

Chapitre 3

Les notions avancées de HTML5

Introduction

Dans ce chapitre, nous allons voir les notions **HTML** plus avancées tels que les tableaux et les formulaires.

I. Les tableaux

1. À quoi servent-ils ?

Un tableau est une suite de lignes et de colonnes qui forment un ensemble de cellules. Les tableaux servent avant tout à présenter des données tabulaires dont voici un exemple :

Elève	Math	Web	SGBD	Sport
Ali	14	10	11	11
Mohamed	15	14.5	10.5	12
Bilel	7.5	18	12	14
Imène	18.5	19	19	16

TABLEAU 1 : EXEMPLE D'UN TABLEAU

Malgré cela, l'emploi le plus fréquent des tableaux reste la mise en page des documents. En effet, l'implémentation des tableaux étant assez intuitive, s'ajoute à cela les éditeurs WYSIWYG des logiciels de création de sites qui génèrent ce genre de code automatiquement.

2. Du tableau simple au tableau complexe

a. Les bases d'un tableau

Un tableau est délimité par la balise "**<table>**". Le contenu d'un tableau **HTML** est généré horizontalement. C'est-à-dire qu'il n'est pas créé colonne par colonne mais ligne par ligne. Pour créer un tableau, nous commençons par créer des lignes grâce aux balises "**<tr>**". Puis nous insérons dans ces lignes les cellules du tableau grâce aux balises "**<td>**". Le contenu (textes, images, autres balises) se trouve uniquement dans les balises "**<td>**" et ne doit se trouver en aucun cas en dehors.

Donc pour avoir un tableau d'une cellule, soit une ligne et une colonne, on obtient ce code :

Tableau d'une cellule

```
<table border="1">
  <tr> <td>Contenu de la cellule</td> </tr>
</table>
```

Contenu de la cellule

Et pour un tableau de deux lignes et deux colonnes :

Tableau de deux lignes et deux colonnes

```
<table border="1">
<tr>
  <td> Cellule 1</td>
  <td> Cellule 2</td>
</tr>
<tr>
  <td>Cellule 3</td>
  <td>Cellule 4</td>
</tr>
</table>
```

Cellule 1	Cellule 2
Cellule 3	Cellule 4

Par défaut, la bordure n'est pas visible. Dans les codes précédents on a rajouté l'attribut **"border"** à la balise **"<table>"** pour la faire apparaître. On peut évidemment agrandir la bordure en augmentant la valeur.

NB : l'attribut « **border** » est encore toléré en HTML5 mais il est recommandé d'utiliser **CSS** à la place. *Exemple :* **table, th, td {border: 1px solid black;}**

Pour définir la distance qu'il y a entre les bordures de cellules adjacentes d'un tableau on doit utiliser la propriété CSS « **border-spacing** »

Exemple :

```
<table border="2" style="border-spacing: 10px;">
  <tr>
    <td>1</td>
    <td>2</td>
  </tr>
  <tr>
    <td>3</td>
    <td>4</td>
  </tr>
</table>
```

1	2
3	4

Pour ajouter un espace entre le bord et le contenu on doit utiliser la propriété CSS « **padding** » avec l'élément souhaité (table, td, th, etc).

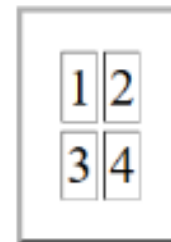
Exemples :

```
<table border="2">
  <tr>
    <td style="padding:10px">1</td>
    <td style="padding:10px">2</td>
  </tr>
  <tr>
    <td style="padding:10px">3</td>
    <td style="padding:10px">4</td>
  </tr>
</table>
```



1	2
3	4

```
<table border="2" style="padding: 10px;">
  <tr>
    <td>1</td>
    <td>2</td>
  </tr>
  <tr>
    <td>3</td>
    <td>4</td>
  </tr>
</table>
```



1	2
3	4

b. Le titre d'un tableau

Pour décrire le contenu d'un tableau, par exemple, pour expliquer dans notre premier exemple de tableau qu'il s'agit des notes de la classe de TI20 (voir tableau 1), il faut placer la balise "**<caption>**" directement après l'ouverture du tableau. Cette balise prend l'attribut "**align**" qui accepte comme valeurs "**top**" (par défaut), "**bottom**", "**left**" et "**right**", en fonction de l'endroit où l'on veut afficher cette légende.

NB : La balise **<caption>** doit être insérée immédiatement après la balise **<table>**.

c. La ligne d'en-tête d'un tableau

Dans l'exemple des notes de la classe de TI20, on remarque que la première ligne peut faire office d'en-tête. En effet, elle renseigne sur le contenu de chacune des colonnes et n'a donc pas le même genre de contenu que tout le reste du tableau.

Pour préciser au navigateur qu'il s'agit donc de cellules spéciales, on va les modifier en remplaçant pour cette ligne seulement les balises "**<td>**" par des balises "**<th>**".

d. La fusion de lignes et de colonnes d'un tableau

On distingue deux types de fusion : la fusion horizontale où sont mises en jeu deux (ou plus) colonnes, la fusion verticale où sont mises en jeu deux (ou plus) lignes. Dans les deux cas, elle s'opère sur les cellules, donc sur la balise "<td>". Elle se traduit par l'utilisation de l'attribut "**colspan**" dans le cas d'une fusion horizontale et par l'attribut "**rowspan**" dans le cas d'une fusion verticale.

Ces attributs prennent pour valeurs le nombre de cellules à fusionner.

Exemple d'utilisation de colspan

```
<table border="1">
  <tr>
    <td colspan="5"> Cellule qui se repend sur 5 colonnes</td>
  </tr>
  <tr>
    <td>Colonne 1</td>
    <td>Colonne 2</td>
    <td>Colonne 3</td>
    <td>Colonne 4</td>
    <td>Colonne 5</td>
  </tr>
</table>
```

Cellule qui se répend sur 5 colonnes				
Colonne 1	Colonne 2	Colonne 3	Colonne 4	Colonne 5

La somme des « **colspan** » de chaque ligne doit être identique à toutes les autres lignes.

En ce qui concerne l'attribut "**rowspan**", on aurait un code tel que :

Exemple d'utilisation de rowspan

```
<table border="1">
  <tr>
    <td rowspan="2">Cellule sur deux lignes</td>
    <td>Ligne 1 - Colonne 2</td>
    <td>Ligne 1 - Colonne 3</td>
    <td>Ligne 1 - Colonne 4</td>
  </tr>
  <tr>
    <td>Ligne 2 - Colonne 2</td>
    <td>Ligne 2 - Colonne 3</td>
    <td>Ligne 2 - Colonne 4</td>
  </tr>
</table>
```

Cellule sur deux lignes	Ligne 1 - Colonne 2	Ligne 1 - Colonne 3	Ligne 1 - Colonne 4
	Ligne 2 - Colonne 2	Ligne 2 - Colonne 3	Ligne 2 - Colonne 4

On pourrait aller plus loin en combinant les deux techniques sur un seul et même tableau :

Exercice : réalisez le tableau suivant

Cellule 1			Cellule 2
Cellule 3	Cellule 4	Cellule 5	Cellule 6
	Cellule 7	Cellule 8	
	Cellule 9		

Correction

```
<table border="1">
  <tr>
    <td colspan="3">Cellule 1</td>
    <td>Cellule 2</td>
  </tr>
  <tr>
    <td rowspan="3">Cellule 3</td>
    <td>Cellule 4</td>
    <td>Cellule 5</td>
    <td>Cellule 6</td>
  </tr>
  <tr>
    <td>Cellule 7</td>
    <td colspan="2" rowspan="2">Cellule 8</td>
  </tr>
  <tr>
    <td>Cellule 9</td>
  </tr>
</table>
```

On remarque que le code est devenu extrêmement compliqué et que la moindre erreur de fusion de notre part entraînerait une interprétation du tableau complètement fausse par les navigateurs.

e. Regroupement de cellule

Ces petits tableaux suffisent dans la plupart des cas, mais il arrivera que vous ayez besoin de réaliser des tableaux plus complexes. Pour cela il est possible de les diviser en trois parties :

- *En-tête avec la balise <thead>*
 - *Corps du tableau avec la balise <tbody>*
 - *Pied de tableau avec la balise <tbody>*
- *La balise <thead>*

La balise **<thead>** est utile pour regrouper le contenu de l'entête d'un tableau. Elle doit être utilisée avec **<tbody>** et **<tfoot>**.

Les navigateurs peuvent utiliser ces éléments pour permettre le défilement du corps du tableau, indépendamment de la tête et le pied. En outre, lors de l'impression d'un grand tableau qui couvre plusieurs pages, ces éléments peuvent permettre à l'en-tête de table et pied de page à imprimer en haut et en bas de chaque page.

- La balise **<tbody>**

La balise **<tbody>** est utilisée pour regrouper le contenu du corps dans une table HTML.

- La balise **<tfoot>**

La balise **<tfoot>** est utilisé pour le contenu de pied de groupe dans un tableau HTML.

Liste des moyennes

Elève	Math	Web	SGBD	Sport
Ali	14	10	11	11
Mohamed	15	14.5	10.5	12
Bilel	7.5	18	12	14
Imène	18.5	19	19	16
Elève	Math	Web	SGBD	Sport

Diagram illustrating the structure of the table with HTML tags:

- <caption>** and **<thead>** point to the first row (headers).
- <tbody>** points to the body rows (data rows).
- <tfoot>** points to the last row (summary row).

Code du tableau

```
<table style="width:40%;">
  <caption>Liste des moyennes</caption>

  <thead> <!-- En-tête du tableau -->
    <tr>
      <th>Elève</th>
      <th>Math</th>
      <th>Web</th>
      <th>SGBD</th>
      <th>Sport</th>
    </tr>
  </thead>

  <tfoot> <!-- Pied de tableau -->
    <tr>
      <th>Elève</th>
      <th>Math</th>
      <th>Web</th>
      <th>SGBD</th>
      <th>Sport</th>
    </tr>
  </tfoot>
```

```

<tbody> <!-- Corps du tableau -->
  <tr>
    <td>Ali</td>
    <td>14</td>
    <td>10</td>
    <td>11</td>
    <td>11</td>
  </tr>
  <tr>
    <td>Mohamed</td>
    <td>15</td>
    <td>14.5</td>
    <td>10.5</td>
    <td>12</td>
  </tr>
  <tr>
    <td>Bilel</td>
    <td>7.5</td>
    <td>18</td>
    <td>12</td>
    <td>14</td>
  </tr>
  <tr>
    <td>Imène</td>
    <td>18.5</td>
    <td>19</td>
    <td>19</td>
    <td>16</td>
  </tr>
</tbody>
</table>

```

NB : il est conseillé d'écrire les balises dans l'ordre suivant : <thead> <tfoot> <tbody>. Le navigateur se chargera d'afficher chaque élément au bon endroit.

3. L'utilité des tableaux aujourd'hui

Grâce aux techniques de fusion vues précédemment, il y a quelques années, la tendance était de profiter de ce système pour mettre en page assez rapidement l'ensemble de son site web. Cependant, si cette méthode peut séduire, elle est fortement déconseillée.

En effet, les tableaux sont censés (et doivent) être utilisés uniquement pour l'affichage de données tabulaires.

Dans le cas d'une mise en page, on conseille d'adopter la solution du couple "**DIV + CSS**". (A Voir dans le chapitre concernant les feuilles de styles CSS).

On peut facilement voir dans notre dernier exemple où l'on combine les deux techniques de fusion, que notre code sera difficilement maintenable dans le cas d'une retouche ou de l'ajout d'une nouvelle cellule.

Voici quelques raisons pour lesquelles il ne faut pas utiliser les tableaux pour la mise en page :

- Ils alourdissent les pages HTML : beaucoup de balises pour finalement y mettre peu de contenu.
- Ils sont très difficiles à maintenir dans le cas d'une retouche.
- Ils nuisent à l'accessibilité : problème d'interprétation des navigateurs spécialisés destinés aux malvoyants par exemple.
- Ils sont mal gérés par les navigateurs à l'impression dans le cas de tableaux complexes.

II. Les formulaires

1. À quoi servent-ils ?

Les formulaires servent à envoyer des données au serveur. Ces informations sont remplies en général par un visiteur. Par exemple : inscription sur un site, un formulaire de contact, etc.

Les formulaires sont dotés de divers contrôles comme des champs de saisie, des boutons, des listes de choix, etc., qui permettent au visiteur d'interagir avec la page qu'il est en train de consulter.

2. La balise `form`

Tous les champs de formulaires doivent se trouver dans cette balise "`<form>`".

C'est cette balise qui va permettre de renseigner la page de destination du formulaire, à l'aide de l'attribut "**action**", qui peut prendre pour valeurs une URL en absolu ou en relatif, voire même une adresse e-mail.

Elle est aussi utilisée pour préciser le mode d'envoi des données grâce à l'attribut "**method**". Cet attribut prend deux valeurs différentes qui sont "**GET**" et "**POST**". Les valeurs envoyées en "GET" passent par l'URL alors qu'en "POST" elles sont envoyées de manière transparente.

Exemple de balise `form`

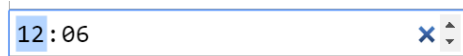
```
<form action="envoi-formulaire.php" method="post">
    <!-- Contenu du formulaire -->
</form>
```


3. Les champs texte mono-lignes

On distingue deux types de champs texte mono-lignes : les champs de type "**texte**" et les champs de type "**mot de passe**".

Le champ de type "**texte**" est l'élément de base d'un formulaire, il permet d'introduire du texte court sur une ligne, comme un *login*, un *e-mail*, ou autres, alors que le champ de type "**mot de passe**" permet de renseigner un mot de passe, dont l'affichage sera camouflé aux yeux des autres personnes. Le champ de texte mono-ligne est caractérisé par la balise "**<input>**" et l'attribut "**type**" dont la valeur sera "**text**", "**password**", **tel**, **url**, **email**, **date**, **time**, **datetime**, **month**, **week**, **color**, **range**, **search** et **number**.

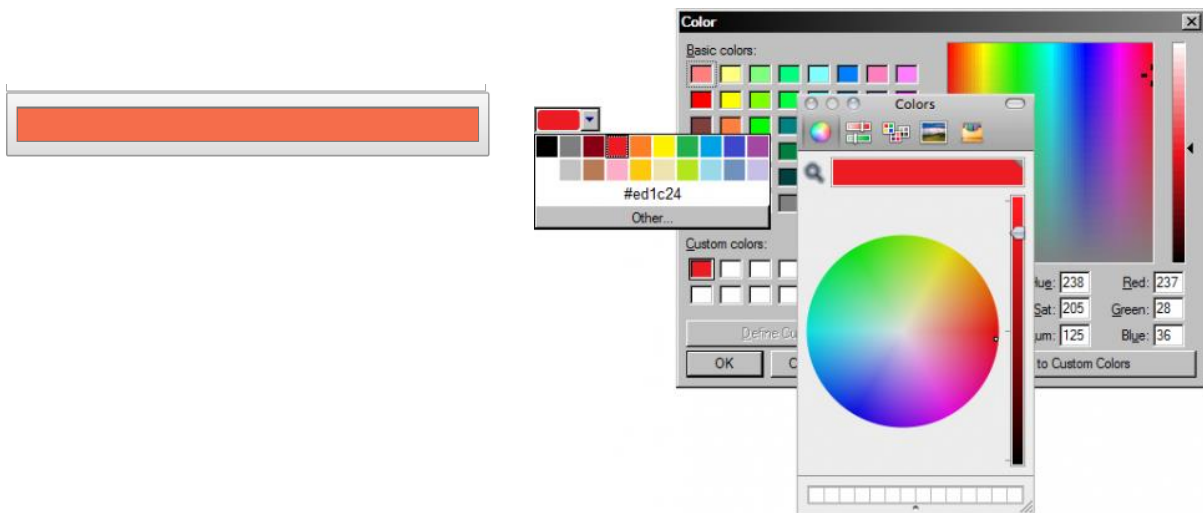
- **text**: définit un champ d'entrée de texte d'une ligne
- **password**: définit un champ de mot de passe (caractères sont masqués).
- **tel**: définit un champ de saisie d'un numéro de téléphone. Sur certains navigateurs de SmartPhone, l'entrée du numéro de téléphone est facilitée par le basculement à un clavier de type numérique.
- **url**: définit un champ de saisie d'une URL. le navigateur attend cette fois un format bien spécifique devant respecter un pattern de type "**url**". Tous les types d'URL sont admis (ftp://, mailto:, http://, etc.).
- **email**: définit un champ pour une adresse e-mail. Ce champ attend au minimum un caractère (caractère non accentué comprenant les séparateurs tiret ou underscore) suivi d'un @ suivi à son tour d'un caractère.
- **date**: est utilisé pour les champs d'entrée qui devrait contenir une date.
- **time**: permet de renseigner une heure, avec plus ou moins de précision. Les attributs **min** et **max** peuvent être appliqués. L'attribut **step** permet d'ajouter un pas
- **datetime**: est une combinaison des deux type **date** et **time**. Il permet de renseigner une date et une heure précise dans un même champ (visuel)
- **month**: permet de renseigner un mois dans une année. La liste des attributs compatibles reste la même que pour le type date. Le format attendu correspond à cette syntaxe : **aaaa-mm** (ex: 2018-01).
- **week**: permet de renseigner une semaine dans une année. Il attend un format de type **2018W05** qui correspond à l'année et au numéro de semaine dans celle-ci.
- **number**: permet de renseigner une valeur numérique. Lorsque ce type de champ est pris en charge par le navigateur, celui-ci renvoie la valeur numérique renseignée



- **range:** Ce champ propose un contenu évalué approximatif. Par défaut la valeur la plus basse est **0**, la plus haute **100**. Ce type de champ retourne une valeur numérique qui correspond à la position du curseur. Les attributs **value**, **step**, **min** et **max** peuvent être utilisés.



- **color:** permet de transformer le champ pour sélectionner un code couleur dans une palette. N'est pas compatible avec tous les navigateurs



On renseignera aussi l'attribut **"name"** qui sera le nom du champ, celui qui va permettre de retrouver la valeur lors de l'envoi du formulaire.

D'autres attributs sont disponibles, comme **"value"** qui permet d'attribuer une valeur par défaut au champ, **"size"** qui permet de préciser la taille du champ en nombre de caractères, **"maxlength"** qui sert à limiter le nombre de caractères possibles, **"readonly"** qui prend pour seule valeur **"readonly"** et verrouille le champ, **"disabled"** qui prend pour seule valeur **"disabled"** et permet de désactiver le champ. Lorsqu'un champ est défini en **"disabled"**, sa valeur n'est pas envoyée au serveur.

Voici quelques exemples de champs texte :

Exemples de champs texte

```
<input type="text" size="5" name="code_postal" maxlength="5">
<input type="text" name="nom_champ" readonly="readonly" value="Champ verrouillé">
<input type="text" name="login" value="Entrez ici votre login" size="20">
<input type="password" name="mdp">
```

a. Les champs texte mul-tilignes

Contrairement au champ de texte mono-ligne, le `<textarea>` est une balise double et son contenu est écrit entre les balises ouvrantes et fermantes et non dans un attribut value.

Deux attributs sont nécessaires au "`<textarea>`" : "`rows`" qui contient le nombre de lignes et "`cols`" qui contient le nombre de colonnes.

Tout comme le champ de texte mono-ligne, on peut, rajouter les attributs "`name`", "`readonly`" et "`disabled`".

L'attribut "`maxlength`" n'existe pas pour cette balise, et pour en limiter le nombre de caractères il est possible de passer par du JavaScript.

Exemples de champs texte multilignes

```
<textarea name="commentaire" rows="5" cols="80"></textarea>
<textarea name="sujet" rows="6" cols="100"> Texte par défaut</textarea>
```

Les attributs liés à la balise `<textarea>` sont :

- **Autofocus:** Spécifie qu'une zone de texte doit automatiquement se concentrer lorsque la page est chargée
- **dirname:** Spécifie que la direction du texte de la zone de texte sera soumise
- **form:** Spécifie un ou plusieurs formulaires auxquels la zone de texte appartient
- **maxlength:** Spécifie le nombre maximum de caractères autorisés dans la zone de texte
- **placeholder:** Spécifie un indice court décrivant la valeur attendue d'une zone de texte.
- **required:** Spécifie qu'une zone de texte est requise / doit être remplie
- **wrap:** Spécifie comment le contenu d'une zone de texte doit être enveloppé lorsqu'il est soumis dans un formulaire

b. Les champs cachés

Il nous arrivera souvent de vouloir envoyer des données au serveur de manière transparente pour le visiteur. On utilisera alors un champ caché, qui ne sera pas affiché dans le navigateur du visiteur mais dont les données seront envoyées. Attention, cette balise apparaît tout de même dans le code source de la page.

Pour cela, on utilise la balise "`<input>`" avec "`hidden`" comme valeur de l'attribut "`type`".

Il est aussi nécessaire de renseigner l'attribut "`name`" pour retrouver ces données cachées du côté du serveur ainsi que l'attribut "`value`" qui va contenir ces données.

Exemple de champ caché

```
<input type="hidden" name="clef" value="a182f7d8e844d956a65b18e84f">
```

c. Les champs de fichiers

Pour permettre à l'internaute, d'envoyer des fichiers par l'intermédiaire du formulaire, que ce soit en pièce jointe d'un e-mail ou le chargement d'une image sur le serveur, on utilise la balise "**<input>**" dont l'attribut "**type**" est renseigné à "**file**".

Ceci fait, un champ avec un bouton "**Parcourir**" qui sera disponible. Mais pour que le champ soit opérationnel, il faut impérativement renseigner l'attribut "**enctype**" de la balise **<form>** à "**multipart/form-data**".

Il est aussi plus qu'utile de préciser l'attribut "**name**" du champ "**file**". On peut aussi renseigner l'attribut "**size**" qui affecte la taille du champ dans lequel sera écrit le nom du fichier.

Exemple de champ file

```
<input type="file" name="image">
```

4. Les boutons radio

Les boutons radio sont utilisés pour laisser aux visiteurs un choix et un seul parmi une liste de propositions. La balise est alors "**<input>**" dont l'attribut "**type**" est renseigné à "**radio**".

Ces boutons radio vont par groupe, c'est-à-dire qu'ils doivent avoir le même nom pour un groupe de propositions. Le nom est renseigné avec l'attribut "**name**".

La valeur de l'attribut "**value**" va être transmise au serveur en fonction du bouton choisi. On peut aussi forcer un bouton radio d'un groupe à être coché au chargement de la page en lui mettant l'attribut "**checked**" dont la seule valeur est "**checked**".

Exemple 1 :

```
<p>
  <input type="radio" name="civilite" value="mlle"> Mademoiselle
  <input type="radio" name="civilite" value="mme"> Madame
  <input type="radio" name="civilite" value="mr"> Monsieur
</p>
```

☒ Mademoiselle
☐ Madame
☐ Monsieur

Exemple 2 :

```
<p>
  <input type="radio" name="genre" value="homme" checked="checked"> Homme
  <input type="radio" name="genre" value="femme"> Femme
</p>
```

```
<input type="radio" name="genre" value="et"> Alien  
<input type="radio" name="genre" value="indéfini"> Indéfini  
</p>
```

Si on donne un nom différent aux boutons d'un groupe, le navigateur ne sait pas qu'ils appartiennent au même groupe, donc plusieurs choix seront possibles et on perd toute l'utilité des boutons radio.

5. Les cases à cocher

Les cases à cocher sont sensiblement identiques aux boutons radio, mais permettent un choix multiple pour un groupe de propositions. Les attributs sont identiques, à la différence que l'attribut **"type"** prend pour valeur **"checkbox"**.

Un problème peut arriver, lorsque vous désignez un groupe de cases à cocher par le même nom, seule la dernière sera envoyée au serveur. Pour remédier à ceci, il suffit de rajouter des crochets ouvrant et fermant au nom du groupe. Il indiquera au serveur qu'il s'agit d'un tableau de valeurs à récupérer et non une seule valeur (comme dans le cas des boutons radio). En revanche, dans le cas d'utilisation d'une case à cocher unique (sans groupe), les crochets sont inutiles.

Exemple de cases à cocher

```
<input type="checkbox" name="qualite[]" value="intelligent"> Intelligent  
<input type="checkbox" name="qualite[]" value="beau"> Beau  
<input type="checkbox" name="qualite[]" value="serviable"> Serviable  
<input type="checkbox" name="qualite[]" value="fort"> Fort  
<input type="checkbox" name="qualite[]" value="généreux"> Généreux
```

☐ Intelligent ☒ Beau ☐ Serviable ☐ Fort ☒ Généreux

6. Les boutons

Il existe deux balises différentes pour présenter les boutons : la balise **"<input>"** et la balise **"<button>"**. La différence entre les deux est que la balise **"<button>"** est double, donc qu'elle a une balise ouvrante et une balise fermante alors que la balise **"<input>"** est simple.

Il y a 4 types différents de boutons :

- Le bouton simple
- Le bouton d'envoi
- Le bouton image
- Le bouton effacer

a. *Le bouton simple*

Il est défini par l'attribut **"type"** des balises **"<input>"** et **"<button>"** ayant pour valeur **"button"**.

Ce bouton n'est utile que lors d'appel de scripts **JavaScript** au moment du clic. Il n'a aucune autre fonction particulière. Exemples de boutons simples :

Exemples de boutons simples

```
<input type="button" name="addition" value="Additionner" onclick="addition();">  
<button type="button" name=" addition " onclick=" addition ();">Additionner </button>
```

b. *Le bouton d'envoi*

Le bouton d'envoi sert, comme son nom l'indique, à envoyer les données du formulaire au serveur. C'est celui qui va permettre de valider le formulaire.

Il est caractérisé par la valeur **"submit"** de l'attribut type des balises **<input>** et **<button>**.

Exemples de boutons d'envoi

```
<input type="submit" name="envoyer" value="Valider">  
<button type="submit" name="soumettre"> Soumettre <br> les réponses </button>
```

c. *Le bouton image*

Le bouton image se comporte exactement comme un bouton d'envoi, c'est-à-dire qu'il sert à la validation du formulaire. Il sert à personnaliser un bouton.

Il est défini par l'attribut **"type"** de la balise **"<input>"** ayant pour valeur **"image"** et l'on renseigne l'adresse de l'image avec l'attribut **"src"**. C'est le seul bouton qui ne peut pas être géré par la balise **"<button>"**.

Exemple de bouton image

```
<input type="image" name="envoyer" src="adresse/de/l/image">
```

d. *Le bouton effacer*

Ce bouton sert à réinitialiser toutes les valeurs du formulaire. Il est caractérisé par la valeur **"reset"** de l'attribut **"type"** des balises **<input>** et **<button>**.

Exemples de boutons effacer

```
<input type="reset" name="effacer" value="Effacer toutes les valeurs">  
<button type="reset" name="effacer"> Réinitialiser </button>
```

7. Les listes déroulantes

Les listes permettent de laisser un choix aux visiteurs, au même titre que les boutons radio ou les cases à cocher, mais sont plus pratiques lorsqu'il s'agit de proposer un grand nombre de choix.

On distingue deux types de listes, les listes normales et les listes déroulantes.

Les deux sont introduites par la balise "**<select>**" qui comprend les différents choix sous forme de balise "**<option>**" dont on renseigne la valeur qui sera envoyée au serveur grâce à son attribut "**value**".

On peut même suggérer une proposition par défaut en précisant l'attribut "**selected**" de la balise "**<option>**" de notre choix. Par défaut, une liste est déroulante.

On doit renseigner le nom de la liste avec l'attribut "**name**".

Exemple de liste

```
<select name="couleur_yeux">
  <option value="bleus">Bleus</option>
  <option value="bruns" selected="selected">Bruns</option>
  <option value="verts">Verts</option>
  <option value="noisettes">Noisettes</option>
  <option value="gris">Gris</option>
  <option value="vairons">Vairons</option>
</select>
```



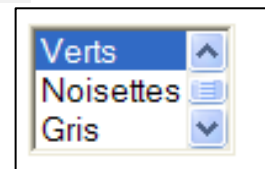
Cette liste est donc une liste déroulante avec un seul choix possible parmi les six proposés.

On peut rendre cette liste non déroulante en rajoutant l'attribut "**size**" qui prend pour valeur le nom de choix à afficher, par exemple "3".

Ce qui donne :

Exemple de liste

```
<select name="couleur_yeux" size="3">
  <option value="bleus">Bleus</option>
  <option value="bruns">Bruns</option>
  <option value="verts" selected="selected"> Verts</option>
  <option value="noisettes">Noisettes</option>
  <option value="gris">Gris</option>
  <option value="vairons">Vairons</option>
</select>
```



Par défaut, un seul choix est possible. Pour pouvoir sélectionner plusieurs choix, comme dans le cas de cases à cocher, il suffit de rajouter l'attribut "**multiple**" qui prend pour

seule valeur "**multiple**".

Exemple de liste à choix multiple

```
<select name="matieres_pereeres" size="5" multiple="multiple">  
  <option value="maths">Maths</option>  
  <option value="français">Français</option>  
  <option value="anglais">Anglais</option>  
  <option value="histoire">Histoire</option>  
  <option value="sport">Sport</option>  
  <option value="dessin">Dessin</option>  
  <option value="musique">Musique</option>  
  <option value="physique">Physique</option>  
  <option value="sciences naturelles">Sciences naturelles</option>  
</select>
```



Le choix multiple ne peut pas se faire sur une liste déroulante. L'attribut size doit être renseigné pour que le multiple fonctionne.

Conclusion

Ce chapitre a comme but de connaître des éléments avancés qu'on peut trouver dans un document html, ainsi les propriétés utiles, cette liste des balises et propriétés reste toujours non exhaustive. Le langage HTML continu à évoluer depuis sa création jusqu'à nos jours.