

# Nursery

A la découverte des métiers du digital



# Le parcours

Découvrir différents métiers et leurs composantes au travers des présentations et de la pratique personnelle

# Les ateliers

Ateliers [2](#) et 3 : Découverte des bases du Front-End et du Back-End via la pratique

# Atelier 2

Découverte de l'univers Front-End

---

## Découverte du métier de développeur via la pratique

### Sommaire :

Présentation du HTML

Premiers essais en HTML

La structure d'une page HTML

Créer une structure en HTML

Créer une maquette

Découverte du CSS et application

# Atelier 2

Découverte de l'univers Front-End

---

## Présentation du WEB

Il faut savoir que le Web fait partie d'Internet.

Internet est un grand ensemble qui comprend, entre autres : le Web, les emails, la messagerie instantanée, etc.

Les langages HTML et CSS sont à la base du fonctionnement de tous les sites web.

Le langage HTML a été inventé par un certain Tim Berners-Lee en 1991, le CSS et le JavaScript en 1996.

# Atelier 2

Découverte de l'univers Front-End

---

## Présentation du WEB

HTML, CSS et JavaScript sont les seuls langages interprétés par les navigateurs (Firefox, Safari, Edge, Opera, Chrome....).

# Atelier 2

## Découverte de l'univers Front-End

# Présentation du WEB

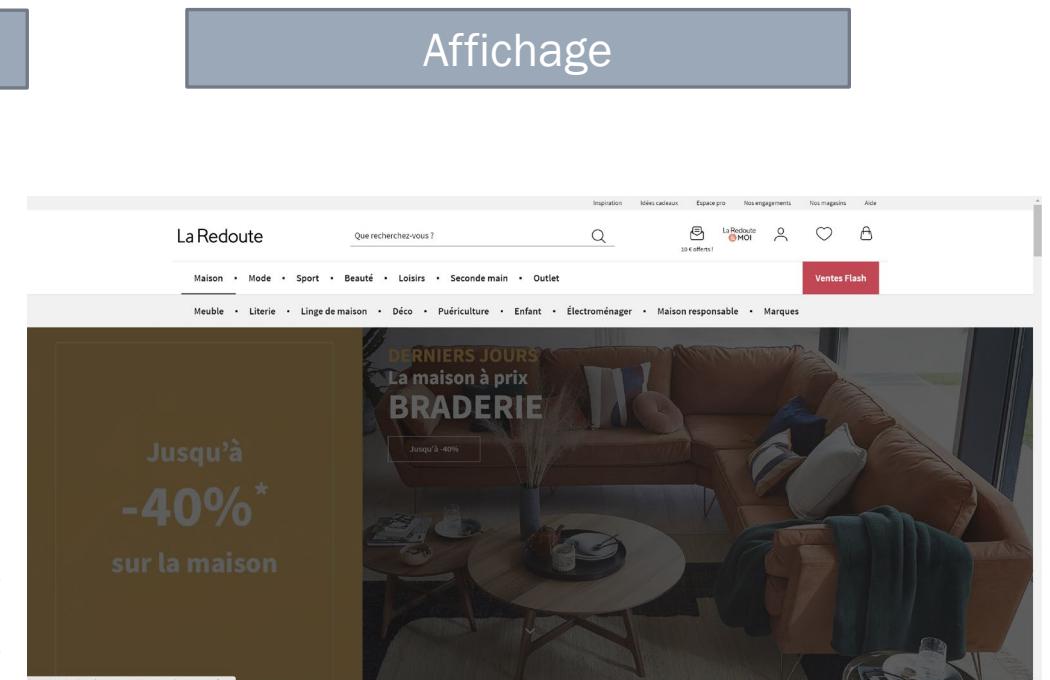
Code



# Interprétation



## Affichage



# Atelier 2

Découverte de l'univers Front-End

---

## Présentation du HTML

HyperText Markup Language (HTML) est le code utilisé pour structurer une page web et son contenu.

Par exemple, le contenu de votre page pourra être structuré en un ensemble de paragraphes, une liste à puces ou avec des images et des tableaux de données.

# Atelier 2

Découverte de l'univers Front-End

---

## Présentation du HTML

HTML n'est pas un langage de programmation. C'est un *langage de balises* qui définit la structure de votre contenu.

L'« hypertexte » désigne les liens qui relient les pages web entre elles, que ce soit au sein d'un même site web ou entre différents sites web. Les liens sont un aspect fondamental du Web. Ce sont eux qui forment cette « toile » (ce mot est traduit par web en anglais).

En téléchargeant du contenu sur l'Internet et en le reliant à des pages créées par d'autres personnes, vous devenez un participant actif du World Wide Web.

# Atelier 2

Découverte de l'univers Front-End

---

## Composition d'une balise

### 1. Balise paire :

<balise> contenu.... </balise>

Balise ouvrante

Balise fermante

< : chevron ouvrant  
> : chevron fermant

exemple: <p> Ceci est un paragraphe. </p>

### 2. Balise orpheline ou auto-fermante :

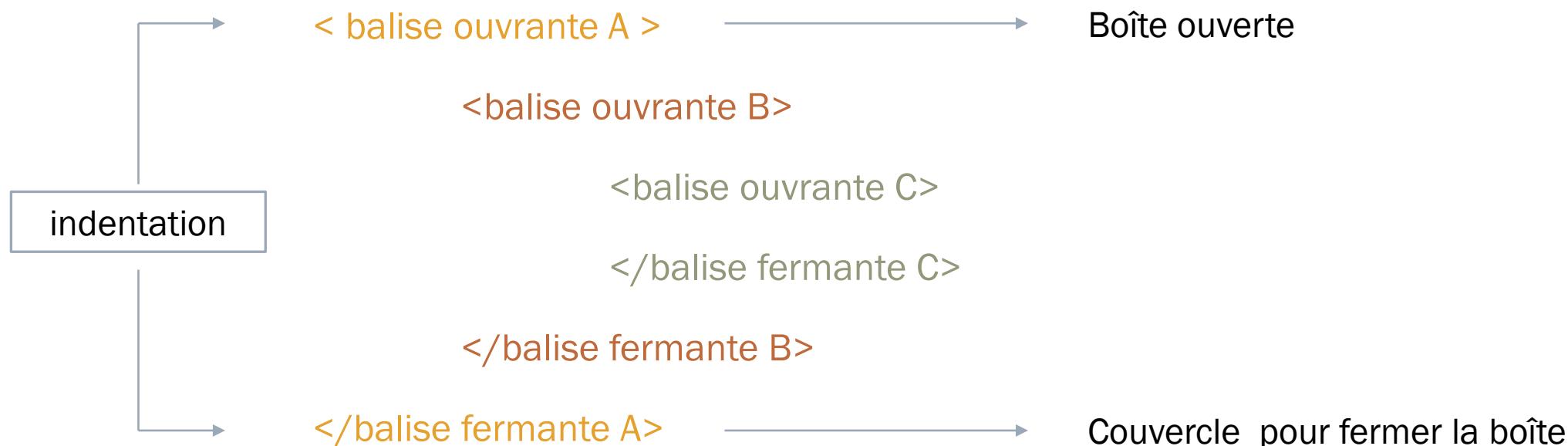
<balise> ou <balise />

exemple : <br /> (balise de retour à la ligne)

# Atelier 2

Découverte de l'univers Front-End

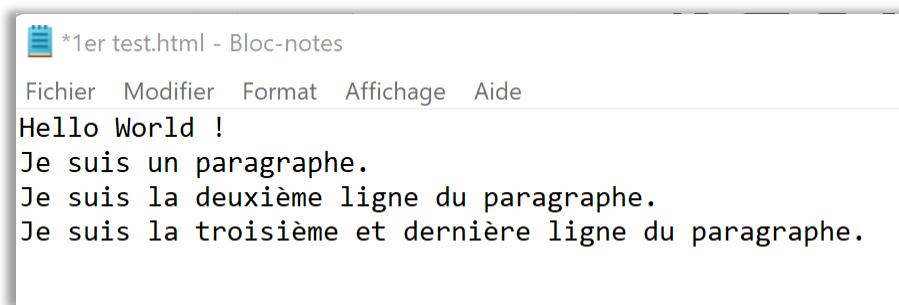
## Imbrication des balises



# Atelier 2

Découverte de l'univers Front-End

## 1<sup>er</sup> test : Créer un nouveau document texte avec le bloc-note



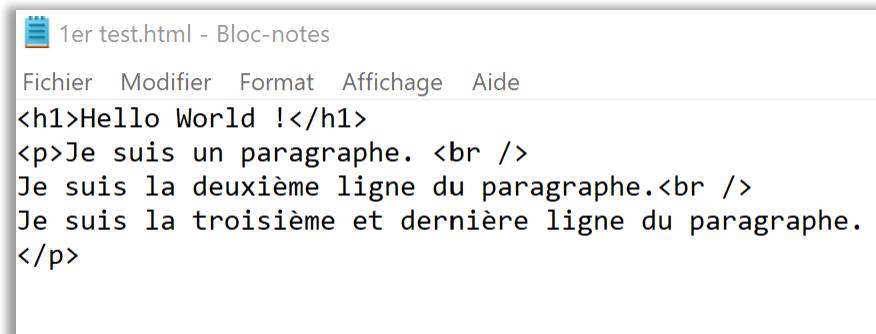
\*1er test.html - Bloc-notes

Fichier Modifier Format Affichage Aide

Hello World !  
Je suis un paragraphe.  
Je suis la deuxième ligne du paragraphe.  
Je suis la troisième et dernière ligne du paragraphe.



Hello World ! Je suis un paragraphe. Je suis la deuxième ligne du paragraphe. Je suis la troisième et dernière ligne du paragraphe.



1er test.html - Bloc-notes

Fichier Modifier Format Affichage Aide

<h1>Hello World !</h1>  
<p>Je suis un paragraphe. <br />  
Je suis la deuxième ligne du paragraphe.<br />  
Je suis la troisième et dernière ligne du paragraphe.  
</p>



# Hello World !

Je suis un paragraphe.  
Je suis la deuxième ligne du paragraphe.  
Je suis la troisième et dernière ligne du paragraphe.

# Atelier 2

Découverte de l'univers Front-End

---

## Les attributs

- ◆ Tous les éléments HTML peuvent avoir des **attributs**
- ◆ Les attributs fournissent **des informations supplémentaires** sur les éléments
- ◆ Les attributs sont toujours spécifiés dans **la balise de début**
- ◆ Les attributs se présentent généralement sous forme de paires nom/valeur telles que : **attribut="valeur"**

# Atelier 2

Découverte de l'univers Front-End

---

## Les attributs

Exemples

```
<a href="https://www.w3schools.com">Visiter Notre Site</a>  

```

**NB :** On peut avoir autant d'attributs qu'on veut pour une balise donnée

```

```

# Atelier 2

Découverte de l'univers Front-End

---

## Présentation du langage HTML

### NOTRE PREMIER CODE

Editeur de texte

Structure de base d'une page web

Notre première page web

Zoom sur le « Head »

Zoom sur le « Body »

# Atelier 2

Découverte de l'univers Front-End

---

## Choix de l'Editeur de code (IDE)

- **Visual Studio Code**
- Atome
- Brackets
- Sublime text
- ...



Visual Studio Code

# Atelier 2

Découverte de l'univers Front-End

---

## Structure d'une page HTML

Structure minimaliste d'une page web :

```
<!DOCTYPE html>
<html>

  <head>
    <title>Page Title</title>
  </head>

  <body>
    <h1>My First Heading</h1>
    <p>My first paragraph.</p>
  </body>

</html>
```

# Atelier 2

Découverte de l'univers Front-End

---

## Structure d'une page HTML

Structure minimaliste d'une page web :

**<!DOCTYPE html>** : document de type HTML

**<html> ... </html>** : racine du document. Indique où commence et fini notre page.

**<head> ... </head>** : l'en-tête contient des informations sur notre page.

Il peut contenir des liens vers des feuilles de style et du JavaScript.

**<body> ... </body>** : les informations dans ces balises seront affichées aux utilisateurs

```
<!DOCTYPE html>
<html>

  <head>
    <title>Page Title</title>
  </head>

  <body>
    <h1>My First Heading</h1>
    <p>My first paragraph.</p>
  </body>

</html>
```

# Structure complète

---

```
<!doctype html>
<html lang="fr">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.0/dist/css/bootstrap.min.css">
  <title>Hello, world!</title>
</head>
<body>
  <h1>Hello, world!</h1>

  <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.0/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```

# Atelier 2

Découverte de l'univers Front-End

---

## Présentation du langage HTML

### NOTRE PREMIER CODE

Notre première page web

Créer un 1<sup>er</sup> fichier html avec VSCode, coller la structure, enregistrer et l'ouvrir dans le navigateur

# La balise « head »

---

La balise "head", contient des informations au sujet de la page ou l'intégration de fichiers utiles à la page. Ce bloc n'est pas affiché sur l'écran.

**<title>Titre de ma page</title>**

Indique le titre de la page, généralement visible dans l'onglet de la page ou lorsque la page est ajoutée dans les favoris.

**<meta charset=utf-8 />**

Indique que le jeu de caractère de la page. UTF-8 qui est maintenant le jeu le plus utilisé sur internet permet d'écrire du texte dans quasiment toutes les langues.

**<meta name="keywords" content="restaurant pizza tiramisu">**

Meta pour les moteurs de recherche. Le site sera mieux référencé lors des recherches

**<meta name="description" content="Joli restaurant en bord de plage">**

Les balises méta fournissent des informations, soit pour le navigateur soit pour les moteurs de recherche.

# La balise « head »

---

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

**link** : que le navigateur doit chercher un document

**rel** : le document est une feuille de style (stylesheet)

**href** : chemin vers le fichier

```
<script type="text/javascript" src="chemin/vers/mon/fichier/javascript.js"></script>
```

**script** : permet d'ajouter un script dans la page

**type** : Par défaut le langage est le JavaScript mais il est conseillé de l'indiquer

**src** : le chemin vers le fichier.

Nous plaçons généralement la balise "script" dans le corps (`<body></body>`) de la page.

# La balise « body »

---

La balise "**body**" contient les éléments devant être affiché à l'utilisateur.

En reprenant le code ci-dessus, il est normal de ne rien voir.

HTML ne dit pas comment afficher les balises, il dit seulement : voilà comment est structurée ma page.

Deux grandes classes d'éléments :

## Block

Prennent toute la largeur disponible

Provoque un retour à la ligne

Peuvent contenir des éléments inline

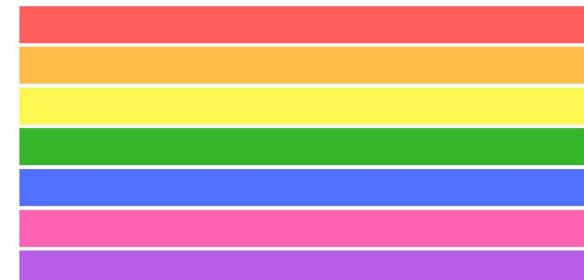
## Inline

Ne provoque pas de retour à la ligne

Ne peuvent pas contenir des éléments block

Peuvent contenir des éléments inline

BLOCK-LEVEL ELEMENTS:



INLINE ELEMENTS:



# Atelier 2

Découverte de l'univers Front-End

---

## LES BALISES

Les titres

Les paragraphes

Les liens

Les images

Les tableaux

Les listes

Les citations

Les commentaires

# Les titres

---

Les titres HTML sont définis avec les balises `<h1>` à `<h6>`.

`<h1>`définit le titre le plus important. `<h6>`définit le titre le moins important.

**Heading 1**

**Heading 2**

**Heading 3**

**Heading 4**

**Heading 5**

**Heading 6**

```
<h1>Heading 1</h1>
<h2>Heading 2</h2>
<h3>Heading 3</h3>
<h4>Heading 4</h4>
<h5>Heading 5</h5>
<h6>Heading 6</h6>
```

# Les paragraphes

---

La balise `<p>` définit un paragraphe.

`<p>C'est un paragraphe</p>`

# Les liens

---

Les liens permettent aux utilisateurs de cliquer dessus pour se déplacer de page en page.

La balise `<a>` en HTML définit un lien hypertexte. Il a la syntaxe suivante :

```
<a href="url"> texte du lien </a>
```

L'attribut le plus important de l'élément `<a>` est l'attribut `href`, qui indique la destination du lien.

Le *texte du lien* est la partie qui sera visible par l'utilisateur.

En cliquant sur le texte du lien, l'utilisateur sera renvoyé à l'adresse URL spécifiée.

## Exemple

```
<a href="https://www.w3schools.com/">Visit W3Schools.com!</a>
```

L'attribut `target` spécifie où ouvrir le document lié.

L'attribut `target` peut avoir l'une des valeurs suivantes :

- `_self` - Défaut. Ouvre le document dans la même fenêtre/onglet que celui sur lequel vous avez cliqué
- `_blank` - Ouvre le document dans une nouvelle fenêtre ou un nouvel onglet
- `_parent` - Ouvre le document dans le cadre parent
- `_top` - Ouvre le document dans tout le corps de la fenêtre

```
<a href="https://www.google.fr" target="_blank">lien dans un nouvel onglet</a>
```

# Les liens

---

## - URL absolues vs URL relatives

```
<h2>Absolute URLs</h2>
<a href="https://www.w3.org/">W3C</a>
<a href="https://www.google.com/">Google</a>
```

Liens vers d'autres sites

## <h2>Relative URLs</h2>

```
<a href="html_images.asp">HTML Images</a>
<a href="/css/default.asp">CSS Tutorial</a>
```

Liens vers d'autres pages du site

Pour utiliser une image comme lien, il suffit de mettre la balise `<img>` à l'intérieur de la balise `<a>`

```
<a href="default.asp">
    
</a>
```

## - Lien vers une adresse e-mail

```
<a href="mailto:contact@domaine.com">Envoi de mail</a>
```

# Les images

---

La balise <img> a deux attributs obligatoires :

- src : spécifie le chemin d'accès à l'image
- alt : spécifie un texte alternatif pour l'image

Exemple

```

```

**Taille de l'image - Largeur et hauteur :**

```

```

# Les Tableaux

---

```
<table>
  <tr>
    <th>Company</th>
    <th>Contact</th>
    <th>Country</th>
  </tr>
  <tr>
    <td>Alfreds Futterkiste</td>
    <td>Maria Anders</td>
    <td>Germany</td>
  </tr>
  <tr>
    <td>Centro comercial Moctezuma</td>
    <td>Francisco Chang</td>
    <td>Mexico</td>
  </tr>
</table>
```

table : début du tableau

tr : une ligne

td : une colonne

th : en-tête de colonne

# Les Listes

---

## Non ordonnée

```
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

## Ordonnée

```
<ol>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>
```

## Listes imbriquées

```
<ul>
  <li>Coffee</li>
  <li>Tea
    <ul>
      <li>Black tea</li>
      <li>Green tea</li>
    </ul>
  </li>
  <li>Milk</li>
</ul>
```

# Les Citations

---

Les balises de citations sont

<blockquote>, <q>, <abbr>, <address>, <cite> et les <bdo>

Exemple :

```
<blockquote cite="http://www.worldwildlife.org/who/index.html">
    For 50 years, WWF has been protecting the future of nature.
    The world's leading conservation organization,
    WWF works in 100 countries and is supported by
    1.2 million members in the United States and
    close to 5 million globally.
</blockquote>
<p>WWF's goal is to: <q>Build a future where people live in harmony with nature.</q></p>
<p>The <abbr title="World Health Organization">WHO</abbr> was founded in 1948
.</p>
<address>
    Written by John Doe.<br>
    Visit us at:<br>
    Example.com<br>
    Box 564, Disneyland<br>
    USA
</address>
<p><cite>The Scream</cite> by Edvard Munch. Painted in 1893.</p>
<bdo dir="rtl">This text will be written from right to left</bdo>
```

# Les Commentaires

---

*<!-- Write your comments here -->*

Les commentaires ne sont pas affichés à l'écran mais sont visibles dans le html.  
Ils servent à donner des renseignements utiles sur la structure et permettent de rapidement comprendre le code des uns et des autres.

! Ne pas mettre « -- » dans les commentaires

# Exercice pratique 1

Recréer cette page en HTML

Sera fourni un dossier avec :

- 1 fichier image du modèle
- 1 fichier image à insérer
- 1 dossier « css »
- 1 fichier test2.html pour débuter

## Mon site web

Lorem ipsum dolor sit amet. Qui eaque voluptas hic voluptas veritatis aut eveniet quia? Eos aperiam repudiandae et pariarur beatae aut assumenda pariarur est autem quod ut vite ipsum eum culpa molestiae eum modi impedit? Sed asperiores suscipit a quam error in temporibus dolore.

33 nesciunt velit et voluptas aliquid non voluptas voluptas et cumque tempora. Ea impedit unde rem repellat atque rem accusamus blanditiis. Aut libero laborum est explicabo aperiam nam error facere et cupiditate omnis sit. Quis excepturi est illum galismus ut voluptatum suscipit.



- Développement Front-End
- Développement Back-End
- Optimisation
- Référencement

Matière	Note	Note Max	Note Min
Front	16	18	14
Back	15	17	12
Design	14	19	10

# Les Formulaires

---

Un formulaire HTML est utilisé pour collecter les entrées de l'utilisateur. L'entrée utilisateur est le plus souvent envoyée à un serveur pour traitement.

La balise `<form>` est utilisée pour créer un formulaire HTML pour la saisie de l'utilisateur

L'attribut `action` définit l'action à effectuer lors de la soumission du formulaire.

L'attribut `method` spécifie la méthode HTTP à utiliser lors de la soumission des données du formulaire.

Les données de formulaire peuvent être envoyées sous forme de variables URL (avec `method="get"`) ou de post-transaction HTTP (avec `method="post"`).

La méthode HTTP par défaut lors de la soumission de données de formulaire est GET.

```
<form action="/action_page.php" method="get">  
</form>
```

# Les Formulaires

---

L' élément <form> en HTML peut contenir un ou plusieurs des éléments de formulaire suivants :

- [`<input>`](#)
- [`<label>`](#)
- [`<select>`](#)
- [`<textarea>`](#)
- [`<button>`](#)
- [`<fieldset>`](#)
- [`<legend>`](#)
- [`<datalist>`](#)
- [`<output>`](#)
- [`<option>`](#)
- [`<optgroup>`](#)

# Les Formulaires

---

Quelques-uns des différents types d'**input** :

`<input type="text">` Affiche un champ de saisie de texte sur une seule ligne

`<input type="number">` Affiche un champ qui n'accepte que des chiffres

`<input type="radio">` Affiche un bouton radio (pour sélectionner l'un des nombreux choix)

`<input type="checkbox">` Affiche une case à cocher (pour sélectionner zéro ou plusieurs choix)

`<input type="submit">` Affiche un bouton de soumission (pour soumettre le formulaire)

`<input type="button">` Affiche un bouton cliquable

# Les Formulaires

---

Liste complète d'**input** :

- <input type="button">
- <input type="checkbox">
- <input type="color">
- <input type="date">
- <input type="datetime-local">
- <input type="email">
- <input type="file">
- <input type="hidden">
- <input type="image">
- <input type="month">
- <input type="number">
- <input type="password">
- <input type="radio">
- <input type="range">
- <input type="reset">
- <input type="search">
- <input type="submit">
- <input type="tel">
- <input type="text">
- <input type="time">
- <input type="url">
- <input type="week">

# Les Formulaires

---

## L'élément <select>

```
<label for="cars">Choose a car:</label>
<select id="cars" name="cars">
  <option value="volvo">Volvo</option>
  <option value="saab">Saab</option>
  <option value="fiat">Fiat</option>
  <option value="audi">Audi</option>
</select>
```

Les éléments <option> définissent une option qui peut être sélectionnée.

Par défaut, le premier élément de la liste déroulante est sélectionné.

Pour définir une option présélectionnée, ajoutez l'attribut selected à l'option :

```
<option value="fiat" selected>Fiat</option>
```

# Les Formulaires

---

Utilisez l'attribut size pour spécifier le nombre de valeurs visibles

Utilisez l'attribut multiple pour permettre à l'utilisateur de sélectionner plusieurs valeurs

```
<label for="cars">Choose a car:</label>
<select id="cars" name="cars" size="4" multiple>
    <option value="volvo">Volvo</option>
    <option value="saab">Saab</option>
    <option value="fiat">Fiat</option>
    <option value="audi">Audi</option>

</select>
```

# Les Formulaires

---

## L'élément <textarea>

L' élément <textarea> définit un champ de saisie multiligne (une zone de texte) :

```
<textarea name="message" rows="10" cols="30">  
The cat was playing in the garden.  
</textarea>
```

L'attribut rows spécifie le nombre de lignes visibles dans une zone de texte.

L'attribut cols spécifie la largeur visible d'une zone de texte.

# Les Formulaires

---

Exemple :

```
<form action="/action_page.php" method="post">
    <label for="fname">First name:</label><br>
    <input type="text" id="fname" name="fname" value="John"><br>
    <input type="radio" id="html" name="fav_language" value="HTML">
    <label for="html">HTML</label><br>
    <input type="radio" id="css" name="fav_language" value="CSS">
    <label for="css">CSS</label><br>
    <input type="radio" id="javascript" name="fav_language" value="JavaScript">
    <label for="javascript">JavaScript</label>
    <input type="checkbox" id="vehicle1" name="vehicle1" value="Bike">
    <label for="vehicle1"> I have a bike</label><br>
    <input type="checkbox" id="vehicle2" name="vehicle2" value="Car">
    <label for="vehicle2"> I have a car</label><br>
    <input type="checkbox" id="vehicle3" name="vehicle3" value="Boat">
    <label for="vehicle3"> I have a boat</label>
    <input type="submit" value="Submit">
    <input type="reset" value="reset">
</form>
```

The screenshot shows a web page with a form. The form has a text input field labeled "First name:" containing "John". Below it are three radio buttons for language preferences: "HTML", "CSS", and "JavaScript", with "HTML" being selected. There are three checkbox options for vehicle ownership: "I have a bike", "I have a car", and "I have a boat", with "I have a bike" checked. At the bottom right of the form are two buttons: "Submit" and "reset".

# Exercice pratique 2

Recréer cette page en HTML

Sera fourni :

- 1 fichier image du modèle
- 1 fichier test3.html pour débuter

Balises utilisées :

```
<label for="">Titre du champ de formulaire</label>
<input type="text" placeholder="exemple de contenu" >
<input type="tel">
<input type="email">
<select> <option value="">item</option></select>
<input type="date">
<input type="number">
<input type="checkbox">
<textarea>,,</textarea>
<input type="radio">
<button type="submit" name="submit">Envoyer ma demande</button>
```

**Votre réservation d'hôtel**

**\* champs obligatoires**

Votre nom\*:

Votre prénom\*:

Votre numéro de téléphone\*:

Adresse e-mail\*:

Selectionner la ville :  

Date d'arrivée:  

Nombre de personnes\*:

Vos options:  
Climatisation  Mini Bar  Jacuzzi

Ajouter un commentaire:

Acceptation des Conditions Générales de Vente (CGV)\*  
 j'accepte  Je refuse

**Envoyer ma demande**

Selectionner la ville :  

Lille  
Paris    
Bordeaux  
Marseille  
Nantes  
Lyon  
La Rochelle  
Poitiers  
Toulouse

Personnes  
Mini  
Des Condi...  
Je refu...  
Demande



# Les Ancres (ou signets)

---

Une ancre est un lien qui renvoie vers un élément de la page.

Il faut donner un identifiant unique à l'élément cible (un id) que l'on place en tant qu'attribut dans la balise de cet élément cible.

Exemple :

Lien -> `<a href="#p1">Paragraphe 1</a>` Pour sélectionner un id il faut ajouter # devant la valeur

Cible -> `<p id="p1" > Lorem ipsum... </p>` La valeur d'un id ne doit pas contenir d'espace  
On peut ajouter un id à n'importe quelle balise html

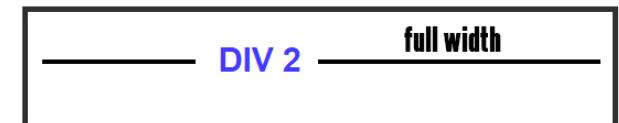
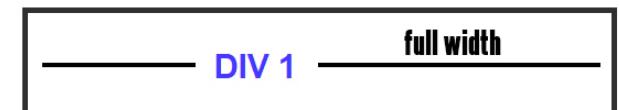
# Les balises "div" et "span"

---

Balises utilisées lorsqu'il n'y a pas d'autre balise qui correspond au besoin.

## **<div> </div>**

- élément contenant d'autres éléments, servant à définir un bloc
- servira essentiellement à faire du positionnement de bloc



## **<span> </span>**

- élément contenant d'autres éléments, inline
- servira essentiellement à regrouper des caractéristiques globales pour des éléments textuels

# HTML 5 : les nouveautés

---

Nouvelles balises (balises sémantiques ou d'organisation) : section, video, article, audio, aside, header, footer ...

De nouvelles balises audio et vidéo qui ne nécessitent plus l'appel à des plug-ins dédiés.

Le dessin 2D et bientôt 3D par la nouvelle balise <canvas>.

Une profusion de formulaires novateurs comme les curseurs ou les calendriers et la prise en charge de façon native par les navigateurs de la validation des données.

Les nouveaux éléments de formulaire : datetime, date, time, month, number ...  
et bien d'autres fonctionnalités...

Liste complète des ajouts de HTML 5 :

<https://fr.wikipedia.org/wiki/HTML5>

<https://www.aurone.com/les-nouveautes-apportees-par-html-5>

# Balises sémantiques

Exemple de disposition

## **<header>**

Section d'introduction d'un article, d'une autre section ou du document entier (en-tête de page).

## **<nav>**

Section possédant les liens de navigation principaux (au sein du document ou vers d'autres pages)

## **<section>**

Section générique regroupant un même sujet, une même fonctionnalité, de préférence avec un en-tête, ou bien une section d'application web

## **<article>**

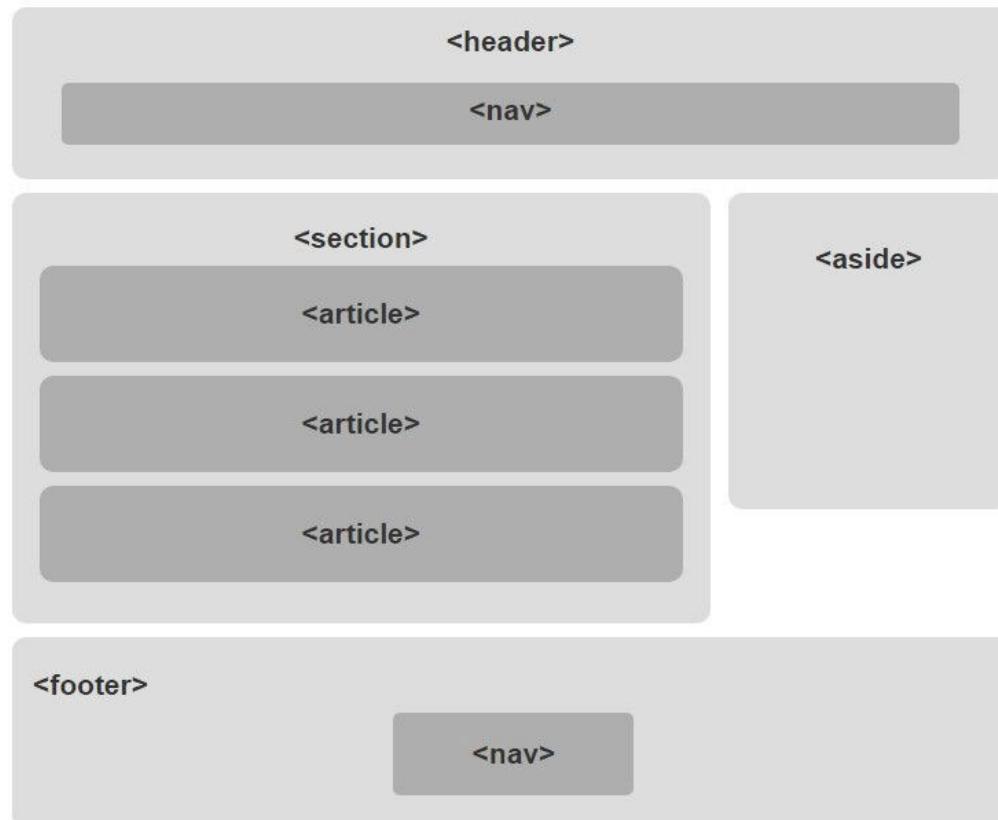
Section de contenu indépendante, pouvant être extraite individuellement du document ou syndiquée (flux RSS ou équivalent), sans pénaliser sa compréhension

## **<aside>**

Section dont le contenu est un complément par rapport à ce qui l'entoure, qui n'est pas forcément en lien direct avec le contenu mais qui peut apporter des informations supplémentaires.

## **<footer>**

Section de conclusion d'une section ou d'un article, voire du document entier (pied de page).



# Balises sémantiques

Autre disposition possible

## `<header>`

Section d'introduction d'un article, d'une autre section ou du document entier (en-tête de page).

## `<nav>`

Section possédant les liens de navigation principaux (au sein du document ou vers d'autres pages)

## `<section>`

Section générique regroupant un même sujet, une même fonctionnalité, de préférence avec un en-tête, ou bien section d'application web

## `<article>`

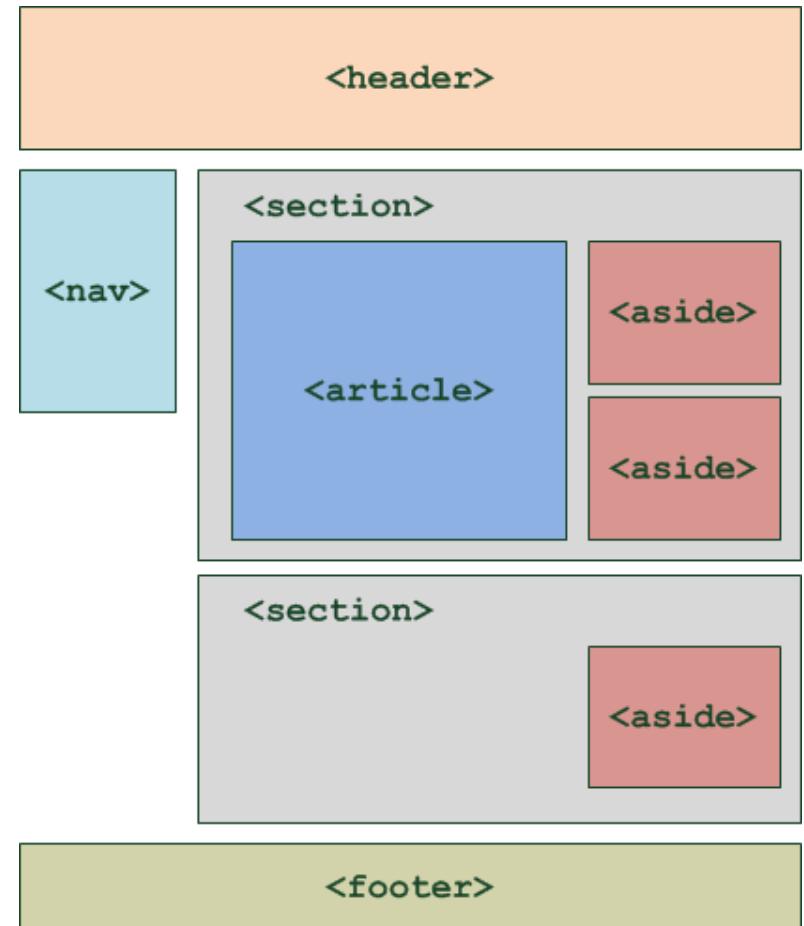
Section de contenu indépendante, pouvant être extraite individuellement du document ou syndiquée (flux RSS ou équivalent), sans pénaliser sa compréhension

## `<aside>`

Section dont le contenu est un complément par rapport à ce qui l'entoure, qui n'est pas forcément en lien direct avec le contenu mais qui peut apporter des informations supplémentaires.

## `<footer>`

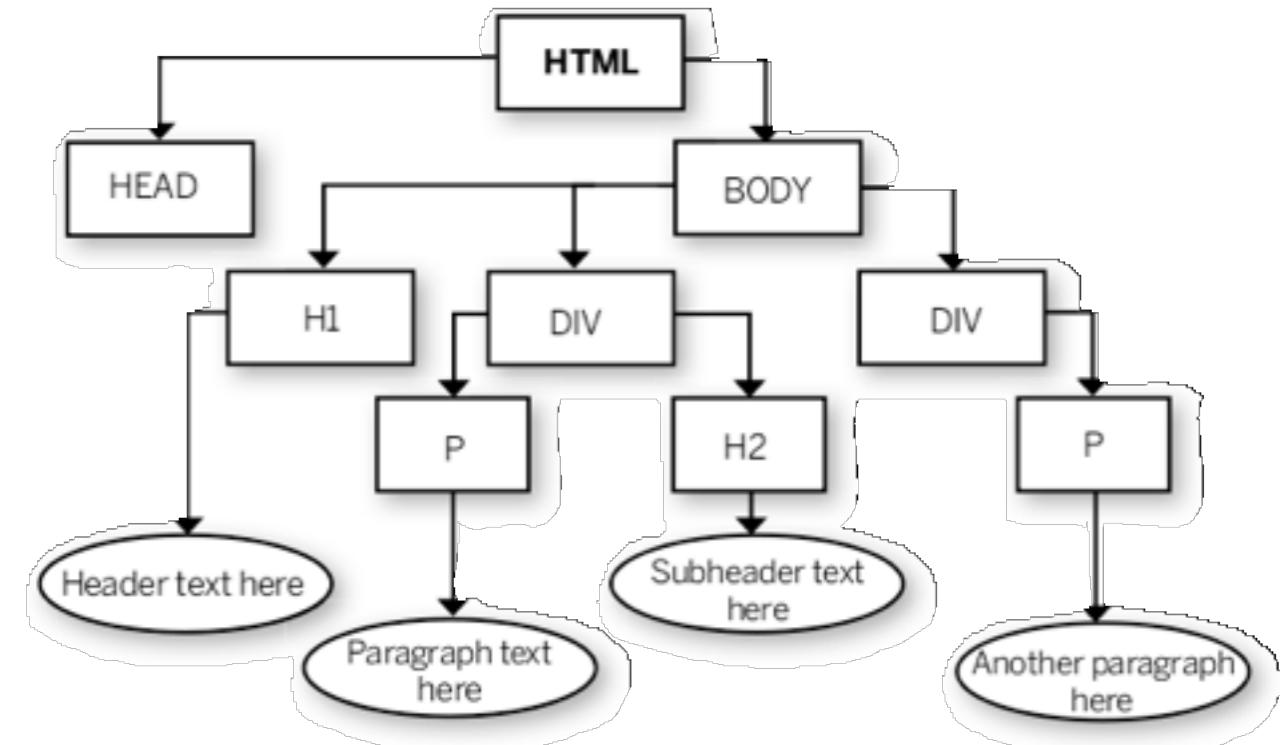
Section de conclusion d'une section ou d'un article, voire du document entier (pied de page).



# HTML génère un arbre

---

- ▶ Un arbre contient une racine, des parents, des frères, des enfants
- ▶ comme un arbre généalogique
- ▶ Appellation technique : « DOM »  
(Document Object Model)



# Exercice pratique 3

Créer votre CV en HTML (structure uniquement)

Étape 1 : Créez une page HTML avec une structure de base.

Étape 2 : Ajoutez votre nom et votre prénom en titre principal dans un Header.

Étape 3 : Ajoutez une image d'avatar dans le Header

Étape 4 : Dans une balise Div Ajoutez 3 Sections avec un titre secondaire (mon expérience, mes compétences, ma formation).

Etape 5 : Suivre les indications sur l'aperçu fourni

**Moi-même**



**Mon expérience**

**Titre 1**  
Entreprise  
*de 2000 à 2000*  
Lorem ipsum dolor sit amet. Qui eaque voluptas hic voluptas veritatis aut eveniet quia

**Titre 2**  
Entreprise  
*de 2000 à 2000*  
Lorem ipsum dolor sit amet. Qui eaque voluptas hic voluptas veritatis aut eveniet quia

**Titre 3**  
Entreprise  
*de 2000 à 2000*  
Lorem ipsum dolor sit amet. Qui eaque voluptas hic voluptas veritatis aut eveniet quia

**Mes compétences**

- HTML5 et CSS3
- Recherches [Google](#) niveau avancé
- Lorum
- Ipsum



**Ma formation**

**Titre 1**  
Ecole  
*de 2000 à 2000*  
Lorem ipsum dolor sit amet. Qui eaque voluptas hic voluptas veritatis aut eveniet quia

**Titre 2**  
Ecole  
*de 2000 à 2000*  
Lorem ipsum dolor sit amet. Qui eaque voluptas hic voluptas veritatis aut eveniet quia

**Titre 3**  
Ecole  
*de 2000 à 2000*  
Lorem ipsum dolor sit amet. Qui eaque voluptas hic voluptas veritatis aut eveniet quia

# Atelier 2 : Web Design

Découverte de l'univers Front-End

## CRÉER UNE MAQUETTE

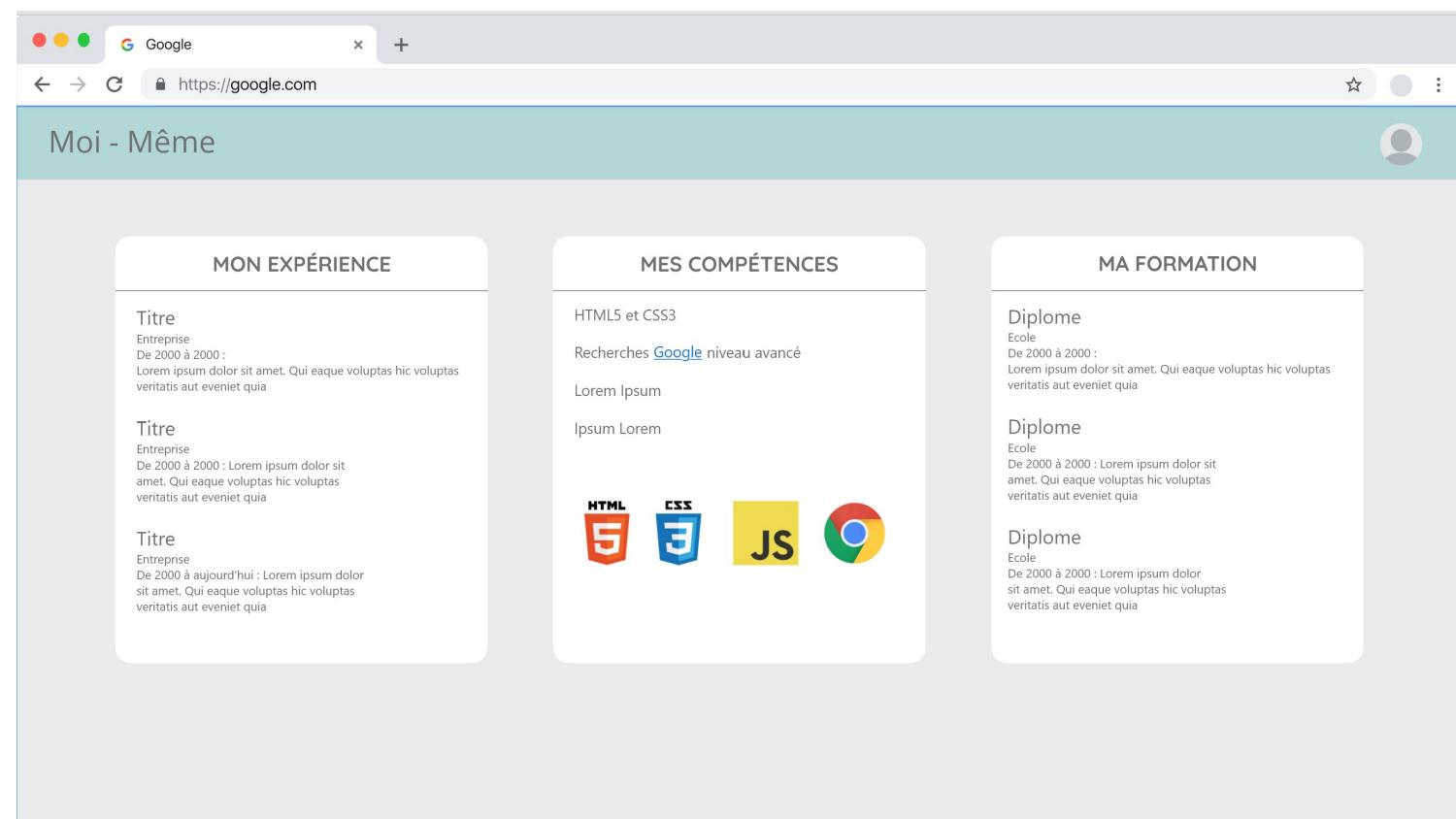
Télécharger et Installer Adobe Xd ->

<https://www.adobe.com/fr/creativecloud/desktop-app.html>

Créer un nouveau projet pour le web  
(FHD par ex.)

Utiliser l'outil pour mettre en forme  
un cv simple en suivant ce modèle :

Imaginer ensuite la version sur  
téléphone. (nouveau plan de travail)



# Atelier 2 : CSS

Découverte de l'univers Front-End

---

## Sommaire

- 1- Le modèle des boîtes
- 2- Ajouter des styles
  - 2.1- A l'intérieur des balises head
  - 2.2- Feuille de style externe
- 3- Syntaxe css
- 4- Le sélecteur d'identifiant CSS
- 5- Le sélecteur de classe CSS
- 6- Le sélecteur de regroupement CSS
- 7- les couleurs
- 8- Couleur d'arrière plan
- 9- Image d'arrière-plan
- 10- Répétition d'arrière-plan
- 11- Position d'arrière-plan
- 12- Défilement d'arrière-plan
- 13- Les bordures
- 14- Bordure arrondie
- 15- Les marges
- 16- Padding
- 17- La hauteur
- 18- La largeur
- 19- Contour
- 20- Alignement du texte
  - 20.1- Horizontal
  - 20.2- Vertical
- 21- Décoration de texte

- 22- Transformation de texte
- 23- Espacement du texte
  - 23.1- L'espacement des lettres (dans le sens horizontal)
  - 23.2- Hauteur de la ligne
  - 23.3- Espacement des mots
  - 23.4- Ombre de texte
  - 23.5- Police de texte
  - 23.6- Taille de la police
  - 23.7- Utiliser les polices Google
- 24- Les liens
- 25- Les listes
- 26- Le display
- 27- Positionnement
- 28- Overflow
- 29- Affichage flottant
- 30- Arrêter l'affichage flottant
- 31- les sélecteurs CSS (avancé)
  - 31.1- Sélecteur de descendants (espace)
  - 31.2- Sélecteur d'enfant (>)
  - 31.3- Sélecteur de frères adjacents (+)
  - 31.4- Sélecteur de frères et sœurs général (~)
  - 31.5- les pseudo-classes
    - 31.5.1- Syntaxe
    - 31.5.2- La pseudo-classe :first-child
  - 31.6- Pseudo-éléments
    - 31.6.1- Syntaxe
    - 31.6.2- La ::first-line
- 31.6.3- La ::première lettre
- 31.6.4- Le ::before (Pseudo-element)
- 31.6.5- Le ::after (Pseudo-élément)
- 31.6.6- Le ::marker Pseudo-élément (sans balise avant)
- 31.6.7- Le ::selection Pseudo-élément
- 32- L'opacité
- 33- Les media queries
  - 33.1- Syntaxe de la media query
- 34- Flexbox
  - 34.1- Modifier la direction
  - 34.2- Gérer le retour à la ligne
  - 34.3- Les alignements
    - 34.3.1- Aligner sur l'axe principal
    - 34.3.2- Aligner sur l'axe secondaire
  - 34.4- Modifier l'ordre des éléments
  - 34.5- Faire grossir ou maigrir les éléments
- 35- Atelier
  - 35.1- Reproduire son CV en version PC
  - 35.2- Faire les ajustements nécessaires pour l'adapter au mobile
- 36- Découverte de Bootstrap (à découvrir par soi-même)

# Atelier 2 : CSS

Découverte de l'univers Front-End

---

Séparation de la structure logique (HTML) et de la présentation (CSS)

Présentation suivant une feuille de style (style sheet) qui traite les éléments de contenu en éléments de présentation

CSS = Cascading Style Sheets (feuilles de style en cascade)

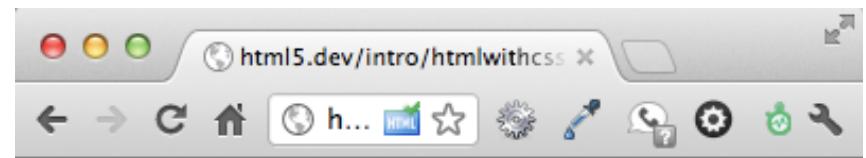
En cascade :

- on peut utiliser plusieurs feuilles de styles
- il y a un degré d'importance pour chaque feuille de style

# Atelier 2 : CSS

Découverte de l'univers Front-End

```
h1{  
    color: red;  
}  
  
h2{  
    color: blue;  
    font-style: italic;  
}  
  
p{  
    color: white; background-color: black;  
}
```



## whichElement?

*Trying to answer that age old question:*

*Should that be a div, a span, or something else?*

- Home
- Contribute
- About

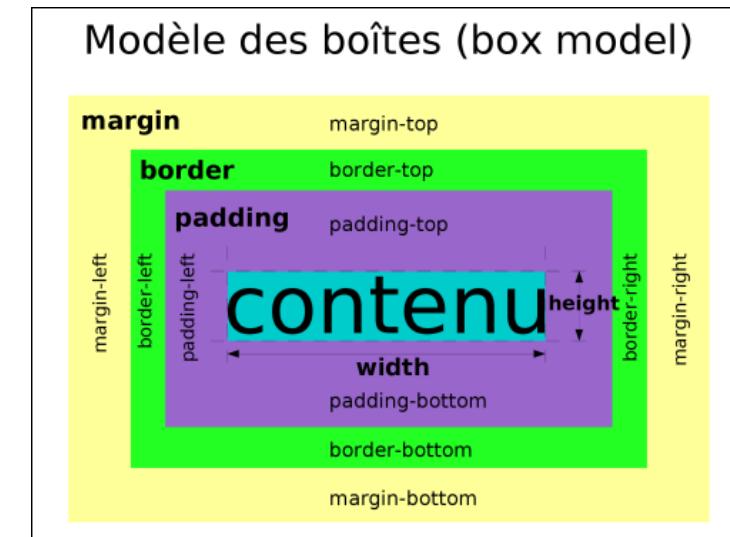
One of the main challenges we see in building semantic content is picking what tag to use when. This site seeks to help with that. Now, before we get all judgy and preachy let me get a few tenants out there:

# Le modèle des boîtes

Découverte de l'univers Front-End

Le modèle des boîtes CSS se base sur le principe que tous les éléments utilisés en HTML sont mis en forme sur la base de zones rectangulaires. Ainsi chaque élément lorsqu'il est rendu dans le navigateur aura les contours suivants :

- Le contenu de l'élément dont la taille est définie par sa largeur width et sa hauteur height
- Son espacement intérieur : propriété padding
- Sa marge : propriété margin
- Sa bordure : propriété border



# Ajouter des styles

Découverte de l'univers Front-End

---

**A l'intérieur des balises head :**

```
<head>
<style type="text/css">
    #boite1 {
        position: relative;
        margin: auto;
        margin-left: 20px;
        height: auto;
        width: 643px;
        float: left;
        background-image: url(images/catalog.png);
    }
</style>
</head>
```

**Feuille de style externe (recommandé) :**

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

# Syntaxe CSS

Découverte de l'univers Front-End

---

```
Selecteur {  
    Attribut-css : valeur ;  
}
```

**Exemple :**

```
p {  
    text-align: center;  
    color: red;  
}
```

# Le sélecteur d'identifiant CSS

---

Dans le html :

```
<p id="para1"> para1 </p>
```

Dans le css :

```
#para1 {  
    text-align: center;  
}
```

# Le sélecteur de classe CSS

---

Dans le html :

```
<p class=" center "> para1 </p>
```

Dans le css :

```
.center {  
    text-align: center;  
    color: red;  
}
```

**L'attribut class, peut supporter plusieurs valeurs**

```
<p class="center large">  
This paragraph refers to two classes.  
</p>
```

```
.center {  
    text-align: center;  
}  
.large {  
    width: 100%;  
}
```

# Le sélecteur de regroupement CSS

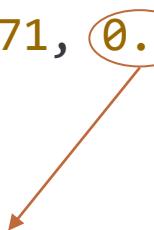
---

```
h1, h2, p {  
    text-align: center;  
    color: red;  
}
```

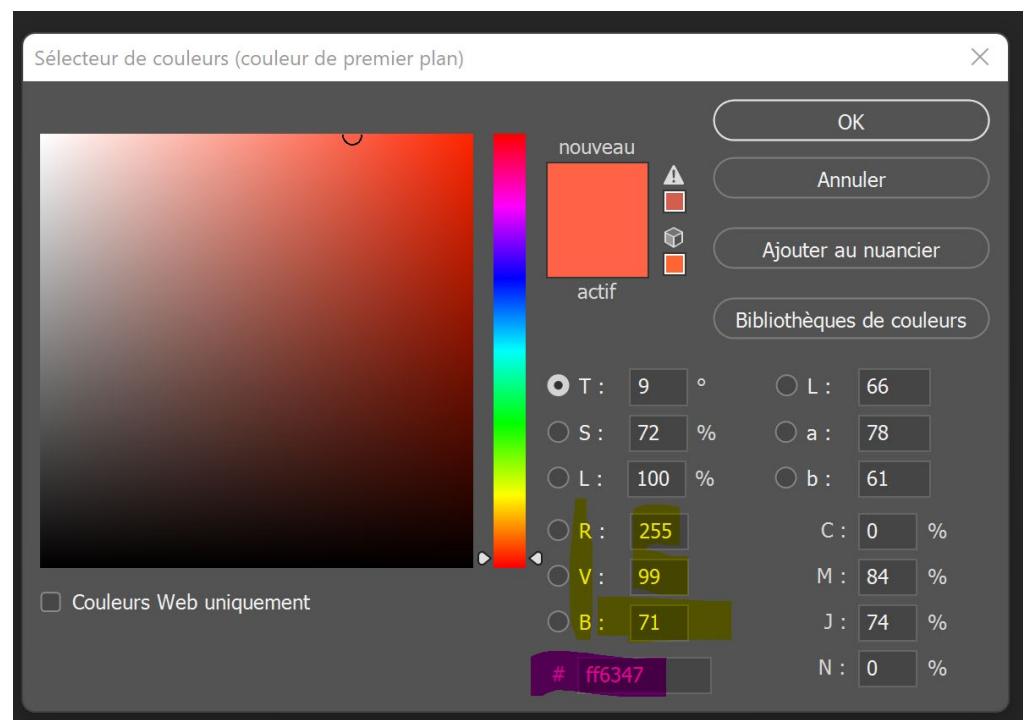
# Les couleurs

---

```
p {  
    color: rgb(255, 99, 71);  
  
    color: #ff6347;  
  
    color: rgba(255, 99, 71, 0.5);  
}
```



Couche Alpha = transparence  
De 0 (objet invisible) à 1 (100% opaque),  
Exemple 0.5 = 50% d'opacité



# Couleur d'arrière plan

---

```
body {  
background-color: lightblue;  
background: rgba(0, 128, 0, 0.3);  
}
```

# Image d'arrière-plan

---

```
body {  
background-image: url("paper.gif");  
}
```

# Répétition d'arrière-plan

---

```
body {  
background-image: url("gradient_bg.png");  
background-repeat: repeat-x;  
background-repeat: repeat-y;  
background-repeat: no-repeat;  
}
```

# Position d'arrière-plan

---

```
body {  
background-image: url("img_tree.png");  
background-repeat: no-repeat;  
background-position: right top;  
}
```

# Défilement d'arrière-plan

---

```
body {  
background-attachment: fixed;  
}
```

# Les bordures

---

```
div {  
    border: 2px solid gray;  
    border: 5px dotted #ff0000;  
}
```

`2px` : taille de la bordure

`solid` : style (normal, pointillés, tirets, double...)

`gray` : couleur

# Les bordures arrondies

---

```
div {  
  
    border-radius: 5px; /* Les coins seront tous arrondis */  
    border-radius: 5px 10px 15px 20px; /* valeurs différentes pour chaque coin en partant du haut gauche puis  
                                         dans le sens des aiguilles d'une montre */  
}
```

**NB :** pour créer un cercle, on peut mettre :

```
div {border-radius: 50%;}
```

# Les marges

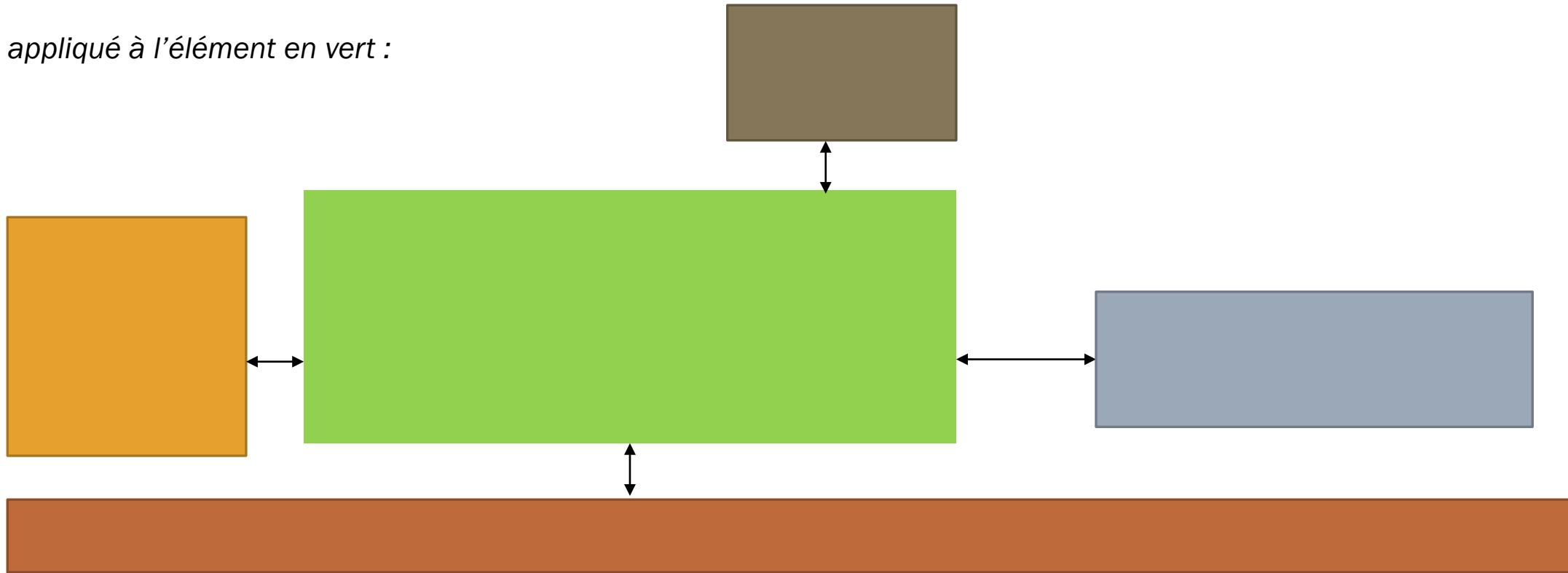
---

```
div {  
  
margin: 5px; /* margin sur tous les cotés */  
  
margin: 3px 6px; /* 3px du haut et du bas et 6px de la droite et de  
la gauche */  
  
margin: 4px 1px 3px 2px; /* respectivement haut, droite, bas et gauche */  
  
margin-bottom: 1px; /* 1px de margin en bas seulement */  
}
```

# Les marges

---

*margin appliqué à l'élément en vert :*



# Le padding

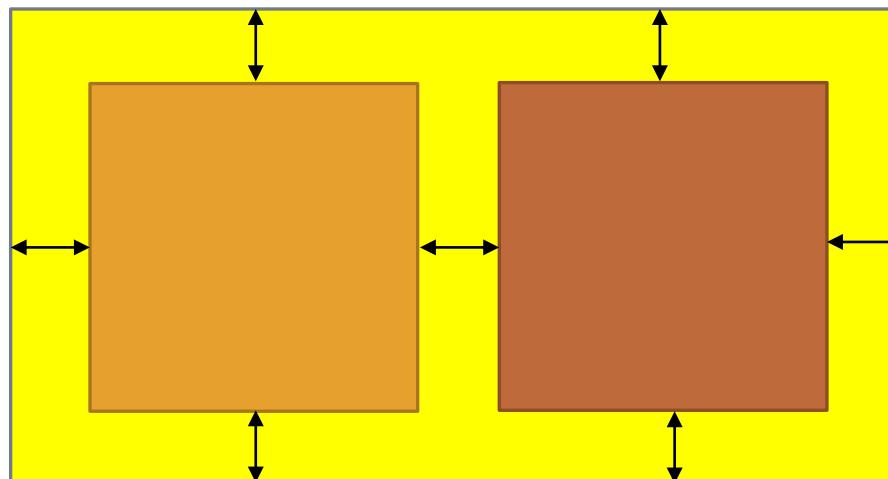
---

Les propriétés CSS padding, sont utilisées pour générer de l'espace autour du contenu d'un élément, à l'intérieur de toute bordure définie.

Même syntaxe que les margins :

- padding-top
- padding-right
- padding-bottom
- padding-left
- padding

*Padding appliqué à l'élément en jaune, ses enfants ne sont donc pas collés aux bords*



# La hauteur

---

```
div {  
    height: 200px;  
    max-height: 400px;  
    min-height: 50%;  
}
```

# La largeur

---

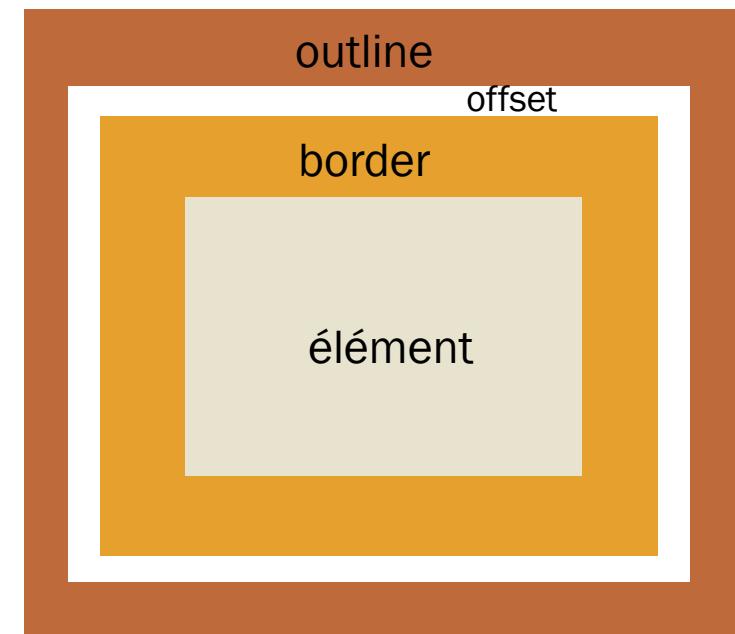
```
div {  
    width: 100%;  
    max-width: 400px;  
    min-width: 100px;  
}
```

# Le contour

---

Un contour est une ligne qui est tracée autour des éléments, EN DEHORS des bordures, pour faire démarquer l'élément.

```
p {  
    outline: red solid 10px;  
  
    outline-offset: 35px; /* décalage du contour */  
}
```



# Décoration du texte

---

```
a {  
    text-decoration: overline;  
  
    text-decoration: line-through;  
  
    text-decoration: underline;  
  
    text-decoration: none; /* utile pour supprimer le style par défaut des liens par  
exemple */ un lien  
}
```

# Espacement du texte

---

```
p {  
text-indent: 50px;  
/* création de padding sur  
la 1ere lettre du texte */  
}
```

## L'espacement des lettres (dans le sens horizontal)

```
p {  
letter-spacing: 3px;  
letter-spacing: -  
3px; /* chevauchement des lettres */  
}
```

## Hauteur de la ligne

```
p {  
line-height: 0.8;  
}
```

## Espacement des mots

```
p {  
word-spacing: 10px;  
}
```

## Ombre de texte

```
p {  
text-shadow: 2px 2px 5px red;  
}
```

## Taille de la police

```
p {  
font-size: 40px;  
font-size: 2.5em;  
font-size: 2.5em; /* 40px/16=2.5em */  
}
```

# Les polices de caractère (fonts)

---

## Police de texte

```
p {  
  
    font-family: Arial, Helvetica, sans-serif; /* choix 1, choix 2, catégorie par défaut */  
    font-style: italic;  
    font-style: oblique;  
    font-weight: bold;  
}
```

## Utiliser les polices Google

Dans le Head du HTML : <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Sofia">  
En CSS : h2 {font-family: 'Sofia', cursive; }

-> <https://fonts.google.com/>

# Les polices de caractère (fonts)

---

**Utiliser une police téléchargée sur un site :**

Placer d'abord les fichiers de police dans un dossier 'fonts' dans le dossier de votre projet.

Puis, en css :

En haut du fichier on met :

```
@font-face {  
    font-family: "Open Sans";  
    src: url("/fonts/ OpenSans-Regular.ttf") format("ttf");  
}
```

Puis on attribue la police à un ou plusieurs éléments :

```
body {  
    font-family: 'Open Sans', sans-serif;          /* en attribuant au body cette Font elle sera utilisée par défaut sur tous les textes. */  
}
```

# Les liens

---

```
/* unvisited link */  
a:link {  
    color: red;  
}
```

```
/* visited link */  
a:visited {  
    color: green;  
}
```

```
/* mouse over link */  
a:hover {  
    color: hotpink;  
}
```

```
/* selected link */  
a:active {  
    color: blue;  
}
```

# Les listes

---

```
li {  
  
list-style-type: circle; /* type de puce, pas de puce : none; */  
  
list-style-image: url('sqpurple.gif'); /* utiliser une image comme puce */  
  
list-style-position: outside;  
}  
  
outside :  
inside :
```

NASA Notable Missions

- Apollo
- Hubble
- Chandra
- Cassini-Huygens
- Spitzer

NASA Notable Missions

- Apollo
- Hubble
- Chandra
- Cassini-Huygens
- Spitzer

# Le display

---

```
div {  
    display: inline;  
    display: block;  
    visibility: hidden;  
    display: none;  
}
```

**NB :** Masquer un élément peut être fait en définissant la propriété display sur none. L'élément sera masqué et la page s'affichera comme si l'élément n'existe pas.

Par contre avec visibility :hidden, l'élément occupera toujours le même espace qu'auparavant. L'élément sera masqué, mais affectera toujours la mise en page.

# Le positionnement

---

```
div {  
  position: static; /* valeur par défaut */  
  position: relative;  
  position: fixed;  
  position: absolute;  
  position: sticky;  
  
  bottom: 0;  
  right: 0;  
}
```

Position : absolute  
top : 0; left : 0;

Position : relative

# Overflow

---

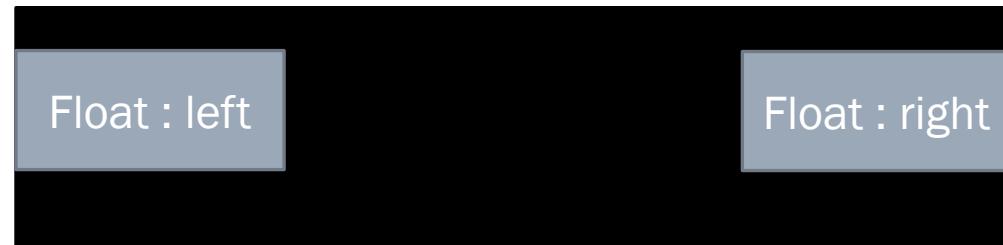
```
div {  
  overflow: visible;  
  overflow: hidden;  
  overflow: scroll;  
  overflow: auto;  
  overflow-x: hidden; /* Hide horizontal scrollbar */  
  overflow-y: scroll; /* Add vertical scrollbar */  
}
```

# Affichage flottant

---

La propriété CSS float spécifie comment un élément doit flotter.

```
div {  
    float: right;  
  
    float: left;  
  
    float: none;  
}
```



**Arrêter l'affichage flottant :** `div { clear: both; }`

# Pseudo élément

---

Un pseudo-élément CSS est utilisé pour styliser des parties spécifiées d'un élément.

Par exemple, il peut être utilisé pour :

1. Styliser la première lettre, ou ligne, d'un élément
2. Insérer du contenu avant ou après le contenu d'un élément

```
selector::pseudo-element {  
    property: value;  
}
```

Exemple : p::first-line {color: red; text-transform: uppercase;}

-> La première ligne de chaque paragraphe sera en rouge et en majuscules.

Liste complète des pseudo-éléments : [https://www.w3schools.com/css/css\\_pseudo\\_elements.asp](https://www.w3schools.com/css/css_pseudo_elements.asp)

# L'opacité

---

```
img {  
    opacity: 0.5; /* opacité de 50% */  
}  
  
img:hover {  
    opacity: 1.0; /* opacité de 100% */  
}
```

# Les media queries

---

Les media queries, introduites dans CSS2, ont permis de définir différentes règles de style pour différents types de médias.

Exemples : vous pouvez avoir un ensemble de règles de style pour les écrans d'ordinateur, un pour les imprimantes, un pour les appareils portables, un pour les appareils de type télévision, et ainsi de suite.

# Les media queries

---

Les requêtes multimédias dans CSS3 ont étendu l'idée des types de médias CSS2 : au lieu de rechercher un type d'appareil, elles examinent la capacité de l'appareil.

Les requêtes multimédias peuvent être utilisées pour vérifier de nombreuses choses, telles que:

- largeur et hauteur de la fenêtre
- largeur et hauteur de l'appareil
- orientation (la tablette/le téléphone est-il en mode paysage ou portrait ?)
- résolution

# Les media queries

---

## Syntaxe de la media query

Dans un fichier css

```
@media not|only mediatype and ( expressions) {  
    CSS-Code;  
}
```

Dans un fichier HTML

```
<link rel="stylesheet" media=" mediotype and|not|only ( expressions)"  
      href=" print.css">
```

# Les media queries

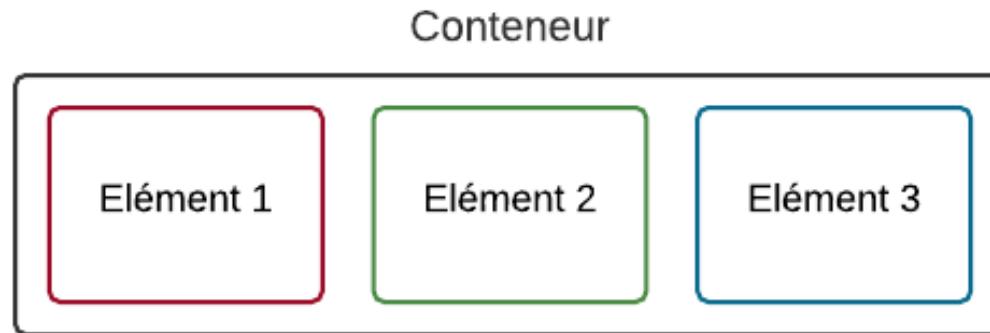
---

```
@media screen and (min-width: 480px) {  
    body {  
        background-color: lightgreen;  
    }  
}  
  
@media screen and (max-width: 900px) and (min-width: 600px) {  
    div.example {  
        font-size: 50px;  
        padding: 50px;  
        border: 8px solid black;  
        background: yellow;  
    }  
}
```

# Flexbox : agencement du contenu

---

Le principe de la mise en page avec Flexbox est simple : vous définissez un conteneur, et à l'intérieur vous y placez plusieurs éléments :



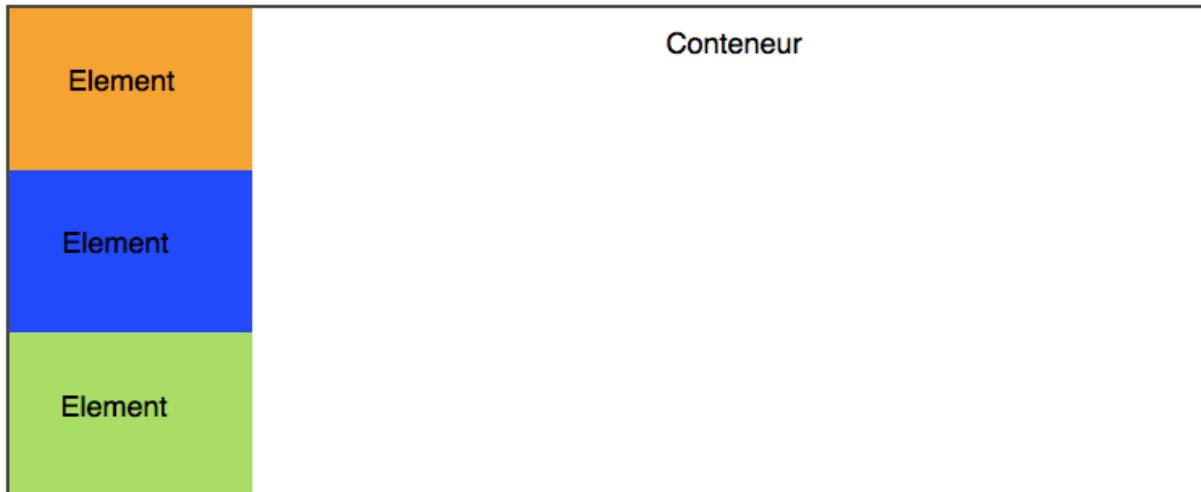
Le conteneur est une balise HTML, et les éléments sont d'autres balises HTML à l'intérieur :

```
<div id="conteneur">
    <div class="element 1">Element </div>
    <div class="element 2">Element </div>
    <div class="element 3">Element </div>
</div>
```

# Flexbox : agencement du contenu

---

Les éléments ici sont de type « block » (div) ils vont donc se placer par défaut les uns en dessous des autres :

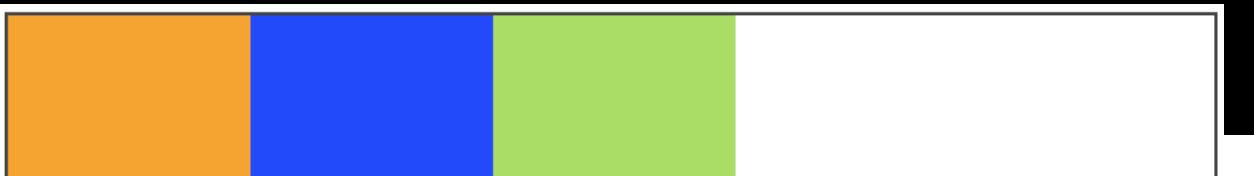


# Flexbox : agencement du contenu

---

En ajoutant à notre conteneur la propriété « flex », la disposition va changer :

```
#conteneur
{
  display: flex;
}
```



# Flexbox : colonne ou ligne

---

## Modifier 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.

# Flexbox : colonne ou ligne

---

Exemple :

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

Les éléments sont disposés en colonne mais l'ordre a été inversé sans modifier le fichier html :



# Flexbox : retour à la ligne

---

## Gérer le retour à la ligne

Flexbox nous permet d'agencer ces éléments dans le sens que l'on veut.

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 :

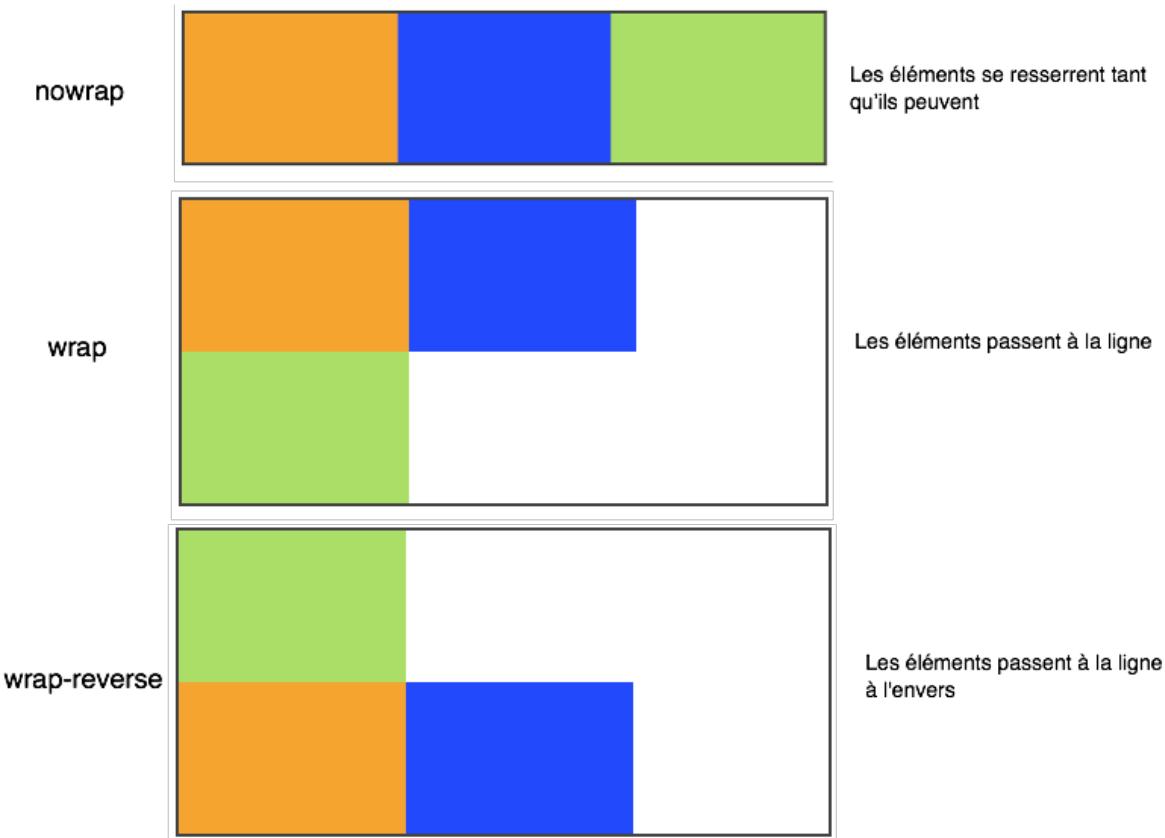
- nowrap : pas de retour à la ligne (par défaut) ;
- wrap : les éléments vont à la ligne lorsqu'il n'y a plus la place ;
- wrap-reverse : les éléments vont à la ligne, lorsqu'il n'y a plus la place, en sens inverse.

# Flexbox : retour à la ligne

---

```
#conteneur
{
  display: flex;
  flex-wrap: wrap;
```

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

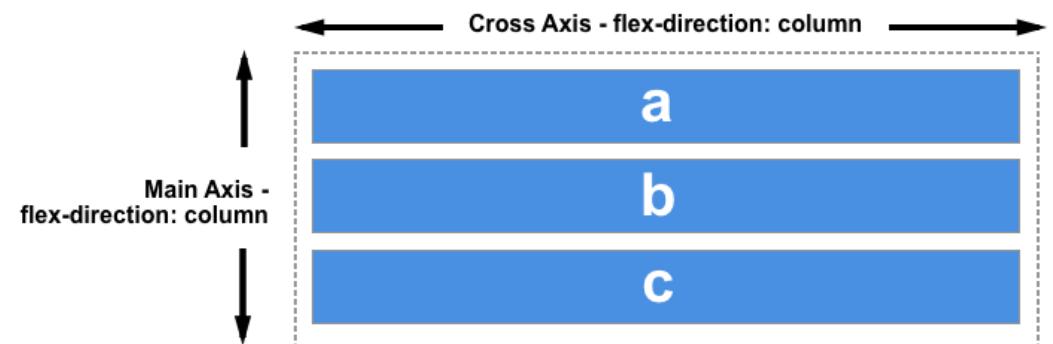
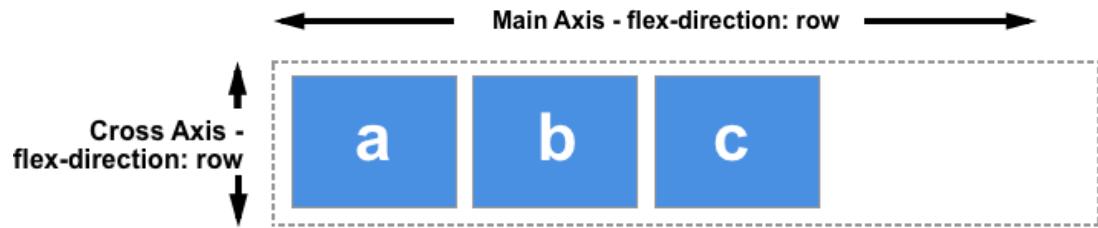


# Flexbox : Les alignements

---

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.



# Flexbox : Aligner sur le Main Axis

---

Pour changer l'alignement des éléments disposés horizontalement (disposition flex par défaut), on va utiliser « `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.

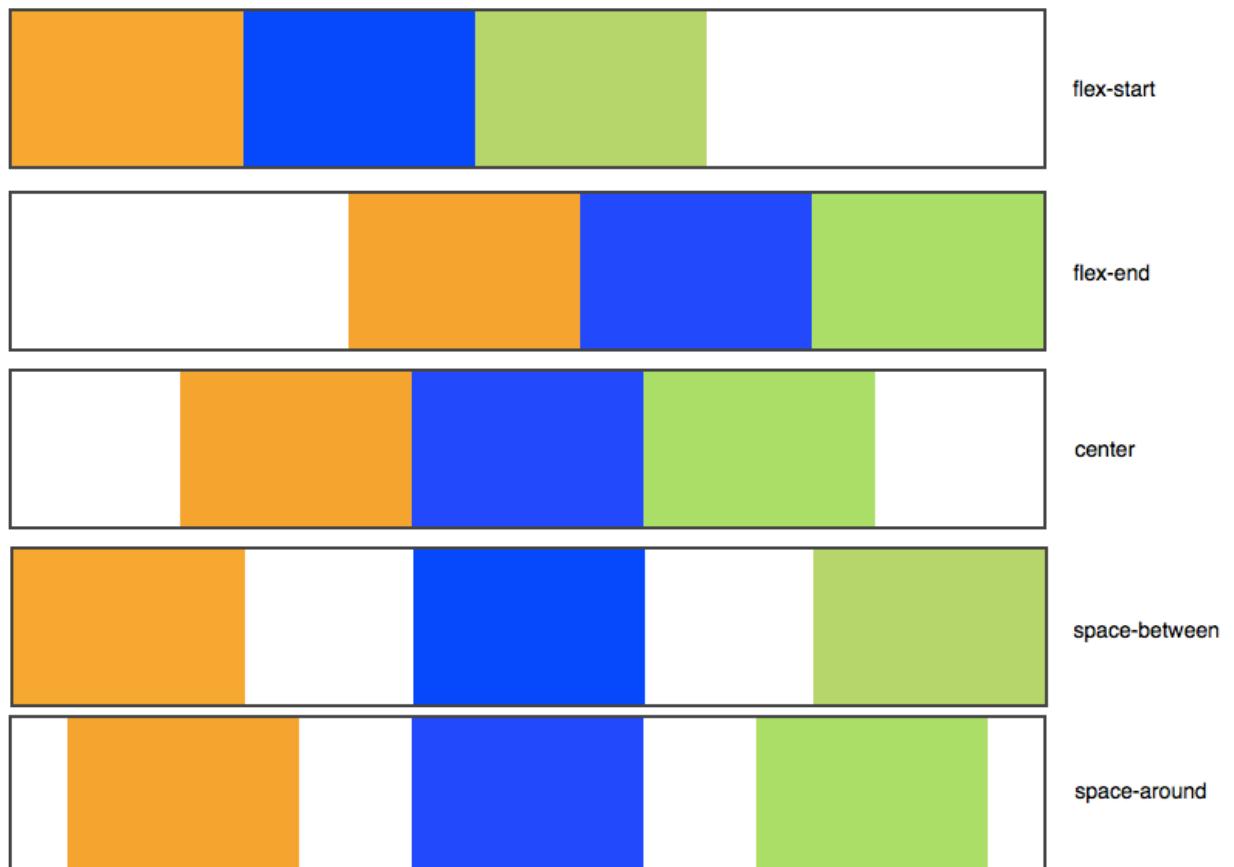
# Flexbox : Aligner sur le Main Axis

---

Exemple :

```
#conteneur
{
  display: flex;
  justify-content: space-around;
}
```

Voici toutes les valeurs possibles :



# Flexbox : Aligner sur le Cross Axis

---

Avec **align-items**, nous pouvons changer leur alignement sur l'axe secondaire.

Il peut prendre ces valeurs :

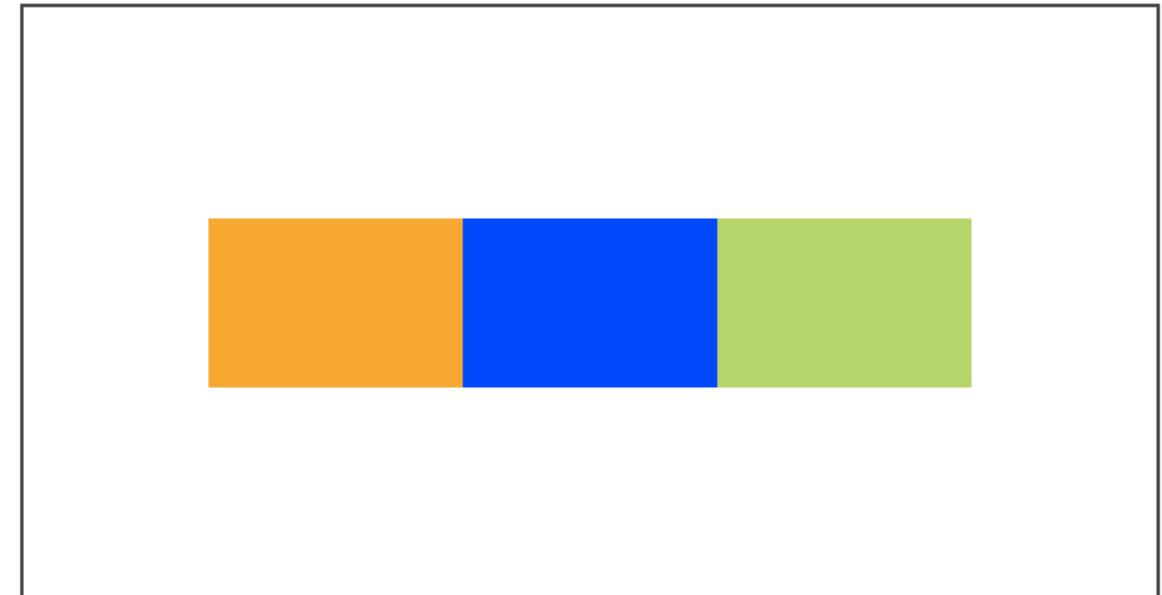
- **stretch** : les éléments sont étirés sur tout l'axe (valeur par défaut) ;
- **flex-start** : alignés au début ;
- **flex-end** : alignés à la fin ;
- **center** : alignés au centre ;
- **baseline** : alignés sur la ligne de base (semblable à flex-start).

# Flexbox : Aligner sur le Cross Axis

---

Pour ces exemples, nous allons partir du principe que nos éléments sont dans une direction horizontale :

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



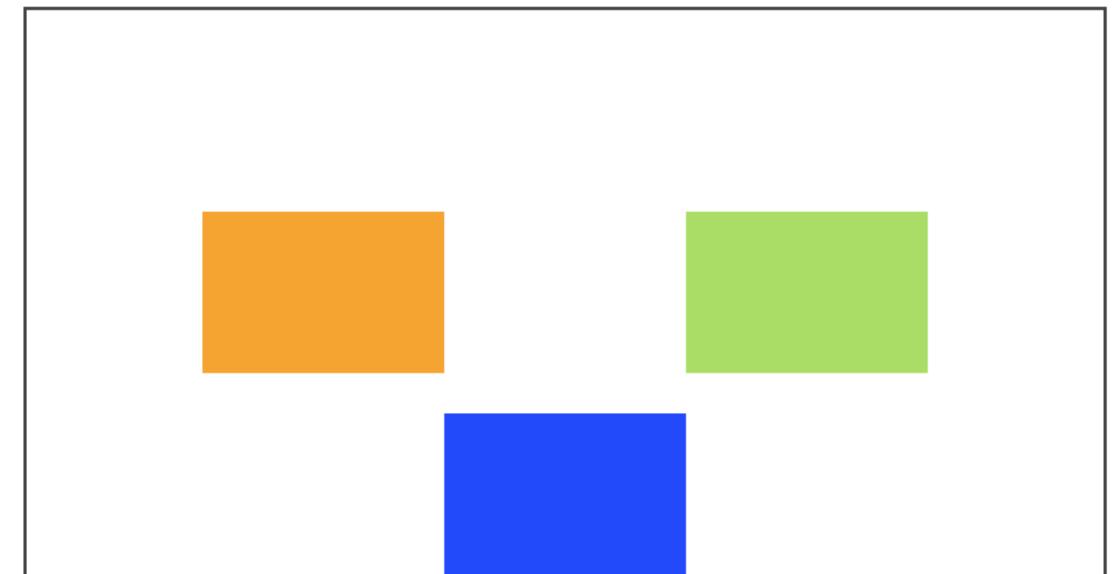
*Un alignement sur l'axe secondaire avec align-items nous permet de centrer complètement l'élément dans le conteneur.*

# Flexbox : Alignement sélectif

---

## Aligner un seul élément sur l'axe secondaire

```
#conteneur {  
  display: flex;  
  flex-direction: row;  
  justify-content: center;  
  align-items: center;  
}  
.element:nth-child(2) /* On sélectionne le deuxième bloc élément */{  
  background-color: blue;  
  align-self: flex-end; /* Seul ce bloc sera aligné à la fin */  
}
```



*Un élément aligné différemment des autres avec align-self.*

# Flexbox : l'ordre des éléments

---

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

Reprendons une simple ligne de 3 éléments :

```
#conteneur
{
    display: flex;
}
```



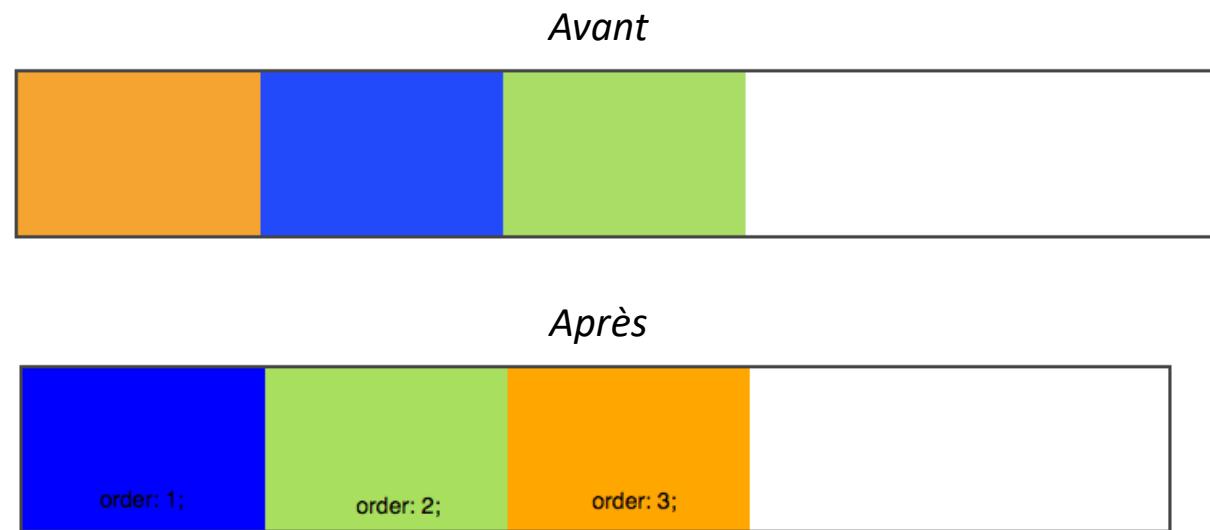
*Une ligne de 3 éléments*

# Flexbox : l'ordre des éléments

---

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

```
.element:nth-child(1) /* selection du 1er enfant du conteneur  
appelé element */  
{  
    order: 3;  
}  
.element:nth-child(2) /* selection du 2ème enfant du conteneur  
appelé element */  
{  
    order: 1;  
}  
.element:nth-child(3) /* selection du 3ème enfant du conteneur  
appelé element */  
{  
    order: 2;  
}
```



Avec `order`, nous pouvons réordonner les éléments en CSS

NB : `order` est différent de `flex-direction : wrap-reverse` et `flex-direction : column-reverse` car il permet de modifier le rang de chaque élément individuellement, ce n'est donc pas une simple inversion de l'ordre initial.

# Flexbox : Faire grossir ou maigrir

---

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

```
.element:nth-child(2) /* sélection du 2ème enfant du conteneur */  
{  
    flex: 1;  
}
```



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

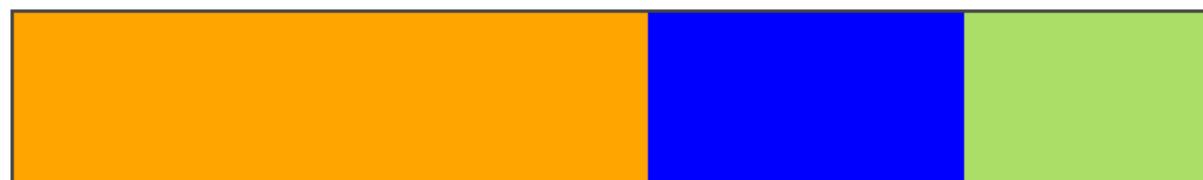
# Flexbox : Faire grossir ou maigrir

---

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

```
.element:nth-child(1) /* sélection du 1er enfant de notre conteneur */  
{  
    flex: 2;  
}  
.element:nth-child(2) /* sélection du 2ème enfant de notre conteneur */  
{  
    flex: 1;  
}
```

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



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