

# Leçon 1 : Introduction et spécifications du HTML5

*Cours\_ UVCI  
PRW2103-2*

DIABATE Nagbegna

# Table des matières



<b>I - Objectifs</b>	<b>3</b>
<b>II - Section 1 - Pourquoi le HTML 5</b>	<b>4</b>
1. 1. Vous avez dit HTML 5 ? .....	4
2. 2. Qu'est ce qui change avec HTML 5 ? .....	4
3. 3. La notion de sémantique avec le HTML 5 .....	5
4. Exercice : Avez-vous compris cette section ? .....	9
<b>III - Section 2 - Les balises introduites ou modifiées par HTML5</b>	<b>10</b>
1. 1. La balise <head> simplifiée avec HTML5 .....	10
2. 2. Les images avec HTML5 .....	11
3. 3. Balise <audio> HTML5 .....	11
4. 4. La balise <video> .....	12
5. 5. de nouveau attributs et éléments .....	12
6. Exercice : Testez votre compréhension... ..	13
<b>IV - Section 3 - La validation des données avec HTML5</b>	<b>15</b>
1. 1. Le placeholder, une indication .....	15
2. 2. Les champs obligatoires avec l'attribut required .....	16
3. 3. Les masques et la validation des données .....	16
4. Exercice : Entraînez-vous avant le devoir ! .....	19



# *Objectifs*

À la fin de cette leçon, vous serez capable de :

- *Reconnaître* les balises et le code HTML 5
- *Exploiter* les nouvelles fonctionnalités et balise de HTML5

# Section 1 - Pourquoi le HTML 5



## 1. 1. Vous avez dit HTML 5 ?

HTML5 est la dernière évolution des standards qui définissent HTML. Le terme HTML5 regroupe deux concepts différents :

Il s'agit de la nouvelle version du langage HTML, avec de nouveaux éléments, attributs et comportements ; mais aussi un ensemble plus large de technologies qui permettent des sites web plus variés et puissants, et des applications web. Cet ensemble est parfois appelé *HTML5 & Cie* et souvent juste abrégé en *HTML5*.

Ainsi, il apporte non seulement l'utilisation de média riches (vidéo, audio, SVG, etc.) mais aussi des fonctionnalités permettant le développement d'applications web bien plus attractives et interactives.

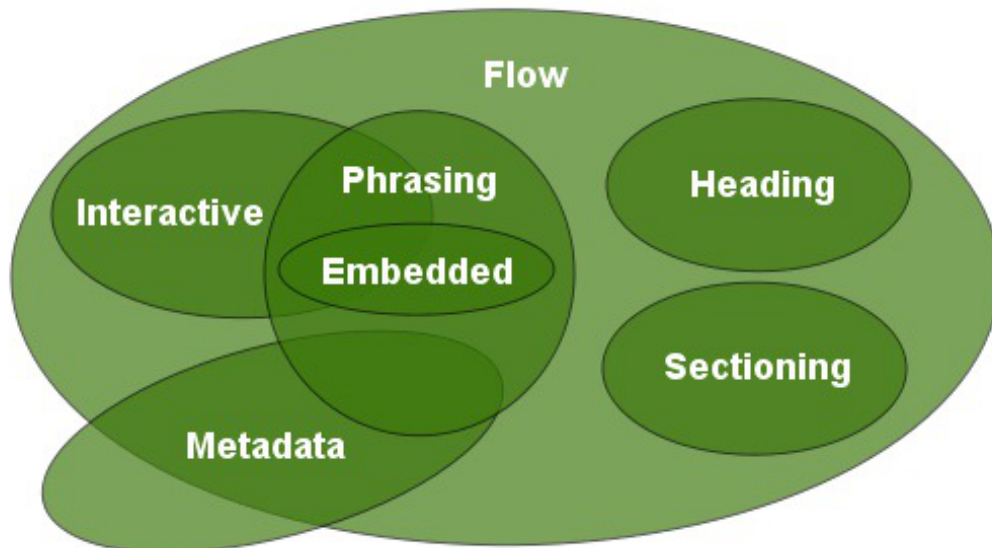
Étant donné que HTML5 est encore en cours d'évolution, certains navigateurs ne supportent pas encore toutes les fonctionnalités offertes par ce nouveau standard.



## 2. 2. Qu'est ce qui change avec HTML 5 ?

C'est tout d'abord toute la structure et l'organisation du code qui vont être revus avec l'avènement du HTML5. Avec cette nouvelle version du langage, le document ne sera plus seulement basé sur les blocs et les lignes. Les concepteurs ont organisé le document avec un agencement par grandes catégories dans lequel on retrouverait les éléments suivants :

- *Metadata content* : On retrouvera dans ce grand thème toutes les informations relatives aux balises *meta* que l'on retrouve toujours dans le *header* d'un document HTML
- *Flow content* : Cette catégorie regroupe tous les éléments qu'on retrouve entre les balises `< body >` et `< /body >`
- *Sectioning content* : Cette catégorie va permettre de définir les différentes sections visibles de la page (comme le *footer*, par exemple)
- *Heading content* : Toutes les informations qu'on retrouve dans le *header* seront désormais stockées dans cette section du document.
- *Phrasing content* : Cette catégorie rassemble tous les éléments permettant de mettre en forme le texte du document *html*.
- *Embedded content* : va permettre d'importer une ressource dans le document ou d'appeler un élément développé dans un langage différent de celui de la page.
- *Interactive content* : On retrouve dans cette catégorie tous les éléments qui permettront l'interaction entre la page et l'utilisateur.



Source : <https://www.alsacreations.com/xmedia/doc/original/contenthtml5.png>

Ainsi, au vu de cette nouvelle organisation, le code suivant est parfaitement valide car l'élément `<a>` peut entourer des paragraphes, des listes, des tableaux voire des sections entières :

```
1 <aside>
2 Welcome!
3 <a href="about.html">
4 This is home of...
5 <h1>The Falcons!</h1>
6 The Lockheed Martin multirole jet fighter aircraft!
7 </a>
8 This page discusses the F-16 Fighting Falcon's innermost secrets.
9 </aside>
```

En plus de cette nouvelle organisation des documents, HTML5 introduit surtout une nouvelle couche d'*API* qui seront proposées dans cette première version du HTML5, parmi lesquelles :

- Une API de *dessin 2D*, grâce à la nouvelle balise *canvas*
- Une API pour jouer des *vidéos* et des *sons/musiques* permis grâce aux balises *video* et *audio*;
- Une API utilisée pour les applications *hors-ligne*;
- Une API d'*édition* en combinaison grâce à l'attribut *contenteditable*
- Une API de « *glisser-déposer* » en combinaison avec l'attribut *draggable*;
- Une API qui permet l'accès à l'historique et qui donne la possibilité aux pages d'en ajouter pour prévenir les *problèmes de bouton retour-en-arrière*.
- Une API de *géo-localisation*
- Une API permettant d'analyser de reproduire une page HTML grâce à la balise *inner-HTML*

L'arrivée du HTML5 sonne aussi le glas pour une série d'attributs dont la présence étant encore acceptée dans les documents en HTML4 et laisse place à une série de nouveaux attributs et de nouvelles balises.

*Ressources :*

- *Comment ça marche.net*
- *Alsacreations*

### 3. 3. La notion de sémantique avec le HTML 5

Le HTML4 offrait la balise `<div>` qui a été largement (trop) utilisée pour structurer le document HTML. Cela présente un inconvénient majeur : *Un code trop souvent difficile à lire, où s'y retrouver parmi toutes les balises <div> est vraiment laborieux!*

Près de 20 nouveaux éléments HTML5 comme `<article>`, `<footer>`, `<nav>` ... ont été créés pour être utilisés dans la structuration d'un document, y compris les titres, formulaire, tableau, ...

Voici les balises HTML5 les plus populaires qui peuvent être utilisés dans la hiérarchisation d'un document en remplacement de la balise `<div>`. Cependant, elles n'excluent pas l'usage des `<div>` dans les documents HTML 5.

### a. Les éléments `<article>` et `<section>`

L'élément `<article>` est un fragment indépendant du contenu général. *Billet de blog, nouvelle article ou autres types de contenu du texte*. Fondamentalement, vous pouvez utiliser cet élément pour le balisage d'un composant destiné à être largement utilisé et distribué.

L'élément `<section>` est assez trompeur car il est largement trop employé par les développeurs Web comme une alternative à `<div>`. Vous devriez savoir que cette balise est étroitement liée à la balise `<article>` et qu'elle est utilisée pour regrouper un contenu qui diffère d'un autre groupement de contenu sur la page. Généralement, les groupes sont fait par thèmes ou sujet identique.

```

1 <article>
2   <p>
3     <a href=
4       "http://www.alsacreations.com/actu/lire/746-xhtml-est-mort-vive-html.html">
        XHTML est mort, vive HTML !</a>
5   <br />
6   Sous ce titre quelque peu provocateur (et faux) se cache une réalité
      officielle :
7   le W3C vient d'annoncer que ses travaux sur XHTML 2 se termineront en 2009.
8 </p>
9 </article>
10
```

### b. Les éléments `<header>` et `<footer>`

L'élément `<header>` a été créé pour une présentation plus sémantique des outils de navigation et des données importantes placées dans l'*en-tête de la page Web*.

Vous pouvez utiliser cet élément autant de fois que vous le souhaitez et aussi lui ajouter quelques balises supplémentaires comme `<nav>` pour les éléments du menu de navigation, `<h1,2,3...6>` pour les titres ...

La balise `<footer>` est similaire à `<header>`, elle est utilisée pour créer la structure de pied de page de votre document web. Vous pouvez également utiliser cet élément plusieurs fois sur une seule page pour des blocs différents. Cette balise peut être utilisée, par exemple, pour marquer le droit d'auteur, Conditions d'utilisation et autres, mais aussi pour le marquage de certaines informations sur l'auteur de l'article ...

### c. La balise `<nav>`

L'élément `<nav>` est utilisé pour créer des menus avec des liens vous permettant de naviguer sur les pages du site Web.

Vous pouvez avoir, par exemple, un bloc avec des liens sponsorisés, un autre pour les différentes catégories ...

Aussi, l'élément `<nav>` peut également être utilisé pour plusieurs fois. Bien souvent cette balise est associée à une liste, ordonnée ou non, de liens : `<ul>` et `<ol>` pour créer des menus.

```

1 <nav>
2   <ul>
3     <li><a href="index.php">Page d'accueil</a></li>
4     <li><a href="contact.php">Contact</a></li>
5   </ul>
6 </nav>
```

#### d. Les éléments `<figure>` et `<figcaption>`

Selon les références du *W3C*, les éléments `<figure>` et `<figcaption>` sont utilisés pour présenter un *bloc de contenu avec une légende*, qui est généralement référencé comme une seule unité à partir du flux principal du document. En d'autres termes, vous pouvez l'utiliser pour marquer divers types de supports de contenu comme des illustrations, des photos, des exemples de code et des diagrammes.

- La balise `<figure>` définit un contenu autonome, comme des illustrations, des diagrammes, des photos, des listes de codes, etc
- La balise `<figcaption>` définit une légende pour un élément `<figure>`.

```
1
2 <figure>
3   
4   <figcaption>Un petit chat mignon tout plein</figcaption>
5 </figure>
6
```

#### e. La balise `<aside>`

L'élément `<aside>` est utilisé pour un contenu secondaire qui n'est emboîté dans aucun élément du document (`<article>` ...).

L'exemple le plus approprié pour utiliser l'élément `<aside>` est la fameuse *Sidebar* ou *colonne latéral*. Nous pouvons utiliser cet élément avec l'élément `<nav>`, les bannières publicitaires ou tout simplement pour le contenu qui doit être placé séparément du contenu principal.

```
1 <aside>
2   <h1>Archives</h1>
3   <ul>
4     <li><a href="/archives/09/05/">Mai 2009</a></li>
5     <li><a href="/archives /09/06/">Juin 2009</a></li>
6     <li><a href="/archives /09/07/">Juillet 2009</a></li>
7   </ul>
8 </aside>
9
```

#### f. Le doctype

Le *doctype* est simplifié :

```
1 <!DOCTYPE html>
2
```

Il n'est pas sensible à la casse (on peut écrire `<!doctype html>` par exemple), et son rôle est uniquement d'éviter un rendu en mode de compatibilité pour les anciens navigateurs.

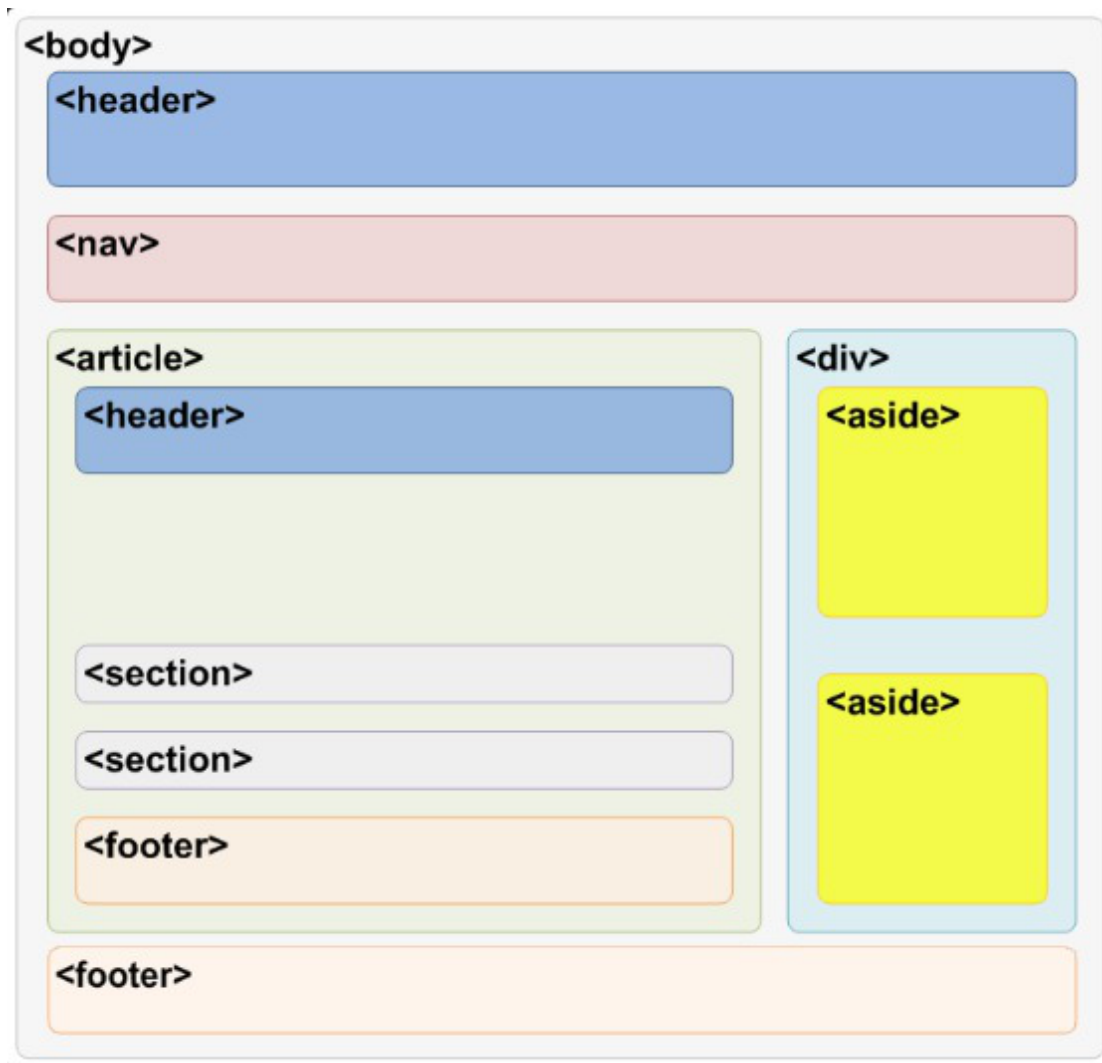


#### Complément : Ressources :

- *HTML5 : Améliorer votre sémantique*

#### Anatomie d'une page html5 de base

Ainsi, de manière générale, une page HTML 5 peut se présenter de la sorte :





#### 4. Exercice : Avez-vous compris cette section ?

## Exercice

1) Quel est le doctype d'un document HTML5 ?

- ☐ <!DOCTYPE html5>
- ☐ <!DOCTYPE html>
- ☐ <!DOCTYPE html PUBLIC "-//W3C//DTD HTML5.0 Strict//EN">

## Exercice

2) Quelle nouvelle balise de section permet de regrouper un contenu tangentiel au contenu principal du document ?

- ☐ `<section id="sidebar">`
- ☐ `<sidebar>`
- ☐ `<aside>`
- ☐ `<details>`

## Exercice

3) Dans quel élément HTML 5 est-il idoine de placer le menu de navigation de ma page ?

- ☐ `<nav>`
- ☐ `<navigate>`
- ☐ `<section>`
- ☐ `<aside>`

## Exercice

4) Lorsque j'ai une vidéo sur Youtube et que je veux l'importer sur la page Web, dans quel élément HTML5 doit-il être placé ?

- ☐ Phrasing content
- ☐ Embedded content
- ☐ Interactive content
- ☐ Sectionning content

## Exercise

5) Lorsque'on commence a utiliser des balise HTML5, de quoi devons-nous nous préoccuper le plus ?

- ☐ La version de l'éditeur de texte que nous utilisons
- ☐ La compatibilité des balises avec les navigateurs
- ☐ La compatibilité des balises avec le système d'exploitation des visiteurs

# Section 2 - Les balises introduites ou modifiées par HTML5



## 1. 1. La balise <head> simplifiée avec HTML5

L'en-tête de document a été simplifié avec HTML 5 gardant les mêmes déclarations mais en simplifiant l'écriture des déclarations.

Voilà par exemple une déclaration simple de l'en-tête d'une page Web avec HTML 5

```

1 <!DOCTYPE html>
2 <html lang="fr">
3 <head>
4
5 <meta charset="UTF-8">
6 <title>Titre de la page</title>
7 <meta name="description" content="Description de la page" >
8 <meta name="viewport" content=
  "width=device-width,initial-scale=1,maximum-scale=1.0">
9
10 <link rel="stylesheet" href="style.css">
11 <link rel="shortcut icon" href="favicon.ico">
12
13 <script src="script.js"></script>
14
15 </head>

```

A partir de là, nous pouvons merquer que :

- Le *doctype* a été *simplifié* et devient plus simple à écrire `<!DOCTYPE html>`
- La langue est définie de façon très simple dans la balise `<html>` : `<html lang="fr">`
- La définition du jeu de caractère a été également simplifiée, désormais on peut écrire simplement `<meta charset="UTF-8">`
- Plus besoin de spécifier le type de script, il suffit juste de pointer le fichier externe de script pour que ça marche : `<script src="script.js"></script>`
- Le lien vers le fichier de CSS est plus simplifié et facile à lire : `<link rel="stylesheet" href="style.css">`

Vous en apprendrez plus sur le ligne

```

1 <meta name="viewport" content=
  "width=device-width,initial-scale=1,maximum-scale=1.0">

```

dans la suite du cours. Cependant, les curieux peuvent déjà jeter un coup d'œil ici : <http://www.conseil-webmaster.com/formation/html5/balise-head-html5.php>

## 2. 2. Les images avec HTML5

Suivant la sémantique du HTML5, nous pouvons déduire l'utilisation de la balise `<figure>` pour améliorer non seulement le rendu de notre document mais aussi la bonne compréhension du contenu.

Cela est possible puisque la balise `<figure>` est en fait une unité de contenu, un encart pour isoler un bloc avec un contenu principalement constitué de un ou plusieurs éléments : multimédias, diagrammes, tableaux, illustrations, exemples, codes, images, vidéos, ou carrément du texte.

la balise `<figcaption>` permet d'afficher une légende ou une description du contenu de la figure.

```
1 <figure>
2 
3 <figcaption>Bootstrap Twitter</figcaption>
4 </figure>
```

Donne l'image suivante avec sa légende



Bien que vous puissiez utiliser la balise `<img>` toute seule ou encore l'entourer par la balise `<p>`, la balise `<figure>` a un rôle sémantique, si votre image apporte du sens à la page ou contribue à la bonne compréhension, vous utiliserez la balise `<figure>`. Ainsi, elle peut être utilisée pour ajouter des légendes à tout autre type de contenu multimédia.

## 3. 3. Balise `<audio>` HTML5

balise `<audio>` définit un flux ou streaming audio (musique ...). Voici une balise `<audio>` utilisée pour ajouter un son (mp3 ou ogg) dans une page web

```
1 <audio controls="controls">
2   <source src="chanson_1.ogg" type="audio/ogg" />
3   <source src="chanson_2.mp3" type="audio/mp3" />
4   Votre navigateur ne supporte pas la balise AUDIO.
5 </audio>
```

Le texte imbriqué entre les balises `<audio>` sera affiché par les navigateurs n'étant pas compatibles avec cet élément du HTML5.

Cette balise prend en compte plusieurs attributs :

- *controls*: pour ajouter les boutons « Lecture », « Pause » et la barre de défilement.
- *width*: pour modifier la 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.
- *preload*: indique si la musique peut être préchargée dès le chargement de la page ou non. Cet attribut peut prendre les valeurs :

*auto*(par défaut) : le navigateur décide s'il doit précharger toute la musique, uniquement les métadonnées ou rien du tout.

*metadata*: charge uniquement les métadonnées (durée, etc.).

*none*: pas de préchargement. Utile si vous ne voulez pas gaspiller de bande passante sur votre site.

### Les formats audio

Pour diffuser de la musique ou un son, il existe de nombreux formats. La plupart d'entre eux sont compressés ce qui permet de réduire leur poids :

- *MP3* : C'est l'un des plus vieux, mais aussi l'un des plus compatibles car tous les appareils savent lire des MP3, ce qui fait qu'il est toujours très utilisé aujourd'hui.
- *AAC* : utilisé majoritairement par Apple sur iTunes, c'est un format de bonne qualité. Les iPod, iPhone et autres iPad savent les lire sans problème.
- *OGG* : le format Ogg Vorbis est très répandu dans le monde du logiciel libre, notamment sous Linux. Ce format a l'avantage d'être libre, c'est-à-dire qu'il n'est protégé par aucun brevet.

## 4. 4. La balise <video>

L'élément `<video>`, cousin de `<audio>` offre en HTML5 une solution simple, native pour les navigateurs pour l'intégration d'une vidéo dans une page web. Elle permet également de proposer une alternative à l'utilisation de Flash pour les plate-formes ne le supportant pas (iOS par exemple avec iPhone, iPod, iPad...)

La syntaxe de base de la balise video est extrêmement simple :

```
1 <video controls src="video.ogv">Ici la description alternative</video>
2
```

L'attribut *src* définit bien entendu l'adresse du fichier vidéo. Si vous indiquez les dimensions avec les attributs *height* et *width*, c'est encore mieux, et si tout va bien, un élément devrait s'afficher dans le navigateur si celui-ci supporte le format de vidéo indiqué dans la source.

On peut également proposer plusieurs sources différentes dans plusieurs formats différents en indiquant les types MIME grâce à l'attribut *type* :

```
1 <video width="400" height="222" controls="controls">
2   <source src="video.mp4" type="video/mp4" />
3   <source src="video.webm" type="video/webm" />
4   <source src="video.ogv" type="video/ogg" />
5   Ici l'alternative à la vidéo : un lien de téléchargement, un message, etc.
6 </video>
7
```

Plusieurs formats tiennent le devant de la scène : *WebM*, *MP4* et *Ogg Theora* dont les différentes extensions sont données dans le code ci-dessus.

Le but est de proposer une solution d'intégration de la balise *video* compatible sur le plus de navigateurs possible.

### les attributs de <video>

- L'attribut *controls* donne accès aux contrôles de lecture (boutons de navigation, volume, etc, selon les possibilités du navigateur), ou les masque s'il est omis.
- L'attribut *preload="auto"* permet de spécifier au navigateur de débiter le téléchargement de la vidéo tout de suite, en anticipant sur le fait que l'utilisateur lira la vidéo.
- L'attribut *autoplay="true"* comme son nom l'indique, permet de lancer la lecture automatiquement. Cela peut également être problématique avec une connexion à faible bande passante ou sur un terminal mobile. De manière générale, évitez d'imposer vos choix à l'utilisateur... et à sa connexion internet.
- L'attribut *poster="image.jpg"* permet d'indiquer une image à afficher par défaut dans l'espace réservé par la vidéo, avant que la lecture de celle-ci ne soit lancée.
- L'attribut *loop* indique que la lecture doit s'effectuer en boucle.

## 5. 5. de nouveau attributs et éléments

le HTML5 a introduit de nouveau attribut parmi lesquels certains sont utilisé pour agir sur des bloc, des zones ou leurs contenu. Ainsi, nous avons :

- *Glisser-déposer* : L'attribut *draggable* permet de rendre un élément déplaçable.

- *Contenteditable* : Cet attribut indique qu'une zone est éditable. L'utilisateur peut en changer le contenu et manipuler le balisage.

- *Spellcheck* : Cet attribut active ou non la correction orthographique sur une zone de saisie.

Il existe d'autres nouveaux attributs HTML5 spécifiques à certains éléments uniquement (par exemple *ping* sur `<a>`, *charset* sur `<meta>`), ou bien applicables à tous. Par exemple *contextmenu*, *hidden*, *data-\**, etc.

Ressource :

Le site *Alsacreation* vous donne plus de détails sur ces attributs :  
<https://www.alsacreations.com/tuto/lire/1401-attributs-hidden-contenteditable-contextmenu-spellcheck>

plusieurs autres nouvelles balises ont été intrduites avec HTML5, la maitrise de certaines necessite la conneaissances d'autres langages comme Javascript ou JQuery :

`<command>`, `<datalist>`, `<details>`, `<dialog>`, `<embed>`, `<figcaption>`,  
`<figure>`, `<keygen>`, `<mark>`, `<meter>`, `<output>`, `<progress>`, `<rp>`, `<rt>`,  
`<source>`, `<summary>`

Cette liste aussi est loin d'être exhaustive mais vous permet de visualiser les principaux changements. On peut noter entre autre que l'attribut *target* dans les liens est à nouveau disponible alors qu'elle était déprécié jusqu'à maintenant.

## 6. Exercice : Testez votre compréhension...

### Exercice

1) Quelle est la syntaxe pour déclarer l'encodage des caractères du document en UTF-8 en HTML5?

- ☐ `<meta encoding="text/html; charset=utf-8">`
- ☐ `<meta charset="text/html; UTF-8">`
- ☐ `<meta charset="utf-8">`

### Exercice

2) Quelle est la méthode pour associer une légende complète à une illustration ?

- ☐ `<figure><figcaption>La légende...</figcaption></figure>`
- ☐ `<figure src="image.jpg" legend="#cap1"></figure><figcaption id="cap1">La légende...</figcaption>`
- ☐ `<figure><legend>La légende...</legend></figure>`

### Exercice

3) Quel attribut permet d'afficher une image par défaut pour l'élément `<video>` ?

- ☐ `<video preview="apercu.jpg">`
- ☐ `<video><param name="thumbnail" value="apercu.jpg" /></video>`
- ☐ `<video poster="apercu.jpg">`

## Exercice

---

4) Le code HTML suivant contient une erreur, indiquez le numéro de la ligne à laquelle l'erreur a été commise.

```
1 <audio controls="controls">
2   <source src="chanson_1.ogg" type="audio/mp3" />
3   <source src="chanson_2.mp3" type="audio/mp3" />
4   Votre navigateur ne supporte pas la balise AUDIO.
5 </audio>
```

- ☐ Ligne 1
- ☐ Ligne 2
- ☐ Ligne 3
- ☐ Ligne 4
- ☐ Ligne 5

## Exercice

---

5) Dans le formulaire de contact de la page de votre entreprise, les visiteurs ramènent souvent des mails saisis avec des erreurs d'orthographe en français et cela agace votre patron qui vous demande de corriger cela.

Quel(s) attribut(s) HTML5 pouvez-vous utiliser sur le champ de saisie du message pour proposer un correcteur d'orthographe à vos visiteurs lors de la saisie ?

- ☐ draggable
- ☐ Contenteditable
- ☐ Spellcheck

# Section 3 - La validation des données avec HTML5



HTML5 introduit de nombreuses nouveautés pour les formulaires pour améliorer l'aide à la saisie et les contrôles disponibles pour l'utilisateur.

Plusieurs attributs simples à mettre en place améliorent la prise en charge des formulaires à partir de la balise `<input>`.

## 1. 1. Le placeholder, une indication

*placeholder* est un attribut qui permet de renseigner un texte indicatif par défaut dans un champ de formulaire.

C'est une valeur qui s'efface dès que l'utilisateur active le champ de formulaire ou commence à écrire dedans. Cependant, la présence d'un *placeholder* ne vous dispense pas de renseigner un *label* pertinent. Bien au contraire, le *placeholder* n'est qu'un indice supplémentaire, il n'est pas indispensable.

```
1 <input type="text" placeholder="Entrez un pseudo">
```

L'attribut *placeholder* peut être placé sur les éléments `<input>` (de type *text*, *search*, *password*, *url*, *tel*, *email*) et `<textarea>`

Quelques styles de *placeholder* selon les navigateurs :

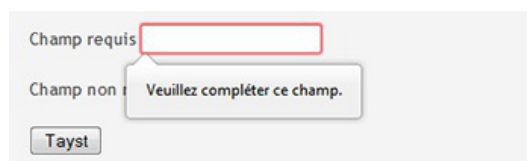


Source : <https://www.alsacreations.com/xmedia/doc/original/html-5-placeholder.png>

## 2. 2. Les champs obligatoires avec l'attribut required

L'attribut *required* permet de rendre obligatoire le remplissage d'un champ et bloquer la validation du formulaire si l'un des champs concernés par cet attribut n'a pas été renseigné.

Cet attribut n'a pas besoin de valeur car sa seule présence suffit, cependant les syntaxes suivantes sont admises :



Aperçu d'une validation sur Firefox 9

```
1 <!-- sans valeur -->
2 <input type="text" required>
3
4 <!-- avec la valeur required -->
5 <input type="text" required="required">
```

L'attribut *required* ne vous dispense pas pour chaque champ obligatoire de rajouter dans l'étiquette associée une astérisque \* et de faire précéder le formulaire du message "Les champs indiqués par une \* sont obligatoires".

L'attribut *required* peut être placé sur les éléments :

- `<input>` : de type *text*, *search*, *password*, *url*, *tel*, *email*, *date*, *datetime*, *datetime-local*, *month*, *week*, *time*, *number*, *checkbox*, *radio*, *file*
- `<textarea>`

## 3. 3. Les masques et la validation des données

De nouveaux types pour l'attribut *type=""* de la balise `<input>` voient le jour. Ils visent à faciliter la saisie des données par l'utilisateur .

*Exemple* : `<input type="date">` devrait permettre d'afficher un calendrier pour sélectionner une date.

Il existe donc à présent de nombreux nouveaux types qui viennent compléter les anciens types *text*, *hidden* et *password*, *checkbox* et *radio*.

Les principaux sont :

- *number* : pour renseigner une valeur numérique.
- *email* : pour accueillir des adresses mails valides.
- *url* : pour accueillir des liens hypertextes.
- *color* : pour accueillir une couleur au format hexadécimal (exemple : #eeaaaff). Certains navigateurs affichent une palette pour inviter l'internaute à sélectionner une couleur.
- *date* : pour accueillir une date, certains navigateurs affichent un calendrier.
- *time* : pour accueillir un horaire, certains navigateurs affichent un sélecteur.
- *file* : pour accueillir un champ afin d'uploader un fichier.
- *search* : pour permettre aux internautes de réaliser une recherche par mot-clé.

Vous pouvez avoir les détails de l'utilisation de chacun de ses type sur le site *Alsacreation* : <https://www.alsacreation.com/tuto/lire/1372-formulaires-html5-nouveaux-types-champs-input.html>

Pour l'instant, voilà quelques illustrations et exemples :

- Forcer une valeur avec *number*

```
1 <p>
2   <label>Age   </label>
3   <input type="number" />
4 </p>
5 <p>
```



```

6 <label>Age </label>
7 <input type="number" min="18" max="120" />
8 </p>

```

donnera le résultat suivant si l'utilisateur saisi une valeur qui n'est pas comprise entre *min* et *max* dans le deuxième champ alors que le premier champ accepte toutes les valeurs pourvu que ce soit des nombres.

On remarque que le champ est fourni éventuellement avec des petits boutons à droite que les utilisateurs peuvent utiliser pour augmenter ou diminuer le chiffre. L'incrément à chaque clic pourra être défini avec l'attribut *step* :

```

1 <p>
2 <label>Age </label>
3 <input type="number" min="18" max="120" step="2" />
4 </p>

```

fera avancer l'âge de deux en deux.

### - La champ de type date

Le champ type date sert à éditer les dates. Selon le navigateur, il peut être associé automatiquement à un calendrier qui facilite la sélection de la date.

```

1 <p>
2   Quelle est votre date de naissance ?<br/>
3   <input type="date" />
4 </p>

```

Différents types de champs de sélection de *date* existent :

- *date*: pour la date (05/08/1985 par exemple) ;
- *time*: pour l'heure (13:37 par exemple) ;
- *week*: pour la semaine ;
- *month*: pour le mois ;
- *datetime*: pour la date et l'heure (avec gestion du décalage horaire) ;

- *datetime-local* : pour la date et l'heure (sans gestion du décalage horaire).

Pour choisir une heure par exemple, on utilisera le code :

```
1 <p>
2   Heure de Livraison souhaitée<br/>
3   <input type="time" />
4 </p>
```




### - *Accepter des conditions*

*Required* est un attribut booléen aussi, s'il est omis le champ est considéré comme facultatif, c'est-à-dire n'étant pas un champ absolument nécessaire à renseigner par l'internaute.

Dans le cas suivant, le paramètre requis est sur un bouton *checkbox* d'acceptation.

```
1 <label>J'accepte </label>
2 <input type="checkbox" required />
```



## 4. Exercice : Entraînez-vous avant le devoir !

### Exercice

---

1) Dans un `<input>`, `placeholder` est désormais le remplaçant du label avec l'arrivée du HTML5

- ☐ Vrai
- ☐ Faux

### Exercice

---

2) Dans un formulaire, quel type HTML5 est approprié dans un `<input>` pour demander la date de la rentrée académique ?

- ☐ `datetime-local`
- ☐ `datetime`
- ☐ `date-local`
- ☐ `date`

### Exercice

---

4) Lesquels de ces codes sont justes ?

- ☐ `<input type="text" required>`
- ☐ `<input type="text" required="Yes">`
- ☐ `<input type="required" >`
- ☐ `<input type="text" required="required">`
- ☐ `<input type="text" required="valid">`

### Exercice

---

5) Lorsqu'on a défini un masque de saisie avec la directive `type`, a-t-on encore besoin d'utiliser `required` sur le même élément de formulaire ?

- ☐ Oui
- ☐ Non