans tenir compte de l'ordre de cascade.

```
p {
        color: blue !important;
}
```

Cascade et priorité des sélecteurs

Et que se passe-t-il lorsque plusieurs sélecteurs concernent le même élément ?

Il faut se référer aux règles de priorités des sélecteurs.
Les règles de priorité des sélecteurs peuvent parfois être difficiles à mémoriser.
C'est souvent la règle la plus précise qui l'emporte sur celle qui l'est moins…
Mais ce n'est pas toujours aussi simple (cf. spécifications).

A retenir, les 3 niveaux de « précisions », du plus précis au moins précis :

1. Le sélecteur d'identifiant
2. Le sélecteur de classe
3. Le sélecteur de type

Cascade et héritage

Enfin, le principe de cascade est également lié au principe d'héritage des propriétés.

Chaque enfant hérite des propriétés indiquées comme « héritables » de son parent (voir la spécification pour chaque balise).

HTML

```
<p>
   Bienvenue
   <span>chez vous</span>
</p>
```

CSS

```
html {
        font-size: 20px;
}
```

A votre avis, quelle sera la taille des textes des balises « p » et « span » ?

Cascade et combinaison

Quand plusieurs sélecteurs concernent le même élément, les propriétés se combinent.

** Bienvenue **
** Découvrez l'offre du jour !... **

```
p {
        font-size: 20px;
}
p.annonce {
        color: red;
}
p.precision {
        text-decoration: underline;
}
```

Bienvenue
Découvrez l'offre du jour…

Les propriétés - texte

Les propriétés de forme du texte

Propriété	Valeurs (exemples)	Description
font-family	*police1, police2, police3*, serif, sans-serif, monospace	Nom de police
@font-face	*Nom et source de la police*	Police personnalisée
font-size	1.3em, 16px, 120%...	Taille du texte
font-weight	bold, normal	Gras
font-style	italic, oblique, normal	Italique
text-decoration	underline, overline, line-through, blink, none	Soulignement, ligne au-dessus, barré ou clignotant
font-variant	small-caps, normal	Petites capitales
text-transform	capitalize, lowercase, uppercase	Capitales
font	-	Super propriété de police. Combine : font-weight, font-style, font-size, font-variant, font-family.

Les propriétés - texte

Les propriétés de forme du texte (suite)

Propriété	Valeurs (exemples)	Description
text-align	left, center, right, justify	Alignement horizontal
vertical-align	baseline, middle, sub, super, top, bottom	Alignement vertical (cellules de tableau ou éléments inline-block uniquement)
line-height	18px, 120%, normal...	Hauteur de ligne
text-indent	25px	Alinéa
white-space	pre, nowrap, normal	Césure
word-wrap	break-word, normal	Césure forcée
text-shadow	5px 5px 2px blue *(horizontale, verticale, fondu, couleur)*	Ombre de text

Les propriétés - couleur

Les propriétés de couleur et de fond

Propriété	Valeurs (exemples)	Description
color	*nom, rgb(rouge,vert,bleu), rgba(rouge,vert,bleu,transparence), #CF1A20...*	Couleur du texte
background-color	*Identique à color*	Couleur de fond
background-image	url('image.png')	Image de fond
background-attachment	fixed, scroll	Fond fixe
background-repeat	repeat-x, repeat-y, no-repeat, repeat	Répétition du fond
background-position	(x y), top, center, bottom, left, right	Position du fond
background	-	Super propriété du fond. Combine : background-image, background-repeat, background-attachment, background-position
opacity	0.5	Transparence

Les propriétés - liste

Les propriétés des listes

Propriété	Valeurs (exemples)	Description
list-style-type	disc, circle, square, decimal, lower-roman, upper-roman, lower-alpha, upper-alpha, none	Type de liste
list-style-position	inside, outside	Position en retrait
list-style-image	url('puce.png')	Puce personnalisée
list-style	20px	Super-propriété de liste. Combine : list-style-type, list-style-position, list-style-image.

Les propriétés - tableau

Les propriétés des tableaux

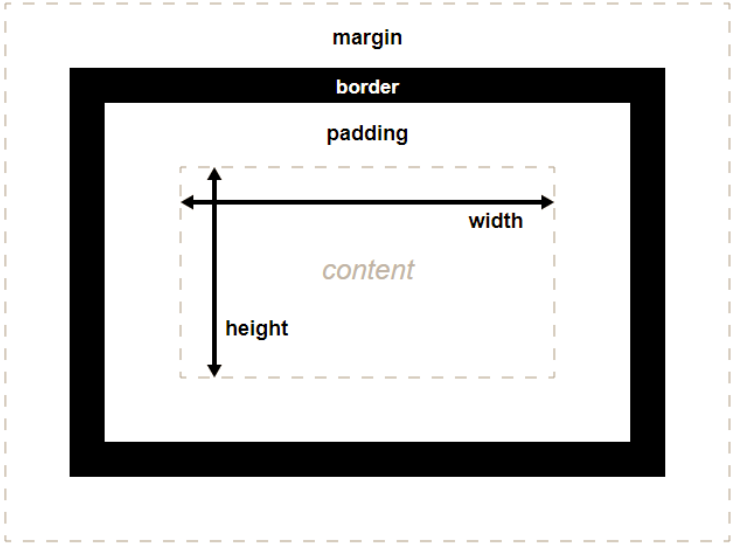
Propriété	Valeurs (exemples)	Description
border-collapse	collapse, separate	Fusion des bordures
empty-cells	hide, show	Affichage des cellules vides
caption-side	bottom, top	Position du titre du tableau

Les boîtes

Chaque élément d'un document est matérialisé soit :

- par une boîte,
- par une ligne,
- de façon particulière qui mélange les deux.

Les boîtes peuvent être ajustées
grâce à des propriétés CSS spécifiques
selon le modèle de boite (« box model »).



Les boîtes

Propriétés des boîtes

Propriété	Valeurs (exemples)	Description
width	150px, 80%...	Largeur
height	150px, 80%...	Hauteur
min-width	150px, 80%...	Largeur minimale
max-width	150px, 80%...	Largeur maximale
min-height	150px, 80%...	Hauteur minimale
max-height	150px, 80%...	Hauteur maximale

Les boîtes

Propriétés des boîtes

Propriété	Valeurs (exemples)	Description
margin-top	23px	Marge en haut
margin-left	23px	Marge à gauche
margin-right	23px	Marge à droite
margin-bottom	23px	Marge en bas
margin	23px 5px 23px 5px (haut, droite, bas, gauche)	Super-propriété de marge. Combine : margin-top, margin-right, margin-bottom, margin-left.

Les boîtes

Propriétés des boîtes

Propriété	Valeurs (exemples)	Description
padding-left	23px	Marge intérieure à gauche
padding-right	23px	Marge intérieure à droite
padding-bottom	23px	Marge intérieure en bas
padding-top	23px	Marge intérieure en haut
padding	23px 5px 23px 5px (haut, droite, bas, gauche)	Super-propriété de marge intérieure. Combine : padding-top, padding-right, padding-bottom, padding-left.

Les boîtes

Propriétés des boîtes

Propriété	Valeurs (exemples)	Description
border-width	3px	Épaisseur de bordure
border-color	nom, rgb(rouge,vert,bleu), rgba(rouge,vert,bleu,transparence), #CF1A20...	Couleur de bordure
border-style	solid, dotted, dashed, double, groove, ridge, inset, outset	Type de bordure
border	3px solid black	Super-propriété de bordure. Combine border-width, border-color, border-style. Existe aussi en version border-top, border-right, border-bottom, border-left.
border-radius	5px	Bordure arrondie
box-shadow	6px 6px 0px black (horizontale, verticale, fondu, couleur)	Ombre de boîte

Les propriétés - positionnement et affichage

Les propriétés de positionnement et d'affichage

Propriété	Valeurs (exemples)	Description
display	block, inline, inline-block, table, table-cell, none...	Type d'élément (block, inline, inline-block, none…)
visibility	visible, hidden	Visibilité
clip	rect (0px, 60px, 30px, 0px) rect (haut, droite, bas, gauche)	Affichage d'une partie de l'élément
overflow	auto, scroll, visible, hidden	Comportement en cas de dépassement
float	left, right, none	Flottant
clear	left, right, both, none	Arrêt d'un flottant
position	relative, absolute, static	Positionnement

Les propriétés - positionnement

Les propriétés de positionnement

Propriété	Valeurs (exemples)	Description
top	20px	Position par rapport au haut
bottom	20px	Position par rapport au bas
left	20px	Position par rapport à la gauche
right	20px	Position par rapport à la droite
z-index	10	Ordre d'affichage en cas de superposition. La plus grande valeur est affichée par-dessus les autres.

Le positionnement

Le positionnement d'un élément dépend de son type (block / inline / etc.), des propriétés de son affichage et de la structure dans laquelle il s'imbrique.

display: block;

display: inline;

display: inline-block;

display: table;

display: grid;

display: flex;

…

position: static;

position: relative;
position: absolute;
position: fixed;

top / bottom / left / right

z-index

float: left;
float: right;
clear: both;

Les tailles

Les textes, les boites, les marges, les bordures, etc. ont besoin d'une unité de taille pour s'afficher.

Les plus utilisées sont les suivantes :

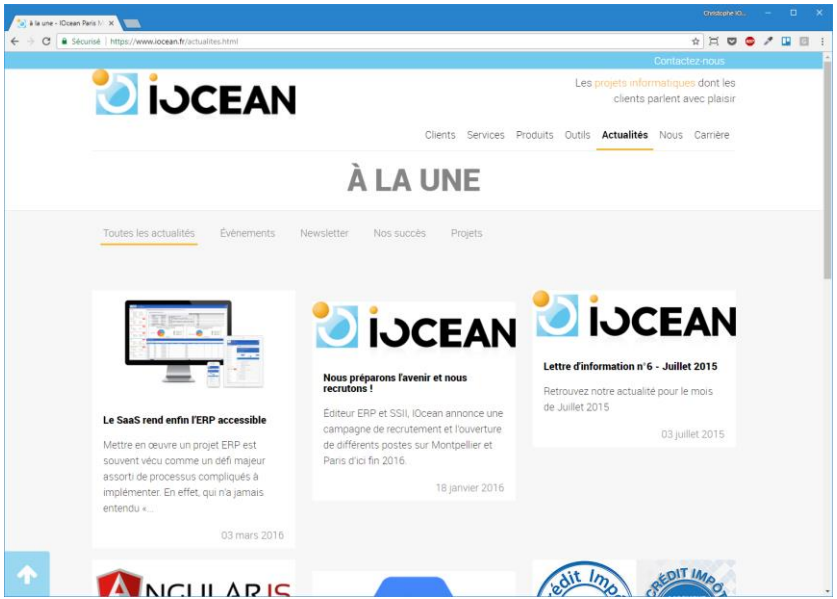
- **px** = pixel, unité fixe
- **%** = pourcentage du parent / ancêtre
- **em** = taille relative à celle du parent
- **rem** = taille relative à la racine

Le responsive design

Le responsive design est le principe qui permet à un site web de s'afficher différemment en fonction de la taille du navigateur.

Ainsi un même site ne s'affichera pas de la même façon si on l'affiche sur un PC ou sur un SmartPhone.

Exemples :



Le responsive design

Pour appliquer ce principe, CSS3 a ajouté la notion de média query.

Il y a deux façons d'utiliser les média queries :

en chargeant une feuille de style différente en fonction de la règle :

```
<link rel="stylesheet" media="screen and (max-width: 1280px)" href="petite_resolution.css" />
```

en écrivant la règle directement dans le fichier CSS :

```
@media screen and (max-width: 1280px) {
    …
}
```

Possibilité de cibler un média avec un format spécifique

Propriétés CSS qui s'appliquent EXCLUSIVEMENT au format spécifié

Le responsive design

Exemple de certains critères disponibles :

- ***color*** : gestion de la couleur (en bits/pixel).
- ***height*** : hauteur de la zone d'affichage (fenêtre).
- ***width*** : largeur de la zone d'affichage (fenêtre).

- ***device-height*** : hauteur du périphérique.
- ***device-width*** : largeur du périphérique.
- ***orientation*** : orientation du périphérique (portrait ou paysage).
- ***media*** : type d'écran de sortie. Quelques-unes des valeurs possibles :
  - **screen :** écran « classique » ;
  - **print** : impression ;
  - **all** : tous les types d'écran.

Le responsive design

On peut rajouter le préfixe ***min-*** ou ***max-*** devant la plupart de ces critères.
Ainsi, ***min-width*** signifie « Largeur minimale », ***max-height*** « Hauteur maximale », etc.
Les règles peuvent être combinées à l'aide des mots suivants :

- ***only*** : « uniquement » ;
- ***and*** : « et » ;
- ***not*** : « non ».

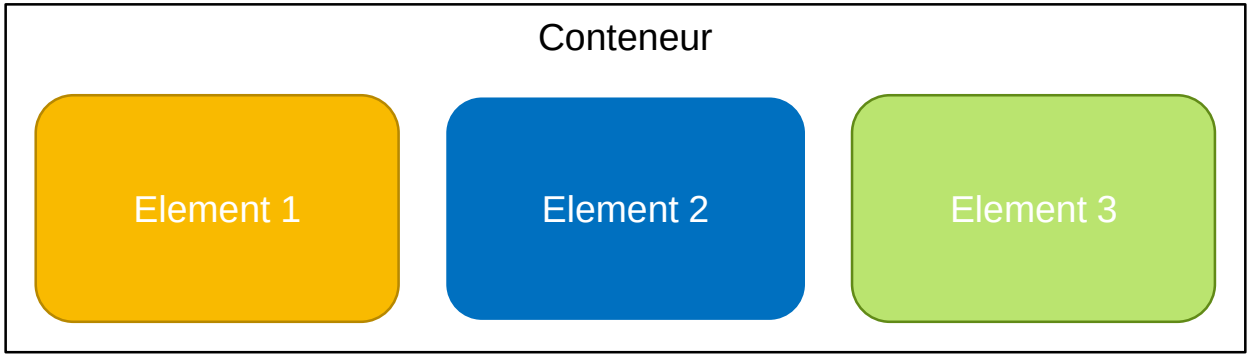
Exemples :

```
/* Sur les écrans, quand la largeur de la fenêtre fait au maximum 1280px */
@media screen and (max-width: 1280px)

/* Sur tous types d'écran, quand la largeur de la fenêtre est comprise entre 1024px et 1280px */
@media all and (min-width: 1024px) and (max-width: 1280px)

/* Pour l'impression */
@media print
```

Flexbox

Flexbox permet de réaliser pratiquement n'importe quelle mise en page.

Le principe de Flexbox est d'avoir un conteneur qui contient plusieurs éléments :



En ajoutant la propriété ***display: flex;*** sur le conteneur, les éléments à l'intérieur sont agencés en mode Flexbox (horizontalement par défaut).

Flexbox

En utilisant la propriété ***flex-direction***, on peut indiquer si les éléments sont agencés horizontalement (par défaut) ou verticalement. Cela définit ce qu'on appelle l'axe principal.

Exemples :

```
#conteneur
{
    display: flex;
    flex-direction: row;
}
```

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

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

Flexbox

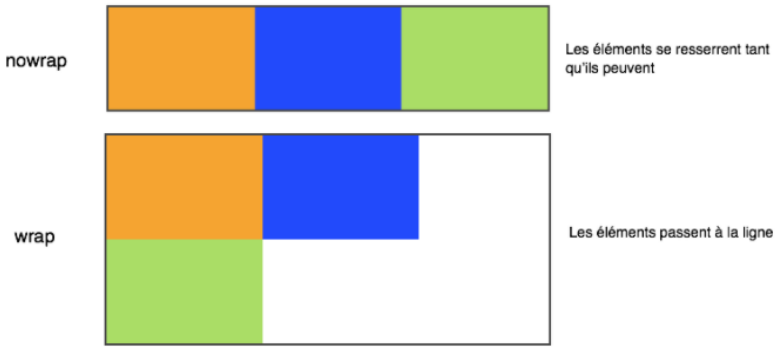
Par défaut, les éléments essaient de rester sur la même ligne s'ils n'ont pas la place Vous pouvez demander à ce que les éléments aillent à la ligne lorsqu'ils n'ont plus la place avec la propriété ***flex-wrap*** qui peut prendre ces valeurs :

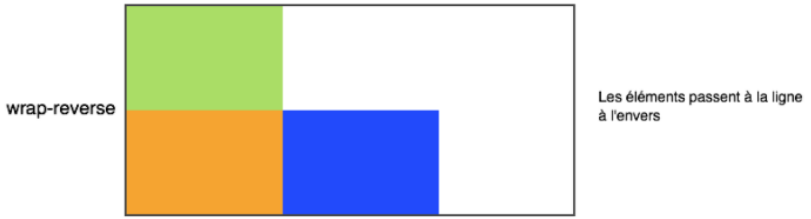
nowrap : pas de retour à la ligne (par défaut)

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

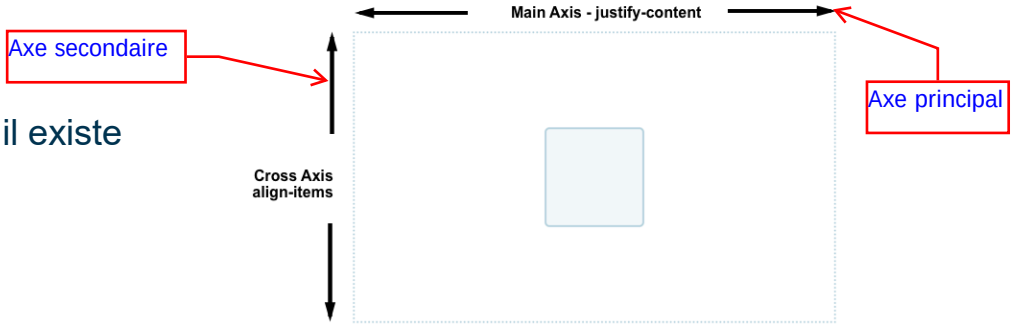
wrap-reverse : les éléments vont à la ligne lorsqu'il n'y a plus la place en sens inverse

Exemples :





Flexbox



Pour changer l'alignement des éléments, il existe les propriétés :

- ***justify-content*** *(pour l'axe principal)*
- ***align-items*** *(pour l'axe secondaire)*

justify-content*, principales valeurs :*

flex-start : alignés au début

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

align-items principales valeurs :

flex-start : les éléments flexibles alignés au début

flex-end : les éléments flexibles alignés à la fin

start : *tous alignés au début*

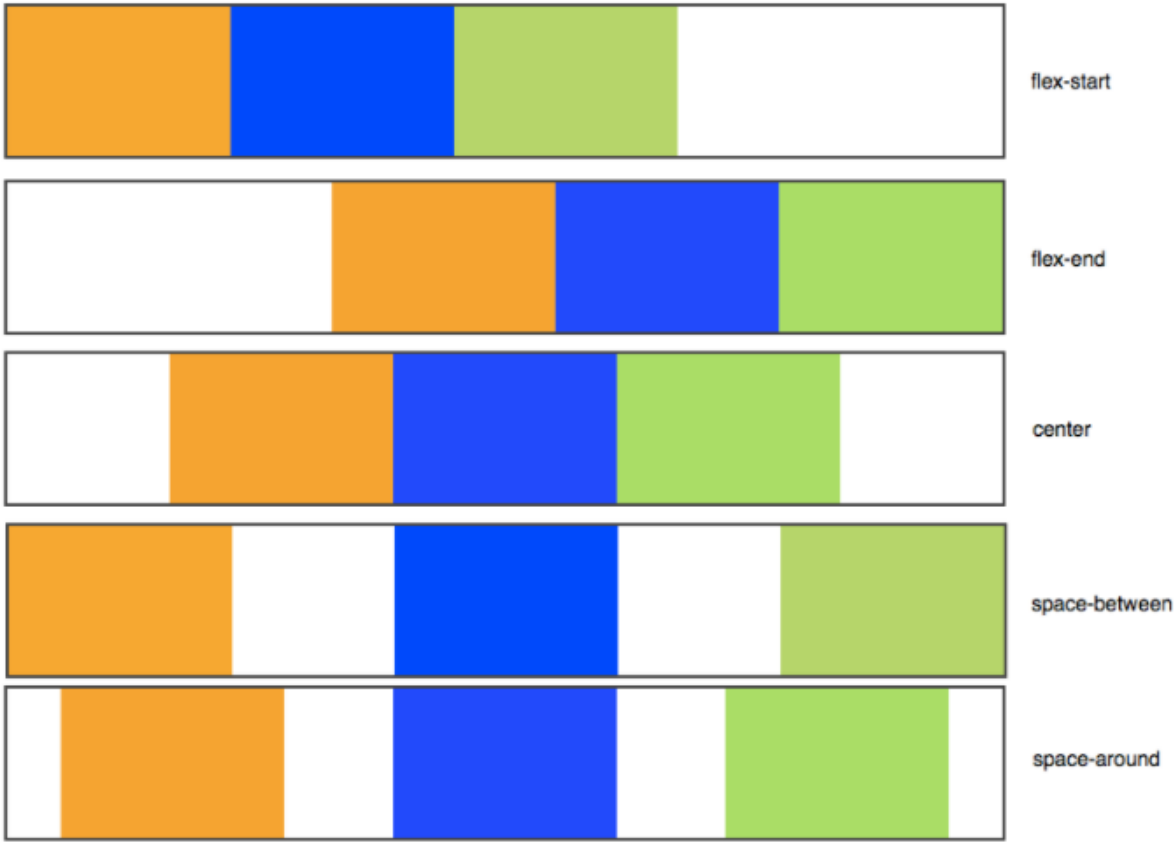
end *: tous alignés à la fin*

center : alignés au centre

stretch : les éléments sont étirés

Flexbox

Exemples :



Pour aller plus loin

CSS 3 introduit de nombreuses fonctionnalités avancées pour réaliser des effets et animations très abouties sans passer nécessairement par Javascript.

Effets

```
h1 {
    text-shadow: 1px 1px 2px black;
}
```

Transition

```
<p class="fade">Mon texte</p>
.fade {
    opacity: 0.5;
    transition: opacity 0.5s linear;
}
.fade:hover {
    opacity: 1;
 }
```

Animation

```
p {
    animation-duration: 2s;
    animation-name: slidein;
}

@keyframes slidein {
    from {
        margin-left: 100%;
    }

    to {
        margin-left: 0%;
    }
}
```

Quelques ressources utiles

https://www.w3.org/TR/CSS/

https://www.w3schools.com/css/default.asp et notamment tous les exercices

https://developer.mozilla.org/fr/docs/Web/CSS

http://overapi.com/html

http://htmlcheatsheet.com/

https://www.sololearn.com/

Restons en contact.

DIGINAMIC
Lionel Cabon, Directeur
contact@diginamic.fr

N° Déclaration OF : 91 34 08867 34

Nantes, Paris, Montpellier

www.diginamic.fr