101 balises HTML5 + exemples

Par Quentin Busuttil

Date de publication : 21 décembre 2014

Dernière mise à jour : 18 janvier 2018

Le HTML5 c'est plein de bonnes choses, des API JavaScript, du CSS3, mais aussi et surtout des balises HTML ! Et cette nouvelle mouture d'HTML apporte son lot de nouveautés. Cette version du langage met en avant la sémantique, ainsi, de nombreuses balises permettent de remplacer les balises génériques div et span. Du layout à la citation, faisons le tour de tout ça, exemples à l'appui voulez-vous ?

Commentez

I - Les balises structurantes

Un des plus gros apports de l'HTML5 a été l'arrivée de balises sémantiques concernant le layout des pages - autrement dit les balises qui définissent la structure globale de la page. Au lieu d'utiliser des div à tour de bras, nous avons maintenant les balises section, article, aside, header, footer, nav, figure, figcaption et main !

main

Représente le contenu principal d'une page ou application. Il ne doit y en avoir qu'une seule par page, donc pas deux fois main dans la même balise body. Vous placez main là où vous auriez avant utilisé ou .

```
<!DOCTYPE html>
<html lang="fr">
<head>
<meta charset="UTF-8" />
<title>Utilisation de main</title>
</head>
<body>
  <header>Mon super site
  <nav>
    <a href="/">Home</a>
    </nav>
  </header>

  <main>
    <article>
      <h1>Mon super article</h1>
      <p>Bla bla bla</p>
    </article>

    <aside>
      <p>Plus d'infos</p>
    </aside>
  </main>
  <footer>Copyright 2015</footer>
</body>
</html>
```

section

Représente une portion ou section générique d'un document. Dans ce contexte, et contrairement à une div, section est un regroupement thématique de contenu. De ce fait, il prend la plupart du temps un titre h1.

article

Un peu plus précis que section, on se retrouve souvent à hésiter entre l'emploi de l'un ou de l'autre. En réalité, article est un type spécifique de section, lequel représente un contenu se suffisant à lui-même, de sorte qu'il pourrait être distribuable indépendamment de la page sur laquelle il se trouve sans perdre de son sens.
Ainsi, on trouvera assez souvent des balises article dans des balises section. Bien que l'inverse soit syntaxiquement correct, c'est un cas que l'on rencontre moins souvent. Pensez donc à un journal, on y trouvera par exemple une section sport avec divers articles sur le sport (foot, rugby…). Néanmoins, section se porte très bien sans aucune autre balise de layout. On y met un titre, quelques paragraphes et roulez jeunesse !

header

S'utilise en en-tête de page. De manière générale, on ouvre cette balise immédiatement après la balise body, et on y place le logo ou titre du site dans un h1, souvent accompagné de sa baseline en h2, ainsi que le menu

principal (vous avez dit nav ?). Néanmoins, contrairement à toute attente, il peut y avoir plusieurs header sur une page. En effet, les spécifications prévoient que chaque <section> puisse avoir son propre header.

nav

Assez explicite, cette balise sert à indiquer une liste de liens dédiée à la navigation générale du site. Il peut donc y avoir plusieurs nav sur une même page. Comme il est courant de voir sur un site un menu principal en haut de page, placé horizontalement, et un second menu, vertical cette fois, lequel permet de naviguer selon une autre logique. Ainsi, dans un tel cas, on aura deux menus, donc deux fois la balise nav, cela ne pose aucun problème.

footer

Alors là, pas de doute, c'est la balise en bas de page dans laquelle on place le copyright et quelques autres infos annexes (mentions légales, etc.)… Mais pas que : de la même manière que pour les header, les balises section prévoient la possibilité d'avoir un footer.

aside

Généralement perçu comme une balise marquant du contenu situé sur le côté, c'est malheureusement une perception un peu simpliste.
C'est une balise pour laquelle les spécifications ont évolué. À la base, une sidebar qui n'a pas de relation spécifique avec le contenu d'un article de blog - lequel est alors le contenu principal de la page - ne « mérite » donc pas le titre de sidebar. Nous l'avons vu pour nav, un menu vertical situé sur le côté d'une page est un élément nav, pas un aside. Cependant, à défaut de mieux, aside peut dorénavant servir à indiquer un contenu annexe comme une blogroll ou de la pub par exemple. Elle doit dans ce cas là ne pas se trouver dans une balise article.
Dans sa forme la plus pure, aside doit être utilisé pour un contenu qui vient en complément de l'information principale, de préférence dans un article. On voit souvent dans les journaux et magazines des encadrés « en savoir plus » ou « focus sur », qui délivrent une information complémentaire et/ou sur un point précis de l'article, pensez à aside en ces termes.

I-A - Précisions sur section et article

```
<section>
  <article>
    <h1>PHP expliqué</h1>
    …
  </article>

  <article>
    …
  </article>
</section>
```

N'utilisez pas la balise section (et encore moins article à des fins de stylisation). De manière générale, si vous hésitez entre les trois, voici comment procéder, du plus spécifique (et sémantique) au général :

- le contenu ferait-il sens en dehors de cette page, pourrait-il être distribué indépendamment de son contexte ? Si oui, alors article est pour lui (le W3C considère d'ailleurs les commentaires comme des articles) ;
- le contenu parle-t-il d'une même thématique et est lié ? Alors utilisez section ;
- votre contenu ne répond pas aux conditions ci-dessus, la div est alors faite pour ça !

Il est aussi important de comprendre la hiérarchisation des titres de différents niveaux et comment ils sont interprétés s'ils sont dans des balises article ou section. Dans une page sans ces balises, on place généralement un h1 en titre, puis un ou plusieurs h2 de même niveau, etc. Cependant, avec les balises section et article, on peut mettre plusieurs h1 sur une même page. Voyez cet exemple du W3C :

```
<!-- Sémantiquement, ces deux notations sont équivalentes !-->
<body>
  <h1>Let's call it a draw(ing surface)</h1>
  <h2>Diving in</h2>
  <h2>Simple shapes</h2>
  <h2>Canvas coordinates</h2>
  <h3>Canvas coordinates diagram</h3>
  <h2>Paths</h2>
</body>

<body>
  <h1>Let's call it a draw(ing surface)</h1>
  <section>
    <h1>Diving in</h1>
  </section>

  <section>
    <h1>Simple shapes</h1>
  </section>
  <section>
    <h1>Canvas coordinates</h1>
    <section>
      <h1>Canvas coordinates diagram</h1>
    </section>
  </section>

  <section>
    <h1>Paths</h1>
  </section>
</body>
```

II - Les balises bloc

figure

Représente une unité de contenu qui peut être dissociée du reste du document sans perdre son sens. figure prend tout son intérêt quand il est associé à figcaption afin de grouper un élément et sa description. C'est la seule manière d'indiquer sémantiquement qu'une description ou légende a trait à un élément en particulier. Par exemple, dans le cas d'une image, sans figure et figcaption, on aurait mis une légende sous cette dernière dans une balise p. Cependant, dans le code, rien n'indique clairement que ce paragraphe de légende se rapporte à l'image qui le précède. On utilisera donc ces deux balises pour associer une légende à un média, un diagramme, un exemple de code et donc souvent en association avec les balises img, audio, video, svg, canvas, code et pre.

```
<!-- on associe trois images à une légende commune !-->
<figure>
  <img src="/couple-bonobo.jpg" alt="Accouplement de bonobos">
  <img src="/bebe-bonobo.jpg" alt="Bébé Bonobo">
  <img src="/baguette-bonobo.jpg" alt="Bonobo qui se nourrit">
  <figcaption>De gauche à droite : un accouplement de bonobos,
un nourrisson et un bonobo utilisant une baguette
pour attraper des termites</figcaption>
</figure>
```

blockquote

Permet de faire des citations. Cette balise de type bloc peut tout à fait contenir d'autres éléments de type bloc, comme des paragraphes. D'ailleurs en XHTML, vous ne validerez pas si le blockquote contient autre chose que d'autres blocs. On lui adjoint souvent la balise cite pour attribuer la citation.

Attention toutefois, l'élément cite doit être placé en dehors de la balise. La logique sémantique veut en effet que tout ce qui est dans la balise blockquote soit une citation. Cette balise possède aussi un attribut optionnel, cite, lequel permet de donner l'URL de la citation. Bien que recommandé par l'HTML5, l'URL précisée dans cet attribut n'est pas affichée par les navigateurs.

```
1. <blockquote cite="http://buzut.fr">
2. <p>Ma petite citation</p>
3. </blockquote>
4. <p><cite>Blog de buzut</cite> - http://buzut.fr</p>
```

cite, qui existe en balise et en attribut, permet de citer le titre d'un ouvrage - ou sa source, comme une URL par exemple - pas son auteur.
On trouve sur le site du W3C un exemple plus complet (complexe ?) mettant en œuvre les balises figure et figcaption pour clairement relier la source de la citation à la citation elle-même :

```
1. <pre class="prettyprint">
2. <figure>
3.   <blockquote>
4.     <p>The truth may be puzzling. It may take some work to
5.     grapple with.</p>
6.   </blockquote>
7.   <figcaption>
8.   Carl Sagan, in "<cite>Wonder and
9.   Skepticism</cite>", from
10.   the <cite>Skeptical Enquirer</cite> Volume 19, Issue 1
11.   (January-February
12.   1995)
13.   </figcaption>
14. </figure>
15. </pre>
```

q

Quant à elle, concerne les citations sur une ligne. Petit exemple :

```
<p>Darth Vador a dit <q>je suis ton père</q></p>
```

hr

Indique un changement thématique. Sémantiquement, c'est assez semblable à la fin d'une section et le début d'une autre. Le WHATWG (Web Hypertext Application Technology Working Group) précise par ailleurs qu'il est inutile de l'utiliser entre deux sections, car les sections, en plus du titre h1 rendent ce changement de thématique évident.

II-A - Les balises de liste

ul

Balise que l'on utilise extrêmement souvent. Elle indique une liste non ordonnée (c'est-à-dire qu'il n'y a pas de hiérarchie spécifique entre les différents éléments de la liste. On place des li à l'intérieur de celle-ci.
Que la liste soit une ul ou une ol, on y place des li ! Attention à une chose, l'**imbrication**. Il arrive que l'on doive mettre une liste dans une autre, une sous-liste en d'autres termes, c'est tout à fait correct. Cependant, une erreur très commune est d'inclure la sous-liste au niveau du li. Exemple ci-dessous :

```
1. <!-- bien :) !-->
2. <ul>
3.   <li>un</li>
4.   <li>deux</li>
5.   <li>trois</li>
6.   <li>
7.     <ul>
8.       <li>3.1</li>
9.       <li>3.2</li>
10.    </ul>
11.  </li>
12.  <li>un</li>
```

```
</ul>

<!-- PAS bien :( !-->
<ul>
  <li>un</li>
  <li>deux</li>
  <li>trois</li>
  <ul>
    <li>3.1</li>
    <li>3.2</li>
  </ul>
  <li>un</li>
</ul>
```

ol

Un petit peu moins souvent utilisé que ul. Cette liste indique qu'il y a une hiérarchie entre les éléments de la liste. Par exemple, dans une recette de cuisine, vous ordonnez les tâches, on ne bat pas les œufs avant de les avoir cassés. Elle s'utilise de la même manière que ul.

dl

Représente une liste de définition (definition list). On trouve à l'intérieur de celle-ci les termes définis et leur(s) définition(s). Il peut en effet y avoir plusieurs définitions pour un même terme. De plus, il peut y avoir plusieurs termes ayant la même signification. Un petit exemple n'est pas de refus, je crois !

```
<!-- On peut aussi l'utiliser pour marquer des métadonnées !-->
<dl>
  <dt>Autheurs</dt>
  <dd>Buzut</dd>
  <dd>Son clone</dd>
  <dt>Outil de rédaction</dt>
  <dd>Un clavier</dd>
</dl>

<!-- exemple inverse, deux mots à définir pour un sens !-->
<dl>
  <dt>Clef</dt>
  <dt>Clé</dt>
  <dd>Un objet servant à ouvrir une serrure</dd>
</dl>
```

III - Les balises inline

strong

Le réflexe est de se dire que cette balise indique que ce qu'elle contient doit être en gras. **FAUX** ! On ne cesse pourtant d'entendre que l'on doit séparer la sémantique de la mise en page, c'est d'ailleurs l'objet de CSS. strong, même si les navigateurs le mettent en gras, sert donc à marquer des éléments qui ont une forte importance, un caractère sérieux ou un caractère insistant.

b

Très trompeur : on a en effet souvent lu, vu et entendu que b signifie bold (gras en anglais). Pourtant, c'est encore et toujours au CSS que revient la tâche de régler la casse. b est aussi affiché en gras par les navigateurs. Le WHATWG (Web Hypertext Application Technology Working Group) préconise l'utilisation de b pour attirer l'attention sur un élément à des fins utilitaires sans porter de sens particulier, d'intonation de voix ou d'humeur.
On peut l'imaginer par exemple pour l'indiquer de manière visible les noms des personnages dans un script de cinéma ou un dialogue de théâtre.

em

Représente une emphase sur un mot ou expression. Les navigateurs l'affichent en italique, mais de la même manière que pour strong ou b, son but n'est pas d'indiquer une mise en forme particulière.

i

Sert à indiquer une portion de texte prononcé dans une voix, tonalité, esprit ou langue alternative au discours principal, de fait, il indique une qualité de texte différente. Cela peutêtre un terme technique, une réflexion, etc.

cite

On a vu plus haut que cite s'utilise souvent conjointement à blockquote. De manière plus générale, cite permet de baliser le nom d'un travail. Ce peut être un livre ou un article, mais aussi un film, une musique, le nom d'une exposition ou d'une comédie musicale. Néanmoins, ce ne peut être le nom d'une personne.

del

Permet de dire qu'un élément a été supprimé. Par défaut, les navigateurs rendent le texte en barré.

s

Représente une portion de texte qui n'est plus juste ou à jour. Les navigateurs affichent le texte en barré.

ins

Représente un ajout au document. Le W3C préconise de ne pas englober de paragraphe avec cette balise. En outre, on voit parfois l'attribut datetime préciser la date de l'insertion (cf. time).

small

Représente les choses moins importantes (telles que les petites lignes d'un contrat). Sur un site e-commerce, on verra par exemple une offre :

```
1. <p>Offre promotionnelle famille : 2 places achetées, 2 places
2. offertes !<br>
3. <small>Places offertes pour enfants de moins de 12 ans
4. uniquement</small></p>
```

time

Permet d'encadrer un élément de type date. Cette balise prend toute sa puissance avec ses différents attributs.

var

Permet d'indiquer une variable. Cela peut être dans une expression mathématique ou encore une variable en programmation informatique.

code

Sert à indiquer qu'une portion de texte représente du code source.

kbd

Représente une entrée utilisateur, typiquement une touche du clavier (ou combinaison de touches). Cependant cela peut aussi correspondre à une autre entrée, comme une commande vocale, etc.

mark

Utile pour attirer l'attention de l'utilisateur sur une portion précise du texte. Soit parce que cette portion est marquée pour référence de par sa pertinence : le W3C donne l'exemple d'un moteur de recherche qui balise dans ses résultats les termes que l'internaute a entrés, soit dans une citation, cela signifiant alors que cette portion n'avait pas été mise en évidence dans le texte cité et que cette mise en avant est faite a posteriori.

dfn

Permet de baliser un terme qui va être défini.

```
<p><dfn>Procrastiner</dfn> est l'art de remettre les
choses au lendemain</p>
```

u

Parmi ses différents usages, il y en a qui m'échappent assez pour que je ne les mentionne même pas. Bref, je ne retiendrai que celui que je seraiS - notez le « s » - un jour amené à utiliser. La balise u sert donc à mettre en évidence les fautes d'orthographe ou de grammaire. Jetez un œil à **l'article de HTML5 doctor** pour en savoir plus.

IV - Les balises de formulaire

Les balises de formulaires sont de types bloc. Cependant, comme elles sont un peu particulières, je leur donne un petit espace à part. En outre, on mettra souvent des div et autres balises de type bloc entre les balises form puisque c'est indispensable si on veut y placer du texte.

form

Permet de déclarer un formulaire. Ainsi, toutes les autres balises de formulaire se trouveront entre deux balises form. Cette balise possède deux attributs essentiels : method et action. Ils servent à indiquer où et comment envoyer le formulaire. method peut prendre comme valeur soit « get », soit « post » et action prend l'URL de la page qui reçoit le formulaire. Dernier petit détail, on trouve aujourd'hui souvent des balises de formulaire type input en dehors des balises form, car les données sont traitées par le biais de JavaScript et envoyées en AJAX.

```
<form method="post" action="page-de-reception.php">
   <input type="text" name="test" placeholder="test">
</form>
```

input

La balise de formulaire la plus classique. Elle affiche un petit champ de texte, lequel permet d'entrer un texte. Elle prend comme attribut type qui définit son type, lequel peut être text, number, password, email, tel, url, range, color, date, datetime, datetime-local, month, time, week, search et file.
Mention spéciale pour le type hidden qui fait un champ caché. Dans ce cas, l' input sert à transmettre des informations entre différentes pages.
Les noms sont plutôt parlants. Sachez qu'ils permettent d'afficher des contrôles plus adaptés à leur contexte : pass n'affiche que des points au lieu du texte entré, d'autres comme email, tel ou url permettent aux terminaux tactiles d'afficher un clavier adapté, etc.
L'attribut name permet de donner un nom au champ et c'est par ce nom que vous retrouverez les données associées sur la page de réception (avec $_GET['nom_de_mon_input'] en PHP par ex.). On peut aussi lui donner une valeur par défaut avec un texte prédéfini, pour cela, il faut renseigner l'attribut value. Enfin, l'id

permettra ici d'associer ce champ avec la balise label de sorte qu'un clic sur cette balise place le curseur dans l'input.
Enfin, petit aparté pour file. Cet input permet, comme son nom l'indique, d'envoyer des fichiers tels que photos, vidéos, et autres documents. Pour que cela fonctionne, il faut bien penser à encoder les fichiers pour l'envoi. Cela se fait simplement en ajoutant le type d'encodage dans la balise form : enctype="multipart/form-data". Rien de plus. Petite astuce, si vous voulez permettre l'envoi de plusieurs fichiers à la fois, ajoutez l'attribut multiple à votre input.

datalist

Complète l'input text en permettant d'avoir des suggestions automatiques au fur et à mesure de la saisie. Voyez l'exemple d'utilisation ci-dessous (j'ai pris **l'exemple de alsacreations**, car je le trouve plutôt sympathique !) :

```
<label for="choix_bieres">Indiquez votre bière préférée :</label>
<input list="bieres" type="text" id="choix_bieres">
<datalist id="bieres">
  <option value="Meteor">
  <option value="Pils">
  <option value="Kronenbourg">
  <option value="Grimbergen">
</datalist>
```

textarea

Similaire à la balise input en mode text, sauf que c'est un bloc de texte, donc sur plusieurs lignes. C'est évidemment utile pour de grandes quantités de texte. En plus du placeholder (cf. les attributs) et du name (même usage que pour l'input), on peut définir sa taille via les attributs rows et cols, lesquels définissent la taille du bloc en nombre de lignes et de colonnes. Personnellement, je préfère définir la taille via le css, mais sachez que c'est possible comme cela. Dernier point, si on veut lui attribuer une valeur par défaut (texte déjà renseigné), il suffit de la placer entre <textarea …> et </textarea>.

```
<textarea rows="50" cols="100" name="mon-bloc-de-texte-">

</textarea>
```

fieldset

Permet de regrouper des champs. Dans le cas où vous avez de nombreux champs, il peut être judicieux de les regrouper par thèmes. Fieldset permet donc des regroupements et à l'aide de la balise legend vous pourrez donner un intitulé à ces groupes.

```
<fieldset>
  <legend>Vos coordonnées</legend>

  <label for="nom">Nom</label>
  <input type="text" name="nom" id="nom">

  <label for="prenom">Prénom</label>
  <input type="text" name="prenom" id="prenom">

   <label for="email">Email</label>
   <input type="email" name="email" id="email">
</fieldset>

<fieldset>
  <legend>Votre adresse</legend>
  <label for="rue">Rue</label>
  <input type="text" name="rue" id="rue">

  <label for="ville">Ville</label>
  <input type="text" name="ville" id="ville">
```

```
21.
22.   <label for="cp">Code postal</label>
23.   <input type="text" name="cp" id="cp">
24. </fieldset>
```

Le bouton du formulaire

C'est un input avec un type particulier. Le type peut prendre les valeurs submit, reset, image et button. Les deux premiers sont explicites. Le type image est similaire au submit à cela près que l'aspect du bouton est défini par une image, il faut donc lui adjoindre un attribut src avec la source de l'image du bouton. Enfin button ne fait rien par défaut, on l'active en général via JavaScript.

IV-A - Les différentes formes de l'input

Nous l'avons vu plus haut, l'input possède de nombreux type. Cependant, dans bien des cas, ça reste un champ de texte dont on a précisé la finalité (mot de passe, téléphone, etc.). Il y a cependant des cas où le type de l'input n'appelle pas à la saisie clavier, mais à une sélection, dans ces cas, la forme est toute autre.

radio

Ce sont des input pour choisir parmi plusieurs options. Un peu comme les cases à cocher, mais où il n'est possible d'en choisir qu'une seule. Il faut pour cela mettre radio en tant que type de l'input, et pour que le navigateur reconnaisse que les différentes options font partie du même groupe, leur donner le même nom. Enfin, le label est très important ici, car les utilisateurs cliquent souvent sur le texte et non pile-poil sur le bouton lui-même.

```
1. <input type="radio" name="sexe" value="femme" id="femme">
2. <label for="femme">Femme</label><br>
3.
4. <input type="radio" name="sexe" value="homme" id="homme">
5. <label for="homme">15-25 ans</label><br>
```

checkbox

similaire à radio à cela près qu'il est possible de cocher plusieurs cases. Elles doivent donc avoir des noms (name) différents.

select

Permet de choisir une option pour les listes déroulantes. Il est possible de grouper certaines sélections pour plus de clarté avec optgroup. Voyez l'exemple ci-dessous :

```
1. <label>Indiquez votre pays :</label>
2. <select name="region">
3.   <option value="France">France Métropolitaine</option>
4.   <optgroup label="Europe">
5.     <option value="1">Allemagne</option>
6.     <option value="3">Danemark</option>
7.     <option value="2">Espagne</option>
8.     <option value="4">Finlande</option>
9.     <option value="6">Autres pays d'Europe</option>
10.   </optgroup>
11.
12.   <optgroup label="Outre-mer">
13.     <option value="OM1">Guadeloupe</option>
14.     <option value="OM1">Martinique</option>
15.     <option value="OM1">Réunion</option>
16.     <option value="OM1">Guyane</option>
17.   </optgroup>
18.
19.   <optgroup label="Amérique">
```

```
20.     <option value="7">USA</option>
21.     <option value="7">Canada</option>
22.     <option value="8">Amérique centrale</option>
23.     <option value="8">Amérique du Sud</option>
24.   </optgroup>
25.
26.   <optgroup label="Afrique">
27.     <option value="6">Maghreb</option>
28.     <option value="7">Afrique centrale</option>
29.     <option value="7">Afrique du Sud</option>
30.   </optgroup>
31. </select>
```

IV-B - Les attributs de formulaire

for

S'applique au label afin de le lier au champ qu'il décrit. De cette manière, un clic sur le label mettra le curseur dans le champ en question (ou le sélectionnera s'il s'agit de cases à cocher). Le label doit correspondre à l'id du champ de référence.

```
1. <input id="femme" type="radio" name="sexe">
2. <label for="femme">Femme</label><br>
3.
4. <input id="homme" type="radio" name="sexe">
5. <label for="homme">15-25 ans</label>
```

name

Donne le nom du champ. Ce nom sera associé à la valeur du champ lors de l'envoi du formulaire. On utilisera donc ce nom pour récupérer la valeur côté serveur. Par exemple en PHP $_POST['nom'].

value

Si value est renseignée, la valeur associée est préremplie dans le champ et sera envoyée avec le formulaire si rien d'autre n'y est défini. En outre, value sert à définir la valeur des inputs qui ne reçoivent pas d'entrée clavier (comme les radio, checkbox, hidden, select, etc.).

placeholder

placeholder permet d'afficher un texte indicatif à l'intérieur des champs de formulaires qui disparaît lorsque l'on y entre des données.

autofocus

Attribut qui permet de placer le focus directement sur l'élément en question au chargement de la page. Ainsi par exemple, inutile pour l'internaute de cliquer avec sa souris dans un champ de recherche avant de taper sa requête, le curseur y est directement et automatiquement positionné.

required

Signifie qu'un champ doit obligatoirement être rempli avant d'envoyer le formulaire. Les navigateurs récents le prenant en charge empêcheront l'envoi du formulaire s'il n'est pas complet et signaleront l'erreur à l'internaute. En plus de required, HTML5 prévoit une API de validation. Bien que certains usages ne requièrent que l'attribut, les cas plus avancés nécessitent le recours à JavaScript. Je n'expliquerai pas l'usage de cette API ici, cependant, **Google Developers**, le **MDN** et **HTML5 Rocks** présentent de bons articles à ce sujet.

pattern

Il s'accompagne régulièrement de required. Cet attribut permet de spécifier une **REGEX** qui permettra de valider le contenu du formulaire

autocomplete

autocomplete, comme son nom le suggère, permet d'indiquer au navigateur que nous souhaitons qu'il complète automatiquement le champ. Ainsi, nous évitons à notre utilisateur la saisie fastidieuse de ses données. Cet attribut **ne fonctionne qu'avec les formulaires en** POST et est ignoré si l'input est de type hidden, password, checkbox, radio, file, ou button. En outre, l'attribut peut être complété par le type de données que nous voulons obtenir. Il y a cinq grands types de données :

Nom avec name ;
Email avec email ;
Adresse avec street-address, locality (la ville), region, postal-code, country ;
Téléphone avec tel ;
Carte de crédit avec cc-name, cc-number, cc-csc (le cryptogramme), cc-exp-month, cc-exp-year, cc-exp, cc-type (le type de carte : CB, Visa…)

```
1. <input type="text" name="name" autocomplete="name">
2. <input type="email" name="email" autocomplete="email">
3. <input type="tel" name="tel" autocomplete="tel">
4. <input type="text" name="street" autocomplete="street-address">
5. <input type="text" name="city" autocomplete="locality">
6. <input type="text" name="cp" autocomplete="postal-code">
```

Sachez aussi que le HTML5 met à disposition **une API JavaScript, requestautocomplete**, permettant de demander programmatiquement au navigateur les données nécessaires à un formulaire.

V - Les attributs

Certaines balises prennent des attributs optionnels qui viennent apporter un complément d'information. Certains sont classiques, tels que le alt de la balise img, mais on en relève aussi de moins connus…

id

Permet de donner un identifiant unique à une balise. **Unique** signifie que cet identifiant ne doit pas apparaître plus d'une fois dans une page. Il sert principalement à retrouver un élément particulier pour lui attribuer un style en CSS ou y avoir accès en JavaScript. Dans le cadre d'un champ de formulaire, cet attribut permet aussi d'associer la balise avec un champ label via l'attribut for. Petite précision, dans les bonnes pratiques, il est conseillé d'éviter de trop utiliser l'id pour le css : je vous laisse avec cet **article sur le sujet (en anglais)**.

class

Remplit les mêmes fonctions que l'id à cela près qu'une classe peut figurer plusieurs fois sur une page et qu'il n'a pas la fonction d'id lié aux formulaires.

data-*

Les data attributes permettent de stocker des informations dans le HTML. À tous ceux qui ont déjà stocké des données dans les attributs id, class ou rel destinées à des traitements JavaScript, le HTML5 a pensé à vous. Vous pouvez créer autant d'attributs data que vous le souhaitez et les nommer comme vous le souhaitez. Ces balises ont comme unique rôle le stockage d'informations arbitraires.

```
1. <ul class="cookies">
2. <li data-price="5" data-calories="1500">Gros cookie</li>
```

```
3. <li data-price="3" data-weight="800">Petit cookie</li>
4. </ul>
```

hidden

Permet de masquer un élément. Il remplace avantageusement l'attribution d'une class css hidden !

lang

Sert à indiquer la langue d'une portion de texte qui est dans une langue différente de celle de la page. Par exemple, une citation (exemple du W3C, la classe) :

```
<p>There is a certain <i lang="fr">je ne sais quoi</i> in the air.</p>
```

datetime

Permet de préciser une date :

```
1. <p>La réunion commencera
2. <time datetime="2014-10-29 15:00+02:00">
3.   Mercredi 29 à 15 heures
4. </time>,
5. soyez ponctuel, c'est une question de respect</p>
```

Si les dates au format anglo-saxon (aaaa/mm/jj) seront compréhensibles pour les moteurs de recherche, les dates au format français ou autres variantes ne sont pas forcément évidentes et peuvent prêter à confusion. Cependant, il serait peu ergonomique d'optimiser la date affichée pour les ordinateurs au détriment de la lisibilité de votre cible (on peut par exemple afficher des dates relatives [dans x jours]). C'est là que l'attribut datetime entre en jeu !
Petite explication pour le datetime, plusieurs formats sont acceptés, celui-ci est le plus complet tout en restant standard et simple. Date à l'anglaise, aaaa-mm-jj puis hh:mm et le + est ici pour préciser que nous parlons en heures françaises, c'est-à-dire en GMT+2, ce qui est facultatif si c'est évident !

V-A - Les attributs de liens

download

Permet d'indiquer les liens qui ont vocation à être téléchargés plutôt qu'ouverts. De nombreux navigateurs ouvrent en effet les liens qu'ils savent lire s'il s'agit de fichiers (image, pdf, audio, vidéo…). Il fallait alors utiliser un langage côté serveur (ce qui impliquait d'avoir un accès au serveur et rendait la tâche complexe avec les CMS (Content Management System ou système de gestion de contenu)) pour envoyer un en-tête au navigateur et lui dire de télécharger l'image au lieu de l'ouvrir. Aujourd'hui, grâce à cet attribut, cette complexité est révolue ! Il suffit de placer cet attribut dans le lien, de lui préciser optionnellement le nom du fichier tel qu'il doit apparaître sur le système de l'utilisateur et voilà.

```
<p>Vous pouvez télécharger mon dernier son en
<a href="musique-de-merde.mp3" download="super-son.mp3">cliquant sur ce
lien tavu !</a></p>
```

target

Assez connu parce qu'il permet d'ouvrir les liens dans de nouveaux onglets. Ainsi target="_blank" permet d'ouvrir un lien dans un nouvel onglet. Il existe aussi le self qui ouvre dans le même onglet (c'est le comportement par défaut sans l'utilisation de target); parent : le document associé est ouvert dans l'onglet qui est le parent immédiat; top : j'ai l'impression que ça se comporte comme self…; enfin name qui permet d'ouvrir le contenu dans une iframe nommée. Tous ces éléments doivent être précédés d'un underscore « _ ».

author

Permet de préciser que le lien pointe vers l'auteur du document courant.

bookmark

Donne un lien permanent vers une section parente à mettre en favoris.

next, prev

Permettent d'indiquer la suite du document courant (pratique pour les articles en plusieurs parties).

nofollow

Indique que l'auteur ne cautionne pas le document lié (en pratique, en SEO (Search Engine Optimization) Google et autres moteurs de recherche ne transfèrent donc pas le « jus » de la page courante vers le lien).

noreferrer

Permet de dire que l'on n'accepte pas que le referrer soit transmis pour ce lien.

prefetch

Indique que la ressource cible doit être mise en cache préalablement.

tag

Indique que le document lié traite d'un sujet en rapport avec le document courant.

Bien entendu, cet article n'a pas vocation à être exhaustif, cependant n'hésitez pas à me faire part de vos suggestions de balises à ajouter ou de cas d'utilisation. Si vous hésitez sur l'usage d'une balise, n'hésitez pas à poser la question dans les commentaires.

Quelles commandes estimez-vous incontournables, pour quoi faire ? N'hésitez pas à me suggérer des ajouts dans les commentaires.

VI - Note de la rédaction de Developpez.com

Nous tenons à remercier **Quentin Busuttil** qui nous a aimablement autorisés à publier son tutoriel : **101 balises HTML5 + exemples**. Nous remercions également **Winjerome** pour la mise au gabarit et **Claude Leloup** pour la relecture orthographique.