



# HTML et CSS

Le web et ses technologies

---

AFPA CENTRE DE POMPEY





# 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...

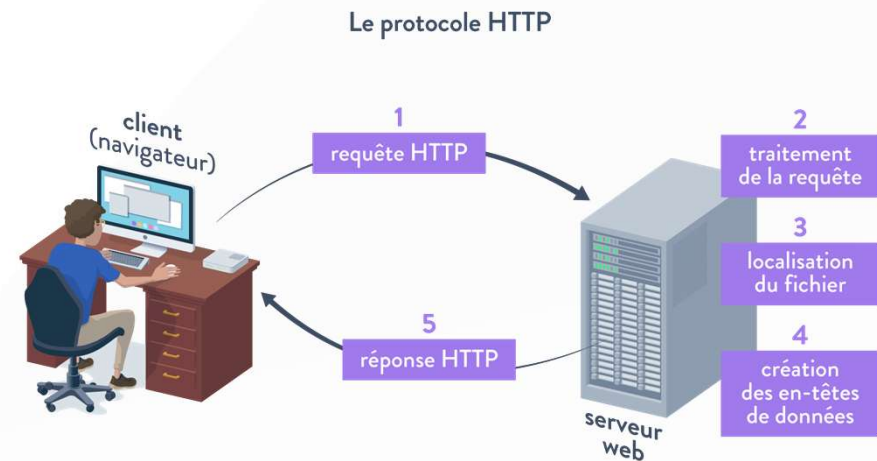


## DNS

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



© SCHOOLMOUV

# 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 script Javascript pour les rendre dynamique



## HTML : Contenu et structure

En-tête, Titre, paragraphe...

---



## CSS : Présentation

Font, Couleur, Bordure...

---



## JAVASCRIPT : Comportement

Affichage dynamique,  
interaction, popup...

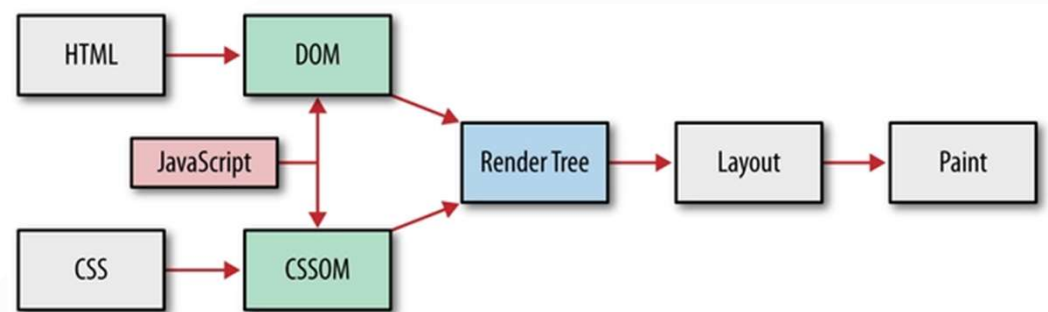
# LES MOTEURS DE RENDU

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'évènements :



[Comprendre-le-navigateur](https://thibault.mahe.io/blog/comprendre-le-navigateur)

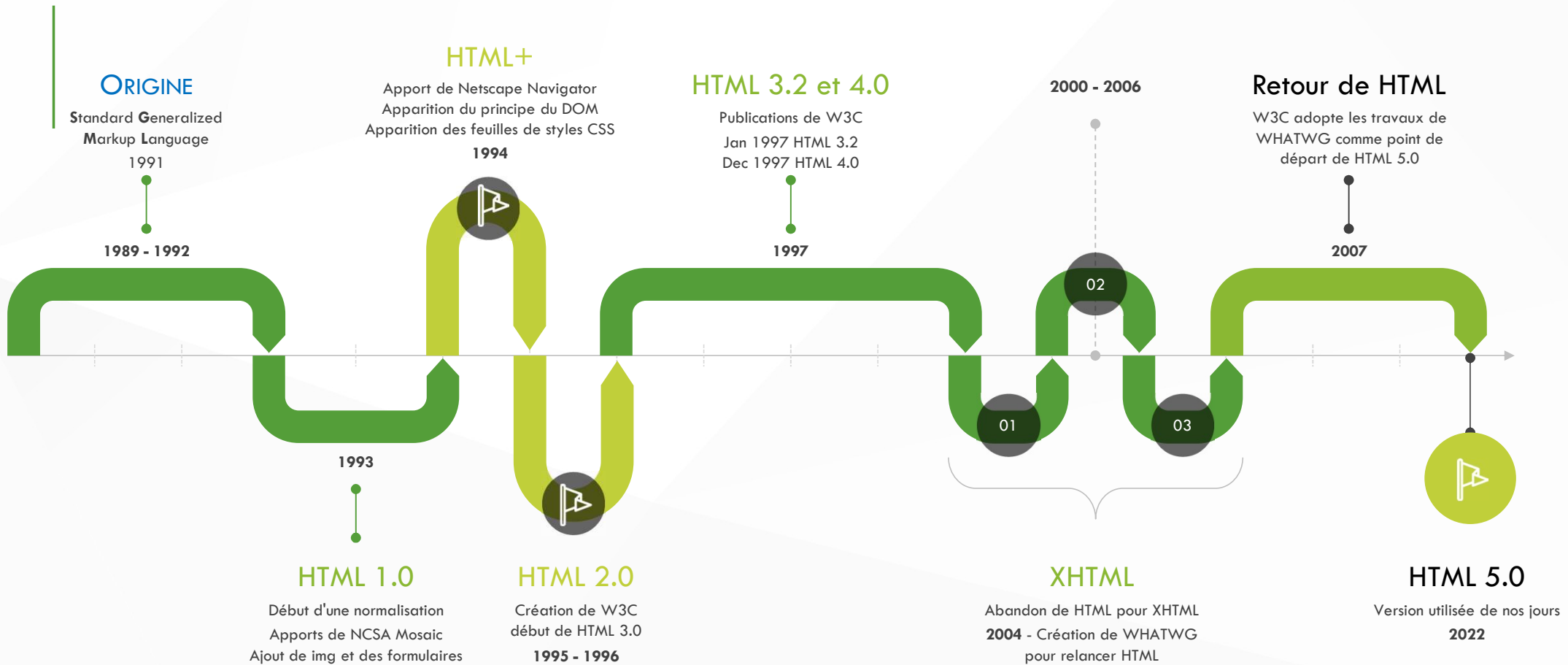
<https://thibault.mahe.io/blog/comprendre-le-navigateur>





# 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.



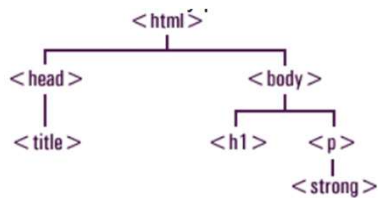
# Composition d'une page HTML

# 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)**



CONCEPTEUR DEVELOPPEUR D'APPLICATIONS

## Structure

## Balises

## Attributs



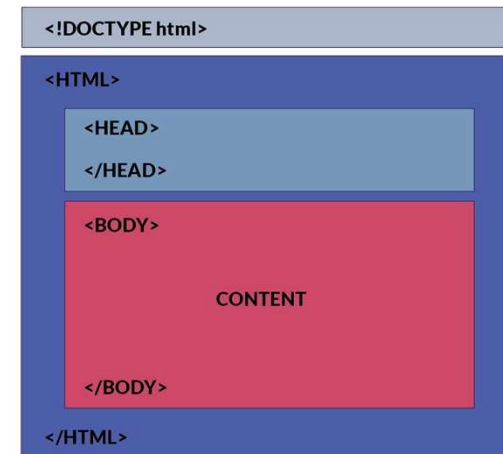
HTML est un langage de balises.

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

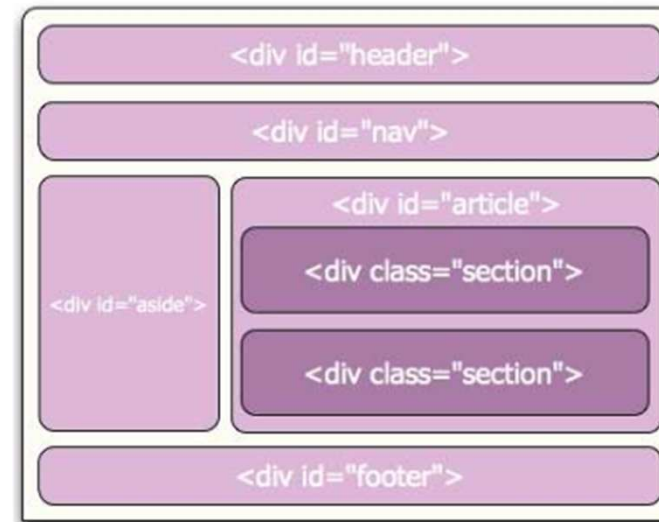


# LE RENDU D'UNE PAGE

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

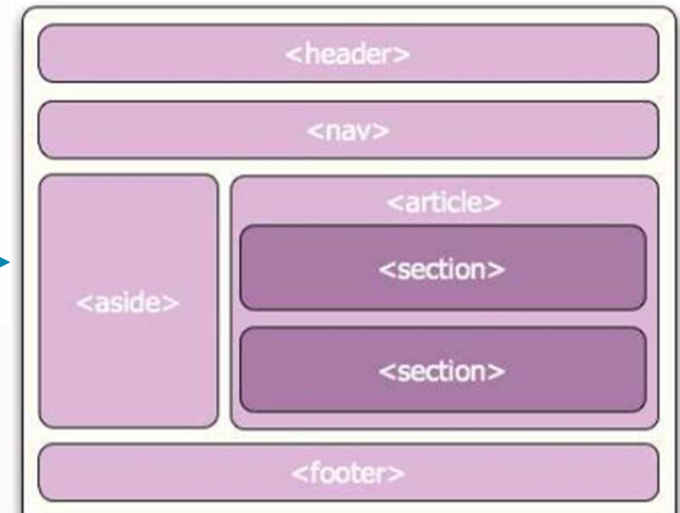
Evolution de la mise en page

- Dans 1<sup>er</sup> 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> </a>`



- `<h1><a href...>... </a></h1>`

## Éléments vides

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



Résumé technique	
<u>Catégories de contenu</u>	<u>Contenu de flux</u>
Contenu autorisé	Aucun, c'est un élément vide.

# HTML V5

Chaque page HTML doit commencer par l'instruction **DOCTYPE** <sup>1</sup> 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">
    <!-- rafraichissement 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

# HEAD

Le [HEAD](#) contient les informations sur le document courant (titre, jeu de caractères, mots-clés, icône de la page, rafraichissement 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" />
```

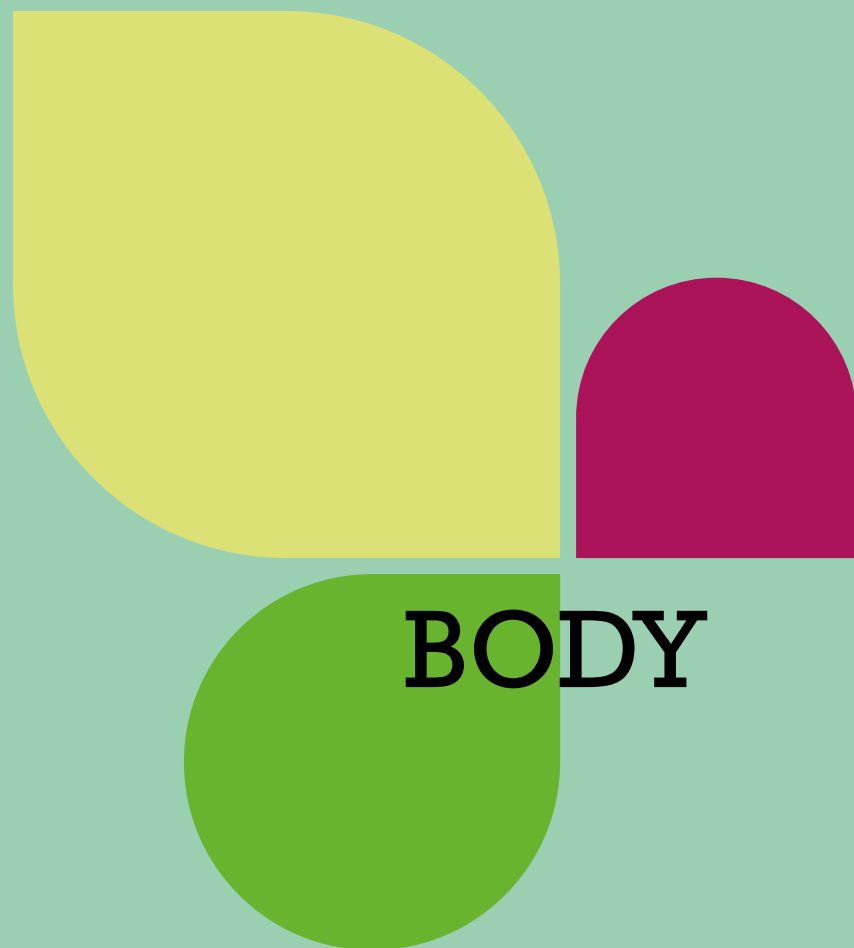
```
<meta name="description" content="Tapez ici en quelques phrases la description de votre site" />
```

```
<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 `<span>`
  - Des mises en exergue `<em>`
  - 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 `<span>` permet d'identifier une portion de phrase.

- Pas d'effet visuel.

Attribut	Utilité
<b>id</b>	Identifiant de la balise.
<b>class</b>	liste de classes séparées par des espaces. Utilisé pour appliquer un style
<b>style</b>	Information locale de style
<b>title</b>	Permet une information supplémentaire concernant l'élément.
<b>align</b>	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 `<br>` 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 `<font>` 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
À	&agrave;	Î	&icirc;	Ç	&ccedil;	Ù	&ugrave;
Á	&aacute;	Ï	&iuml;	È	&egrave;	Ú	&uacute;
Â	&acirc;	Ö	&eth;	É	&eacute;	Û	&ucirc;
Ã	&atilde;	Ñ	&ntilde;	Ê	&ecirc;		
Ä	&auml;	Ô	&ocirc;	Ë	&euml;		
Æ	&aelig;	Ö	&ouml;	Ü	&uuml;		

Caractère	Code ISO	Entité HTML
©	&#169;	&copy;
,	&#130;	
§	&#167;	&sect;
„	&#132;	
...	&#133;	
£	&#163;	&pound;
¶	&#182;	&para;
·	&#183;	&middot;
¼	&#188;	&frac14;
½	&#189;	&frac12;
¾	&#190;	&frac34;
€	&#8364;	&euro;
‘	&#145;	
’	&#146;	
“	&#147;	
”	&#148;	
•	&#149;	
—	&#150;	
—	&#151;	
Espace insécable	&#160;	&nbsp;
<	&#60;	&lt;
>	&#62;	&gt;

# 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

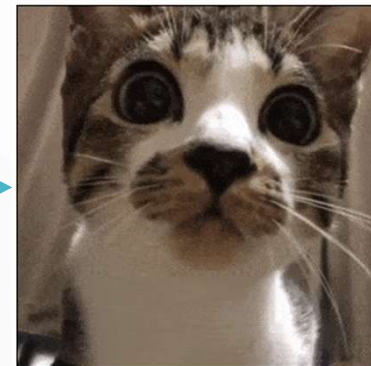
Balise `<img src="" alt="">`

- 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></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
  - `<a href="page2.htm" >Cliquez vers la page 2</a>`
  - `<a href ="page2.htm#cible" >Cliquez ici</a>`
- Lien absolu (lien externe) :
  - Il faut connaître l'adresse complète de la page cible
  - `<a href =" http://www.afpa.fr" >site de l'AFPA</a>`
- Lien email :
  - Il faut connaître l'adresse complète de la page cible
  - `<a href="mailto:mfay@dom.com">Ecrivez-nous</a>`
  - `<a href=mailto:mfay@dom.com?cc=rleco@dom.com&subject=commande>Ecrivez-nous</a>`

# 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>
    <td>Total</td>
    <td>1337</td>
  </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>
```

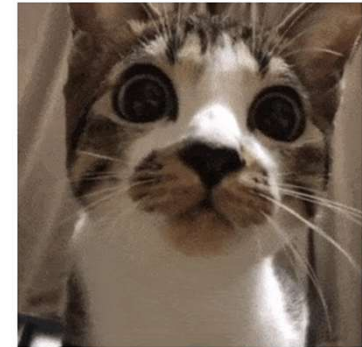
PAGE 1

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

▼ Ici se trouve du texte

Lorem ipsum dolor sit amet consecte

# 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 débiter 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/>

[illegible]





# 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'envoi 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 contrôle 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>
```

Fichier à joindre

Sélectionner le fichier à envoyer :  Choisir un fichier MySQL Exos.ptx

## 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

```
<label for="range">Choix couleur</label>
<input type="color" name="color"
  id="color" value="carmin" />
```

Choix couleur



```
<label for="range">Saisie d'un nombre en 0 et 10</label>
<input type="range" name="range"
  id="range" min="0" max="10" step="1" />
```

Saisie d'un nombre en 0 et 10



## 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 :

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-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>
```