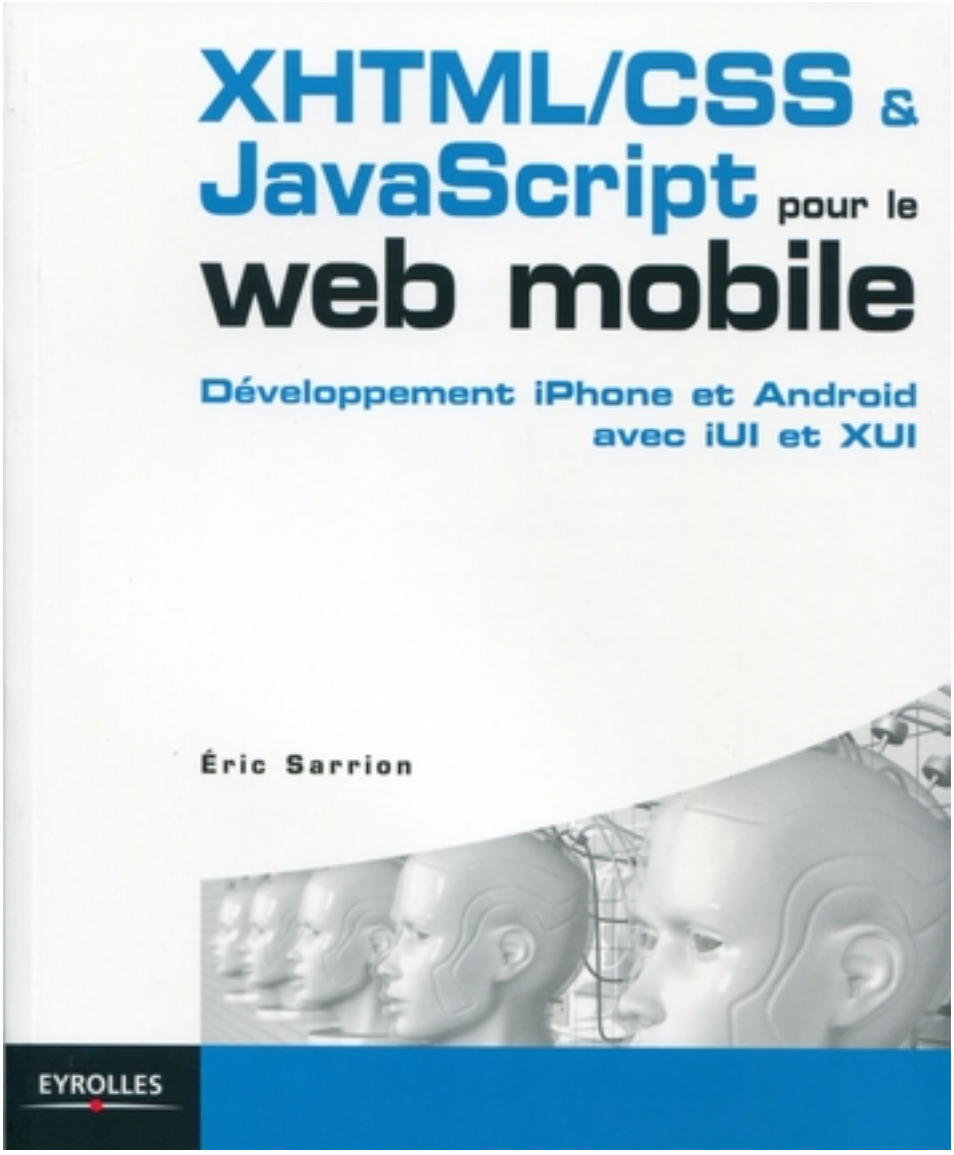
Développement Web pour mobiles Les bases du HTML



par Eric Sarrion (Site Developpez.com)

Date de publication : 23/09/2010

Dernière mise à jour : 13/10/2010

Le langage HTML est le langage de base permettant de construire des pages web, que celles-ci soient destinées à être affichées sur un iPhone / Android ou non. Dans notre cas, HTML sera associé à CSS et JavaScript pour construire des applications pour mobiles, qui seront accessibles sur le web via une URL (applications web), l'AppStore d'Apple ou l'Android Market (applications natives).

N'hésitez pas à nous faire part de votre avis sur cet article :

Préambule

HTML ou XHTML ? Ici c'est pareil !

Nous emploierons ici les termes HTML ou XHTML de façon identique. En effet, même si notre page HTML n'est pas formatée strictement comme l'exige XHTML, le navigateur web sait l'interpréter de façon correcte et restituer l'affichage adéquat.

Notions générales

Une page HTML correspond à un fichier texte pouvant être écrit sous n'importe quel éditeur de texte, l'éditeur le plus simple d'utilisation étant dans ce cas le meilleur.

Le fichier texte contiendra principalement deux types de données :

- les balises HTML, qui permettent d'indiquer ce que l'on désire afficher (un paragraphe, une liste, une table, une image...) ;
- les attributs associés aux balises, qui permettent de préciser les paramètres de la balise (la taille de l'image, la couleur du texte, ...).

Balises

Une balise HTML est entourée des caractères < et >, par exemple pour indiquer un paragraphe. Une balise s'applique aux éléments qui la suivent. Pour indiquer la fin, on utilise la même balise, mais dont le nom est précédé de /.
Donc pour indiquer la fin du paragraphe, on écrira .

Un paragraphe en HTML

```
<p>Voici un paragraphe écrit en HTML</p>
```

Certaines balises ne possèdent pas de contenu. Par exemple, la balise
 permettant d'effectuer un retour à la ligne, n'a pas de contenu. Pour cela, elle s'écrira sous la forme suivante :

Une balise sans contenu

```
<br />
```

Attributs

Une balise utilisée sans attributs produira un fonctionnement standard. Par exemple, le paragraphe précédent aura la forme standard (police de caractères, couleur, *etc.*) associée à la balise employée ici. Si nous souhaitons avoir un paragraphe de couleur rouge (au lieu du noir standard), nous devons l'indiquer dans les attributs de la balise :

Un paragraphe de couleur rouge

```
<p style="color:red">Un paragraphe de couleur rouge</p>
```

Chaque balise a ses attributs propres, mais le plupart des balises possèdent les deux attributs style et class qui sont très utilisés pour définir des styles CSS (voir le prochain chapitre).

Nous avons ici utilisé l'attribut style, dont la valeur est "color:red". La forme générale d'écriture d'un attribut est nom="valeur". Des espaces peuvent figurer de part et d'autre du signe =. La valeur de l'attribut est généralement entourée de ' ou " (simple guillemet ou double guillemet). Ceux-ci peuvent être omis dans le cas où la valeur de l'attribut ne comporte pas d'espace. Dans ce cas, le prochain espace ou le </p> indiquant la fin de la balise, sert au navigateur à connaître la valeur de l'attribut.

Dans le cas précédent, on aurait donc aussi pu écrire :

Pas de guillemets pour les valeurs d'attributs

```
<p style=color:red>Un paragraphe de couleur rouge</p>
```

Alors que ceci aurait été mal interprété :

Un espace dans la valeur d'un attribut (sans guillemets)

```
<p style=color: red>Un paragraphe de couleur rouge</p>
```

Dans ce cas, l'attribut style aurait la valeur color:, tandis que le mot red serait interprété comme le nom d'un nouvel attribut (qui n'existe évidemment pas !). Le paragraphe dans ce cas restera en couleur standard, c'est-à-dire noir.

Pour inclure un espace dans la valeur d'un attribut, il suffit de l'entourer de guillemets :

Un espace dans la valeur d'un attribut (avec guillemets)

```
<p style="color: red">Un paragraphe de couleur rouge</p>
```

Dans le cas où plusieurs attributs sont utilisés dans une balise, chaque couple nom="valeur" est séparé du suivant par au moins un espace, voire un retour à la ligne. L'ordre dans lequel les attributs sont écrits n'a pas d'importance dans le résultat qui sera affiché.

Forme générale d'une page HTML

Une page HTML doit suivre quelques règles précises pour être correctement interprétée par le navigateur de l'utilisateur (dans notre cas le navigateur Safari de l'iPhone ou le navigateur Chrome d'un mobile Android).

Format standard d'une page HTML (fichier index.html)

```
<html>
  <head></head>
  <body>
    Voici une application pour iPhone !
  </body>
</html>
```

La page HTML indiquée ici correspond à un fichier texte dont le nom est par exemple index.html. L'extension du fichier permet de connaître le type de son contenu, ici du HTML !

Cette page HTML commence par la balise <html>, et se termine naturellement par cette même balise (</html>). Deux nouvelles balises sont introduites ici : <head> et <body>.

- La balise <head> correspond à la partie déclarative de la page : quels styles seront utilisés ? Quels fichiers de styles sont à inclure ? Quelle sera la taille maximum de l'écran ? Quel titre aura la page ? *etc.*
- La balise <body> correspond à ce qui sera affiché à l'utilisateur : le texte, les images, les listes, *etc.*

Visualisation d'une page HTML

La page HTML écrite précédemment peut être visualisée dans n'importe quel navigateur actuel : Internet Explorer, Firefox, Google Chrome, ou encore Safari.

Si vous souhaitez visualiser cette page depuis votre ordinateur personnel (PC, Mac...), il suffit de faire glisser le fichier dans votre navigateur. Pour l'afficher sur votre iPhone / Android, il faut d'abord installer un serveur web sur votre ordinateur personnel, par exemple un serveur PHP (ou tout autre serveur). Dans cet ouvrage nous avons utilisé deux types de serveurs :

- AppServ sous Windows permettant d'afficher des pages écrites en PHP (et bien sûr aussi HTML) ;
- MAMP sous Mac permettant d'afficher des pages écrites en PHP (et bien sûr aussi HTML) ;
- Ruby on Rails (Windows ou Mac) permettant d'afficher des pages écrites en Ruby (et également en HTML).

Des serveurs Java, Microsoft .Net, *etc.*, fonctionneraient de manière similaire.

Visualisons notre page HTML en ouvrant, par exemple, Safari sur un iPhone et en tapant l'adresse de notre serveur local dans la barre d'adresse :

Une page HTML affichée sur l'iPhone

Le texte inclus dans la balise <body> s'affiche, mais dans une taille très petite et pas du tout adaptée à l'iPhone ! Ceci car notre page HTML, sauf indication contraire, est écrite pour être affichée dans un navigateur d'un ordinateur de bureau, et pas pour le navigateur d'un téléphone portable !

Il suffit d'ajouter la ligne suivante dans la partie <head> du code HTML pour afficher correctement la page HTML :

Avoir la taille de la page adaptée à la taille de l'écran

```
<meta name="viewport" content="user-scalable=no,width=device-width" />
```

Une fois cette balise <meta> insérée dans la partie <head>, l'affichage est maintenant :

Adaptation de l'affichage à la taille de l'écran

Par la suite, nous conserverons cette balise <meta> dans la partie <head> de chacune des pages HTML que nous écrirons.

ASTUCE : et pour Android, utile ou pas ? *Android n'a peut être pas besoin de cette balise <meta>. Toutefois, nous l'inclurons dans toutes nos sources de façon à être compatible pour iPhone et Android.*

Le texte

Nous avons précédemment utilisé une balise <p> pour afficher un paragraphe de texte. D'autres balises existent permettant d'afficher du texte sous différentes formes :

- <p> : affiche un paragraphe. Deux paragraphes qui se suivent seront donc espacés verticalement à l'écran ;
- <b> : met le texte en gras (bold) ;
- <i> : met le texte en italique ;
-
 : insère un retour à la ligne. Cette balise ne possède pas de contenu, d'où sa forme d'écriture ;
- <hr /> : insère une ligne de séparation horizontale. Cette balise ne possède également pas de contenu ;
- <h1> : affiche un titre de paragraphe. Pour différencier différents niveaux de titres, on utilise les autres balises <h2>, <h3>, <h4>, <h5> et <h6>. Plus l'indice qui suit **h** augmente, plus le titre est petit. L'indice qui suit **h** symbolise donc l'imbrication ou le niveau du titre.

En plus des balises décrites ci-dessus, il existe des codes spéciaux permettant d'écrire certains caractères. Ces codes spéciaux s'appellent des entités HTML. En voici quelques unes parmi les plus utilisées :

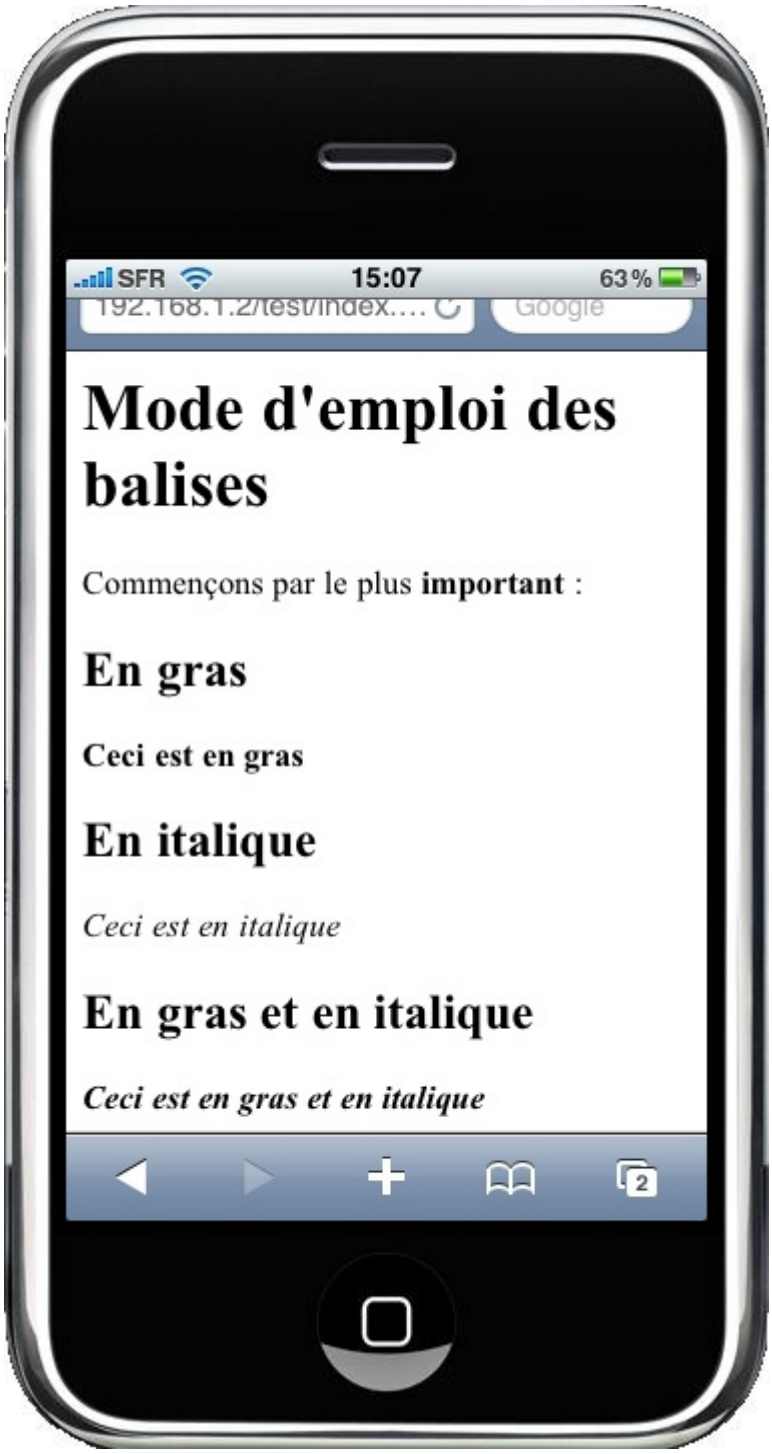
-   : insère un espace dit insécable entre deux mots. En effet, le navigateur ne tient pas compte du formatage du texte que nous tapons dans notre fichier HTML. Les retours à la ligne effectués par l'appui sur la touche **Return** ne sont pas pris en compte (lors de l'affichage), pas plus que les espaces insérés entre les mots (au delà du premier espace). On a vu que la balise
 permettait d'insérer un retour à la ligne. L'entité   permet d'insérer un espace supplémentaire entre deux mots ;
- < : le caractère < est un caractère réservé en HTML, signifiant le début ou la fin d'une balise. Pour afficher le signe < à l'écran, on utilise donc l'entité < ;
- > : idem que <, mais pour afficher le caractère > (qui sinon serait interprété comme la fin de la balise).

Voici un petit exemple contenant quelques balises HTML :

Quelques balises HTML

```
<html>
  <head>
    <meta name="viewport" content="user-scalable=no,width=device-width" />
  </head>
  <body>
    <h1>Mode d'emploi des balises</h1>
    <p>Commençons par le plus <b>important</b> : </p>
    <h2>En gras</h2>
    <b>Ceci est en gras</b>
    <h2>En italique</h2>
    <i>Ceci est en italique</i>
    <h2>En gras et en italique</h2>
    <b><i>Ceci est en gras et en italique</i></b>
  </body>
</html>
```

On peut voir l'affichage produit :



Utilisation de balises dans la page

Les images

Les images s'affichent par l'intermédiaire de la balise <img>. Celle-ci ne possède pas de contenu, donc on l'utilise sous la forme <img />.

Des attributs sont nécessaires pour afficher l'image :

- src : permet d'indiquer l'adresse Internet où se trouve l'image. Cet attribut est obligatoire (sinon on ne peut pas afficher l'image !).

Si l'image se trouve sur le serveur (ce qui est souvent le cas), l'adresse peut être relative à la page HTML qui affiche l'image. On n'indique donc pas dans ce cas, le chemin complet d'accès à l'image (commençant par http://), mais le chemin relatif (par exemple images/img1.png si l'image se trouve dans un répertoire images situé au même niveau que la page HTML).

Si l'image n'est pas sur le serveur, le chemin complet commençant par http:// sera nécessaire ;

- width : cet attribut permet d'indiquer une largeur à l'image. Si l'image est plus petite que la valeur indiquée, l'image est agrandie, sinon elle est rétrécie ;
- height : idem que width, mais pour la hauteur de l'image.

Par exemple, pour afficher l'image img1.png sur une hauteur de 200 pixels :

Une image de 200 pixels de hauteur

```
<img src=img1.png height=200 />
```

On remarque que l'image est redimensionnée en proportion de la hauteur indiquée, sauf si l'attribut width est également indiqué.

ASTUCE : pour ne pas agrandir trop ? max-height et max-width !
Les attributs width et height peuvent être remplacés par les propriétés CSS max-width et max-height qui permettent de ne pas agrandir l'image si la taille est plus petite que celle indiquée (voir chapitre suivant).

Les liens

Un lien correspond à l'utilisation de la balise <a>. Le lien permet de naviguer vers une autre page HTML, voire la même. Pour cela on utilise l'attribut href dans la balise <a> :

- si href commence par #, cela signifie que l'on désire atteindre une partie de la même page HTML. Le nom qui suit href (par exemple href=#partie1) correspond à un endroit du code dans la page HTML identifié par ce nom. Ce mécanisme sera utilisé par la bibliothèque iUI que nous étudierons dans la partie suivante ;
- sinon (href est indiqué mais ne commence pas par #), cela référence une autre page HTML. Lorsque le lien est cliqué, la page est affichée.

Des valeurs particulières de href sont possibles, permettant des actions spéciales :

- mailto:une_adresse_email : permet de passer en mode d'écriture d'un nouveau mail qui sera adressé à l'adresse email indiquée ;
- tel:un_numero_de_telephone : propose de téléphoner au contact indiqué par le téléphone ;
- sms:un_numero_de_telephone : idem que tel, mais pour envoyer un sms.

COMPATIBILITÉ : et avec les sites web classiques ?

Les valeurs tel et sms dans l'attribut href ne sont disponibles que sur iPhone / Android, alors que mailto peut aussi s'utiliser dans des pages web classiques.

Voici un code HTML permettant d'envoyer un mail, de téléphoner et d'envoyer un sms.

Envoyer un mail, un sms ou téléphoner.

```
<html>
  <head>
```

```
 <meta name="viewport" content="user-scalable=no,
  width=device-width" />
  </head>

  <body>
    <a href=mailto:ericsarrion@gmail.com >
   Envoyer un mail </a><br /><br />
    <a href=tel:0612345678 >Téléphoner </a><br /><br />
    <a href=sms:0612345678 >Envoyer un sms </a>
  </body>
</html>
```

Lors de l'exécution de cette page HTML, on peut remarquer que le clic sur le lien sms ouvre une nouvelle fenêtre, mais ne retourne pas à notre application après l'envoi du sms, au contraire des liens mailto et tel qui permettent le retour à celle-ci.

Les listes

Les listes sont un élément essentiel pour afficher des informations sur l'iPhone / Android. En effet, de part leur petit affichage, il va être nécessaire de présenter la plupart des informations sous forme de liste. Un élément de liste prendra en général la largeur de l'affichage, tandis que les éléments de liste seront placés les uns sous les autres.

HTML permet d'afficher les listes au moyen des balises <ul> et <li>. Il existe également la balise <ol>, mais celle-ci ne sera pas utilisée dans cet ouvrage.

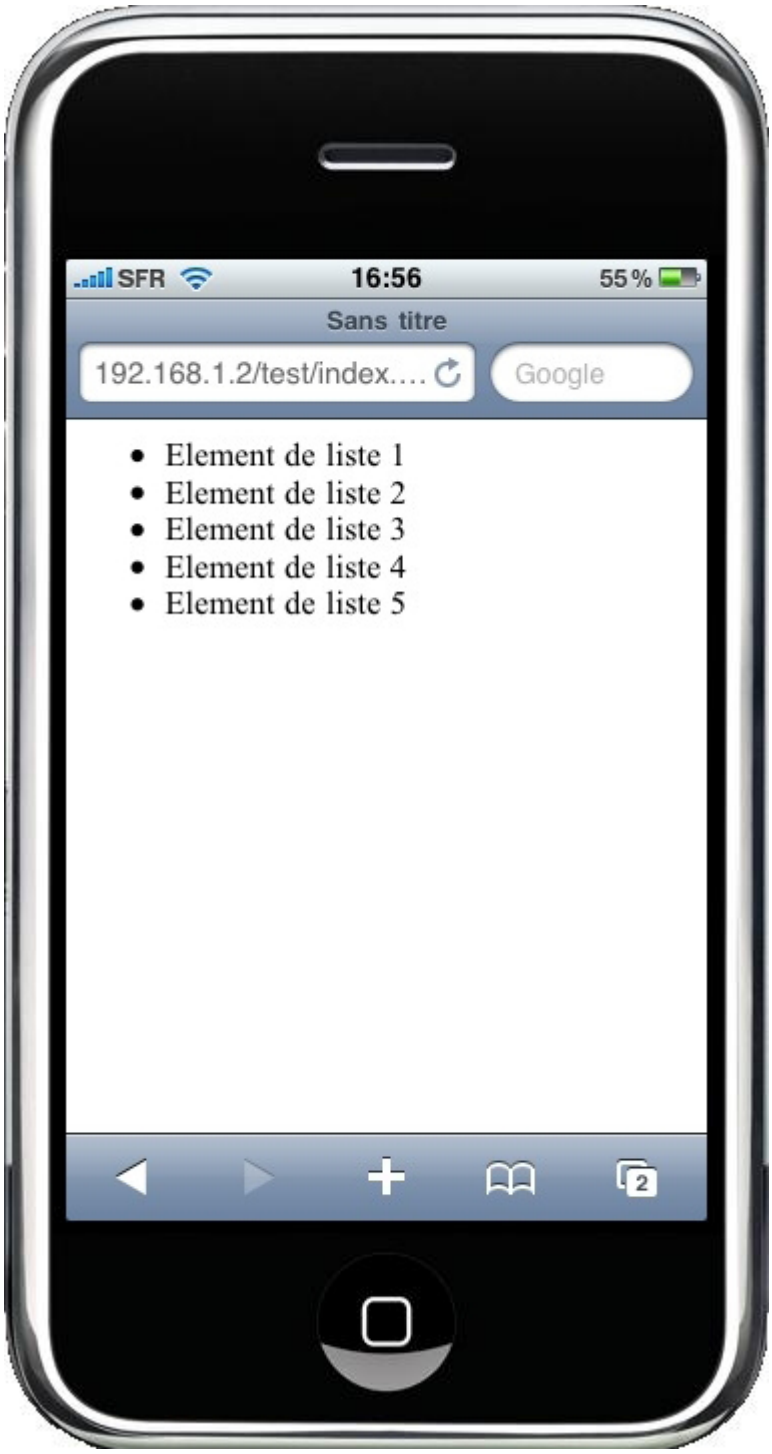
- : permet d'indiquer le début d'une liste, qui se terminera donc par . signifie *unordered list*, ce qui se traduit par liste non ordonnée, au contraire de <ol> qui signifie *ordered list* (liste ordonnée).
- : permet d'indiquer un élément dans la liste, qui correspondra à une ligne de l'affichage. Cette ligne pourra contenir d'autres éléments HTML (une image, un texte, un lien, ...).

Voici une liste composée de 5 éléments :

Liste d'éléments

```
<html>
  <head>
    <meta name="viewport" content="user-scalable=no, width=device-width" />
  </head>
  <body>
    <ul>
      <li> Element de liste 1 </li>
      <li> Element de liste 2 </li>
      <li> Element de liste 3 </li>
      <li> Element de liste 4 </li>
     <li> Element de liste 5 </li>
    </ul>
  </body>
</html>
```

Elle s'affiche comme ceci dans l'iPhone :



Affichage d'éléments de liste

L'affichage est dans le style d'une page HTML tel qu'on le rencontre sur la plupart des sites Internet, mais n'est pas tellement dans le style iPhone ou Android ! Nous verrons que l'utilisation des styles CSS permet d'obtenir un meilleur affichage

Les blocs

Comme les listes, les blocs sont une composante essentielle de l'affichage sur iPhone / Android. Ils correspondent aux éléments et de HTML :

- : permet d'afficher un bloc qui ne pourra être mitoyen que de l'élément situé au dessus et / ou au dessous, mais n'aura pas de mitoyenneté sur les côtés droit ou gauche ;
- : à l'inverse du , il pourra avoir une mitoyenneté sur les côtés droit et / ou gauche, en plus du haut et du bas de l'élément.

On voit donc que les éléments ne pourront s'afficher que les uns sous les autres, tandis que les éléments pourront s'afficher les uns à côté des autres.

De plus, un élément peut contenir d'autres éléments ou (qui s'afficheront selon les mêmes règles), tandis qu'un élément ne pourra pas contenir d'éléments (un est sensé être un bloc occupant toute la ligne, donc il est difficile d'en mettre deux ou plus par ligne...).

Voici un exemple de code HTML contenant ces deux éléments :

Elements div et span

```
<html>
  <head>
    <meta name="viewport" content="user-scalable=no, width=device-width" />
  </head>
  <body>
    <div>Un premier div</div>
    <div>Un second div</div>
    <div>
      <span>Du texte</span>
      <span>contigu à celui-ci</span>
    </div>
  </body>
</html>
```

Cela s'affiche de la façon suivante :

Utilisation de div et span

Les éléments se mettent bien les uns sous les autres, tandis que les éléments se mettent les uns à côté des autres. Pour que deux éléments soient l'un en dessous de l'autre, il faut soit les mettre dans deux éléments différents, soit casser la séquence en insérant par exemple un élément
 entre les deux.

La suite ?

Vous avez aimé cet article et vous voulez avoir la suite de l'histoire ? Elle est dans mon livre **XHTML / CSS et JavaScript pour le Web mobile**.

Remerciements

Un grand merci à **eusebe19** pour sa relecture.