

# Fiche Révision HTML

## Table des matières

I.	Structure de base d'une page HTML :	2
II.	Les balises	3
1.	Balises de type block	3
2.	Balises de types inline :	4
3.	Balises génériques :	5
4.	Balises orphelines	5
5.	Autres balises	5
III.	Les listes	6
1.	Les listes non ordonnées	6
2.	Les listes ordonnées	6
3.	Les dictionnaires	6
IV.	Les attributs	6
V.	Les liens	7
VI.	Les images	8
VII.	Les figures	8
VIII.	Les tableaux	9
IX.	Les formulaires	12
1.	Structure générale d'un formulaire. Exemple zone de saisie monoligne	12
2.	Les différentes zones de saisie	13
3.	Les cases à cocher : type= "checkbox"	14
4.	Les zones d'options : type="radio"	15
5.	Les listes déroulantes	16
6.	Exemple de formulaire de contact basique	17
7.	Informations supplémentaires	18
8.	Envoyer le formulaire	18
9.	Vérifier le formulaire	18
X.	Intégrer de la vidéo et de l'audio	20
1.	Insérer un élément audio	20
2.	Insérer un élément vidéo	21
3.	Balise iframe	21
XI.	Outils HTML	22
1.	Minifier	22
2.	Compresser les images	22
3.	Evaluer la performance de son site internet	22
4.	SEO	23

Yes, it's going to be challenging.  
But it's also going to be fun.  
And it might even be life changing, too.

## I. Structure de base d'une page HTML :

```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <!-- En tête de la page (Ceci est un commentaire) -->
    <meta charset="utf-8" />
    <title>Ma Page Web</title>
    <link rel="stylesheet" href="style.css"/>
    <script src="script.js"></script>
  </head>
  <body>
  </body>
</html>
```

### Explication :

<!-- Commentaire -->	Balise pour les commentaires.
<head></head>	Rassemble les métadonnées de la page
<title></title>	Apparaît dans l'onglet du navigateur, dans les résultats du moteur de recherche
<link>	Sert à connecter le fichier CSS à la page.
<script>	Sert à connecter le fichier javascript à la page.
<body></body>	Contient la majeure partie du code

### Autres balises *meta*

```
<meta name="description" content="Une description de la page..." />
<meta name="author" content="Nicolas AUNE" />
<meta name="copyright" content="© CC BY-NC-SA 4.0" />
```

## II. Les balises

Il existe deux types de balise : **type block** et **type inline**.

Les balises de **type block** commencent sur une nouvelle ligne et occupent toute la largeur de la ligne. Elles créent automatiquement un retour à la ligne après. Ils peuvent être imbriqués les uns dans les autres et peuvent contenir des éléments de type inline.

[https://developer.mozilla.org/en-US/docs/Web/HTML/Block-level\\_elements](https://developer.mozilla.org/en-US/docs/Web/HTML/Block-level_elements)

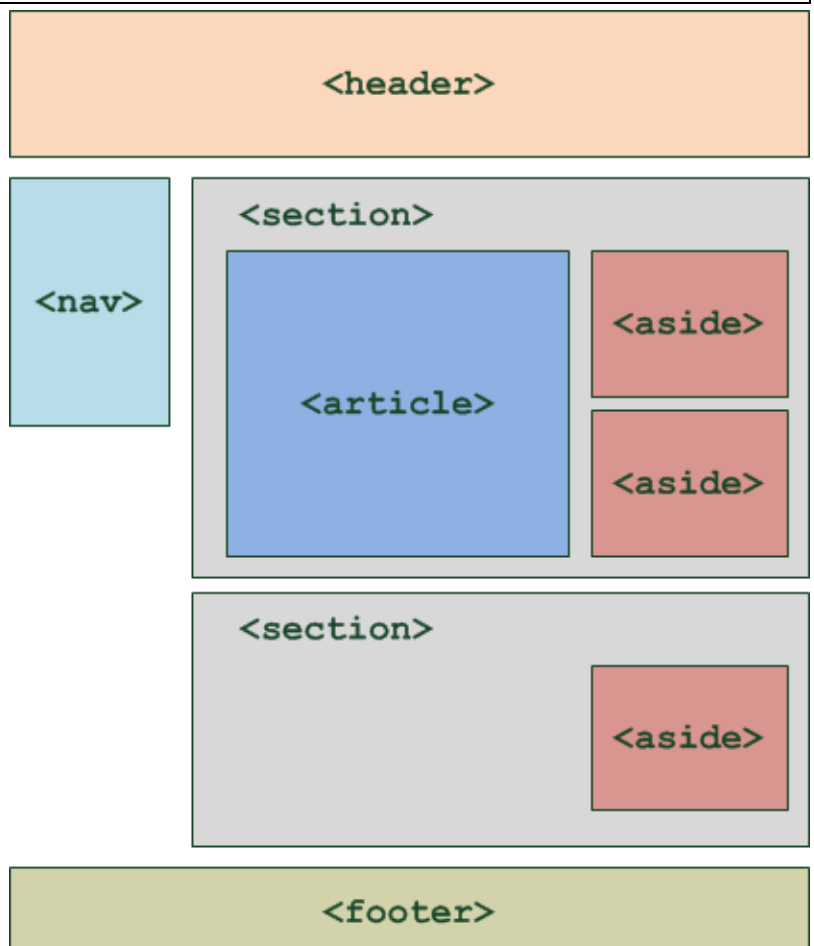
Les éléments de **type inline** s'insèrent dans une ligne. Ils se trouvent obligatoirement dans une balise block. Ils occupent seulement la largeur supplémentaire. Ils peuvent s'imbriquer les uns dans les autres mais ne peuvent pas contenir des éléments de type block.

[https://developer.mozilla.org/en-US/docs/Web/HTML/Inline\\_elements](https://developer.mozilla.org/en-US/docs/Web/HTML/Inline_elements)

### 1. Balises de type block

<code>&lt;p&gt;&lt;/p&gt;</code>	Les paragraphes
<code>&lt;h1&gt;&lt;/h1&gt;</code>	Les titres. Valable aussi avec <code>&lt;h2&gt; Titre ici &lt;/h2&gt; ... &lt;h6&gt; Titre ici &lt;/h6&gt;</code>
<code>&lt;header&gt; &lt;/header&gt;</code>	Pour l'en-tête du site.
<code>&lt;nav&gt; &lt;/nav&gt;</code>	Pour la navigation dans le site
<code>&lt;main&gt; &lt;/main&gt;</code>	
<code>&lt;section&gt; &lt;/ section &gt;</code>	Pour regrouper les contenus en fonction de leur thématique. Chaque section peut avoir son titre <code>&lt;h1&gt;</code> .
<code>&lt;article&gt; &lt;/ article &gt;</code>	Pour un article indépendant.
<code>&lt;aside&gt; &lt;/ aside &gt;</code>	Pour contenir des informations complémentaires. En général sur le côté.
<code>&lt;footer&gt; &lt;/ footer &gt;</code>	Pour le pied de page

Permet de mieux structurer le code, et utile pour les moteurs de recherche. Ci-contre : Un exemple d'organisation.



```

<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>Zozor - Le Site Web</title>
  </head>

  <body>
    <header>
      <h1>Zozor</h1>
      <h2>Carnets de voyage</h2>
    </header>
    <nav>
      <ul>
        <li><a href="#">Accueil</a></li>
        <li><a href="#">Blog</a></li>
        <li><a href="#">CV</a></li>
      </ul>
    </nav>
    <section>
      <aside>
        <h1>À propos de l'auteur</h1>
        <p>C'est moi, Zozor ! Je suis né un 23 novembre 2005.</p>
      </aside>
      <article>
        <h1>Je suis un grand voyageur</h1>
        <p>Bla bla bla bla (texte de l'article)</p>
      </article>
    </section>
    <footer>
      <p>Copyright Zozor - Tous droits réservés<br />
      <a href="#">Me contacter !</a></p>
    </footer>

  </body>
</html>

```

Il est tout à fait possible de réaliser cela avec des balises div (voir plus loin). Ces balises servent à indiquer ce qu'il y a dans les balises. Attention, ne pas utiliser les balises pour faire la mise en page. C'est au CSS de faire ça.

Il faut faire attention à l'indentation pour une meilleure clarté du code.

## 2. Balises de types inline :

<code>&lt;strong&gt; &lt;/strong&gt;.</code>	En gras par défaut.
<code>&lt;em&gt; &lt;/em&gt;.</code>	En italique par défaut.
<code>&lt;mark&gt; &lt;/mark&gt;.</code>	Surligne par défaut.
<code>&lt;a href="..."&gt;Lien&lt;/a&gt;</code>	Pour mettre un lien. On met l'adresse en valeur de l'attribut href.
<code>&lt;img /&gt;</code>	Pour les images
<code>&lt;abbr title='Cascading Style Sheets'&gt;CSS&lt;/abbr&gt;</code>	Affiche le contenu de l'attribut title au passage de la souris.

Ne pas utiliser cela pour faire de la mise en forme. Voir le CSS.

### 3. Balises génériques :

**Type block :** <div> </div>

**Type inline :** <span></span>

Peuvent remplacer les balises de types block ou inline précédente mais cela est moins clair. Ne pas en abuser.

### 4. Balises orphelines

<b>&lt;br&gt;</b>	Pour faire un saut de ligne	
<b>&lt;hr&gt;</b>	Rupture thématique entre deux paragraphes. Possibilité avec le css (voir après) de tracer une ligne.	
	<pre>hr {   border: none;   border-top: 3px double #333;   color: #333;   overflow: visible;   text-align: center;   height: 5px; }</pre>	<pre>hr:after {   background: #fff;   content: '§';   padding: 0 4px;   position: relative;   top: -13px; }</pre>

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

### 5. Autres balises

Balise <map></map> permet de créer des cartes sur une image avec des zones clickable.

<address></address> tag qui affiche les informations de contact de l'auteur du document.

<code></code> Balise qui indique qu'on y place du code

### III. Les listes

#### 1. Les listes non ordonnées

```
<ul>
  <li>Item N°1</li>
  <li>Item N°2</li>
  <li>Item N°3</li>
</ul>
```

#### 2. Les listes ordonnées

```
<ol>
  <li>Item N°1</li>
  <li>Item N°2</li>
  <li>Item N°3</li>
</ol>
```

#### 3. Les dictionnaires

```
<dl>
  <dt>Mot 1</dt>
  <dd>def 1</dd>
  <dt>Mot 2</dt>
  <dd>def 2</dd>
</dl>
```

Il est tout à fait possible d'imbriquer les listes.

Pour faire disparaître les puces des listes, on peut utiliser le CSS et utiliser la propriété *list-style*. Pour changer le type de puce on utilise la propriété *list-style-type*.

```
ul li {
  list-style: none;
}
```

<https://developer.mozilla.org/en-US/docs/Web/HTML/Element/ul>

<https://developer.mozilla.org/en-US/docs/Web/HTML/Element/ol>

### IV. Les attributs

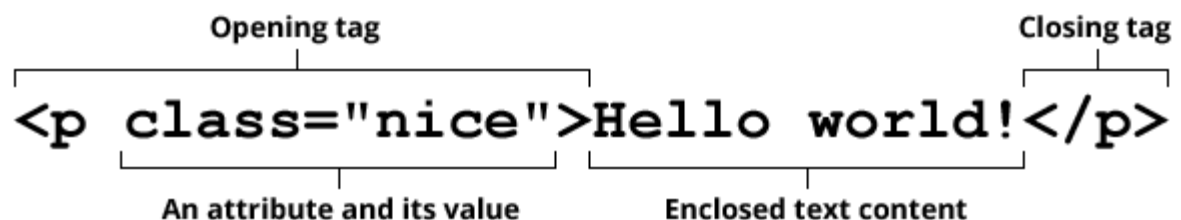
Les attributs se placent dans les balises ouvrantes.

Exemples :

```
<a href="https://www.theodinproject.com/about">click me</a>
<h1 class="title">Titre</h1>
<p id="ancree">Paragraphe</p>
```

Les attributs *class* et *id* vont servir pour donner du style dans le fichier CSS. La différence entre *class* et *id* est que l'on peut utiliser la même classe plusieurs fois dans le document HTML alors qu'un *id* ne peut être présent qu'une seule fois.

*Anatomy of an HTML element*



## V. Les liens

<ul style="list-style-type: none"><li>- Vers une page du même dossier :</li></ul> <pre>&lt;a href="page.html"&gt;click me&lt;/a&gt;</pre> <pre>&lt;a href="./page.html"&gt;click me&lt;/a&gt;</pre> <ul style="list-style-type: none"><li>- Vers une page d'un sous dossier :</li></ul> <pre>&lt;a href="content/page.html"&gt;click me&lt;/a&gt;</pre>	<ul style="list-style-type: none"><li>- Vers une page d'un dossier parent</li></ul> <pre>&lt;a href="../page.html"&gt;click me&lt;/a&gt;</pre> <ul style="list-style-type: none"><li>- Vers une url</li></ul> <pre>&lt;a href="http://..."&gt;click me&lt;/a&gt;</pre>
---	--

- Vers une url dans un nouvel onglet avec l'attribut "target"

```
<a href="https://youtube.com" title="Site internet YouTube !" target="_blank">YouTube</a>
```

- Vers une partie de la page (une ancre)

Une ancre dans la page	Une ancre vers une autre page
<pre>&lt;a href="#ancree1"&gt;Allez à ancre 1&lt;/a&gt;</pre>	<pre>&lt;a href="page.html#ancree1"&gt;Allez à ancre 1&lt;/a&gt;</pre>

Préalablement il faut place l'ancre par exemple :

```
<h1 id="ancree1">Titre</h1> ou encore <p id="ancree1">Paragraphe</p>
```

- Lien pour envoyer un mail :

```
<p><a href="mailto:votrenom@bidule.com">Envoyez-moi un e-mail !</a></p>
```

- Lien vers un fichier dans le même dossier :

```
<p>Télécharger mon<a href="fichier.pdf">fichier</a></p>
```

On peut rajouter l'attribut *download* pour lancer le téléchargement direct si l'utilisateur clique sur le lien.

La balise <a> peut aussi avoir un attribut title :

```
<p>Visitez <a href="https://openclassrooms.com" title="Vous ne le regretterez pas !">OpenClassrooms</a> ?</p>
```

Pour vérifier validité du code :

- Html : <http://validator.w3.org>
- CSS : <http://jigsaw.w3.org/css-validator>

## VI. Les images

Les images doivent être insérées dans des paragraphes, pas seules.

Pour rajouter une image :

```

```

L'attribut src est pour le chemin de l'image. Ici dans le sous dossier 'img'. L'attribut alt permet de décrire l'image.

Utile pour les mal-voyants, ce message sera affiché également en cas de problème avec l'image.

On peut également rajouter un titre en infobulle :

```

```

Pour faire une miniature et l'agrandir au clic à l'aide d'un lien:

```
<a href="img/florent.jpg" title="Ma Photo"></a>
```

## VII. Les figures

On utilise les figures pour illustrer le texte. Il peut être image, code source, citations etc...

```
<figure>
  
  <figcaption>Voici l'illustration</figcaption>
</figure>
```

On place l'image dans un paragraphe si elle n'apporte aucune information au texte, dans une figure si elle apporte une information. Surtout un rôle sémantique. Une figure peut comporter plusieurs images :

```
<figure>
  
  
  
  <figcaption>Logos des différents navigateurs</figcaption>
</figure>
```



## VIII. Les tableaux

Dans un tableau, on utilise les balises suivantes :

- `<table></table>` pour créer le tableau.
- `<tr></tr>` pour une ligne
- `<th></th>` pour une cellule qui est une entête
- `<td></td>` pour une cellule du tableau

Pour structurer un tableau on utilise :

- `<thead></thead>` pour l'entête du tableau
- `<tbody></tbody>` pour le corps du tableau
- `<tfoot></tfoot>` pour le pied du tableau

Il existe aussi la balise :

- `<caption></caption>` qui permet d'ajouter la légende du tableau

Pour appliquer du style plus facilement on peut utiliser la balise

- `<colgroup></colgroup>` qui va contenir les balises :
  - o `<col>` pour une colonne (un élément sans contenu)

On met autant de `<col>` qu'il y a de colonnes et avec l'attribut `style` on peut attribuer un style à chaque colonne.

Pour une fusion horizontale des cellules, on utilise l'attribut `colspan` et pour une fusion verticale, on utilise l'attribut `rowspan`.

Pour l'accessibilité (lecteur d'écran notamment), on utilise l'attribut `scope`. Il va permettre de réaliser automatiquement les associations entre les en-têtes et les cellules correspondantes (même ligne et même colonne). Il peut prendre plusieurs valeurs différentes :

- `colgroup` quand la cellule de tête s'étend sur plusieurs colonnes
- `rowgroup` quand la cellule de tête s'étend sur plusieurs lignes
- `col` pour une cellule de tête de colonne
- `row` pour une cellule de tête de ligne

Une alternative à l'attribut `scope` sont l'utilisation des attributs `id` et `headers` :

- on donne un identifiant unique `id` à chaque élément `<th>` (cellule de tête)
- on ajoute un attribut `headers` à chaque élément `<td>`. Chaque attribut `headers` doit contenir une liste des `id` de tous les éléments `<th>` qu'il contient séparés par des espaces.

```
<table>
  <caption>How I chose to spend my
money</caption>
  <thead>
    <tr>
      <th>Purchase</th>
      <th>Location</th>
      <th>Date</th>
      <th>Evaluation</th>
      <th>Cost (€)</th>
    </tr>
  </thead>
  <tfoot>
    <tr>
      <td colspan="4">SUM</td>
      <td>118</td>
    </tr>
  </tfoot>
  <tbody>
    <tr>
```

```
<thead>
  <tr>
    <th id="purchase">Achats</th>
    <th id="location">Où</th>
    <th id="date">Date</th>
    <th id="evaluation">Avis</th>
    <th id="cost">Coût (€)</th>
  </tr>
</thead>
<tbody>
  <tr>
    <th id="haircut">Coupe de
cheveux</th>
    <td headers="location
haircut">Coiffeur</td>
    <td headers="date
haircut">12/09</td>
    <td headers="evaluation
haircut">Bonne idée</td>
    <td headers="cost haircut">30</td>
```

```

        <td>Haircut</td>
        <td>Hairdresser</td>
        <td>12/09</td>
        <td>Great idea</td>
        <td>30</td>
    </tr>
    <tr>
        <td>Lasagna</td>
        <td>Restaurant</td>
        <td>12/09</td>
        <td>Regrets</td>
        <td>18</td>
    </tr>
    <tr>
        <td>Shoes</td>
        <td>Shoeshop</td>
        <td>13/09</td>
        <td>Big regrets</td>
        <td>65</td>
    </tr>
    <tr>
        <td>Toothpaste</td>
        <td>Supermarket</td>
        <td>13/09</td>
        <td>Good</td>
        <td>5</td>
    </tr>
</tbody>
</table>

```

```

    </tr>

    ...

</tbody>

```

```

<table>
  <caption>Items Sold August 2022</caption>
  <thead>
    <tr>
      <td>&nbsp;</td>
      <td>&nbsp;</td>
      <th colspan="3" scope="colgroup">Clothes</th>
      <th colspan="2" scope="colgroup">Accessories</th>
    </tr>
    <tr>
      <td>&nbsp;</td>
      <td>&nbsp;</td>
      <th scope="col">Trousers</th>
      <th scope="col">Skirts</th>
      <th scope="col">Dresses</th>
      <th scope="col">Bracelets</th>
      <th scope="col">Rings</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th rowspan="3" scope="rowgroup">Belgium</th>
      <th scope="row">Antwerp</th>

```

```

        <td>56</td>
        <td>22</td>
        <td>43</td>
        <td>72</td>
        <td>23</td>
    </tr>
    <tr>
        <th scope="row">Gent</th>
        <td>46</td>
        <td>18</td>
        <td>50</td>
        <td>61</td>
        <td>15</td>
    </tr>
    <tr>
        <th scope="row">Brussels</th>
        <td>51</td>
        <td>27</td>
        <td>38</td>
        <td>69</td>
        <td>28</td>
    </tr>
    <tr>
        <th rowspan="2" scope="rowgroup">The Netherlands</th>
        <th scope="row">Amsterdam</th>
        <td>89</td>
        <td>34</td>
        <td>69</td>
        <td>85</td>
        <td>38</td>
    </tr>
    <tr>
        <th scope="row">Utrecht</th>
        <td>80</td>
        <td>12</td>
        <td>43</td>
        <td>36</td>
        <td>19</td>
    </tr>
</tbody>
<tfoot>
    <th colspan="2">Total</th>
    <td>322</td>
    <td>113</td>
    <td>243</td>
    <td>323</td>
    <td>123</td>
</tfoot>
</table>

```

## IX. Les formulaires

[https://developer.mozilla.org/fr/docs/Learn/Forms/Your\\_first\\_form](https://developer.mozilla.org/fr/docs/Learn/Forms/Your_first_form)

<https://web.dev/learn/forms/>

On arrive aux limites du langage HTML car il faudra ensuite pouvoir analyser les informations que le visiteur a saisi :

- Q1 : Comment envoyer le texte saisi par le visiteur ?
- Q2 : Comment traitées les données envoyées ?

### 1. Structure générale d'un formulaire. Exemple zone de saisie monoligne

Les formulaires s'écrivent dans les balises `<form></form>`. Il va falloir ajouter deux attributs à la balise `<form>` pour répondre à ces deux questions.

- `method` : cet attribut indique par quel moyen les données vont être envoyées (Q1).
  - o `method="get"` : peu adaptée car limitée à 255 caractères car transite par barre d'adresse.
  - o `method="post"` : la plus utilisée car peut envoyer un grand nombre d'informations.
- `action` : adresse de la page ou du programme qui va traiter les infos (Q2). La page se charge d'envoyer un email avec le message ou de les enregistrer dans une base de données par exemple. C'est impossible en HTML/CSS, il faudra utiliser un autre langage comme PHP.

```
<form method="post" action="traitement.php">
  <div>
    <label for="pseudo">Votre pseudo :</label>
    <input type="text" name="pseudo" id="pseudo" placeholder="Ex : Flo" size="30"
maxlength="10" />
  </div>
</form>
```

Les champs d'un formulaire sont donc composés d'un `<label></label>` suivi d'un `<input />`.

Les attributs

- `"id"` sert à identifier l'élément HTML pour y accéder et le manipuler,
- `"name"` réfère à la variable du formulaire que l'élément concerne.
- L'attribut `"for"` sert à lier le label au champ. Il doit donc avoir la même valeur que `"id"`.

Dans la balise `<input>` il peut y avoir d'autres attributs :

- `type` : pour préciser la nature de l'entrée. Ici du texte.
- `size` : pour agrandir le champ
- `maxlength` pour limiter le nombre de caractère que l'on peut saisir
- `value` pour préremplir le champ avec une valeur par défaut
- `placeholder` : donner une indication dans le champ (disparaît dès que le visiteur aura cliqué dans le champ)

### Pour une zone de saisie multiligne :

On utilise la balise `<textarea> </textarea>` à la place de `<input>`.




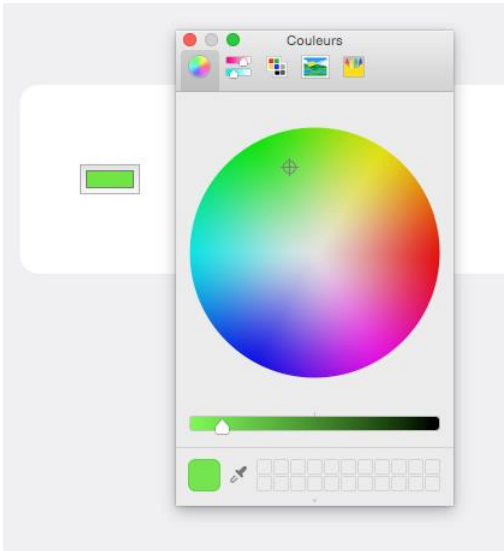
```
<label for="bio">Présentez-vous :</label><br>
<textarea id="bio" name="bio" rows="3" cols="30" placeholder="I like coding on the
beach..."></textarea>
```

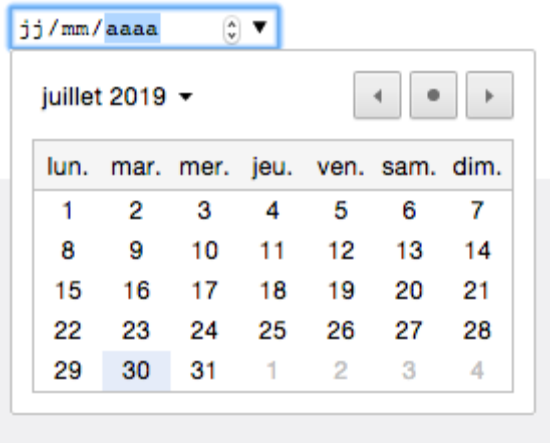
On peut modifier la taille de la zone `<textarea>` :

- En CSS avec les propriétés `width` et `height`

## 2. Les différentes zones de saisie

L'attribut "type" de la balise <input> peut prendre d'autres valeurs depuis HTML5. Attention, certains navigateurs ne prennent pas encore en charge toutes ces nouvelles fonctionnalités. Ils se comporteront comme si on avait saisi type="text" donc autant les utiliser tout de même. Il n'y aura pas d'erreurs.

<p>Un mot de passe :</p> <pre>&lt;input type="password" name="pass" id="pass" pattern="[a-z0-5]{8,}"/&gt;</pre>	<p>Les caractères saisis ne s'afficheront donc pas.</p>
<p>Un email</p> <pre>&lt;input type="email" /&gt;</pre> 	<p>Le champ semblera identique mais le navigateur sait que l'utilisateur doit saisir un email.</p> <p>Firefox entourera de rouge un champ mal renseigné.</p> <p>Certains navigateurs mobiles afficheront un clavier adapté à la saisie d'un email. Avec le @.</p>
<p>Une URL</p> <pre>&lt;input type="url" /&gt;</pre>	<p>Même principe que précédemment.</p> <p>Les navigateurs mobiles afficheront par exemple un clavier adapté à la saisie d'une URL</p>
<p>Un numéro de téléphone</p> <pre>&lt;input type="tel" /&gt;</pre>	<p>Même principe que précédemment.</p> <p>Les navigateurs mobiles afficheront par exemple un clavier adapté à la saisie d'un numéro de téléphone.</p>
<p>Un nombre</p> <pre>&lt;input type="number" /&gt;</pre> 	<p>Un champ s'affichera avec en général des petites flèches pour changer la valeur.</p> <p>On peut personnaliser ce champ avec les attributs :</p> <ul style="list-style-type: none"><li>- Min : valeur minimale autorisée</li><li>- Max : valeur maximale autorisée</li><li>- Step : pas de déplacement.</li></ul>
<p>Un curseur</p> <pre>&lt;input type="range" /&gt;</pre> 	<p>Un curseur s'affichera.</p> <p>On peut personnaliser ce champ avec les attributs :</p> <ul style="list-style-type: none"><li>- Min : valeur minimale autorisée</li><li>- Max : valeur maximale autorisée</li><li>- Step : pas de déplacement.</li></ul>
<p>Une couleur</p> <pre>&lt;input type="color" /&gt;</pre> 	<p>Attention tous les navigateurs ne connaissent pas encore ce type de champ mais la compatibilité progresse.</p>

<p>Une date</p> <pre>&lt;input type="date" /&gt;</pre> 	<p>Il existe différents type de champs possibles :</p> <ul style="list-style-type: none"> <li>- Date</li> <li>- Time</li> <li>- Week</li> <li>- Month</li> <li>- Datetime</li> <li>- Datetime-local</li> </ul>
<p>Une recherche</p> <pre>&lt;input type="search" /&gt;</pre>	<p>Le navigateur décide ensuite comment afficher le champ de recherche. Ainsi, il peut ajouter une petite loupe au champ pour signifier que c'est un champ de recherche, et éventuellement mémoriser les dernières recherches effectuées par le visiteur.</p>
<p>Un fichier</p> <pre>&lt;input type="file" /&gt;</pre>	<p>Permet à l'utilisateur de pouvoir envoyer un fichier. On peut rajouter <i>multiple</i> pour autoriser l'utilisateur à mettre plusieurs fichiers.</p>

### 3. Les cases à cocher : type= "checkbox"

Ce sont des éléments qui demandent au visiteur de faire un choix parmi une liste de possibilités :

Cochez les aliments que vous aimez manger :

- ☒ Frites
- ☒ Steak haché
- ☐ Epinards
- ☐ Huitres

```
<form method="post" action="traitement.php">
  <p>
    Cochez les aliments que vous aimez manger :<br />
    <input type="checkbox" name="frites" id="frites" /> <label
for="frites">Frites</label><br />
    <input type="checkbox" name="steak" id="steak" /> <label for="steak">Steak
haché</label><br />
    <input type="checkbox" name="epinards" id="epinards" /> <label
for="epinards">Epinards</label><br />
    <input type="checkbox" name="huitres" id="huitres" /> <label
for="huitres">Huitres</label>
  </p>
</form>
```

Ici on donne un nom différent à chaque case à cocher. Ce qui permettra d'identifier celles qui ont été cochées par le visiteur.

On peut faire en sorte qu'une case soit cochée par défaut avec l'attribut *"checked"* qui n'attend pas de valeur.

```
<input type="checkbox" name="choix" checked />
```

On peut également procéder comme cela :

```
<fieldset class="toppings">
  <legend>What toppings would you like?</legend><br>
  <input type="checkbox" name="topping" id="lettuce" value="lettuce">
  <label for="lettuce">Lettuce</label>
  <input type="checkbox" name="topping" id="tomato" value="tomato">
  <label for="tomato">Tomato</label>
  <input type="checkbox" name="topping" id="onion" value="onion">
  <label for="onion">Onion</label>
</fieldset>
```

On donne le même nom à chaque checkbox mais on met l'attribut "value" auquel on affecte donc une valeur.

Lorsque le formulaire est envoyé les données sont transmises sous la forme *topping=lettuce&topping=tomato* si deux cases sont cochées par exemple.

#### 4. Les zones d'options : type="radio"

Elles permettent de faire un choix (et un seul !) parmi une liste de possibilités. Elles doivent être organisées en groupes. Les options d'un même groupe possèdent le même nom (attribut name), mais une valeur (attribut value) différente. Cela permet de ne pouvoir en choisir qu'un seul. Sinon on pourrait tout cocher.

```
<form method="post" action="traitement.php">
  <p>Veuillez indiquer la tranche d'âge dans laquelle vous vous situez :<br>
    <input type="radio" name="age" value="moins15" id="moins15" /> <label
for="moins15">Moins de 15 ans</label><br />
    <input type="radio" name="age" value="medium15-25" id="medium15-25" /> <label
for="medium15-25">15-25 ans</label><br />
    <input type="radio" name="age" value="medium25-40" id="medium25-40" /> <label
for="medium25-40">25-40 ans</label><br />
    <input type="radio" name="age" value="plus40" id="plus40" /> <label
for="plus40">Encore plus vieux que ça ?!</label></p>
</form>
```

L'attribut *checked* est aussi disponible pour sélectionner une valeur par défaut.

Si on a deux zones d'options différentes, il faut donner un *name* unique à chaque groupe :

```
<form method="post" action="traitement.php">
  <p>Veuillez indiquer la tranche d'âge dans laquelle vous vous situez :<br />
    <input type="radio" name="age" value="moins15" id="moins15" /> <label
for="moins15">Moins de 15 ans</label><br />
    <input type="radio" name="age" value="medium15-40" id="medium15-40" /> <label
for="medium15-40">15-40 ans</label><br />
    <input type="radio" name="age" value="plus40" id="plus40" /> <label
for="plus40">Encore plus vieux que ça ?!</label></p>
  <p>Sur quel continent habitez-vous ?<br />
    <input type="radio" name="continent" value="europe" id="europe" /> <label
for="europe">Europe</label><br />
    <input type="radio" name="continent" value="amerique" id="amerique" /> <label
for="amerique">Amérique</label><br />
    <input type="radio" name="continent" value="autre" id="autre" /> <label
for="autre">Autre</label></p>
</form>
```

## 5. Les listes déroulantes

Les listes déroulantes sont un autre moyen élégant de faire un choix parmi plusieurs possibilités. On va utiliser la balise `<select></select>`. On ajoute un attribut *name* à la balise pour donner un nom à la liste.

A l'intérieur de la balise `<select>`, on va ajouter des balises `<option></option>` (une par choix possible). On ajoute à chacune d'elle un attribut *value* pour identifier ce que le visiteur a choisi.

```
<form method="post" action="traitement.php">
  <p>
    <label for="pays">Dans quel pays habitez-vous ?</label><br />
    <select name="pays" id="pays">
      <option value="france">France</option>
      <option value="espagne">Espagne</option>
      <option value="italie">Italie</option>
      <option value="royaume-uni">Royaume-Uni</option>
      <option value="canada">Canada</option>
      <option value="etats-unis">États-Unis</option>
      <option value="chine">Chine</option>
      <option value="japon">Japon</option>
    </select>
  </p>
</form>
```

Dans quel pays habitez-vous ?



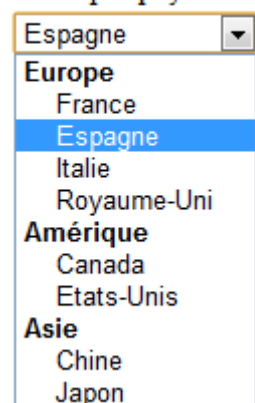
Pour sélectionner une option par défaut, on utilise l'attribut *selected*.

```
<option value="canada" selected>Canada</option>
```

On peut regrouper les options avec la balise `<optgroup></optgroup>`. Dans cet exemple, on pourrait classer les pays par continent :

```
<form method="post" action="traitement.php">
  <p>
    <label for="pays">Dans quel pays habitez-vous
?</label><br />
    <select name="pays" id="pays">
      <optgroup label="Europe">
        <option value="france">France</option>
        <option value="espagne">Espagne</option>
        <option value="italie">Italie</option>
        <option value="royaume-uni">Royaume-Uni</option>
      </optgroup>
      <optgroup label="Amérique">
        <option value="canada">Canada</option>
        <option value="etats-unis">Etats-Unis</option>
      </optgroup>
      <optgroup label="Asie">
        <option value="chine">Chine</option>
        <option value="japon">Japon</option>
      </optgroup>
    </select>
  </p>
</form>
```

Dans quel pays habitez-vous ?



Les groupes ne peuvent pas être sélectionnés. On ne peut pas choisir Europe. Seuls les noms des pays sont sélectionnables.



## 6. Exemple de formulaire de contact basique

```
<section id="contact">
  <h2>Contact the Royal Kingdom</h2>
  <p>Fill in the form bellow to contact the Royal Kingdom</p>
  <form id="contactForm">
    <label for="name">Name</label><br>
    <input type="text" id="name" name="name" value="your name"><br>
    <label for="email">E-mail</label><br>
    <input type="email" id="email" name="email" value="myname@example.com"><br>
    <label for="message">Your message to the King</label><br>
    <textarea id="message" name="message">Write your message here</textarea><br>
    <input type="submit" value="submit">
  </form>
</section>
```

Si le formulaire est gros et comporte beaucoup de champs, il peut être utilisé de les regrouper au sein de plusieurs balises `<fieldset>`, qui peut contenir une légende avec la balise `<legend>` :

Vos coordonnées

Quel est votre nom ?

Quel est votre prénom ?

Quel est votre e-mail ?

Votre souhait

Faites un souhait que vous voudriez voir exaucé :

☐ Etre riche

☐ Etre célèbre

☐ Etre **encore** plus intelligent

☐ Autre...

```
<form method="post" action="traitement.php">
  <fieldset>
    <legend>Vos coordonnées</legend> <!-- Titre du fieldset -->
    <label for="nom">Quel est votre nom ?</label><br>
    <input type="text" name="nom" id="nom" /><br>
    <label for="prenom">Quel est votre prénom ?</label><br>
    <input type="text" name="prenom" id="prenom" /><br>
    <label for="email">Quel est votre e-mail ?</label><br>
    <input type="email" name="email" id="email" />
  </fieldset>
  <fieldset>
    <legend>Votre souhait</legend> <!-- Titre du fieldset -->
    <span>Faites un souhait que vous voudriez voir exaucé :</span><br>
    <input type="radio" name="souhait" value="riche" id="riche" />
    <label for="riche">Etre riche</label><br>
    <input type="radio" name="souhait" value="celebre" id="celebre" />
    <label for="celebre">Etre célèbre</label><br>
    <input type="radio" name="souhait" value="intelligent" id="intelligent" />
    <label for="intelligent">Etre encore plus intelligent</label><br>
    <input type="radio" name="souhait" value="autre" id="autre" />
    <label for="autre">Autre...</label>
    <label for="precisions">Si "Autre", veuillez préciser :</label><br>
    <textarea name="precisions" id="precisions" cols="40" rows="4">
  </fieldset>
</form>
```

## 7. Informations supplémentaires

Pour placer automatiquement le curseur dans l'un des champs du formulaire, on utilise l'attribut *autofocus* :

```
<input type="text" name="prenom" id="prenom" autofocus />
```

Pour rendre un champ obligatoire, on lui donne l'attribut *required* :

```
<input type="text" name="prenom" id="prenom" required />
```

Le navigateur indiquera alors au visiteur, si le champ est vide, qu'il doit être impérativement rempli.

Les anciens navigateurs, qui ne reconnaissent pas cet attribut, enverront le contenu du formulaire sans vérification. Il sera alors nécessaire de compléter les tests avec des scripts JavaScript.

On dispose de pseudo-formats en CSS pour changer le style des éléments requis (*:required*) et invalides (*:invalid*). Et aussi le pseudo-format *:focus* qui permet de changer l'apparence d'un champ lorsque le curseur se trouve à l'intérieur.

```
:required
{
    background-color: red;
}
```

## 8. Envoyer le formulaire

Il faut créer un bouton d'envoi. On utilise aussi la balise `<input />` avec plusieurs valeurs possibles à l'attribut *type* :

- *type="submit"* : le principal bouton d'envoi de formulaire. C'est celui qu'on utilise le plus souvent. Il conduit le visiteur à la page indiquée dans l'attribut *action* du formulaire.
- *type="reset"* : remise à zéro du formulaire.
- *type="image"* : équivaut à *submit* mais présenté sous forme d'image. Il faut rajouter l'attribut *src* pour indiquer l'URL de l'image.
- *type="button"* : bouton générique, qui n'a pas défaut aucun effet. Il est en général géré en Javascript pour exécuter des actions sur la page.

On peut changer le texte affiché à l'intérieur des boutons avec l'attribut *value*.

```
<input type="submit" value="Envoyer" />
```

On ne peut donc pas créer une page avec un formulaire uniquement en HTML. Il est nécessaire d'apprendre un nouveau langage comme le PHP pour pouvoir récupérer les informations saisies et décider quoi en faire.

## 9. Vérifier le formulaire

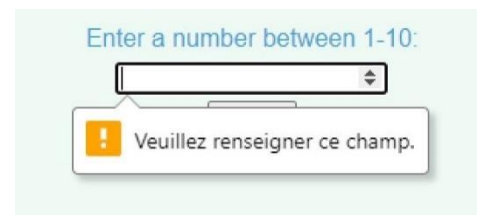
Il y a différents types de validation :

- La validation côté serveur qui nécessite d'envoyer les données sur le serveur pour qu'il les vérifie. C'est ce que se passe pour les login et mot de passe.
- La validation côté client : elle a lieu avant que les données soient envoyées sur le serveur. Cela permet de gagner du temps et d'éviter d'attendre que les données soient validées ou rejetées par le serveur et donc de faire remplir à nouveau le formulaire par l'utilisateur si elles sont rejetées. Cela peut aussi protéger de code ou de données malveillantes.

### a. Champ requis

En cas de champ requis il faut rajouter le mot *required* dans la balise `input`

```
<label for="guess">Enter a number between 1-10:</label><br>
<input type="number" name="guess" id="guess" required><br>
```



b. Minimum ou maximum

```
<input type="number" name="guess" id="guess" min="1" max="10" required><br>
<input type="submit" id="submission" value="Submit">
```

c. Longueur d'un texte

Pour mettre une valeur minimum ou maximum dans une saisie de texte, on utilise les attributs *minlength* et *maxlength*

```
<input id="username" name="username" type="text" minlength="3" maxlength="15" required>
```

d. Correspondance d'un motif

On utilise l'attribut *pattern* auquel on affecte une expression régulière ou *regex*. Exemple valider un numéro de carte bancaire à 14 ou 16 chiffres :

```
<input id="payment" name="payment" type="text" required pattern="[0-9]{14,16}">
```

Pour limiter la saisie qu'à des chiffres ou des lettres :

```
<input id="username" name="username" type="text" required minlength="3" maxlength="15" pattern="[a-zA-Z0-9]+">
```

[https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Regular\\_Expressions](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Regular_Expressions)

<https://towardsdatascience.com/regular-expressions-clearly-explained-with-examples-822d76b037b4>

[https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Regular\\_Expressions/Cheatsheet](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Regular_Expressions/Cheatsheet)

e. Attribute title

L'attribut *title* va permettre d'afficher un message en cas de mauvais pattern et donner une indication à l'utilisateur. Exemple pour un code postal. Un *placeholder* peut aussi faire l'affaire pour montrer le type attendu.

```
<input type="text" id="zip_code" name="zip_code" pattern="(\d{5}([\d-]\d{4})?)"
title="Please enter a valid zip code, example: 65251" required>
```

Pour le style on pourra donc utiliser les pseudo classe *:valid* et *:invalid* pour styliser.

f. Barre de progression <progress>

```
<progress max="100" value="75">75/100</progress>
```

g. Les jauges <meter>

Jaugé délimitée par les attributs *min* et *max*. Cette valeur s'affiche comme une barre.

Les attributs *low* et *high* partage l'intervalle en trois parties :

- La partie inférieure entre *min* et *low* (inclus)
- La partie intermédiaire entre *low* et *high* (inclus)
- La partie supérieure entre *high* et *max* (inclus)

La valeur de l'attribut *optimum* définit la valeur optimale de l'élément <meter> :

- Si *optimum* est dans la partie inférieure, cette partie est privilégiée. Intermédiaire est considérée comme moyenne et supérieure comme la pire.
- Si *optimum* est dans la partie intermédiaire, cette partie est privilégiée. Inférieure et supérieure sont considérées comme moyenne.
- Si *optimum* est dans la partie supérieure, cette partie est privilégiée. Intermédiaire est considérée comme moyenne et inférieure comme la pire.

Les navigateurs implémentant <meter> utilisent ces valeurs pour changer la couleur de la jauge. Elle sera verte si elle est dans la partie privilégiée, jaune si elle est dans la partie moyenne et rouge si elle est dans la pire partie.

```
<meter min="0" max="100" value="75" low="33" high="66" optimum="50">75</meter>
```

## X. Intégrer de la vidéo et de l'audio

En HTML5, les balises <video> et <audio> ont été créées.

Les formats audios :

- MP3 : un des plus vieux formats mais l'un des plus compatibles. Il est donc très utilisé.
- AAC : utilisé majoritairement par Apple sur iTunes. Bonne qualité. iDevice savent les lire sans soucis.
- Ogg : très répandu dans le monde du logiciel libre, notamment Linux. Format libre (aucun brevet).
- WAV : non compressé. Il faut éviter de l'utiliser car très volumineux.

La vidéo est plus complexe. On a besoin de 3 éléments :

- Un format conteneur. On le voit à l'extension : AVI, MP4, MKV
- Un codec audio : cf juste avant.
- Un codec vidéo : le format qui va compresser les images. Ces formats sont complexes et on ne peut pas toujours les utiliser gratuitement. Il faut principalement connaître :
  - o H.264 l'un des plus puissants et le plus utilisé aujourd'hui... mais pas 100% gratuit. On peut l'utiliser gratuitement dans certains cas (diffusion de vidéos sur un site web personnel), mais il y a un flou juridique. Il est donc risqué de l'utiliser à tout va.
  - o Ogg Theora : gratuit et libre de droit, mais moins puissant que H.264. Bien reconnu sous Linux mais sous windows il faut installer des programmes pour pouvoir le lire.
  - o WebM : codec gratuit et libre de droits. Plus récent. Il est proposé par Google. C'est le concurrent le plus sérieux du H.264.

Il est conseillé de proposer si possible chaque vidéo dans plusieurs formats pour qu'elle soit lisible sur un maximum de navigateur.

### 1. Insérer un élément audio

```
<audio src="musique.mp3"></audio>
```

Si on ne fait que cela, il ne se passe rien. Il faut compléter la balise avec les attributs suivants :

- controls : pour ajouter les boutons "lecture", "pause", et la barre de défilement. Ils ne sont pas présents par défaut car certains sites web préfèrent créer eux-mêmes leurs propres boutons et commander la lecture avec du JavaScript.
- width : largeur de l'outil de lecture audio
- loop : la musique sera jouée en boucle
- autoplay : la musique sera jouée dès le chargement de la page (il faut éviter d'en abuser car cela peut être irritant d'arriver sur un site qui joue de la musique tout seul).
- preload : indique si la musique peut être préchargée dès le chargement de la page ou non. Cet attribut peut prendre les valeurs suivantes :
  - o Auto : (défaut). Le navigateur décide s'il doit précharger toute la musique, uniquement les métadonnées ou rien du tout.
  - o Metadata : charge uniquement les métadonnées (durée, etc...)
  - o None : pas de préchargement. Utile si on ne veut pas gaspiller de bande passante sur notre site.

On ne peut pas forcer le préchargement de la musique. C'est toujours le navigateur qui décide. Les navigateurs mobiles, par exemple, ne préchargent jamais la musique pour économiser la bande passante.

```
<audio src="valkyries.mp3" controls>Veuillez mettre à jour votre navigateur !</audio>
```

Le message ne s'affiche que si le navigateur ne gère pas cette balise.

Il est conseillé de proposer plusieurs versions du fichier audio pour éviter les problèmes de compatibilité :

```
<audio controls>
  <source src="hype_home.mp3">
  <source src="hype_home.ogg">
</audio>
```

## 2. Insérer un élément vidéo

```
<video src="sintel.webm"></video>
```

Encore une fois, rien ne se passe. Il faut rajouter des attributs, la plupart similaire à la balise <audio> :

- poster : image à afficher à la place de la vidéo tant que celle-ci n'est pas lancée. Par défaut le navigateur prend la première image de la vidéo mais comme s'agit souvent d'une image noire ou d'une image peu représentative de la vidéo, il est conseillé d'en créer une, en faisant tout simplement une capture d'écran d'un moment de la vidéo.
- controls : idem élément audio.
- width : modifier la largeur de la vidéo
- height : modifier la hauteur de la vidéo
- loop : vidéo jouée en boucle
- autoplay : vidéo se lance automatiquement
- preload : idem audio.

Encore une fois on ne peut pas forcer le préchargement de la vidéo. C'est toujours le navigateur qui décide. Les proportions de la vidéo sont toujours conservées. Si on définit une largeur et une hauteur, le navigateur fera en sorte de ne pas dépasser les dimensions indiquées mais il conservera les proportions :

```
<video src="sintel.webm" controls poster="sintel.jpg" width="600"></video>
```

Il est conseillé de mettre un petit message si le navigateur ne reconnaît pas la balise :

```
<video src="sintel.webm" controls poster="sintel.jpg" width="600">
  Il est temps de mettre à jour votre navigateur !
</video>
```

Il est également conseillé de proposer différents formats pour la vidéo :

```
<video controls poster="sintel.jpg" width="600">
  <source src="sintel.mp4">
  <source src="sintel.webm">
  <source src="sintel.ogv">
</video>
```

Les iPhone, iPad ne reconnaissent que le format H.264 (fichier ...mp4) uniquement si celui-ci apparaît en premier dans la liste. Il est donc conseillé de le mettre en premier dans la liste.

Il n'est pas possible de protéger une vidéo pour en empêcher le téléchargement. Le navigateur doit bien la télécharger pour que l'utilisateur puisse la voir.

## 3. Balise iframe

```
<iframe
  id="video"
  height="315"
  src="https://www.youtube-
nocookie.com/embed/y8Yv4pn07qc?rel=0&controls=0&showinfo=0"
  frameborder="0"
  allowfullscreen
></iframe>
```

La balise ifram permet d'ajouter un cadre dans lequel on peut mettre un vidéo mais aussi une carte par exemple ou une autre page web.

## XI. Outils HTML

### 1. Minifier

Pour transformer ça :

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <title>Hello World</title>
  </head>
  <body>
    <h1>Hello, World</h1>
  </body>
</html>
```

En ça :

```
<!doctypehtml><html lang=en><meta charset=UTF-8><meta content="width=device-width,initial-scale=1"name=viewport>
<meta content="ie=edge"http-equiv=X-UA-Compatible><title>Hello World</title><h1>Hello,
World</h1>
```

On peut utiliser cet outils :

- <https://kangax.github.io/html-minifier/>
- Ce plugin Visual Studio : <https://marketplace.visualstudio.com/items?itemName=HookyQR.minify>
- Parcel, Webpack Encore....

### 2. Compresser les images

Les images trop lourdes sont souvent cause de mauvaises performances du site internet.

Sur windows : directement dans la visionneuse via le menu ou un click-droit

Sur Linux en utilisant nautilus :

- nautilus --version
- sudo apt install imagemagick
- sudo apt install nautilus-image-converter
- click droit sur l'image pour resize ou rotate.

Via des sites internet comme <https://tinypng.com/>

### 3. Evaluer la performance de son site internet

En utilisant des sites internet :

- <https://gtmetrix.com/>
- <https://web.dev/measure/>

Ou des outils de développement chrome :

- <https://developer.chrome.com/docs/lighthouse/overview/#devtools>

## 4. SEO

Pour savoir si son site est sur google, il suffit de rechercher son url dans google.

### **La balise title :**

Il ne faut jamais la laisser vide ou avec des noms de titre par défaut. Il faut écrire un titre clair, et placer des mots clés dans les titres des pages de cette façon : Mot clé principal - Mot clé secondaire | Nom du site.

Exemple : Développeur web fullstack - Javascript PHP Symfony Lille | Florent Vasseur

### **Meta description :**

La balise méta description est très importante pour le SEO. Elle est utilisée par google pour afficher des informations dans leur page de résultats.

<https://www.searchmetrics.com/glossary/meta-description/>

Une bonne pratique est d'y ajouter un contenu autour de 150-160 caractères. On peut y placer des mots clés en rapport avec le sujet du site web :

```
<meta name="description" content="Full Stack Web Developer Portfolio. Front-end & Back-end Web Developer based in Lille, France. Tech stack include React, JavaScript, Node, SQL.">
```

### **Insérer des mots clés dans le contenu :**

Le contenu doit avoir un grand nombre de mots clés en rapport au sujet. Il faut que ce soit unique et pas répété dans les différentes pages de son site web. Google vérifie que le contenu soit unique, c'est pourquoi il faut éviter de copier/coller du contenu depuis des articles de presse ou d'autres médias.

### **Utiliser les bonnes balises :**

Il faut respecter la sémantique HTML. Notre code HTML doit aider le robot de google à comprendre la structure de notre site web. Il faut donc utiliser les titres h1, h2, h3 de la bonne manière.

### **Utiliser du texte cohérent pour les liens :**

Le texte des liens doit être clair. Il est plus judicieux d'utiliser du texte tel que :

- Accéder à mon portfolio de développeur web plutôt que cliquer ici

Il est possible d'utiliser l'attribut rel="nofollow" pour indiquer cette page ne doit pas être visitée par les robots de google.

```
<a href="https://cheese.example.com/Appenzeller_cheese" rel="nofollow">Portfolio</a> cheese.
```

### **Ne pas oublier alt dans les images :**

C'est aussi utilisé par les robot de google. Il faut donc aussi utiliser des mots clés dans le alt.

Il faut également donner un nom significatif pour nos images.

<https://www.business2community.com/seo/15-seo-best-practices-need-know-tips-use-02018245>