Afpa

se former, avancer

Feuille de Style CSS

Mettre du style à vos pages Web

AFPA CENTRE DE POMPEY

Historique

Feuille de Style

```
p {
  font-size: 16px;
  color: #333;
}
```

CSS est développé par niveaux imbriqués et non pas versions successives.

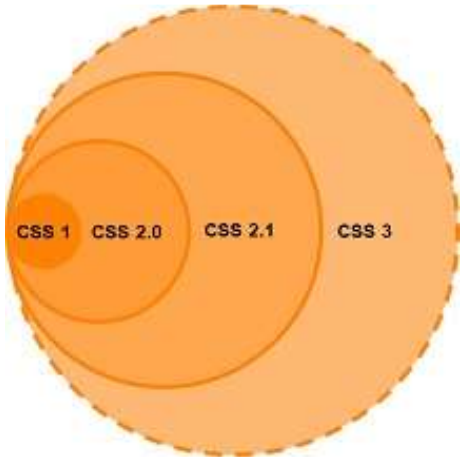
Style

Sélecteur

Mise en Page

> CSS, créé par W3C dans les années 1990, pris en charge totalement par les navigateurs dans les années 2000.

Imbrications des anciennes versions



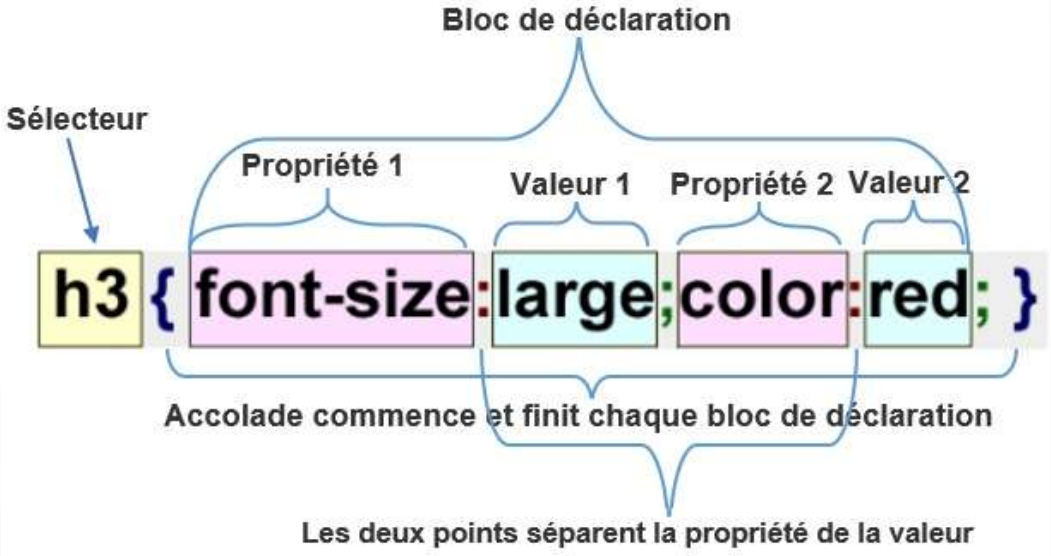
DÉFINITION

CSS : Cascading Style Sheet

- Langage de définition de propriétés de format et de présentation au contenu des éléments définis par des balises HTML.

Les styles peuvent s'appliquer :

- Balise par balise.
- De manière commune à toutes (ou certaines) balises d'un type donné.
- Autant de fois que nécessaire, sans avoir à le redéfinir à chaque usage.
- À l'intérieur d'une page HTML mais peuvent (doivent) être définis dans un fichier externe qui sera associé aux pages HTML : **Charte graphique**



Intégration dans HTML

Il existe 3 méthodes pour utiliser et intégrer les feuilles de style dans un document HTML

- Intra-ligne
- Globale
- Importée (conseillée)

1

CSS Intra-lignes :
Insertion dans la balise qu'elle définit

```
<h1 style="font-family: Arial; font-style: italic">
    Formulaire de Saisie</h1>
```

2

CSS Globale :
À l'intérieur des balises HEAD

```
<style type="text/css">
    h1 {
        font-family: Arial;
        font-style: italic;
    }
</style>
```

3

CSS importé (conseillée)
À l'aide de la balise link

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

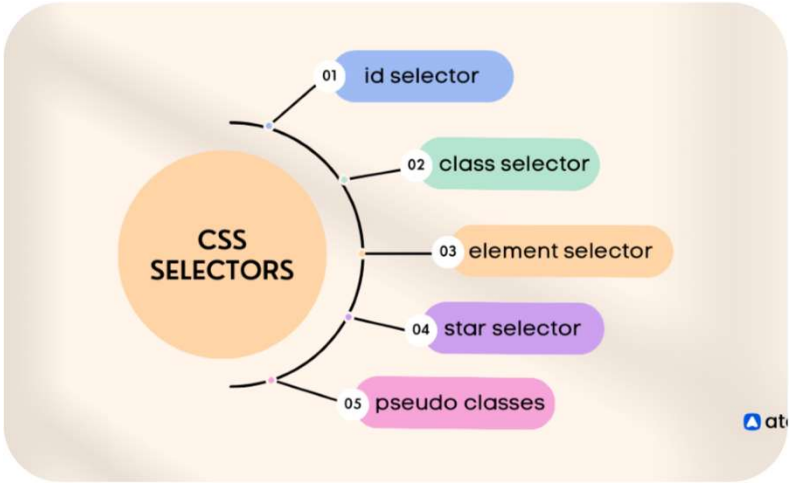
ressources > style.css > h1

```
1   h1 {
2       font-family: Arial;
3       font-style: italic;
4   }
```

SELECTEURS CSS

Il existe toute une combinaison de sélecteurs en CSS

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



Cible une balise elem { propriétés : valeur; }	Sélecteur de type
Cible un ensemble de balises elem1, elem2, … { propriétés : valeur; }	Sélecteurs regroupés
Cible l'ensemble du document HTML * { propriétés : valeur; }	Sélecteur universel
Cible les éléments désignés par l'attribut id : #demo { … }	Sélecteur d'identifiant
Cible les éléments désignés par l'attribut class : .demo { … }	Sélecteur de classe
Contextuel : elem propriété { … } Descendants : elemA elemB { … }	Sélecteur contextuels ou descendants

Sélecteurs d'attributs

Les sélecteurs d'attributs permettent de cibler un élément selon la présence d'un attribut ou selon la valeur donnée d'un attribut.

Sélecteurs d'attribut simple

a[target] { … }
a[href] [title] { … }

Sélecteurs de valeur d'attribut exacte

p[lang=fr] { … }
a[href="https…"][title="MyHome"] { … }

https://developer.mozilla.org/fr/docs/Web/CSS/Reference/Selectors/Attribute_selectors

Sélecteurs d'attributs

Il existe donc plusieurs types de sélecteurs d'attributs

Sélecteurs de valeur d'attribut de correspondance partielle

[element~="web"]

- Sélectionne n'importe quel élément avec un attribut élément dont la valeur contient le mot web dans une liste séparée par des espaces de mots

[element*="web"]

- Sélectionne n'importe quel élément avec un attribut élément dont la valeur contient la sous-chaîne web

[element^="web"]

- Sélectionne n'importe quel élément avec un attribut élément dont la valeur commence par web

[element$="web"]

- Sélectionne n'importe quel élément avec un attribut élément dont la valeur se termine par web

[element|="web"]

- Sélectionne tout élément avec un attribut élément dont la valeur commence par web suivi d'un tiret (U+002D) ou dont la valeur est exactement égale à web

Les Combinateurs CSS

Les jokers

Afpa

>

Cible tous les enfants

```
div > p {
    background-color: purple;
    color: white;
}
```

+

Cible le voisin direct

```
div + p {
    background-color: purple;
    color: white;
}
```

~

Cible tous les voisins directs

```
div ~ p {
    background-color: purple;
    color: white;
}
```

LES PSEUDO-CLASSES

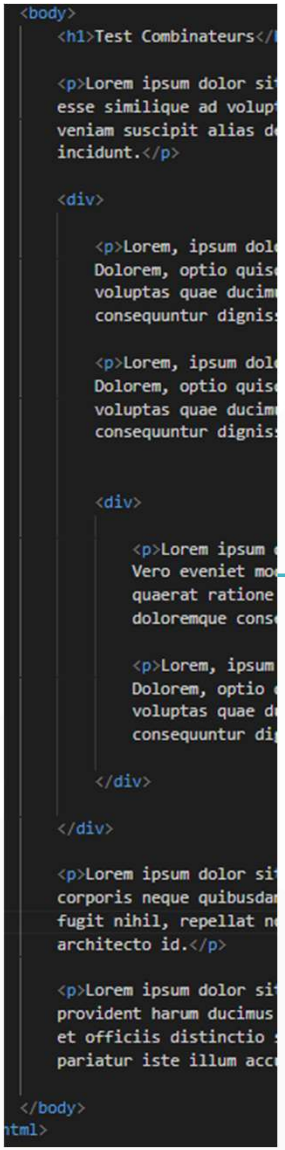
Une pseudo-classe est un mot clé préfixé par deux points qui s'ajoute au sélecteur pour appliquer un style.

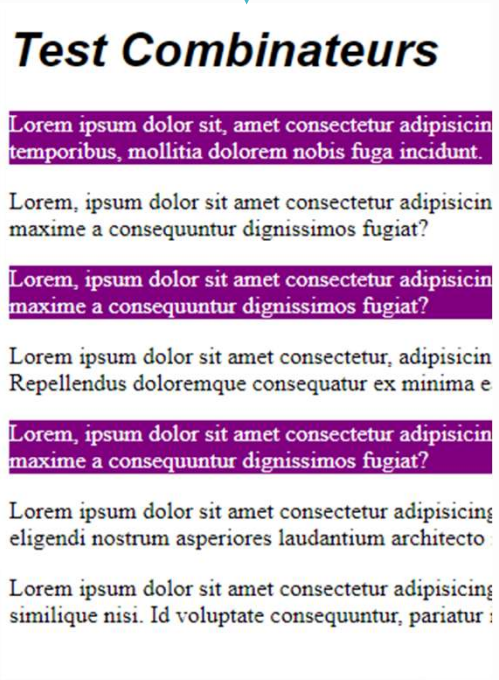
Les pseudo classes sont des classes fantômes qui correspondent à un état des éléments dans le document HTML.

Selecteur:pseudo-classe { …. }

Voici une liste exhaustive des pseudo-classes :

- Pseudo-classe :link — lien non visité
- Pseudo-classe :visited — lien visité
- Pseudo-classe :active — lien lors d'un clic
- Pseudo-classe :hover — survol souris
- Pseudo-classe :focus — focus actif
- Pseudo-classe :first-child — 1er enfant
- Pseudo-classe :nth-child(n) — n-ième enfant
- …

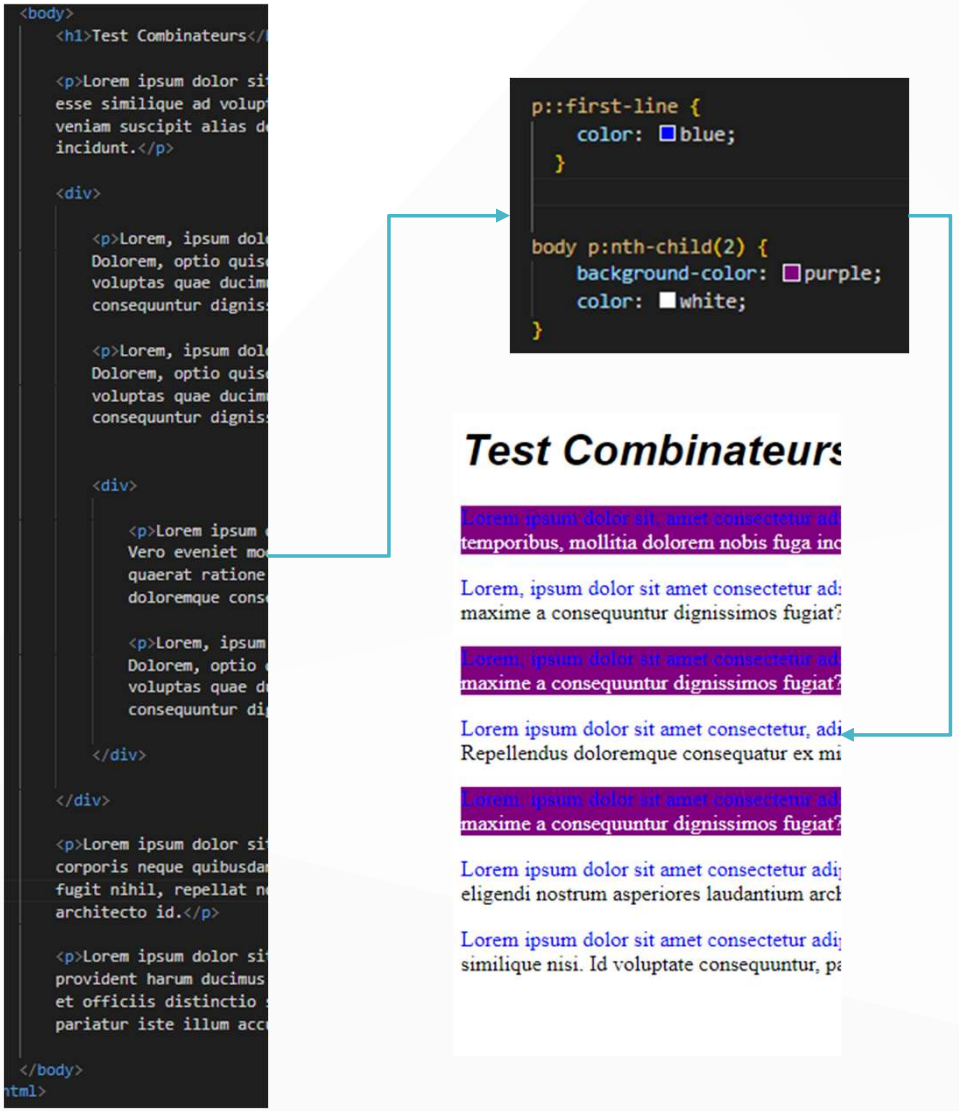




LES PSEUDO-ÉLÉMENTS

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

- ::after (:after)
- ::before (:before)
- ::cue (:cue)
- ::first-letter (:first-letter)
- ::first-line (:first-line)
- ::selection
- ::slotted()
- ::spelling-error



UNITÉS DE MESURE EN CSS

Il existe plusieurs unités de mesure en CSS :

- Unités relatives
  - Ne représente pas une longueur prédéfinie mais une longueur relative à une référence
- Unités absolues
  - Applicables sur toutes les propriétés nécessitant une valeur correspondant à une distance
- Unités d'angles
  - Utilisable pour définir des transformations
- Unités de temps
  - Utilisable pour les transitions et les animations

Les unités relatives :

- % représente une valeur en %. La référence peut être le parent, l'élément lui-même ou tout autre valeur.
- em représente une proportion de la taille de la police courante. Affecte la taille de police du parent. Permet d'avoir des feuilles de style adaptables d'un média à l'autre.
- ex Représente la hauteur du glyphe 'x' (ou équivalent) dans la police courante. Exprime la hauteur des caractères.
- ch Représente la dimension du glyphe '0'
- rem Représente la taille de la police de l'élément html.
- fr Représente une fraction de l'espace restant.
- dpi Représente la résolution du périphérique.
- vw Représente la largeur du viewport du périphérique.
- vh Représente la hauteur du viewport.
- vm Représente la plus petite valeur du viewport (largeur ou hauteur).

https://www.w3.org/Style/Examples/007/units.fr.html

UNITÉS DE MESURE EN CSS

Les unités absolues

Ces unités sont applicables sur toutes les propriétés nécessitant une valeur correspondant à une distance.

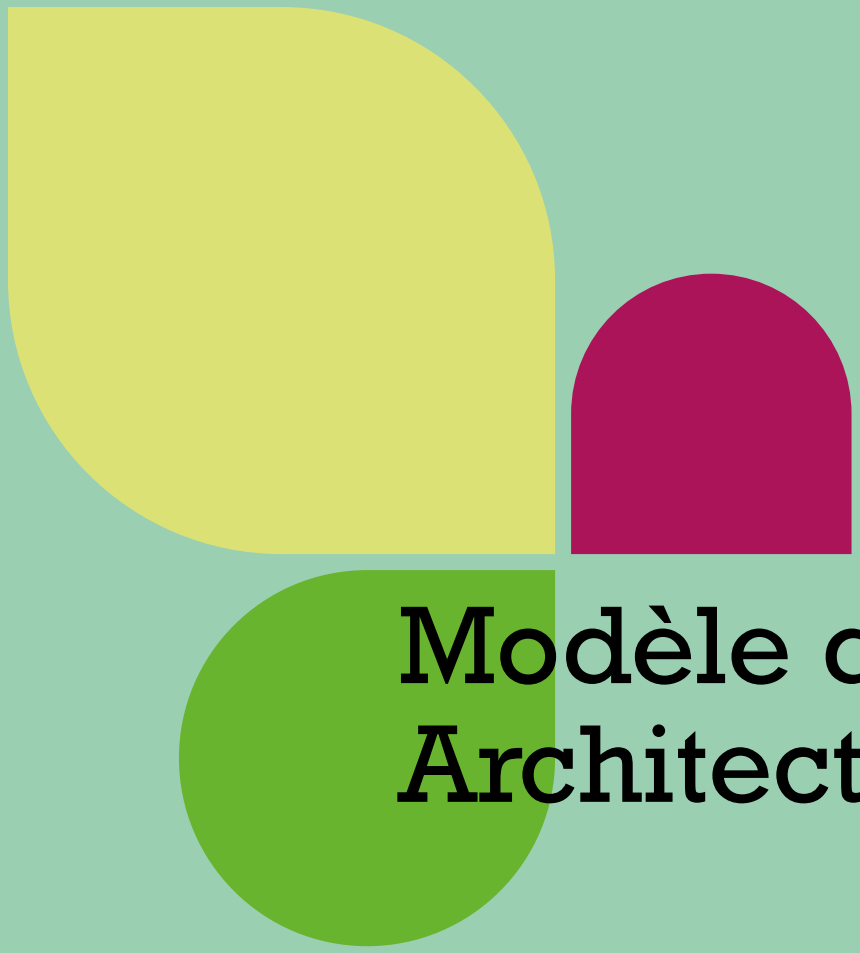
- cm centimètre
- mm millimètre
- in pouce, c'est environ 2,54 cm
- px pixel, initialement 1 px est égal à 1/96 ème de pouce
- pt point, initialement 1 pt est égal à 1/72 ème de pouce
- pc pica, 1 pc est égal 12 pt

Les unités d'angles et de temps

Nous les utiliserons notamment pour définir des transformations et/ou animations.

- deg degré, 1 tour = 360 deg
- grad grade, 1 tour = 400 grad
- rad radian, 1 tour = 2π =6,28 rad
- turn tour, 0,5 tour = 180 deg
- s seconde
- ms milliseconde,1 000 ms = 1s

https://developer.mozilla.org/fr/docs/Learn/CSS/Building_blocks/Values_and_units

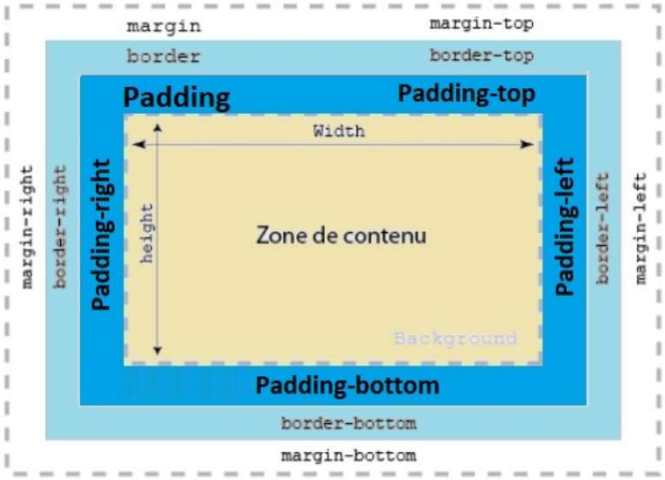


Modèle de boîtes
Architecture de boîtes

Définition

Le Box Model est la manière dont les navigateurs doivent afficher les boites CSS.

- On parle de box Model pour parler de design et mise en page.

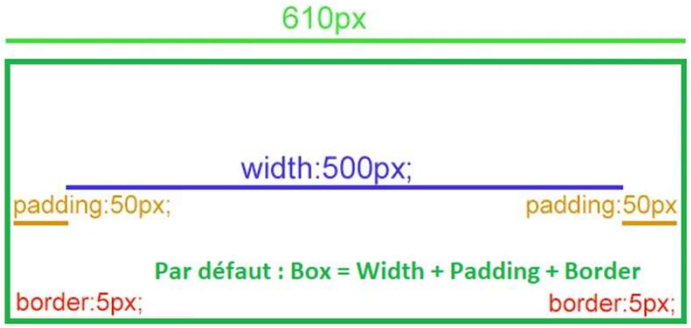


Tous les éléments HTML peuvent être considérés comme des boîtes.

- Elle comprend donc :

- Une marge externe (margin).
- Une bordure (border).
- Une marge interne (padding).
- Une zone de contenu avec une largeur et une hauteur.

Par défaut, la largeur totale de la boîte correspond à la somme de sa largeur + son padding + sa bordure.



Positionnement en Flux normal

Le positionnement en flux normal correspond à la mise en forme résultat de l'interprétation au fur et à mesure de la transcription des balises utilisées dans la page HTML.

- Boîte de type **bloc** en flux normal
  - 100 % de la largeur

- Boîte de type **inline** en flux normal
  - Se positionne les uns derrières les autres dans le flux

Les principaux éléments de type bloc

- l'élément div ;
- les titres h1, h2, h3, h4, h5, h6 ;
- le paragraphe p ;
- Les listes et éléments de liste ul, ol, li, dl, dd
- Le bloc de citation blockquote ;
- Le texte pré-formaté pre ;
- L'adresse address.

BLOCK

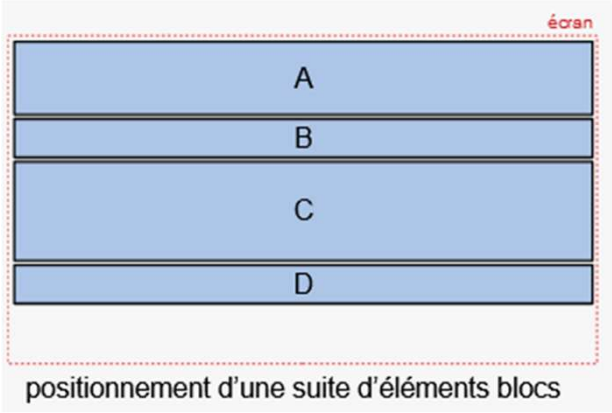
Un élément block (p, h*, ul, li, div par exemple) peut contenir n'importe quel élément ainsi que du texte.

Positionnement

prennent toute la largeur, peu importe leur contenu, et donc fort logiquement s'empilent sans jamais être côte-à-côte.

Dimension

Déterminées par son contenu. La boîte du bloc parent s'étire pour épouser les contours du contenu.



positionnement d'une suite d'éléments blocs

inline

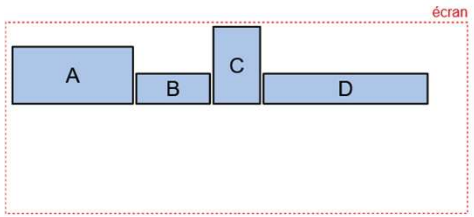
Un élément inline (b, i, u, a, img par exemple) ne peut contenir que du texte et des éléments de type inline.

Positionnement

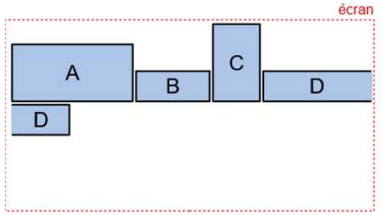
En ligne, dicté par le contenu. Sur la même ligne vers la droite dans l'ordre du flux, collés les uns aux autres.

Dimension

La largeur, ne dépend que du contenu textuel dans les limites de la largeur du navigateur, auquel cas un retour à la ligne est provoqué. La hauteur déterminée par la police définie dans un de ces ancêtres ou à défaut définie dans le navigateur.



positionnement d'une suite d'éléments en ligne



positionnement d'une suite d'éléments en ligne avec retour à la ligne

Les propriétés d'affichages

Pour choisir le type d'affichage et de positionnement des éléments HTML, le CSS va nous fournir des propriétés très puissantes qui vont nous permettre de modifier le flux normal de la page, c'est-à-dire de modifier l'ordre d'affichage des éléments ou la place réservée par défaut à chacun d'entre eux.

- La propriété display va nous permettre de définir un type d'affichage pour un élément.
- La propriété position va nous permettre de positionner nos éléments de différentes façons dans une page.
- La propriété float va nous permettre de faire « flotter » des éléments HTML dans la page.

Propriété DISPLAY

La propriété Display redéfinit le type d'affichage utilisée pour le rendu d'un élément.

- *Il en existe d'autres comme table pour les tableaux, les listes d'éléments… par exemple.*
- Ces types conditionnent l'affichage sur deux aspects essentiels : le positionnement et la dimension d'une boîte

Block — Modifie le comportement de l'élément en type bloc

Inline — Modifie le comportement de l'élément en type inline

Inline-block — Modifie le comportement de l'élément en type inline-bloc (possibilité de largeur et hauteur)

Grid — Modifie le comportement du parent mais aussi des enfants

Flex — Modifie le comportement du parent mais aussi des enfants

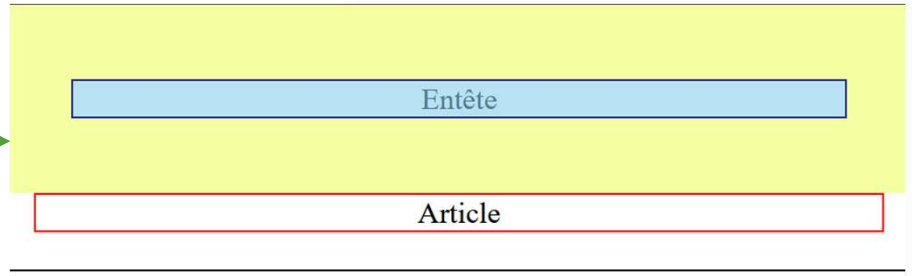
FUSION DES MARGES

FUSION DES MARGES

La fusion des marges extérieures (margin) fait que deux marges contiguës, dans le cas où deux éléments se suivent ou deux éléments sont imbriqués, s'affichent comme un espacement du maximum des deux marges.

```
<style>
    body {
        margin: 0;
        border: 1px solid black;
    }
    header {
        margin: 40px;
        border: 1px solid blue;
        text-align: center;
    }
    article {
        margin: 20px;
        border: 1px solid red;
        text-align: center;
    }
</style>
```

```
<header>Entête</header>
<article>Article</article>
```



L'entête a bien 40 pixels tout autour et l'article suit immédiatement, c'est-à-dire que la marge haute de 20 pixels de l'article est absorbée par la marge basse de l'entête (on n'a pas 60 pixels d'écart).

propriété	en ligne	bloc
marges hautes et basses	NON	OUI
marges droites et gauches	OUI	OUI

LARGEUR ET HAUTEUR DU CONTENU

LARGEUR ET HAUTEUR

Les dimensions d'une boîte, en largeur et hauteur, dépendent de son contenu et correspondent au comportement associé à la valeur par défaut auto.

- Dans tous les cas la boîte s'étire pour épouser les formes de son contenu une fois celui-ci affiché.

A noter que les éléments sans contenu ont des comportements de dimension particuliers, comme l'image par exemple qui possède une dimension intrinsèque.

propriété	en ligne avec contenant	bloc
largeur	Le contenu textuel s'étale au maximum vers la droite puis vers le bas, limité en général par une largeur fixée pour un parent bloc ou par la fenêtre du navigateur. Il n'est pas possible d'en fixer la valeur.	En automatique, **la boîte occupe toute la largeur** (équivalent à width:100%) La largeur peut être fixée.
hauteur	La hauteur est aussi déterminée par le contenu textuel qui s'étire vers le bas selon la largeur du parent (élément avec largeur fixé ou fenêtre du navigateur) et n'inclue pas les marges intérieures et les bordures. Il n'est pas possible d'en fixer la valeur.	En automatique, la hauteur épouse le contenu de la boîte, y compris les marges et les bordures. La hauteur peut être fixée.

QUE SE PASSE-T-IL SI LA LARGEUR OU LA HAUTEUR FIXÉ POUR UN ÉLÉMENT EST INFÉRIEURE À CELLE DE SON CONTENU ?

Il y a débordement, le contenu est toujours affiché, débordant de sa boîte. Toutefois la boîte garde la taille qui lui est fixée, en particulier sa bordure s'affiche selon cette taille. Mais les éléments qui suivent la boîte dans le flux HTML seront poussés par le contenu qui déborde.

C'est la propriété CSS overflow qui détermine le rendu visuel de ce débordement.

Propriété	Description	Valeurs possibles	
Overflow	Comportement si le contenu déborde de sa boîte	Visible	Le contenu est affiché et déborde de son conteneur
		Hidden	Tout ce qui déborder est effacé et n'affecte pas l'affichage
		scroll	Ce qui déborde est masqué mais accessible par un ascenseur affiché en toute circonstance
		auto	Comme scroll mais l'ascenseur n'est affiché que s'il y a du contenu masqué

POSITIONNEMENT DES ÉLÉMENTS

position

Positionnement des éléments

Ces différents positionnements des éléments vous donnent la possibilité de mieux gérer l'apparence de vos pages Web en fonction de l'écran (notion de responsive design).

Position

La propriété position : valeur permet d'agir sur le positionnement de l'élément.

Absolute

Positionnement absolu, mesuré à partir du bord de l'élément parent. Peut défiler.

fixed

Positionnement absolu, mesuré à partir du bord de l'élément parent. Reste fixe lors du défilement

Relative

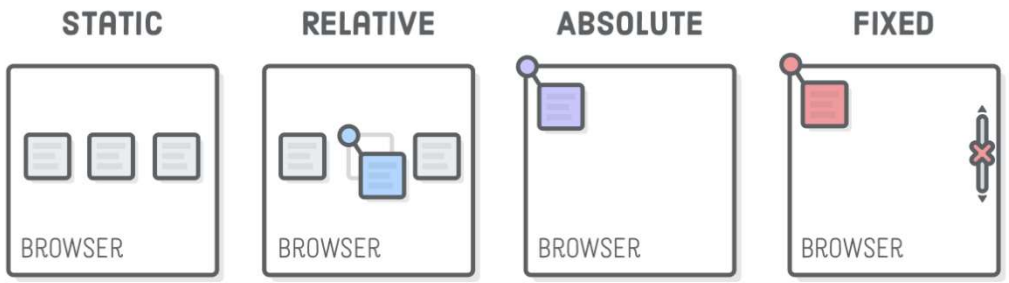
Positionnement relatif mesuré à partir de la position de départ de l'élément proprement dit.

Static

Pas de positionnement spécial, flux normal de l'élément (**réglage par défaut**)

Positionnement

Float / Absolu / fixe



Positionnement Flottant propriété float

La position float retire une boîte du flux normal pour la placer le plus à droite (float: right) ou le plus gauche (float: left) possible dans son conteneur.

Le contenu suivant cette boîte flottante s'écoule le long de celle-ci, dans l'espace laissé libre.

Ce mécanisme est bien connu en traitement de texte pour écrire du texte autour d'une image.

Positionnement absolu ou fixe

Une boîte en positionnement absolu peut être placée n'importe-où dans le code HTML et s'afficher à l'endroit de votre choix.

- Le positionnement absolu « retire » totalement du flux le contenu concerné : sa position est déterminée par référence aux limites du conteneur.

La position fixe est une spécialité de la position absolue. Le contenu est retiré du flux. A utiliser avec précaution car fige le contenu.

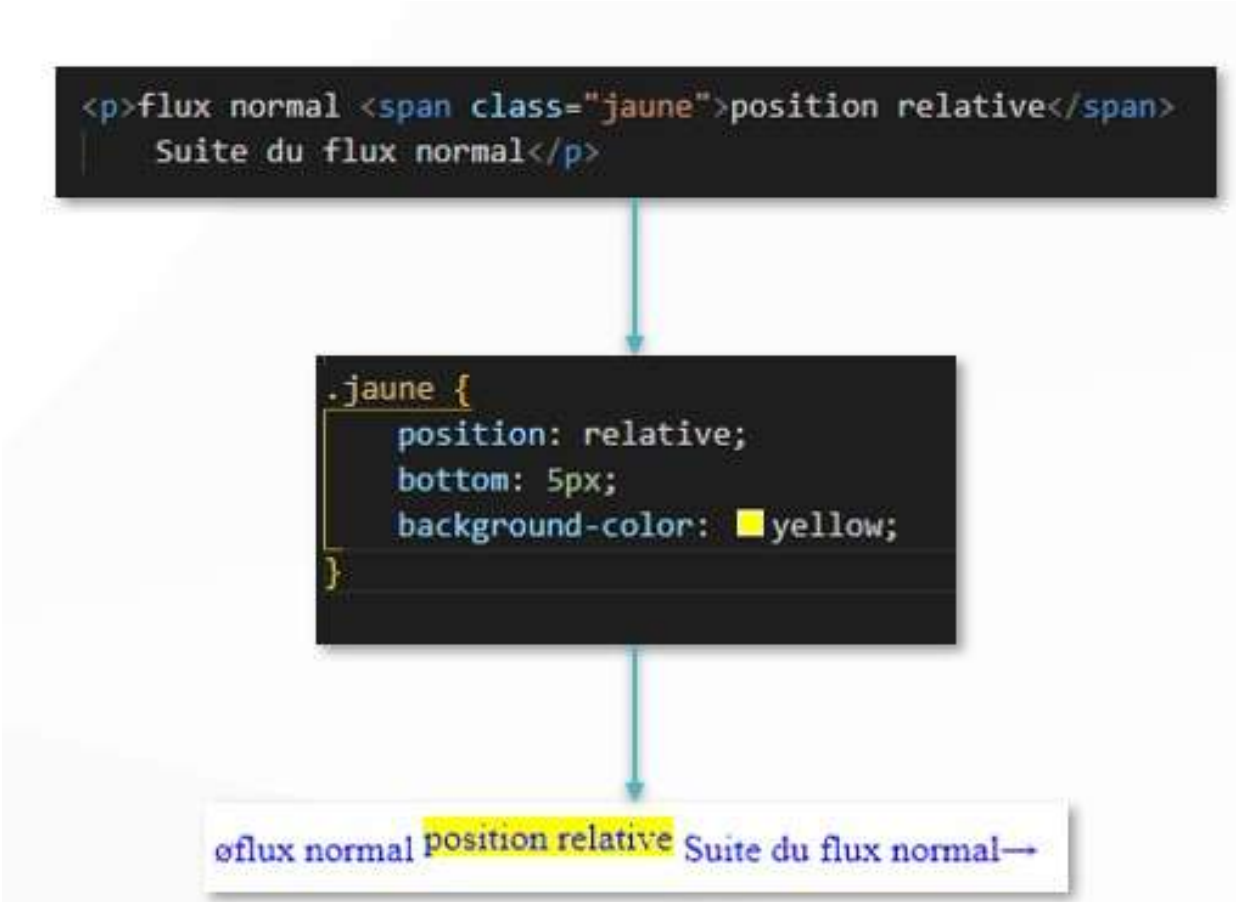
- La propriétés z-index permet de jouer sur l'ordre dans le flux. L'élément le plus élevé est affiché au-dessus des autres.

Positionnement

en Flux relatif

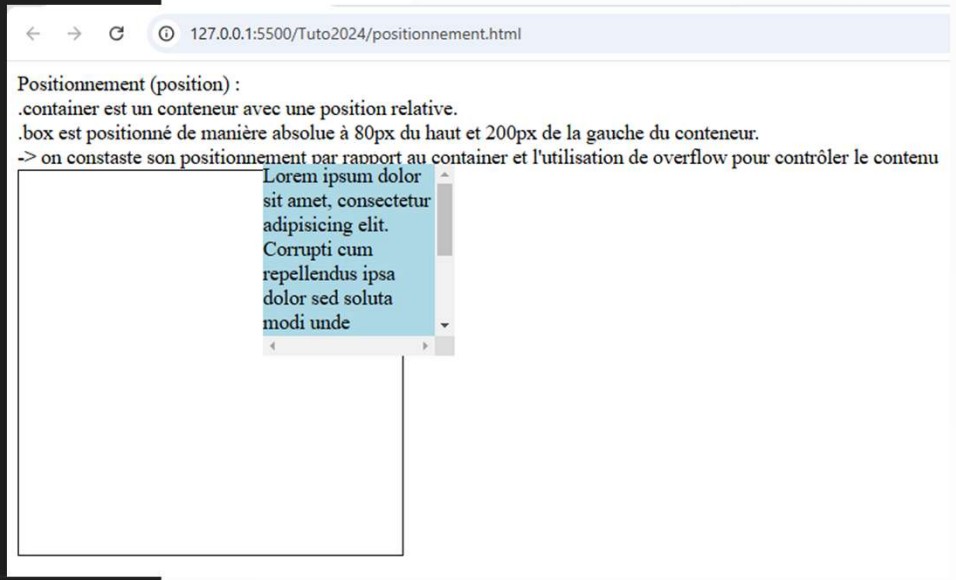
Le positionnement relatif permet d'inscrire un contenu en flux normal, puis de le décaler horizontalement et/ou verticalement.

Le contenu suivant n'est pas affecté par ce déplacement. Cela peut donc entraîner des chevauchements.



Exemple de position relative

```html
<!DOCTYPE html>
<html lang="fr">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Exemple de Positionnement</title>
    <style>
        .container {
            position: relative;
            width: 300px;
            height: 300px;
            border: 1px solid black;
        }

        .box {
            position: fixed;
            top: 80px;
            left: 200px;
            width: 150px;
            height: 150px;
            background-color: lightblue;
            overflow: scroll;
        }
    </style>
</head>

<body>
    <section>
        Positionnement (position) : <br>
        .container est un conteneur avec une position relative.<br>
        .box est positionné de manière absolue à 80px du haut et 200px de la gauche du conteneur.<br>

        -> on constaste son positionnement par rapport au container et l'utilisation de overflow pour
        contrôler le contenu
    </section>


    <div class="container">
        <div class="box">
            Lorem ipsum dolor sit amet, consectetur adipisicing elit. Corrupti cum
            repellendus ipsa dolor sed soluta modi unde perferendis error, animi omnis, velit expedita,
            nostrum laboriosam adipisci aut a reprehenderit cumque.
        </div>
    </div>
</body>
</html>
```



GRID

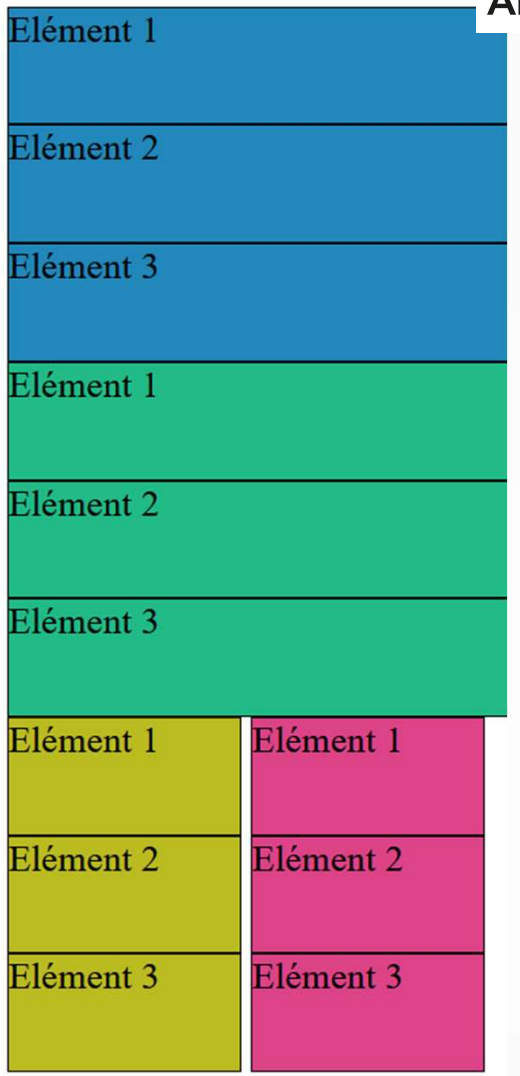
Utilisation des grilles

GRID

Le modèle de grille :

- le modèle de disposition CSS puissant.
- précieux pour aligner les contenus selon les deux dimensions horizontale et verticale.
- Pour définir un conteneur de grille, on va appliquer à un élément soit :
  - display : grid
    - Grille de niveau bloc
    - Occupera tous l'espace de son parent
  - display : inline-grid
    - Grille de niveau inline
    - Occupera l'espace nécessaire à son contenu

```
    <style>
        .container{
            display: grid;
        }

        .inline-container{
            display: inline-grid;
        }

        .item{
            border: 1px solid black;
            min-width: 100px;
            min-height: 50px;
        }

        .bleu{background-color: #28B;}
        .vert{background-color: #2B8;}
        .jaune{background-color: #BB2;}
        .rouge{background-color: #D48;}
    </style>


</head>
<body>
    <div class="container">
        <div class="item bleu">Elément 1</div>
        <div class="item bleu">Elément 2</div>
        <div class="item bleu">Elément 3</div>
    </div>
    <div class="container">
        <div class="item vert">Elément 1</div>
        <div class="item vert">Elément 2</div>
        <div class="item vert">Elément 3</div>
    </div>
    <div class="inline-container">
        <div class="item jaune">Elément 1</div>
        <div class="item jaune">Elément 2</div>
        <div class="item jaune">Elément 3</div>
    </div>
    <div class="inline-container">
        <div class="item rouge">Elément 1</div>
        <div class="item rouge">Elément 2</div>
        <div class="item rouge">Elément 3</div>
    </div>
</body>
```



VOCABULAIRE DES GRILLES

Une grille est composée

- de colonnes et de rangées,
- de lignes imaginaires qui définissent les cellules,
- d'espaces entre deux lignes adjacentes qu'on appelle piste de grilles (soit une ligne ou une colonne de notre grille)
- D'un espace délimité par deux lignes de colonnes et deux lignes de rangées pas forcement adjacentes : zone de grille

DÉFINIR ET POSITIONNER NOS GRILLES

grid-template-columns et grid-template-rows

On peut définir :

- le nombre et la taille de chacune de ces colonnes et rangées.

On va pouvoir passer des valeurs :

- de type longueur (en px par exemple)
- pourcentage %,
- fraction fr
- la valeur auto

grid-column-start, grid-column-end, grid-row-start, grid-column-end et grid-column, grid-row

Le modèle des grilles est un modèle de positionnement bidimensionnel

- selon l'axe horizontal et selon l'axe vertical.

Par défaut, les éléments de grille n'occupent qu'une colonne et qu'une ligne et se placent dans l'ordre de leur écriture.

ALIGNEMENT LES ÉLÉMENTS DE GRILLE

justify-items et justify-self

- aligner les éléments le long de l'axe inline du document.

start
- Aligné contre le début de la zone

end
- Aligné contre la fin de la zone

center
- Centré dans leur zone

stretch
- (défaut) occupe toute la zone

align-items et align-self

- aligner les éléments de grille selon l'axe de bloc ou axe des colonnes (axe vertical) grâce aux propriétés align-items et align-self.

start
- Aligné contre le début de la zone

end
- Aligné contre la fin de la zone

center
- Centré dans leur zone

stretch
- (défaut) occupe toute la zone

ALIGNEMENT LES ÉLÉMENTS DE GRILLE

place-items et place-self

Propriétés raccourcies qui nous permettent d'aligner un ou les éléments de grille par rapport aux deux axes d'un coup : place-items et place-self.

On va pouvoir leur passer une première valeur qui va définir le comportement de align-{items|self} et une deuxième qui définira le comportement de justify-{items|self}.

On peut également ne préciser qu'une valeur qui sera alors utilisée pour l'alignement sur les deux axes.

justify-content align-content

justify-content permet un alignement selon l'axe de ligne.

align-content permet un alignement selon l'axe de bloc.

start
- Aligné au début de son conteneur

end
- Aligné à la fin de son conteneur

center
- Centré dans son zone

stretch
- Redimensionne sur toute l'espace du conteneur

space-around
- équitablement l'espace extérieur entre les pistes

space-between
- équitablement l'espace entre les pistes sans marge contre le conteneur

space-evenly
- répartit équitablement l'espace (extérieur et intérieur) entre les différentes pistes.

RACCOURCIE

Apprendre en jouant

https://cssgridgarden.com/#fr

grid

La propriété grid est une propriété raccourcie qui permet de définir toutes les propriétés liées aux grilles CSS, qu'elles soient explicites (grid-template-rows, grid-template-columns et grid-template-areas), implicites (grid-auto-rows, grid-auto-columns et grid-auto-flow).

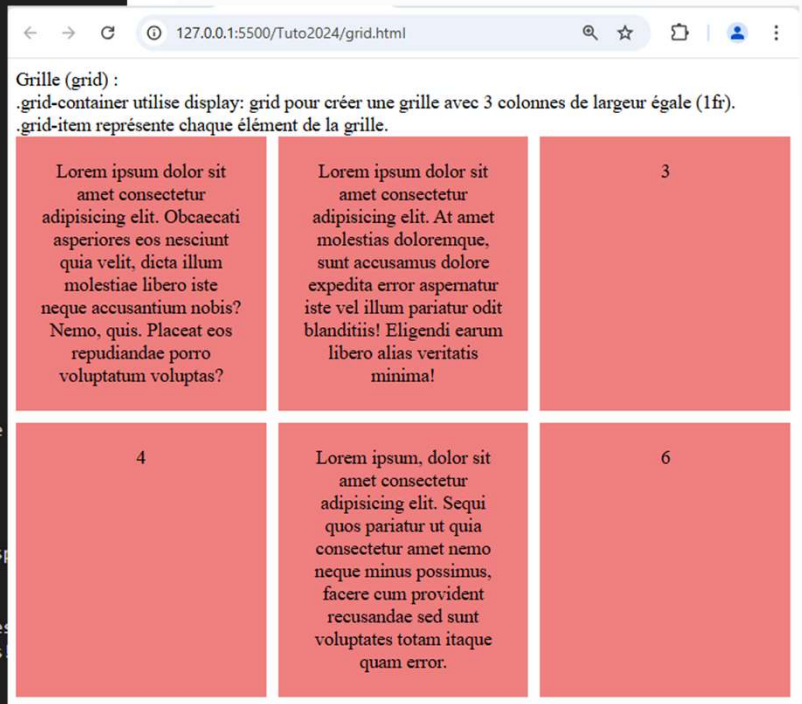
```
grid: auto-flow / 1fr 1fr 1fr;
grid: auto-flow dense / 40px 40
grid: repeat(3, 80px) / auto-flc
```

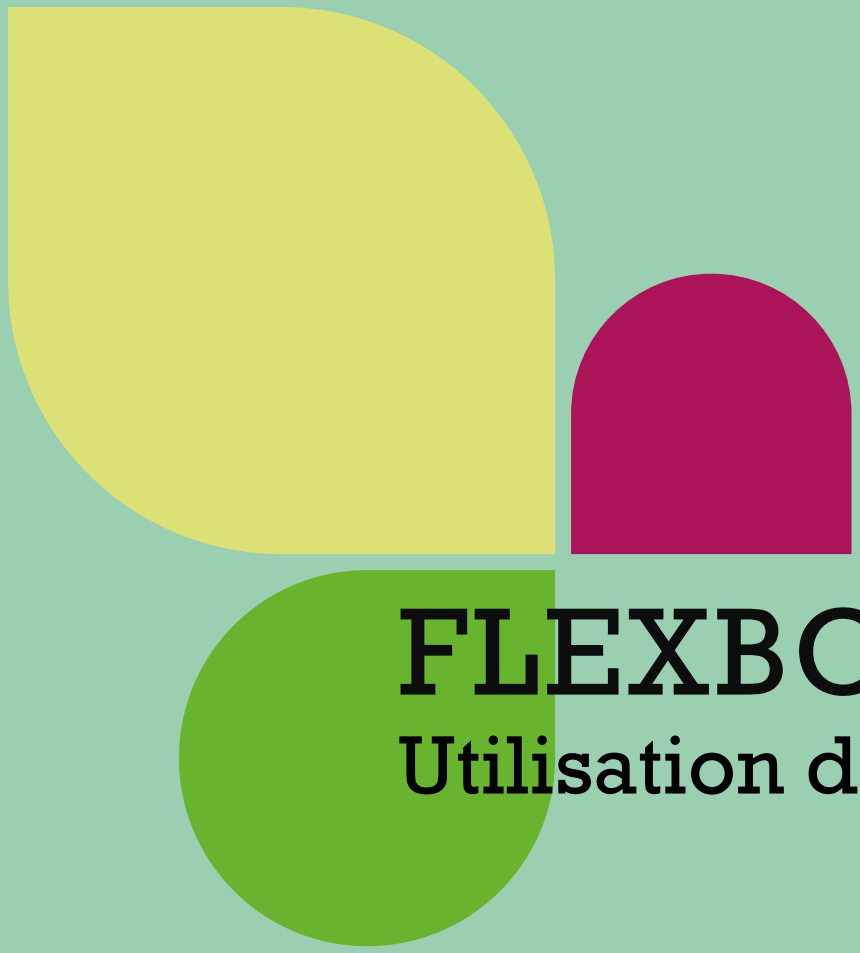
One Three Two

Grid Garden

Un exemple de GRID

```html
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Exemple de Grille</title>
    <style>
        .grid-container {
            display: grid;
            grid-template-columns: repeat(3, 1fr);
            grid-gap: 10px;
        }
        .grid-item {
            background-color: lightcoral;
            padding: 20px;
            text-align: center;
        }
    </style>
</head>
<body>


    <section>
        Grille (grid) :<br>
        .grid-container utilise display: grid pour créer une grille avec 3 colonnes de largeur égale
        .grid-item représente chaque élément de la grille.<br>
    </section>

    <div class="grid-container">
        <div class="grid-item">Lorem ipsum dolor sit amet consectetur adipisicing elit. Obcaecati asp
            quia velit, dicta illum molestiae libero iste neque accusantium nobis? Nemo, quis.
            Placeat eos repudiandae porro voluptatum voluptas?</div>
        <div class="grid-item">Lorem ipsum dolor sit amet consectetur adipisicing elit. At amet moles
            sunt accusamus dolore expedita error aspernatur iste vel illum pariatur odit blanditiis!
            earum libero alias veritatis minima!</div>
        <div class="grid-item">3</div>
        <div class="grid-item">4</div>
        <div class="grid-item">Lorem ipsum, dolor sit amet consectetur adipisicing elit. Sequi quos pariatur ut
            quia consectetur amet nemo neque minus possimus, facere cum provident recusandae
            sed sunt voluptates totam itaque quam error.</div>
        <div class="grid-item">6</div>
    </div>
</body>
</html>
```





FLEXBOX

Utilisation des boîtes flexibles

FLEXBOX

Le modèle Flexbox :

- modèle de disposition CSS puissant
- Permet de créer des dispositions complexes
- Un conteneur flexible composé d'éléments flexibles : la taille va s'adapter en fonction de l'espace disponible.
- Un contexte flexible crée un nouveau contexte d'empilement : les descendants directs deviendront immédiatement flexibles auxquels on va pouvoir appliquer les propriétés du flexbox.

Le flexbox est :

- à la différence du modèle des grilles, un modèle de disposition principalement unidimensionnel : nous allons devoir définir un axe principal et un axe secondaire et la majorité du positionnement et de l'alignement des éléments flexibles va se faire selon l'axe principal.

- Pour définir un conteneur flexible, on va appliquer à un élément soit :
  - display : flex
    - Conteneur flexible de niveau bloc
    - Occupera tous l'espace de son parent
  - display : inline-flex
    - Conteneur flexible de niveau inline
    - Occupera l'espace nécessaire à son contenu

EXEMPLE

Ici, on utilise 4 éléments div class="container" et div class="inline-container" qu'on définit comme conteneurs flexibles.

Les éléments à l'intérieur de ces conteneurs (les div class="item") deviennent de facto des éléments flexibles.

Les deux premiers conteneurs flexibles, de type bloc, occupent toute la largeur dans leur élément conteneur

les deux derniers, de niveau inline n'occupent que la place nécessaire à leur contenu et se placent côte-à-côte et non pas sur une nouvelle ligne.


```
    <style>
        .container{
            display: flex;
            padding: 10px;
            margin-bottom: 10px;
        }

        .inline-container{
            display: inline-flex;
            padding: 10px;
        }

        .bleu{background-color: #28B;}
        .vert{background-color: #2B8;}
        .jaune{background-color: #BB2;}
        .rouge{background-color: #E24;}

    </style>
</head>
<body>
    <div class="container rouge">
        <div class="item un bleu">Elément 1</div>
        <div class="item deux vert">Elément 2</div>
        <div class="item trois jaune">Elément 3</div>
    </div>
    <div class="container rouge">
        <div class="item un bleu">Elément 1</div>
        <div class="item deux vert">Elément 2</div>
        <div class="item trois jaune">Elément 3</div>
    </div>
    <div class="inline-container rouge">
        <div class="item un bleu">Elément 1</div>
        <div class="item deux vert">Elément 2</div>
        <div class="item trois jaune">Elément 3</div>
    </div>
    <div class="inline-container rouge">
        <div class="item un bleu">Elément 1</div>
        <div class="item deux vert">Elément 2</div>
        <div class="item trois jaune">Elément 3</div>
    </div>
</body>
```

DIRECTION ET CONTRÔLER LES ÉLÉMENTS

flex-direction

flex-direction définie la direction et le sens de l'axe principal utilisé par le conteneur flexible pour placer les éléments flexibles.

row
- (valeur par défaut) : la direction suit le flux normal inline (de gauche à droite).

row-reverse
- la direction suit le flux normal inline (de droite à gauche).

column
- la direction suit le flux normal en block. (de haut vers le bas).

column-reverse
- la direction suit le flux normal en block. (du bas vers le haut).

flex-wrap

flex-wrap permet d'indiquer si les éléments flexibles ont le droit de passer à la ligne ou pas ainsi que la direction des éléments dans laquelle les lignes doivent être empilées

nowrap
- (valeur par défaut) : Les éléments n'ont pas le droit d'aller à la ligne.

wrap
- Les éléments vont se placer à la ligne plutôt que de dépasser du conteneur.

wrap-reverse
- Les éléments vont se placer à la ligne plutôt que de dépasser du conteneur.

La propriété flex-flow est une propriété raccourcie qui permet de définir les valeurs de flex-direction et de flex-wrap en une seule fois.

La propriété writing-mode permet d'indiquer si les lignes de texte sont disposées horizontalement ou verticalement et la direction dans laquelle les blocs progressent.

CONTRÔLE DE LA FLEXIBILITÉ

flex-grow

La propriété flex-grow permet de définir le facteur d'agrandissement d'un élément flexible.

```
flex-grow : number;
```

- Un nombre qui correspond au facteur de grossissement utilisé. Plus la valeur est élevée, plus l'élément sera étendu si nécessaire.
- *Les valeurs négatives sont invalides.*
- *La valeur par défaut est 0.*

flex-shrink

La propriété flex-shrink permet de définir le facteur de rétrécissement d'un élément flexible.

```
flex-shrink : number;
```

- Un nombre qui correspond au facteur de rétrécissement utilisé. Plus la valeur est élevée, plus l'élément sera compressé si nécessaire.
- *Les valeurs négatives sont invalides.*
- *La valeur par défaut est 1.*

CONTRÔLE DE LA FLEXIBILITÉ

flex-basis

La propriété flex-basis permet de définir la taille de base des éléments flexibles avant distribution de l'espace restant ou rétrécissement

flex-basis : width ou content;

- Un longueur absolue (px) ou un pourcentage relatif (%) à la taille principale du conteneur flexible ou encore le mot-clé auto.
  - Les valeurs négatives ne sont pas autorisées.
  - La valeur par défaut est auto.

flex

Raccourcie qui permet de définir les valeurs de flex-grow, flex-shrink et flex-basis d'un seul coup.

On peut soit passer les valeurs des propriétés flex-grow, flex-shrink et flex-basis à la suite à flex, soit lui passer les mots clefs suivants :

- flex : initial; équivalent flex : 0 1 auto;
- flex : auto; équivalent flex : 1 1 auto;
- flex : none; équivalent flex : 0 0 auto;

CONTRÔLER L'ALIGNEMENT

JUSTIFY-CONTENT , ALIGN-ITEMS , ALIGN-SELF

justify-content va nous permettre de gérer l'alignement des éléments flexibles selon l'axe principal choisi pour le conteneur flex.

A appliquer au conteneur flex

flex-start
- (valeur par défaut) : au début de l'axe principal

flex-end
- à la fin de l'axe principal

center
- centre de l'axe principal

space-around
- même espace entre chaque élément mais 1er et dernier éléments ont une marge avec le conteneur

space-between
- même espace entre chaque élément mais 1er et dernier éléments collés contre le conteneur

align-items permet d'aligner les éléments flexibles selon l'axe secondaire du conteneur flex.

align-self permet de régler l'alignement des éléments flexibles de manière individuelle.

A appliquer au conteneur flex

Auto
- (uniquement align-self) L'alignement contrôlé par align-items du parent de l'élément.

flex-start
- au début de l'axe secondaire

flex-end
- à la fin de l'axe secondaire

center
- au centre de l'axe secondaire

baseline
- aligner dans l'axe secondaire de telle manière à ce que leur ligne de base soit alignée.

stretch
- s'étend pour occuper toute la place dans l'axe secondaire du conteneur flex

CONTRÔLER L'ALIGNEMENT

Apprendre en jouant

https://flexboxfroggy.com/#fr

Flexbox Froggy

align-content

align-content va servir à aligner les différentes lignes selon l'axe secondaire du conteneur, c'est-à-dire en plaçant cet espace avant, après ou entre les lignes.

flex-start

- grouper au début du conteneur flexible

flex-end

- grouper à la fin du conteneur flexible

space-between

- L'espace restant réparti équitablement entre les lignes selon l'axe secondaire. pas d'espace entre le départ du conteneur et la première ligne ni entre la fin du conteneur et la dernière ligne

Space-around

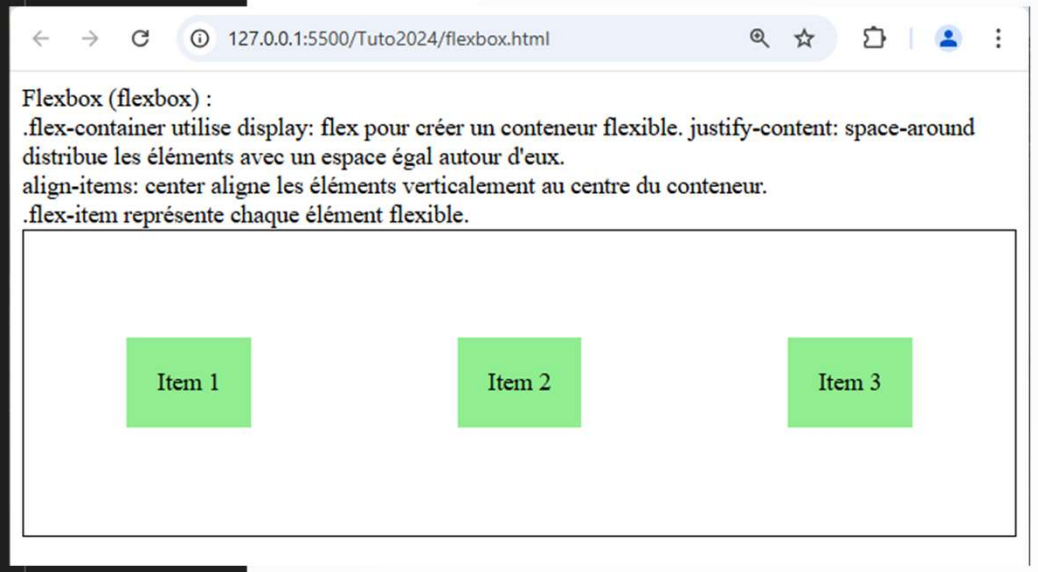
- L'espace restant réparti équitablement entre les lignes selon l'axe secondaire.

stretch

- (valeur par défaut) : Les lignes vont s'étendre afin d'occuper tout l'espace selon l'axe secondaire du conteneur.

Un exemple de FLEXBOX

```html
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Exemple de Flexbox</title>
    <style>
        .flex-container {
            display: flex;
            justify-content: space-around;
            align-items: center;
            height: 200px;
            border: 1px solid black;
        }
        .flex-item {
            background-color: lightgreen;
            padding: 20px;
            margin: 10px;
            text-align: center;
        }
    </style>
</head>
<body>

    <article>
        Flexbox (flexbox) : <br>
        .flex-container utilise display: flex pour créer un conteneur flexible.
        justify-content: space-around distribue les éléments avec un espace égal autour d'eux.<br>
        align-items: center aligne les éléments verticalement au centre du conteneur.<br>
        .flex-item représente chaque élément flexible.<br>
    </article>

    <div class="flex-container">
        <div class="flex-item">Item 1</div>
        <div class="flex-item">Item 2</div>
        <div class="flex-item">Item 3</div>
    </div>
</body>
</html>
```



QUELQUES ÉLÉMENTS CSS

Un peu de style

Quelques éléments CSS

Du style aux boîtes

box-shadow : Ombre portée par la boîte

- Décalage à droite, décalage en bas, taille de l'ombre, couleur de l'ombre

text-shadow : Ombre portée par le texte

- Décalage à droite, décalage en bas, taille de l'ombre, couleur de l'ombre

Border-radius : Arrondis circulaires des angles de la bordure

Border-top-left-radius, border-bottom-right : Arrondis elliptiques des angles de la bordure (2 paramètres séparés par un espace)

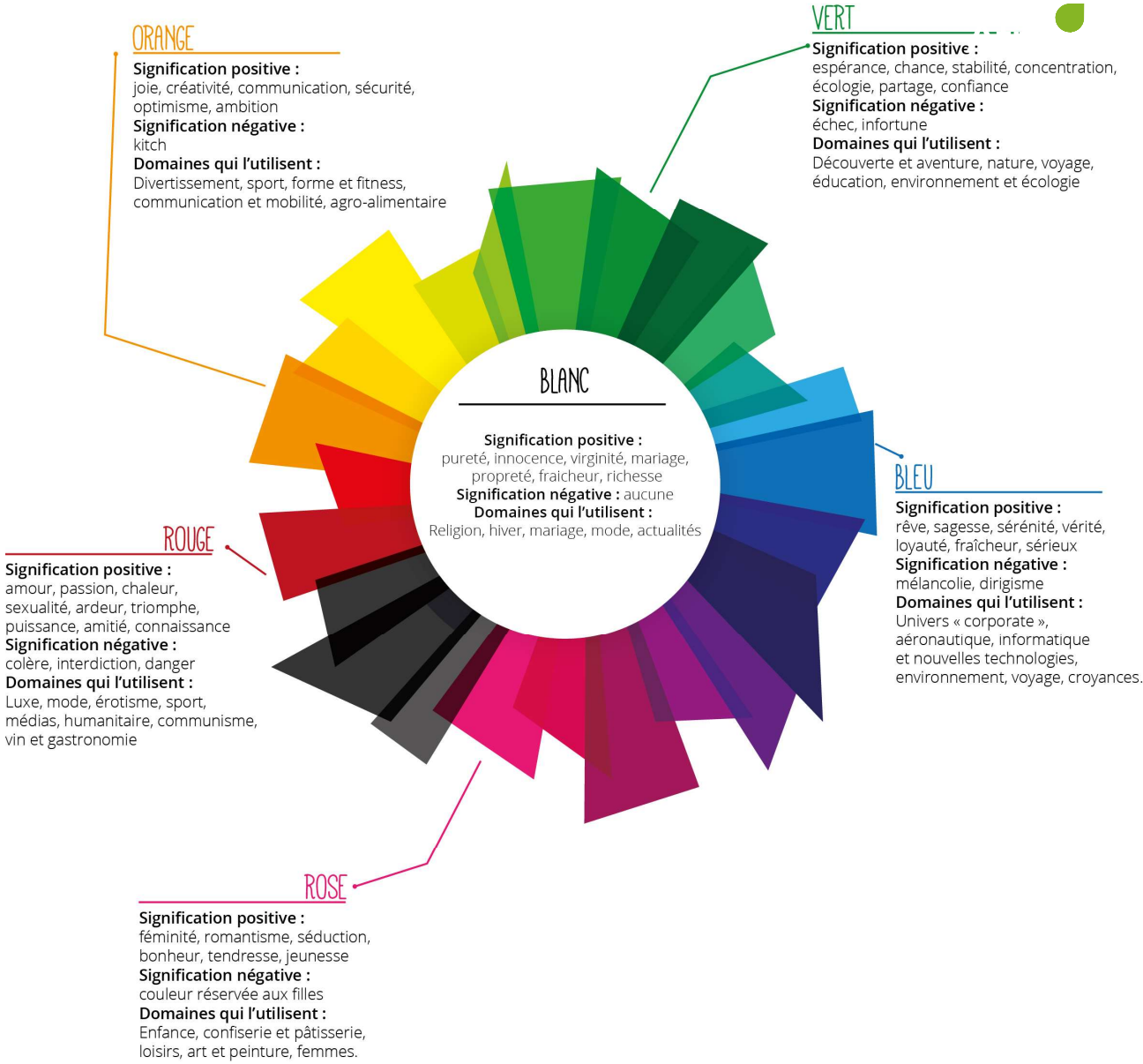
Les couleurs dans le web

https://www.creads.com/blog/decryptage/comment-faire/couleurs-site-web-conseils/

Quelques règles à connaitre :

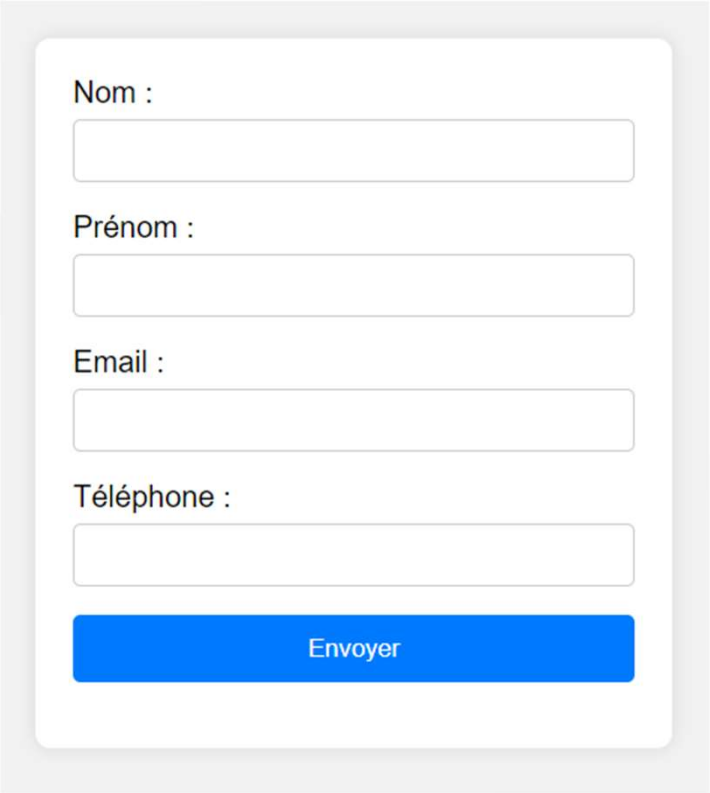
- Choisir 3 couleurs maximum en dehors du noir, du blanc et du gris. (lisibilité)
- Adopter des couleurs appropriées à l'auditoire en prenant en compte la symbolique des couleurs
- Définir la bonne teinte pour chacune des couleurs.

- Repartir les couleurs de façon proportionnée : règle des 60-30-10
  - 60% de l'espace pour la couleur dominante
  - 30% de l'espace pour la couleur secondaire (contraster avec le 1er couleur)
  - 10% de l'espace pour la couleur complémentaire (accentuer certains éléments)



Exemple de formulaire

EXEMPLE DE FORMULAIRE




```html
<html lang="fr">
<head>
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Formulaire de contact</title>
    <link rel="stylesheet" href="formulaire.css">
    <style>

    </style>
</head>
<body>
    <form action="#" method="post">
        <div>
            <label for="nom">Nom :</label>
            <input type="text" id="nom" name="nom" required>
        </div>
        <div>
            <label for="prenom">Prénom :</label>
            <input type="text" id="prenom" name="prenom" required>
        </div>
        <div>
            <label for="email">Email :</label>
            <input type="email" id="email" name="email" required
                pattern="[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$"
                title="Veuillez entrer une adresse email valide">
        </div>
        <div>
            <label for="telephone">Téléphone :</label>
            <input type="tel" id="telephone" name="telephone"
                pattern="[0-9]{10}"
                title="Veuillez entrer un numéro de téléphone valide (10 chiffres)">
        </div>
        <div>
            <button type="submit">Envoyer</button>
        </div>
    </form>
</body>
</html>
```

EXEMPLE DE FORMULAIRE : CSS

```css
body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    display: flex;
    justify-content: center; /* Centrer le contenu horizontalement */
    align-items: center; /* Centrer le contenu verticalement */
    height: 100vh; /* 100% de la hauteur de la fenêtre */
    background-color: #f2f2f2;
}
form {
    width: 300px;
    padding: 20px;
    background-color: #fff;
    border-radius: 8px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}
form div {
    margin-bottom: 15px;
}
```

```css
label {
    display: block;
    margin-bottom: 5px;
}
input[type="text"],
input[type="email"],
input[type="tel"] {
    width: 100%;
    padding: 8px;
    border: 1px solid #ccc;
    border-radius: 4px;
    box-sizing: border-box;
}
button[type="submit"] {
    width: 100%;
    padding: 10px;
    background-color: #007bff;
    color: #fff;
    border: none;
    border-radius: 4px;
    cursor: pointer;
}
button[type="submit"]:hover {
    background-color: #0056b3;
}
```

Exemple de page avec et sans mise en forme

BLOG DE RECETTES DE CUISINE

Mes Recettes de Cuisine

- Recette 1
- Recette 2
- Recette 3

Recette 1 : Tarte aux Pommes

Ingrédients : ...

Instructions : ...

Recette 2 : Soupe à l'Oignon

Ingrédients : ...

Instructions : ...

Recette 3 : Poulet Rôti

Ingrédients : ...

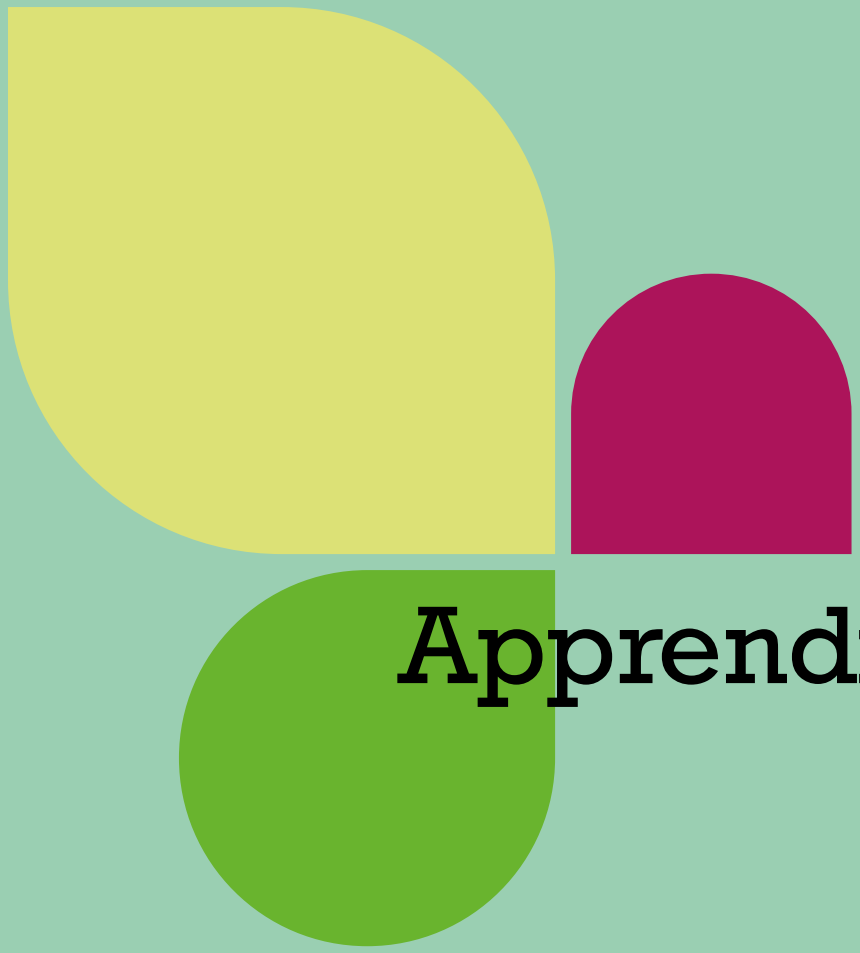
Instructions : ...

© 2024 Mes Recettes de Cuisine

```html
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Recettes de Cuisine</title>
    <link rel="stylesheet" href="./Tuto2024/Style/testcss.css">
</head>
<body>
    <header>
        <h1>Mes Recettes de Cuisine</h1>
    </header>
    <nav>
        <ul>
            <li><a href="#recette1">Recette 1</a></li>
            <li><a href="#recette2">Recette 2</a></li>
            <li><a href="#recette3">Recette 3</a></li>
        </ul>
    </nav>
    <main>
        <section id="recette1">
            <h2>Recette 1 : Tarte aux Pommes</h2>
            <p>Ingrédients : ...</p>
            <p>Instructions : ...</p>
        </section>
        <section id="recette2">
            <h2>Recette 2 : Soupe à l'Oignon</h2>
            <p>Ingrédients : ...</p>
            <p>Instructions : ...</p>
        </section>
        <section id="recette3">
            <h2>Recette 3 : Poulet Rôti</h2>
            <p>Ingrédients : ...</p>
            <p>Instructions : ...</p>
        </section>
    </main>
    <footer>
        <p>&copy; 2024 Mes Recettes de Cuisine</p>
    </footer>
</body>
</html>
```

AVEC UN PEU DE CSS...

```css
/* styles.css */
body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    background-color: #f4f4f4;
}

header {
    background-color: #4CAF50;
    color: white;
    padding: 1em 0;
    text-align: center;
}

nav ul {
    list-style-type: none;
    padding: 0;
    background-color: #333;
    text-align: center;
}

nav ul li {
    display: inline;
    margin: 0 1em;
}

nav ul li a {
    color: white;
    text-decoration: none;
}

main {
    padding: 2em;
}

section {
    margin-bottom: 2em;
    background-color: white;
    padding: 1em;
    border-radius: 8px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}

footer {
    background-color: #333;
    color: white;
    text-align: center;
    padding: 1em 0;
    position: fixed;
    width: 100%;
    bottom: 0;
}
```



Apprendre en jouant

JEUX SUR LE CSS

https://flukeout.github.io/

L'objectif est simple : utiliser le CSS pour accomplir les diverses tâches demandées.