Cours CSS de Technofutur (2009)

Objectif :

L'objectif de ce cours est de vous familiariser avec les principes de base du langage CSS.

Le langage CSS va vous permettre de concevoir des feuilles de style qui prendront en charge la mise en forme de vos pages Web ; le langage HTML ne servant dès lors plus qu'à structurer les informations en paragraphes, titres, listes, définitions, tableaux, etc.

Utilité des feuilles de style :

Séparer le fond de la forme, respectivement au moyen des langages HTML et CSS, présentera de nombreux avantages :

· alléger le code source de la page Web (les balises de structure `p`, `h1`, `ul`, etc. ne côtoieront plus des balises de mise en forme telles que `i`, `u`, `font`, `b`, etc.),
· pouvoir modifier rapidement l'aspect d'un ensemble de pages Web constituant un site (la mise en forme étant généralement codée dans un fichier externe unique au moyen du langage CSS),
· dépasser les limites de mise en forme imposées par le langage HTML.

Objectifs :

Dans ce module, nous allons voir quelques exemples concrets de sites qui utilisent les feuilles de style CSS. Nous allons également voir comment relier un document HTML et une feuille de style CSS.

Dans la communauté Web, le terme CSS est plus souvent utilisé que feuille de style. Pour le cours, nous utiliserons donc également ce terme.

Après la lecture de ce module, vous aurez compris l'utilité des CSS et vous serez capable de créer une liaison entre un document HTML et une feuille de style.

Conclusion :

Une **règle CSS** comporte

· un sélecteur,

· une propriété

· une valeur,

ce qui donne :

```
sélecteur{propriété: valeur;}
```

Memo :

CSS = Cascading Style Sheet = feuille de style en cascade

Pour lier une feuille de style :

<link rel="stylesheet" type="text/css" href="mafeuilledestyle.css" />

(à insérer dans la partie `<head>` du document HTML).

Une règle CSS s'écrit comme suit:

sélecteur {propriété: valeur;}

Par exemple : `p {font-size: 11px;}`

Objectifs :

Maintenant que vous avez vu quelques exemples d'utilisation de CSS, vous allez créer vous-même des CSS.

Pour cela, il vous faut apprendre les caractéristiques du langage : règles, mots-clés, valeurs, etc.

Dans ce module, vous apprendrez à définir le style de titres, de paragraphes et de listes en modifiant la taille d'un texte, la police de caractère, la couleur, …

Synthèse :

Le but du langage CSS est d'appliquer des styles sur une ou plusieurs pages Web d'un site.

Avec la création du langage CSS, le langage HTML tend à se simplifier car le document HTML ne contient plus que la structure des informations (découpe en titres, paragraphes, listes, etc.) ; la mise en forme est en effet présente dans un fichier externe (la feuille de style).

Vous avez vu que l'on peut définir une règle de style pour n'importe quelle balise. Il suffit d'utiliser le libellé de la balise en tant que sélecteur.

Mémo :

Propriétés CSS vues dans le module :

Propriété	Valeur(s)	Description
font-family	Arial, Helvetica, sans-serif Verdana, Arial, Helvetica, sans-serif "Times New Roman", Times, serif "Courier New", Courier, mono Georgia, "Times New Roman", Times, serif Geneva, Arial, Helvetica, sans-serif	Définit la liste des polices de caractère appliquée au sélecteur
font-size	[valeur numérique] [px,pt,em,%,…] medium small smaller large larger …	Définit la taille du texte
font-style	normal italic oblique	Définit le style (l'inclinaison) du texte
font-weight	lighter normal bold bolder 100, 200, …	Définit l'épaisseur, la « graisse » du texte
color	[nom de couleur (en anglais)] #[code couleur]	Définit la couleur du texte
text-align	left, right, center, justify	Définit l'alignement du texte

---

Objectifs :

Maintenant que vous avez appris à appliquer une règle de style sur des paragraphes, des titres et des listes, vous allez apprendre à appliquer des styles à une partie de texte, tout comme vous le faisiez avec la balise <font> avant de connaître les CSS...

Ce module explique comment utiliser les CSS pour appliquer des styles à des blocs de texte ou à des morceaux de texte.

On parle de morceau de texte lorsque l'on souhaite sélectionner une partie de contenu au sein d'une ligne. Exemple :

« On fait tout pour gagner un cœur et bien peu pour le garder. », Jacques Deval

On parle de bloc de texte lorsque l'on souhaite regrouper plusieurs parties du contenu dans un bloc commun. Exemple :

Maître Corbeau, sur un arbre perché,
Tenait en son bec un fromage.
Maître Renard, par l'odeur alléché,
Lui tint à peu près ce langage :

"Hé ! Bonjour, Monsieur du Corbeau.
Que vous êtes joli ! Que vous me semblez beau !
Sans mentir, si votre ramage
Se rapporte à votre plumage,
Vous êtes le Phénix des hôtes de ces bois."

A ces mots le corbeau ne se sent pas de joie;
Et pour montrer sa belle voix,
Il ouvre un large bec et laisse tomber sa proie.

Extraits de « Le Corbeau et Le Renard » de Jean de la Fontaine.

Conclusion :

Toute balise HTML peut contenir l'attribut `class` lui permettant d'appliquer un style **personnalisé**.

On peut également appliquer des styles à des éléments spécifiques d'une page :

· morceau de texte,
· bloc de texte.

Mémo :

Syntaxe d'une classe :

.styleperso{…}

Application d'une classe au sein d'une balise :

<h1 class="styleperso">…</h1>
…
<li class="styleperso">…

Formatage d'un morceau de texte:

… … …

Formatage d'un bloc de texte:

… … …

Objectifs :

Ce module est consacré à tout ce que les CSS peuvent faire avec des images :

· mettre des images de fond sur des éléments : **image de fond dans une page**, **image de fond pour un lien**, …
· mettre une image à la place des puces d'une liste à puces.
· modifier la façon dont sont affichées les images dans le texte (bordure, marges…)

A la fin de ce module, vous serez capables de créer des règles de style qui utilisent les **images de fond** et des règles de style qui modifient **l'image d'une liste à puces**.

Synthèse :

Dans un site, les images peuvent être utilisées à diverses fins.

· Les images en tant qu'illustration du texte.
· Les images en tant que fond d'élément.
· Les images dans les listes à puce.

Grâce aux CSS, vous pouvez gérer les images de fond de tout élément, les images d'illustration et les images dans les listes à puces.

Toutes les propriétés relatives se trouvent dans le mémo ci-après.

Mémo :

Propriété	Valeur(s)	Exemple

background-image	[URL, chemin vers l'image]	body{
		background-image :
		url(images/fleur.gif);
		}

background-repeat	repeat (= valeur par défaut)
	repeat-x
	repeat-y
	no-repeat
		body{
		background-image :
		url(images/fleur.gif);
		background-repeat:repeat-y;
		}

background-	fixed	body{

attachment	scroll(= valeur par défaut)	background-image : url(images/fleur.gif);
		background-attachment:scroll;
		}

list-style-image	[URL, chemin vers l'image]	ul{
		list-style-image:url(images/fleche.gif);
		}

Objectifs :

L'objectif de ce module est d'utiliser toutes les notions CSS vues jusqu'à présent et de les **appliquer aux liens**.

Le lien hypertexte est le concept le plus important du Web car il permet d'interconnecter de l'information.

Par défaut, on reconnaît un lien à sa couleur bleue et à son soulignement. Avec les CSS, on peut appliquer des règles de style sur des liens en fonction de leur état: non visité, actif, visité ou survolé.

A la fin de ce module, vous serez capable d'appliquer des styles CSS sur des liens selon leurs états et de créer des menus soignés.

Enfin, un exercice récapitulatif vous sera proposé. Il vous permettra de mettre en pratique les différentes notions abordées dans les cinq premiers modules de ce cours.

Mémo :

Suppression du souligné : `text-decoration: none;`

Sélecteur	**Description**
a	Tous les liens hypertextes, quel que soit leur état
a:link	Les liens hypertextes non visités
a:hover	Les liens hypertextes survolés
a:active	Les liens hypertextes actifs
a:visited	Les liens hypertextes visités

Objectifs :

Jusqu'il y a peu, la grande faiblesse du Web était le positionnement des éléments dans une page.

En effet, pour positionner une image ou du texte dans une page, on avait souvent recours aux tableaux HTML.

Quels problèmes posent les tableaux ?

1. Les tableaux n'ont pas été créés pour positionner du contenu dans la page mais pour mettre en page des données structurées !
2. Les tableaux compliquent inutilement le code HTML.
3. Les tableaux alourdissent les pages et donc le chargement est plus long.
4. Les tableaux posent des problèmes de consultation sur des navigateurs spécifiques aux personnes handicapées.

Ce temps est révolu ! Grâce aux CSS vous pouvez donner une position précise aux éléments de votre page.

Dans ce module, vous allez apprendre à travailler avec les propriétés CSS de positionnement afin de positionner un bloc de texte dans une page ou dans un autre bloc de texte.

Synthèse :

Pour positionner un élément de manière exacte, on utilise les propriétés `position`, `left`, `top` et `z-index`.

La position d'un élément dépend ;

· de la position de l'élément qui le contient (élément parent), si l'élément parent est positionné,
· de la fenêtre du navigateur, si aucun élément parent n'est positionné.

De plus, la propriété `display` permet de gérer l'espace qu'occupera l'élément : **avec ou sans** retours chariot autour du contenu.

Vous trouverez un tableau récapitulatif de ces propriétés dans le mémo.

Grâce au positionnement CSS, l'utilisation des tableaux revient à son origine : présenter des données de façon tabulaire, comme vous allez le voir dans le module suivant…

Mémo :

Propriété	Valeur(s)
position	absolute : position à partir du coin supérieur gauche de l'élément qui le contient

	(élément parent) ou de la page relative : position à partir du contenu de l'élément qui le contient (élément parent) ou de la page static : valeur par défaut
left	[valeur] px [valeur] %
top	[valeur] px [valeur] %
z-index	valeur (du plus profond au plus proche)
display	none : élément invisible block : élément affiché en tant que block inline : élément affiché en tant que élément d'une ligne

---

Objectifs :

Avant la venue de CSS, le positionnement d'éléments dans une page relevait de bricolages avec des tableaux imbriqués dans des tableaux, dans des tableaux, etc.

Maintenant que nous avons les CSS pour s'occuper de la mise en page ainsi que du positionnement, les tableaux HTML retrouvent leur but initial : regrouper du contenu de type données (**tableau de données, liste de contacts, calendrier**, etc.)

Fini les tableaux imbriqués pour placer le menu d'une page !

Ce module a pour objectif de remettre en place une bonne utilisation des tableaux. Vous verrez quelques nouvelles propriétés applicables aux tableaux afin de ne plus utiliser les attributs HTML `width`, `height`, `border`, etc.

A la fin de ce module, vous serez capables d'appliquer un style CSS sur un tableau de façon à **remplacer** tous les attributs HTML prévus initialement.

Notez bien : ce module utilise les tableaux pour illustrer les propriétés `margin`, `padding`, `border`, etc. Ne perdez cependant pas de vue que ces propriétés sont également utilisables avec d'autres éléments du langage HTML (paragraphes, images, titres, etc.)

Synthèse :

Dans ce module, vous avez utilisé, sur des tableaux, des propriétés vues précédemment telles que `background-color`, `background-image`.

Vous avez également découvert de nouvelles propriétés :

·`width`,
·`height`,
·`border`,
·`margin`,
·`padding`.

Ces propriétés peuvent également être utilisées par d'autres éléments comme les paragraphes, les images, les balises `div`, …

Mémo :

Propriété	**Valeur(s)**	**Exemple**
width	Numérique px Numérique %	div{width :80% ;}
height	Numérique px Numérique %	div{height :350px ;}
border	Epaisseur : Numérique px style : solid / dashed / dotted couleur : pink, red, …. Ou une couleur hexadécimale.	td{border :1px solid pink ;}
margin	Top right bottom left Numérique px Numérique px, Numérique px Numérique px	Table{margin : 50px 0px 50px 0px ;}
padding	Top right bottom left Numérique px Numérique px, Numérique px Numérique px	Table{padding : 30px 30px 30px 30px ;}

Objectifs :

Dans ce module, nous allons apprendre à mettre en œuvre le concept d'héritage. Une fois cette notion acquise, vous serez en mesure de condenser le code de vos feuilles de styles afin de les rendres plus simples à consulter et plus rapides à exécuter.

Mais tout d'abord, un court rappel sur les notions étudiées dans les précédents modules…

Le concept parent-enfant dans le langage HTML :

L'un des aspects fondamentaux de la technique des feuilles de styles est qu'elle repose entièrement sur la structure du langage HTML. Celui-ci gère son code selon une structure d'arbre : dans ce cas, chacune des balises HTML possède un parent et peut engendrer un ou plusieurs enfants. Observons cela à travers une illustration.

Exemple de code source HTML et de sa structuration en arbre :

```
<html>
  <head>
  <title>Charlot – Filmographie</title>
  </head>
  <body>
    <h1>La ruée vers l'or</h1>
    <p>Fiche descriptive</p>
    <ul>
    <li>Acteur principal : Charlie Chaplin</li>
    <li>Année : 1925</li>
    </ul>
  </body>
</html>
```

D'après ce schéma, nous pouvons dire que la balise `body` est le parent des balises `h1`, `p`, `ul` et que les balises `li` sont, ici, les enfants de la balise `ul`.

Le concept parent-enfant dans le langage CSS :

En quoi la structure en arbre évoquée au point précédent concerne-t-elle les feuilles de styles ?

Et bien, de la même façon que les enfants héritent de certaines caractéristiques de leurs parents, les balises HTML « enfant » recevront les propriétés de style de leurs « parents ».

En d'autres termes, si une règle de style est définie pour la balise body, alors les balises qu'elle contient (dans l'exemple du point précédent : `h1`, `p` et `ul`) la possèderont également (sauf contre-ordre). Cet héritage s'étend à tous les niveaux.

Pour échapper à cet « héritage », il suffit de redéfinir la propriété concernée au niveau de l'élément HTML « enfant ». Dans l'exemple qui suit, c'est la propriété font-size qui nous intéresse.

```
body   { color: red;
   text-align: left;
   font-size: 10pt;}

h1   { font-size: 14pt;}
```

A priori, la balise h1 hérite de son élément parent body de la propriété font-size. Pas dans ce cas-ci car la propriété font-size est redéfinie pour h1 et possède une nouvelle valeur.

Quelques conseils :

Voici quelques conseils qui vous permettront d'améliorer vos feuilles de styles.

Ne retirez pas le ; après la dernière déclaration ! Pensez au futur…

Exemple :

div {color :#FFF; border: 1px solid red; width:340px;}

Le ; après 340px n'est pas indispensable mais on le laisse en vue d'un prochain attribut à déclarer.

Evitez d'écrire toutes les propriétés sur la même ligne ! Clarté, lisibilité…

Préférez :

```
body
  {
  font-size: 10px;
  font-family: arial, helvetica sans-serif;
  font-height: normal;
  color: #FF9900;
  }
```

à

body {font:normal 10px Arial,Helvetica,sans-serif; color :#FF9900;}

Utilisez des noms de classes pertinents

Préférez :

`.commentaires` et `.basdepage`

Et évitez `.style45` !

Le cas des polices relatives : em, % :

Il est recommandé d'utiliser des tailles de polices relatives (em principalement) afin de permettre aux mal-voyants de pouvoir modifier la taille des polices de caractère selon leur convenance.

En effet, si vous utilisez des unités de mesure absolues (px et pt par exemple), le visiteur ne pourra pas modifier la taille des polices de caractères.

Les unités em et % sont **relatives à la police de référence** : 1 em est égal à la taille de cette police.

Cependant, il faut bien comprendre que la taille de la police de référence **se transmet également par héritage**.

Par exemple, si vous définissez une taille de référence de 2em dans le `<body>`, puis une autre taille de 2em dans un élément enfant du `body` (par exemple `<table>`), les textes contenus dans l'élément enfant auront une taille de 2em par rapport à 2em, soit 4em !

Et ainsi de suite si vous cumulez les imbrications de balises enfants

Conclusion :

Nous voici déjà arrivés au terme de ce cours. Nous espérons que les feuilles de styles CSS vous sont devenues plus familières.

Au fil des différents modules, vous avez ainsi pu apprendre comment dissocier la structure d'une page Web (le code HTML) de sa mise en forme (le code CSS) et concevoir des design soignés pour vos sites Web.

Memo :

Le modèle parent-enfant est une notion fondamentale des feuilles de styles CSS.

L'élément enfant hérite des propriétés de l'élément parents (sauf dans certains cas comme les propriétés des blocs tels que `margin`, `padding`).

Prendre en compte la notion parent-enfant dès le début de la conception...

- permet l'optimisation de la CSS,
- évite la création de classes répétitives.

Comment valider son code (X)HTML & CSS :

Pour valider vos pages HTML et CSS, vous pouvez utiliser les validateurs du W3C :

Validateur HTML

http://validator.w3.org/

Validateur CSS

http://jigsaw.w3.org/css-validator/

Vous avez aussi de nombreux outils en ligne pour valider votre code. Retenons ceux-ci :

Validateur W3C Multipages

Comme son nom l'indique, ce validateur en ligne vous permet de valider plusieurs pages en une opération.

http://www.validateur.ca/

HTML Tidy Validator

Cette extension (prévue pour Firefox) vous permet la validation de votre page dés l'affichage de celle-ci dans Firefox. Vous pouvez également avoir un aperçu de votre code, ainsi que les lignes et le type d'erreurs rencontrées dans votre page html

http://users.skynet.be/mgueury/mozilla/