

# HTML



Le langage de balisage hypertexte, ou HTML, est l'épine dorsale du Web. Il fournit le contenu et la structure de ce contenu qui s'affiche dans votre navigateur Web.

Le HTML (HyperText Markup Language) est le langage de balisage standard pour structurer les documents sur le Web. Les documents HTML forment une arborescence de nœuds, avec des éléments HTML et des nœuds de texte. Les éléments HTML, qui incluent des balises, attributs, et contenu, sont utilisés pour la sémantique et la mise en forme, comme créer des paragraphes, des listes, des tableaux, etc...

# HTML



- HTML - Hyper Text Markup Language
- `<élément>contenu</élément>`
- `<h1>titre</h1>` === **En-tête**
- `<p> texte </p>` === **Paragraphe**
- `` == **Image**
- `<br/>` == **Saut de ligne**

# HTML



- Google Chrome - Navigateur web
- *Visual Studio Code - Éditeur de texte*

# Structure basique d'une page HTML



```
<!DOCTYPE html>    ===  Version HTML
<html>              ===  Élément racine
  <head>             ===  Informations à propos de la page (métadonnées, titre, ressources extérieures, etc...)
    <meta charset="UTF-8" />    ==== Encodage des caractères
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />  ==== Taille et échelle
    <title>Document</title>    ===  Titre de la page
  </head>
  <body>             ===  Ce qu'on affiche sur la page
    Contenu de la page
  </body>
</html>
```

# Titres



Il y a six éléments de titre de section: `<h1>`, `<h2>`, `<h3>`, `<h4>`, `<h5>` et `<h6>`. Chacun représente l'un des six niveaux d'en-tête de section, `<h1>` étant le niveau le plus élevé ou le plus important, et `<h6>` le niveau le plus bas.



```
<h1>Bonjour, je suis un titre de niveau un</h1>  
<h2>Bonjour, je suis un titre de niveau deux</h2>  
<h3>Bonjour, je suis un titre de niveau trois</h3>  
<h4>Bonjour, je suis un titre de niveau quatre</h4>  
<h5>Bonjour, je suis un titre de niveau cinq</h5>  
<h6>Bonjour, je suis un titre de niveau six</h6>
```

# Les bases du texte



- Les paragraphes
- Les entités HTML



```
<p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Nesciunt, error?</p>  
<p>&copy;</p> <!-- © -->
```



## Mise en valeur du texte

Plusieurs éléments peuvent être utilisés pour accentuer du texte en fonction de la raison sémantique de cette mise en forme (plutôt que pour des raisons de présentation, car il s'agit d'un travail pour CSS).

- L'élément **<em>** met en avant une partie du contenu. Les navigateurs affichent généralement le contenu en italique.
- L'élément **<strong>** identifie le texte comme ayant une grande importance. Les navigateurs affichent généralement le contenu avec une épaisseur de police plus importante.



# Les images

Les images décoratives, telles que les dégradés d'arrière-plan sur des boutons ou les images de fond sur des sections de contenu ou sur la page entière, sont de type présentation et doivent être appliquées avec CSS. Lorsqu'une image ajoute du contexte à un document, elle est considérée comme du contenu et doit être intégrée avec du code HTML.

La principale méthode pour inclure des images consiste à utiliser la balise `<img>` avec l'attribut `src` qui fait référence à une ressource image et l'attribut `alt` décrivant l'image.



```

```



# Les liens



Bien que cela ne soit pas obligatoire, l'attribut `href` est présent dans presque toutes les balises `<a>`. L'adresse du lien hypertexte transforme l'élément `<a>` en lien. L'attribut href permet de créer des liens hypertextes vers des emplacements sur la page active, sur d'autres pages d'un site ou vers d'autres sites. Il peut également être codé pour télécharger des fichiers ou pour envoyer un e-mail à une adresse spécifique. Il peut même inclure un objet et des suggestions de contenu du corps de l'e-mail.



```
<a href="https://dwwm-html-course.netlify.app/">Cours HTML</a>  
<a href="#teachers">Nos professeurs</a>  
<a href="mailto:thomas_3004@hotmail.fr">Email Thomas</a>  
<a href="tel:0669449688">Appeler Thomas</a>
```

# Les listes



Le HTML nous offre plusieurs façons différentes de baliser des listes. Il existe des listes ordonnées (<ol>), des listes non ordonnées (<ul>) et d'autres. Les éléments de liste (<li>) sont imbriqués dans des listes ordonnées et des listes non ordonnées.

Dans les listes non ordonnées, les éléments de la liste sont généralement affichés sous forme de puces. Dans les listes ordonnées, elles sont généralement précédées d'un compteur croissant tel qu'un chiffre ou une lettre. Les puces et l'ordre de numérotation peuvent être contrôlés ou inversés à l'aide du code HTML ou CSS, ou des deux.

```

<!-- Liste non ordonnée -->
<ul>
  <li>Front-end</li>
  <li>Back-end</li>
  <li>Fullstack</li>
</ul>
```

```

<!-- Liste ordonnée -->
<ol>
  <li>HTML</li>
  <li>CSS</li>
  <li>Javascript</li>
</ol>
```

# Les tableaux



Les tableaux HTML sont utilisés pour afficher des données tabulaires avec des lignes et des colonnes. La décision d'utiliser un `<table>` doit être basée sur le contenu que vous présentez et sur les besoins de vos utilisateurs par rapport à ce contenu. Si des données sont présentées, comparées, triées, calculées ou croisées, `<table>` est probablement le bon choix. Si vous souhaitez simplement disposer soigneusement le contenu non tabulaire, par exemple un grand groupe de vignettes, les tableaux ne sont pas appropriés: vous devez créer une liste d'images et appliquer un style à la grille avec CSS.

## Légende du tableau

En tant qu'élément sémantique natif, `<caption>` est la méthode privilégiée pour attribuer un nom à une table. `<caption>` fournit un titre descriptif de table associé au programme. Par défaut, elle est visible et disponible pour tous les utilisateurs.

## Section des données

Le contenu des tableaux comprend jusqu'à trois sections: aucun ou plusieurs en-têtes de tableau (`<thead>`), corps de tableau (`<tbody>`) et pieds de page de tableau (`<tfoot>`). Toutes ces sections sont facultatives et sont acceptées.

# Les tableaux



## Contenu du tableau

Les tableaux peuvent être divisés en en-têtes, corps et pieds de page, mais aucune de ces actions ne change rien si les tableaux ne contiennent pas de lignes, de cellules ni de contenu. Chaque ligne du tableau `<tr>` contient une ou plusieurs cellules. Si une cellule est une cellule d'en-tête, utilisez `<th>`. Sinon, utilisez `<td>`.

# Les tableaux



```
<table>
  <caption>
    Les étudiants de la formation DWM
  </caption>
  <thead>
    <tr>
      <th>ID</th>
      <th>Nom</th>
      <th>Prénom</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th>1</th>
      <td>John</td>
      <td>Doe</td>
    </tr>
    <tr>
      <th>2</th>
      <td>Peter</td>
      <td>Smith</td>
    </tr>
    <tr>
      <th>3</th>
      <td>Susan</td>
      <td>Bennett</td>
    </tr>
  </tbody>
</table>
```

# Les formulaires



Un formulaire sert à collecter des données auprès de l'utilisateur.

```
<form>
  <label for="animal">Quel est votre animal préféré ?</label>
  <input type="text" id="animal" name="animal">
  <button>Envoyer</button>
</form>
```

Lors de l'envoi d'un formulaire (par exemple, lorsque l'utilisateur clique sur le bouton **Envoyer**), le navigateur envoie une requête. Un script peut répondre à cette requête et traiter les données.

Par défaut, la demande est envoyée à la page sur laquelle le formulaire est affiché.



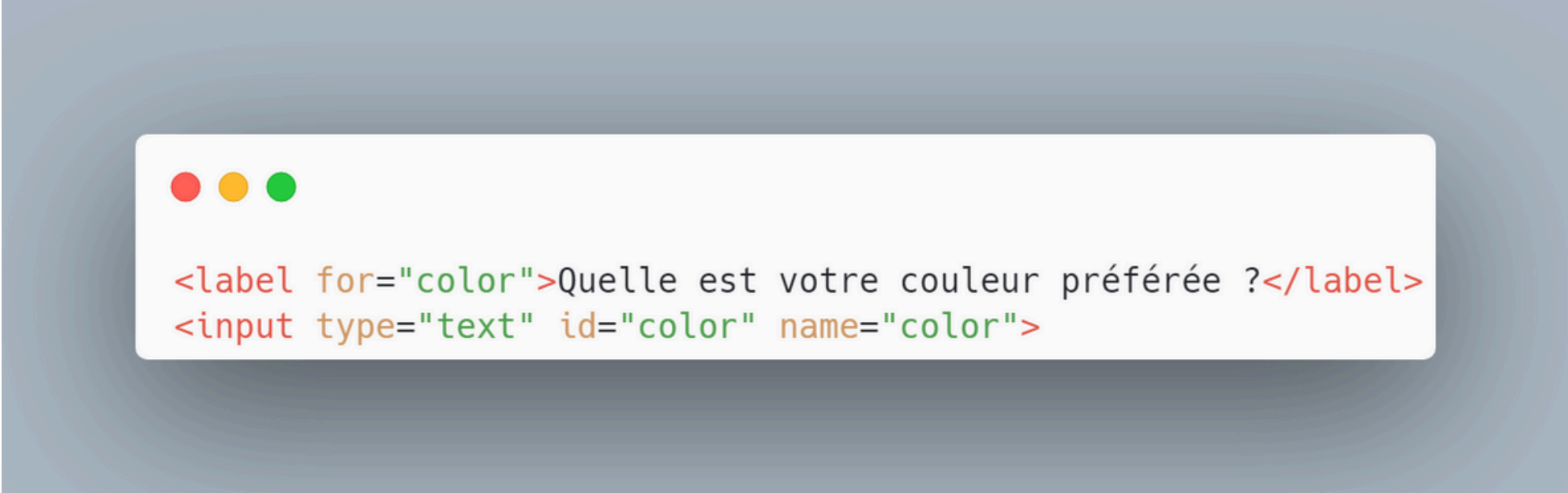
# Les formulaires

Pour rendre un formulaire interactif, vous devez y ajouter des éléments. Il existe des commandes pour saisir et sélectionner des données, des éléments qui décrivent des commandes, des éléments qui regroupent des champs et des boutons permettant d'envoyer un formulaire.

## Que sont les éléments d'un formulaire ?

- **Libellés des éléments du formulaire**

Pour décrire les commandes de formulaire, utilisez un `<label>` pour chaque commande de formulaire.

A code editor window with a light blue background and a white rounded rectangle containing the code. The code is color-coded: red for tags, green for attribute names, and orange for attribute values. The code defines a label for a color input field.

```
<label for="color">Quelle est votre couleur préférée ?</label>
<input type="text" id="color" name="color">
```

L'attribut **for** de l'élément de libellé correspond à l'attribut **id** de l'entrée.

# Les formulaires

- Capturer les entrées utilisateur

Comme son nom l'indique, l'élément **<input>** permet de recueillir les entrées des utilisateurs.



```
<label for="color">Quelle est votre couleur préférée ?</label>  
<input type="text" id="color" name="color">
```

Comme indiqué précédemment, l'attribut **id** connecte **<input>** à **<label>**. Qu'en est-il des attributs de nom et de type dans cet exemple ?



# Les formulaires



## Attribut name

Utilisez l'attribut **name** pour identifier les données saisies par l'utilisateur à l'aide de la commande. Si vous envoyez le formulaire, ce nom sera inclus dans la demande. Supposons que vous ayez nommé une commande de formulaire **color** et que l'utilisateur ait saisi **bleue**. Ces informations sont incluses dans la requête sous la forme **color=bleue**.

## Type d'entrée

Il existe différents types de commandes de formulaire, qui disposent toutes de fonctionnalités intégrées utiles qui fonctionnent sur différents navigateurs et plates-formes. En fonction de l'attribut type, le navigateur affiche différentes interfaces utilisateur, affiche différents claviers à l'écran, utilise différentes règles de validation, etc.

# Les formulaires

- Autoriser plusieurs lignes de texte

Imaginons que vous ayez besoin d'un champ dans lequel l'utilisateur peut rédiger un commentaire. Ne serait-il pas formidable de pouvoir saisir plusieurs lignes de texte ? Il s'agit de l'objectif de l'élément **<textarea>**.



```
<label for="comment">Commentaire</label>  
<textarea id="comment" name="comment"></textarea>
```

# Les formulaires

- Sélectionner dans une liste d'options

Comment proposer aux utilisateurs une liste d'options parmi lesquelles choisir ? Pour ce faire, vous pouvez utiliser un élément **<select>**.



```
<label for="color">Couleur</label>
<select id="color" name="color">
  <option value="orange">Orange</option>
  <option value="pink">Rose</option>
</select>
```



# Les formulaires

- **Regroupement des commandes du formulaire**

Vous devez parfois regrouper des commandes de formulaire. Pour ce faire, vous pouvez utiliser l'élément **<fieldset>**.

Chaque élément **<fieldset>** nécessite un élément **<legend>**, tout comme chaque commande de formulaire a besoin d'un élément **<label>** associé. **<legend>** doit également être le tout premier élément de la **<fieldset>**. Après l'élément **<legend>**, vous pouvez définir les commandes de formulaire qui doivent faire partie du groupe.

- **Case d'option**

Lorsque vous sélectionnez une case d'option dans un groupe de cases d'option, vous ne pouvez en sélectionner qu'une seule à la fois, en raison de l'attribut **name**. Cet effet est créé en attribuant la même case d'option **name** à chaque case d'option d'un groupe.

# Les formulaires



- **Cases à cocher**

Toutes les cases à cocher d'un groupe peuvent avoir le même **name**. Seules les cases **name** et value sont envoyées avec le formulaire.

```
<fieldset>
  <legend>Suppléments :</legend>
  <input
    type="checkbox"
    name="supplements"
    id="guacamole"
    value="guacamole"
  />
  <label for="guacamole">Guacamole</label>

  <input
    type="checkbox"
    name="supplements"
    id="chorizo"
    value="chorizo"
  />
  <label for="chorizo">Chorizo</label>

  <input
    type="checkbox"
    name="supplements"
    id="cheddar"
    value="cheddar"
  />
  <label for="cheddar">Cheddar</label>
</fieldset>
```

# Les formulaires



```
<fieldset>
  <legend>Quelle est votre technologie web préférée ?</legend>

  <label for="html">HTML</label>
  <input type="radio" name="webfeature" value="html" id="html">

  <label for="css">CSS</label>
  <input type="radio" name="webfeature" value="css" id="css">
</fieldset>
```

Un **name** doit être unique au groupe: si vous utilisez accidentellement le même **name** pour deux groupes distincts, la sélection d'une case d'option dans le deuxième groupe désélectionne toute sélection effectuée dans le premier groupe avec le même **name**.

Le **name** et le **value** de la case d'option sélectionnée sont envoyés avec le formulaire. Assurez-vous que chaque case d'option est associée à un **value** pertinent (et généralement unique). Les valeurs des cases d'option non sélectionnées ne sont pas envoyées.

# Les formulaires

- **Envoi du formulaire**

Après avoir appris à ajouter des commandes de formulaire et à les regrouper, vous vous demandez peut-être comment un utilisateur peut envoyer un formulaire.

La première option consiste à utiliser un élément **<button>**.



Lorsqu'un utilisateur clique sur le bouton **Envoyer**, le navigateur envoie une requête à l'URL spécifiée dans l'attribut d'action de l'élément **<form>** avec toutes les données des commandes de formulaire.



# Les formulaires

Après avoir appris à ajouter des commandes de formulaire, vous vous demandez peut-être comment un utilisateur peut envoyer un formulaire.

Vous pouvez également utiliser un élément **<input>** avec **type="submit"** au lieu d'un élément **<button>**. L'entrée a l'apparence et se comporte comme une **<button>**. Au lieu d'utiliser un élément **<label>** pour décrire **<input>**, utilisez plutôt l'attribut **value**.



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



# Sémantique



En programmation, la sémantique fait référence au sens d'une partie de code — par exemple "quel est le rôle ou le but de cet élément HTML" (plutôt que "à quoi ressemble-t-il ?").

## Ressources :

- [HTML Semantic Elements](#)

# Accessibilié



L'accessibilité numérique, communément abrégée "a11y", consiste à concevoir et à créer des sites Web et des applications Web avec lesquels les personnes handicapées peuvent interagir de manière significative et équivalente.

## Ressouces :

- [A11Y Style Guide](#)