HTML - CSS

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

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 e-mails, la messagerie instantanée, etc.

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

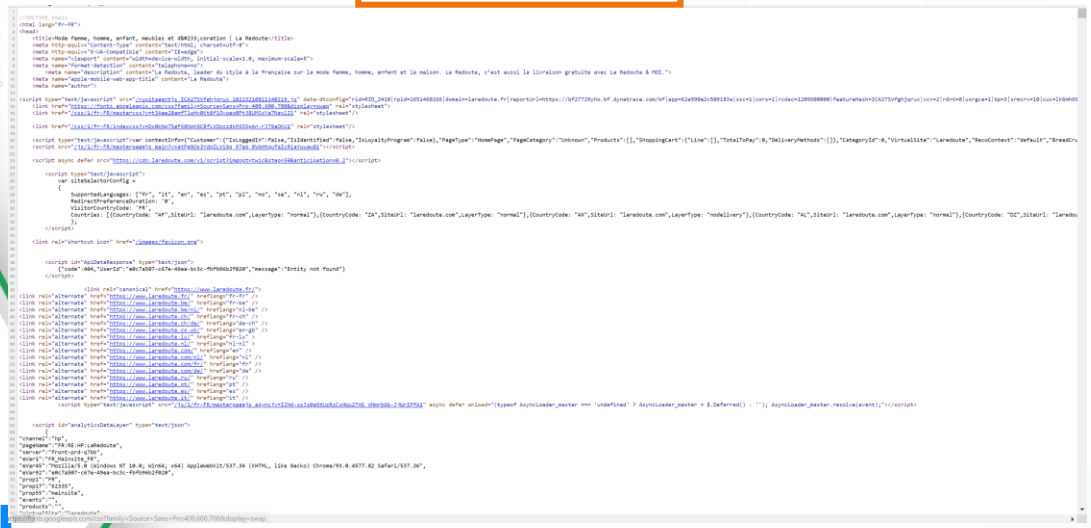
Le langage HTML a été inventé par un certain Tim Berners-Lee en 1991, le CSS et le JavaScript en 1996.

La 1*ère page web créée :*
http://info.cern.ch/hypertext/WWW/TheProject.html

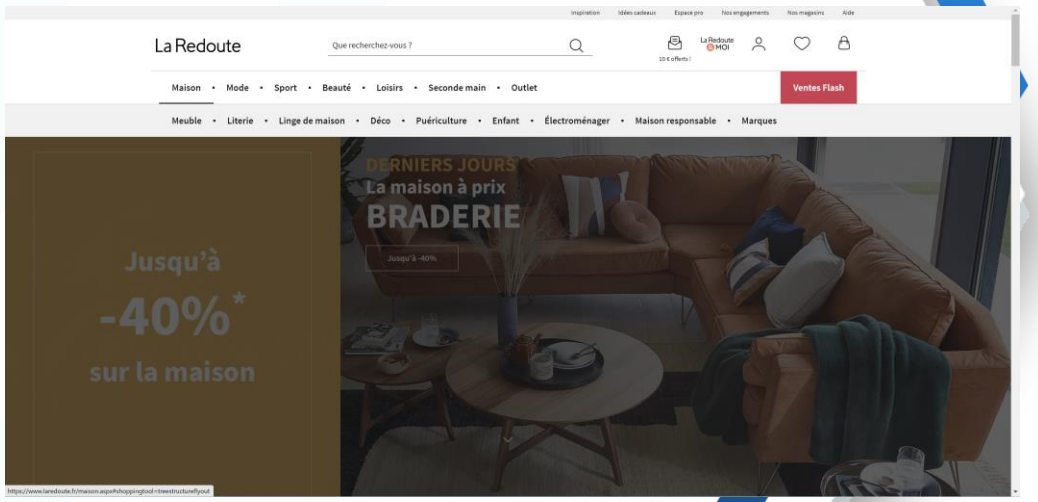
PRÉSENTATION DU WEB

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

Code → **Interprétation** → **Affichage**



(Clic droit dans une page web -> afficher le code source)



PRÉSENTATION DU WEB

Hyper**T**ext **M**arkup **L**anguage (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.

PRÉSENTATION DU WEB

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.

PRÉSENTATION DU HTML

Composition d'une balise

< : chevron ouvrant
> : chevron fermant

1. Balise paire :

<balise> *contenu….* **</balise>**

exemple : Ceci est un paragraphe.

Balise ouvrante

Balise fermante

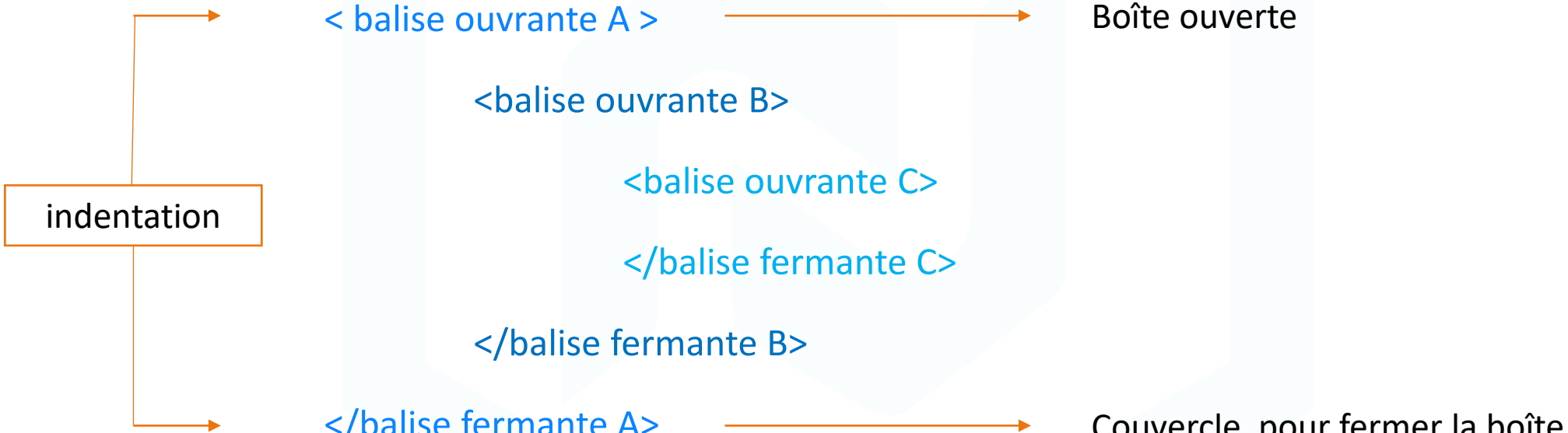
2. Balise orpheline ou auto-fermante :

<balise> ou **<balise />**

exemple :
 (balise de retour à la ligne)

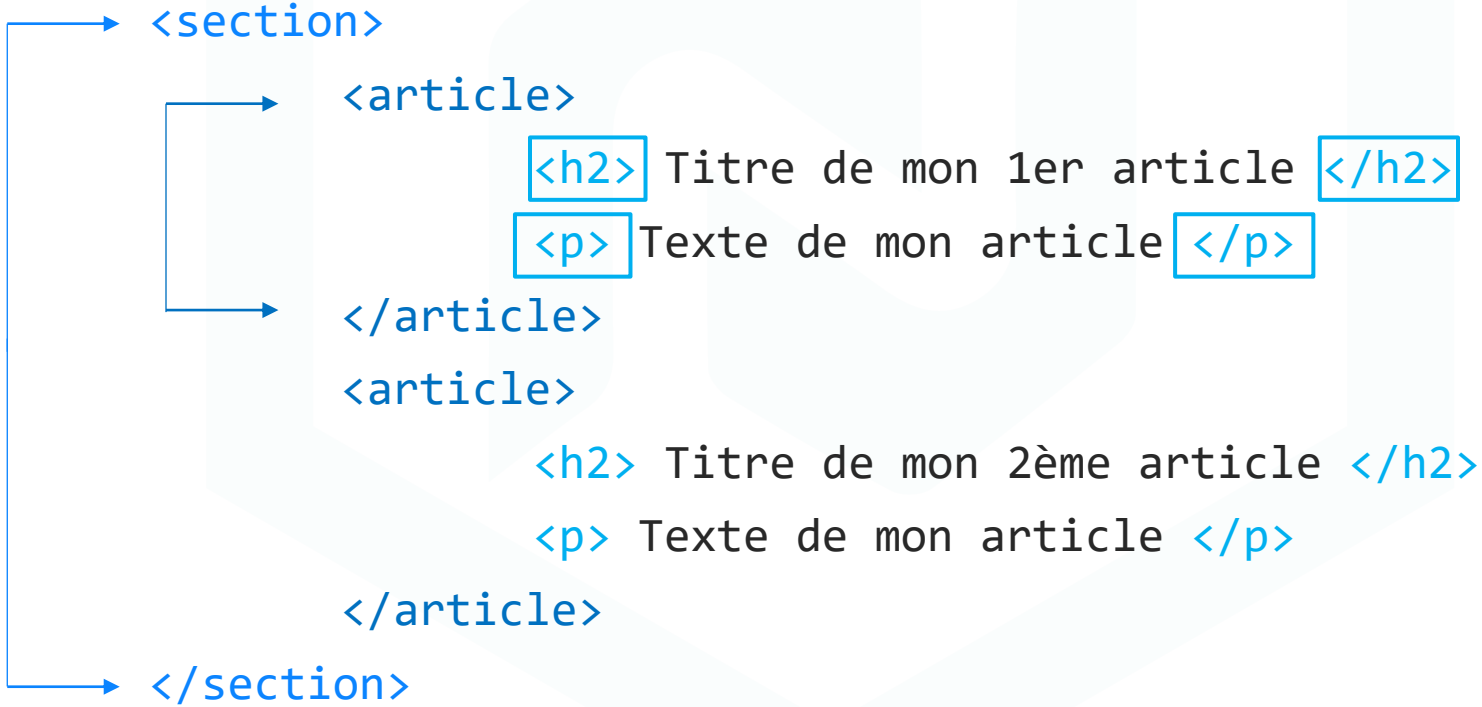
PRÉSENTATION DU HTML

Imbrication des balises



PRÉSENTATION DU HTML

Imbrication des balises : exemple



PRÉSENTATION DU HTML

1er test : Créer un nouveau document html avec VSCode

Exemple text brut

test1.html

```
1   Hello World !
2   Je suis un paragraphe.
3   Je suis la deuxième ligne du paragraphe.
4   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.

Exemple text avec balises

test1.html > p

```
1   <h1>Hello World !</h1>
2   <p>Je suis un paragraphe.<br />
3   Je suis la deuxième ligne du paragraphe.<br />
4   Je suis la troisième et dernière ligne du paragraphe.
5   </p>
```

Résultats dans le navigateur

Hello World !

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

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"**

Les attributs

Exemples

```
<a href="https://www.w3schools.com">Visiter Notre Site</a>
<img src="images/img_girl.jpg" >
```

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

```
<img src="images/img_typo.jpg" alt="Girl with a jacket">
```

PRÉSENTATION DU HTML

NOTRE PREMIER CODE

Structure de base d'une page web

Notre première page web

Zoom sur le « Head »

Zoom sur le « Body »

Structure d'une page HTML

Structure minimaliste d'une page web :

<!DOCTYPE html> : document de type HTML5

<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>
```

PRÉSENTATION DU HTML

Structure complète

```
<!doctype html>
<html lang="fr">
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="css/style.css">
    <title>Hello, world!</title>
</head>
<body>
    <h1>Hello, world!</h1>

    <script src="js/main.js"></script>
</body>
</html>
```

PRÉSENTATION DU HTML

Notre première page web

Créer un fichier .html avec VSCode, générer la structure de base, enregistrer et l'ouvrir dans le navigateur.

(raccourci emmet : ! + entrer ou html:5 + entrer)

PRÉSENTATION DU HTML

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.

PRÉSENTATION DU HTML

La balise « head »

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

PRÉSENTATION DU HTML

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. (la balise body est vide)

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

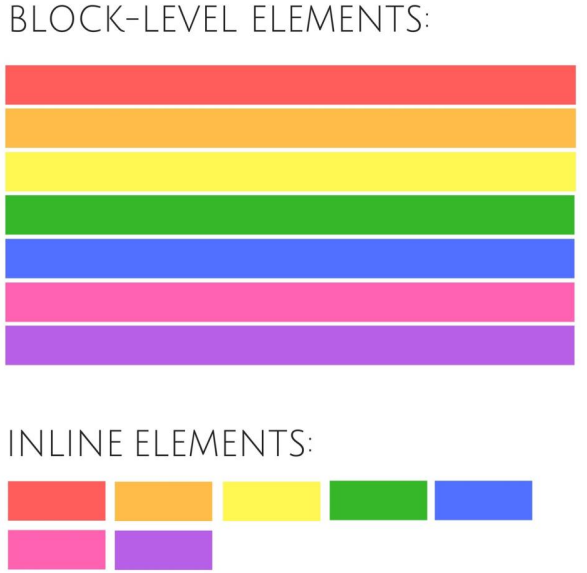
Peuvent contenir des éléments inline

Inline

Ne provoque pas de retour à la ligne

Ne peuvent pas contenir des éléments block

Peuvent contenir des éléments inline



PRÉSENTATION DU HTML

LES BALISES

Les titres

Les paragraphes

Les liens

Les images

Les tableaux

Les listes

Les commentaires

PRÉSENTATION DU HTML

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.

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

Heading 1

Heading 2

Heading 3

Heading 4

Heading 5

Heading 6

PRÉSENTATION DU HTML

LES PARAGRAPHES

La balise **<p>** définit un paragraphe.

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

Exercice titres / paragraphes

PRÉSENTATION DU HTML

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 hypertext. Il a la syntaxe suivante :

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

L'attribut le plus important de l'élément **<a>** est le **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 envoyé 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. Il accepte plusieurs valeurs dont :

- _self : Par défaut. Ouvre le document dans la page en cours
- _blank : Ouvre le document dans un nouvel onglet

Exemple

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

PRÉSENTATION DU HTML

LES LIENS

- URL absolues vs 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>Relatives URLs</h2>
<a href="page2.html">Page 2</a>
<a href="contacts.html">Contacts</a>
```

Liens vers d'autres pages du site

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

```
<a href="https://www.google.com">
        <img src="image/smiley.gif" alt="lien vers google">
</a>
```

Il est également possible de gérer le **href** pour créer un lien directement vers une adresse mail, ou un numéro de tel.

PRÉSENTATION DU HTML

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

n.b : La balise **<img />** doit souvent être à l'intérieur d'une balise de type bloc, soit à l'intérieur de la balise ** ** soit à l'intérieur de la balise **<figure></figure>**

Exemple

```
<img src="images/voiture.jpg" alt="Ma belle voiture" />
```

Taille de l'image - Largeur et hauteur :

```
<img src="images/img_girl.jpg" alt="Girl in a jacket" width="500" height="600">
```

Exercice liens / images

PRÉSENTATION DU HTML

LES TABLEAUX

Structure de base d'un tableau HTML :

```
<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

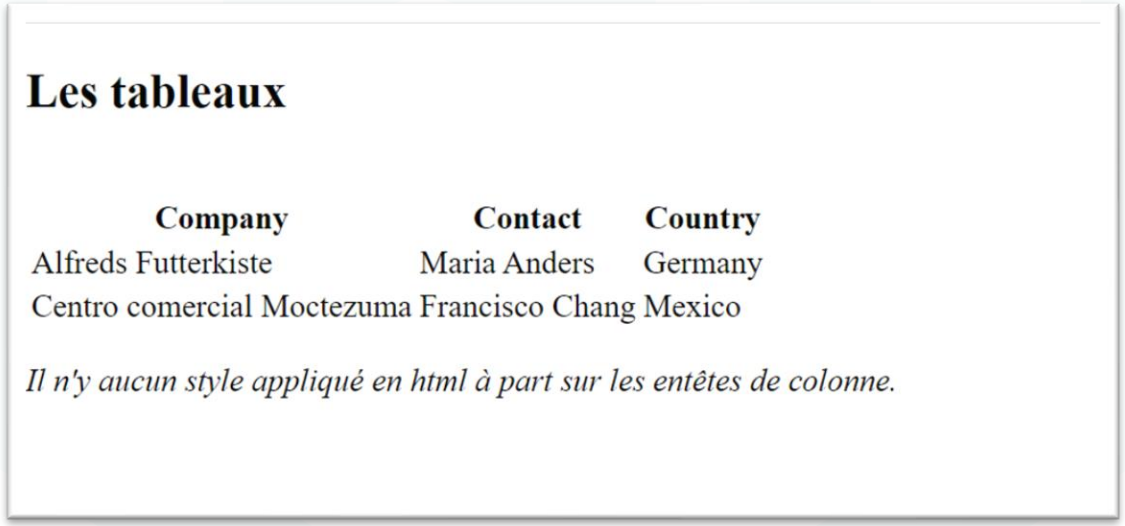
tr : une ligne

th : en-tête de colonne

td : une colonne

PRÉSENTATION DU HTML

LES TABLEAUX



Les tableaux

Company	Contact	Country
Alfreds Futterkiste	Maria Anders	Germany
Centro comercial Moctezuma	Francisco Chang	Mexico

Il n'y aucun style appliqué en html à part sur les entêtes de colonne.

PRÉSENTATION DU HTML

LES LISTES

Non ordonnée

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

- Coffee
- Tea
- Milk

Ordonnée

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

1. Coffee
2. Tea
3. Milk

Listes imbriquées

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

PRÉSENTATION DU HTML

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

Raccourcis :
Sélectionner les lignes, ou placer le curseur sur la ligne à mettre en commentaire, puis :

Win => Ctrl + :
Mac => Cmd + Shift + :

PRÉSENTATION DU HTML

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>
```

Ici, nous ne verrons pas la partie **traintement**, donc on ne précisera pas l'attribut **action**.
Par contre, **methode="get"** nous permettra de visualiser l'envoie des données du formulaire.

PRÉSENTATION DU HTML

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>

Pour en savoir plus :
https://developer.mozilla.org/fr/docs/Web/HTML/Element#formulaires

PRÉSENTATION DU HTML

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

PRÉSENTATION DU HTML

LES FORMULAIRES

Liste complète d'**input** :

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

Page listant les différents types d'input, et leurs attributs

https://developer.mozilla.org/fr/docs/Web/HTML/Element/input

PRÉSENTATION DU HTML

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>
```

PRÉSENTATION DU HTML

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.

(Il conviendra de prévenir l'utilisateur que la sélection **multiple** est activée, et qu'il lui suffit d'appuyer sur la touche **Ctrl** en plus du click)

```
<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>
```

PRÉSENTATION DU HTML

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>
```

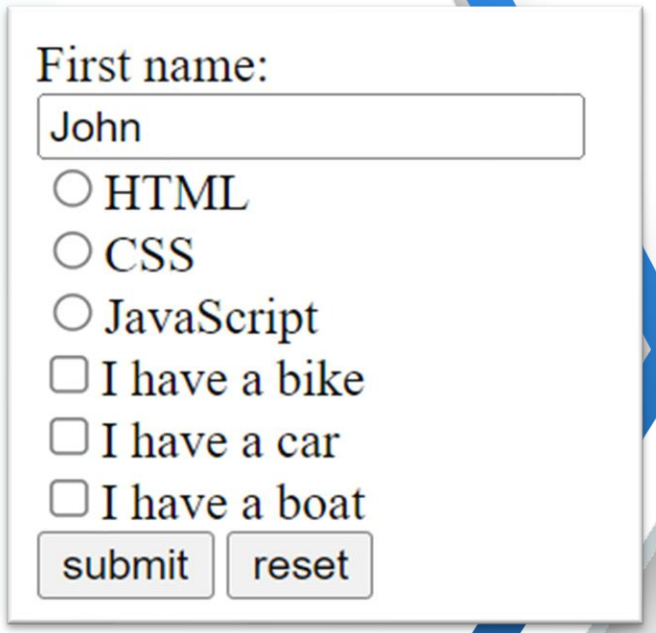
L'attribut **rows** spécifie le nombre de lignes visibles dans une zone de texte.

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

PRÉSENTATION DU HTML

LES FORMULAIRES

```
<form method="get">
        <label for="fname">First name :<br>
                <input type="text" name="fname" value="John">
        </label><br>
        <label for="html">
                <input type="radio" name="fav_language" value="html">HTML
        </label><br>
        <label for="css">
                <input type="radio" name="fav_language" value="css">CSS
        </label><br>
        <label for="javascript">
                <input type="radio" name="fav_language" value="javascript">JavaScript
        </label><br>
        <label for="vehicle1">
                <input type="checkbox" name="vehicle1" value="bike">I have a bike
        </label><br>
        <label for="vehicle2">
                <input type="checkbox" name="vehicle2" value="car">I have a car
        </label><br>
        <label for="vehicle3">
                <input type="checkbox" name="vehicle3" value="boat">I have a boat
        </label><br>
        <input type="submit" value="submit">
        <input type="reset" value="reset">
</form>
```



PRÉSENTATION DU HTML

LES FORMULAIRES : EXERCICE

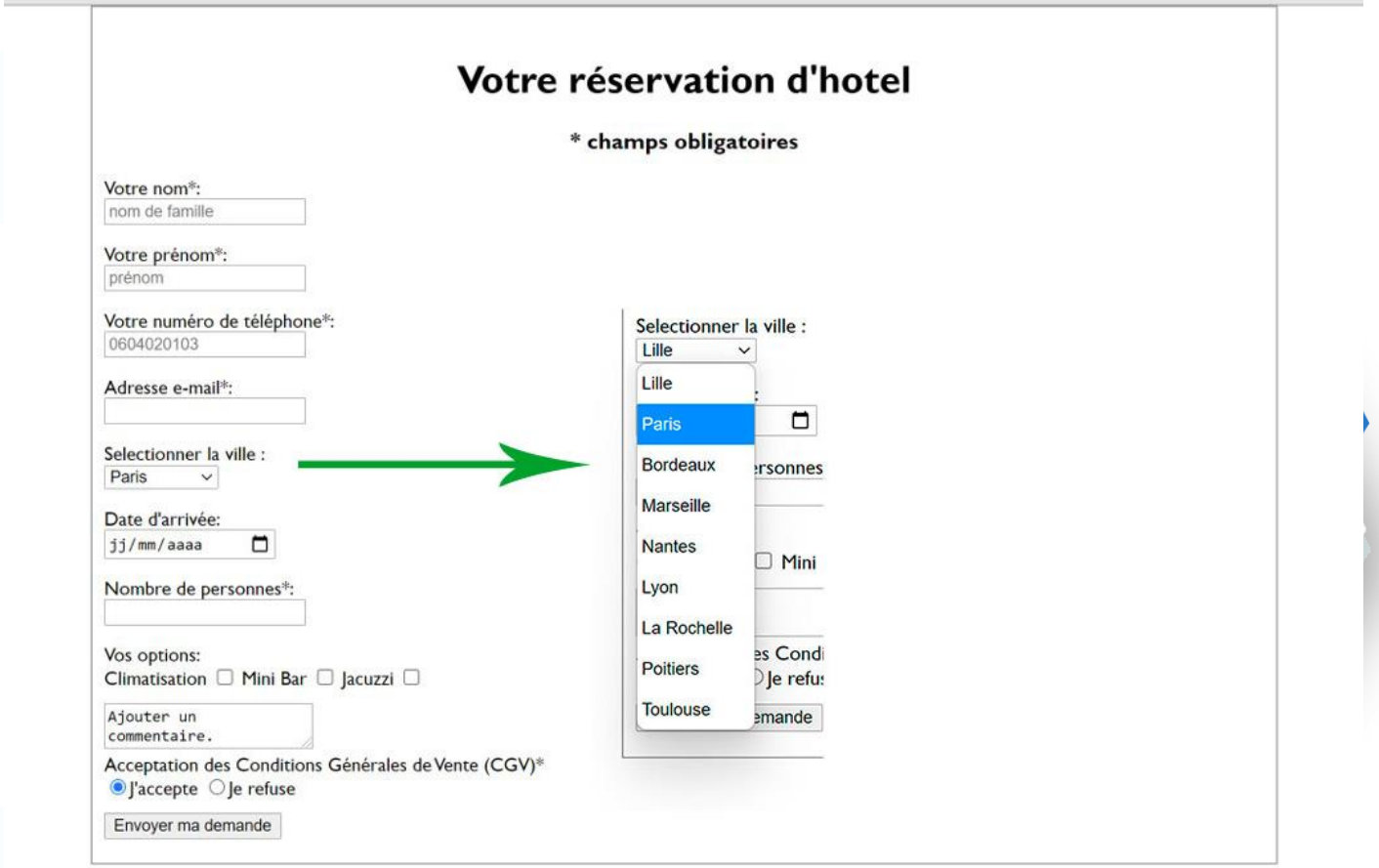
Recréer cette page en HTML

Sera fourni :

- 1 fichier image du modèle
- 1 fichier formulaire.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>
```



PRÉSENTATION DU HTML

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 -> Paragraphe 1 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

PRÉSENTATION DU HTML

LES AUTRES BALISES

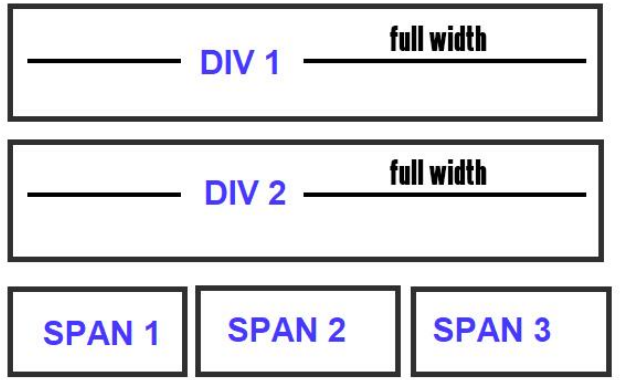
Balises utilisées lorsqu'il n'y a pas d'autre balise qui correspond au besoin.

** **

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

****

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



PRÉSENTATION DU HTML

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 3D par la nouvelle balise <canvas>.

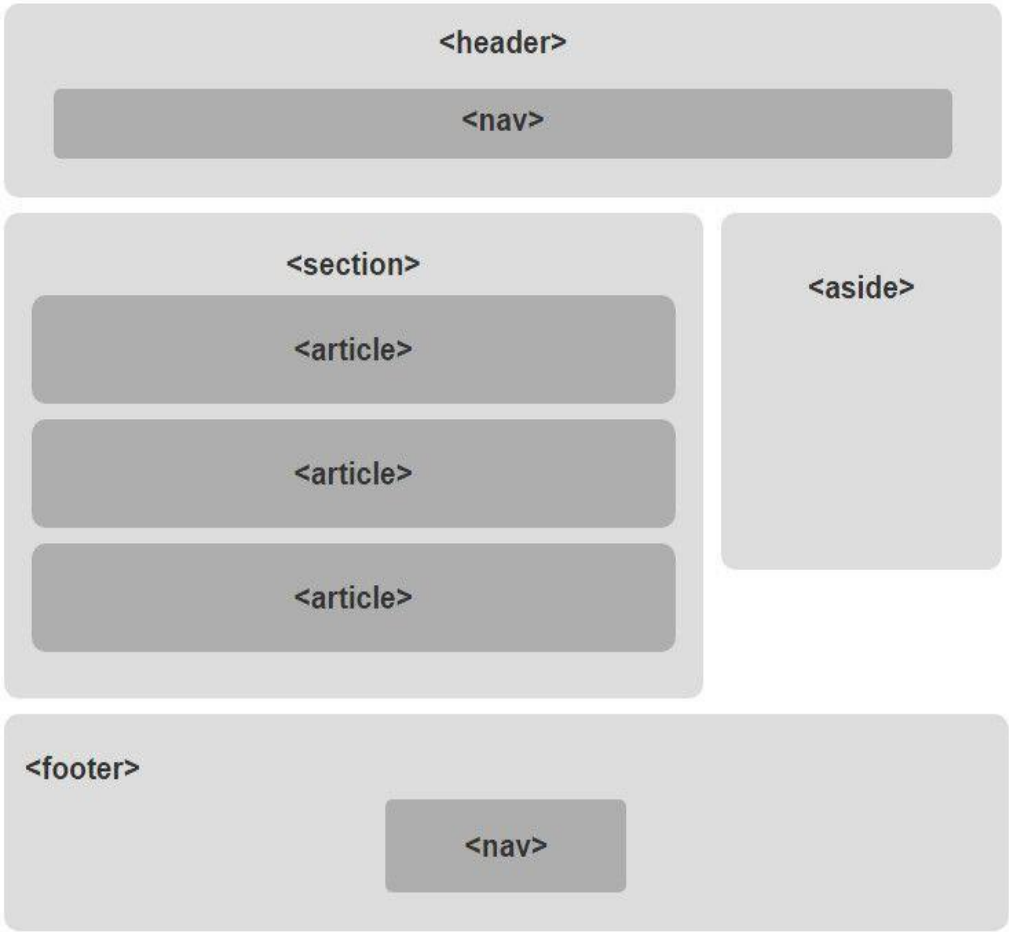
Une profusion de formulaires novateurs comme les curseurs ou les calendriers et la prise en charge de façon native par les navigateurs de la validation des données.

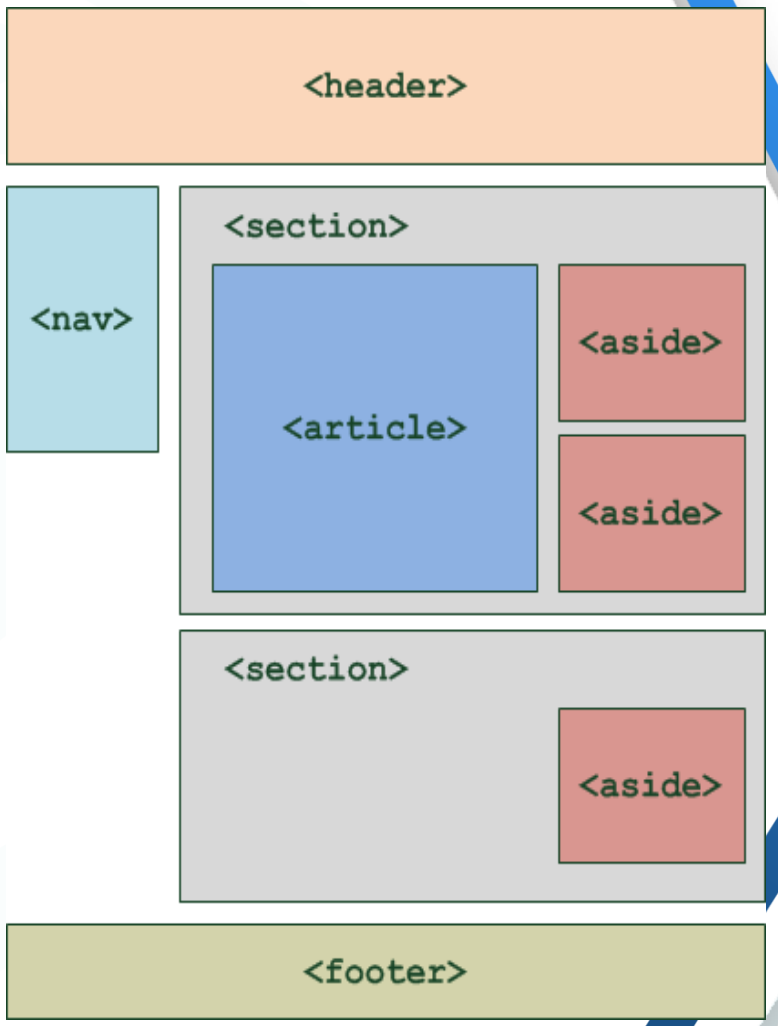
Les nouveaux éléments de formulaire : datetime, date, time, month, number…
et bien d'autres fonctionnalités...

Liste complète des ajouts de HTML 5 :
https://fr.wikipedia.org/wiki/HTML5
https://www.aurone.com/les-nouveautes-apportees-par-html-5

PRÉSENTATION DU HTML

LES BALISES SÉMANTIQUES





PRÉSENTATION DU HTML

LES BALISES SÉMANTIQUES

<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.

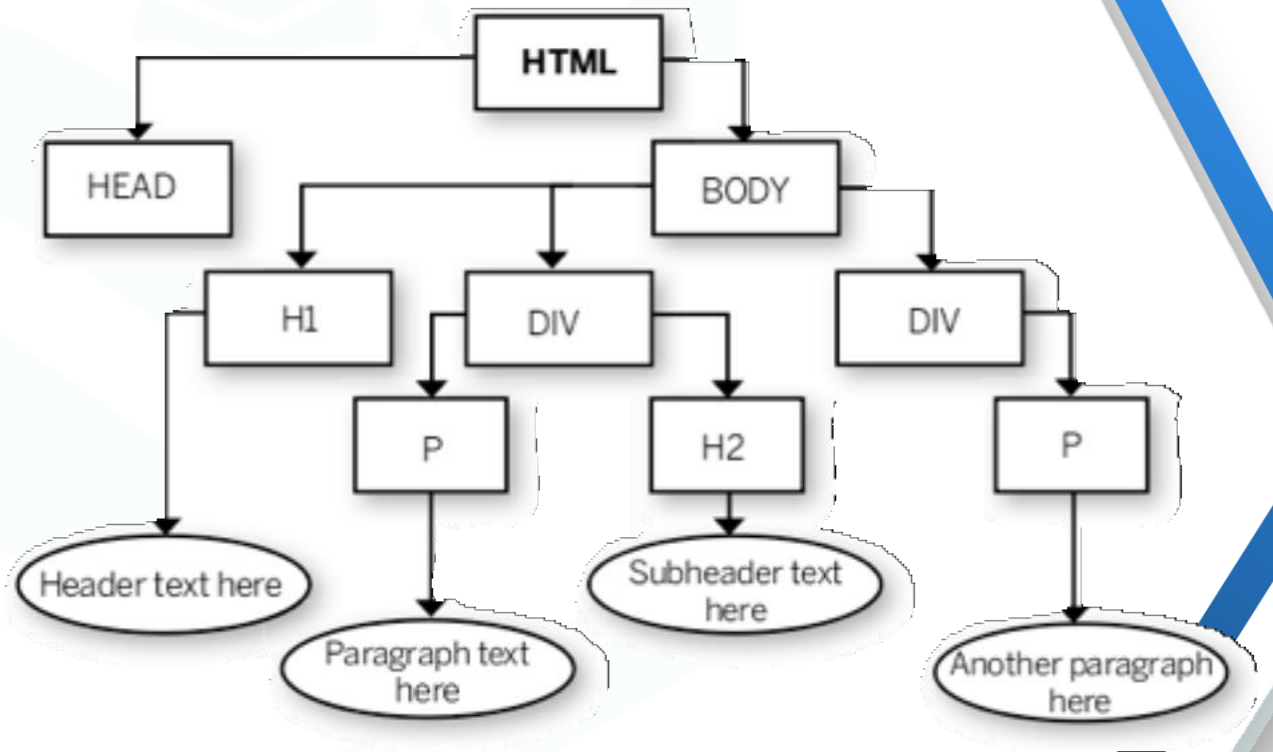
<footer>

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

PRÉSENTATION DU HTML

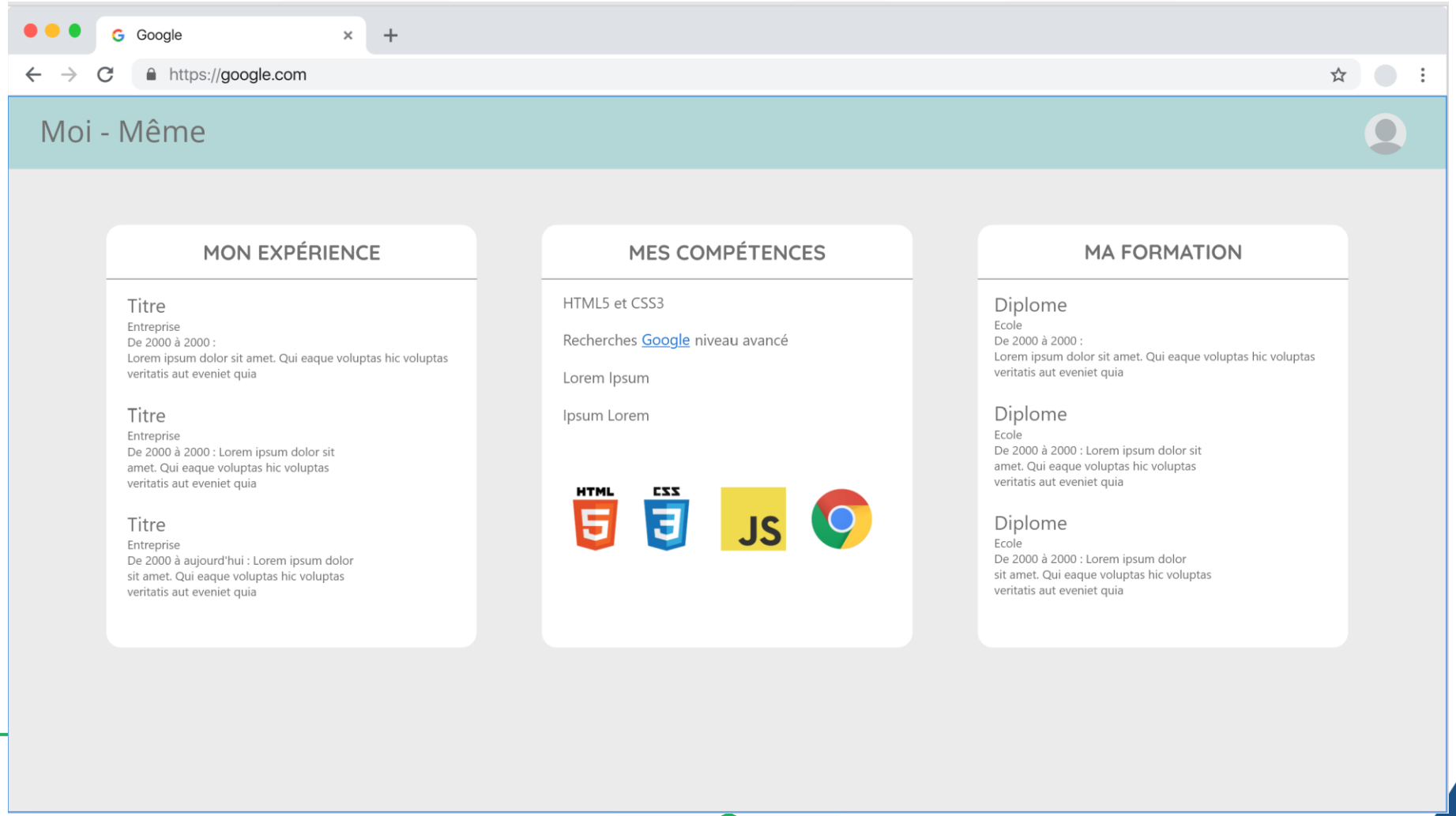
ARBORESCENCE HTML

- L'arborescence HTML contient une racine, des parents, des frères, des enfants
- comme un arbre généalogique
- Appellation technique : **« DOM »** (Document Object Model)



PRÉSENTATION DU HTML

Aperçu du cv une fois stylisé



DÉCOUVERTE DU CSS ET APPLICATION

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

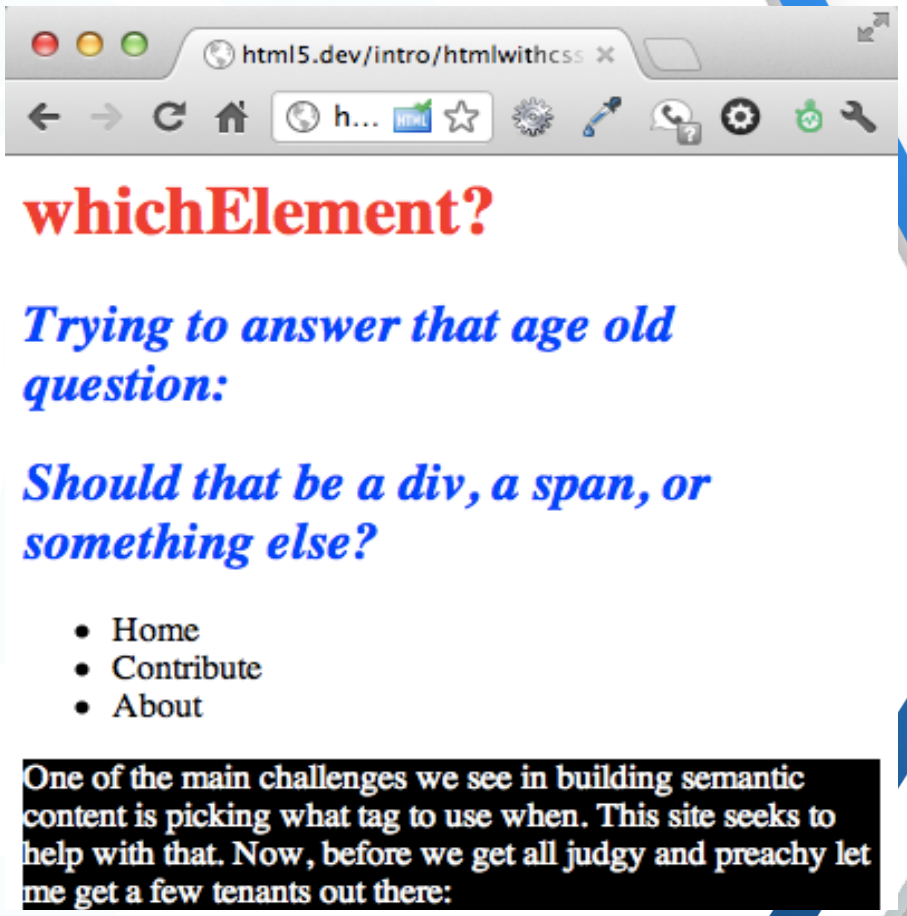
CSS = Cascading Style Sheets (feuilles de style en cascade)

En cascade :

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

DÉCOUVERTE DU CSS ET APPLICATION

```
h1{
   color: red;
}

h2{
   color: blue;
   font-style: italic;
}

p{
   color: white;  background-
color: black;
}
```



DÉCOUVERTE DU CSS ET APPLICATION

Ajouter des styles

Dans le head avec une balise <style></style> :

```
<head>
<style type="text/css">
    #boite1 {
        position: relative;
        margin-left: 20px;
        height: auto;
        width: 643px;
        float: left;
    }
</style>
</head>
```

Feuille de style externe (recommandé) :

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

DÉCOUVERTE DU CSS ET APPLICATION

Sélecteur CSS

Syntaxe:

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

Selecteur de type (nom des balise):

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

DÉCOUVERTE DU CSS ET APPLICATION

Le sélecteur d' identifiant CSS

Dans le html :

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

Dans le css :

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

DÉCOUVERTE DU CSS ET APPLICATION

Le sélecteur de classe CSS

Dans le html :

```
<p class="bg-blue center "> para1 </p>
```

Dans le css :

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

L'attribut class, peut supporter plusieurs valeurs:

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

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

Il est également possible de sélectionner tous les éléments d'un type contenant une certaine class:

```
p.bg-blue {
    Background-color: blue;
}
```

DÉCOUVERTE DU CSS ET APPLICATION

Le sélecteur de regroupement CSS

Le selecteur de regroupenment, espacé par une virgule

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

Sélectionne tous les h1, tous les h2, et tous les p.

Le selecteur d' "enfant", juste séparé par un espace

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

Sélectionne tous les p enfant d'une div

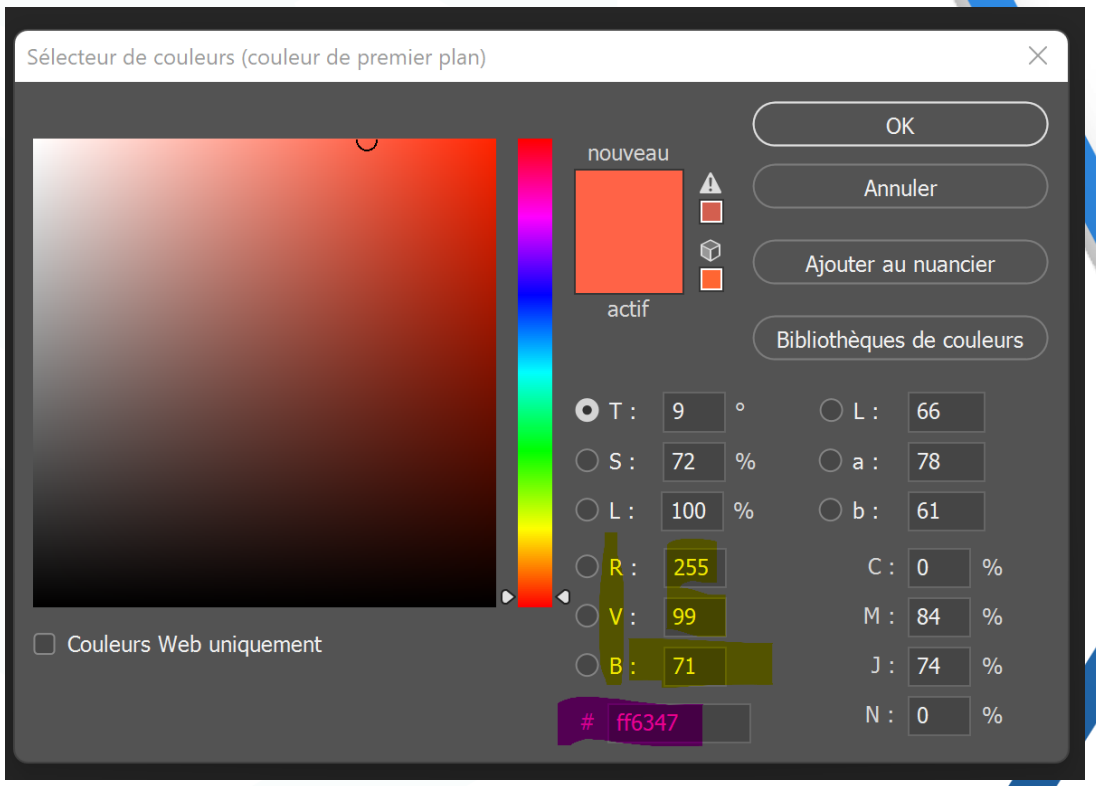
Mini-jeu sur les selecteurs CSS
css diner: https://flukeout.github.io/

DÉCOUVERTE DU CSS ET APPLICATION

Les couleurs

```
p {

    color: rgb(255, 99, 71);

    color: #ff6347; (hexadecimal)

    color: rgba(255, 99, 71, 0.5);



}
```

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



DÉCOUVERTE DU CSS ET APPLICATION

Couleurs d'arrière-plan

```
body {

  background-color: lightblue;

  background: rgba(0, 128, 0, 0.3);

}
```

DÉCOUVERTE DU CSS ET APPLICATION

Image d'arrière-plan

```
body {

  Background-image: url("images/paper.gif");

}
```

DÉCOUVERTE DU CSS ET APPLICATION

Répétition d'arrière-plan

```
body {

  background-image: url("images/gradient_bg.png");

  background-repeat: repeat-x;

  background-repeat: repeat-y;

  background-repeat: no-repeat;

}
```

DÉCOUVERTE DU CSS ET APPLICATION

Défilement d'arrière-plan

```
body {

    background-attachment: fixed;

}
```

DÉCOUVERTE DU CSS ET APPLICATION

Le modèle des boîtes

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

DÉCOUVERTE DU CSS ET APPLICATION

Les bordures

```
div {

  border: 2px solid gray;

  border: 5px dotted #ff0000;

}
```

2px : taille de la bordure

solid : style (normal, pointillés, tirets, double…)

gray : couleur

DÉCOUVERTE DU CSS ET APPLICATION

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 :

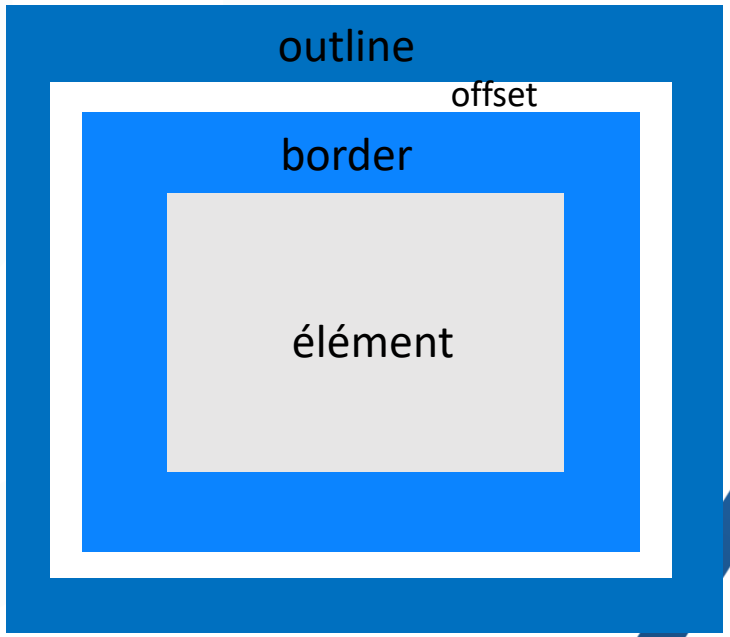
```
div {border-radius: 50%;}
```

DÉCOUVERTE DU CSS ET APPLICATION

Les contours

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 */

}
```



DÉCOUVERTE DU CSS ET APPLICATION

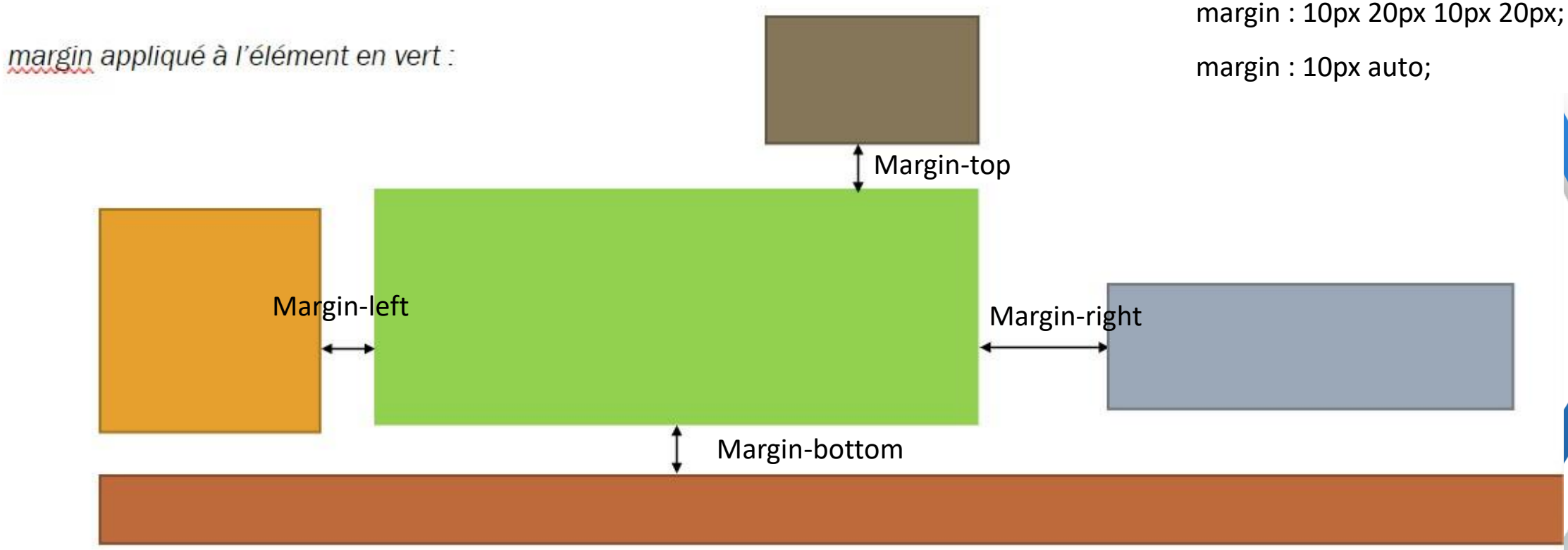
Les marges

margin : 10px;

margin : 10px 20px;

margin : 10px 20px 10px 20px;

margin : 10px auto;



DÉCOUVERTE DU CSS ET APPLICATION

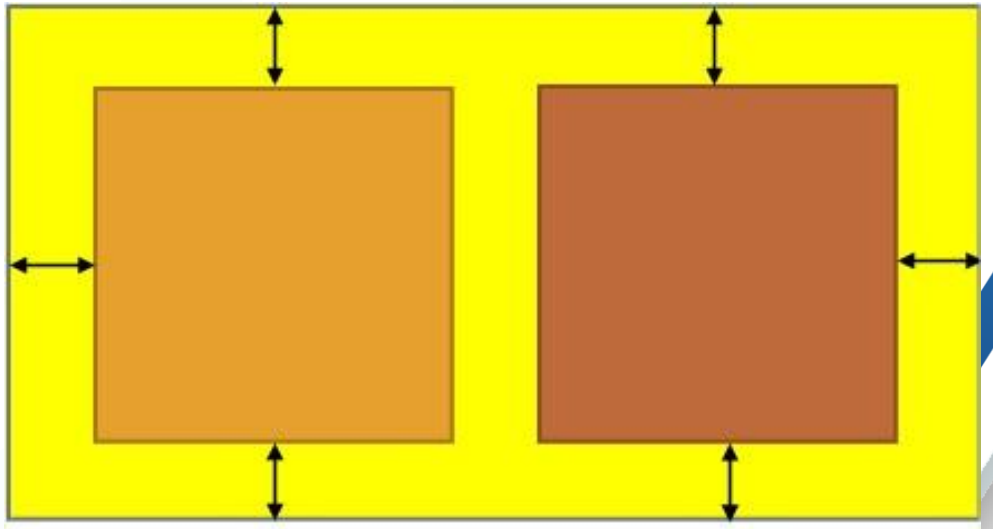
Le padding

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



DÉCOUVERTE DU CSS ET APPLICATION

La hauteur

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

DÉCOUVERTE DU CSS ET APPLICATION

La largeur

```
div {

    width: 100%;

    max-width: 400px;

    min-width: 100px;

}
```

DÉCOUVERTE DU CSS ET APPLICATION

Décoration du texte

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

Lien par defaut : Google

Et en utilisant la valeur "none" : Google

DÉCOUVERTE DU CSS ET APPLICATION

Les liens

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

}

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

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

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

DÉCOUVERTE DU CSS ET APPLICATION

Aligner les textes

```
p {
    text-align: left;  /* alignement à gauche (c'est la valeur par défaut) */
}

p {
    text-align: center; /* texte centré dans son conteneur */
}

p {
    text-align: right;  /* alignement à droite */
}

p {
    text-align: justify;  /* le texte est justifié et prend toute la largeur du conteneur */
}
```

DÉCOUVERTE DU CSS ET APPLICATION

Les polices de caractères (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/

DÉCOUVERTE DU CSS ET APPLICATION

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("truetype");
}
```

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. */
}
```

ESPACEMENT DU TEXTE

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

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: 2.5rem; /* 40px/16=2.5em */
}
```

FORMATAGE DES CARACTÈRES

Textes en MAJUSCULES

```
p {
  text-transform: uppercase;
}
```

Majuscules En Début De Mot

```
p {
  text-transform: capitalize;
}
```

Textes en miniscules

```
p {
  text-transform: lowercase;
}
```

PETITES MAJUSCULES

```
p {
  font-variant: small-caps;
}
```

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

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

```
@media screen and (min-width: 480px) {  /* à partir de 480px */
  body {
    background-color: lightgreen;
  }
}

@media screen and (max-width: 900px) and (min-width: 600px) {  /*de 600px à 900px */
  div.example {
    font-size: 50px;
    padding: 50px;
    border: 8px solid black;
    background: yellow;
  }
}
```

DÉCOUVERTE DU CSS ET APPLICATION

Le display : rappel

Exemples :

<P>
<H1,2,3…>

Block

Prennent toute la largeur disponible

Provoque un retour à la ligne

Peuvent contenir des éléments inline

Exemples :
<img>
<a>

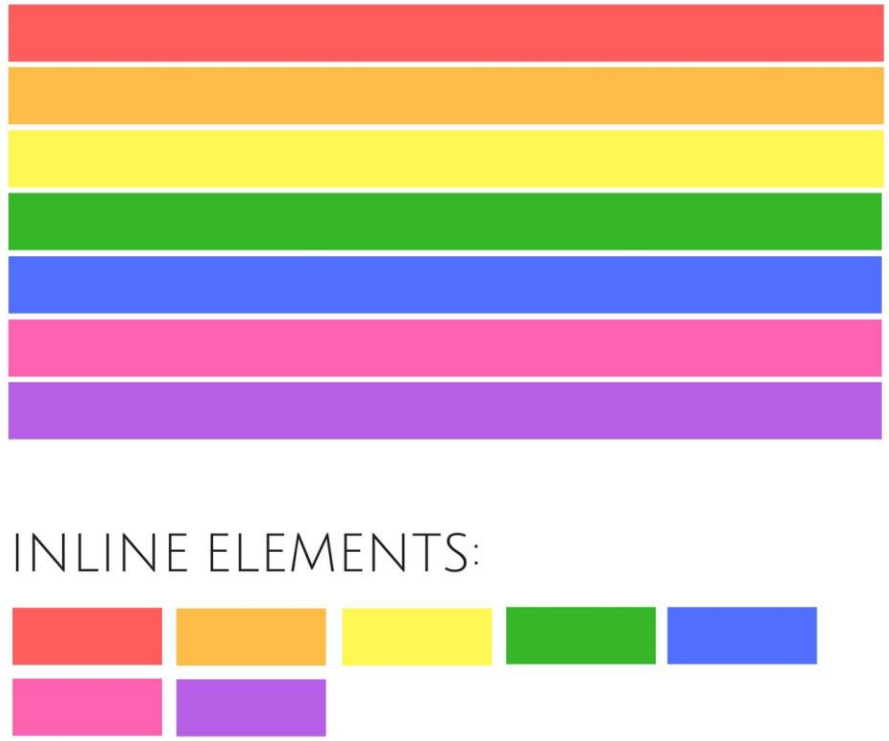
<button>
<input>

Inline

Ne provoque pas de retour à la ligne

Ne peuvent pas contenir des éléments block

Peuvent contenir des éléments inline



DÉCOUVERTE DU CSS ET APPLICATION

Le display

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

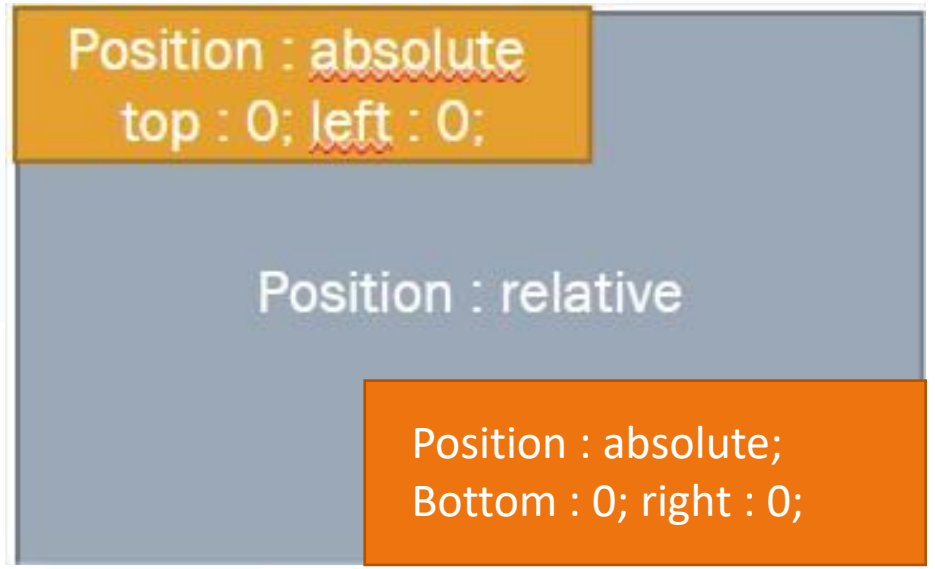
- **NB :** Masquer un élément peut être fait en définissant la propriété display sur none. L'élément sera masqué et la page s'affichera comme si l'élément n'existait pas.

- 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.

DÉCOUVERTE DU CSS ET APPLICATION

Le positionnement

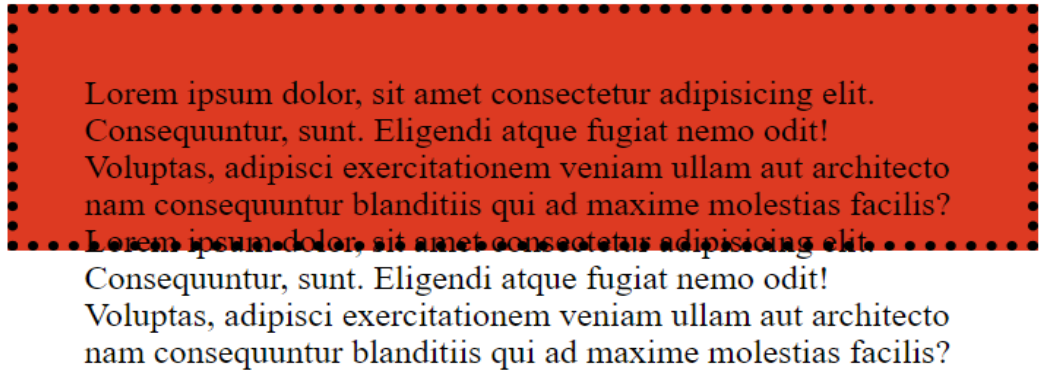
```
div {
  position: static; /* valeur par défaut */
  position: relative;
  position: fixed;
  position: absolute;
  position: sticky;

  bottom: 0; /* ou top: ; */
  right: 0; /* ou left: ; */

}
```



DÉCOUVERTE DU CSS ET APPLICATION

Overflow

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

}
```



DÉCOUVERTE DU CSS ET APPLICATION

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;}

DÉCOUVERTE DU CSS ET APPLICATION

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 complete des pseudo-éléments : https://www.w3schools.com/css/css_pseudo_elements.asp

DÉCOUVERTE DU CSS ET APPLICATION

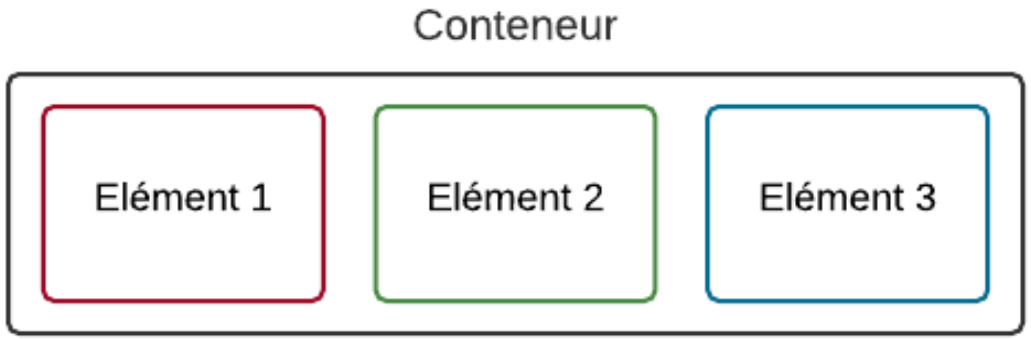
L'opacité

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

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

DÉCOUVERTE DU CSS ET APPLICATION

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 :



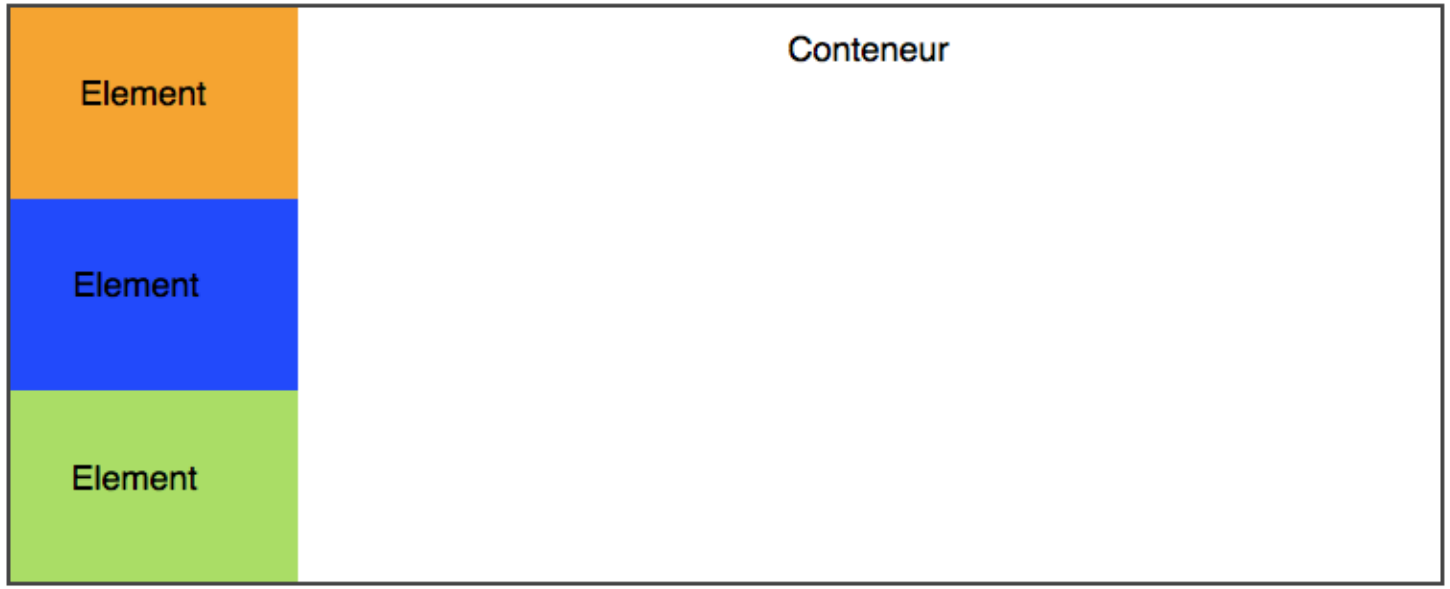
Le conteneur est une balise HTML, et les éléments sont d'autres balises HTML à l'intérieur :

```
<div id="conteneur">
    <div class="element e1">Element 1</div>
    <div class="element e2">Element 2</div>
    <div class="element e3">Element 3</div>
</div>
```

DÉCOUVERTE DU CSS ET APPLICATION

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 :



DÉCOUVERTE DU CSS ET APPLICATION

Flexbox : agencement du contenu

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

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

DÉCOUVERTE DU CSS ET APPLICATION

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.

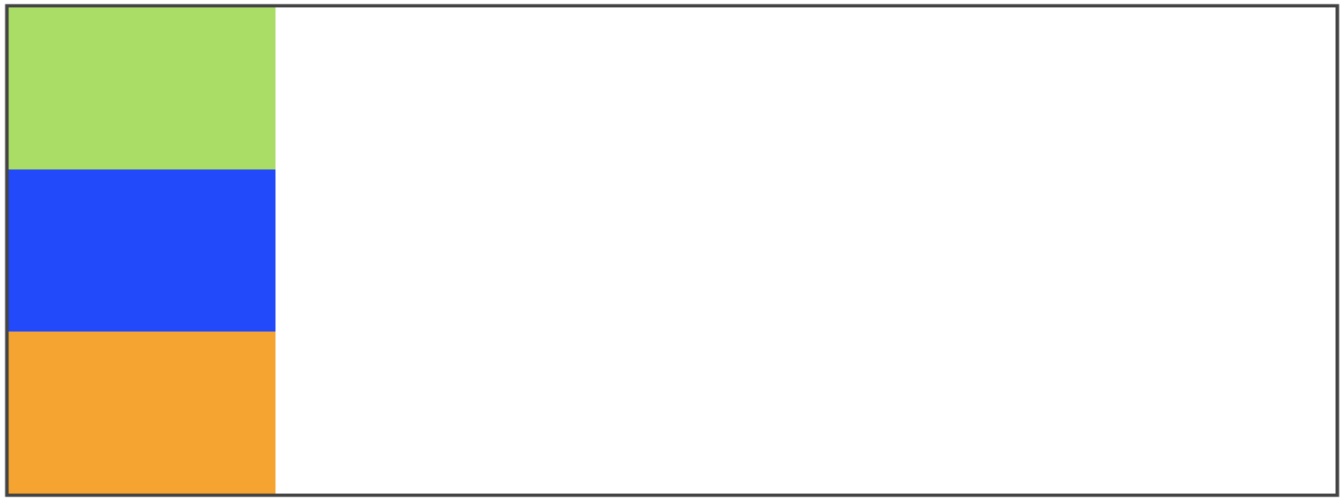
DÉCOUVERTE DU CSS ET APPLICATION

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 :



DÉCOUVERTE DU CSS ET APPLICATION

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 :

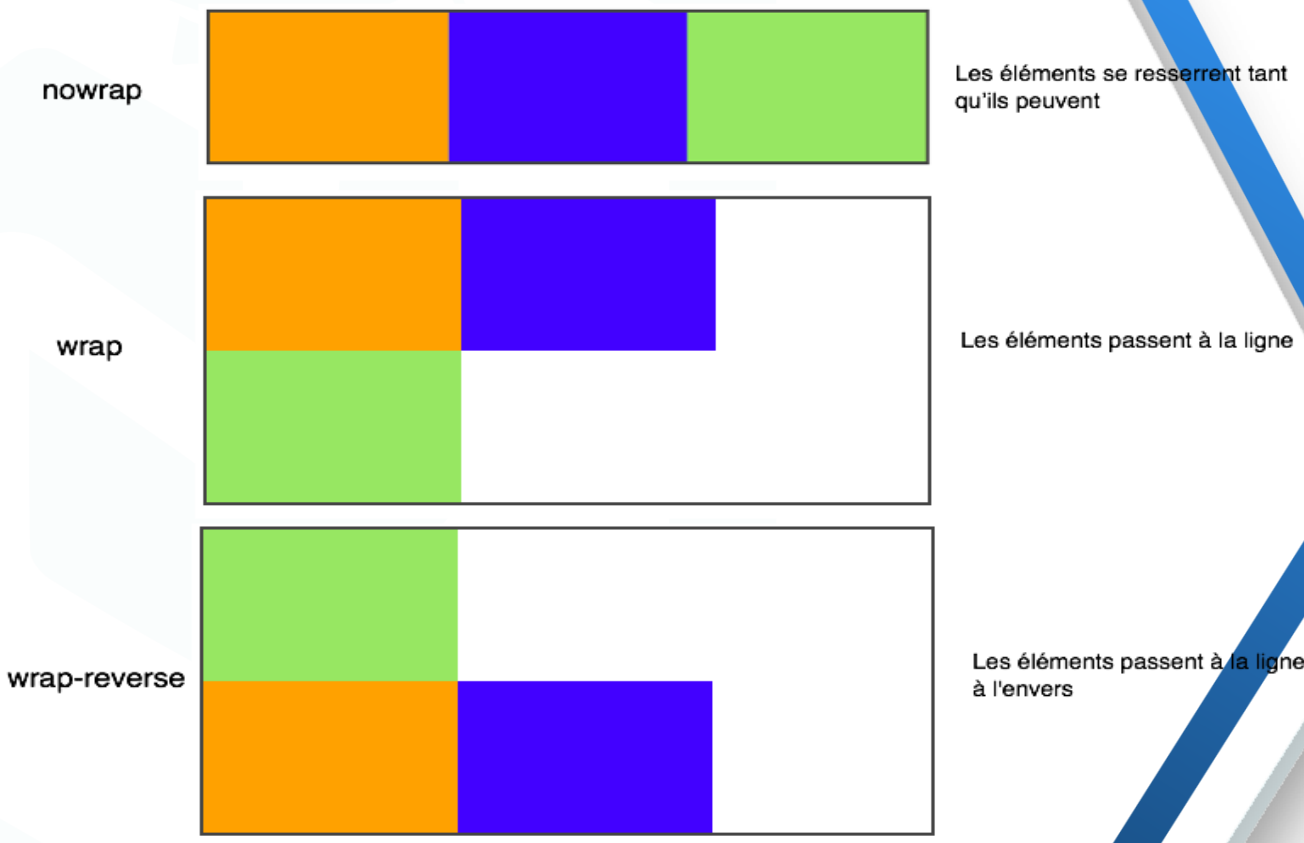
- nowrap : pas de retour à la ligne (par défaut) ;
- wrap : les éléments vont à la ligne lorsqu'il n'y a plus la place ;
- wrap-reverse : les éléments vont à la ligne, lorsqu'il n'y a plus la place, en sens inverse.

DÉCOUVERTE DU CSS ET APPLICATION

Flexbox : retour à la ligne

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

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

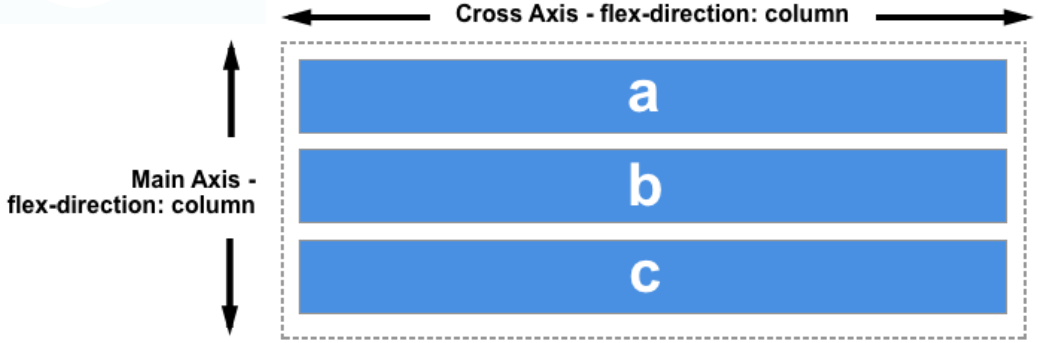


DÉCOUVERTE DU CSS ET APPLICATION

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.



DÉCOUVERTE DU CSS ET APPLICATION

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.

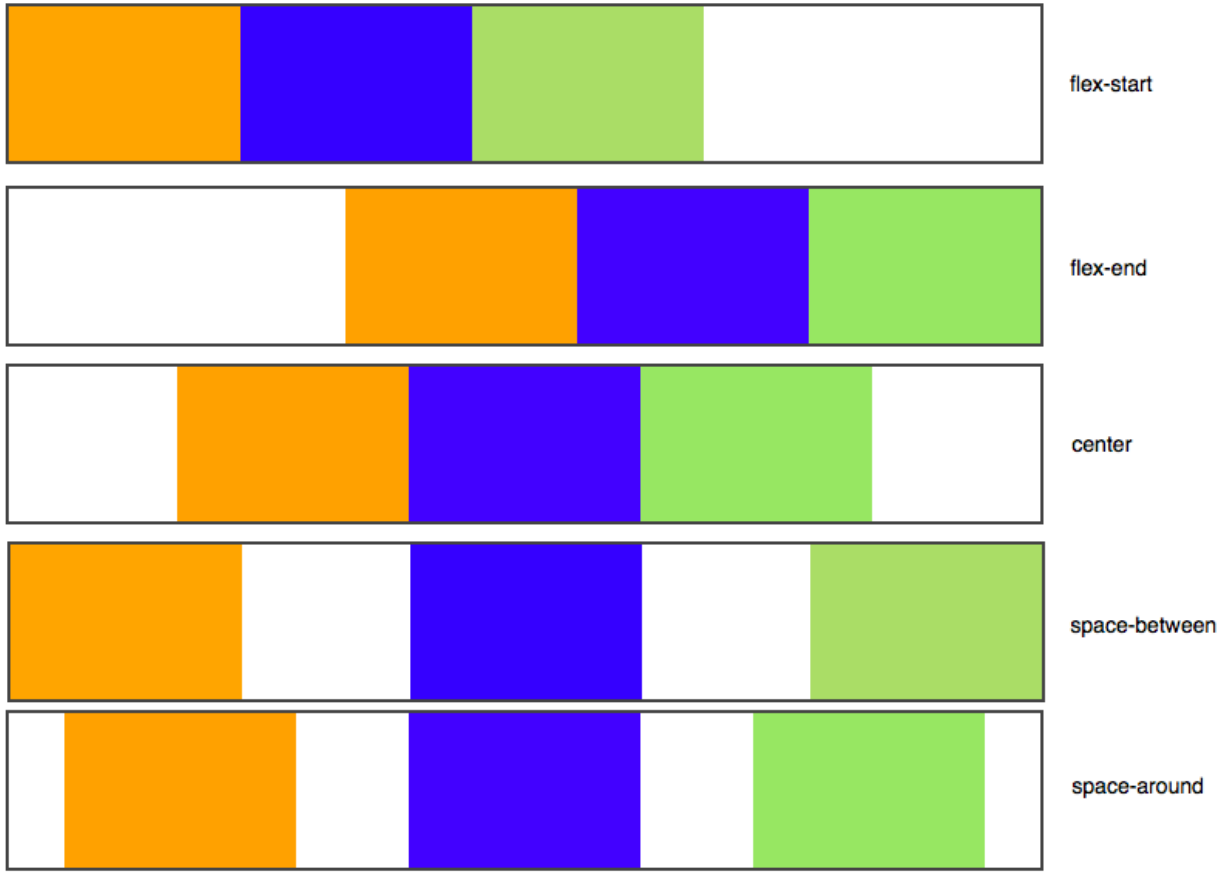
DÉCOUVERTE DU CSS ET APPLICATION

Flexbox : aligner sur le Main Axis

Exemple :

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

Voici toutes les valeurs possibles :



DÉCOUVERTE DU CSS ET APPLICATION

Flexbox : aligner sur le Cross Axis

Avec **align-items** , nous pouvons changer leur alignement sur **l'axe secondaire**.
Il peut prendre ces valeurs :

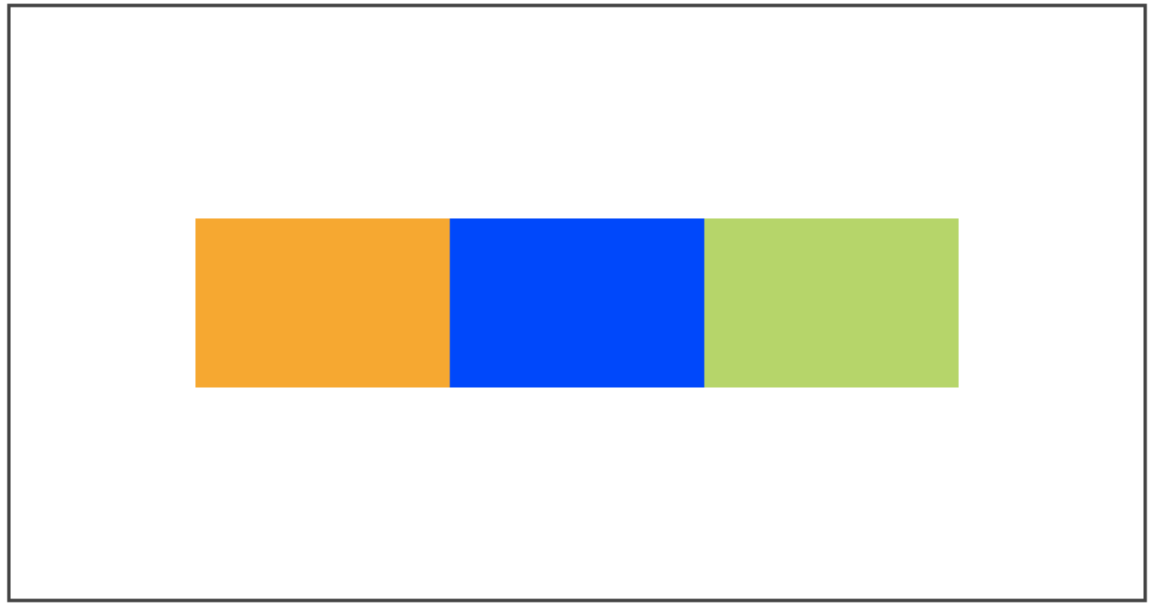
- **stretch** : les éléments sont étirés sur tout l'axe (valeur par défaut) ;
- **flex-start** : alignés au début ;
- **flex-end** : alignés à la fin ;
- **center** : alignés au centre ;
- **baseline** : alignés sur la ligne de base (semblable à flex-start).

DÉCOUVERTE DU CSS ET APPLICATION

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;
}
```



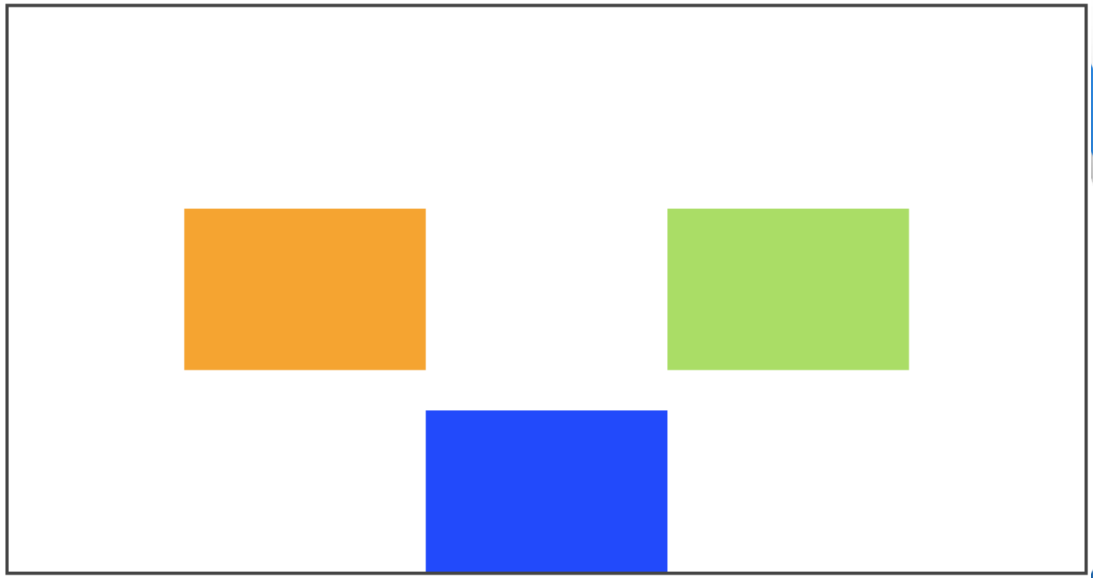
Un alignement sur l'axe secondaire avec align-items nous permet de centrer complètement l'élément dans le conteneur.

DÉCOUVERTE DU CSS ET APPLICATION

Flexbox : alignement sélectif

Aligner un seul élément sur l'axe secondaire

```css
#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.

DÉCOUVERTE DU CSS ET APPLICATION

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.

Reprenons une simple ligne de 3 éléments :

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

Une ligne de 3 éléments

DÉCOUVERTE DU CSS ET APPLICATION

Flexbox : l'ordre des éléments

```
.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;
}
```

Avant

Après

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.

Avec order, nous pouvons réordonner les éléments en CSS

DÉCOUVERTE DU CSS ET APPLICATION

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

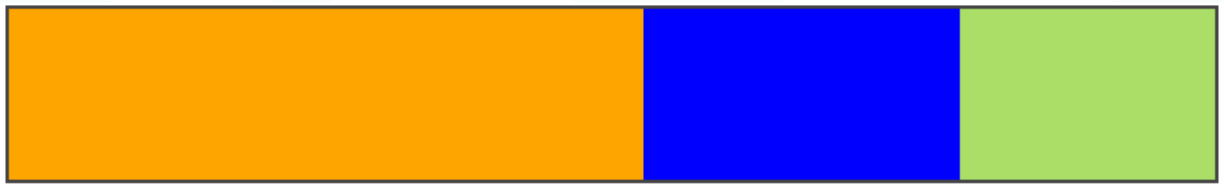
DÉCOUVERTE DU CSS ET APPLICATION

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 troisième élément s'adapte pour occuper l'espace restant

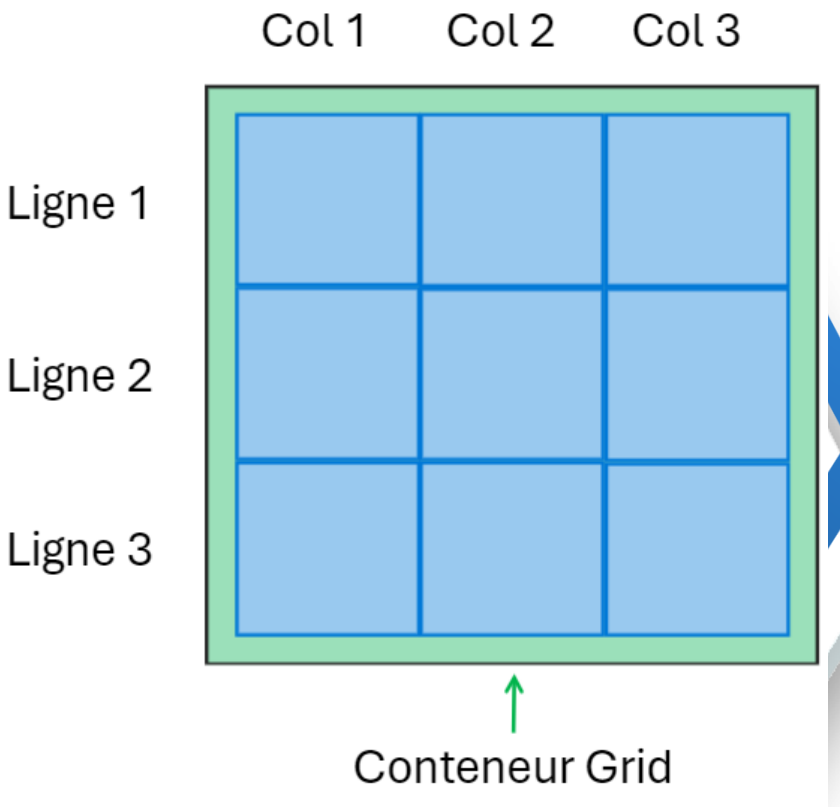
DÉCOUVERTE DU CSS ET APPLICATION

CSS Grid

Les CSS Grids sont une méthode de mise en page puissante en utilisant des lignes et des colonnes.

C'est une grille (grid) de lignes horizontales et verticales qui permet de placer les éléments HTML de manière précise sur une page web.

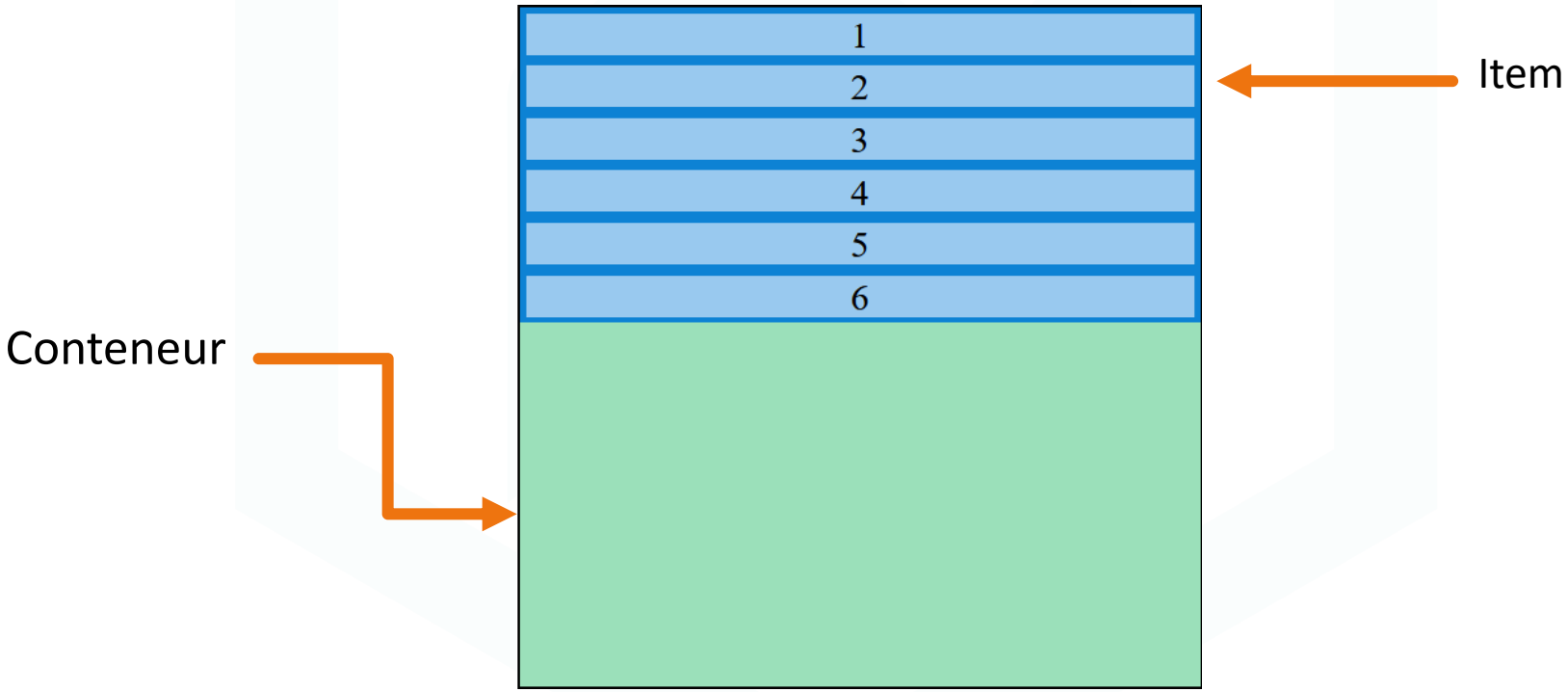
```
<div id="container">
        <div class="item" id="item1">1</div>
        <div class="item" id="item2">2</div>
        <div class="item" id="item3">3</div>
        <div class="item" id="item4">4</div>
        <div class="item" id="item5">5</div>
        <div class="item" id="item6">6</div>
    </div>
```



DÉCOUVERTE DU CSS ET APPLICATION

CSS Grid

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

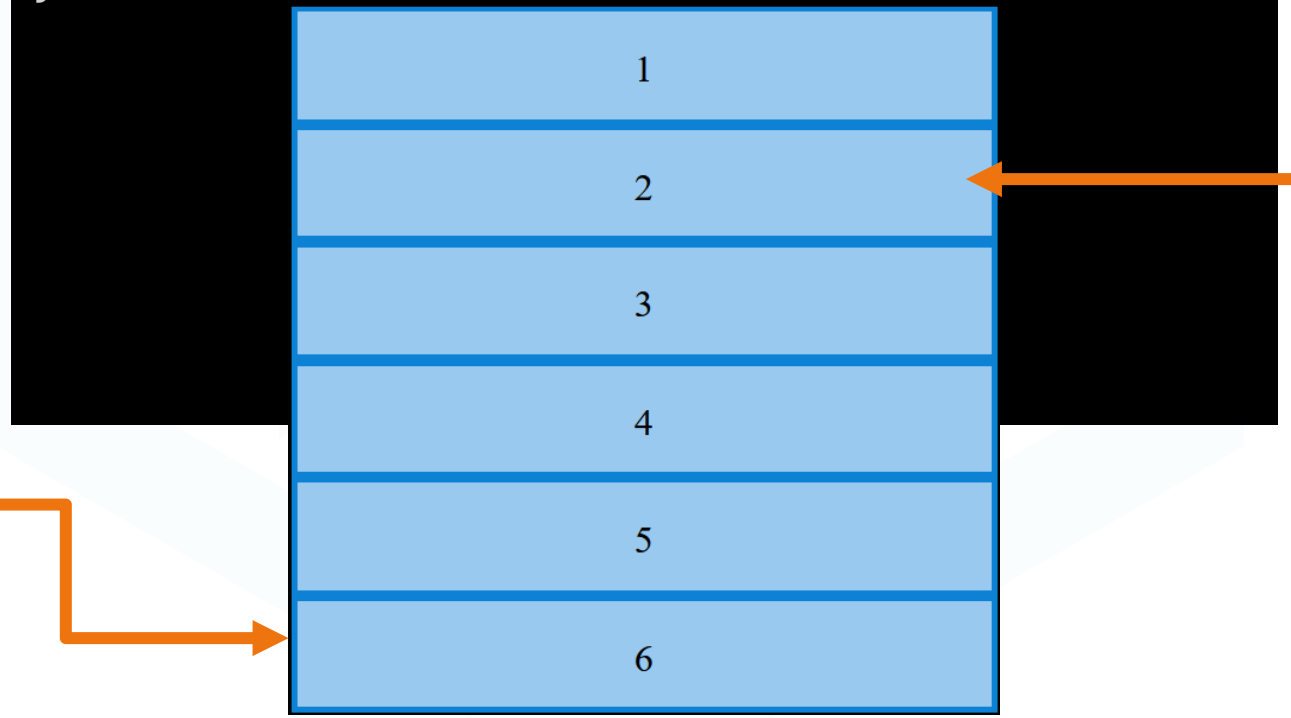


DÉCOUVERTE DU CSS ET APPLICATION

CSS Grid

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

```
#container {
    display: grid;
}
```

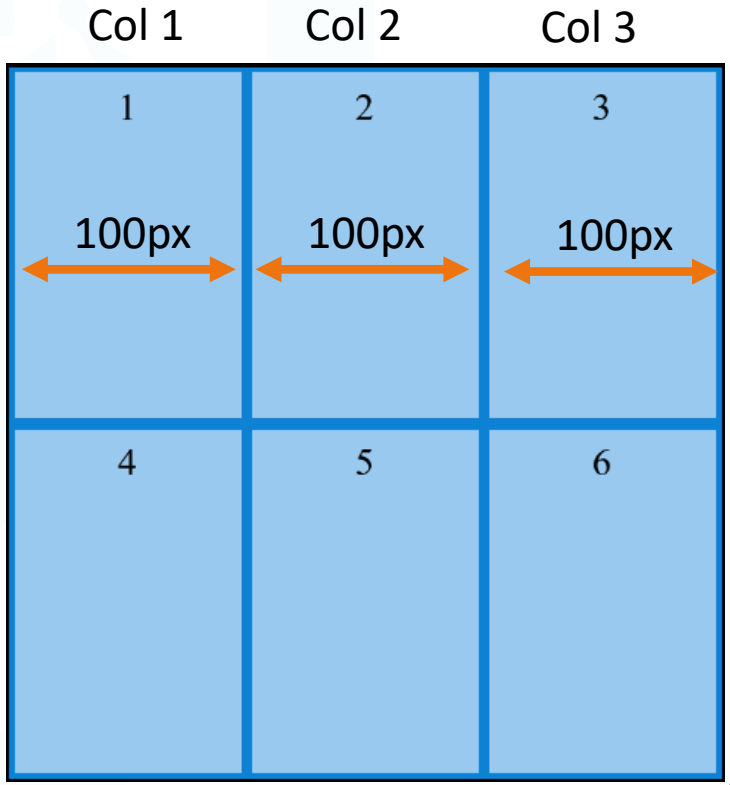


DÉCOUVERTE DU CSS ET APPLICATION

Grid : Les colonnes

La propriété *grid-template-columns* permet de définir le nombre de colonnes et la largeur des colonnes dans une grille CSS.

```
#container {
   display: grid;
   grid-template-columns: 100px 100px 100px;
}
```



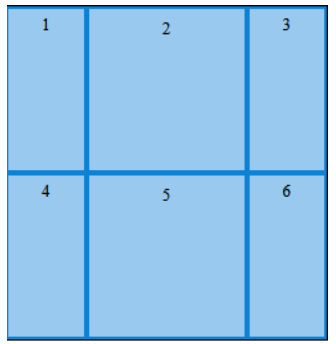
DÉCOUVERTE DU CSS ET APPLICATION

Grid : Les colonnes

Valeurs possibles pour *grid-template-columns* :

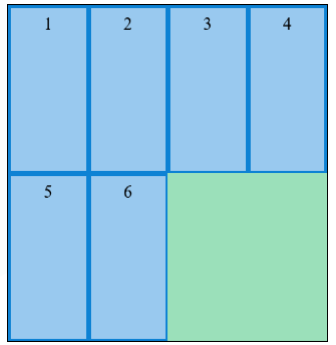
- pourcentage (%) :

```
#container {
 display: grid;
 grid-template-columns: 25%  50%  25%;
}
```



- fractions (fr) :

```
#container {
  display: grid;
  grid-template-columns: 1fr 1fr 1fr 1fr;
}
```



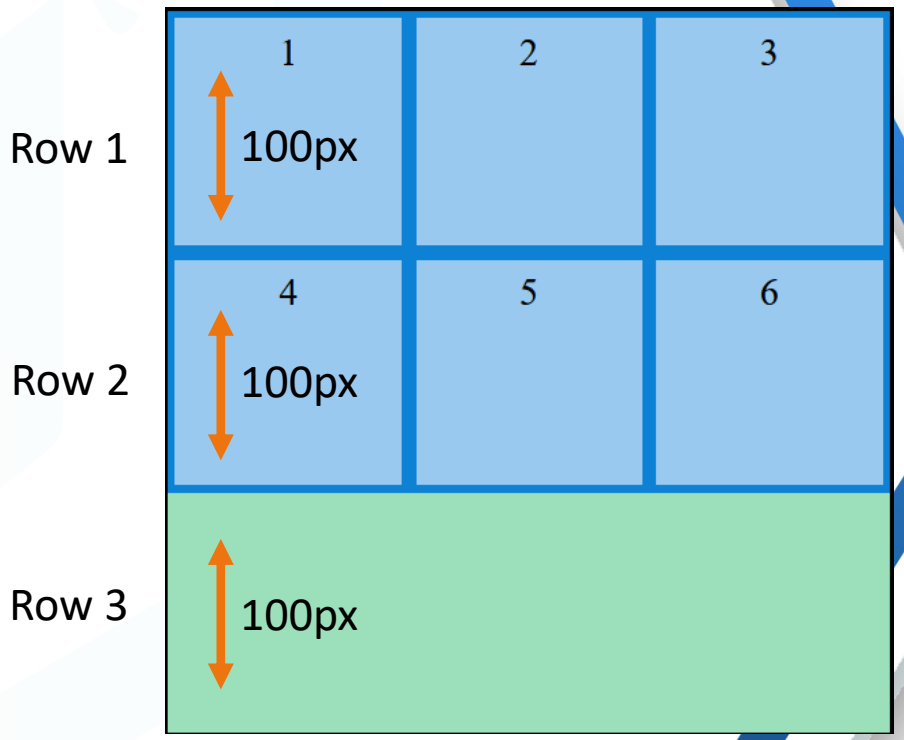
Nombre de colonnes : Le nombre de valeurs définies correspond au nombre de colonnes dans la grille.

DÉCOUVERTE DU CSS ET APPLICATION

Grid : Les lignes

La propriété *grid-template-rows* permet de définir le nombre de ligne et la hauteur des lignes dans une grille CSS.

```
#container {
  display: grid;
  grid-template-columns: 100px 100px 100px;
  grid-template-rows: 100px 100px 100px;
}
```



Similitude avec *grid-template-columns :*
Les deux propriétés offrent un contrôle similaire mais pour les dimensions verticales et horizontales respectivement.

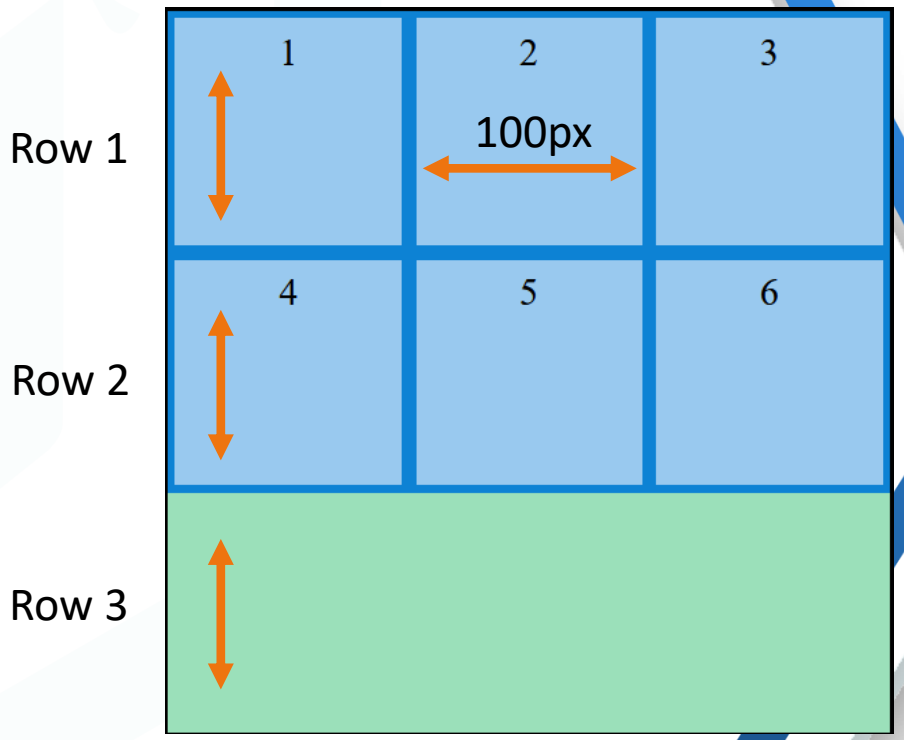
Diversité des unités :
Les valeurs peuvent être définies en pixels (px), pourcentage (%), fractions (fr).

DÉCOUVERTE DU CSS ET APPLICATION

Grid : repeat()

La fonction *repeat()* en CSS Grid permet de définir de manière concise et répétitive la structure de la grille pour les colonnes et les lignes.

```
#container {
   display: grid;
   grid-template-columns: repeat(3,100px);
   grid-template-rows: repeat(3, 1fr);
}
```



DÉCOUVERTE DU CSS ET APPLICATION

Grid : Le positionnement des éléments

Les propriétés suivantes permettent de positionner un élément spécifique dans une grille :

- ***grid-column-start*** **:**
  Détermine où commence un élément dans la grille sur l'axe des colonnes.

- ***grid-column-end :***
  Indique où se termine un élément dans la grille sur l'axe des colonnes.

- ***grid-row-start :***
  Spécifie où commence un élément dans la grille sur l'axe des lignes.

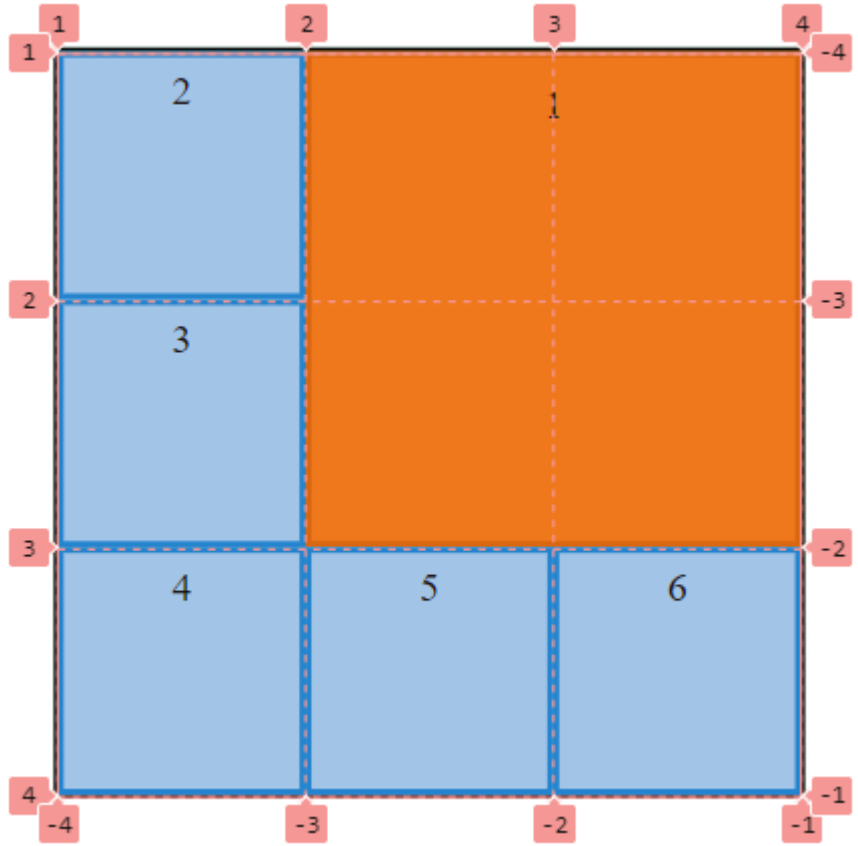
- ***grid-row-end :***
  Marque où se termine un élément dans la grille sur l'axe des lignes.

DÉCOUVERTE DU CSS ET APPLICATION

Grid : Le positionnement des éléments

Pensez à utiliser l'inspecteur de grille dans les outils de développement de votre navigateur pour voir comment les éléments se placent sur les lignes d'une grille.

```
#item1{

    background-color: #EE740F;
    border: 3px solid #d66102;

    grid-column-start: 2 ;
    grid-column-end: 4 ;
    grid-row-start: 1 ;
    grid-row-end: 3 ;

}
```



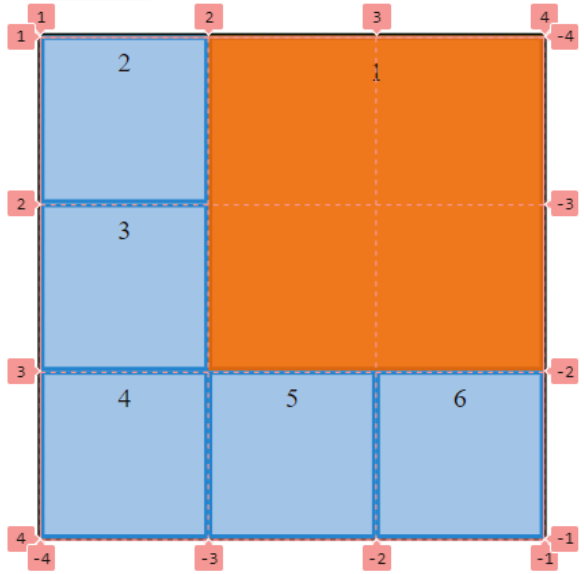
DÉCOUVERTE DU CSS ET APPLICATION

Grid : Le positionnement des éléments

grid-column : combine ***grid-column-start*** et ***grid-column-end***, définissant à la fois le début et la fin sur l'axe des colonnes dans une grille.

grid-row : combine ***grid-row-start*** et ***grid-row-end***, spécifiant à la fois le début et la fin sur l'axe des lignes dans une grille

```
#item1{
  background-color: #EE740F;
  border: 3px solid #d66102;
  grid-column: 2 / 4 ;
  grid-row: 1  / 3;
}
```



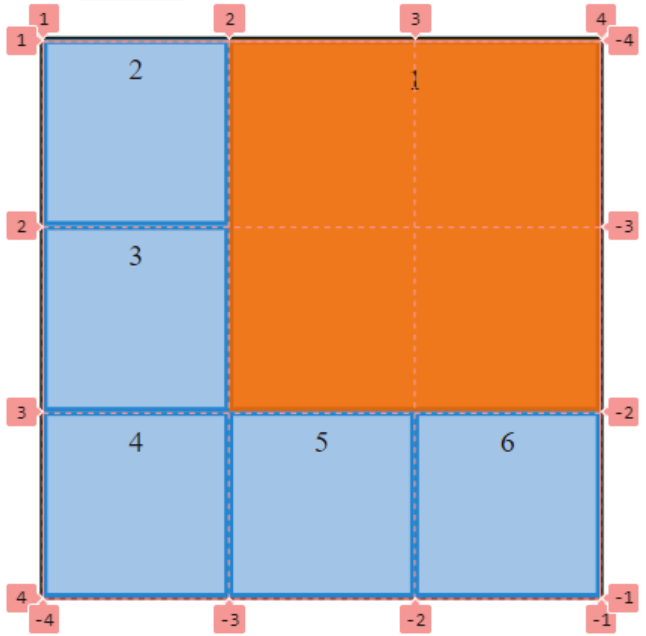
DÉCOUVERTE DU CSS ET APPLICATION

Grid : Le positionnement des éléments

grid-area : Propriété abrégée permettant de définir à la fois la ligne de départ, la colonne de départ, la ligne de fin et la colonne de fin d'un élément dans une grille.

Elle combine dans l'ordre ***grid-row-start***, ***grid-column-start***, ***grid-row-end*** et ***grid-column-end*** en une seule déclaration.

```
#item1{
  background-color: #EE740F;
  border: 3px solid #d66102;
  grid-area: 1/ 2 / 3 /4 ;
}
```

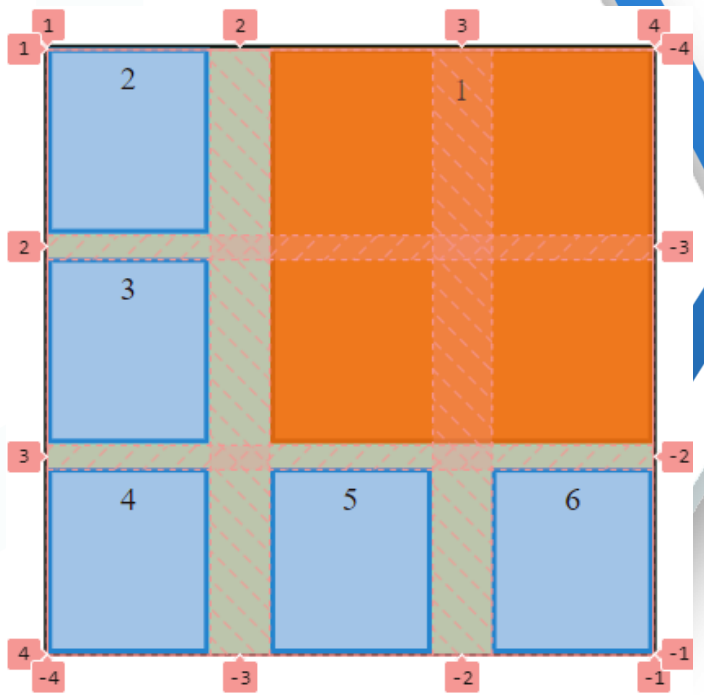


DÉCOUVERTE DU CSS ET APPLICATION

Grid : L'espacement des éléments

la propriété ***gap*** en CSS Grid permet de définir l'espacement entre les lignes et les colonnes à l'intérieur d'une grille.
gap peut prendre une valeur unique pour définir un espacement uniforme entre les lignes et les colonnes, ou bien deux valeurs pour définir un espacement différent entre les lignes et les colonnes.

```
#container {
  display: grid;
  grid-template-columns: repeat(3, 1fr);
  grid-template-rows: repeat(3, 1fr);
  gap: 5% 10%;
}
```



DÉCOUVERTE DU CSS ET APPLICATION

Grid : Similarités avec flexbox

Les grids et les flexbox possèdent de propriété Similaire tel que :

justify-content :
Cette propriété est commune à Flexbox et Grid. Elle contrôle l'alignement des éléments le long de l'axe principal (horizontal en Flexbox et suivant les colonnes en Grid).

Align-content :
Cette propriété aligne les éléments le long de l'axe transversal (vertical en Flexbox et suivant les lignes en Grid).

place-content :
Combine à la fois ***align-content*** et **justify-content** en une seule déclaration. Elle permet de définir à la fois l'alignement vertical et horizontal des éléments d'une grille en CSS Grid.

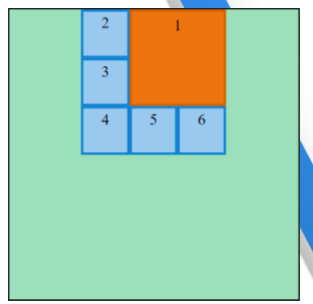
Peut accepter soit une valeur unique pour définir à la fois l'alignement vertical et horizontal, soit deux valeurs dans l'ordre, la première pour ***align-content*** (alignement vertical) et la seconde pour ***justify-content*** (alignement horizontal).

DÉCOUVERTE DU CSS ET APPLICATION

Grid : Similarités avec flexbox

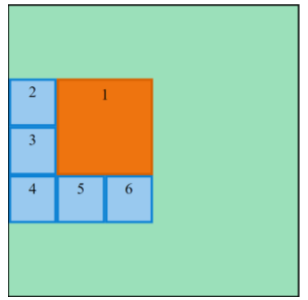
Justify-content :

```
#container {
   display: grid;
   grid-template-columns: repeat(3, 50px);
   grid-template-rows: repeat(3, 50px);
   justify-content: center;
}
```



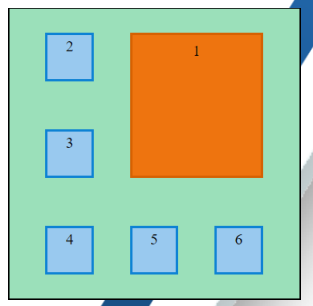
Align-content :

```
#container {
   display: grid;
   grid-template-columns: repeat(3, 50px);
   grid-template-rows: repeat(3, 50px);
   align-content: center;
}
```



Place-content :

```
#container {
   display: grid;
   grid-template-columns: repeat(3, 50px);
   grid-template-rows: repeat(3, 50px);
   place-content: space-around space-evenly;
}
```



DÉCOUVERTE DU CSS ET APPLICATION

Grid : Similarités avec flexbox

align-items :
Cette propriété s'applique aux éléments enfants de la grille et contrôle leur alignement le long de l'axe transversal (vertical par défaut). Elle peut prendre des valeurs telles que **start**, **end** , **center**, **stretch** , pour aligner les éléments individuels dans leur cellule de grille en fonction de l'axe des lignes.

justify-items :
De manière similaire à **align-items**, cette propriété s'applique également aux éléments enfants de la grille, mais elle contrôle leur alignement le long de l'axe principal (horizontal par défaut). Elle prend les mêmes valeurs pour aligner les éléments individuels dans leur cellule de grille en fonction de l'axe des colonnes.

place-items **:**
Combine à la fois **align-items** et **justify-items** en une seule déclaration. Elle permet de définir à la fois l'alignement vertical et horizontal des éléments enfants d'une grille.

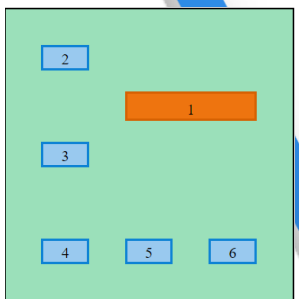
Peut accepter soit une seule valeur pour définir à la fois l'alignement vertical et horizontal des éléments enfants d'une grille, soit deux valeurs dans l'ordre pour align-items (alignement vertical) et pour justify-items (alignement horizontal).

DÉCOUVERTE DU CSS ET APPLICATION

Grid : Similarités avec flexbox

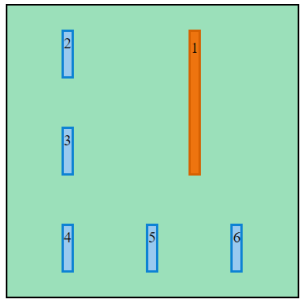
Align-items :

```
 #container {
    display: grid;
    grid-template-columns: repeat(3, 50px);
    grid-template-rows: repeat(3, 50px);
    place-content: space-around space-evenly;
    align-items: center;
}
```



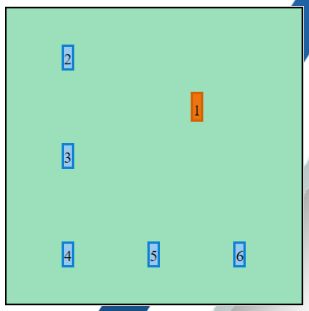
Justify-items:

```
#container {
   display: grid;
   grid-template-columns: repeat(3, 50px);
   grid-template-rows: repeat(3, 50px);
   justify-items: center;
}
```



Place-items :

```
#container {
   display: grid;
   grid-template-columns: repeat(3, 50px);
   grid-template-rows: repeat(3, 50px);
   place-items: center;
}
```



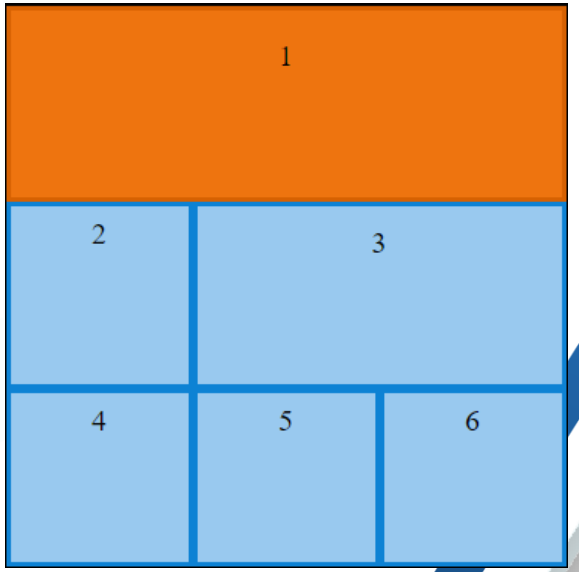
DÉCOUVERTE DU CSS ET APPLICATION

Grid : grid-template-areas

La propriété ***grid-template-areas*** consiste à attribuer des noms (sous forme de chaînes de caractères) à chaque cellule de la grille pour créer une structure visuelle et identifiable.

Vous pouvez ensuite assigner ces noms à vos éléments enfants de la grille à l'aide de la propriété ***grid-area***, ce qui leur permettra d'occuper les cellules correspondantes.

```
#container {
   display: grid;
   grid-template-areas: "head head head "
                        "nav article article"
                        "foot1  foot2 foot3";
}
#item1{
  background-color: #EE740F;
  border: 3px solid #d66102;
  grid-area: head;
}
 /* assignation des autres items */
```



DÉCOUVERTE DU CSS ET APPLICATION

Grid : grid-template-areas cellule vide

Dans ***grid-template-areas***, le point « **.** » est utilisé pour représenter une cellule vide ou non définie dans la structure de la grille

```
#container {
   display: grid;
   grid-template-areas: "head head head "
                        "nav . article"
                        "foot1  foot2 foot3";
}
#item1{
   background-color: #EE740F;
   border: 3px solid #d66102;
   grid-area: head;
}
/* assignation des autres items */
```