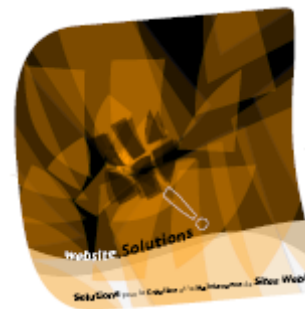


• apprendre le langage XHTML :: pas à pas

Vous voulez mieux maîtriser les étapes du processus de fabrication de votre Site Web? Vous voulez savoir comment est interprété et affiché tout le contenu de vos pages? Regardez plutôt à la source! Et explorez le monde du Web dans ses rouages en mode texte! En effet, tout ce que vous consultez dans votre butineur est l'interprétation d'un code quasi universel: le HTML pour 'HyperText Markup Language', comprendre son fonctionnement et sa syntaxe vous permettra sans doute de mieux appréhender la conception de votre Site!



[sommaire]

1. [Préambule](#)
2. [Texte](#)
3. [Paragraphe](#)
4. [Image & autres](#)
5. [Lien](#)
6. [Liste](#)
7. [Tableau](#)
8. [En-tête & autres](#)
9. [Conclusion](#)

>>> Préambule

:: D'où ça vient? L'informatique est riche de langages pour *communiquer* avec les machines, et ce à différents niveaux. Le HTML qui est considéré comme le langage qui est à l'origine de l'essor du Web en mettant en place le principe de lien hypertexte, est en fait un descendant modifié (et nettement simplifié) d'un langage complexe: le SGML. Pour l'interprétation du code HTML, des logiciels sont développés depuis plusieurs années, ce sont les [butineurs](#) comme par exemple: Netscape, Internet Explorer, Opera, Firebird, Konqueror, Safari, etc.

:: Quelle structure? En tant que descendant du SGML, le HTML a conservé grossièrement son architecture en arborescence représentée par des balises (qui s'ouvrent et se ferment). Cependant la simplification a été un peu trop brutale, aussi préfère-t-on le langage XHTML qui est dit "bien formé" car il respecte intégralement la grammaire primitive du SGML.

en HTML	en XHTML	
<BaliseEncadrante>"Ce qui est écrit dans cette balise est faux"</BaliseEncadrante>	<BaliseEncadrante>"Ce qui est écrit dans cette balise est faux"</BaliseEncadrante>	Une balise est une sorte de 'mot' précédé et suivi d'un signe '<' et '>' respectivement.
<BaliseSolitaire>	<BaliseSolitaire />	Quand on veut arrêter le champ d'action d'une balise, on doit la répéter en insérant
<grave>lala<fort>lala</grave>la</fort>	<grave>lala</grave><fort><grave>lala</grave>la</fort>	entre '<' et le 'mot': le signe '/'. Pour se conformer à la règle, les balises qui n'encadrent rien sont affublées du signe '/' entre le 'mot' et '>'. Les balises peuvent s'imbriquer mais

jamais se croiser!

Le langage XHTML qui est donc une évolution du HTML (pas seulement à cet égard) est toujours autant conçu pour pouvoir être écrit *à la main*, à condition de connaître ce à quoi correspondent les balises employées!

:: Comment se construit ce squelette? Pour créer votre premier fichier Web à la main, lancez un [éditeur texte](#). Le document doit être vierge avant de commencer. On va mettre en place le squelette de page Web. Cet aspect peut sembler rébarbatif, mais il est réellement nécessaire pour indiquer à votre butineur qu'il est en train de lire un document web et pas un simple fichier texte.

- Inscrivez la balise: **<html>** puis **</html>**
- Le HTML vous permet d'inclure une balise d'en-tête pour y laisser diverses informations, c'est **<head></head>**
- Enfin, vous devez insérer la balise **<body>** (et la refermer, évidemment), qui représente le corps de la page. C'est là que votre page va contenir le texte, les tableaux, images ou tout autre objet...

```
<html>
<head>
</head>
<body>
</body>
```

Voilà un exemple de structure basique de page Web.

[connexe]

- [CSS: Cascading Style Sheet](#)
Feuille de style en cascade
- [XML, Java, PHP, MySQL, etc.:](#)
des technologies puissantes pour les sites Web aux fonctionnalités avancées.
- [Recommandation:](#)
documentation exhaustive et officielle publiée par le W3C portant sur les diverses technologies du Web (non propriétaires).

[vocabulaire]

- **Butineur:** logiciel servant à explorer le Web
- **Editeur texte:** logiciel servant travailler un fichier en texte brut (sans formatage)

[logiciel]

- [Butineur](#)
- [Editeur texte](#)

</html>

:: Qu'est-ce que ça donne? Enregistrez votre document (en texte brut: aucun formatage) avec l'extension .htm ou .html

- *L'utilisation de l'extension .xhtml peut induire des difficultés sous certains butineurs.*

Lancez votre butineur, puis entrez l'adresse de votre fichier. Vous pouvez recourir à la commande 'Ouvrir'. Dès lors le fichier ouvert, le chargement et l'affichage de la page s'effectue. Vous devriez obtenir une page blanche. Un peu décevant? En fait non! Votre butineur a interprété la structure de votre fichier, et il n'y a autre information *utilisateur* à afficher.

[Cette procédure d'ouverture de fichier n'est pas à recommencer à chaque fois, si vous avez modifié (et enregistré) votre fichier et que vous retournez au butineur, cliquez plutôt sur l'icône Actualiser.]

Maintenant que tout est en place, on va détailler un ensemble de balises: en donnant leur signification et leur emploi.

>>> Texte

:: Comment afficher un peu de texte? Comme échantillon de test, on va utiliser une célèbre phrase paradoxale: "Je rase tous ceux qui ne se rasent pas eux-mêmes et seulement ceux-là". On peut lui affecter un formatage en changeant certains éléments comme la police, la taille, la couleur. On utilise la balise ****. Tester en insérant la phrase de test produira la phrase avec le formatage par défaut de votre butineur. Pour aller plus loin, on va ajouter des attributs à la balise ouvrante, ainsi pour changer...

- la police: tapez face="Nom de la police"
- la taille: tapez size="n", n est un entier compris entre 1 et 7
- la couleur: tapez color="#RRVBB", avec RougeRougeVertVertBleuBleu, et le tout en hexadécimal s'il vous plaît!

*[Le hexadécimal est un système à 16 *chiffres* contrairement au décimal qui en compte 10. Les chiffres du hexadécimal sont donc (dans l'ordre croissant): {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f}. Pour obtenir le code couleur, on inscrit en lieu et place de RR,VV,BB variant de 00 à ff, où 00 correspond au minimum de la couleur et ff au maximum. Donc du coup #000000 = noir et #ffffff = blanc. Mais pour ne rien cacher, il existe un autre moyen moins précis mais plus pratique, tapez color="nom de la couleur en anglais", c'est à dire black, white, red, green, blue, orange, yellow, purple, pink... et vous obtiendrez ce que vous voulez !o)]*

:: Et les autres? Pour préciser le format des caractères (gras, italique, souligné, etc.), il existe différentes de balises bien pratiques.

- pour souligner: **<u></u>** (u pour *underline* en anglais),
- en gras: **** (b comme *bold*),
- en italique: **<i></i>**,
- en indice: ****,
- en exposant: ****,
- barré **<s></s>** (s pour *stroke*).

<html>

<head>

</head>

<body>

<u> "Je rase tous ceux qui ne se rasent pas eux-mêmes et phrases, du code ou n'importe quoi seulement ceux-là." </u>

<i> "Je rase tous ceux qui ne se rasent pas eux-mêmes et seulement ceux-là." </i>

<!-- afficher un commentaire -->

</body>

</html>

On a introduit au passage un commentaire (<!-- puis -->). Tout ce qui se trouve à l'intérieur: que vous mettiez des chiffres, des d'autre (même sur plusieurs lignes) butineur!

- *On ne peut pas imbriquer les balises de commentaire*

:: Et les caractères spéciaux? Pour ce qui est de l'utilisation des accents et du ç, tout va bien. Pour les symboles et caractères exotiques, il faudra passer par un code particulier qui est tenu à jour par une organisation unique appelé: unicode. Chaque caractère est muni d'un code (variant selon le standard d'encodage choisit) dont on va se servir pour l'insérer dans la page Web. Par exemple, pour afficher le symbole *Copyright*: © on doit écrire (accollé) '&' '#' '169' ';'. Des ressources bien précises sont [disponibles](#).

Qui dit texte, dit paragraphe...

la ligne, ou ajouter des lignes, vous pouvez utiliser la balise **
. Elle marque un retour chariot simple et n'indique rien de plus. Pour faire apparaître un semblant de structure, on peut employer deux balises complémentaires **<div></div> et ****. La première marque le début d'un bloc paragraphe et ajoute automatiquement un retour à la ligne, la deuxième marque seulement la présence d'un élément particulier mais ne modifie pas l'affichage. Cela paraît à ce niveau inutile, mais elles sont très appréciées quand on les utilise conjointement les [feuilles de styles \(CSS\)](#).

[On peut déjà affecter un attribut à <div> d'alignement par: align="alignement souhaité" c'est à dire: left, center, right, justify.]

:: Et les citations? Une balise spéciale a été créée pour l'ajout de citation, c'est **<blockquote></blockquote>**. Le texte compris dans les balises est mis en valeur par un retrait à gauche, à droite, en haut et en bas. Si vous essayez d'imbriquer ce type de balises, le texte contenu sera décalé d'autant à chaque fois.

:: Et les lignes de séparation? On peut aussi apprécier la séparation de groupes de paragraphes par des lignes horizontales. Une balise solitaire nommée **<hr />** y est dédiée. Par défaut elle occupera toute la largeur possible, on peut cependant ces paramètres:

- d'abord sa taille/épaisseur avec `size="épaisseur souhaité en pixels"`,
- ensuite sa largeur via `width="largeur en pixels ou pourcentage"`,
- puis son alignement `align="ce que vous voulez"`, c'est à dire left, right, center,
- enfin cerise sur le gâteau pour annuler l'effet de relief, avec `noshade`.

```
<html>
  <head>
  </head>
  <body>
    <font color="red"> "Je rase tous ceux qui ne se rasent pas eux-mêmes et
seulement ceux-là." <br /><!-- Retour simple à la ligne --></font>
    <font size="1"> Mais alors, le barbier est-il rasé? <br /><br /><br />
<!-- Saut de deux (petites) lignes --></font>
    <blockquote><!-- Utilisation pour un retrait --> Text décalé par rapport
aux précédentes phrases. </blockquote>
    <div align="center"> Le texte qui est affiché est centré, si si! </div>
    <hr align="center" width="33%" />
  </body>
</html>
```

Après le textuel, le visuel...

>>> Image & autres

:: Comment afficher une image? La méthode est unique en XHTML, elle consiste en une balise solitaire **** dans laquelle on indique différentes informations:

- la source de l'image: `src="lechemin/devotre/image.extension"`

[L'image se charge normalement, mais on préfère ajouter deux autres éléments: la largeur et la hauteur pour faire apparaître rapidement la présentation de la page durant son chargement...]

- insérez `width="largeur de l'image"` et `height="hauteur de l'image"`,
- on peut ajouter, si l'image est un peu volumineuse (en octets), une petite bulle de texte: `alt="bulle de texte"`

:: Comment mieux l'intégrer au document? Vous vous rendez vite compte qu'une image posée là à côté du texte impressionne peu... Vous pouvez utiliser un alignement particulier: `align="alignement particulier"`, c'est à dire top: middle, bottom, left, right (mais aussi: texttop, absmiddle, baseline, absbottom).

*[Petite parenthèse: de retour sur les paragraphes,
 peut prendre l'attribut: clear="l'alignement de l'image", c'est à dire left, right, all; afin de supprimer l'alignement de l'image]*

Mais cette technique soulève un léger problème, le texte semble un peu trop collé. Une parade existe: `hspace="espace horizontal"` et `vspace="espace vertical"` (en pixels). Autre souci esthétique rencontré fréquemment, si vous liez cette image, elle va se parer d'une bordure de couleur, pour la supprimer: `border="largeur de la bordure, ici 0"`.

```
<html>
  <head>
  </head>
  <body>
```

[Bien que le détail n'en sera pas fait ici, sachez que la balise <map></map> permet de créer des zones particulières de liens]

```


<font> Ceci est une petite image accompagnée d'un peu de texte... </
font>
<br clear="all"/>
<font> Voilà une phrase solitaire cette fois-ci... </font>
</body>
</html>

```

:: Et les autres médias? On dispose d'autres ressources que les images, à savoir le son et la vidéo même si à cause des débits limités ces ressources restent accessibles indirectement. On ne doit pas oublier non plus les Applets Java ou Python, etc. qui peuvent apporter une valeur ajoutée également. Cela étant multiplier les ressources apporte des problèmes notamment de compatibilité. Une balise appelée **<object></object>** offre d'intéressantes possibilités. On peut par exemple imbriquer plusieurs types d'objets pour être à peu près certain qu'un d'eux sera compris par le butineur (ou les plug-ins) de l'internaute; mieux, elle offre une syntaxe commune pour les différents types d'objets. On peut entre autres définir trois attributs:

- le titre de l'objet utilisé, title="titre de l'objet"
- la source, data="chemin/del/objet.extension"
- son type (MIME, c'est à dire à quelle famille il appartient), type="type de l'objet" à choisir parmi: image/png, image/jpeg, image/gif, image/tiff, image/svg+xml, etc. ou application/mpeg, application/mp3, etc.

L'essence du Web: le lien hypertexte...

>>> Lien

:: Quel enjeu? Le concept de lien hypertexte a été la révolution qui a fait que le Web a pris son essor: en permettant un accès rapide par simple clic à une nouvelle zone d'informations.

:: Quelle syntaxe? Pour créer un lien, il suffit d'inclure entre les balises **<a>** l'objet que l'on veut (texte, image, etc.). Ensuite dans la balise, via href="protocole:chemin", vous devez préciser le protocole puis le chemin du lien (Internet regroupe une foule de protocoles dédiés aux différents types de liens).

- Pour créer un lien envoyant un courrier, mailto:adresse@serveur.extension
- Pour pointer vers un forum de discussion, news:adresse.newsgroup
- Pour lier un site (ou une page, ou objet d'un site), http://www.serveur.extension
 - *N'oubliez surtout pas http:// s'il s'agit d'un lien extérieur à votre site*
- Il existe différents protocoles pour le téléchargement de fichier
 - Pour le transfert traditionnel de fichiers (via le FTP), ftp://ftp.serveur.extension
 - Pour les échanges entre utilisateurs par l'intermédiaire des logiciels de *peer to peer (p2p)*, de nouveaux noms sont apparus comme magnet:, gnutella:, ed2k:, etc. (la liste est loin d'être exhaustive)
- Pour pointer vers un objet local (et qui le reste!), file:///lechemin/kivabien/dela/page.html

[Certains butineurs et/ou système d'exploitation ne respectent pas cette syntaxe pourtant standardisée]

- Pour lancer une fonction en EcmaScript ou JavaScript, mafonction;

Par ailleurs, pour le fonctionnement des cadres, vous pouvez donner la cible du lien via target="cible du lien" c'est à dire le nom, mais on y reviendra. Cependant si vous souhaitez ouvrir une nouvelle fenêtre par un lien, vous pouvez utiliser target="new".

:: Une autre facette? Cependant, la balise **<a>** ne sert pas seulement à rediriger vers une nouvelle page (ou objet), elle peut aussi pointer vers un secteur particulier. Vous avez certainement déjà rencontré ce type de lien, en cliquant dessus, l'écran se déplace à un endroit précis de la page. Pour cela il faut définir au sein de la page, un repère via: name="nom du repère". Ensuite dans le lien pointant vers ce repère, tapez ceci href="#nom du repère". Le caractère '#' indique au butineur qu'il doit se déplacer jusqu'à ce secteur. Cependant, si vous souhaitez créer un lien vers ce secteur à partir d'une autre page, il suffit d'indiquer href="lechemin/dela/page.html#nom du secteur"

- *#nom du secteur doit être placé après l'extension!*

```

<html>
<head>
</head>
<body>

```

[Pour accéder à une page de votre site, il est déconseillé de taper le chemin complet (ça ne fonctionnerait qu'en local), il faut

```

<a href="http://www.google.com/" target="new"> Rechercher via Google!
<a>
  <a href="#secteur situé plus loin"> Pointe vers un secteur de la page <a>
<!-- Pour s'éloigner un peu justement! -->
<br /><br /><br />
<br /><br /><br />
<br /><br /><br />
<br /><br /><br />
<br /><br /><br />
<br /><br /><br />
<br /><br /><br />
<br /><br /><br />
  <a name="secteur situé plus loin"> Vous voici en lieu et place du secteur
d'arrivée! <a>
  </body>
</html>

```

préférer les chemins relatifs, prenons un exemple. Votre page contient un lien une autre situé un dossier plus loin, le lien peut très bien s'écrire comme ça . Pour revenir à la page précédente en créant un lien, utilisez ceci './' comme suit . Enfin, cette technique du parcourt des dossiers et fichiers est universelle et s'applique partout sur le Web et dans les autres balises.]

Vous avez justement une liste de liens... mais comment l'afficher?

>>> Liste

Pour énumérer ou organiser une progression logique, utiliser les listes est bien pratique. Il en existe de plusieurs sortes.

- D'abord ce que l'on appelle communément les puces. Cette liste commence toujours par **** (pour *Unordered List*) pour finir par ****. En fonction de l'imbrication, le dessin de la puce varie. Sachez cependant que vous pouvez le préciser de cette manière: type="type de la puce" à remplacer par disc, circle ou square. Les éléments sont introduits entre **** et ****.
- Mais on peut aussi préférer les numérotations, dans ce cas utilisez **** (pour *Ordered List*). L'imbrication détermine également la numérotation que l'on peut préciser par: type="type de numérotation" c'est à dire soit A, a, I, i, l. Les éléments sont introduits entre **** et ****.
- Pratique aussi, même si l'assimilation avec une liste est moins évidente, **<dl></dl>** permet d'énumérer des éléments et leur définition. Pour ce faire inscrivez le mot dans **<dt>** et sa définition dans **<dd>**.

```

<html>
  <head>
  </head>
  <body>
    <ul><!-- Liste a puce simple -->
      <li>Elément de la liste</li>
      <ul type="square"><!-- Liste imbriquée -->
        <li>Elément de la liste</li>
      </ul>
    </ul>
    <ol><!-- Liste numérotée -->
      <li>Premier élément de la liste</li>
      <li>Deuxième élément de la liste</li>
    </ol>
    <dl><!-- Liste de définition -->
      <dt>Elément</dt>
      <dd>définition</dd>
    </dl>
  </body>
</html>

```

Pour ne pas surcharger l'exemple de code, chaque bloc de liste contient au plus deux éléments, ce qui dans la réalité est rarement le cas. Aussi, il faut attacher de l'importance au retrait au niveau des balises pour laisser explicite la structure des listes...

[Ceci n'apporte qu'une lisibilité supplémentaire, mais ne modifie aucunement le rendu de la page web.]

Après les listes... les tableaux!

>>> Tableau

:: Un tableau? Pour celui qui découvre ce langage, les tableaux n'ont pas tout à fait la signification *réelle*. En fait, on appelle tableau tout ce qui comporte des cases mais pas forcément de bords bien visibles. Ils peuvent en effet être utilisés (outre leur usage classique) comme ossature pour l'apparence d'une page Web, permettant des présentations nettement plus complexe que le vulgaire retour à la ligne... Si vous ne suivez pas vraiment, vous allez comprendre.

- *cette notion est donc très large!*

:: Quelle syntaxe? Un tableau commence et se finit par `<table></table>`. Tout le monde sait qu'un tableau comporte une (ou plusieurs) ligne et une (ou plusieurs) colonne. Ici, on commence toujours par les lignes `<tr></tr>`, pour finir par les colonnes `<td></td>` (une habitude de mathématicien, sans doute). Pour ce qui ne connaissent pas les tableaux, le croisement ligne/colonne s'appelle case, certes, mais surtout cellule. Avant de poursuivre voilà comment ça s'organise...

```
<html>
  <head>
  </head>
  <body>
    <table><!-- Construction du tableau -->
      <tr><!-- Construction de la ligne -->
        <td> Je suis une cellule </td>
        <td> Et moi aussi! </td>
      </tr>
    </table>
  </body>
</html>
```

On va maintenant affecter des attributs aux différentes balises:

- En ce qui concerne `<table>`, on peut définir:
 - la largeur de sa bordure, via `border="x"` où `x` égale une certaine valeur en pixels.
 - la largeur du tableau: `width="x"` où `x` est un nombre ou un pourcentage,
 - l'espacement et l'ajustement des cellules: `cellspacing="x"` `cellpadding="y"`, respectivement où `x`

et `y` sont des nombres (petits).

- Ca va très vite pour `<tr>` car seul l'alignement horizontal et vertical sont à voir:
 - `align="alignement horizontal"`, c'est à dire `left`, `right`, `center`;
 - `valign="alignement vertical"`, soit `top`, soit `middle/center`, soit `bottom`.
- Pour `<td>` c'est plus fourni. Comme précédemment, vous pouvez déterminer
 - l'alignement vertical et horizontal,
 - ainsi que la largeur et la hauteur (comme pour `<table>`).
- De plus, cette notion est parfois difficile à assimiler/visualiser, on peut fusionner soit horizontalement, soit verticalement. Pour ce faire, inscrivez `rowspan="nombre de cellules fusionnées à l'horizontale"` et/ou `colspan="nombre de cellules fusionnées à la verticale"`.

```
<html>
  <head>
  </head>
  <body>
    <table width="500" border="1"><!-- Construction du tableau de 500px de
    largeur avec une bordure -->
      <tr align="center"><!-- Construction de la ligne alignée au milieu -->
        <td height="20"> Je suis une cellule dont la hauteur vaut 20px </
        td>
        <td> Et, moi je suis une cellule qui essaie de chanter: lala la laaaa! </
        td>
      </tr>
      <tr align="right"><!-- Construction d'une nouvelle ligne -->
        <td height="300" colspan="2"> Je suis une grande cellule qui occupe
        toute la place </td>
      </tr>
    </table>
  </body>
</html>
```

On peut aussi configurer l'aspect général du tableau:

- l'espacement des cellules via, `cellspacing="la valeur (petite)"`
- le retrait à l'intérieur des cellules, `cellpadding="une autre (petite) valeur"`

:: Et les couleurs? Il est possible d'inclure une image de fond dans un tableau, placez vous dans **<table>** et insérez `background="lechemin/devotre/image.extension"`. Vous pouvez appliquer une couleur de fond au tableau mais aussi pour chaque cellule indépendamment. Il suffit d'inscrire `bgcolor="couleur souhaitée"`, c'est à dire soit des valeurs hexadécimales ou des noms (anglais) de couleurs comme expliqué pour le texte...

Derniers paramètres... dans l'en-tête essentiellement!

>>> En-tête & autres

:: Comment fournir des informations étendues? L'en-tête du document regroupe en fait un ensemble d'informations qui ne sont pas directement affichées. On les trouve **uniquement** entre les balises **<head>** et **</head>** C'est ce qu'on appelle généralement les *meta data* ou méta données. On peut y inscrire un peu ce qu'on veut, mais prioritairement le nom de l'auteur, la date de création, les mots clefs, une description, etc. selon la syntaxe `'name="type de renseignement" content="contenu du renseignement"`, à l'intérieur de balises solitaires **<meta />**. On peut aussi donner le nom de la page (qui apparait dans la fenêtre du butineur), tout simplement via la balise **<title></title>**.

:: Comment mieux identifier les documents? Ici, on sort du corps de l'arborescence du fichier xhtml... Avant la balise initiale **<html>**, il faut inscrire ceci: `<?xml version="1.0" encoding="ISO-8859-1" ?>` ainsi que `<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">`. Il s'agit d'une identification à la famille de technologie XML (une simplification et optimisation du SGML fondateur) puis d'une déclaration du type de document lu: on peut lire qu'il s'agit du langage XHTML version 1.1. Enfin, à l'intérieur de la balise **<html>**, on peut insérer cette instruction d'espace de nom `xmlns="http://www.w3.org/1999/xhtml"` (non abordé ici) et une indication quant à la langue utilisée `xml:lang="fr"`.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/
xhtml11/DTD/xhtml11.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr">
  <head>
    <meta name="keywords" content="html, xhtml, xml, css"/>
    <meta name="author" content="votre nom/pseudo ici"/>

    <meta http-equiv="Refresh" content="10; URL=next.html"/>

    <title> Le titre de votre page Web! </title>
  </head>
  <body>
    <!-- Le contenu de votre page Web -->
  </body>
</html>
```

On a ajouté ici une ligne particulière dans l'en-tête pour effectuer un changement de page automatique (bien pratique, lors d'un changement d'adresse). Ainsi dans 10 secondes, le butineur ira automatiquement butiner la page appelée 'next.html'.

• Dans le titre de la page Web, aucun formatage n'est possible!

[La balise **<body>** peut aussi prendre des attributs...]

Attention, tremplin!

>>> Conclusion

:: Bilan Au terme de ce tutoriel, vous connaissez a peu près bien le XHTML: sa construction, son fonctionnement, ses règles (plus strictes que le HTML). Pour autant, comme tout langage, le XHTML est en évolution et en outre ce qui a été exposé ici n'embrasse pas avec exhaustivité toutes ses facettes. Cependant avec ce bagage vous pouvez déjà réaliser un bon nombre de pages de qualité!

:: Comment améliorer son code? C'est une bonne idée et plusieurs solutions vous sont offertes. Pour un usage du XHTML particulièrement pointu: optez pour l'achat d'un bouquin spécialisé; vous pouvez aussi réutiliser le code existant contenu dans les pages des sites que vous butinez! La procédure est simple: dans votre butineur, cliquez sur le menu "Affichage" puis rechercher quelque chose comme "HTML" ou "Voir Source". Cela fait, une nouvelle fenêtre apparait vous montrant le code HTML de la page en cours, c'est une mine d'or pour comprendre l'encodage de telle ou telle présentation! La boucle est bouclée, vous apprenez par vous même... Cela étant rien n'empêche d'appliquer les deux méthodes ;^)

:: W3C Le Web possède une organisation nommée le W3 Consortium qui s'occupe de l'évolution de toutes technologies qui y sont liées. Plusieurs groupes de travail publie relativement fréquemment des recommandations qui précisent explicitement tous les rouages des langages... Aussi pour être au faite des dernières publications, vous pouvez obtenir une

copie des [recommandations officielles](#) du XHTML (versions: 1.0, 1.1 et 2), ainsi que du HTML 4.01. Cependant prudence, les dernières versions des langages ne sont prises en compte que progressivement, aussi un code trop basé sur les dernières version risque d'être mal interprété par les butineurs...

+ Contact: cotito@free.fr • Web: <http://cotito.free.fr>

website.Solutions • Copyright © 2002-2003, Tous droits Réservés.