TD2 : XHTML/CSS

Le but de ce TP est d'apprendre les bases du langage XHTML. Voici deux sites de référence pour le XHTML et le HTML : `http://www.w3.org/TR/xhtml1`
`http://www.w3.org/TR/1999/REC-html401-19991224`.

La version 4.01 de HTML recommande dorénavant une séparation du *contenu* et de la *forme*. Les balises HTML 4.01/XHTML sont ainsi utilisées seulement pour structurer le document, et les *feuilles de styles* ou *CSS* en anglais (*Cascading Style Sheets*) doivent être utilisées pour indiquer au navigateur la manière de présenter chaque élément du contenu.

XHTML, basé sur le langage XML, est le langage qui a pour but de remplacer HTML. XHTML 1.0 est quasiment identique à HTML 4.01, il se veut simplement plus strict et propre. Un document XHTML décrit une page destinée à être affichée par un navigateur web. On décrit la page à l'aide d'éléments XHTML. Ces éléments expriment la façon de structurer la page.

Normalement, un élément encadre son contenu à l'aide de balises. Par exemple, pour écrire le mot "bonjour" en italique, on utilise l'élément `i`, ce qui s'écrit en xhtml : `<i> bonjour </i>`. Autrement dit, un élément `el` commence par la balise ouvrante `<el>` et se termine par la balise fermante `</el>`. Dans un document, tout élément ouvert doit être fermé. De plus, les éléments doivent être correctement imbriqués (`<b> <i> bonjour </b> </i>` n'est pas correct, on doit écrire `<b> <i> bonjour </i> </b>`).

Toutefois, quelques éléments sont dits vides. Ils n'encadrent rien et n'ont qu'une balise (qui est donc ouvrante et fermante). C'est le cas de l'élément `br` (qui permet de passer à la ligne). La syntaxe de XML permet de fermer un élément en lui attribuant une barre oblique (un slash) en fin de balise. Ainsi si `el` est un élément vide de XHTML, il faut écrire `<el />` et non `<el>`. On peut citer par exemple `<br />`, ou encore `<img />`. Il ne faut pas oublier d'inclure un espace entre le contenu de l'élément et la barre oblique, car autrement, les anciens navigateurs, en particulier Netscape 4.x, ne pourront l'interpréter et l'ignoreront tout bonnement

Notons enfin que le nom des éléments dans les balises doit être en minuscule (autrement dit, on n'écrira pas `<B>bonjour</B>`). En effet XHTML est sensible à la casse (on fait la différence entre les majuscules et les minuscules), ce qui n'est pas le cas de HTML.

D'autre part, la balise ouvrant un élément peut (et parfois doit) prendre un ou plusieurs attributs. Ceux-ci ont alors la forme `attribut="valeur"` (et on notera les lettres minuscules ainsi que les guillemets autour de la valeur). Par exemple, pour inclure une image *image.jpg* dans un document, on utilisera `<img src="image.jpg" alt="une image" />`.

De plus, un document XHTML doit commencer par la déclaration d'une DTD (Document Type Definition) suivi du tag HTML qui englobe tout le reste du document XHTML. Une DTD est un document permettant de décrire un modèle de document XML. Une DTD indique les noms des éléments pouvant apparaître et leur contenu, i.e. les sous-éléments et les attributs. En dehors des attributs, le contenu est spécifié en indiquant le nom, l'ordre et le nombre d'occurrences autorisées des sous-éléments. En XHTML on distingue 3 DTDs, le *strict* qui impose une séparation entre le contenu et la forme, le *transitional*, pour permettre un passage en douceur entre les mauvaises habitudes de HTML et les nouvelles habitudes de XHTML et le *frameset* qui permet l'utilisation des *frames* HTML en plus (ce que les deux autres ne permettent pas). On va privilégier le transitional. La déclaration de la DTD sera toujours :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

(Remarquons que cette déclaration n'est pas à proprement parler un élément XHTML et n'a pas de balise fermante, ni ne finit par `/>`)

L'élément racine d'un document doit impérativement être l'élément `html` et celui-ci se doit d'avoir un espace de nom (namespace) utilisant l'attribut `xmlns` et une déclaration de la langue utilisée principalement dans le document. Par exemple :

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr">
```

Au minimun, l'élément HTML doit contenir un élément `head` suivi d'un élément `body`. L'élément `head` doit contenir un élément `title` qui définit le titre de la page, ainsi que la ligne suivante :
`<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />`
qui définit l'encodage des caractères de la page (en l'occurence, ici, iso-8859-1). Notons que cette déclaration n'est pas à proprement parler obligatoire ; le validateur (voir section suivante) acceptera quand même de valider votre page sans elle en utilisant un encodage par défaut, mais il se plaindra. L'élément `body` quant à lui, contient la page en elle-même.

Il est recommandé d'utiliser en début de page la déclaration `xml` lorsque l'on utilise un encodage différent de l'encodage par défaut (UTF-8 ou UTF-16). Si l'encodage utilisé est le iso-8859-1, on le déclare ainsi :

```
<?xml version="1.0"encoding="iso-8859-1"?>
```

Pour résumer, le squelette minimum d'une page XHTML est le suivant :

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
<head>

<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
<title></title>
</head>
<body>
</body>
</html>
```

Nous rappelons qu'un document xhtml doit avoir l'extension `.html` (ou `.htm`).

Exercice 1 :

1. Ecrire une page web de titre "Première page" contenant simplement le texte "Bonjour tout le *monde* !" (ou "monde" est en italique). Il faut maintenant s'assurer que votre première page est un document XHTML valide. Le w3c, qui est l'organisme chargé de développer les standards pour le web et qui a créé le langage XHTML, fournit un validateur que l'on trouve à l'adresse `http://validator.w3.org/`.
2. Vérifiez si votre page est valide. Si ce n'est pas le cas, rendez-la valide.
3. Lorsque votre page est valide, le w3c vous propose d'ajouter un petit logo à votre page pour signifier aux futurs visiteurs de votre page que vous prenez soin de respecter les standards. Rajoutez ce logo à votre page.

1 Des balises par millions

Dans l'annexe A, une liste non exhaustive d'éléments XHTML non vides est présenté dans le tableau 1. Certains éléments vides sont présentés dans le tableau 2. D'autre part, certains caractères ne sont pas accessibles directement dans un document XHTML car ils ont une signification particulière (par exemple le caractère $<$). Pour afficher ces caractères, on utilise des entités (voir tableau 3). Précisons également que les commentaires commencent par `<!--` et finissent par `-->` et on ne peut pas les imbriquer. Vous trouverez ici `http://www.w3schools.com/default.asp` une liste exhauxtive des éléments et des entités de XHTML et un certain nombre de tutoriaux.

Exercice 2 :

La page correspondant au fichier `/net/cremi/mkante/tmp/reseau/tp_errone.html` n'est pas valide. Récupérez-la et corrigez les erreurs afin de la rendre valide.

Vous avez peut-être remarqué que la page erronée s'affiche "presque" de la même façon que la page valide (essayez de voir le pourquoi du "presque"). Toutefois, seule la validité de vos pages vous assure d'une part, que l'affichage est "réellement" comme vous le voulez et surtout qu'il sera conforme à vos attentes sur tous les navigateurs.

Exercice 3 :

Essayez de reproduire le plus fidèlement possible la page web représentée par l'image suivante : `/net/cremi/mkante/tmp/reseau/page.png`. Vous devez ensuite valider le résultat.

Exercice 4 :

Rajoutez un lien depuis votre "Première page" vers la page de l'exercice précédent.

2 Feuilles de style

XHTML fait la différence entre la forme et le contenu. Nous avons vu dans la section précédente comment organiser le contenu. Dans cette section nous allons introduire les feuilles de style ou CSS en anglais (pour Cascading Style Sheets) qui nous permettent de controler l'affichage des éléments XHTML dans un navigateur. Elles sont complémentaires d'une page XHTML. On peut trouver de la documentation ici `http://www.w3.org/TR/CSS21`.

Une feuille de style consiste en une suite de déclarations de la forme :

```
selecteur_1, selecteur_2, ..., selecteur_m {
propriété_1: valeur_1;
propriété_2: valeur_2;
. . .
propriété_n: valeur_n
}
```

où `selecteur` est souvent un simple élément XHTML comme `body` ou `h1` (mais nous verrons qu'on peut être plus précis), où `propriété` est une propriété CSS (à ne pas confondre avec un attribut d'un élément) permise pour cet élément et où `valeur` est une valeur permise pour la propriété concernée. Ces définitions vont expliciter l'affichage de(s) élément(s) sélectionné(s). Notez les deux points après le nom de la propriété et le point virgule entre chaque déclaration. Chaque élément possède ses propres propriétés (par exemple, la propriété list-style pour l'affichage des listes) mais quelques propriétés peuvent être utilisées avec tous les éléments. Voyons celles qui vont nous servir dans ce TP :

background-color : permet de spécifier une couleur pour l'arrière plan. Les valeurs peuvent être soit un nom de couleur (mais il n'en existe qu'une dizaine de prédéfinies) soit une valeur hexadécimale de la forme #xxxxxx où x est un chiffre hexadécimal (le raccourcis #xyz désigne la couleur #xxyyzz). Plusieurs moyens s'offrent à vous pour connaître la valeur hexadécimale d'une couleur. Par exemple, vous pouvez utiliser un logiciel de graphisme (comme gimp) ou trouver une ressource sur le web (comme la page `http://www.w3schools.com/css/css_colornames.asp`).

color : Permet de définir une couleur pour l'élément sélectionné. Les valeurs permises sont les mêmes que pour la propriété background-color

font-style : Pour spécifier le style de police. Les valeurs possibles sont normal, italic ou oblique.

text-align : Permet de définir l'alignement horizontal d'un texte. Les valeurs possibles sont left, right, center ou justify (pour que les marges gauches et droites soient régulières).

margin-left, margin-right, margin-top, margin-bottom : Permet de définir un espace entre l'élément sélectionné et l'élément qui l'encadre. Les valeurs sont des nombres suivis d'une unité (par exemple cm pour le centimètre, px pour le pixel, em pour l'unité liée à la taille de la police de l'élément, ...).

Exercice 5 : Couleur de fond

Créez un fichier style.css et définissez une nouvelle couleur de fond (par exemple la couleur #C8C8C8) pour la page correspondant au fichier `/net/cremi/aguermou/reseau/xhtml/tp_css.html`.

Exercice 6 : Ancrage

On a vu que l'élément `a` pouvait avoir comme attribut `href="url"` lorsqu'on voulait afficher un lien vers une page externe située à l'adresse `url`. Mais on peut en fait être plus précis que ça et indiquer où exactement dans la page, le navigateur doit se positionner. Il faut pour cela avoir défini un point d'ancrage (d'où le nom de la balise) pour l'endroit que l'on veut pointer. Cela se fait grâce aux attributs `id="nom"` et `name=''nom''` de l'élément `a` à l'endroit voulu dans la page, les attributs `id` et `name` doivent avoir la même valeur, name n'apparaît en fait que pour une compatibilité ascendante. Ensuite, on peut y faire référence avec l'attribut `href="url#nom"` (ne pas oublier le #). Lorsque l'ancre est située sur la même page que le pointeur, on omet l'url `url`. Modifier le sommaire et les sections pour que l'on puisse depuis le sommaire accéder à n'importe quelle section et vice-versa.

Exercice 7 : Intégration à une page

Rajoutez la ligne suivante dans l'entête de la page : `<link href="style.css" rel="stylesheet" type="text/css" />`. Vérifiez que la couleur de fond est bien appliquée.

Exercice 8 :

Pour aérer un peu la lecture d'un document XHTML, on met en général un espace entre les bords de la fenêtre et le corps du document. Définissez une marge gauche, haute et droite pour le document de façon à ce qu'elle soit appliquée à tout le contenu de la page.

Exercice 9 : Liste de sélecteurs

À l'aide d'une seule définition, spécifiez une même couleur pour les éléments `h1` et `h2`.

Le fait de définir des propriétés pour plusieurs éléments en même temps n'empêche pas de rajouter plus loin dans la feuille de style des définitions pour un élément particulier de la liste. Rajoutez une définition pour que l'élément `h1` soit de plus centré sur la page.

Exercice 10 :

Les valeurs de type numérique peuvent être négatives. Faites en sorte que les titres de type h2 soient un peu en retrait (vers la gauche) vis à vis des autres éléments.

2.1 Les sélecteurs de classe

Il est parfois utile de pouvoir définir plusieurs comportements différents pour un même élément. Pour cela, on utilise des sélecteurs de classe qui sont de la forme `element.nom_de_classe` (le corps de la définition suit les mêmes règles qu'avant). Pour que cette définition s'applique dans le document XHTML, il faut alors rajouter à l'élément `el` l'attribut `class="nom_de_classe"`.

Exercice 11 : Un paragraphe spécial

Définissez deux comportements différents pour l'élément `p` : l'un général qui justifie les lignes d'un paragraphe, et un autre spécial qui affiche le texte en rouge. Éditez le document XHTML pour que l'avant dernier paragraphe applique cette dernière propriété.

Exercice 12 :

On va changer quelque peu l'affichage du tableau :

1. Ajoutez une entête à chaque colonne du tableau.
2. Définissez une nouvelle couleur de fond pour le tableau et décalez-le vers la droite.
3. Définissez une autre couleur de fond pour l'affichage des entêtes des colonnes.
4. En général, l'affichage des titres de livres se fait à l'aide d'une police italique. Réalisez cet affichage.

2.2 Les sélecteurs de pseudo-classe

La dernière forme de sélecteur que nous verrons dans ce TP concerne celle des *pseudo-classes*, de la forme `element :pseudo_classe`. Elle est surtout employée pour changer l'affichage des liens dans une page web : `a :link` pour les liens non visités, `a :visited` si l'internaute a déjà cliqué sur ce lien, `a :hover` lorsque la souris passe au dessus du lien et `a :active` lorsque l'internaute est en train de cliquer sur le lien.

Exercice 13 :

1. Définissez une couleur différente pour chacun des événements ci dessus.
2. `first-letter` est une pseudo-classe de l'élément `p` qui permet d'appliquer un style à la première lettre d'un paragraphe afin de produire un effet typographique. Définissez une pseudo-classe `first-letter` pour l'élément `p` qui double la taille et met en italique la premiére lettre d'un paragraphe.
3. Tester vos pseudo-classes.

Exercice 14 : Validation

De même qu'il existe un vérificateur pour les pages XHTML, il existe aussi un vérificateur pour les feuilles de style. Rendez vous sur la page `http://jigsaw.w3.org/css-validator/` pour valider votre feuille ou la corriger s'il y a des erreurs.

2.3 Les formulaires

XHTML permet la création de formulaires grâce à un certain nombre d'éléments. Les formulaires occupent une place un peu à part dans une page car ils constituent une composante interactive de la page. Un formulaire permet à un internaute de remplir des champs textuels, de cocher des cases, etc... mais surtout d'envoyer ces informations au serveur qui héberge la page. Ces informations peuvent ensuite être traitées pour, par exemple, afficher une page correspondant aux choix faits par l'internaute.

Nous allons pour l'instant nous intéresser simplement à la déclaration de formulaires. L'aspect traitement de l'information sera abordé les semaines suivantes. Commençons par une revue des éléments utiles pour un formulaire :

`form` **:** Pour déclarer un formulaire. C'est l'élément qui englobe tous les autres.

`input` **:** Pour définir un champ. Il contient en général l'attribut type qui peut avoir les valeurs suivantes :

- `text` **:** Pour définir une plage de texte.
- `password` **:** Pour obtenir un mot de passe (le texte sera caché mais non crypté).
- `button` **:** Crée un bouton.
- `checkbox` **:** Crée des cases à cocher.
- `radio` **:** Crée aussi des cases à cocher mais un seul choix sera possible.
- `file` **:** Pour pouvoir donner le chemin d'accès d'un fichier sur l'ordinateur de l'internaute.
- `hidden` **:** Pour interdire l'accès à ce champs là.
- `reset` **:** Pour réinitialiser le formulaire
- `submit` **:** Pour envoyer les informations (nous y reviendrons un peu plus tard).

D'autre part, il peut (et doit dans le cas des types checkbox et radio) contenir l'attribut value=”nom” dont la signification dépend de la valeur de type :
- Si le type est button, reset ou submit, alors nom est le texte inscrit sur le bouton.
- Si le type est password, text ou hidden, alors nom est le texte inscrit par défaut.
- Si le type est checkbox ou radio, alors nom est la valeur qui sera transmise au moment de l'envoi des données. Les boutons radio d'un même groupe doivent avoir le même nom.

`select` **:** Définit un menu déroulant. Le choix sera proposé via l'élément option en utilisant l'attribut `value="choix"`.

Les éléments `input` et `select` peuvent de plus contenir l'attribut `name="nom"` où nom sera plus tard utilisé pour traiter les données. Pour prendre de bonnes habitudes, dans la suite du TP vous ajouterez systématiquement un attribut name à tous les éléments destinés à être traités.

Exercice 15 :

Créez une page web intitulée Mon formulaire et proposant les champs suivants :
- Nom
- Prénom
- Mot de passe
- Sexe : choix masculin ou féminin
- Votre âge : choix dans un menu déroulant proposant des âges de 18 à 25 ans.
- Vous aimez : choix non exhaustif entre thé et café
- fichier : champ permettant de télécharger un fichier.

Vous n'oublierez pas le bouton d'envoi ni celui de réinitialisation.

Exercice 16 :

Écrivez une feuille de style qui centre le formulaire et change la couleur de fond des champs de texte lorsque l'on clique dessus.

A Annexe

Éléments	Nom	Description
b	bold	Affiche sont contenu en gras (bold)
h1 . . . h6	heading	Définit un entête (titre pour une section, paragraphe, . . .). h1 représente des titres de plus grande importance que h2, h2 que h3, etc . . .
a	anchor	Déclare un lien hypertexte. Ce tag contient en général l'attribut href pour déclarer la page vers laquelle pointe le lien.
ul	unordered list	Déclare une liste. Chaque élément de la liste est déclaré avec l'élément li
ol	ordered list	Déclare une énumération. Chaque élément de l'énumération est déclaré avec l'élément li.
p	paragraph	Déclare un paragraphe de texte.
pre	preformatted	Affiche un texte en préservant les espaces et les sauts de ligne.
table	table	Déclare un tableau. Il peut prendre l'attribut border pour définir l'épaisseur de la bordure. On définit alors les lignes avec l'élément tr, et dans chaque ligne, les cases avec l'élément td.

TAB. 1 – Éléments non vides

br	break	Passe à la ligne
hr	horizontal rule	Affiche une ligne horizontale

TAB. 2 – Éléments vides

Caractère	Entités
espace	;
< (less than)	&lt ;
> (greater than)	&gt ;
& (ampersand)	&amp ;
”	&quot ;

TAB. 3 – Entités