

1. Les doctypes

Le but d'une Document Type Definition est de définir les tags du type de document choisi (HTML, XHTML, XML, ...) Il définit une structure du document avec une liste d'éléments prédéfinie. N'importe qui peut créer sa propre DTD. Les éléments à définir sont des "blocs de constructions généraux" tel BODY; TABLE; ...

L'élaboration d'un tag passe par 3 étapes :
Création du tag
Définition des « attributes et entities »

Ex :

< ! element TR !> → Crée la balise <TR>

< ! ATTLIST TR → Liste des attributs de <TR>

[nom de l'attribut] [Type de donnée] [options tel « #IMPLIED required an ' 1 ' »]

→ Fixe les attributs

!> → Fin de la liste des attributs

Tags : balises dans lesquelles on peut introduire du texte (des données) de façon à le formater

Attributes : Attributs définissant les balises: information complémentaire

Entities : Codification des caractères spéciaux: < est défini dans le code HTML par exemples PCDATA (Données insérées entre les "TAGS" et qui seront formatées) ou CDATA (Données insérées dans les "TAGS" (valeur des attributs))

2. Le CSS

Syntaxe :

sélecteur {propriété :valeur ;}

Ex :

Pour une balise → BODY { color :navy ;}

Pour une classe → .titre {color :navy ;}

Pour un id → #24 {color :navy ;}

Les commentaires dans une page CSS externe (page.css) s'inscrivent comme des commentaires dans le langage C : /* commentaire */

Inheritance (principe d'héritage)

Chaque élément dans une page html peut être parent ou enfant d'un autre élément.

L'élément HTML est le parent de l'élément HEAD et BODY,

ainsi que l'élément BODY serait le parent de l'élément TABLE.

Dans l'autre sens, l'élément TABLE est l'enfant de BODY, et BODY est l'enfant de HTML.

L'élément HTML est le parent le plus élevé, mais on peut considérer que la DTD peut être le parent de HTML.

Comme ci-dessous:

[Parent] HTML >>> BODY >>> TABLE >>> TR >>> TD [Enfant]

[Parent] HTML >>> HEAD >>> META [Enfant]

Specificity (valeur)

Simple selector: 1 (<p>)

Class selector: 10 (<... class="">)

ID selector: 100 (<... id="">)

Style selector: 0

➔ Celui qui aura le plus de poids primera sur les autres dans une page HTML

Importance

!important

ex:

p.sombre { color: #333 !important; background:white; }

-> Ce style aura quasiment toujours prédominant

!important specificity: 10'000

On distingue 3 styles de CSS:

Feuille de style (CSS) interne

Feuille de style (CSS) externe

Feuille de style (CSS) utilisateur (propre au browser)

On ne peut pas dire qu'il y a un ordre de valeur absolu entre ces 3 feuilles de styles différentes, tout dépend du facteur "specificity" et non du type de feuille de style utilisée.

Cascade (ordre)

• Où apparaît l'élément "style"?

-> De même valeur, c'est le dernier qui est actif

ex: ici c'est le "blue" qui sera actif

p { color: red; }

p { color: blue; }

• évaluation specificity

Cfr ci plus haut. Plus le specificity est élevé, plus il primera sur les styles plus généraux ou de moindre specificity

• évaluation importance

->

ex: p.sombre { color: #333 !important; background: white; }

• évaluation héritage

Si on applique un style à un tableau (TABLE), les enfants de cet élément adopteront le même style (TR, TD, TH)

ID & CLASS : Les différences

- inheritance

l'ID selector ne permet pas l'héritage des styles

Le CLASS selector est héréditaire

- specificity

ID : 100

CLASS : 10

- style

ID est appelé dans une feuille de style par le caractère # →

Dans le fichier html → id="bonjour"

Définition du style → #bonjour {property:value;}

CLASS est appelé dans une feuille de style par le caractère .

Dans le fichier html → class="classique"

Définition du style → .classique {property:value;}

Les feuilles de styles peuvent être insérées dans une page de 4 manières différentes :

- via la balise <style> dans <head>
- via la balise <style> dans <body>
- directement dans la balise (inline) (ex : <div style= 'color :navy'>)
- via une feuille externe (balise <link> dans <head>

3. XML - XHTML

Différences majeures entre l'html et l'xhtml:

XHTML: est basé sur l'XML et fait la dissociation de la forme du contenu

HTML: est basé sur le SGML contenu et forme font partie du même fichier

Le XHTML : utilise les mêmes balises que l'HTML 4 cependant voici quelques règles qui caractérisent le XHTML et qui le différencient de l'HTML :

- Il est sensible à la casse (distinction minuscule et la majuscule)
- Toutes les balises doivent être écrites en minuscule
- tout documents XHTML doivent disposer d'une entête correcte (DTD, iso...)
- toutes les balises doivent être fermées (ex du
 qui devient
)
- valeur d'attributs doivent être écrits entre guillemets.
- tout attribut doit avoir une valeur

L'XHTML ne contient plus les balises suivantes :

<center><base font><dir><?.index><applet><menu><strike><u><s>

L'attribut name n'existe plus

Il n'y a plus de minimized-attribute value, c'est-à-dire que pour un attribut HTML sans valeur, on doit répéter l'attribut et le mettre comme valeur, entre "".

Cela peut paraître ridicule, mais cela sert à ce que l'interpréteur XML comprenne bien le message.

Le css est indispensable quand on utilise le XHTML.

Le XHTML va dans le sens de compatibilité accrue, ce qui fait que si il est correct, les browsers sauront le lire, tandis que si il y a des erreurs dans l'HTML, ils ne sauront peut être pas l'ouvrir de la même manière.

Entête XHTML

```
//Prologue XML
<?xml version="1.0" encoding="UTF-8" 2>
//DTD XHTML transitional, frameset et strict
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
//Spécification de la langue
<html xmlns="http://www.w3c.org/1999/xhtml" xml:lang="fr" lang="fr">
//Ne jamais doubler les balises HEAD, HTML, BODY
//Balise HEAD :
<head>
<meta http-equiv="content-type" content="text/html; charset=UTF-8" />
</head>
```

4. UTF-8

L'UTF-8 utilise un système 8 bits.

Pour les Operating System, c'est le plus souvent la langue américaine (en 7 bits) qui est utilisée pour la programmation. Mais pour pouvoir exprimer le système de la langue française ou allemande, par exemple, il faudrait utiliser un espace 8 bits, car ces langues utilisent des lettres accentuées que la langue américaine ne connaît pas. Or, au départ, tout a été conçu avec un système 7 bits. C'est un choix local, peut-être pas très judicieux et très probablement politique. Mais peut-être ne se rendait-on pas compte à ce moment de la portée que cela allait avoir plus tard...

$7 \text{ bits} = 2^{\text{exp}7} = 128$ -> correspond plus ou moins à notre alphabet, en minuscules(a-z) et en majuscules(A-Z), plus les caractères de base (. ; : / ? + = ~ - _ etc.)

Pour pouvoir utiliser le système des langues d'Europe Occidentale (Western Europe), il faudrait utiliser un système 8 bits (1 byte) -> $8 \text{ bits} = 2^{\text{exp}8} = 256$ caractères.

Mais ce système ne comprendrait pas les langues bidirectionnelles, comme l'hébreux, le japonais ou l'arabe.

Si on utilisait un système UTF-16 -> $16 \text{ bits} = 2^{\text{exp}16} = 65536$ caractères, cela doublerait le poids du fichier, ce qui est loin d'être conseillé dans le domaine du Web.

L'UTF-8 permet de faire un compromis entre poids de fichier et internationalisation.

L'inconvénient est que le côté commerciale de cette utilisation ne permet pas de développer des langages pour les langues marginales ou marginalisées à cause de leur faible pouvoir économique. C'est déjà une bonne chose que les langages soient développés en français.

Les Arabes et les Japonnais, par exemple, devaient jusqu'il y a peu, utiliser un langage de retranscription en caractères latins pour pouvoir envoyer leurs mails...

5. Les Formulaires <FORM></FORM>

Attributs principaux de <form >

Action : Cet attribut spécifie le programme devant traiter le formulaire. La valeur de l'attribut peut être une URL HTTP (emplacement du programme) ou une URL MAILTO (envoi du formulaire par courrier électronique).

Method :

METHOD = POST affecté à une variable d'environnement \$Contentlength

METHOD = GET affecté à une variable d'environnement \$Pathinfo

Target : comme pour les liens

Enctype= 'multipart/form-data' : Lorsque l'on veut attacher un fichier.

Les champs

<input type= 'checkbox'> → cases à cocher

<input type= 'radio'> → bouton type radio

<input type= 'reset'> → crée un bouton initialise les champs du formulaire

<input type= 'button '> → crée un bouton

<input type='submit'> → crée un bouton qui enverra les données du formulaire

<textarea></textarea> → zone de texte éditable

<input type='text'> → zone de texte en 1 ligne

<input type='password'> → zone de texte en 1 ligne dont les caractères sont cachés par ***

<select><option></option></select> → menu déroulant

Attributs de ces derniers :

- La balise qui permet à l'utilisateur de saisir une ligne de texte est le champ de saisie.

<input type="text" name="nom" size=40 maxlength=40 value="ce qui sera dans la case">

ex:

Name = le nom de la donnée (obligatoire).

Size = la taille de la fenêtre dans le navigateur.

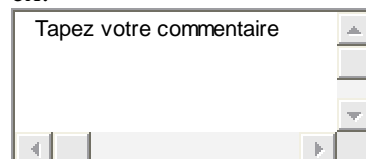
Maxlength = le nombre maximale de caractères dans la fenêtre.

Value = ce qui sera dans la fenêtre.

- La balise qui permet à l'utilisateur de réaliser un texte sur différentes lignes est le texte multi lignes.

<textarea name="commentaire" cols=30 rows=4>Tapez votre commentaire</textarea>

ex:



Name = le nom de la donnée (obligatoire).

Rows = le nombre de lignes dans la fenêtre.

Cols = le nombre de colonnes (en caractères) dans la fenêtre.

Readonly = est en mode lecture seulement.

- La balise qui permet à l'utilisateur de choisir un élément dans un menu déroulant.

```
<select name="liste">  
<option value="Choix 1">Choix 1</option>  
<option value="Choix 2">Choix 2</option>  
<option value="Choix 3">Choix 3</option>  
</select>
```

ex:



Name = le nom de la donnée (obligatoire).

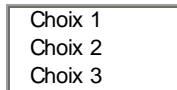
Value = nom de la donnée pour le choix.

Selected = si présent dans la balise <option> alors choix reconnu comme choix par défaut.

- La balise qui permet à l'utilisateur de choisir plusieurs éléments dans une liste.

```
<select name="liste" size=3 multiple>  
<option value="Choix 1">Choix 1</option>  
<option value="Choix 2">Choix 2</option>  
<option value="Choix 3">Choix 3</option>  
</select>
```

ex:



Name = le nom de la donnée (obligatoire).

Size = le nombre de lignes dans la liste (minimum 2).

Value = nom de la donnée pour le choix.

Selected = si présent dans la balise <option> alors choix reconnu comme choix par défaut.

- La balise qui permet à l'utilisateur de cocher des cases (choix multiple).

```
<input type="checkbox" name="CHOIX" value="CASE 1"> CASE 1  
<input type="checkbox" name="CHOIX" value="CASE 2"> CASE 2  
<input type="checkbox" name="CHOIX" value="CASE 3"> CASE 3
```

ex:



Name = le nom de la donnée (obligatoire).

Value = nom de la donnée pour le choix.

Checked : Si est présent dans la balise <input>, le bouton sera coché par défaut.

- La balise qui permet à l'utilisateur de sélectionner un bouton (choix unique).

```
<input type="radio" name="CHOX" value="Bouton 1"> Bouton 1  
<input type="radio" name="CHOX" value="Bouton 2"> Bouton 2  
<input type="radio" name="CHOX" value="Bouton 3" checked> Bouton 3
```

ex:



Name = le nom de la donnée (obligatoire).

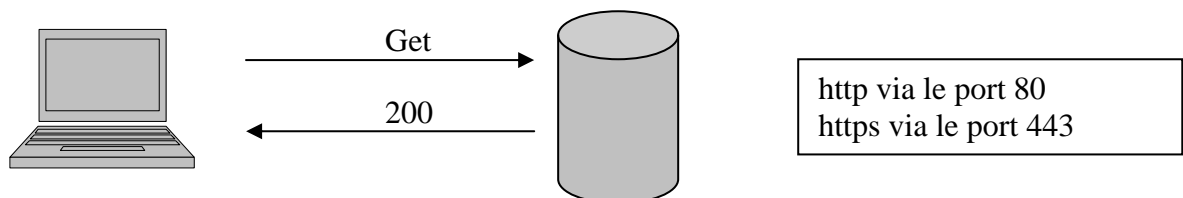
Value = nom de la donnée pour le choix.

Checked : Si est présent dans la balise <input>, le bouton sera coché par défaut.

Comment ça marche

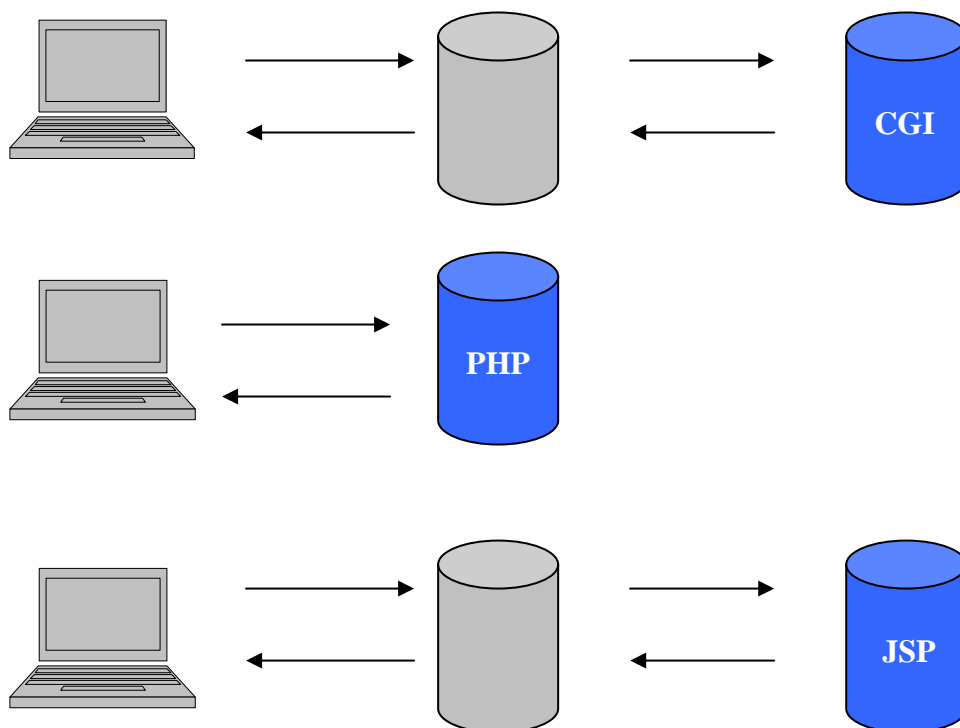
Les formulaires permettent d'entrer en interaction avec l'utilisateur. Les formulaires se font avec plusieurs champs dans lesquels le visiteur entre une information ou opte pour un choix. . L'information est ensuite envoyée, à l'aide de scripts, sur le serveur puis renvoyée sur un Email.

Enquête statique.



Le client (le browser) devient serveur pour 1 moment. Il est d'abord client car il reçoit l'information, ensuite il valide le questionnaire et l'envoie donc il devient serveur et le serveur devient client. Les rôles sont donc interchangeables.

Enquêtes dynamiques



CGI = Common Gateway Interface.

Le script peut être écrit en Perl (peut être transféré sous forme de texte), shellscript, python, c et même c++ (ces deux derniers ne sont pas l'idéal car le script est très lourd et qu'il est compilé. Maintenant un script basé côté serveur peut s'exécuter sans CGI.

PHP offre une très bonne rapidité si il est installé comme module (sous apache) plutôt que comme programme.

Le JSP est plus rapide que le CGI car il est dans le cache du serveur.

Forme permet l'interactivité côté client par le biais d'un script. Le script côté serveur va exécuter le formulaire. Utilisation d'un script CGI: www.scriptarchive.com/formmail.html

Cookies et « Session Management »

Principe :

L'information est conservée dans le navigateur (netscape, safari, modzilla...)

Structure des données : site web / nom du cookie / chemin vers le serveur distant / date d'expiration /contenu. Cela permet d'identifier tout utilisateur à chaque connexion à 1 site, il est donc possible de reconnaître 1 visiteur.

Lors d'une connexion, les variables d'environnement sont échangées : comparaison avec un pot de confiture, le browser doit mettre la confiture dans le pot = saisie d'une url, envoi au serveur.

Il existe plusieurs variables d'environnement.

- \$HTTP_USER_AGENT : toujours la même variable si on utilise toujours le même browser, c'est l'empreinte du browser.

- \$HTTP_REMOTE_USER : l'adresse IP liée à chaque requête, adresse IP de l'utilisateur distant.

- \$QUERY_STRING : url + extrait path

- \$CONTENT_LENGTH

6. Le format SVG (Scalable, Vector Graphics)

- Le SVG est un format de description de dessin vectoriel au format XML, qui permet donc de faire des formes, dégradés, textes qui suivent une ligne, etc. Il est de plus bien optimisé pour le web.

- Le SVG est proposé par le w3C, c'est donc un format ouvert (il n'appartient à aucune société, il sera donc toujours accessible par tous) et utilisable sous Linux et Windows.

- Pour l'instant, ce format n'est pas encore bien implémenté sur le web : peu de navigateurs acceptent ce format, mais il devrait être de plus en plus reconnu à l'avenir. Le browser Mozilla par exemple accepte déjà le SVG. Plus tard, il sera possible de faire de l'animation au format SVG, qui pourrait devenir un sérieux concurrent pour Flash!

7. Open Source et licences GPL...

SODIPODI (sodipodi.sourceforge.net) est un éditeur WYSIWYG SVG. C'est un logiciel libre et gratuit.

Il permet donc de faire du dessin vectoriel au format SVG, qui sera sauvé en format texte !

Il ne faut pas comparer SODIPODI avec des logiciels propriétaires comme Illustrator; on utilise rarement toutes les fonctions d'un tel programme, et SODIPODI permet déjà de faire énormément de choses.

Il faut éviter de limiter son champ de vision à un seul logiciel sous prétexte qu'on a l'habitude de l'utiliser ou que son entourage professionnel l'utilise; on peut très bien imposer un logiciel libre dans son travail. Un logiciel commercial APPARTIENT à une société qui en fait ce qu'elle veut, par exemple arrêter son développement...

Il faut payer pour une license limitée, sans compter les mises à jour, le hardware, l'O.S.,... ce qui représente un coût total très élevé surtout en cas de faille de sécurité.

De plus, dans le cas d'un logiciel commercial, quand il y a une faille de sécurité, on est obligé d'attendre le fix de sécurité proposé par le propriétaire puisque le code est protégé.

L'ordinateur devient vulnérable pendant ce laps de temps qui peut durer longtemps.

Dans le cas des logiciels libres (mais aussi pour les commerciaux), il existe des PEER

Molitor Yannick
2WB

REVIEW (inspection de la communauté) où tout le monde peut participer au développement du logiciel et faire partager ses trouvailles.

Donc dès qu'il y a une faille de sécurité, la réparation est quasi instantanément disponible.

- Conclusion: Il devient de plus en plus intéressant d'utiliser des logiciels libres qui comportent de nombreux avantages!

Il faut savoir faire des choix.

En bref :

The Gimp 1.3 permet de créer des documents au format SVG / Traitement bitmap

Sodipodi permet de créer des document SVG / Traitement d'image vectorielles

Scribus est un DTP (Desktop Publishing) / Réplique de Quark Xpress et permet de transformer une page en PDF ou en SVG

8. Divers

Utiliser le CGI de l'école

Host : 10.0.0.111
Username : tiW2
Password : 2000it2
Répertoire : public – HTML
<http://10.0.0.111/~tiw2//> « nom de votre répertoire »

Exemple de formulaire utilisant Formmail :

```
<form method="get" action="http://10.0.0.111/cgi-bin/formmail.pl">
```

```
<form method="post" action="http://10.0.0.111/cgi-bin/formmail.pl">
```

- Get : l'information est traduite en url puis transmise par la barre de navigation. Dans ce cas, tout se trouve dans la variable QUERY_STRING, on a alors accès à l'url.

L'espace mémoire d'une url est limitée à environ 2000 caractères (2051). Si il y a plus de caractères, on utilise la méthode Post.

- Post : dans ce cas, tout se trouve dans la variable CONTENT_LENGTH. Par exemple, cette méthode est plus appropriée si le formulaire contient un champ upload (attachement d'un fichier dans un email) car on dépasse alors les 2000 caractères.

Content_length est donc comme un pot de confiture beaucoup plus grand que Query_string. Mais lui n'est pas accessible par l'url.

Un avantage de Get sur Post est qu'on peut directement faire un script sans passer par un formulaire. Un inconvénient est le manque de confidentialité.

Javascript « Detect OS & Browser » (et sa correction)

Code Javascript pour détecter le navigateur employé et rediriger vers une CSS correspondante:

Version du prof (mais elle ne fonctionne pas → la mienne est en dessous ;o)

```
// verification css
// declaration variable
var os = "win" ;
var browser = "ie" ;
var browserinfo = navigator.userAgent.toLowerCase();
// verifiacion plate-forme
if(browserinfo.indexOf("mac") != -1) os = 'mac';
if(browserinfo.indexOf("win") != -1) os = 'win';
if(browserinfo.indexOf("linux") != -1) os = 'linux';
// verification du browser
if(browserinfo.indexOf("mozilla" || "netscape") != -1) browser = 'ns';
if(browserinfo.indexOf("netscape6") != -1) browser = 'ns6';
if(browserinfo.indexOf("gecko") != -1) browser = 'ns6';
if((browser == 'ns6') && (os == 'mac')) browser = 'ns6mac';
if((browserinfo.indexOf("msie") != -1) browser = 'ie';
if((browserinfo.indexOf("msie5") != -1) && (os == 'mac')) browser = 'ie5mac';
if((browserinfo.indexOf("opera") != -1) browser = 'opera';
if((browserinfo.indexOf("icab") != -1) browser = 'icab';
// chargement feuille de styles
document.write("<link rel='stylesheet' type='text/css' href='css/"+os+"_"+browser+".css'>");
// fin du script
```

Ma version :

```
var os="win";
var browser="ie";
var browserinfo=navigator.userAgent.toLowerCase();
if (browserinfo.indexOf("mac") != -1 ) { os="mac";}
if (browserinfo.indexOf("win") != -1 ) { os="win";}
if (browserinfo.indexOf("linux") != -1 ) { os="linux";}
if (browserinfo.indexOf("unix") != -1 ) { os="unix";}
if (browserinfo.indexOf("mozilla"||"netscape") != -1 ) { browser="mozilla";}
if (browserinfo.indexOf("netscape6") != -1 ) { browser="net6";}
if (browserinfo.indexOf("gecko") != -1 ) { browser="gecko";}
if ((browser=="net6") && (os=="mac") ) { browser="netmac";}
if (browserinfo.indexOf("msie") != -1 ) { browser="ie";}
if ((browserinfo.indexOf("msie 5") != -1 ) (os=="mac")) { browser="ie5mac"; }
if (browserinfo.indexOf("opera") != -1 ) { browser="opera";}
if (browserinfo.indexOf("icab") != -1 ) { browser="icab";}
document.write("<link rel='stylesheet' type='text/css' href='css/"+os+"_"+browser+".css'>");
```

9. Lexique

SGML : Standard Generalized Markup Language

HTML : HyperText Markup Language → langage web basé sur le SGML

XML : eXtensible Markup Language → Métalangage web permettant de créer d'autre langage web tel le XHTML, WML ,SVG ou RDF. Les applications XML utilisent uniquement des balises structurelles des données

DNS : Domain Name Service → Attribue un nom de domaine à une adresse IP

RFC : Request For Comment → Règles de syntaxes et structures d'un protocole

URL : Uniform Ressource Locator → adresse d'un fichier peut être absolue
(<http://www.google.be/monfichier.html>) ou relative (../data/monfichier)

Structure d'une url absolue :

Protocole : // sous-domaine / nom de domaine / répertoire / file

SSL : Socket Secure Layer → Connection sécurisée (cryptage des données basées sur des certificats)

CMS : Content Management System → Système de gestion du contenu tel spip ou wikiweb

TLS : Transfert Logger Security → Système sécurisé (comme ssl, ssh,...)

XHTML → basé sur le XML (et non plus le SGML), la forme est dissociée du contenu la forme est interprétée par des fichier CSS

RDF Resource Description Framework → échange de données intersite

DTD : Document Type Definition → Prologue qui permet de définir le type de document.

IP : Internet Protocol

TCP : Transfert Protocol

FTP : File Transfert Protocol

CSS : Cascading Style Sheets → Feuille de style

CGI : Common Gateway Interface → programme

SVG : Scalable Vector Graphics

WML : Wireless Markup language → Langage pour la diffusion sans fil.

DOM : document object model → Model sheet

10. Sources

- Mes notes
- <http://users.skynet.be/preumont/2tiwb1/>