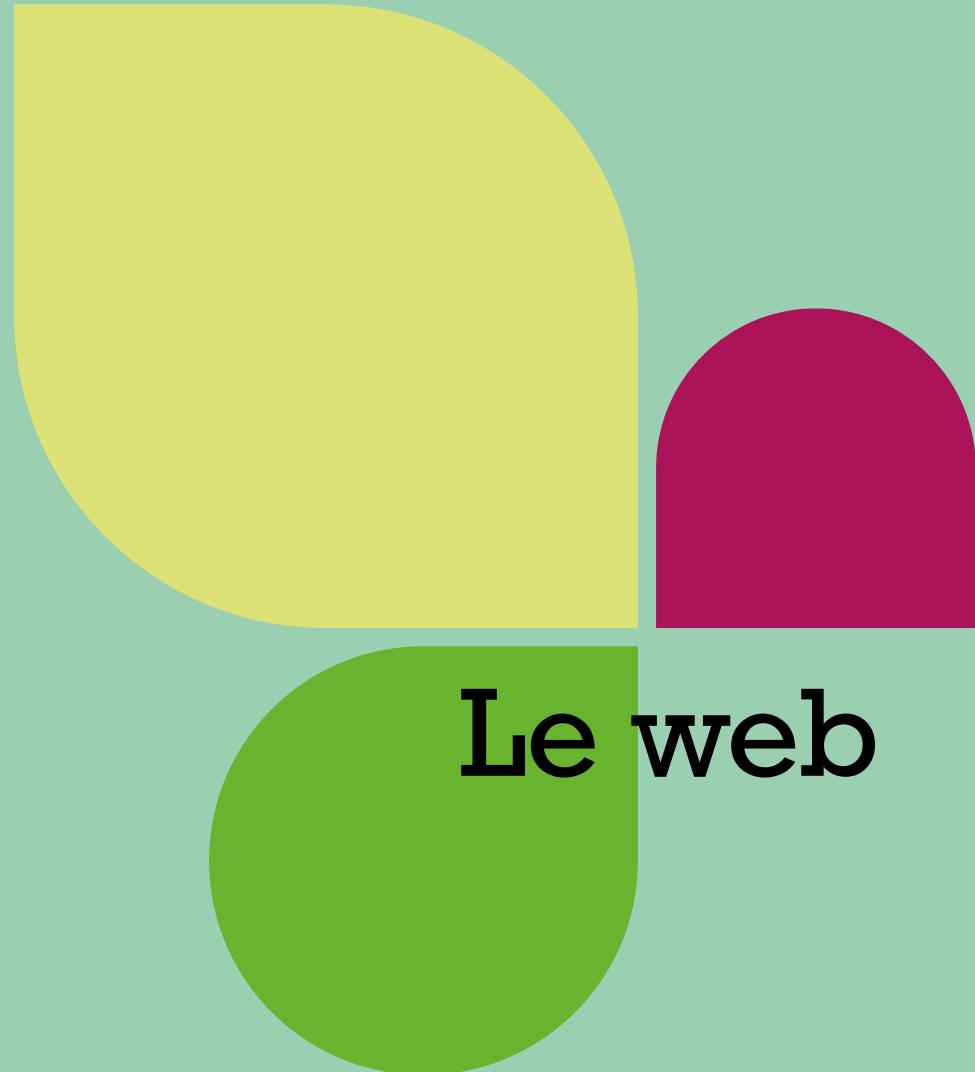


HTML et CSS

Le web et ses technologies





Le web

HISTORIQUE

Quelques dates...

Année 90 : Explosion du web, avec la sortie de nombreux navigateurs, la mise en place de milliers de serveurs web et la création de millions de pages web.



Fin des années 60,
l'armée américaine a
développé un réseau
de communication
nommé **ARPANET**.



En 1980, **Tim Berners-Lee** (souvent abrégé TimBL) a écrit un programme nommé **ENQUIRE**, basé sur le concept de liens entre différents points.



Fin des années 90, TimBL avait créé tout le nécessaire pour faire fonctionner une première version du web — **HTTP**, **HTML**, le premier navigateur, qui s'appelait **WorldWideWeb**, un serveur **HTTP** et quelques pages web à lire

En 1989, où TimBL a écrit **Information Management: A Proposal** et **HyperText at CERN**, deux ouvrages fournissant tout le contexte du fonctionnement du Web.

W3C

World Wide Web Consortium



Organisme de standardisation, fondé en 1994, chargé de promouvoir la comptabilité des technologies du World Wide Web.



Gestion conjointe par le MIT (USA) et l'ERCIM en Europe, l'Université Keio au Japon et l'Université Beihang en Chine.



Elle définit et développe l'ensembles des standards qu'on utilise dans le développement d'applications web ou de pages Web : HTML, CSS, DOM, SVG etc...

QUELQUES DÉFINITIONS

Un peu de lexique

https://192.30.253.45:443

Protocol

IP address

Port #

Anatomie de l'IP reçue par le navigateur



Une Page

Écrit en Hyper Text Markup Language ([HTML](#)), décrivant la structure et la mise en forme de la page, utilisant le protocole [HTTP](#) et repéré par une adresse unique appelée [URL](#)



World Wide Web

C'est l'ensemble des pages [HTML](#) reliées entre elles, stockées sur des serveurs WEB accessibles via une [URL](#) et accessible via un navigateur depuis un poste client.



URL

[Uniform Resource Locator](#) : Format de nommage universel pour désigner une ressource sur Internet, utilisant un protocole composé d'une adresse IP ou un nom de domaine et d'un port



Protocole et port

HTTP (80 ou 8080), HTTPS (443), FTP (20 ou 21), etc...



DNS

Domain Name System permet de traduire les noms de domaine en une adresse IP correspondant à une machine (serveur). [Ceci grâce aux fichiers de zone placés sur les serveurs DNS](#)

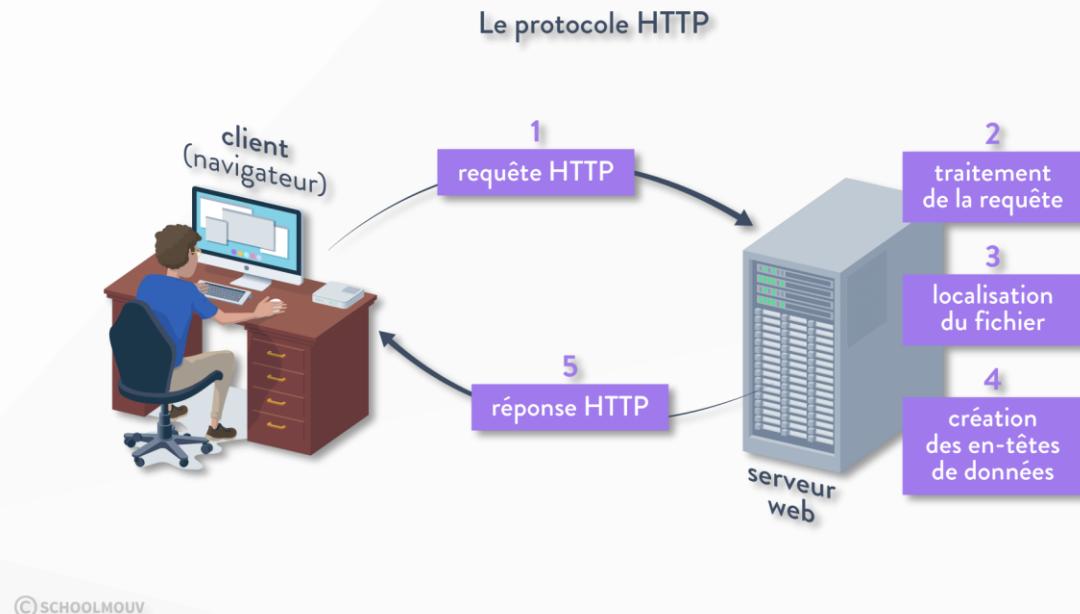
PROTOCOLE HTTP ET REQUÊTES HTTP

1. Le navigateur effectue une requête HTTP.

- Une requête est composée :
 - D'un en-tête et du corps de la requête
- Différentes méthodes :
 - GET, POST
 - HEAD, PUT, DELETE

2. Le serveur traite la requête puis envoie une réponse HTTP.

- Une Réponse est composée :
 - D'un statut :
 - protocole utilisé
 - code de retour et de son libellé
 - D'un en-tête,
 - Du corps de la réponse



HTML + CSS + JAVASCRIPT

- On sépare la présentation du contenu du document en utilisant des feuilles de style CSS. Les pages HTML étant statiques, on utilise des scripts Javascript pour les rendre dynamique



HTML : Contenu et structure

En-tête, Titre, paragraphe...



CSS : Présentation

Font, Couleur, Bordure...



JAVASCRIPT : Comportement

Affichage dynamique,
interaction, popup...

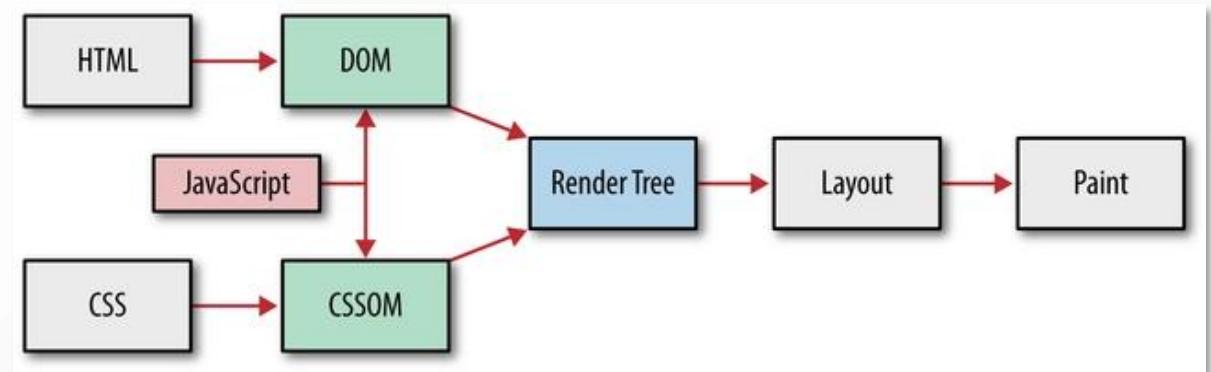
LES MOTEURS DE RENDU

Chaque navigateur possède son propre moteur de rendu.

- composant logiciel permettant d'afficher une page Web en transformant le document HTML, ainsi que toutes les ressources associées en une représentation visuelle interactive.

Plusieurs moteurs de rendu existent.

- Firefox utilise par exemple Gecko,
 - Safari, Chrome et Opera utilisent Webkit (ou un dérivé : Blink),
 - Internet Explorer utilise Trident.
- Leurs processus pour afficher une page web peuvent légèrement différer les uns des autres, mais il s'agit généralement d'une même séquence d'événements :

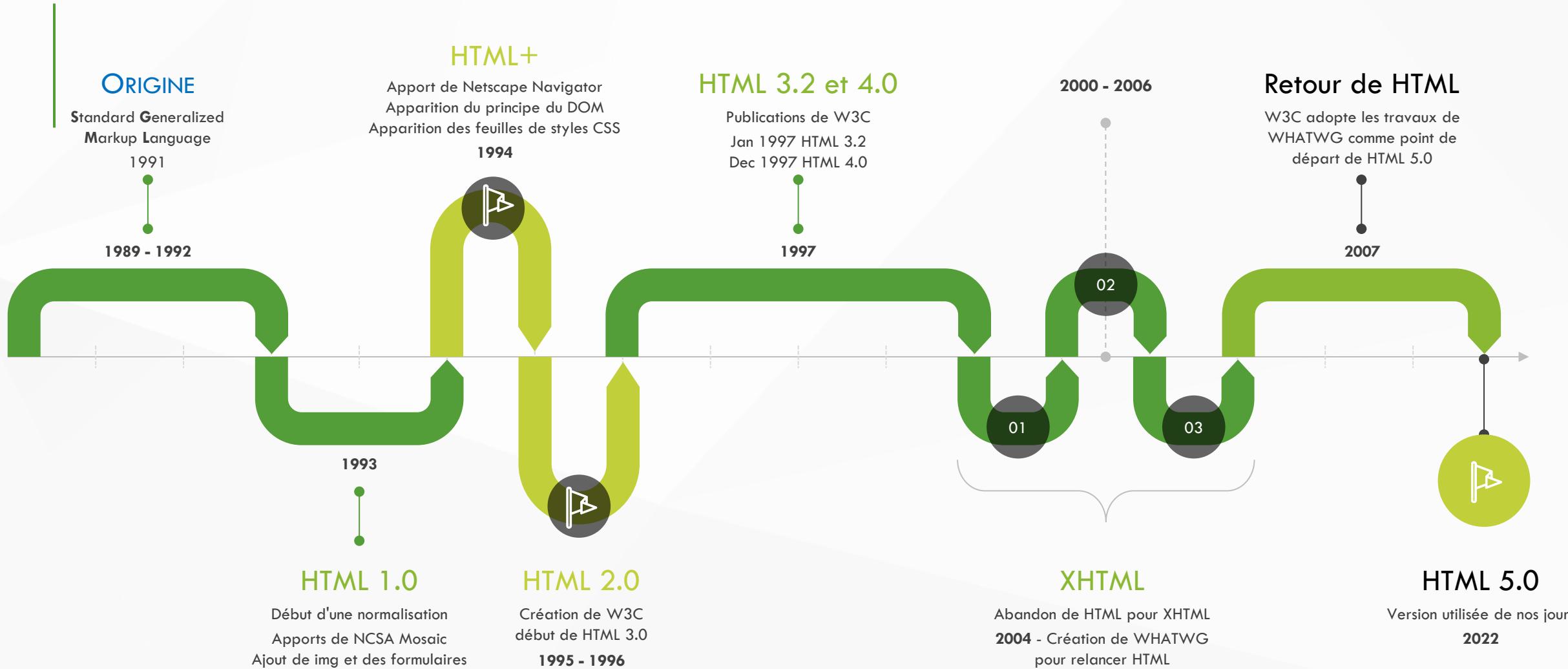


[Comprendre-le-navigateur](#)

<https://thibault.mahe.io/blog/comprendre-le-navigateur>



HTML un langage de balisage



HTML - UN LANGAGE DE BALISAGE

- Présentation
 - HTML (HyperText Markup Language) est le langage utilisé pour structurer le contenu d'une page web.
 - Il organise le texte, les images, les liens et les éléments multimédias grâce à des balises.
 - C'est la base de toute page internet, souvent utilisée avec CSS et JavaScript.Documentations

Documentations :

- <https://developer.mozilla.org/fr/>
- <https://devdocs.io/>
- <https://alsacreations.com/>

HTML 5.0

Il a conservé de XHTML, sa rigueur du langage XML, tout en simplifiant sa syntaxe :

- Structuré en arbre
 - Utilisation de balise en respectant avec la sémantique de l'élément (une liste n'est pas un tableau).
- Sépare la structure du document du style
- Quelques règles :
 - N'omettre aucune des balises qui définissent la structure de l'arbre du document.
 - Fermer les balises.
 - Balises et attributs sont toujours en minuscules.
 - Les valeurs des attributs sont entre guillemets.
 - Tous les attributs ont des valeurs explicites.



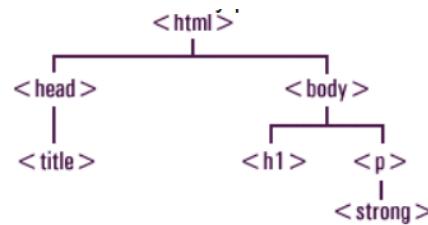
Composition d'une page HTML

Structure d'une page HTML

HTML et les balises.

</> <tag>...</tag>

La structure d'une page peut être représentée sous la forme d'un arbre appelé **Document Object Model (DOM)**



Structure

Balises

Attributs

“

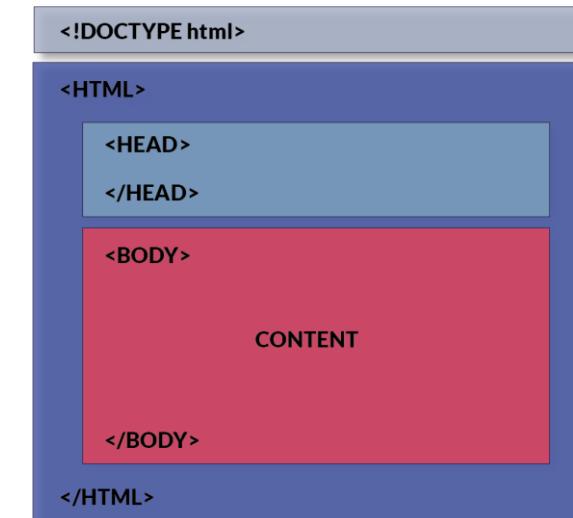
HTML est un langage de balises.

Toutes les instructions sont exprimées dans des balises (ou "tags"), qui sont encadrées par des crochets <>.

Chaque balise doit être ouverte et fermée de manière à encadrer le contenu sur lequel doit s'appliquer la balise.

Les balises de fermeture sont les mêmes que les balises d'ouverture, mais sont précédées d'un caractère slash (/).

Composition et structure d'une page HTML

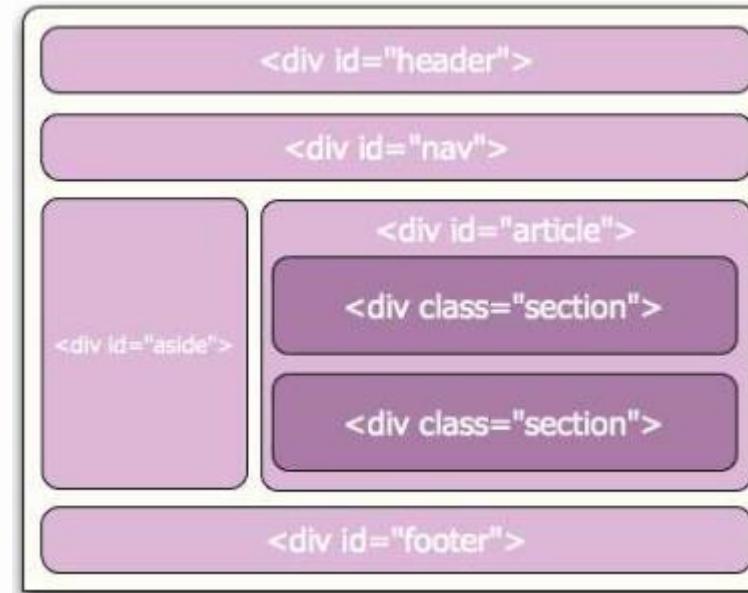


LE RENDU D'UNE PAGE

Le développeur doit toujours raisonner en **proportions et en pourcentages** et non en nombre de pixels pour effectuer la mise en page.

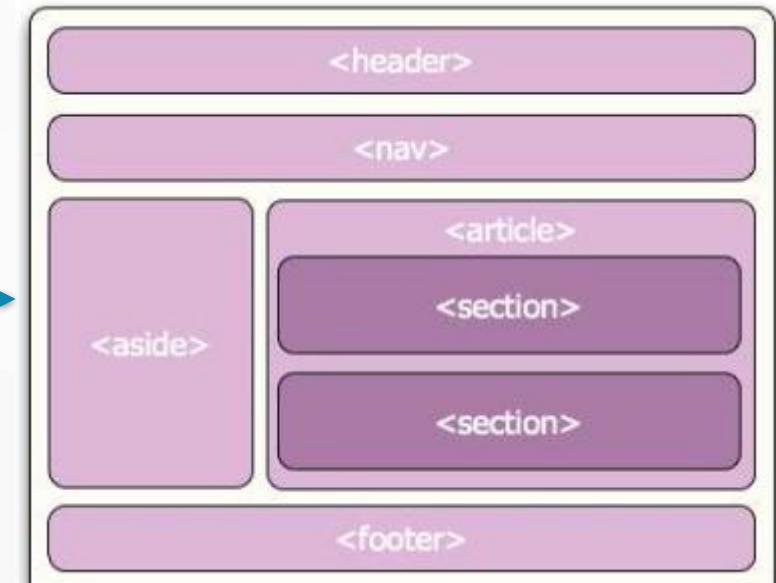
Evolution de la mise en page

- Dans 1^{er} temps, les tableaux HTML ont été le moyen d'assurer et de contrôler la mise en page. **Lourd car pas trop évolutif**
- Dans un second temps, la tendance a ensuite été de définir les parties d'une page à l'aide de balise DIV et de CSS. **Gros travail !!**
- HTML5 apporte des éléments de solution à ces problèmes. **Nouveaux éléments propres + CSS.**



HTML4

HTML5



BALISE

Nos balises peuvent avoir des attributs.

- Ces attributs se positionnent dans la balise ouvrante :
 - `<p class="important">... </p>`
 - Pour connaître les attributs d'une balise, il faut aller dans la documentation.
- Le chevauchement de balises n'est pas autorisé :
 -  `<h1><a href...> ...</h1> `
 -  `<h1><a href...>... </h1>`

Eléments vides

- Pas de balises fermantes
 - Indiqué dans la documentation
- ``
- `<hr>`
- ...



Résumé technique

Catégories de contenu

Contenu de flux

Contenu autorisé

Aucun, c'est un élément vide.

HTML V5

Chaque page HTML doit commencer par l'instruction **DOCTYPE** ¹ qui permet de définir le standard HTML utilisé.

- *Par chance HTML 5 a simplifié la syntaxe à l'extrême.*

Ensuite, votre page doit contenir les balises `<html>` ... `</html>` qui indique le début et la fin de votre document. **C'est la racine.**

Entre ces balises, on y trouve l'en-tête `<head>` `</head>` et le corps `<body>` `</body>` du document.

```
<!DOCTYPE html>

<html lang="fr">

  <head>

    <!-- ENCODAGE -->
    <meta charset="text/html UTF-8">
    <!-- Comptabilité navigateur selon version -->
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <!-- prise en charge du contexte mobile -->
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <!-- rafraîchissement de la page -->
    <meta http-equiv="refresh" content="2;url=file:///C:/Users/jboeb/
Documents/PROJETS/HTML/CoursHTML/index.html">
    <!-- titre de la page -->
    <title>Document</title>

  </head>

  <body>

  </body>

</html>
```

STRUCTURE MINIMALE

Au minima dans une page HTML, on va retrouver dans le body :

- 1 header
- 1 main
- 1 footer

Ensuite, on peut dès lors construire sa page.

Note : évidemment, toutes nos pages ne vont pas contenir la même structure.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="description" content="tuto HTML">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>firstDocument</title>
</head>
<body>

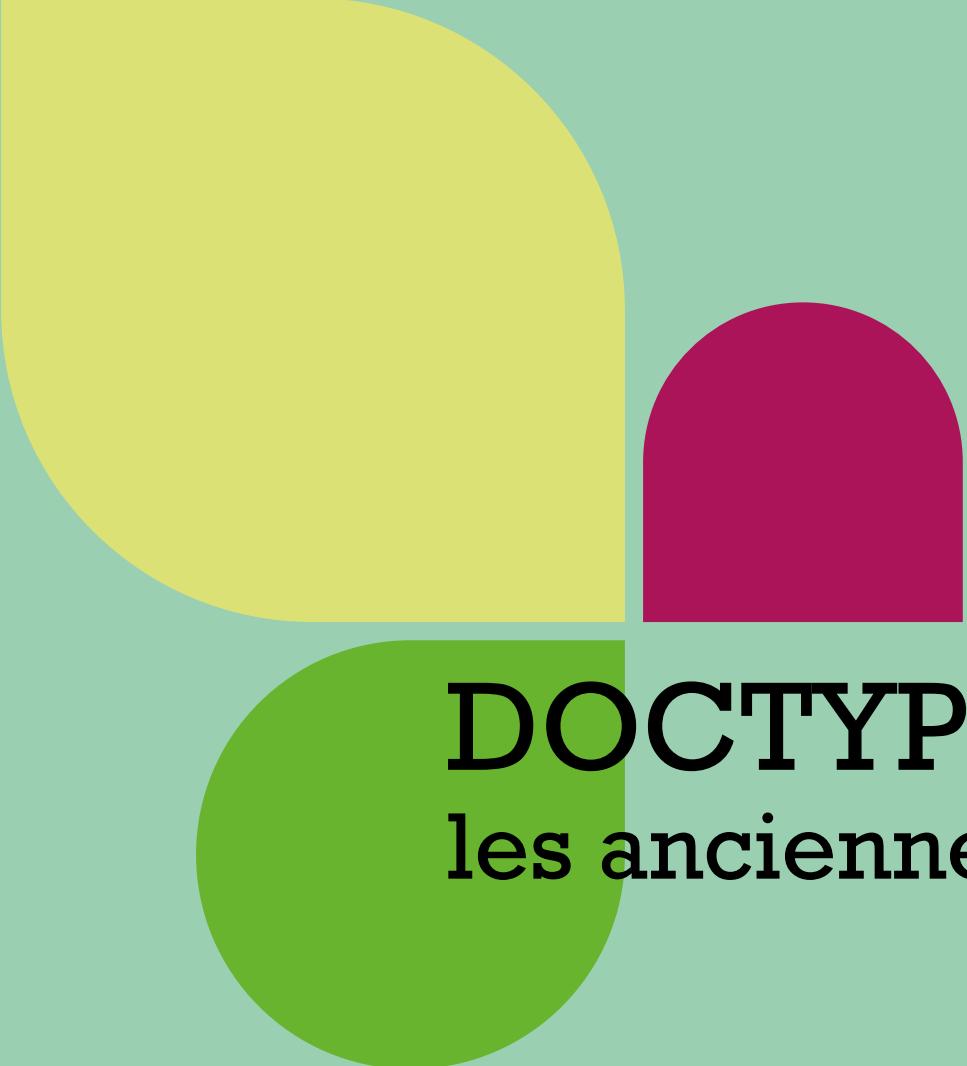
    <!-- ceci est un commentaire -->

    <header>
        <!-- le header -->
    </header>

    <main>
        <!-- le main -->
    </main>

    <footer>
        <!-- le footer -->
    </footer>

</body>
</html>
```



DOCTYPE

les anciennes versions de HTML

DOCTYPE

Exemple de DOCTYPE pour les anciennes versions HTML.

HTML 4.01 Strict

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"  
"http://www.w3.org/TR/html4/strict.dtd">
```

HTML 4.01 Transitional

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"  
"http://www.w3.org/TR/html4/loose.dtd">
```

HTML 4.01 Frameset

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"  
"http://www.w3.org/TR/html4/frameset.dtd">
```

XHTML 1.0 Transitionnel

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"  
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

XHTML 1.0 Strict

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"  
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

XHTML 1.0 Frameset

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"  
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
```

XHTML 1.1

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"  
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
```



LES ÉLÉMENTS SÉMANTIQUE

Valeur des éléments

LES ÉLÉMENTS SÉMANTIQUES

Les éléments sémantiques sont une des innovations significatives en [HTML5](#).

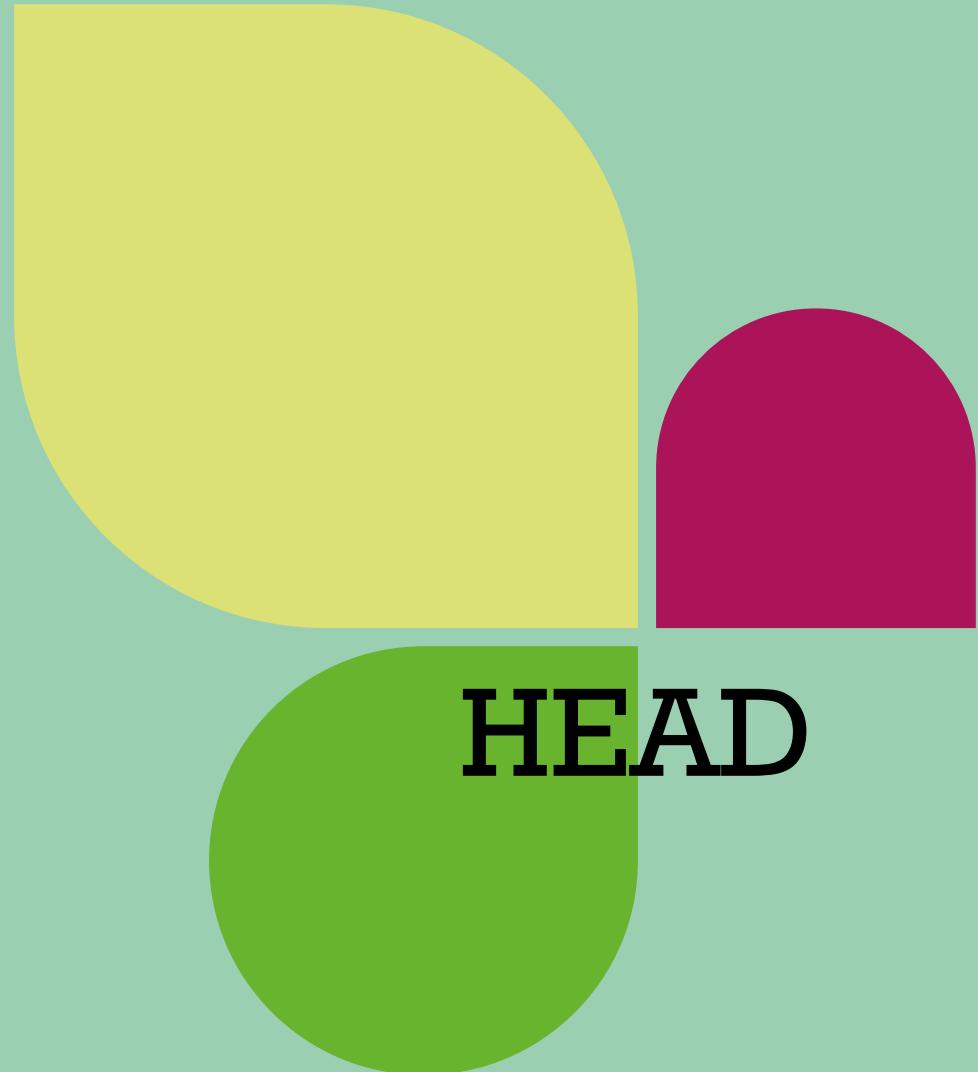
- Avant leur apparition, toutes les balises des pages Web étaient créées à l'aide des éléments `<div>`, auxquels étaient attribués des identificateurs (`id`) ou des classes (`class`).
- Pour placer des panneaux latéraux, des entêtes, des pieds de page, des éléments de navigation et d'autres blocs de construction, des blocs `div` avec les valeurs appropriées (par exemple, `div = "footer"`) ont été utilisés.

HTML5 introduit de nouveaux éléments sémantiques pour la structuration, le regroupement du contenu et le balisage du contenu du texte.

Ils décrivent clairement quel contenu ils ont et donne ainsi une valeur à ce contenu.

On trouve parmi ces éléments :

- `<header>` : en-tête
- `<nav>` : bloc principal des liens de navigation.
- `<article>` : bloc indépendant
- `<section>` : regrouper le contenu thématique
- `<aside>` : section contenant des informations complémentaires
- `<footer>` : pied de page
- `<address>` : coordonnées de l'auteur du document
- `<main>` : contenu principal
- `<figure>` : un contenu autonome au flux principal
- `<figcaption>` : légende
- `<time>` : affichage en 24 heures
- `<mark>` : mise en évidence
- `<bdi>` : formulation bidirectionnelle
- `<wbr>` : rupture dans une chaîne



HEAD

Le **HEAD** contient les informations sur le document courant (titre, jeu de caractères, mots-clés, icône de la page, rafraîchissement automatique, etc...)

Ces informations ne sont pas destinées à apparaître sur le corps de la page HTML.

```
<head>
<meta http-equiv="content-type" content="text/html;
charset=utf-8"/>
<title>Your page title</title>
<meta name="description" content="your page description." />
<meta name="Keywords" content="your keywords" />
<link href='your css file' rel='stylesheet' type='text/css'>
<script src="your js file"></script>
</head>
```

Titre du document

`<title>`

Encodage de la page

`charset`

Cette balise sert à fournir certaines informations.

`meta`

Lien vers une ressource externe

`link`

Permet d'insérer directement du Javascript

`script`

Permet d'insérer directement du jCSS (pas conventionnel)

`style`

La balise meta

La balise `<meta>` fait partie des *informations de services* placées en entête de la page Web (`<head>`).

- Elle permet de fournir une description de votre site par le biais de mots-clés et de phrases.

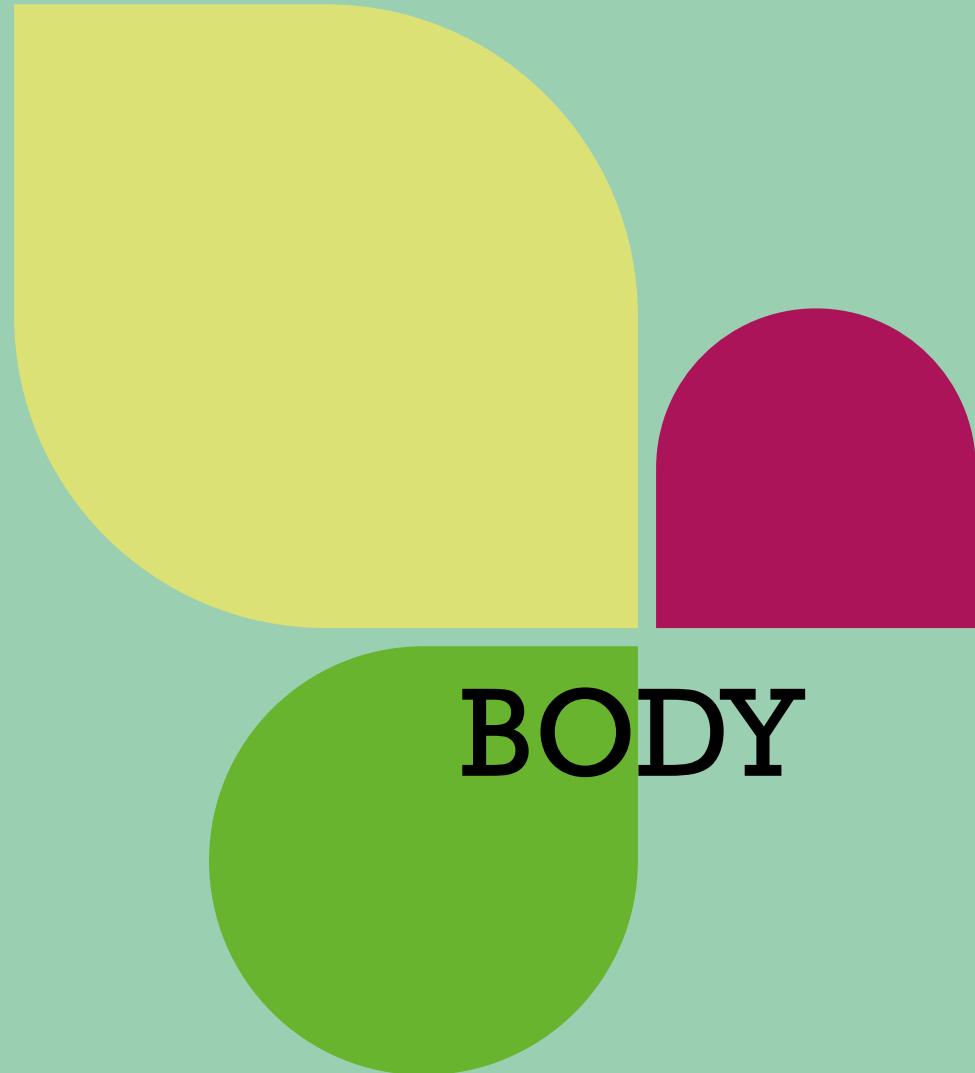
De plus, les balises `<meta>` ont d'autres fonctionnalités, comme notamment la redirection automatique vers une autre URL

Recharger une page régulièrement
`<meta http-equiv="refresh" content="5" />`

Définir une date limite de votre page
`<meta http-equiv="expires" content="Wed, 19 Nov 2025 14:01:00 GMT" />`

Interdire de mettre en cache
`<meta http-equiv="pragma" content="no-cache" />`

indexer et documenter votre site
`<meta name="author" lang="fr" content="jero" />`
`<meta name="description" content ="Tapez ici en quelques phrases la description de votre site" />`
`<meta name="keywords" lang="fr" content ="Tapez ici les mots-clés séparés par une virgule" />`
`<meta name="copyright" content ="COPYRIGHT NOM" />`
`<meta name="reply-to" content ="webmaster@domaine.fr"/>`



BODY

Le **BODY** contiendra ce que la page va afficher au travers du navigateur soit le corps du document.

Le corps du document peut être aussi structuré en plusieurs parties afin de donner du sens aux éléments de la page.

- **<section>** est un groupe thématique de contenus
- **<article>** est une portion de page qui se suffit à elle-même.
- **<header>** est un entête pouvant servir à une introduction, un titre, des éléments de navigation, etc..
- **<footer>** contient des informations sur la page : l'auteur, copyright etc...
- **<nav>** barre de navigation
- **<aside>** cet élément représente une portion de document dont le contenu a une relation avec le contenu principal, sans pour autant être essentiel

Les éléments en ligne :

- Affichés au fil du texte (dans le flux courant),
- Apporte un sens au texte, sans marges internes et externes
 - Des portions de phrase ****
 - Des mises en exergue ****
 - Des liens **<a>**
 - Des images ****
 - Les images doivent toujours avoir une source et un alt en attribut.

Les éléments de type bloc :

- **<div>** conteneur possédant des marges internes et externes et des dimensions (hauteur et largeur)
 - Il n'a pas d'interprétation sémantique pour HTML
 - Se positionne en dehors du flux courant, structure les autres éléments
- Des titres de **<h1>** le plus gros à **<h6>** le plus petit
 - Ils doivent être utilisés dans l'ordre hiérarchique

Les commentaires :

- **<!-- ceci est un commentaire -->**

MISE EN FORME DES TEXTES ET PARAGRAPHES

La balise `<div>` est un conteneur permettant de structurer d'autres éléments HTML, mais aussi d'autres `<div>`

- Effet visuel : utilise toute la largeur disponible et retour à la ligne selon le contenu.

La balise `` permet d'identifier une portion de phrase.

- Pas d'effet visuel.

Attribut	Utilité
<code>id</code>	Identifiant de la balise.
<code>class</code>	liste de classes séparées par des espaces. Utilisé pour appliquer un style
<code>style</code>	Information locale de style
<code>title</code>	Permet une information supplémentaire concernant l'élément.
<code>align</code>	Permet l'alignement du texte (left, center, right, justify) (cet attribut est déprécié, et remplacé par des styles CSS)

CONTENU

Paragraphe

En HTML, on découpe le document en paragraphe pour une meilleure lisibilité.

La balise `<p> </p>` permet de définir un paragraphe.

- L'attribut `align` de la balise permet l'alignement de son contenu

Une autre balise `
` permet de faire des sauts de ligne à l'intérieur d'un paragraphe. **Mais utilisation à proscrire**

Les titres

Les balises titre permettent de structurer et mettre en évidence des parties de texte selon un formatage défini par le navigateur.

- plusieurs styles
- retour à la ligne automatique (changement de paragraphe).
- attribut `align` disponible

CONTENU

Les lignes horizontales

On peut insérer des lignes horizontales pour séparer des paragraphes par exemple.

La balise `<hr>` joue ce rôle.

Des attributs (color, border, width, align etc..) de présentation sont disponibles mais par soucis de bonne convention, on utilisera des feuilles de styles CSS.

Mise en forme des caractères et polices de caractères

La mise en forme se fait avec la balise `style` permettant de donner un sens au contenu. La balise `` agit sur la police de caractères.

Nous n'allons pas forcément voir tous les attributs disponibles et il faut à force de pratique en faire la découverte.

Cependant, afin de respecter les bonnes pratiques, nous utiliserons les feuilles de style CSS pour appliquer du style à notre document.

LES COULEURS

Les trois couleurs de base RGB (rouge, vert, bleu) permettent de composer les couleurs à l'écran.

- Exprimé en Héxadécimal de 00 à FF
- 256 nuances pour chacune des couleurs

De nombreux sites vous permettent de choisir la bonne couleur :

- <https://www.code-couleur.com/>
- <https://www.toutes-les-couleurs.com/code-couleur-html.php>
- *Remarques : les goûts et les couleurs ça ne se discute pas mais en tant que concepteur, il faut faire en sorte de conserver une harmonie de vos pages WEB.*



Couleur	Valeur RGB
black	#000000
silver	#C0C0C0
gray	#808080
white	#FFFFFF
maroon	#800000
red	#FF0000
purple	#800080
fuchsia	#FF00FF
green	#008000
lime	#00FF00
olive	#808000
yellow	#FFFF00
navy	#000080
blue	#0000FF
teal	#008080
aqua	#00FFFF

CARACTÈRES SPÉCIAUX

Nous pouvons internaliser notre texte afin de s'assurer de la bonne prise en compte des caractères spéciaux.

Pour cela, il faut utiliser l'entité HTML associé :

Caractère	Entité	Caractère	Entité	Caractère	Entité	Caractère	Entité
A	à	î	î	ç	ç	ù	ù
A	á	ï	ï	è	è	ú	ú
A	â	ð	ð	é	é	û	û
Ã	ã	ñ	ñ	ê	ê		
ä	ä	ô	ô	ë	ë		
æ	æ	ö	ö	ü	ü		

Caractère	Code ISO	Entité HTML
©	©	©
,	‚	
§	§	§
"	„	
...	…	
£	£	£
¶	¶	¶
.	·	·
¼	¼	¼
½	½	½
¾	¾	¾
€	€	€
‘	‘	
’	’	
“	“	
”	”	
•	•	
–	–	
—	—	
Espace insécable	 	
<	<	<
>	>	>

LES LISTES

Liste ordonnée

Balise

```
<ol>
|   <li></li>
</ol>
```

L'attribut type de la balise ol permet de choisir le type de l'ordonnancement.

```
<h3>liste ordonnée</h3>
<div>
|   <ol type="a">
|   |   <li>Pomme</li>
|   |   <li>Poire</li>
|   |   <li>Fraise</li>
|   </ol>
</div>
```

liste ordonnée

- a. Pomme
- b. Poire
- c. Fraise

Liste non ordonnée

Balise

```
<ul>
|   <li></li>
</ul>
```

L'attribut type de la balise ul permet de modifier le type de la puce

```
<h4>liste non ordonnées</h4>
<div>
|   <ul>
|   |   <li>Pomme</li>
|   |   <li>Poire</li>
|   |   <li>Fraise</li>
|   </ul>
</div>
```

liste non ordonnées

- Pomme
- Poire
- Fraise

LES LISTES

Liste de définitions

Balise

```
<dl>
  <dt></dt>
  <dd></dd>
</dl>
```

```
<h3>liste de définition</h3>
<div>
  <dl>
    <dt>les Listes</dt>
    <dd>la liste ol est ordonnée</dd>
    <dd>la liste ul est non ordonnée</dd>
    <dd>la liste dl sert à définir</dd>
  </dl>
</div>
```

liste de définition

les Listes
la liste ol est ordonnée
la liste ul est non ordonnée
la liste dl sert à définir

Combiner les listes

On peut bien sûr combiner nos différentes listes entre elles

```
<h3>liste combinée</h3>
<div>
  <ol type="i">
    <li>item1
      <dl>
        <dt>Terme</dt>
        <dd>def1</dd>
      </dl>
    </li>
  </ol>
</div>
```

liste combinée

i. item1
Terme def1

CONTENU

Les images

Balise ``

- Attention à la taille de nos images car source de ralentissement
- une photo de 120ko = 120000 caractères = 24 000 mots
- 3 formats pour le WEB : PNG, JPEG et GIF
- Préciser la taille en pixel de l'image avec `width` et `height` mais il est préférable de stocker les images dans le bon format et la bonne résolution et ainsi de s'éviter à jouer avec ces deux attributs.
- D'autres attributs de présentation sont disponibles
 - `Border`
 - `Hspace` et `vspace`
 - `align`

Exemple

Affichage d'une image avec une bordure et un espace vertical et horizontal

```
<div>
  
</div>
```



LES LIENS HYPERTEXTES

Balise `<a>`

- Crée des liens au sein de la page, entre deux pages, vers une page extérieure.
- Ses attributs sont :
 - **Href** : l'adresse cible
 - **Title** : description pour l'info-bulle
 - **Style** : style de présentation
 - **Target** : dans le cas de frame, définir le cadre dans lequel afficher
 - **Name** : définir un signet ou ancre à l'intérieur de la page HTML

Lien entre deux pages

- Lien relatif (lien interne) :
 - On ne connaît pas l'adresse complète de la page cible
 - `Cliquez vers la page 2`
 - `Cliquez ici`
- Lien absolu (lien externe) :
 - Il faut connaître l'adresse complète de la page cible
 - `site de l'AFPA`
- Lien email :
 - Il faut connaître l'adresse complète de la page cible
 - `Ecrivez-nous`
 - `Ecrivez-nous`

LES TABLEAUX

Balise `<table></table>`

Les attributs de cette balise sont :

- `<caption>` : titre du tableau
- `<colgroup>` : propriétés pour un groupe de colonnes.
- `<thead>` : en-tête du tableau
- `<tfoot>` : pied du tableau
- `<tbody>` : corps du tableau [`tr`, `th`, `td`]
- `<tr>` : ligne du tableau [`th`, `td`]
- `tr` : ligne du tableau. Il contient un ou plusieurs `td` et/ou `th`
- `td` : cellule qui contient des données.
- `th` : élément spécialisé qui contient le titre de la colonne.

```

<h5>les tableaux</h5>
<table>
  <caption>Exemple de tableau</caption>
  <thead>
    <tr>
      <th>Participant</th>
      <th>Score</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Rodolphe</td>
      <td>337</td>
    </tr>
    <tr>
      <td>Raphaël</td>
      <td>1000</td>
    </tr>
  </tbody>
  <tfoot>
    <tr>
      <td>Total</td>
      <td>1337</td>
    </tr>
  </tfoot>
</table>

```

les tableaux

Exemple de tableau

Participant Score

Rodolphe 337

Raphaël 1000

Total 1337

LES IFRAMES

Balise `<iframe></iframe>` permet d'ouvrir une fenêtre imbriquée à n'importe quel endroit de votre page.

Cette balise possède les attributs suivants :

- **src** : url de la page html qui sera affiché dans la iframe
- **name** : nom de la zone pour identifier la iframe par l'attribut target
- **marginwidth** : marge à droite et gauche en pixel
- **marginheight** : marge haute et basse en pixel
- **frameborder** : bordure ou pas
- **border** : bordure de la frame en pixel
- **bordercolor** : couleur de la bordure
- **noresize** : bloque le redimensionnement
- **scrolling** : scrolling de la page
- **height** et **width**

A utiliser avec précaution car en termes de sécurité, il faut se méfier d'intégrer une page extérieure dont on ne maîtrise pas le contenu

```
<iframe src="page1.html" align="left" width="200" height="200"  
scrolling="yes" frameborder="1"></iframe>
```

PAGE 1

Lorem ipsum dolor, sit amet consectetur adipisicing elit. Nisi earum quia, dolores amet, quis possimus iste modi blanditiis asperiores quas excepturi? Qui velit quisquam labore, animi commodi quae aperiam

QUELQUES ÉLÉMENTS SYMPA...

- La balise `<figure>` combinée avec `<figcaption>` permet de grouper et de donner un sens à une image et son titre ou sa légende.
- La balise `<details>` et `<summary>` permettent de reproduire un standard (quand supportée) de bloc à 2 états condensé/étendu.

```
<p>
  <figure>
    
    <figcaption>Un Chat</figcaption>
  </figure>
</p>
```



Un Chat

```
<footer>
  <details>
    <summary>Ici se trouve du texte</summary>
    <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Aliquid omnis aliquam distinctio aperiam corrupti pariatur temporibus, dolores consectetur tempore aspernatur vel sit, magnam neque. Ducimus illo sint nobis nostrum vitae.</p>
  </details>
</footer>
```

▼ Ici se trouve du texte
Lorem ipsum dolor sit amet conse

► Ici se trouve du texte

MULTIMEDIA

Vidéo

Balise `<video>`

- Cette balise comporte de nombreux attributs, dont les principaux :
 - `src` : URL du fichier vidéo.
 - `preload="auto"`, qui permet de spécifier au navigateur de débuter le téléchargement de la vidéo tout de suite
 - `poster` : affiche une image lorsque le média n'est pas disponible.
 - `autoplay="true"` : joue le fichier automatiquement.
 - `loop` : joue en boucle.
 - `controls` : affiche les contrôles

Attention, tous les navigateurs ne supportent pas tous les formats de vidéo.

Audio

Balise `<audio>`

- Cette balise comporte les attributs suivants :
 - `src` : URL du fichier audio.
 - `autoplay` : joue le fichier automatiquement.
 - `loop` : joue en boucle.
 - `controls` : affiche les contrôles.

CONCLUSION

Il faudrait beaucoup d'heures pour parcourir tout ce qui HTML 5 offre comme balises et possibilités pour construire votre document HTML.

C'est avec la pratique et la lecture de documentation qu'on enrichira notre connaissance des balises du langage.

<https://developer.mozilla.org/fr/>

<https://devdocs.io/>

CHAPITRE 3 Navigateurs et support..... 53 Panorama des navigateurs web et moteurs de rendu 54 Prise en charge de HTML 5 55 Bibliothèque de détection et de modernisation 56 Moderniser 56 html5shim (ou html5shiv) 58	download 88 Liens et blocs 88 Sections et titres 89 <i>Le cas Internet Explorer</i> 93 <i>Le cas Internet Explorer</i> <i>sans JavaScript</i> 96 <i>Le cas Internet Explorer</i> <i>sans JavaScript, b活着</i> 97 <section> 97	CHAPITRE 4 Éléments et attributs Modèles de contenu Le doctype avant tout Rappel des préfixes Éléments racines et m <i>html</i> 211 <i>manifest</i> 212 <i>dir</i> 213 <i>draggable</i> 213 <i>dropzone</i> 213 <i>hidden</i> 213 <i>id</i> 214 <i>itemid</i> , <i>itemref</i> , <i>itemscope</i> , <i>itemtype</i> 215 <i>itemprop</i> 215 <i>lang</i> 215 <i>tabindex</i> 216 <i>title</i> 218 <i>meta</i> 219 <i>spellcheck</i> 220 <i>style</i> 220 Attribut des liens 221 <i><meta charset</i> 379 <i><meta http-equiv</i> 380 <i><link</i> 382 <i><style></i> 382 <i><media</i> 383 <i><base</i> 384 <i>bref</i> 384 <i>target</i> 384 CHAPITRE 5 Les Groupement 385 <i><div></i> 385 <i></i> 386 <i><div></i> 387 <i>Liens</i> 389 <i><a></i> 390 <i>bref et brefx</i> 391 <i>rel</i> 391 <i>id et ancrs</i> 392 <i>target</i> 392	data 211 dir 212 draggable 213 dropzone 213 hidden 213 id 214 itemid, itemref, itemscope, itemtype 215 itemprop 215 lang 215 tabindex 216 title 218 meta 219 spellcheck 220 style 220 Attribut des formulaires 251 <i>datetime-local</i> 251 <i>month</i> 252 <i>week</i> 253 <i>number</i> 254 <i>range</i> 254 <i>color</i> 255 <i><datalist></i> 256 <i><textare></i> 257 <i><select></i> 258 <i><option></i> 259 <i><optgroup></i> 260 <i><button></i> 261 <i>autocorrect</i> 262 Autres éléments de formulaire 257 <i>Lire des fichiers avec FileReader</i> 426 <i>Utiliser Canvas</i> 430 <i>CreateObjectURL</i> 432 Contrôle clavier et souris 433 <i>Souris</i> 436 <i>Clavier</i> 439 Animation et jeux 439 <i>Jeu</i> 439 <i>Vidéo et audio</i> 439 Prise en charge 440 <i>Libraries</i> 440 <i>Et la 3D ?</i> 440 Et le graphisme vectoriel 440 <i>Création au form</i> 543 <i>Inclusion HTML</i> 544 <i>Syntaxe</i> 544 <i>Support</i> 544 <i>Alternatives et lib</i> 545 CHAPITRE 6 Géolocalisation.... <i>Principe</i> 545 <i>Les mains dans le cod</i> 545 <i>Déclencher la loc</i> 545 <i>Travailler avec la J</i> 545 <i>et les coordonnées</i> 545 <i>Gestion des erreurs</i> 545 <i>Options supplémentaires</i> 545 <i>Utiliser une carte</i> 545 <i>Prise en charge de l'AI</i> 545 <i>par les navigateurs</i> 545 <i>Alternative avec geor</i> 546	download 88 Liens et blocs 88 Sections et titres 89 <i>Le cas Internet Explorer</i> 93 <i>Le cas Internet Explorer</i> <i>sans JavaScript</i> 96 <i>Le cas Internet Explorer</i> <i>sans JavaScript, b活着</i> 97 <section> 97	CHAPITRE 7 Applications web hors ligne.... 543 <i>Principe</i> 544 <i>En ligne ou hors ligne?</i> 545 <i>Structure complète</i> 547 <i>Liste des fichiers à mettre en cache</i> 548 <i>(manifest)</i> 548 <i>Syntaxe pour le manifeste</i> 549 <i>La section CACHE</i> 550 <i>La section NETWORK</i> 550 <i>La section Fallback</i> 551 CHAPITRE 8 Historique de navigation..... 561 <i>Navigation dans l'historique</i> 562 <i>History</i> 562 <i>Location</i> 563 <i>Modification dynamique de l'historique</i> 564 <i>pushState()</i> 565 <i>replaceState()</i> 566 <i>The king of popstate</i> 567 <i>Simulation</i> 567 <i>Cas pratique</i> 568 <i>Réécriture d'adresse</i> 574 <i>Les ancrs et l'événement hashchange</i> 575 <i>Détection</i> 576 <i>Prise en charge</i> 576	CHAPITRE 9 JavaScript, le DOM et l'API Selectors..... 587 <i>Elaboration et test du cache</i> 587 <i>Les bases de JavaScript</i> 589 <i>Variables</i> 589 <i>Types simples</i> 590 <i>Objets</i> 591 <i>Fonctions</i> 592 <i>Boutons</i> 593 <i>Méthodes de sélection DOM</i> 594 <i>getElementsByClassName()</i> 594 <i>getElementsByName()</i> 595 <i>getElementsByTagName()</i> 595 <i>querySelector()</i> 595 <i>querySelectorAll()</i> 595 <i>Propriétés et méthodes DOM</i> 595 <i>Méthodes de manipulation DOM</i> 596 <i>createElement()</i> 596 <i>createAttribute()</i> 596 <i>appendChild()</i> 597 <i>removeChild()</i> 597 <i>insertBefore()</i> 597 <i>createTextNode()</i> 597 <i>Méthodes pour formulaires</i> 597 <i>Gestionnaires d'événements</i> 598 <i>Autres fonctions</i> 600 <i>Prise en charge</i> 600	CHAPITRE 10 Interactions avec l'API <i>Principe</i> 601 <i>Fonctionnement</i> 601 <i>Événement ondu</i> 601 <i>Recueillir les info</i> 601 <i>des fichiers sélects</i> 601	CHAPITRE 11 JavaScript en (multi)tâche de fond : les Web Workers..... 577 <i>Principe</i> 579	CHAPITRE 12 Événements envoyés par le serveur (« push »)..... 467 <i>Push-toi, j'arrive</i> 468 <i>Principe</i> 469 <i>Sous le capot</i> 469 <i>Côte client (navigateur)</i> 469 <i>Drag & Drop</i> 439 <i>Et bientôt, écrire des fichiers,</i> 440 <i>accéder au système</i> 440 <i>Prise en charge</i> 440	CHAPITRE 13 Microdata à la rescousse..... 290 <i>Attributs globaux et vocabulaires</i> 291 <i>itemscope</i> 291 <i>itemtype</i> 291 <i>Vocabulaires</i> 292 <i>itemprop</i> 294 <i>itemid</i> 295 <i>itemref</i> 296 API DOM Microdata 297 <i>document.getElementsByTagName()</i> 299 <i>itemType, itemRef, itemID</i> 300 <i>properties</i> 300 <i>properties.namodlem()</i> 300	CHAPITRE 14 Interface de contrôle et événements..... 171 <i>Impliquer avec <object> et éléments média</i> 173 <i><object></i> 174 <i>Le cas de Flash</i> 176 <i><param></i> 177 <i><video></i> 178 <i><audio></i> 178 <i>
</i> 179	CHAPITRE 15 Stockage des données locales (Web Storage)..... 509 <i>Deux espaces de stockage</i> 511 <i>Stockage de session</i> 511 <i>Local Storage</i> 512 <i>Mise en place d'un flux continu</i> 471 <i>Syntaxe des messages source</i> 475	CHAPITRE 16 Aller plus loin..... 507 <i>Prise en charge</i> 507
ANNEXE A Fonctionnalités modifiées et obsolètes..... 605 <i>Differences HTML 5 par rapport à HTML 4</i> 605 <i>Fonctionnalités obsolètes</i> 605 <i>Fonctionnalités obsolètes non conformes</i> 606 <i>Eléments</i> 606 <i>Attributs</i> 606	ANNEXE C Accessibilité et ARIA..... 623 <i>Qu'est-ce que l'accessibilité du Web ?</i> 624 <i>HTML, sémantique</i> 627 <i>WAI, WCAG et ARIA</i> 628 <i>Les rôles et propriétés de WAI-ARIA</i> 631 <i>Rôles avec l'attribut role</i> 632 <i>Points de repère (landmark roles)</i> 632 <i>Structure de document</i> 634 <i>Composants graphiques (widget)</i> 635 <i>Propriétés et états avec les attributs aria-</i> 638 <i>Globaux</i> 639 <i>Contrôles d'interface</i> 640 <i>Il y a de la magie sur cette planète</i> 643 <i>Drag & drop (gérer-déposer)</i> 645 <i>Relations</i> 645 <i>Valider et tester</i> 647 <i>L'aide de JavaScript</i> 649	ANNEXE B Feuilles de style CSS..... 609 <i>Principe général</i> 611 <i> Sélecteurs</i> 612 <i>Propriétés</i> 613 <i>Pseudo-classes et pseudo-éléments</i> 619 <i>Règles @</i> 620 <i>Media queries</i> 621	Index..... 651										



LES FORMULAIRES SAISIE DE DONNÉES

LES FORMULAIRES

Un des éléments importants en HTML sont les formulaires.

Ils permettent de mettre en place la saisie et l'envoie des données saisies depuis son navigateur vers le serveur Web.

Les saisies s'effectuent par des éléments graphiques HTML comme des champs de saisie, des zones de texte multiligne, des listes de choix déroulantes, des cases à cocher ou des radio-boutons et bien entendu des boutons de commandes ;

HTML5 apporte de nouveaux éléments graphiques permettant une meilleure ergonomie et un meilleur contrôle de saisie (saisie de nombres, de dates, d'adresse email, de couleur, ou encore curseur et barre de progression)

La balise `<form> ... </form>` permet d'encadrer le formulaire.

Cette balise possède les attributs suivants :

- **action** : indique l'url du script serveur qui doit exécuter lors de l'envoi du formulaire
- **method** : get ou post
- **enctype** : format des données envoyées [text/plain = données simple] ou [multipart/form-data = formulaire avec fichier envoyé]
- **name** et **id** : nom du formulaire et son identifiant - utile pour des contrôles de saisie en script

LES FORMULAIRES

Label

Le **label** permet d'indiquer un libellé explicatif à une zone de saisie.

Cela permet de donner du sens et de le lier avec **for** à la zone de saisie identifié par son id.

```
<label for="nom">Nom</label>
```

Encadré et regroupement de zones

Au sein d'un formulaire, on peut réaliser un encadré avec **fieldset** pour regrouper certains éléments qui ont un sens commun.

Cet élément a un but de représentation et n'est pas porteur de données en lui-même.

Pour finir, il est judicieux de lui donner un nom par la balise **legend**

```
<fieldset>
|   <legend>Input</legend>
```

LES FORMULAIRES

Les champs de saisie

Balise `<input type="text" ... />`

- `name` et `id` : nom et identifiant de la donnée
- `required` : obligatoire
- `disabled` : active ou pas
- `pattern` : règle de validation
- `placeholder` : texte d'invite
- ...

```
<label for="nom">Nom</label>
<input type="text" name="nom" id="nom"
       placeholder="votre nom" required/>
```

Zone de saisie multilignes

La balise `<textarea>...</textarea>` permet de mettre en place une zone de texte à la saisie.

- `name` et `id` : nom et identifiant de la donnée
- `required` : obligatoire
- `disabled` : active ou pas
- `pattern` : règle de validation
- `placeholder` : texte d'invite

```
<label for="zonetext">Texte</label>
<textarea name="zonetext" id="text" cols="30" rows="10"
          placeholder="ceci est un texte"></textarea>
```

LES FORMULAIRES

Fichier à joindre

`<input type="file" ... />` permet d'activer l'explorateur pour sélectionner un fichier à joindre.

```
<fieldset>
    <legend>Fichier à joindre</legend>
    <p>Sélectionner le fichier à envoyer : <input type="file"
        name="fichier" id="fichier" size="50" /></p>
</fieldset>
```

A screenshot of a web browser showing a file input field. The legend above the field says "Fichier à joindre". Below it, the text "Sélectionner le fichier à envoyer :" is followed by a button labeled "Choisir un fichier" and the file path "MySQL Exos.pptx".

Données masquées

`<input type="hidden" ... />` permet de masquer un champs de saisie. Cela peut être utile lorsqu'on souhaite faire apparaître des champs de saisie en fonction d'options etc...

LES FORMULAIRES

Cases à cocher et boutons radio

`<input type ="checkbox"/>`

- pour les cases à cocher.

`<input type ="radio"/>`

- pour les boutons radio.

Ces balises possèdent les attributs suivants :

- **name** et **id** : nom et identifiant de la donnée
- **value** : valeur correspondant à l'option à choisir
- **checked** : sélectionnée par défaut
- **disabled** : active ou pas

Listes de choix (déroulantes ou non)

`<select>` permet de mettre en place une liste de choix possibles. Chaque choix sera défini par une balise `<option>`

- **name** et **id** : nom et identifiant de la donnée
- **size** : permet de définir liste de choix ou liste déroulante.
- **multiple** : si liste déroulante, autorise plusieurs valeurs
- **value** : valeur de l'option sélectionnée par l'utilisateur

La balise `<option>`

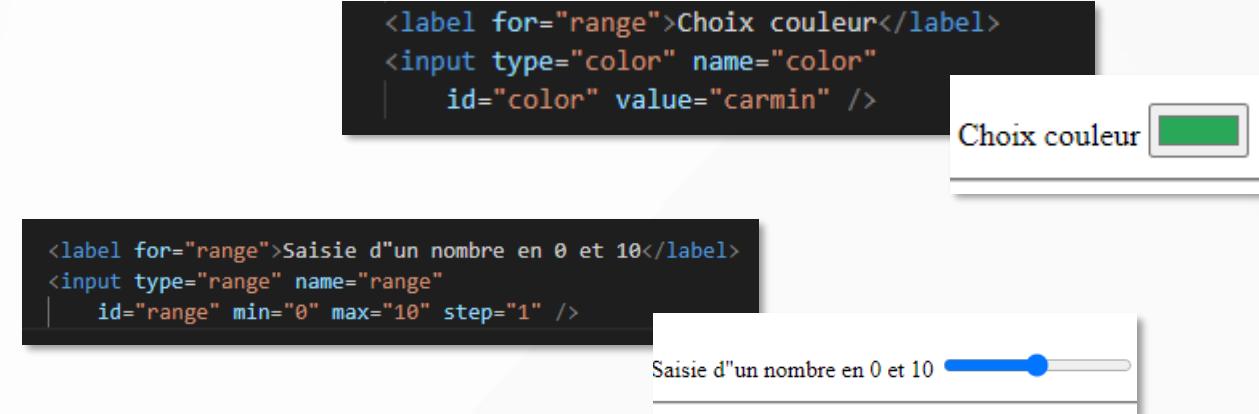
- **value** : valeur pour l'option correspondante
- **selected** : détermine le choix proposé par défaut.

LES FORMULAIRES

Saisie assistée

HTML 5 permet de sécuriser la saisie de données courantes. Sur la balise `<input>`, on peut indiquer un type spécifique pour notre champs de saisie.

- Email : `type="email"`
- Téléphone : `type="tel"`
- Dates : `type="date/week/month/time/datetime"`
- Nombre : `type="number/range"`
- Mot de passe : `type="password"`
- Couleur : `type="color"`

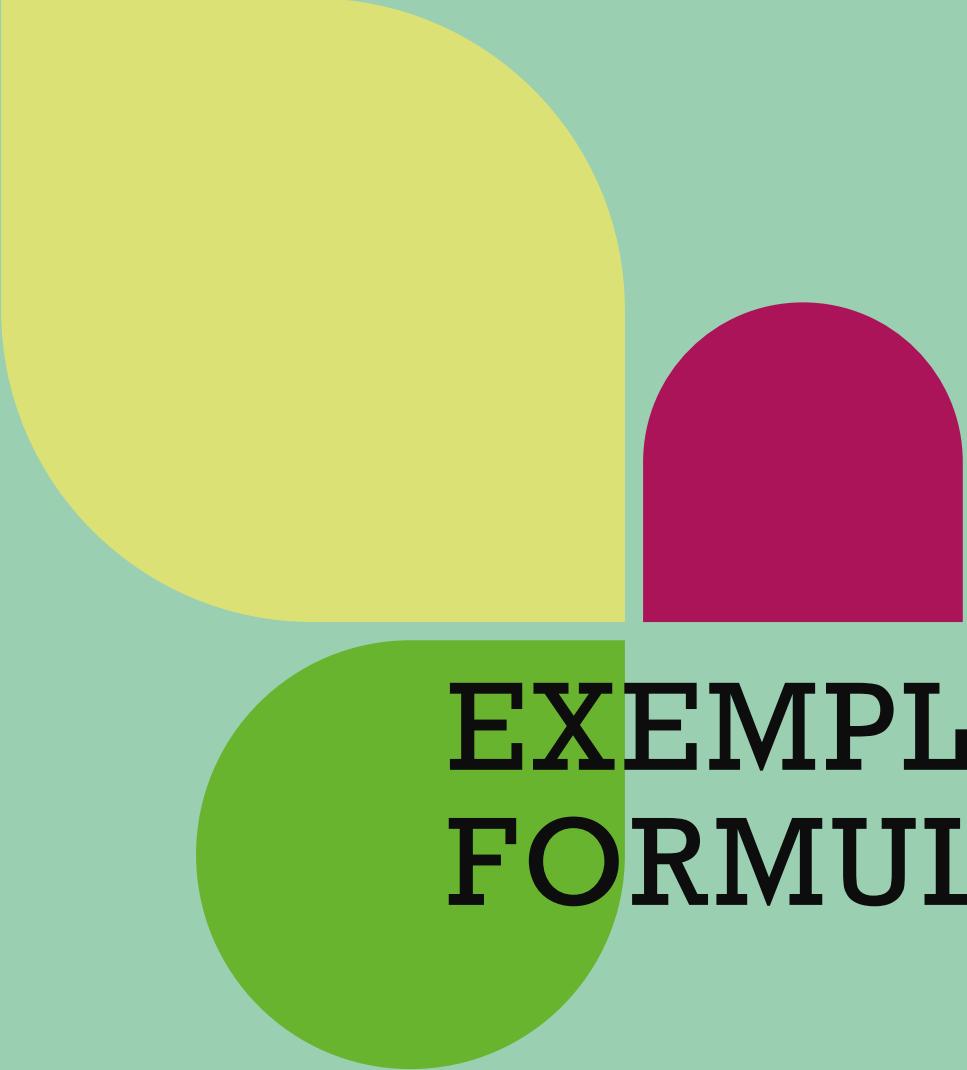


Soumission et Reset

La soumission du formulaire déclenche l'appel du module de traitement serveur associé, défini par l'attribut `action` indiqué dans la balise `form` et communique toutes les informations placées entre la balise `form` : `<input type="submit" ...>`

Pour réinitialiser le formulaire, `<input type="reset" ... />`

Enfin, on peut aussi utiliser le type `image` pour réaliser un bouton de soumission



EXEMPLE DE FORMULAIRE

EXEMPLE DE FORMULAIRE SANS MISE EN FORME

Nom :

Prénom :

Email :

Téléphone :

Envoyer

A noter dans cet exemple :

- *le required pour obliger la saisie du champ*
- *Pattern pour indiquer un pattern de contrôle à la saisie*
- *title pour le message d'informatif à la mauvaise saisie*

```
<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-Z]{2,}$"
                title="Veuillez entrer une adresse email valide">
        </div>
        <div>
            <label for="telephone">Téléphone :</label>
            <input type="tel" id="telephone" name="telephone"
                pattern="^\d{10}$"
                title="Veuillez entrer un numéro de téléphone valide (10 chiffres)">
        </div>
        <div>
            <button type="submit">Envoyer</button>
        </div>
    </form>
</body>
</html>
```



RÉCAPITULATIF DES BALISES HTML

TABLEAU DES PRINCIPALES BALISES HTML

les principales que l'on rencontre sur le Web

Balise	Description / Fonction
<html> ... </html>	Encadre l'ensemble du document HTML (contenu visible et invisible).
<head> ... </head>	Contient les informations techniques de la page (titre, encodage, liens, scripts, etc.).
<body> ... </body>	Contient tout le contenu affiché dans le navigateur (textes, images, vidéos...).
<link />	Lie une feuille de style CSS externe à la page.
<meta />	Ajoute des métadonnées (charset, auteur, description, viewport...).
<script> ... </script>	Intègre du code JavaScript à la page.
<style> ... </style>	Ajoute du code CSS interne à la page.
<title> ... </title>	Définit le titre de la page (visible dans l'onglet du navigateur).
<abbr>	Signale une abréviation, avec un attribut title (infobulle).
<blockquote>	Insère une citation longue mise en forme distinctement.
<q>	Insère une citation courte (entre guillemets automatiquement).

TABLEAU DES PRINCIPALES BALISES HTML

les principales que l'on rencontre sur le Web

Balise	Description / Fonction
<cite>	Indique la source d'une œuvre, d'un événement ou d'une publication.
<sub>	Texte en indice (ex : formule chimique).
<sup>	Texte en exposant (ex : puissance mathématique).
<h1> ... <h6>	Hiérarchie des titres : <h1> étant le plus important.
	Insère une image avec les attributs src (source) et alt (texte alternatif).
<mark>	Surligne du texte (mise en évidence).
	Texte important (gras, signification sémantique forte).
	Texte à souligner par l'italique (emphase, accentuation).
<figure>	Encapsule du contenu comme des images, du code, des graphiques.
<figcaption>	Ajoute une légende à une <figure>.

TABLEAU DES PRINCIPALES BALISES HTML

les principales que l'on rencontre sur le Web

Balise	Description / Fonction
<audio> ... </audio>	Permet d'intégrer un lecteur audio.
<video> ... </video>	Permet d'intégrer un lecteur vidéo.
<source>	Définit les sources pour <audio> ou <video> (formats alternatifs).
<a> ... 	Crée un lien hypertexte à l'aide de l'attribut href.
 	Effectue un simple saut de ligne (sans nouveau paragraphe).
<p> ... </p>	Définit un paragraphe de texte.
<hr />	Ajoute une ligne horizontale (séparateur visuel).
<address>	Indique les coordonnées de contact d'un auteur ou d'un site.

TABLEAU DES PRINCIPALES BALISES HTML

les principales que l'on rencontre sur le Web

Balise	Description / Fonction
	Indique un contenu supprimé (souvent barré).
<ins>	Indique un contenu ajouté (souvent souligné).
<dfn>	Balise de définition pour un terme introduit.
<kbd>	Utilisé pour représenter une entrée clavier utilisateur.
<progress>	Affiche une barre de progression (ex. : chargement).
<time>	Indique une date ou une heure (machine-readable).
<pre> ... </pre>	Affiche du texte formaté (code, indentation préservée).

BALISES DE STRUCTURATION DU TEXTE

Ces balises permettent d'organiser, de mettre en forme et d'enrichir le contenu visible sur une page HTML.

Balise	Description / Fonction
<abbr>	Indique une abréviation. L'attribut title peut afficher le terme complet au survol.
<blockquote>	Utilisée pour une citation longue, souvent indentée visuellement.
<q>	Pour les citations courtes. Les navigateurs ajoutent automatiquement des guillemets.
<cite>	Utilisée pour indiquer la source d'une œuvre ou d'un auteur (livre, film, article...).
<sub>	Texte affiché en indice (ex : H ₂ O).
<sup>	Texte en exposant (ex : x ²).
<h1> ... <h6>	Balises de titre hiérarchisées, du plus important (h1) au moins important (h6).
	Affiche une image. Nécessite l'attribut src (URL) et alt (texte alternatif).
<mark>	Met en évidence une portion de texte avec un surlignage.
	Texte à forte importance, souvent affiché en gras.
	Texte mis en emphase, affiché en italique. Peut avoir un sens sémantique (voix, accent, etc.).

BALISES DE STRUCTURATION DU TEXTE

Ces balises permettent d'organiser, de mettre en forme et d'enrichir le contenu visible sur une page HTML.

Balise	Description / Fonction
<figure>	Utilisé pour grouper un média (image, graphique, code) avec sa légende.
<figcaption>	Contient la légende associée à une balise <figure>.
<audio>	Intègre un lecteur audio HTML natif dans la page.
<video>	Permet d'intégrer une vidéo HTML5 avec contrôles.
<source>	Définit une source alternative pour une balise <audio> ou <video>.
<a>	Crée un lien hypertexte vers une URL grâce à l'attribut href.
 	Insère un simple retour à la ligne sans créer de nouveau paragraphe.
<p>	Définit un paragraphe. Chaque bloc de texte distinct devrait être dans une balise <p>.

BALISES DE STRUCTURATION DU TEXTE

Ces balises permettent d'organiser, de mettre en forme et d'enrichir le contenu visible sur une page HTML.

Balise	Description / Fonction
<hr />	Insère une ligne de séparation horizontale. Utilisée comme rupture thématique.
<address>	Indique les coordonnées d'un auteur ou d'un site (adresse email, postale...).
	Marque un texte supprimé. Visuellement barré dans la plupart des navigateurs.
<ins>	Marque un texte ajouté (souvent affiché souligné).
<dfn>	Signale une définition. Peut être stylisée différemment par le navigateur.
<kbd>	Utilisée pour du texte à taper au clavier (ex : raccourcis, commandes terminal).
<progress>	Affiche une barre de progression interactive avec une valeur.
<time>	Spécifie une date, une heure ou une durée lisible par une machine (utile pour les moteurs).
<pre>	Affiche du texte formaté avec espaces, tabulations et sauts de ligne respectés (souvent pour du code).

BALISES SECTIONNANTES

Ces balises permettent de structurer les grandes zones d'un site web. Elles donnent un sens sémantique aux différentes parties de la page, ce qui aide aussi bien les développeurs que les moteurs de recherche à comprendre la hiérarchie du contenu.

Balise	Description / Fonction
<code><header> ... </header></code>	Représente l'en-tête d'une page ou d'une section. Contient souvent le logo, le titre, le menu principal.
<code><nav> ... </nav></code>	Zone de navigation principale contenant des liens vers d'autres pages ou sections du site.
<code><footer> ... </footer></code>	Bas de page ou de section. Contient généralement les mentions légales, informations de contact, copyright, etc.
<code><section> ... </section></code>	Représente une section thématique du document, souvent avec un titre (<code><h2></code> , etc.).
<code><article> ... </article></code>	Bloc autonome et réutilisable de contenu (ex : un article de blog, une fiche produit, une actualité).
<code><aside> ... </aside></code>	Contenu complémentaire, souvent sous forme de sidebar (barre latérale), widgets ou encarts.

BALISES GÉNÉRIQUES

Ces balises n'ont pas de signification sémantique particulière.

Balise	Description / Fonction
<code> ... </code>	<p>Balise inline sans signification sémantique. Utilisée pour styliser une portion de texte dans un paragraphe.</p> <ul style="list-style-type: none">• Ne provoque pas de saut de ligne.• Ne se redimensionne pas avec width ou height.• Respecte les marges gauche/droite mais pas haut/bas.
<code><div> ... </div></code>	<p>Balise block générique, souvent utilisée pour regrouper et structurer des blocs de contenu.</p> <ul style="list-style-type: none">• Provoque un retour à la ligne automatique.• Accepte les propriétés CSS de width et height.• Respecte toutes les marges (haut, bas, gauche, droite).

BALISES DE LISTES

Ces balises permettent de créer des listes structurées dans vos pages HTML. Il en existe trois types : les listes non ordonnées (à puces), les listes ordonnées (numérotées), et les listes de définitions (terme + description).

Balise	Description / Fonction
 ... 	Liste non ordonnée (à puces). Chaque élément est défini par une balise .
 ... 	Liste ordonnée (numérotée automatiquement).
 ... 	Élément d'une liste, utilisé à l'intérieur des balises ou .
<dl> ... </dl>	Liste de définitions. Permet de structurer des paires « terme + définition ».
<dt> ... </dt>	Terme défini dans une <dl>.
<dd> ... </dd>	Description ou définition du terme introduit par <dt>.

BALISES DE TABLEAU

Ces balises permettent de créer des tableaux HTML avec une structure logique en lignes (`<tr>`) et cellules (`<td>` pour les données, `<th>` pour les en-têtes)

Balise	Description / Fonction
<code><table> ... </table></code>	Balise principale qui encadre l'ensemble du tableau.
<code><caption> ... </caption></code>	Titre du tableau. S'affiche généralement au-dessus du tableau.
<code><tr> ... </tr></code>	Définit une ligne du tableau.
<code><th> ... </th></code>	Cellule d'en-tête (souvent en gras et centrée).
<code><td> ... </td></code>	Cellule de données standard.
<code><thead> ... </thead></code>	Encadre l'en-tête du tableau (lignes de titres).
<code><tbody> ... </tbody></code>	Encadre le corps principal du tableau (les données).
<code><tfoot> ... </tfoot></code>	Encadre le pied de tableau (utilisé pour les totaux, remarques, etc.).

BALISES DE FORMULAIRE

Ces balises permettent de créer des formulaires HTML

Balise	Description / Fonction
<form> ... </form>	Encadre tout le formulaire. Utilise les attributs : <ul style="list-style-type: none">• method : méthode d'envoi (souvent post ou get)• action : URL de traitement des données (script ou page de destination)
<fieldset> ... </fieldset>	Permet de regrouper plusieurs champs dans une même section logique.
<legend> ... </legend>	Titre ou légende associée à un <fieldset>.
<label> ... </label>	Texte descriptif lié à un champ de formulaire. Peut être associé à un champ via l'attribut for.
<input />	Champ de formulaire à usage unique (texte, bouton radio, case à cocher, bouton, etc.). Nécessite l'attribut type pour définir la nature du champ : <ul style="list-style-type: none">• type="text" : champ texte simple• type="email" : champ pour adresse mail• type="checkbox" : case à cocher• type="radio" : bouton radio• type="submit" : bouton d'envoi• type="password" : champ de mot de passe masqué
<textarea> ... </textarea>	Zone de saisie multiligne, configurable avec rows et cols.
<select> ... </select>	Liste déroulante permettant de choisir une ou plusieurs options.
<option> ... </option>	Élément d'une liste déroulante (valeur affichée ou envoyée).
<optgroup> ... </optgroup>	Permet de regrouper plusieurs options dans une liste déroulante avec une étiquette commune.

ATTRIBUTS DES BALISES GÉNÉRIQUES

Ces attributs permettent d'identifier, de styliser ou de manipuler les balises avec du CSS ou du JavaScript.

Attribut	Description / Fonction
class	<p>Spécifie une ou plusieurs classes CSS associées à la balise.</p> <ul style="list-style-type: none">• Permet de styliser plusieurs éléments avec le même nom de classe.• Peut être utilisée pour cibler des éléments dans une feuille de style ou en JavaScript.• Syntaxe CSS associée : <code>.nomdelaclass { ... }</code>
id	<p>Attribut unique servant à identifier un élément précis dans la page.</p> <ul style="list-style-type: none">• Ne peut apparaître qu'une seule fois par page HTML.• Utile pour créer des ancrés, pour appliquer un style CSS spécifique ou pour manipuler l'élément via JavaScript.• Syntaxe CSS associée : <code>#nomdelid { ... }</code>
style	<p>Permet d'appliquer directement du CSS en ligne, dans la balise HTML.</p> <ul style="list-style-type: none">• À utiliser uniquement pour des tests ou des styles ponctuels.• Exemple : <code><div style="color: red;">Texte rouge</div></code>• À éviter pour des raisons de maintenance : privilégiez les feuilles de style externes.

Feuille de Style CSS

Mettre du style à vos pages Web



Historique

Feuille de Style



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

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

Style

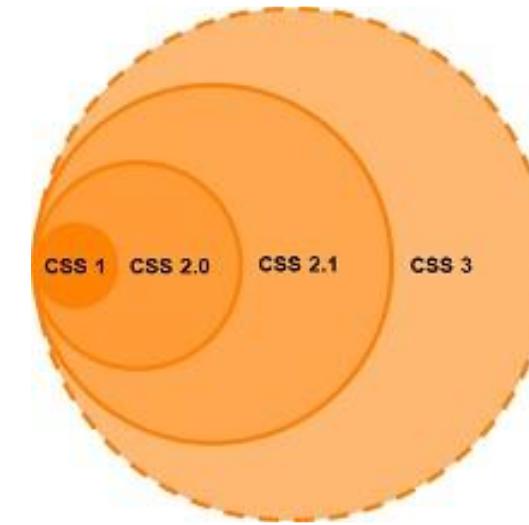
Sélecteur

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



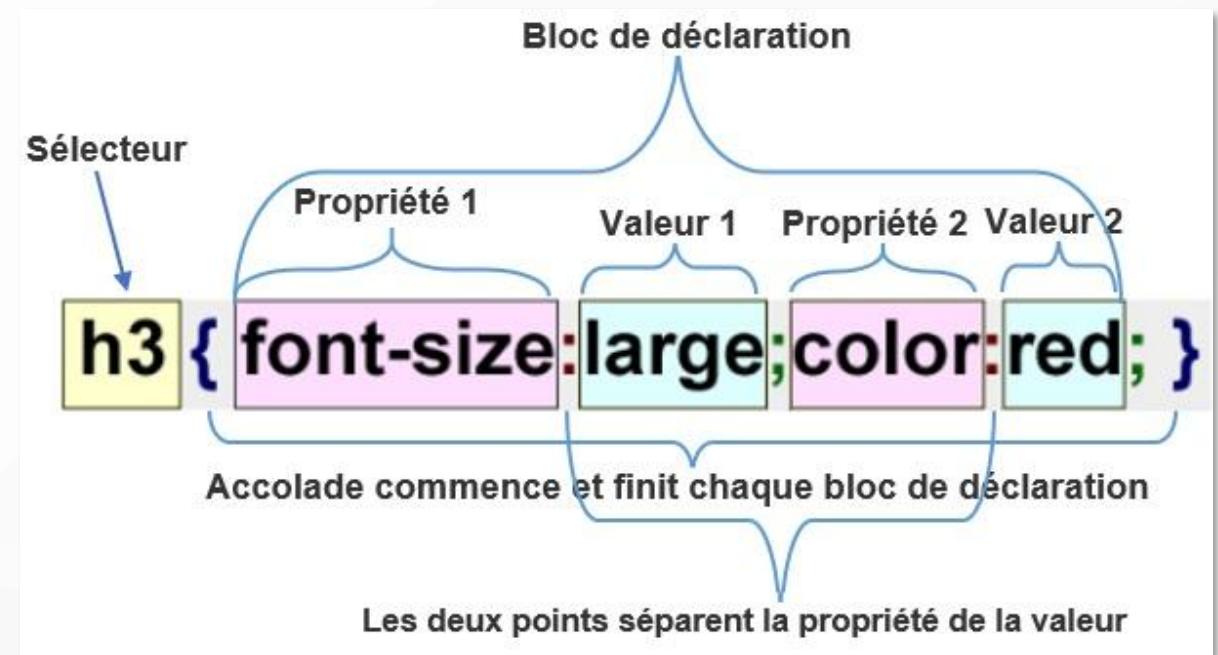
DÉFINITION

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

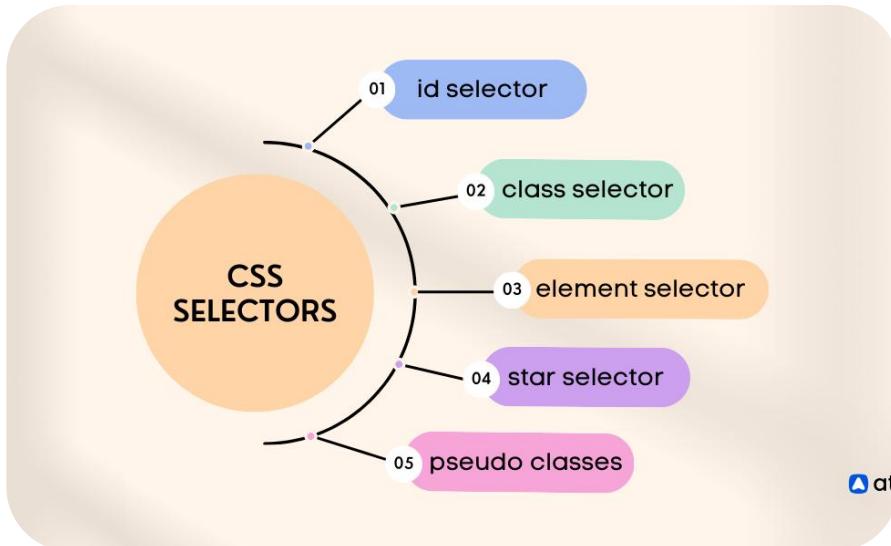
```
<link rel="stylesheet" type="text/css" href="ressources/style.css">
```

```
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



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 1^{er} enfant
- Pseudo-classe :nth-child(n) n-ième enfant
- ...

```
<body>
  <h1>Test Combinateurs</h1>

  <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>

  <div>
    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>
    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>
  </div>
  <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>
  <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>
</body>
<html>
```

```
body p:nth-child(2) {
  background-color: #purple;
  color: #white;
}
```

Test Combinateurs

Lore*m ipsum dolor sit, amet consectetur adipiscin
temporibus, mollitia dolorem nobis fuga incidunt.*

Lore*m, ipsum dolor sit amet consectetur adipiscin
maxime a consequuntur dignissimos fugiat?*

Lore*m, ipsum dolor sit amet consectetur adipiscin
maxime a consequuntur dignissimos fugiat?*

Lore*m ipsum dolor sit amet consectetur, adipiscin
Repellendus doloremque consequatur ex minima e*

Lore*m, ipsum dolor sit amet consectetur adipiscin
maxime a consequuntur dignissimos fugiat?*

Lore*m ipsum dolor sit amet consectetur adipiscin
eligendi nostrum asperiores laudantium architecto*

Lore*m ipsum dolor sit amet consectetur adipiscin
similique nisi. Id voluptate consequuntur, pariatur i*

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

```
<body>
    <h1>Test Combinateurs</h1>

    <p>Lorem ipsum dolor si
    esse similique ad volupt
    veniam suscipit alias de
    incidunt.</p>

    <div>

        <p>Lorem, ipsum dol
        Dolorem, optio quis
        voluptas quae ducim
        consequuntur digniss

        <p>Lorem, ipsum dol
        Dolorem, optio quis
        voluptas quae ducim
        consequuntur digniss

        <div>

            <p>Lorem ipsum c
            Vero eveniet mod
            quaerat ratione
            doloremque conse

            <p>Lorem, ipsum
            Dolorem, optio
            voluptas quae d
            consequuntur di

        </div>

    </div>

    <p>Lorem ipsum dolor si
    corporis neque quibusdan
    fugit nihil, repellat no
    architecto id.</p>

    <p>Lorem ipsum dolor si
    provident harum ducimus
    et officiis distinctio
    pariatur iste illum acc

</body>
<html>
```

```
p::first-line {  
    color: blue;  
}  
  
body p:nth-child(2) {  
    background-color: purple;  
    color: white;  
}
```

Test Combinateurs

Lore ipsum dolor sit, amet consectetur adi
temporibus, mollitia dolorem nobis fuga in

Lore, ipsum dolor sit amet consectetur adi
maxime a consequuntur dignissimos fugiat?

Lore, ipsum dolor sit amet consectetur adi
maxime a consequuntur dignissimos fugiat?

Lore ipsum dolor sit amet consectetur, adi
Repellendus doloremque consequatur ex mi

Lore, ipsum dolor sit amet consectetur adi
maxime a consequuntur dignissimos fugiat?

Lore ipsum dolor sit amet consectetur adi
eligiendi nostrum asperiores laudantium arch

Lore ipsum dolor sit amet consectetur adi
similique nisi. Id voluptate consequuntur, pa

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.

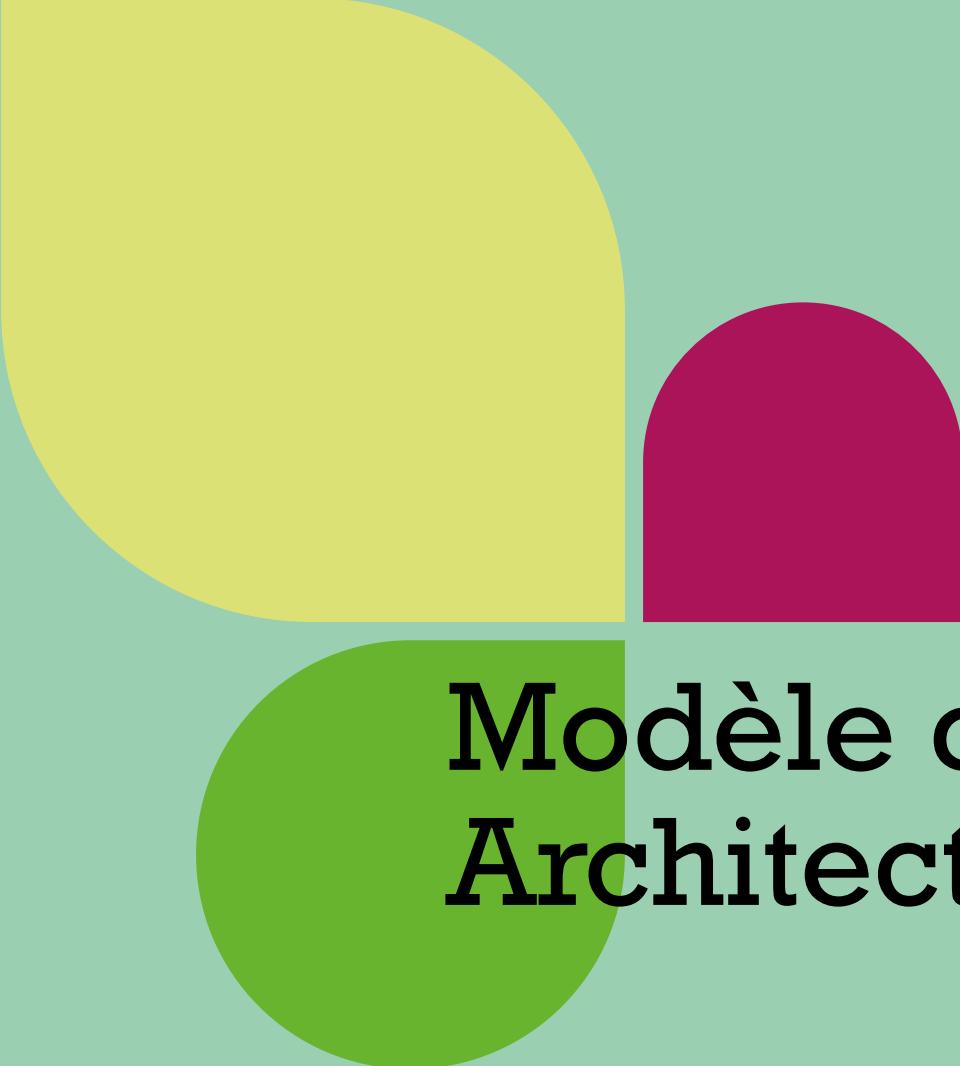
- **cm** centimètre
- **mm** millimètre
- **in** pouce, c'est environ 2,54 cm
- **px** pixel, initialement 1 px est égal à 1/96 ème de pouce
- **pt** point, initialement 1 pt est égal à 1/72 ème de pouce
- **pc** pica, 1 pc est égal 12 pt

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 = $2\pi = 6,28$ 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

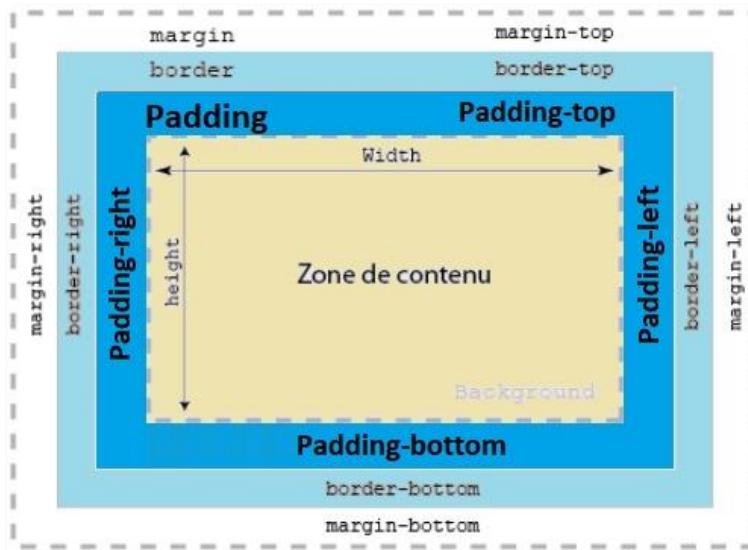


Modèle de boîtes Architecture de boîtes

Définition

Le **Box Model** est la manière dont les navigateurs doivent afficher les boîtes 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.



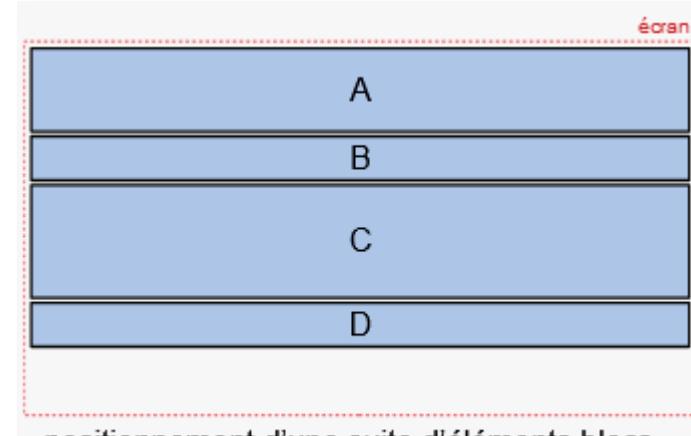
Positionnement

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.



inline

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



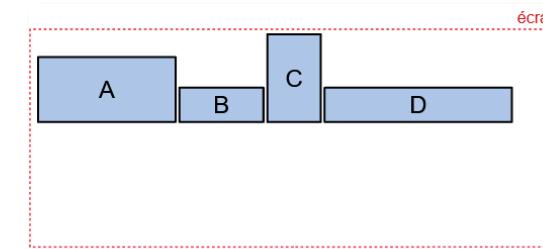
Positionnement

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

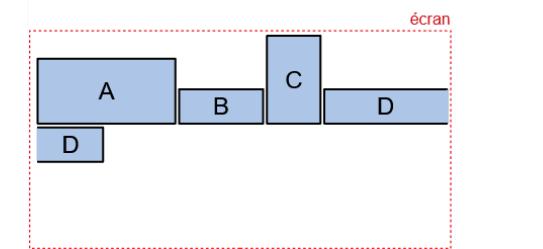


Dimension

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)



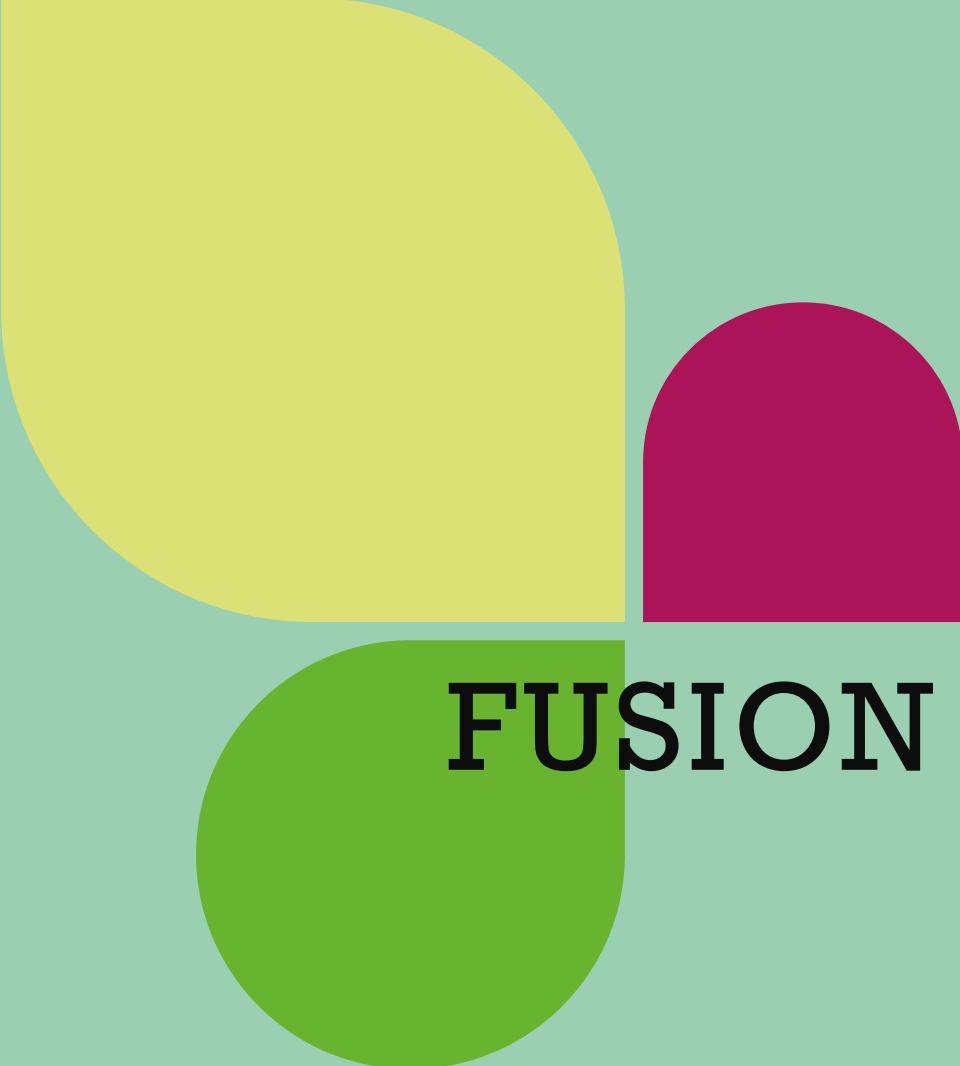
Grid

Modifie le comportement du parent mais aussi des enfants



Flex

Modifie le comportement du parent mais aussi des enfants



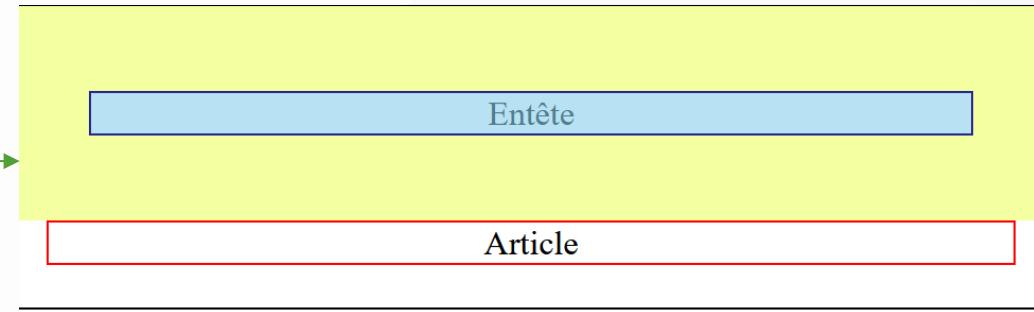
FUSION DES MARGES

FUSION DES MARGES

La fusion des marges extérieures (`margin`) fait que deux marges contigües, 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 contenu	bloc
largeur	<p>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.</p> <p>Il n'est pas possible d'en fixer la valeur.</p>	<p>En automatique, la boîte occupe toute la largeur (équivalent à width:100%)</p> <p>La largeur peut être fixée.</p>
hauteur	<p>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.</p> <p>Il n'est pas possible d'en fixer la valeur.</p>	<p>En automatique, la hauteur épouse le contenu de la boîte, y compris les marges et les bordures.</p> <p>La hauteur peut être fixée.</p>

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.

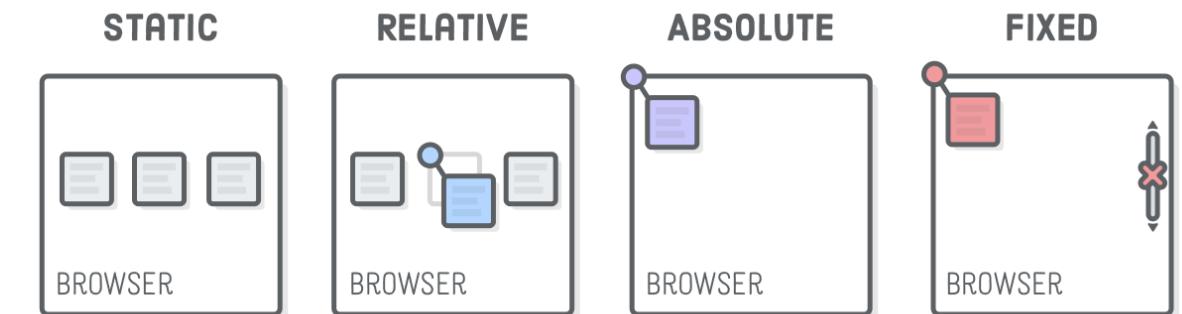


Static

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

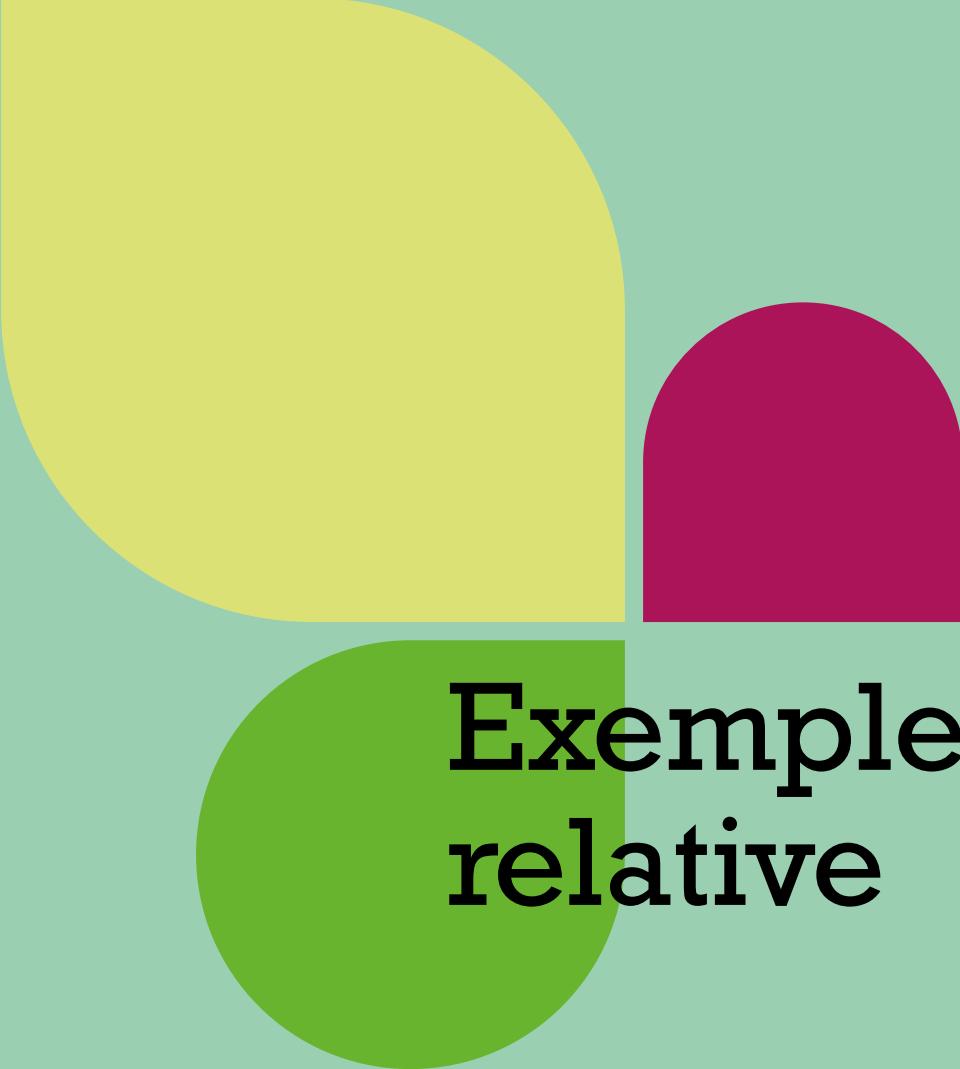
Le positionnement **relatif** permet d'inscrire un contenu en flux normal, puis de le décaler horizontalement et/ou verticalement.

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

```
<p>flux normal <span class="jaune">position relative</span>  
| Suite du flux normal</p>
```

```
.jaune {  
    position: relative;  
    bottom: 5px;  
    background-color: yellow;  
}
```

```
oflux normal position relative Suite du flux normal→
```



Exemple de position
relative

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

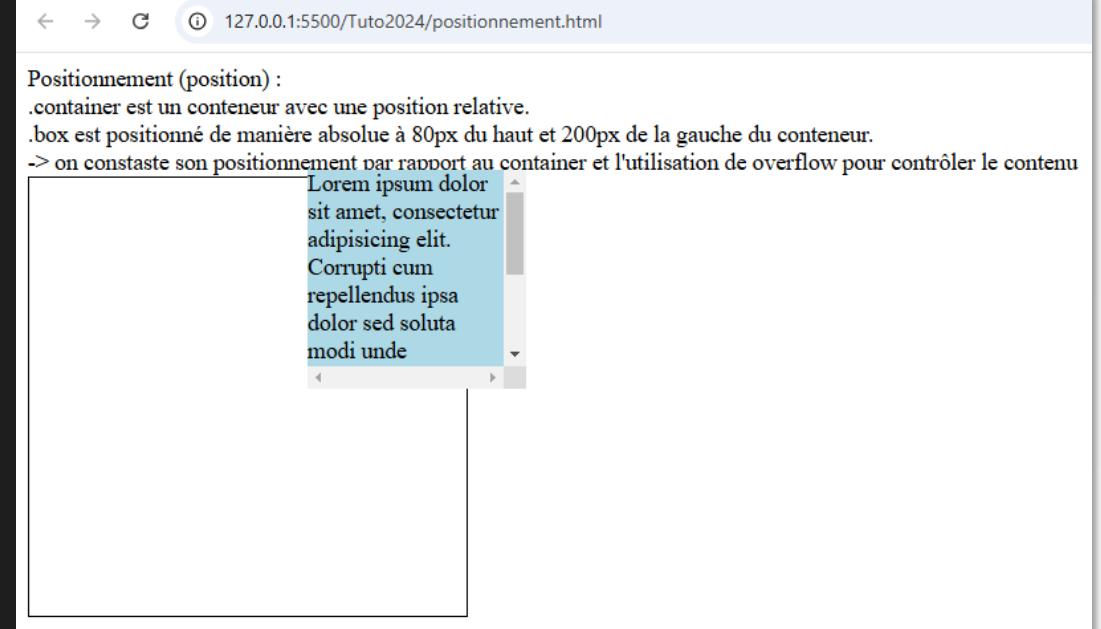
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Exemple de Positionnement</title>
    <style>
        .container {
            position: relative;
            width: 300px;
            height: 300px;
            border: 1px solid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

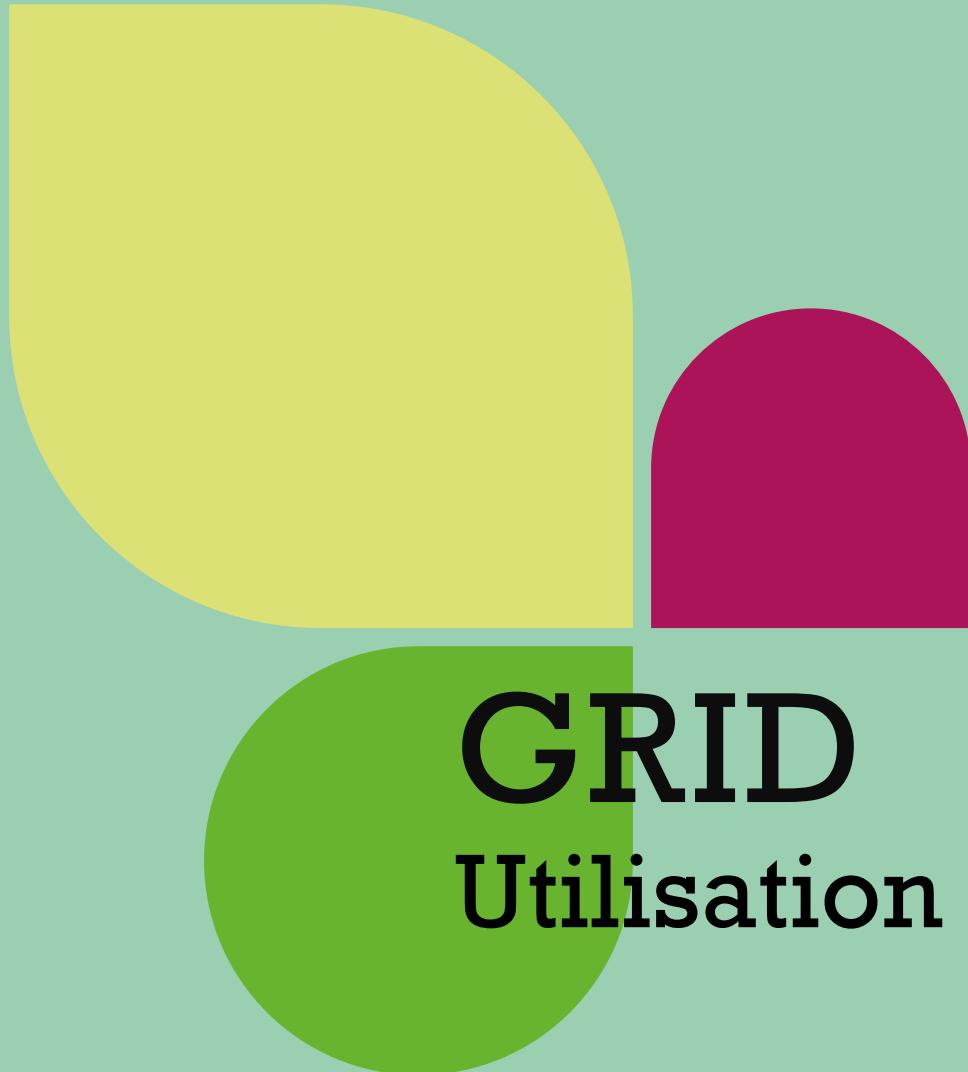
        .box {
            position: fixed;
            top: 80px;
            left: 200px;
            width: 150px;
            height: 150px;
            background-color: lightblue;
            overflow: scroll;
        }
    </style>
</head>

<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 constate 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 preferendis 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;
  }

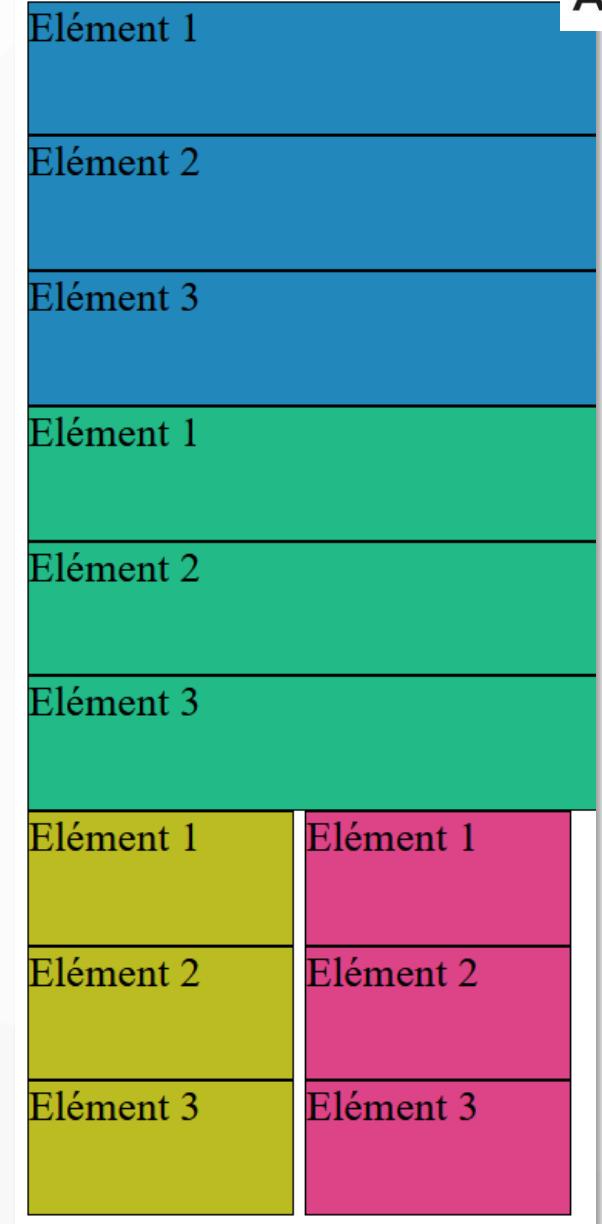
  .inline-container{
    display: inline-grid;
  }

  .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 forcément 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.

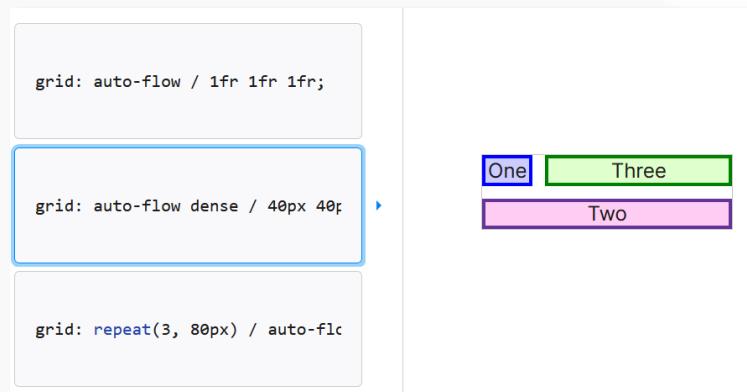
RACCOURCIE

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`).





Un exemple de GRID

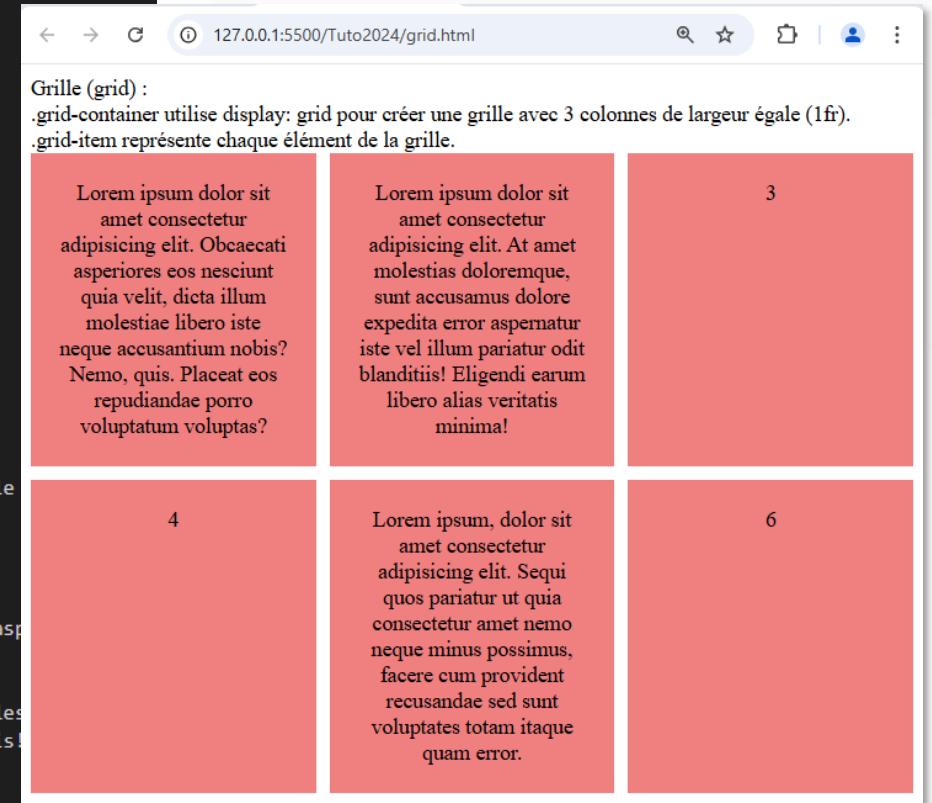
```

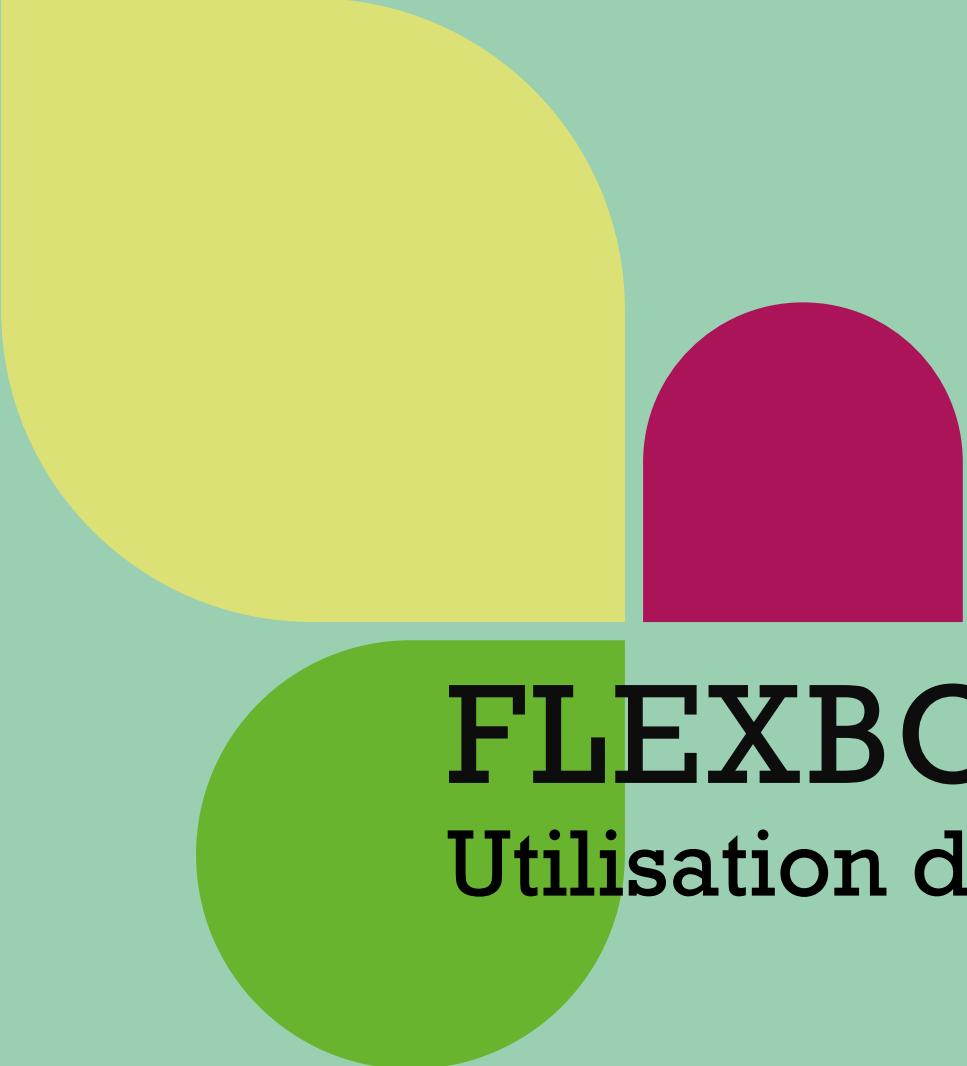
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Exemple de Grille</title>
    <style>
        .grid-container {
            display: grid;
            grid-template-columns: repeat(3, 1fr);
            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 asperiores eos nesciunt 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 molestias dolores 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 côté-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;
  }

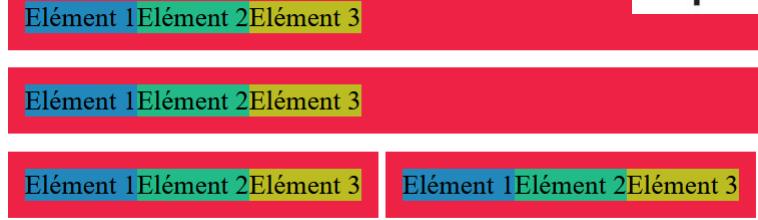
  .bleu{background-color: #28B; }
  .vert{background-color: #2B8; }
  .jaune{background-color: #BB2; }
  .rouge{background-color: #E24; }

</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 définit 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).



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.

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.

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 1^{er} et dernier éléments ont une marge avec le conteneur

space-between

- même espace entre chaque élément mais 1^{er} 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

align-content

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.

Apprendre en jouant

<https://flexboxfroggy.com/#fr>

Flexbox Froggy



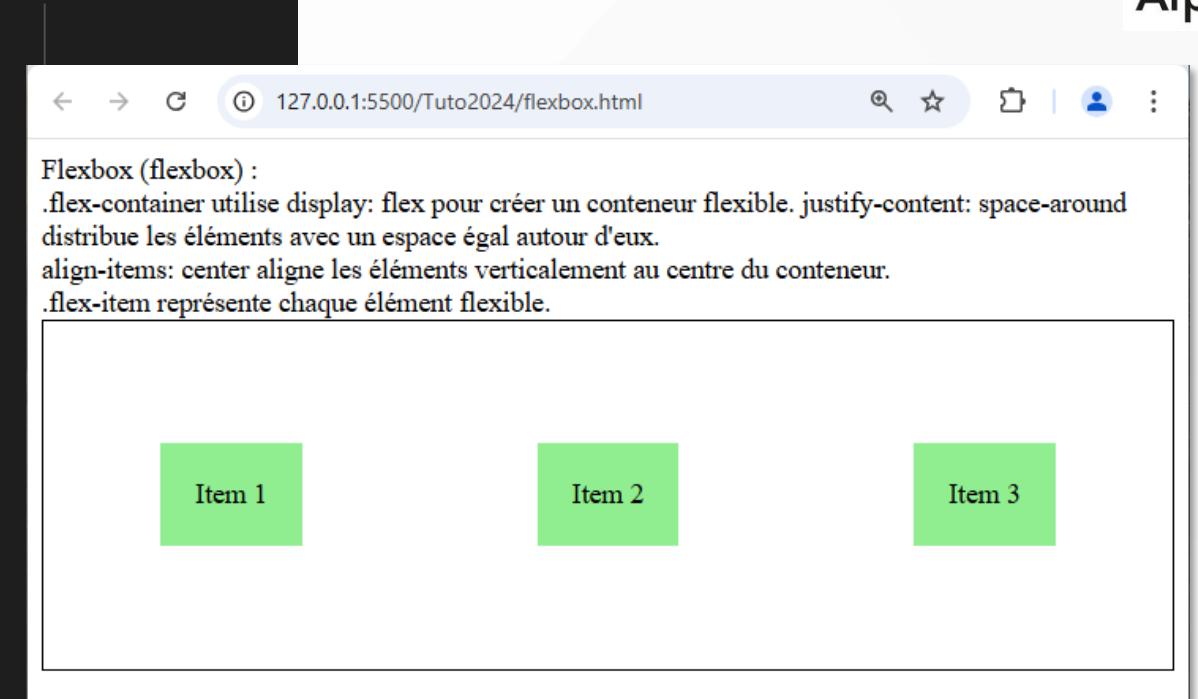


Un exemple de FLEXBOX

```
<!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

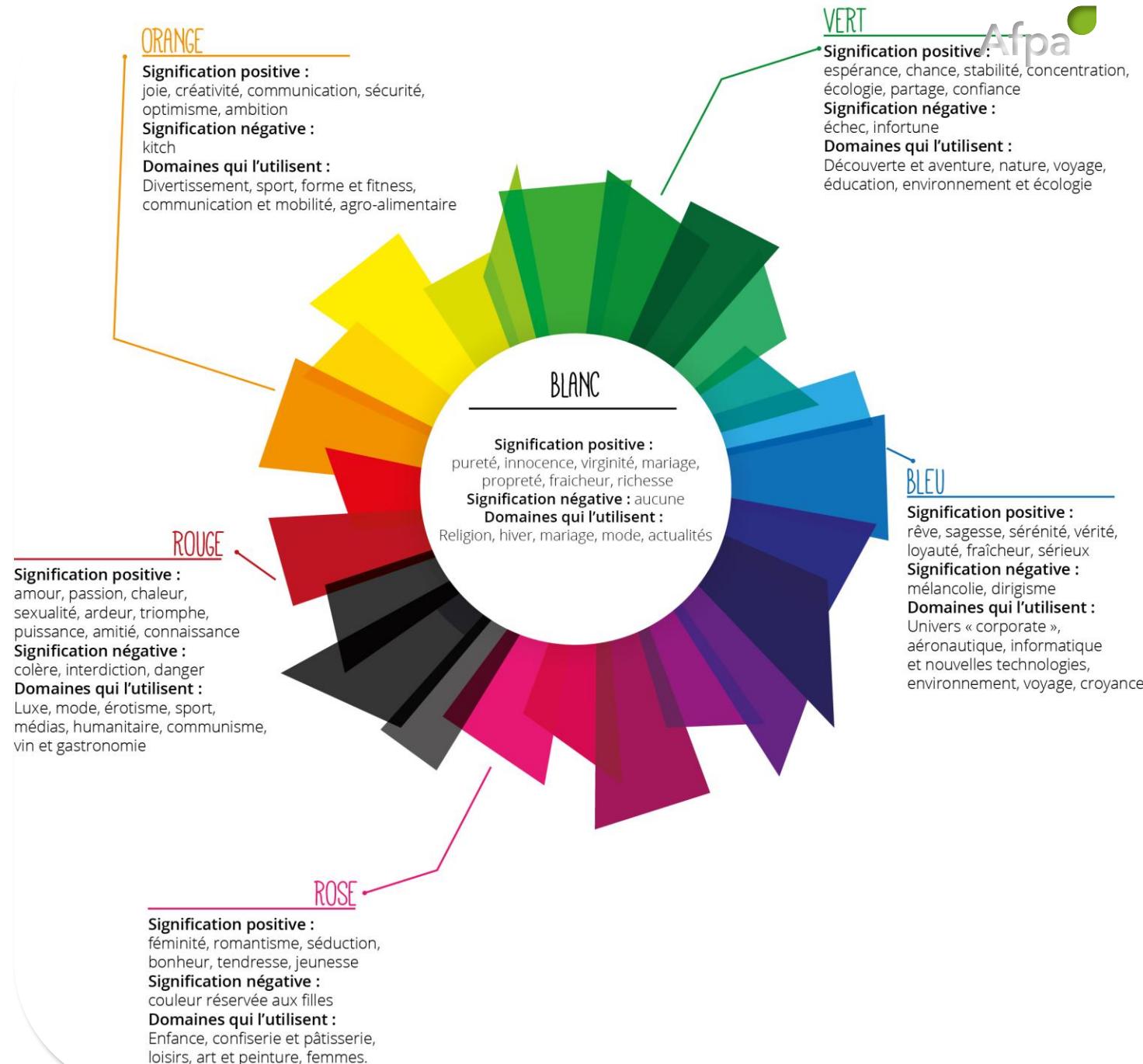
- **Border-top-left-radius, border-bottom-right** : Arrondis elliptiques des angles de la bordure (2 paramètres séparés par un espace)

Les couleurs dans le web

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

Quelques règles à connaître :

- 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

Nom :

Prénom :

Email :

Téléphone :

Envoyer

```
<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

```
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;  
}
```

```
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

Ingédients : ...

Instructions : ...

Recette 2 : Soupe à l'Oignon

Ingédients : ...

Instructions : ...

Recette 3 : Poulet Rôti

Ingédients : ...

Instructions : ...

© 2024 Mes Recettes de Cuisine

```
<!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>Ingédients : ...</p>
            <p>Instructions : ...</p>
        </section>
        <section id="recette2">
            <h2>Recette 2 : Soupe à l'Oignon</h2>
            <p>Ingédients : ...</p>
            <p>Instructions : ...</p>
        </section>
        <section id="recette3">
            <h2>Recette 3 : Poulet Rôti</h2>
            <p>Ingédients : ...</p>
            <p>Instructions : ...</p>
        </section>
    </main>
    <footer>
        <p>© 2024 Mes Recettes de Cuisine</p>
    </footer>
</body>
</html>
```

AVEC UN PEU DE 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;
}

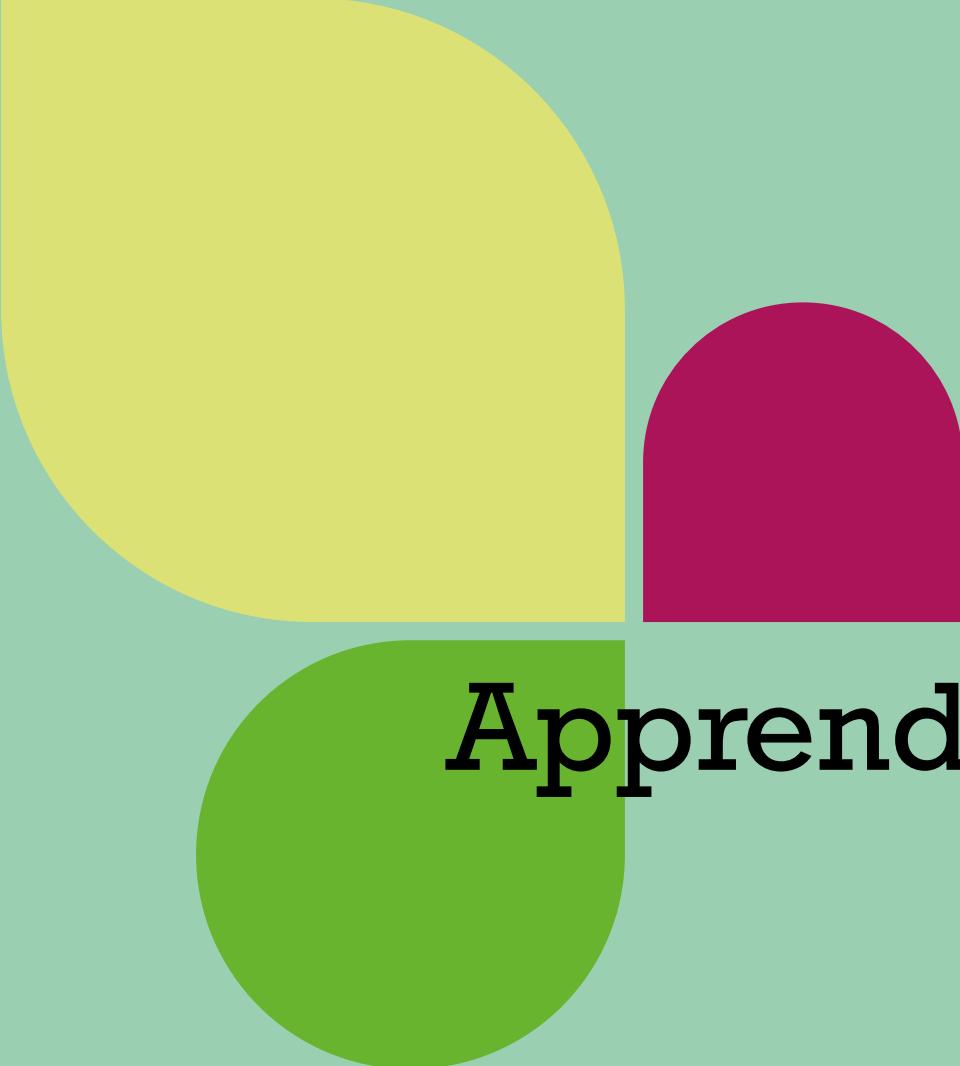
nav ul li a {
    color: white;
    text-decoration: none;
}

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;
}
```

The screenshot shows a website titled "Mes Recettes de Cuisine" with a green header. Below the header is a navigation bar with three items: "Recette 1", "Recette 2", and "Recette 3". The main content area contains two recipe cards. The first card is for "Recette 1 : Tarte aux Pommes" and includes sections for "Ingrédients : ..." and "Instructions : ...". The second card is for "Recette 2 : Soupe à l'Oignon" and also includes "Ingrédients : ..." and "Instructions : ...". At the bottom of the page is a dark footer with the text "© 2024 Mes Recettes de Cuisine".



Apprendre en jouant

JEUX SUR LE CSS

<https://flukeout.github.io/>

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

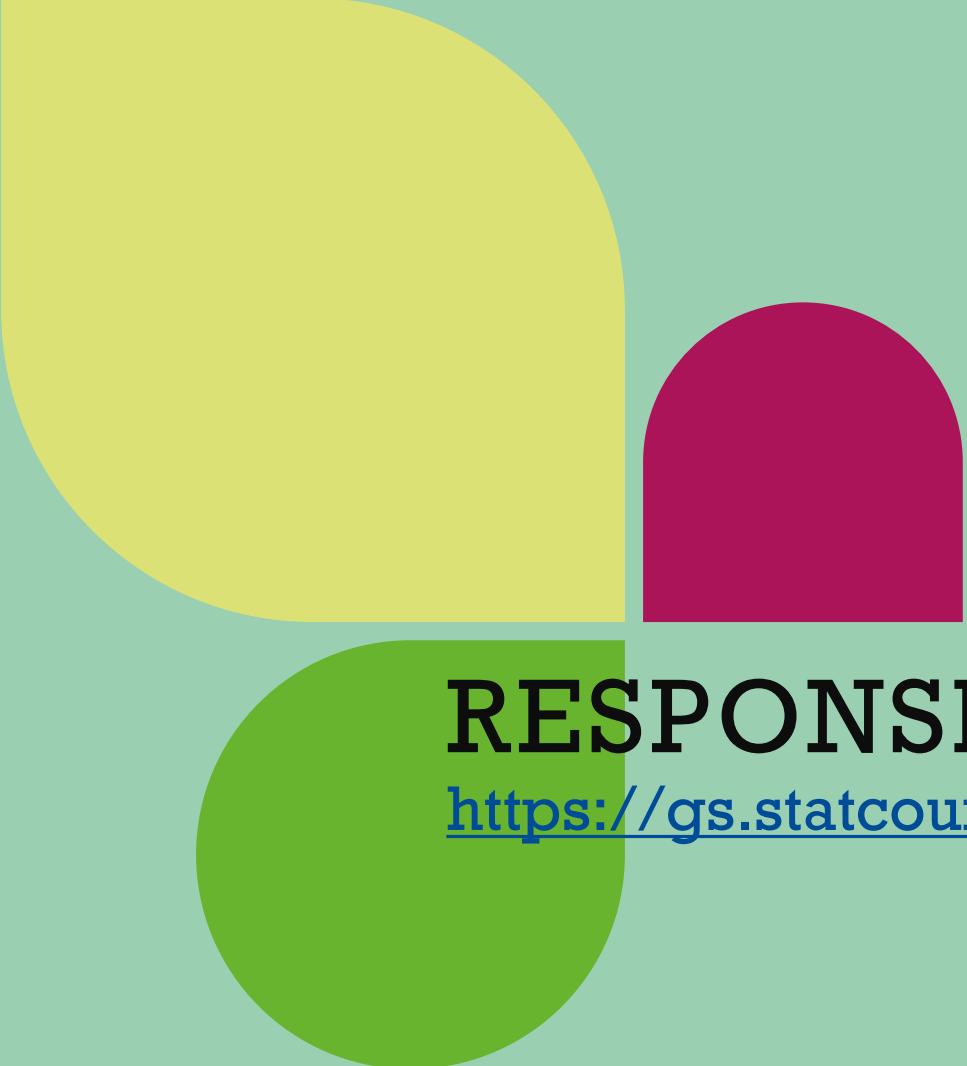
The screenshot shows a game interface titled "CSS Diner" with a level indicator "Level 1 of 32". The main title is "Select the plates". Below it is a button "Help, I'm stuck!". In the center is a graphic of a yellow table with two white plates. At the bottom is a "CSS Editor" window with the file "style.css" containing:

```
1 Type in a CSS selector 
2 {
3 /* Styles would go here. */
4 }
5
6 /*
7 Type a number to skip to a
8 level.
9 Ex → "5" for level 5
10 */
11
12
```

Next to it is an "HTML Viewer" window with the file "table.html" containing:

```
1 <div class="table">
2   <plate />
3   <plate />
4 </div>
5
6
7
8
9
10
11
12
```

To the right, under "Type Selector", it says "Select elements by their type" and shows a box labeled "A". Below it is explanatory text: "Selects all elements of type A. Type refers to the type of tag, so <div>, <p> and are all different element types." There are also "Examples" sections for "div" and "p".



RESPONSIVE DESIGN

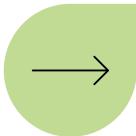
<https://gs.statcounter.com/>

Mise en place

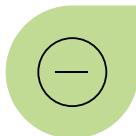
Structurer d'abord, présenter ensuite

Une des premières étapes est de construire la structure HTML de sa future page web.

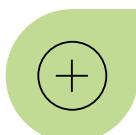
- Définir progressivement les sélecteurs et styles CSS
- Définir les dimensions et placements de manière relative (en em, en rem, en %)
- En ajoutant progressivement les jeux et styles assurant la mise en page et le rendu graphique par les "media queries".
- A chaque étape , des phases de tests et validations sont nécessaires.



Media Queries : outil fondamental de CSS qui nous permet de mettre en œuvre cette adaptabilité. Elles agissent comme des "conditions" : *Si l'écran a telle largeur, alors applique ces styles.*



Desktop First : concevoir le site pour les écrans larges (ordinateurs de bureau) et utilisez des Media Queries avec `max-width` pour "réduire" les styles pour les écrans plus petits.



Mobile-First : concevoir le site pour les plus petits écrans (mobiles) en écrivant les styles par défaut. Ensuite, vous utilisez des Media Queries avec `min-width` pour "ajouter" ou "modifier" des styles à mesure que l'écran devient plus grand.

QU'EST-CE QU'UNE MEDIA QUERY ?

Une Media Query est une règle CSS qui permet d'appliquer un ensemble de styles uniquement si certaines conditions liées aux caractéristiques du *média* (le type d'appareil ou ses propriétés) sont remplies.

Une Media Query se présente comme une instruction conditionnelle pour votre CSS :
`@media type_de_média and (caractéristique_de_média) { /* styles CSS à appliquer */ }`

```
/*
Jeu de styles pour écrans en orientation paysage affichant
entre 200 et 639 pixels en largeur
*/
@media screen and (min-width :200px) and (max-width :639px) and
(orientation:landscape) {
    /* règles */
}

/*
Jeu de styles pour téléviseurs et périphériques
affichant en format 16/9 ou 16/10
*/
@media tv, (device-aspect-ratio :16/9), (device-aspect-ratio :16/10) {
    /* règles */
}

/*
jeu de styles CSS pour écrans extra-larges
(affichant au moins 1200 pixels)
*/
@media screen and (min-width :1200px) {
    /* règles */
}
```

Type de média

Spécifie le type de support pour lequel les styles sont destinés.

all

Pour tous les types d'appareils. C'est la valeur par défaut si aucun type n'est spécifié

screen

Pour les écrans d'ordinateur, les tablettes, les smartphones, etc. (la plupart des usages interactifs).

print

Pour les documents imprimés. Utile pour adapter les styles lors de l'impression d'une page (ex: masquer les barres de navigation, optimiser les marges).

speech

Pour les synthétiseurs vocaux.

Exemple

`@media screen { styles pour les écrans }`

Les Caractéristiques de Média

Expression qui vérifie une propriété spécifique du média. C'est ici que réside la puissance des Media Queries. Les plus utilisées sont celles liées aux dimensions de l'écran

width, height

Largeur et hauteur de la zone d'affichage (viewport).

**min-width,
max-width**

Conditions pour une largeur *minimum* ou *maximum*. C'est **extrêmement important** pour le Responsive Design.

**min-height,
max-height**

Conditions pour une hauteur *minimum* ou *maximum*

orientation

Si l'appareil est en mode portrait (hauteur > largeur) ou landscape (largeur > hauteur).

resolution

Résolution de l'écran (ex: `min-resolution: 300dpi`).

Les Caractéristiques de Média

Ces caractéristiques peuvent être combinées avec des opérateurs logiques

prefers-color-scheme

Permet d'adapter les styles en fonction du thème sombre/clair préféré par l'utilisateur (light ou dark).

and

Pour combiner plusieurs conditions.

`@media screen and (min-width: 768px) and (orientation: landscape)`

not

Pour inverser une condition.

`@media not screen`

, ou or

Pour spécifier plusieurs Media Queries, si l'une ou l'autre est vraie. `@media screen and (min-width: 768px), print`

Exemple

Structure générale et exemple concret

Appliquer une couleur de fond différente si la largeur de l'écran est inférieure à 600 pixels.

```
/* La structure générale d'une Media Query est la suivante : */
@media type_de_média and (caractéristique_de_média: valeur) {
    /* Vos règles CSS spécifiques pour cette condition */
    sélecteur {
        propriété: valeur;
    }
}

/* Styles généraux (appliqués par défaut) */
body {
    background-color: lightblue;
    font-family: sans-serif;
}

/* Media Query : Ces styles s'appliqueront
si l'écran a une largeur maximale de 600px */
@media screen and (max-width: 600px) {
    body {
        background-color: lightcoral; /* La couleur change pour les petits écrans */
    }
    h1 {
        font-size: 1.5em; /* La taille du titre diminue */
    }
}
```

Point de rupture

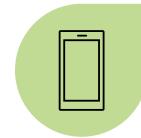
Les points de rupture ou breakpoints sont les largeurs d'écran spécifiques où votre design doit changer pour s'adapter. Il n'y a pas de valeurs universelles, mais des standards se sont établis

- Ces valeurs sont des points de départ. L'important est de définir vos propres breakpoints là où le *contenu* de votre site commence à mal s'afficher.



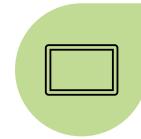
Petit mobile

jusqu'à 320px



Mobile

321px à 480px



Tablette

481px à 768px (ou 992px)



Petit ordinateur grande tablette

769px (ou 993px) à 1200px



Grand ordinateur

1201px et plus

- *width=device-width* : Dit au navigateur de faire correspondre la largeur du viewport à la largeur de l'appareil en pixels CSS.
- *initial-scale=1.0* : Définit le niveau de zoom initial quand la page est chargée. 1.0 signifie qu'il n'y a pas de zoom.

Le meta Tag viewport

L'indispensable

Pour que les Media Queries fonctionnent correctement sur les appareils mobiles, vous devez impérativement inclure le meta tag viewport dans la section <head> de votre document HTML

```
<head>
    <!-- ENCODAGE --&gt;
    &lt;meta charset="UTF-8"&gt;
    <!-- description du site --&gt;
    &lt;meta name="description" content="Cours HTML"&gt;
    <!-- prise en charge du contexte mobile --&gt;
    &lt;meta name="viewport" content="width=device-width, initial-scale=1.0"&gt;
    <!-- titre de la page --&gt;
    &lt;title&gt;Document&lt;/title&gt;
    &lt;link rel="stylesheet" type="text/css"
        href="ressources/style/style.css"&gt;

    <!-- Meilleur méthode --&gt;
    &lt;script type="text/javascript"
        src="ressources/script/script.js"&gt;&lt;/script&gt;
&lt;body&gt;</pre>
```

Unités de mesures CSS

Dans le responsive

Les nouvelles unités de mesure CSS (`vh`, `vw`) pourront être utilisées pour définir des dimensions proportionnellement aux caractéristiques du viewport :

- `1 vh = 1/100 de la hauteur du viewport`
- `1 vw = 1/100 de la largeur du viewport`

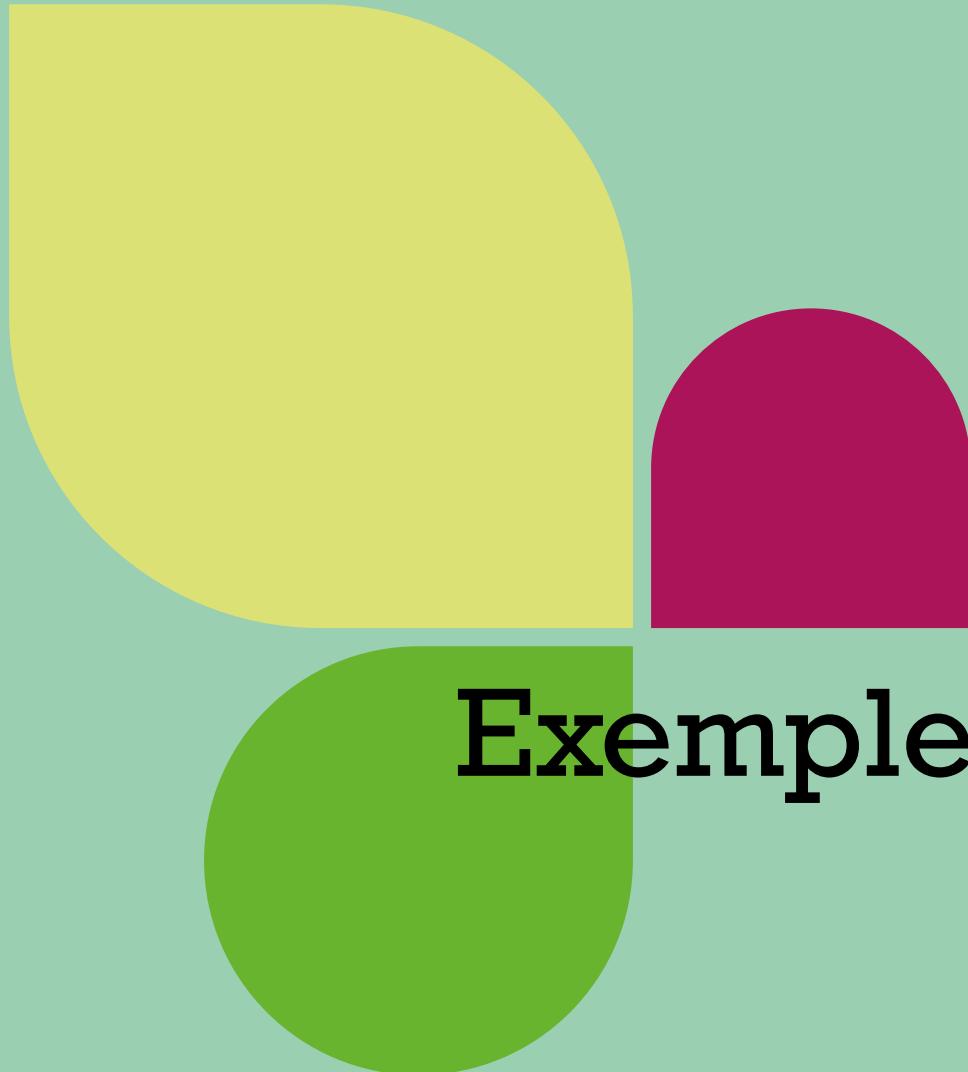
Complément sur la notion em

Le terme `em` est issu de la typographie qui utilise toujours des polices à **espacements proportionnels** pour lesquelles les caractères n'occupent pas tous la même place

- Les navigateurs par défaut ont adopté la taille de 16 pixels.
- H1 a une taille de `2em`, H2 de `1,5em` etc...

Le Responsive Design est donc basé sur ce principe : pour chaque media query, définir la taille de police par défaut puis redéfinir les tailles en `em`, par rapport à leur propre contexte (ou `rem`).

`1em = 1 fois la taille de la police de l'élément parent`



UN SITE WEB



```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Page Web Responsive</title>
    <link rel="stylesheet" href="./Style/media.css">
</head>
<body>
    <header>
        <h1>Mon Site Web</h1>
        <nav>
            <ul>
                <li><a href="#">Accueil</a></li>
                <li><a href="#">À propos</a></li>
                <li><a href="#">Services</a></li>
                <li><a href="#">Contact</a></li>
            </ul>
        </nav>
    </header>
    <main>
        <section>
            <h2>Bienvenue sur mon site web</h2>
            <p>Ceci est un exemple de page web responsive utilisant des media queries.</p>
        </section>
        <section>
            
        </section>
    </main>
    <footer>
        <p>© 2023 Mon Site Web</p>
    </footer>
</body>
</html>
```

LE CSS

Dans le fichier CSS, la partie intéressante se trouve en bas dans l'encadré jaune.

- `@media(max-width : 780px)` indique pour les médias jusqu'à 780px, les éléments HTML auront le comportement défini dans le media queries
 - Modification du comportement de la navigation avec un affichage par block et plus inline
 - Modification des dimensions de l'image à 70% de sa taille d'origine.
 - Modification du padding des éléments de la page.

```

/* Styles de base */
body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    line-height: 1.6;
}

header {
    background-color: #333;
    color: #fff;
    padding: 1rem 0;
    text-align: center;
}

header h1 {
    margin: 0;
}

nav ul {
    list-style: none;
    padding: 0;
}

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

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

main {
    padding: 2rem;
}

img {
    width: auto;
    height: 500px;
}

section {
    text-align: center;
}

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

/* Media Query pour les écrans mobiles */
@media (max-width: 768px) {
    header {
        padding: 1rem;
    }

    nav ul li {
        display: block;
        margin: 0.5rem 0;
    }

    main {
        padding: 1rem;
    }

    section {
        text-align: center;
    }

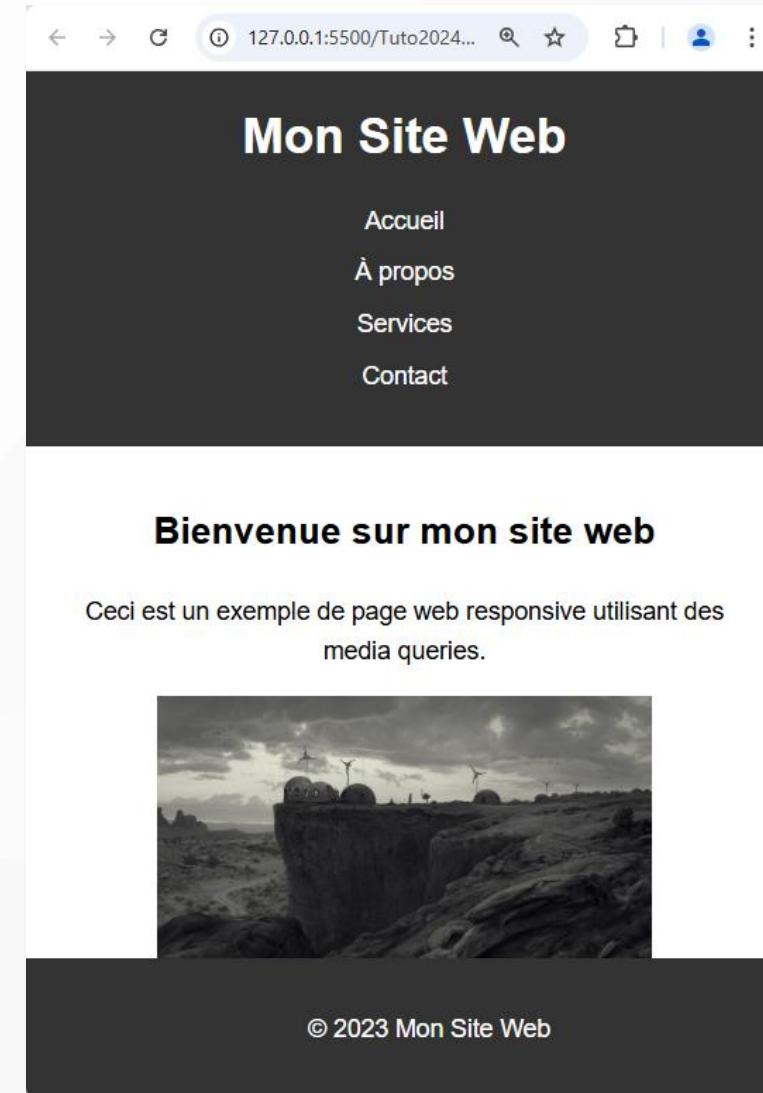
    img {
        width: 70%;
        height: 70%;
    }

    footer {
        padding: 1rem;
    }
}

```

EN MODE MOBILE

Avec l'application du media queries, la page modifie son comportement dès qu'on atteint la taille paramétrée, permettant ainsi de rendre le site responsive et adapté à toutes type d'écran.



Concepteur Développeur d'Applications

Version 5 - révision 2025

afpa.fr



Jérôme BOEBION