

Jean Engels

HTML5 et CSS3

Cours et exercices corrigés

Cours et exercices corrigés

EYROLLES

Jean Engels

Enseignant en mathématiques et concepteur de sites Web, Jean Engels est auteur de nombreux ouvrages portant sur les technologies du Web : HTML, CSS, JavaScript, PHP et MySQL.

Cours d'initiation aux standards du Web, avec exercices corrigés

HTML 5 et CSS 3 sont aujourd'hui pris en compte par tous les navigateurs du marché et suscitent un tel niveau d'adhésion que tout professionnel ou futur professionnel se doit de maîtriser ces nouvelles versions des standards du Web sans attendre leur publication définitive par le W3C.

Premier ouvrage en langue française spécifiquement conçu pour les étudiants, ce manuel met l'accent sur les bonnes pratiques de développement (séparation stricte entre structure et présentation), en préconisant l'emploi d'une syntaxe rigoureuse héritée de XHTML. Il consacre une large place aux nouveautés de HTML 5 et CSS 3 : nouveaux éléments de structure sémantique, nouveaux éléments audio et vidéo, Media Queries pour l'adaptation des pages Web au terminal de l'utilisateur (écran d'ordinateur, tablette, smartphone), stockage local, nouveaux sélecteurs CSS, nouveaux types de contours et d'ombrage, arrière-plans multiples et dégradés de couleur, multicolonnage et effets de texte, etc.

Pour vous aider à valider et mettre en œuvre vos connaissances, vous trouverez en fin de chaque chapitre une série d'exercices dont les corrigés et le code source sont disponibles sur les sites www.editions-eyrolles.com et www.funhtml.com.

À qui s'adresse ce livre ?

- Aux étudiants en cursus d'informatique ou de design Web.
- À tous les concepteurs de sites Web souhaitant s'autoformer à HTML 5 et CSS 3.
- Aux enseignants et formateurs à la recherche d'une méthode pédagogique pour enseigner les standards du Web.

Au sommaire

HTML 5. De HTML 4 et XHTML à HTML 5 • Structure d'un document HTML 5 • Structurer le contenu d'une page • Créer des listes • Insérer des images • Insérer du multimédia (audio, vidéo...) • Créer des liens hypertexte • Créer des tableaux • Créer des formulaires. **Feuilles de style CSS 3.** Introduction aux CSS : styles, feuilles de styles, cascade et héritage, unités • Couleurs et images de fond • Créer des bordures, des marges, des espacements et des contours • Les styles du texte (polices internes et externes, taille, graisse et styles, ombrage, interlignage, alignement et espacement, césures, styles des liens...) • Mise en page : dimensionnement, positionnement et transformation des éléments • Style des tableaux • Style des listes • Compléments : Media Queries, stockage local de données, dessin vectoriel SVG. **Annexes.** Aide-mémoire HTML 5 • Aide-mémoire CSS 3 • Codes des couleurs • Entités de caractères • Bibliographie et adresses utiles.

Code article : 613400
ISBN : 978-2-212-13400-1

HTML5 **et** **CSS3**

Cours et exercices corrigés

J. ENGELS. – **PHP 5. Cours et exercices.**
N°12486, 2009, 638 pages.

Autres ouvrages sur HTML et CSS

R. RIMELÉ. – **HTML5.**
N°12982, 2011, 604 pages.

R. GOETTER. – **CSS avancées. Vers HTML5 et CSS3.**
N°13405, 2^e édition, 2012, 386 pages.

R. RIMELÉ. – **Mémento HTML5.**
N°13420, à paraître 2012, 14 pages.

R. GOETTER. – **Mémento CSS 3.**
N°13281, 2011, 14 pages.

J. KEITH, préface de J. ZELDMAN. – **HTML 5 pour les Web Designers.**
N°12861, 2010, 90 pages (A Book Apart).

D. CEDERHOLM. – **CSS 3 pour les Web designers.**
N°12987, 2011, 132 pages (A Book Apart).

Autres ouvrages de développement Web

A. BOUCHER. – **Ergonomie web.**
N°13215, 3^e édition, 2011, 356 pages.

A. BOUCHER. – **Ergonomie web illustrée. 60 sites à la loupe.**
N°12695, 2010, 302 pages.

N. CHU. – **Réussir un projet de site web.**
N°12742, 6^e édition, 2010, 256 pages.

S. BORDAGE, D. THÉVENON, L. DUPAQUIER, F. BROUSSE. – **Conduite de projet Web.**
N°13308, 6^e édition, 2011, 406 pages.

E. MARCOTTE. – **Responsive Web Design.**
N°13331, 2011, 160 pages.

F. DAOUST, D. HAZAËL-MASSIEUX. – **Relever le défi du Web mobile. Bonnes pratiques de conception et de développement.**
N°12828, 2011, 300 pages.

E. SARRION. – **jQuery et jQuery UI.**
N°12987, 2011, 132 pages.

E. SARRION. – **jQuery Mobile.**
N°13388, 2012, 600 pages.

E. DASPET, C. PIERRE DE GEYER. – **PHP 5 avancé.**
N°13435, 6^e édition, 2012, 900 pages environ.

J.-M. DEFRANCE. – **Ajax, jQuery et PHP.**
N°13271, 2011, 500 pages.

G. SWINNEN. – **Apprendre à programmer avec Python 3.**
N°12708, 2^e édition, 2010.

Jean Engels

HTML5 et CSS3

Cours et exercices corrigés

EYROLLES

The logo for EYROLLES, featuring the word "EYROLLES" in a bold, sans-serif font, centered above a horizontal line with a small circle in the middle.

ÉDITIONS EYROLLES
61, bd Saint-Germain
75240 Paris Cedex 05
www.editions-eyrolles.com

En application de la loi du 11 mars 1957, il est interdit de reproduire intégralement ou partiellement le présent ouvrage, sur quelque support que ce soit, sans l'autorisation de l'Éditeur ou du Centre Français d'exploitation du droit de copie, 20, rue des Grands Augustins, 75006 Paris.

© Groupe Eyrolles, 2012, ISBN : 978-2-212-13400-1

Table des matières

Avant-propos	XV
--------------------	----

PARTIE I

Le langage HTML 5	1
--------------------------------	----------

CHAPITRE 1

Introduction à HTML 5	3
------------------------------------	----------

Généalogie de HTML 5	4
-----------------------------------	----------

Les éléments, balises et attributs	4
---	----------

Les attributs de base de HTML	6
-------------------------------------	---

Intérêt de la sémantique	8
--------------------------------	---

Règles de base HTML 5	9
------------------------------------	----------

Un document bien formé	9
------------------------------	---

Un document conforme	10
----------------------------	----

Validation d'un document : le label du WHATWG et W3C	11
--	----

L'environnement de travail	12
---	-----------

Les éditeurs visuels	12
----------------------------	----

Les éditeurs classiques	12
-------------------------------	----

Tests et mise en place du site	13
---	-----------

Référencement du site	15
------------------------------------	-----------

CHAPITRE 2

Structure d'un document HTML 5	17
Les éléments de base	17
La déclaration DOCTYPE	19
L'élément racine <html>	19
L'en-tête d'un document : l'élément <head>	20
Les commentaires	23
Les méta-informations : l'élément <meta />	23
Le corps du document : l'élément <body>	29
Exercices	32

CHAPITRE 3

Structurer le contenu d'une page	35
Les grandes divisions de la page	35
L'élément <div>	35
Les nouveaux éléments HTML 5	37
Les divisions secondaires	43
Les titres et groupes de titres	43
Les paragraphes : l'élément <p>	47
Les articles et les sections	49
Les divisions de bloc locales	51
Les éléments des divisions sémantiques du texte	57
Les styles physiques	64
Mettre un texte en gras	64
Mettre un texte en italique	64
Modifier la taille du texte	65
Créer des exposants et des indices	65
Afficher du texte dans une police à espacement fixe	65
Créer un retour à la ligne	66
Surligner un texte	66
Les listes	67
Les listes ordonnées	68
Les listes non ordonnées	70
L'élément de menu	72

Les listes imbriquées	74
Les listes de définitions	77
Exercices	78
CHAPITRE 4	
Insérer des images et du multimédia	81
Les types d'images	81
L'insertion d'images	82
L'élément 	82
Titre d'une image	86
Les images réactives	87
L'insertion d'images en tant qu'objets	91
Images et boutons	93
L'insertion du multimédia	95
L'insertion d'une animation Flash en HTML 5	96
L'insertion d'une vidéo avant HTML 5	98
L'insertion d'une vidéo en HTML 5	99
L'insertion d'éléments audio avant HTML 5	102
L'insertion d'éléments audio en HTML 5	103
L'insertion d'une applet Java	104
Jauges et barre de progression	106
Exercices	109
CHAPITRE 5	
Créer des liens	111
Les liens à partir d'un texte	111
Les liens externes	112
Les liens ciblés : les ancres	118
Lien à partir d'une image ou d'un bouton	126
Créer plusieurs liens sur la même image	129
Lien déclenchant l'envoi d'un e-mail	132
Les liens déclenchant un script JavaScript	135
Exercices	137

CHAPITRE 6

Créer des tableaux	139
La structure générale d'un tableau	139
Créer des groupes de lignes et de colonnes	141
Les groupes de lignes	142
Les groupes de colonnes	145
Créer des tableaux irréguliers	148
Fusion de colonnes	149
Un cas particulier de fusion de colonnes	151
Fusion de lignes	152
Imbrication de tableaux	154
Organisation d'une page à l'aide de tableaux	157
Exercices	161

CHAPITRE 7

Créer des formulaires	165
Structure d'un formulaire	165
Les attributs de l'élément <form>	167
Les composants communs	169
Bouton d'envoi et de réinitialisation	169
Les composants de saisie d'informations	173
La saisie d'une ligne de texte	173
La saisie de mot de passe	177
La saisie de texte long	179
La saisie d'adresse e-mail	182
La saisie de numéro de téléphone	183
La saisie d'URL	184
La saisie de l'heure, la date, la semaine, le mois	185
La saisie des nombres	189
Choisir une couleur	191
Le composant de recherche	193
Les boutons radio et les cases à cocher	194
Les listes de sélection	197

Les champs cachés	203
Le transfert de fichiers	204
Un formulaire complet	206
Organisation des formulaires à l'aide de tableaux	210
Exercices	214

PARTIE II

Les styles CSS	217
-----------------------------	-----

CHAPITRE 8

Introduction à CSS 2 et 3	219
Créer des styles	219
Les règles générales	219
Les sélecteurs	221
Les sélecteurs d'attributs	226
Pseudo-classes et pseudo-éléments	236
La déclaration !important	243
Écrire des feuilles de style	244
Dans l'élément <style>	244
Dans un fichier externe	244
Dans l'attribut style	245
Cascade et héritage	246
Sélection selon le média	246
Sélection selon le créateur du style	247
Sélection par spécificité	248
Sélection selon l'ordre d'apparition	248
L'héritage	249
Les unités	250
Les unités de longueur	250
Les couleurs	251
Les couleurs et opacités en CSS 3	251
Exercices	253

CHAPITRE 9

Couleurs et images de fond	257
La couleur d'avant-plan	257
La couleur de fond	260
CSS 3 : couleurs de fond en dégradé	262
Dégradé linéaire	262
Dégradé radial	264
Les images de fond	267
Définir une image de fond	267
Positionner une image de fond	269
Un raccourci pour les fonds	276
Les images de fond multiples en CSS 3	277
Exercices	279

CHAPITRE 10

Créer des bordures, des marges, des espacements et des contours	283
Le modèle CSS des boîtes	283
Les bordures	285
Le style de la bordure	285
La largeur de la bordure	288
La couleur de la bordure	291
Définition globale d'une bordure	292
Bordures en CSS 3	294
Bordures multicolores	295
Bordures en images	296
Bordures arrondies	298
À l'ombre des boîtes	300
Les marges	302
Les espacements	304
Les contours	307
Le style du contour	307
La couleur du contour	308

La largeur du contour	308
Exercices	310
CHAPITRE 11	
Style du texte et des liens	313
Les polices	313
Les polices externes en CSS 3	316
La taille des polices	317
Les tailles absolues	318
Les tailles relatives	320
Les tailles dimensionnées	322
Les tailles en pourcentage	323
La graisse du texte	325
Le style des polices	327
Ombrage de texte en CSS 3	331
Régler l'interligne	333
Définir plusieurs propriétés en une fois	334
L'alignement et l'espacement du texte	336
L'alignement horizontal du texte	336
L'indentation du texte	337
L'espacement des mots et des caractères	339
Césure des mots en CSS 3	343
Le style des liens	344
Exercices	347
CHAPITRE 12	
Dimensionnement, positionnement et transformation des éléments	349
Le dimensionnement des éléments	349
Le rendu des éléments	356
Le positionnement des éléments	358
Le flottement	359

Empêcher le flottement	363
Positionnement relatif	364
Positionnement absolu	368
Positionnement fixe	377
Visibilité et ordre d'empilement	379
Affichage multicolonne en CSS 3	382
Redimensionnement par l'utilisateur en CSS 3	385
Transformations des éléments en CSS 3	387
Les translations en CSS 3	388
Les rotations en CSS 3	389
Agrandissement et réduction en CSS 3	391
Déformation de la boîte du conteneur	394
Les transitions des transformations en CSS 3	395
Exercices	397

CHAPITRE 13

Style des tableaux	401
Le modèle de gestion des tableaux	401
Les couleurs des cellules	402
Les titres des tableaux	406
La gestion des bordures des cellules	409
Les bordures séparées	409
Les bordures fusionnées	412
Déterminer la largeur d'un tableau	416
Présentation d'un formulaire	420
Exercices	423

CHAPITRE 14

Style des listes	425
La numérotation des listes	425
Les listes à puces	438
Les puces prédéfinies	439

Les puces graphiques	440
Les listes mixtes	442
Affichage des listes en ligne	444
Affichage d'éléments divers sous forme de liste	446
Exercices	449
CHAPITRE 15	
Compléments	451
Les Media Queries CSS 3	451
Cibler un média précis et adapter les styles	451
Mise en pratique	453
L'impression du contenu	457
Stockage local de données	466
Enregistrement des données	467
Effacement des données	469
Une application de gestion de contacts	470
Dessin vectoriel SVG	473
Inclusion dans le code HTML 5	473
Les formes	474
Le texte	477
Exercices	480
ANNEXE A	
Référence des éléments HTML 5	483
ANNEXE B	
Référence CSS 3	511
Référence des propriétés	511
ANNEXE C	
Codes des couleurs	527
ANNEXE D	
Entités de caractères	533

ANNEXE E

Bibliographie et adresses utiles	537
Bibliographie	537
Adresses utiles	537
 Index	 539

Avant-propos

On ne peut pas encore vraiment qualifier HTML 5 de révolution ; cependant, il marque des avancées dans le domaine de la création de pages web. Avancées en termes de multimédia avec des éléments audio et vidéo natifs affranchissant l'utilisateur des plug-ins habituels. Avancées également, mais peut être moins perceptibles pour le grand public, en termes de sémantique avec l'introduction de nombreux éléments aux rôles prédéterminés qui permettent de structurer précisément les contenus des sites Internet. Son association indispensable avec les feuilles de styles CSS 3 (*Cascading Style Sheets*) rend possible la consultation des pages sur les terminaux les plus divers, allant de l'ordinateur classique à la tablette et au téléphone portable, grâce aux multiples possibilités d'adaptation offertes par CSS.

La conception des sites devant s'effectuer idéalement en deux phases (contenu et présentation), cet ouvrage est lui-même divisé en deux grandes parties.

La première traite du langage HTML 5 en tant qu'outil de structuration des documents. Elle vous permet d'acquérir une bonne connaissance de tous les éléments disponibles dans ce but. À ce stade, et même si nous indiquons la présentation par défaut liée à chacun d'eux, vous ne devriez pas vous préoccuper outre mesure de cet aspect et ne jamais choisir tel élément en fonction de son aspect prédéterminé, mais en fonction de son rôle logique et sémantique dans la structuration du contenu. C'est cette différence fondamentale de conception initiée par XHTML qui est poursuivie heureusement par HTML 5.

Cette première partie comprend les chapitres suivants.

- Le chapitre 1 est une introduction générale qui présente les notions à connaître pour utiliser un langage de balisage tel HTML 5. Il définit les règles pour créer un document conforme aux spécifications du WHATWG (*Web Hypertext Application Technology Working Group*) qui est à l'origine de HTML 5. Il indique également les outils et les procédures à suivre pour mettre en ligne les pages web que vous allez créer.
- Le chapitre 2 vous aide à créer la structure générale d'une page conforme à HTML 5. Cette structure fait apparaître les éléments essentiels qui sont communs à toutes les pages, comme la déclaration DOCTYPE. Vous y trouverez tous ceux qui constituent l'entête d'une page et qui, s'ils ne créent généralement pas de parties visibles dans un navigateur, ont un rôle important souvent négligé. En particulier, ils permettent de lier la page à des ressources externes comme une feuille de style ou des scripts JavaScript.

Certains autres éléments jouent aussi un rôle essentiel dans le référencement de votre site, en incluant des informations sur le document, dites méta-informations. Vous trouverez enfin dans ce chapitre une présentation de l'ensemble des catégories d'éléments incorporables dans le corps d'un document et les types d'inclusions autorisées entre éléments.

- Le chapitre 3 fait tout d'abord un tour d'horizon complet de tous les éléments utiles à la structuration en grandes divisions d'une page. Il aborde ensuite les divisions secondaires en particulier celles qui permettent d'organiser un texte et d'en marquer certains passages. Enfin, nous présentons comment structurer l'information au moyen des différentes formes de listes utilisées couramment dans la création de menus par exemple.
- Le chapitre 4 vous fait découvrir comment insérer les différents types d'éléments multimédias disponibles. Il peut s'agir aussi bien d'images que de musiques ou de vidéos à l'aide des nouveaux éléments audio et vidéo de HTML 5. Nous verrons également comment rendre une image réactive aux clics sur plusieurs zones différentes. Nous aborderons enfin la création de jauge et de barre de progression qui sont aussi des nouveautés HTML 5.
- Le chapitre 5 présente ce qui est la base des documents hypertextes, à savoir la création de liens déclenchés à partir d'un texte, d'un bouton ou d'une image, soit entre les différents éléments d'une même page, soit entre les pages d'un même site pour créer un système de navigation complet. Ces liens peuvent servir également à déclencher le téléchargement de documents externes non affichables dans une page web, à envoyer un e-mail ou encore à lancer un script JavaScript.
- Le chapitre 6 vous propose d'utiliser les différents éléments qui interviennent dans la création de tableaux. Il aborde et définit tout d'abord la structure générale commune à tous les tableaux. Les éléments de création de lignes, de cellules et leurs regroupements sémantiques éventuels, en groupes de ligne ou de colonnes, sont ensuite traités. Les tableaux sont envisagés ici aussi bien pour la structuration de données mais également comme moyen d'organisation d'une page. Nous précisons cependant les limites des tableaux en tant que technique de mise en page, l'usage des styles CSS étant une bonne alternative à ce type d'organisation.
- Le chapitre 7 présente le seul moyen de rendre une page web interactive, en y incorporant des formulaires. Après avoir défini la structure globale de ces derniers, nous décrivons l'ensemble des nombreux nouveaux composants qui permettent de saisir du texte, des mots de passe, des adresses e-mails, des URL, des dates, des nombres, et d'effectuer des choix à l'aide de boutons radio, de cases à cocher ou de listes de sélection d'options. Nous exposons enfin comment réaliser le transfert de fichiers du poste client vers un serveur. Nous montrons également dans ce chapitre comment utiliser des tableaux pour améliorer la structure d'un formulaire.

La seconde partie de cet ouvrage traite de la création des feuilles de styles au moyen de CSS (*Cascading Style Sheets* ou feuilles de styles en cascade). Elles sont le complément indispensable de HTML 5, la séparation du contenu et de la présentation d'un site ayant été bien mise en évidence dans la première partie.

- Dans le chapitre 8, vous découvrirez le fonctionnement des CSS et la syntaxe utilisée dans la création des styles applicables à un document HTML. Ce chapitre constitue une étape essentielle dans l'apprentissage des CSS car il aborde les nombreux sélecteurs CSS 3 qui permettent, entre autre, d'appliquer un même style à toutes les occurrences d'un élément ou d'appliquer des styles différents à un même élément en fonction de son contexte. C'est du bon usage de ces sélecteurs que dépendra toute la puissance et la diversité d'utilisation des styles que vous allez créer par la suite. Nous y étudions enfin les règles d'héritage et de cascade des propriétés CSS.
- Le chapitre 9 aborde les propriétés de gestion des couleurs, aussi bien pour le texte que pour le fond d'un élément, puis celles destinées à la création des images de fond et leurs différents types de positionnement dans tous les éléments HTML. S'ajoutent à cela les différentes améliorations apportées par CSS 3 pour obtenir des images de fond multiples ou des dégradés de couleurs.
- Le chapitre 10 présente tout d'abord le modèle de boîte applicable à tous les éléments CSS, puis traite de la création des bordures (style, épaisseur, couleur, image) applicables à chaque élément individuellement. Nous y abordons également la création des marges entre la boîte d'un élément et son environnement, ainsi que l'espacement entre son contenu et ses bordures. Toutes les propriétés exposées ici sont de nature à affiner la présentation à l'intérieur d'un document. La notion de contour y est également étudiée. Par rapport à la version précédente, CSS 3 propose des bordures multicolores ou contenant des images, des angles arrondis sans avoir recours à des images, comme c'était le cas auparavant, et des ombres très faciles à créer.
- Le chapitre 11 dresse un panorama des propriétés applicables aux textes, qu'il s'agisse du choix d'une police de caractères, des différentes façons de définir sa taille de manière absolue ou relative par rapport au contexte, du choix de sa couleur, de son style, de sa casse, de sa graisse, sans oublier les nombreuses autres possibilités décoratives comme l'ombrage de texte. Nous décrivons aussi la gestion des interlignes, de l'alignement, de la césure, de l'indentation et de l'espacement du texte. Ce chapitre présente enfin les propriétés spécifiques aux liens hypertextes et les sélecteurs spécifiques aux effets dynamiques.
- Le chapitre 12 apporte les éléments essentiels à la présentation et à la mise en page globale d'un document. Nous y étudions les méthodes de dimensionnement des éléments ainsi que les méthodes de positionnement qui sont des avancées essentielles de l'association CSS/HTML. Elles remplacent les méthodes de mise en page habituelles comme celles qui usent et abusent des tableaux ou encore celles qui emploient des cadres. La richesse de ces propriétés permet également d'agir sur la visibilité et l'ordre d'empilement des éléments. Toutes ces propriétés rendent possibles la création des mises en page les plus diverses. De nouvelles propriétés CSS 3 vous donnent aussi la faculté de présenter du texte sur plusieurs colonnes de façon très simple et de laisser l'utilisateur redimensionner un élément. Enfin, dans le domaine des effets visuels, CSS 3 s'est enrichi de propriétés pour transformer les éléments, qu'il s'agisse de translation, de rotation, de changement de taille ou de déformations diverses, tout ceci

pouvant être déclenché par le visiteur de la page. À cela s'ajoute encore l'opportunité de gérer ces modifications afin de créer des animations dynamiques.

- Le chapitre 13 est spécialement dédié aux tableaux qui possèdent un modèle de gestion particulier. Nous montrons comment traiter la couleur des cellules en fonction, par exemple, de leur appartenance à un groupe de lignes ou de colonnes. Nous étudions également la gestion des bordures des cellules, de la détermination de la largeur des colonnes ou de l'ensemble d'un tableau, des alignements spécifiques à l'intérieur des cellules ou des groupes, etc.
- Le chapitre 14 est destiné spécifiquement aux listes et aux menus qui constituent un moyen de structuration efficace, très approprié aux menus par exemple. Nous y abordons les multiples styles de numérotation disponibles pour les listes ordonnées ou à puces graphiques et leur position par rapport aux items. L'emploi des compteurs est une autre façon de numéroter automatiquement des listes générées dynamiquement. La modification du rendu habituel des éléments nous permet de créer des listes en ligne, constituant par exemple des menus horizontaux, ou encore le rendu sous forme de liste d'un ensemble d'éléments dont ce n'est pas la vocation initiale.
- Le chapitre 15 présente plusieurs compléments utiles qui n'ont pas trouvé leur place ailleurs ! Il s'agit d'abord des Media Queries qui constituent l'outil d'adaptation automatique des styles d'une page en fonction du terminal qui la lit, et en particulier en fonction de la taille de son écran qui peut être très variée. Cette adaptation est aussi consacrée aux moyens disponibles pour obtenir un rendu correct du contenu d'un document web à l'impression, et plus généralement sur tout support constitué de pages calibrées (fichiers PDF, présentations de style diaporama...).

Nous abordons ensuite le stockage local pour enregistrer des données, y compris en grande quantité, sur le poste client bien mieux que ne le font les cookies. À titre d'exemple nous y développons une application de gestion de contacts. Enfin, nous donnons une initiation à l'incorporation de dessins vectoriels SVG (*Scalable Vector Graphics*) dans une page HTML 5 pour vous mettre l'eau à la bouche sur cette technologie très riche de possibilités et encore pas assez connue. Sont abordées ici la création de formes géométriques, aussi bien que les styles que l'on peut leur appliquer, et la manipulation du texte vectoriel. D'autres fonctionnalités complémentaires comme les animations SVG seront présentées sur mon site (www.funhtml.com) dans plusieurs nouveaux exemples.

- Enfin, les annexes A, B, C et D proposent des références sur des éléments HTML et de leurs attributs, des propriétés CSS 3 (essentiellement celles qui fonctionnent aujourd'hui) et de leurs valeurs, des codes de couleurs conseillées sur le Web et des entités de caractères.

Les exercices proposés à la fin de chaque chapitre vous proposent de mettre en œuvre immédiatement les points étudiés et de tester l'ensemble des connaissances acquises.

Les corrigés de ces exercices ne figurent pas dans cet ouvrage pour ne pas l'alourdir inutilement, mais ils sont téléchargeables librement sur le site des éditions Eyrolles (www.editions-eyrolles.com) et sur mon site dédié à ce sujet (www.funhtml.com). Ils vous permettront de mesurer votre compréhension des notions abordées.

Pour être tout à fait honnête, je me dois de signaler aussi ce que vous ne trouverez pas dans cet ouvrage et qui existe pourtant dans HTML 5.

- L'élément `<canvas>` qui fait beaucoup de bruit chez certains et qui est une sorte d'application de dessin bitmap dans un navigateur mais qui demande beaucoup de code JavaScript, ce qui n'était pas supposé être un acquis pour les lecteurs.
- Les bases de données SQL Web qui sont un sujet très intéressant qui demandent également des connaissances en JavaScript et SQL que nous n'aurions pas eu la place de développer ici.
- Le drag and drop (glisser-déposer).

Notez que la géolocalisation ne figure pas non plus dans ce livre ; ce n'est d'ailleurs pas du HTML 5.

Sur ces sujets on pourra se référer utilement à d'autres ouvrages aux éditions Eyrolles.

Enfin, vous trouverez aussi sur le site www.funhtml.com au fur et à mesure des nouvelles possibilités des navigateurs, des bonus sur certains points non abordés ici.

Partie I

Le langage HTML 5

1

Introduction à HTML 5

Après XHTML 1.1 on attendait, et moi le premier, une version XHTML 2.0 pour confirmer l'orientation prise par le W3C (*World Wide Web Consortium*), l'organisme qui édite les recommandations des langages du Web. Or, le W3C n'est pas à l'origine de HTML 5. Évolution positive ou régression ? Contrairement aux avis qui courent sur le Web, je dirais qu'il s'agit des deux à la fois, mais aussi que ce langage peut constituer un progrès ; c'est l'objet de ce livre.

Progrès parce que sans se réclamer le moins du monde de XML et d'une rigueur qui était celle de XHTML, il introduit nombre d'éléments qui accentuent son rôle de langage de balisage sémantique. Sémantique parce qu'il n'est plus question d'utiliser le même élément pour des contenus aux sens très différents, mais au contraire d'employer des balises dont le nom indique clairement le contenu. Ainsi, `<header>` est évidemment un en-tête, pas nécessairement celui d'une page, mais toujours un en-tête et pas un pied de page. Il faut reconnaître que cette démarche est la bonne et qu'elle convient parfaitement aux moteurs de recherche dont la tâche d'indexation s'en trouve facilitée.

Régression parce que se trouvent réintroduites des pratiques qui ne devraient pas à mon sens être tolérées par un informaticien. On sait que dans le code d'un programme l'oubli d'un point-virgule en fin de ligne peut bloquer son exécution. L'informatique a des bases mathématiques et réclame donc des habitudes rigoureuses. Un cercle n'est pas un disque : c'est une différence de nature même ! J'ai donc préféré ne pas indiquer toutes les tolérances qu'accepte HTML 5 dans l'écriture du code par rapport aux exigences de XHTML, pour que vous ne les adoptiez pas, dans votre intérêt d'ailleurs. Le laxisme laisse toujours des traces mais c'est peut être encore une idée à contre-courant. En travaillant avec une syntaxe rigoureuse, vous obtiendrez du code propre et qui sera sans doute analysé et affiché plus vite par les navigateurs.

J'ai donc adopté dans ce livre le parti pris de profiter des progrès qu'apporte HTML 5 tout en gardant les bonnes pratiques de formation du code initiées par HTML 4 strict puis XHTML. Le code résultant est évidemment conforme aux spécifications HTML 5 mais, en plus, il est bien écrit.

Généalogie de HTML 5

HTML 5 (*HyperText Markup Language*) est un langage de balisage (dit aussi langage de marquage) qui permet de structurer le contenu des pages web dans différents éléments. Voilà une définition bien abstraite, reconnaissons-le, mais nous y reviendrons en détail dans la section suivante en présentant la notion de balisage.

Historiquement, les langages de balisage sont issus du langage SGML (*Standard Generalized Markup Language*) créé en 1986 pour structurer des contenus très divers. Ce langage s'est révélé trop complexe pour être appliqué tel quel au Web, d'où la nécessité d'en créer une version allégée respectant les mêmes principes essentiels.

L'inventeur du HTML (1992), Tim Berners-Lee, l'avait conçu à l'origine comme un outil de structuration des contenus, principalement textuels, et non pas pour créer des présentations diversifiées. Ce sont les développements successifs, l'essor populaire du Web et la concurrence acharnée entre Netscape et Microsoft pour s'emparer du marché des navigateurs, qui ont détourné HTML de sa vocation première avec l'ajout d'éléments de présentation qui n'avaient rien à y faire. Voulant faire mieux que l'autre, chacun des deux grands a empilé des couches superflues sur HTML. Il faut également reconnaître que l'entrée du Web dans le grand public nécessitait de répondre à une demande d'interfaces graphiques plus esthétiques.

L'absence d'un langage particulier dédié uniquement à la présentation poussait effectivement les webmasters à utiliser tous les moyens pour créer des présentations visuelles agréables. L'apparition de CSS (*Cascading Styles Sheets*), créé en 1996 par Håkon Wium Lie, aurait dû résoudre le problème du détournement de HTML de sa destination première. Mais les mauvaises habitudes étaient prises et la facilité faisait le reste.

L'apparition de HTML 4, et particulièrement de sa version *strict* associée à l'emploi systématique de CSS 2 (publié en 1998), pouvait apporter une solution efficace à ce problème. La création de XML (*eXtensible Markup Language*) en 1998 et son succès dans de multiples domaines d'application ont conduit le W3C à créer le langage XHTML, non plus comme une nouvelle version de HTML, mais comme une reformulation de HTML en tant qu'application XML. Au niveau des éléments et des attributs disponibles, il existait à vrai dire très peu de différences entre HTML 4 strict et XHTML 1.1.

L'impossibilité pour le W3C de trouver un consensus entre les éditeurs de navigateurs et les créateurs de moteurs de recherche pour faire évoluer XHTML a conduit un groupe indépendant, le WHATWG (*Web Hypertext Application Technology Working Group*) dirigé par Ian Hickson (aujourd'hui chez Google !), à entamer le développement de HTML 5.

Les éléments, balises et attributs

Mais au juste comment fonctionne HTML 5 et qu'est-ce qu'un langage de balisage ?

Vous avez sûrement déjà utilisé un traitement de texte tel que Word. Votre texte peut comprendre des titres, des paragraphes, des images, des tableaux, et vous pouvez employer

différentes polices et tailles de caractères dans le même document. Le document final que vous avez réalisé ne laisse apparaître que le résultat de votre mise en page, mais en arrière-plan, votre traitement de texte a enregistré tous les paramètres de mise en page que vous avez utilisés en plus du texte lui-même.

Dans un langage de balisage, tout contenu, qu'il s'agisse de texte, d'images ou d'éléments multimédias les plus divers, doit être enfermé dans un élément. En HTML, chaque élément possède un nom déterminé ; la liste des éléments utilisables est limitative et clairement définie dans la spécification du langage. C'est la grande différence entre HTML et XML, langage dans lequel c'est le programmeur qui crée ses propres éléments selon ses besoins. À quelques exceptions près, un élément a la structure suivante :

```
<nom_element> Contenu </nom_element>
```

Son contenu est précédé par une balise d'ouverture `<nom_element>` et suivi par une balise de fermeture `</nom_element>`. Toutes les balises d'ouverture (ou marqueur) commencent par le signe `<` et se terminent par le signe `>`. La balise de fermeture suit la même règle mais le nom de l'élément est précédé d'un *slash* (`/`). Les navigateurs interprètent donc les contenus en fonction du nom de l'élément et attribuent un style par défaut à chacun de ses contenus.

Les caractéristiques de chaque élément peuvent être précisées par des informations complémentaires que l'on désigne en tant qu'attributs de l'élément. Il peut s'agir par exemple de la définition de la largeur, de la hauteur ou de l'adresse du contenu. Comme nous le verrons dans les sections suivantes, un certain nombre d'attributs sont communs à quasiment tous les éléments de base.

Les attributs d'un élément sont toujours définis dans la balise d'ouverture et doivent être séparés les uns des autres par au moins un espace typographique. Chaque attribut a généralement une valeur, même s'il ne peut prendre qu'une valeur unique. Presque toutes les valeurs ne sont donc pas implicites du moment que l'attribut figure dans la balise d'ouverture. La présence de certains attributs est obligatoire dans quelques éléments particuliers, ce que nous préciserons systématiquement le cas échéant. La plupart du temps, les attributs d'un élément sont facultatifs et il appartient au programmeur de déterminer leur définition par rapport au cas qu'il doit traiter. Nombre d'attributs ont une valeur par défaut. Cela signifie que même si on ne les définit pas dans l'élément, celui-ci se comporte comme si c'était le cas. Il est donc important de connaître ce type d'attribut et de ne pas négliger de les définir avec une autre valeur si ce comportement par défaut n'est pas désiré. La valeur de tous les attributs doit être définie entre guillemets. La syntaxe conforme d'un élément ayant des attributs est donc la suivante :

```
<nom_element attribut1="valeur1" attribut2="valeur2" > Contenu de l'élément  
➡ </nom_element>
```

Le contenu d'un élément peut être constitué de texte ou d'autres éléments qui, eux-mêmes, peuvent en contenir d'autres, et ainsi de suite. Cet ensemble d'inclusion constitue la hiérarchie du document HTML 5.

Les attributs de base de HTML

Dans leur quasi-totalité, les éléments disponibles en HTML ont en commun un ensemble d'attributs ayant chacun le même rôle. Ces attributs se répartissent en trois catégories. Chaque élément peut posséder par ailleurs d'autres attributs particuliers. Quand nous définirons par la suite les différents éléments, nous signalerons s'ils possèdent ces attributs sans rappeler leur définition.

Les attributs globaux (ou noyau)

Ils s'appliquent à quasiment tous les éléments.

- L'attribut `accesskey` définit la touche de raccourci clavier pour accéder à un élément.
- L'attribut `id` sert à identifier un élément de manière unique en lui donnant un nom, soit pour lui attribuer un style, soit pour y faire référence sans ambiguïté dans un script JavaScript.
- L'attribut `class` contient le nom d'une classe CSS qui contient des définitions de styles. Comme nous le verrons dans la seconde partie, son usage est très répandu pour affecter ponctuellement des styles à un élément.
- L'attribut `contenteditable`, qui est un booléen, prend les valeurs `true` ou `false`. Si la valeur est `true` le contenu de l'élément en général textuel est modifiable par le visiteur.
- L'attribut `contextmenu` contient le nom de l'identifiant d'un menu contextuel (créé avec l'élément `<menu>`) qui s'affiche quand on utilise l'élément auquel il est attaché (pas encore opérationnel).
- L'attribut `draggable` prend les valeurs booléennes `true` ou `false` qui indiquent si l'élément est déplaçable dans la page. Peu fonctionnel pour l'instant.
- L'attribut `dropzone` définit les contenus que l'on peut déposer dans l'élément auquel il se rapporte.
- L'attribut `dir` indique le sens de lecture du contenu textuel d'un élément ; il peut prendre les valeurs `ltr` (lecture de gauche à droite) ou `rtl` (de droite à gauche).
- L'attribut `hidden` prend une valeur booléenne `true` ou `false` et indique que l'élément ne sera pas visible dans la page (valeur `true`). Il faudra utiliser un script JavaScript pour modifier sa valeur et le faire apparaître (valeur `false`).
- L'attribut `lang` contient le code de la langue utilisée dans l'élément.
- L'attribut `style` permet de définir un style localement pour un élément donné. Il est encore accepté en HTML mais déconseillé au profit des styles regroupés dans l'élément `<style>` ou dans un fichier externe à l'extension `.css`, ces solutions permettant une maintenance plus facile.
- L'attribut `tabindex` est généralisé à tous les éléments. Sa valeur est un nombre entier qui donne une position à l'élément dans l'ordre des tabulations.
- L'attribut `title` contient un texte qui apparaît dans une bulle quand l'utilisateur positionne le curseur quelques instants (ce n'est pas instantané) sur un élément. Le texte qu'il contient peut servir à fournir une information ou une explication sur le rôle de l'élément.

Les attributs de gestion d'événements

Ces attributs permettent de gérer les événements dont un élément peut être le siège et qui sont créés par l'utilisateur. Leur contenu est un script écrit en général en langage JavaScript. HTML 5 définit un grand nombre d'attributs de gestion d'événements, y compris pour des éléments qui ne peuvent pas être le support de ces événements. Il appartient donc aux programmeurs d'effectuer des tests pour vérifier la réalité des événements pour un élément donné. Vous trouverez ci-après la liste des gestionnaires de base et la description de l'événement correspondant.

Tableau 1-1. Les attributs gestionnaires d'événements communs (suite page 8)

attribut	Action de l'utilisateur
onabort	Arrêt d'une opération
oncanplay	Un fichier multimédia peut être lu (au moins son début)
oncanplaythrough	Un fichier multimédia peut être lu entièrement
onclick	Clic sur le contenu de l'élément
oncontextmenu	Un menu contextuel est déclenché
ondblclick	Double-clic sur le contenu de l'élément
ondrag	Un élément est en cours de déplacement
ondragend	Le déplacement est fini
ondragenter	Un déplacement a une cible qui l'accepte
ondragleave	L'origine du déplacement est acceptable
ondragover	Un élément déplacé survole l'élément concerné par l'attribut
ondragstart	Début du déplacement
ondrop	Dépose d'un élément
ondurationchange	Modification de l'attribut <code>duration</code> d'un élément multimédia
onemptied	Un élément multimédia n'est plus utilisable (chargement incomplet ou interrompu)
onended	Fin de lecture d'un son ou d'une vidéo
oninput	Un élément de formulaire est complété
oninvalid	Une saisie est non valable (par exemple en dehors d'un intervalle précisé)
onkeydown	Maintien d'une touche enfoncée
onkeypress	Frappe sur une touche
onkeyup	Relâchement d'une touche enfoncée
onloadeddata	Une donnée est chargée (comme un son, une vidéo)
onloadedmetadata	Une métadonnée est chargée (comme un script, des styles externes)

attribut	Action de l'utilisateur
onloadstart	Début d'un téléchargement d'un média
onmousedown	Enfoncement d'un bouton de la souris
onmousemove	Le curseur de la souris bouge dans la zone de l'élément
onmouseout	Le curseur de la souris quitte la zone de l'élément
onmouseover	Le curseur de la souris est au-dessus de la zone de l'élément
onmouseup	Relâchement d'un bouton de la souris au-dessus de la zone de l'élément
onpause	Pause dans la diffusion
onplay	Démarrage de la lecture
onplaying	Média prêt pour la lecture
onprogress	Média en cours de chargement
onratechange	Modification de la vitesse de lecture
onscroll	Défilement d'un contenu
onstalled	Erreur pendant le chargement d'un média
onsuspend	Arrêt déclenché avant la fin d'un chargement
ontimeupdate	La position actuelle de lecture est modifiée par l'utilisateur
onvolumechange	Modification du volume sonore
onwaiting	Arrêt de la lecture en attente de la disponibilité de la suite

Intérêt de la sémantique

Il s'agit bien plus selon moi d'un changement de pensée et d'organisation qui doit s'opérer dans la création des pages web.

Une page web créée avec HTML doit être pensée en distinguant deux parties.

- Un contenu, structuré au moyen des éléments HTML (grandes divisions, titres, paragraphes, tableaux, images et liens, etc.). À ce stade, et même s'il en a déjà une idée, le créateur ne doit pas nécessairement avoir une vue définitive de la présentation finale. Il lui faut maîtriser principalement l'organisation des informations à fournir à un utilisateur.
- Une feuille de style CSS, définissant la mise en page de ces éléments en fonction du média qui va opérer le rendu du contenu (polices et tailles de caractères, bordures, marges, couleurs, positionnement dans la page, etc.). Les médias se diversifiant en effet de plus en plus en devenant des éléments portables dotés de petits écrans, le traditionnel écran d'ordinateur n'est plus le principal vecteur d'affichage d'une page web. La séparation du contenu et de la présentation étant réalisée, il est possible d'associer à chaque média une feuille de style adaptée au terminal.

L'utilisation de ces méthodes présente les avantages suivants.

- Une meilleure organisation du contenu.
- Une meilleure qualité du code et une plus grande rapidité d'affichage sur les navigateurs récents (Firefox, Internet Explorer, Opera, Safari, Chrome...).
- Une réduction des coûts de développement et de maintenance des sites web ainsi qu'une réutilisabilité accrue et rapide du code. En effet, en ayant respecté les principes précédents, il est très facile de modifier rapidement toute la présentation d'une page sans toucher au code HTML.

Les spécifications HTML et CSS sont aujourd'hui incontournables pour tous ceux qui veulent concevoir un site web de manière professionnelle, et tous les étudiants en informatique et les professionnels du Web se doivent d'acquérir ou de mettre à jour leurs connaissances sur ces techniques.

Règles de base HTML 5

Un document bien formé

Un document HTML doit respecter certaines règles simples.

- Les éléments et les attributs sont insensibles à la casse. Par exemple, `<body>` et `<BODY>` sont acceptés. Choisissez une casse selon votre goût et conservez-la. Par habitude du XHTML, tout le code de ce livre est en minuscules.
- Les éléments non vides doivent avoir une balise d'ouverture et une balise de fermeture. Cela facilite la tâche des navigateurs. Par exemple, plutôt que d'écrire :

```
<ol>
  <li>Item 1
  <li>Item 2
```

on préférera le code suivant :

```
<ol>
  <li>Item1 </li>
  <li>Item2 </li>
</ol>
```

- Les éléments vides ne comportent qu'une seule balise et il est plus correct de les terminer par les caractères `</>` précédés d'un espace pour marquer la fin de l'élément. Par exemple, plutôt que d'écrire :

```
<img src= "monimage.gif"> <hr> <br>
```

on préférera le code suivant :

```
<img src= "monimage.gif" /> <hr /> <br />
```

- Les éléments ne doivent pas se chevaucher et donc obéir au principe premier ouvert, dernier fermé. Dans ce cas, le premier élément est le parent du second et celui-ci est enfant du premier. Par exemple, le code suivant est incorrect :

```
<div> Cette division contient un titre <h1> Important ! </div> </h1>
```

et doit être remplacé par :

```
<div> Cette division contient un titre <h1> Important ! </h1></div>
```

- Tous les attributs doivent avoir une valeur incluse entre guillemets ("). Les différents attributs du même élément doivent être séparés par au moins un espace. Par exemple, plutôt que d'écrire :

```
<p class=styleperso title=attention> Texte important</p>
```

on préférera le code suivant :

```
<p class="styleperso" title="attention" > Texte important</p>
```

- Une valeur doit être donnée à tous les attributs utilisés, y compris à ceux dont la valeur est unique. Par exemple, plutôt que d'écrire :

```
<input type= "checkbox" checked disabled />
```

on préférera le code suivant :

```
<input type= "checkbox" checked="checked" disabled="disabled" />
```

Il y a des exceptions car certains attributs ne doivent pas avoir de valeur en particulier quand elle est booléenne. Dans ce cas, la présence seule du nom de l'attribut signifie implicitement la valeur `true` et l'absence de la valeur `false`.

- Les scripts et les feuilles de style qui contiennent les caractères `<` et `&` peuvent figurer dans des sections CDATA de la façon suivante :

```
<script type="text/javascript">  
< ![CDATA[  
Code du script...  
]]>  
</script>
```

- Une autre solution efficace consiste à placer les scripts ou les feuilles de style contenant ces caractères dans des fichiers séparés et à les inclure à l'aide de l'élément `<link>` sur lequel nous reviendrons en détail par la suite.

Un document conforme

Un document HTML 5 se doit également de respecter les règles d'inclusion des éléments les uns dans les autres, telles qu'elles sont définies dans la spécification. En effet, elle définit la liste limitative de tous les éléments HTML 5 utilisables et énumère ceux qui peuvent y être

inclus. Le respect de ces contraintes est impératif pour que le document soit déclaré conforme par le validateur. Vous trouverez à cet effet tout au long de cet ouvrage lors de la description des éléments, et dans l'annexe A, pour chaque élément, la liste de ses éléments enfants (ceux qu'il peut inclure) et de ses éléments parents (ceux dans lesquels il peut être inclus).

Validation d'un document : le label du WHATWG et W3C

Malgré toutes les vérifications auxquelles vous pouvez procéder personnellement, il peut rester des erreurs de conformité dans votre code. Comme peut le faire un compilateur qui signale les éventuelles erreurs de syntaxe, les validateurs du W3C et du WHATWG permettent de vérifier si le code est bien formé et conforme. Pour lancer cette validation automatique, vous devez soumettre l'URL ou le code de vos documents HTML au validateur du W3C accessible à l'adresse suivante : <http://validator.w3.org>, ou à celui du WHATWG à l'adresse : <http://html5.validator.nu/>.

La figure 1-1 montre la page de validation du site du whatwg.org dans laquelle plusieurs moyens sont mis à disposition pour valider un document. Dans la liste déroulante (repère ❶) vous pouvez choisir entre saisir l'URL de la page si le document est déjà transféré sur un serveur, choisir le fichier sur votre ordinateur, ou encore copier le code à vérifier directement dans une zone de saisie (repère ❷). Le bouton ❸ entraîne la vérification, et la récompense est l'annonce de la validité du code (repère ❹).

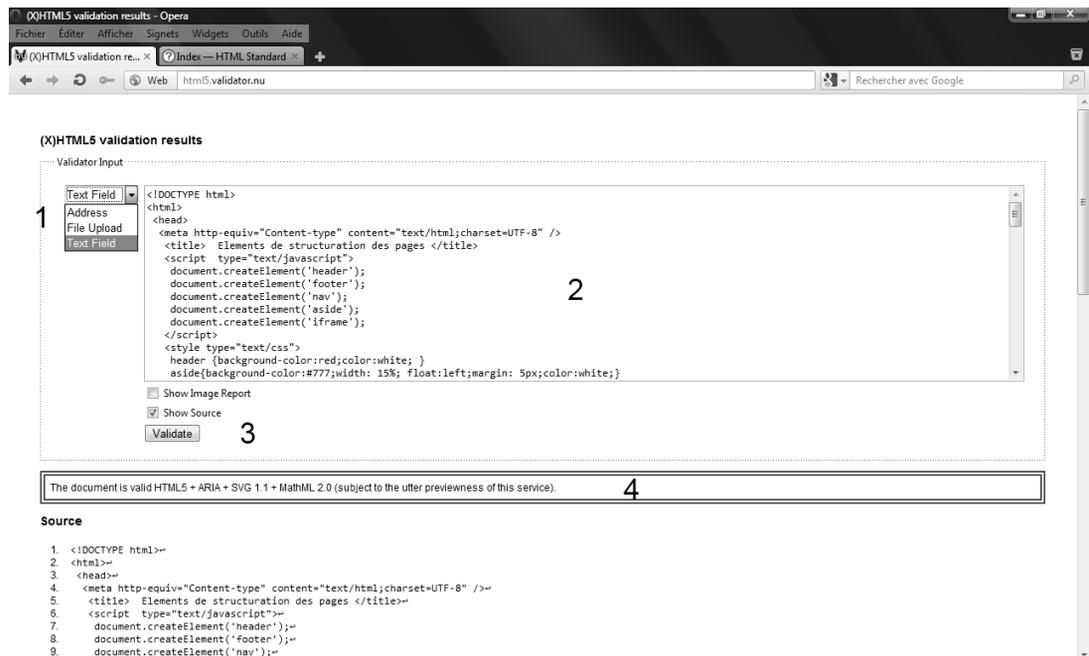


Figure 1-1

La page de validation du WHATWG

L'environnement de travail

Pour créer des pages web avec HTML 5 et CSS 3, il faut être doté d'un environnement de travail adapté. En théorie, un simple éditeur de texte tel que le Bloc-notes de Windows ou Emacs sous Linux peut suffire. Cependant, comme la saisie des noms des différents éléments HTML 5 dans ce type d'éditeur peut devenir à la longue plutôt rébarbative, nous mentionnons quelques outils susceptibles de faire gagner du temps.

Les éditeurs visuels

Dans ce type d'éditeurs, vous travaillez graphiquement sur une page en y incluant des éléments sans saisir une ligne de code. Le plus connu est Dreamweaver, mais il en existe bien d'autres. L'inconvénient de ces éditeurs tient en fait à ce qui paraît être leur avantage : le code que l'on ne saisit pas est généré automatiquement et rien ne garantit qu'il convienne ou qu'il soit conforme aux dernières spécifications. De plus, toute génération automatique éloigne du travail de programmation et ne présente pas un avantage évident pour la création de pages, alors que ce peut être le cas dans un éditeur dédié à un langage de programmation, Java par exemple, et qui peut décharger le programmeur de tâches répétitives.

Outre le temps d'apprentissage et son prix élevé, je ne pense pas qu'il soit utile de faire appel à ce genre d'éditeurs qui auraient tendance à vous rendre passif.

Les éditeurs classiques

Entre le Bloc-notes Windows et Dreamweaver, certains éditeurs offrent un compromis en apportant à la fois une aide à la saisie, qui dispense de taper soi-même le nom des éléments, tout en autorisant de les choisir librement. Outre le fait qu'ils présentent souvent l'avantage d'être gratuits, ces éditeurs, très nombreux et disponibles en téléchargement, permettent de créer rapidement la structure d'un document en utilisant un squelette commun à toutes les pages HTML. Nous donnerons la structure de base d'une telle page au chapitre 2. Il vous restera ensuite à inclure dans le corps de la page les différents éléments qui vont structurer son contenu. Si vous avez procédé à une analyse préalable sur le papier, comme il se doit avant tout codage, cette phase de travail d'inclusion sera très rapide. Parmi les nombreux éditeurs, j'ai choisi pour ma part HTMLPad qui s'avère très pratique pour incorporer rapidement les différents éléments HTML 5 et propriétés CSS 3. Pas tout à fait gratuit mais bon marché, vous pouvez le télécharger y compris en version d'essai complète sur le site : <http://www.blumentals.net/htmlpad/>.

La figure 1-2 montre l'aspect de l'environnement de travail qu'il fournit. La fenêtre de l'éditeur est divisée en plusieurs zones. La zone ❶ contient la liste des éléments HTML 5 et un double-clic sur l'un d'eux incorpore le code dans la page située au centre à la position du curseur (zone ❷). La zone ❸ contient l'ensemble des propriétés CSS 3 et leurs valeurs quand elles sont à choisir dans une liste ; on peut ainsi incorporer d'un clic n'importe quel style sans erreur de saisie. Cet outil peut faire gagner beaucoup de temps dans la création d'une page.

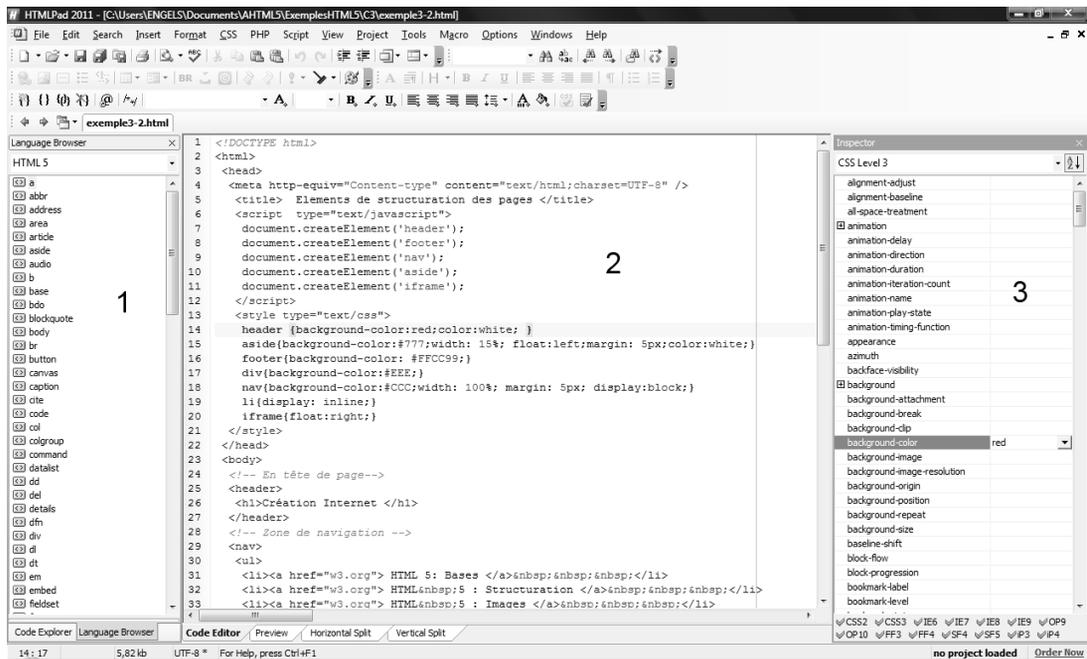


Figure 1-2
L'éditeur HTMLPad

Tests et mise en place du site

Une fois que vous aurez créé l'ensemble des pages de votre site, il faudra vous assurer qu'elles ont un aspect identique ou du moins très semblable dans les différents navigateurs du marché et les résolutions d'écrans afférentes. Ce dernier point doit d'ailleurs avoir fait l'objet d'un choix initial, en particulier si les dimensions retenues pour les différents éléments de la page sont définies de manière fixe (en pixels par exemple), et non en pourcentage, ce qui représente la meilleure solution. Choisir de créer des mises en page en ciblant le format le plus répandu (1 024 par 768 pixels actuellement par exemple) peut produire des résultats désagréables si le poste client est en 800 par 600 pixels ou moins encore.

Si cette phase de test vous amène à constater des résultats divergents et indésirables, c'est sans doute que vous avez utilisé des fonctionnalités non encore prises en compte par un navigateur particulier, et le plus souvent il s'agira de propriétés CSS encore marginales dans Internet Explorer par exemple. Il vous appartiendra alors de renoncer à certaines d'entre elles, au moins provisoirement, pour assurer l'universalité de vos pages si tel est votre objectif. Nous signalons à cet effet dans la seconde partie de cet ouvrage les propriétés qui risquent de poser problème dans certains navigateurs.

Une fois cette phase de test effectuée, il ne reste plus qu'à transférer l'ensemble des pages du site vers le serveur distant qui va les héberger pour les mettre à la disposition de tous. Vous devez pour cela utiliser un logiciel de transfert FTP (*File Transfert Protocol*).

Les hébergements gratuits

Les hébergeurs gratuits de sites personnels peuvent mettre à votre disposition des pages spécialisées permettant ce transfert via une interface web simple. Si vous disposez d'un nom de domaine sur un serveur dédié ou mutualisé, il vous faut généralement utiliser un logiciel FTP.

Il existe de nombreux logiciels de ce type, gratuits ou payants. Je recommande pour ma part d'utiliser FileZilla téléchargeable sur le site <http://filezilla.fr/>, gratuit et très pratique. La figure 1-3 présente l'interface de FileZilla. Pour établir une connexion avec le serveur et acquérir les droits en vue d'effectuer les transferts de vos fichiers HTML, de vos images ou des supports multimédias, vous devez connaître les paramètres de connexion qui sont fournis par l'hébergeur du site. Il s'agit des données suivantes.

- Hôte : adresse du serveur FTP, par exemple `ftp.funhtml.com` (repère ❶).
- Identifiant : par exemple `funhtml` (repère ❷).
- Mot de passe d'accès au site (repère ❸).
- Éventuellement, indiquez le port de communication (repère ❹) (par défaut, il s'agit de la valeur 21).

Si ces paramètres sont correctement reconnus, vous avez accès à un gestionnaire de fichiers côté serveur similaire à celui que vous pouvez connaître dans Windows ou Linux (repère ❺). Dans la partie gauche, vous trouvez le même type de gestionnaire de fichiers pour sélectionner les fichiers présents sur votre ordinateur (repère ❻). Il vous suffit alors de réaliser une opération de glisser-déposer entre ces deux zones pour que le transfert commence. Sa durée dépend évidemment de la taille du fichier et de la vitesse de votre connexion. Le dossier principal côté serveur est nommé `www`. C'est dans ce dossier que le serveur ira chercher le fichier nommé `index.html` ou `index.htm` en réponse à la requête générale effectuée sur votre site sous la forme `http://www.votresite.com`. C'est dans ce fichier que vous devez créer la page d'accueil du site (ou la page `index.php` si elle contient du code PHP). Dans ce dossier `www`, vous pouvez créer autant de sous-dossiers que vous le désirez. Si vous constituez par exemple un sous-dossier nommé `html`, son contenu principal devra figurer dans un nouveau fichier nommé encore `index.html`, et il sera alors accessible directement par la requête `http://www.votresite.com/html`. Dans les deux cas (répertoire principal ou sous-répertoire), si vous ne créez pas de fichier nommé `index.html`, l'utilisateur devra écrire une requête complète, comprenant le nom du fichier auquel il veut accéder, de la forme `http://www.votresite.com/html/page1.html`.

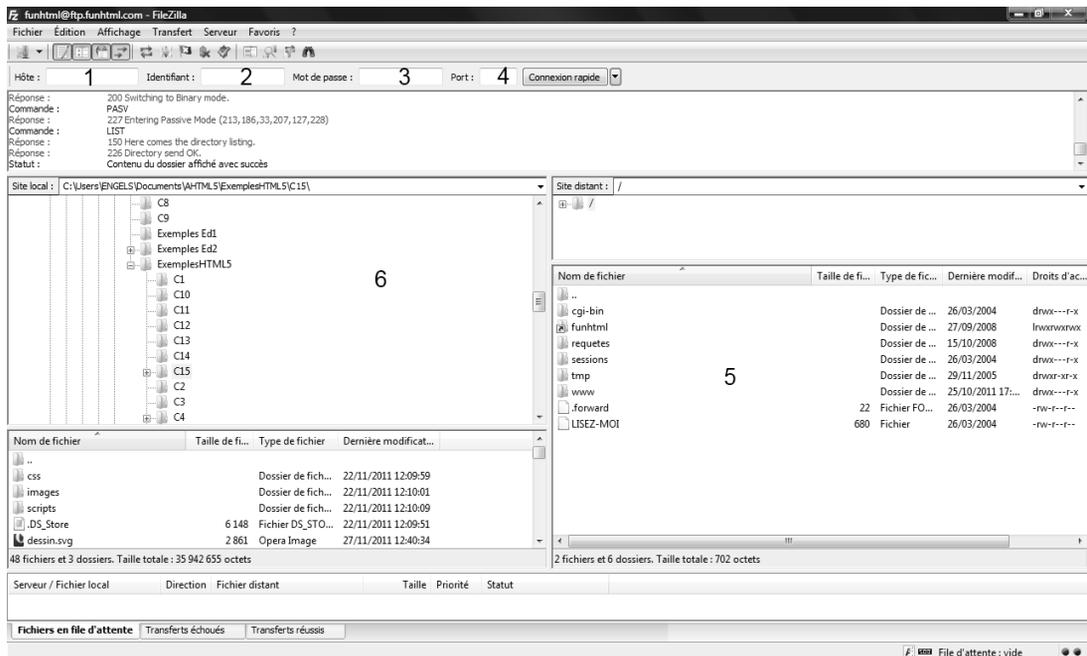


Figure 1-3
L'interface de FileZilla

Référencement du site

Après avoir créé votre superbe site conforme à HTML et CSS, il reste encore une phase importante à accomplir, à savoir le faire connaître. À moins de disposer de moyens publicitaires conséquents, la fréquentation souhaitée ne se produira pas du jour au lendemain. Même si c'est l'adaptation du contenu à un besoin qui fait venir et revenir les internautes, il vous faut d'abord faire connaître l'existence de votre site.

Le moyen le plus simple, et généralement gratuit, est de le référencer dans les annuaires et moteurs de recherche les plus connus comme Google ou Yahoo! afin qu'il soit bien placé en réponse à la recherche d'un internaute. Des ouvrages entiers sont consacrés au référencement, apportant des conseils avisés pour augmenter la fréquentation d'un site. Je recommande le site <http://www.abondance.com> ou de se procurer l'ouvrage *Réussir son référencement Web* d'Olivier Andrieu aux éditions Eyrolles portant sur le référencement. Nous retiendrons cependant les quelques points suivants.

- Si vous achetez un nom de domaine – ce qui, lié à un hébergement mutualisé, est aujourd'hui possible à bon marché et présente mieux que l'hébergement gratuit offert par votre fournisseur d'accès dont les adresses sont très longues –, choisissez de préférence un nom court et facile à mémoriser. Les noms longs, et particulièrement s'ils sont composés de plusieurs mots, posent soucis à l'utilisateur qui se demande alors si

les mots se suivent ou sont séparés par des tirets, d'où des interprétations divergentes et des erreurs. Certains moteurs de recherche affichent les sites répondant aux mêmes critères par ordre alphabétique et il vaut mieux que votre nom de domaine commence par a plutôt que par z.

- Les extensions `.com` (ou `.fr` en France) sont préférables à `.org` ou `.net` car ce sont les premières qui viennent à l'esprit des internautes s'ils ont oublié l'extension réelle. La notion de site commercial qui était liée à l'extension `.com` a désormais disparu, et votre site peut présenter aucun aspect commercial et posséder cette extension.
- La définition d'un maximum de mots-clés rapportant objectivement le contenu de votre site est essentielle. Elle doit être réalisée à l'aide de l'élément (voir le chapitre 2) :

```
<meta name="keywords" content="liste des mots clés" />
```

qui est situé dans l'en-tête du document. Cette liste mérite toute votre attention car elle est utilisée par les moteurs de recherche pour indexer vos pages et mettre en adéquation la demande d'un internaute avec les sites qui lui correspondent. Les mots-clés se doivent de refléter fidèlement le contenu du site et les idées qui lui sont associées, et rien d'autre. Inutile par exemple d'identifier les mots les plus recherchés et de les inclure dans votre liste pour attirer du monde. Outre que vous risquez de tromper les internautes, vous pouvez surtout les décevoir si votre contenu ne correspond pas à leur attente. En revanche, n'hésitez pas à fournir une longue liste de mots-clés avec leurs variantes masculin/féminin et singulier/pluriel, ainsi que les mots dérivés qui peuvent correspondre à votre contenu. Vous augmenterez ainsi vos chances de figurer en bonne place dans les résultats des recherches effectuées par les internautes. Les moteurs de recherche se basant aussi sur le contenu des pages, rien ne vous empêche de faire apparaître plusieurs fois dans le contenu les mots importants. Une astuce consistait à répéter ces mots dans la page et à les écrire de la même couleur que le fond de la page en créant des styles CSS appropriés et non plus les balises et les attributs utilisés pour définir les couleurs. Leur répétition est ainsi invisible dans la page, mais perçue par les robots qui analysent le texte.

- Référez votre site dans tous les principaux moteurs de recherche. Et si vous avez les moyens, Google se fera un plaisir de vous vendre une bonne place dans ses résultats de recherche très objectifs.

2

Structure d'un document HTML 5

Avant de créer des pages web et de leur donner un contenu, nous allons déterminer une structure générale commune à toute page en conformité avec les spécifications HTML 5. En fonction des besoins, les codes des exemples 2-1 à 2-4 serviront de base à la constitution de toutes nos pages. Il suffira donc de les copier dans votre éditeur préféré, puis de les compléter avec un contenu particulier pour chaque page.

Les éléments de base

Le langage HTML 5 est une amélioration du langage HTML 4, avec des simplifications par rapport à la version XHTML qui était de mise avant lui. Tout document peut donc débiter de la même manière par la déclaration suivante (exemple 2-1 repère ❶) :

```
❶ <!DOCTYPE html>
```

Vient ensuite l'élément racine `<html>` (repère ❷) qui inclut les éléments `<head>` (repère ❸) et `<body>` (repère ❹). Chacun de ces éléments a un contenu et donc une balise d'ouverture et une balise de fermeture, `<head>` incluant obligatoirement l'élément `<title>` (repère ❺) et un élément `<meta />` (repère ❻) qui contient la définition du jeu de caractères utilisé dans la page, et `<body>` ayant au moins un élément enfant ; ici, il s'agit de `<h1>` (repère ❼). La structure minimale d'un document HTML 5 est donc semblable à celle de l'exemple 2-1. Le fichier contenant ce code doit avoir une extension `.html` ou `.htm`.

Exemple 2-1 Structure minimale d'un document HTML 5

```
<!DOCTYPE html> ❶  
<html> ❷  
  <head> ❸  
    <meta http-equiv="Content-type" content="text/html; charset=UTF-8" /> ❹  
    <title> HTML 5 et CSS 3 </title> ❺  
  </head>  
  <body> ❻  
    <!-- Tout le contenu de la page -->  
    <h1>Le corps de la page minimale</h1> ❼  
  </body>  
</html>
```

Nous retrouvons bien dans cet exemple la structure arborescente décrite au chapitre 1. L'élément racine, au sens XML du terme, est `<html>` et inclut les éléments `<head>` et `<body>`. L'élément `<head>` contient l'élément `<title>` qui est obligatoire ainsi que la déclaration du jeu de caractères dans un élément `<meta />`; l'élément `<body>`, qui ne doit pas être vide (ce qui est évident), contient un titre de niveau 1 `<h1>` sur lequel nous reviendrons plus loin. Du point de vue hiérarchique, `<html>` est bien le parent ou l'ancêtre de tous les autres.

Les commentaires

Tout ce qui est contenu entre les symboles `<!--` et `-->` est considéré par le navigateur comme des commentaires et n'est pas affiché dans la page, même s'ils se trouvent dans l'élément `<body>`. Comme pour tout langage de programmation, nous avons tout intérêt à commenter le code HTML afin d'en faciliter la relecture a posteriori. Notez toutefois que les commentaires seront visibles par l'internaute si celui-ci choisit d'afficher le code source de la page dans son navigateur. Évitez donc d'y inclure des informations confidentielles et d'y faire figurer des informations privées.

Un document HTML 5 peut incorporer du code PHP pour créer des pages dynamiques (interaction avec une base de données et création automatique de pages). Dans ce cas, le code PHP est compris entre les balises `<?php` et `?>`. Notez que le code de l'exemple 2-2 passé au validateur n'est pas déclaré conforme car il ne reconnaît pas ces balises, mais le code créé par l'exemple sera conforme.

Pour éviter les problèmes d'interprétation divergente entre les différents navigateurs, nous utiliserons systématiquement la déclaration du jeu de caractères avec l'élément `<meta />` dans chaque document. La structure minimale de ce type de page est donc celle de l'exemple 2-2. Notez que le fichier PHP a une extension propre, du type `.php` ou `.php5` par exemple, toujours en fonction de la configuration du serveur. Pour que le document HTML, que le serveur va finalement envoyer au navigateur, soit conforme au standard, il faut que le premier script placé au début du document (repère ❶) ne crée aucun code HTML (il peut par exemple ne contenir que des fonctions) et que le second (repère ❷) ne crée que du code HTML conforme. En respectant ces conditions, il n'y a aucune limite à l'utilisation de scripts PHP à l'intérieur d'un document.

Exemple 2-2. Structure d'une page HTML 5 contenant un script PHP

```
<?php ❶
// Placez ici du code PHP
?>
<!DOCTYPE html>
<html lang="fr">
  <head>
    <meta http-equiv="Content-type" content="text/html;charset=UTF-8" />
    <title> HTML 5 et CSS 3 </title>
  </head>
  <body>
    <!-- Le corps de la page -->
    <h1>Le corps de la page</h1>
    <?php ❷
      echo "<h2> Placez ici du code PHP créant un titre</h2>";
    ?>
  </body>
</html>
```

Dans toute la suite de cet ouvrage, nous n'utiliserons que la structure de base présentée à l'exemple 2-1.

La déclaration DOCTYPE

Nous avons déjà indiqué que le code d'une page HTML devait se conformer à des règles précises. La déclaration `DOCTYPE`, obligatoire dans tout document, précise le type de document qui va être créé. Dans HTML 5, cette déclaration est désormais réduite à sa plus simple expression par rapport à XHTML :

```
<!DOCTYPE html>
```

Dans ce code, la partie `html` donne le nom de l'élément racine du document.

L'élément racine <html>

L'élément `<html>` est l'élément racine du document, au sens XML du terme. C'est donc lui qui est le parent de tous les autres, soit directement, comme `<head>` et `<body>`, soit indirectement par l'intermédiaire de ces derniers. Il est donc le conteneur de premier niveau placé en haut de la hiérarchie de tous les éléments du document. Il n'existe que deux éléments enfants de l'élément `<html>`. En HTML 5, son contenu est constitué de l'en-tête du document, introduit par la balise `<head>` et terminé par la balise `</head>`, puis par le corps du document introduit par `<body>` et terminé par `</body>`, comme nous pouvons le vérifier dans les exemples.

L'élément racine possède les attributs communs dont les plus utiles sont :

- l'attribut `lang` dont la valeur est un code de langue normalisée qui indique la langue utilisée par défaut dans la page. Cette valeur sera reconnue par les moteurs de recherche

pour leur permettre d'indexer les pages du site en effectuant un tri par langue. Elles n'apparaîtront dans Google par exemple que si l'utilisateur a choisi le bouton France ;

- l'attribut `dir` qui indique le sens de lecture du texte de la page. Il peut prendre les valeurs `ltr` pour le texte qui se lit de gauche à droite (langues européennes) ou `rtl` pour le texte qui se lit de droite à gauche (langues orientales : hébreu, arabe).

Un élément `<html>` complet tel que nous pouvons l'utiliser s'écrira donc :

```
<html lang="fr" dir="ltr">
  <!--suite des éléments inclus -->
</html>
```

En pratique, pour des sites ayant un contenu dans une langue européenne, nous omettrons l'attribut `dir` ; dans nos exemples, nous n'utiliserons pas systématiquement l'attribut `lang`.

L'en-tête d'un document : l'élément `<head>`

L'élément `<head>` englobe un certain nombre d'informations utiles au bon affichage de la page web. Ces informations dites métadonnées sont contenues dans six éléments différents qui ont chacun un rôle bien déterminé. Il s'agit des éléments `<base/>`, `<link />`, `<meta />`, `<script>`, `<style>` et `<title>` dont nous allons étudier les rôles respectifs.

Aucun d'eux n'a de répercussion directement visible dans la page et seul le contenu de l'élément `<title>` sera visible, non dans la page mais dans la zone de titre du navigateur. Le bloc d'en-tête a donc la structure suivante, dans laquelle seuls les éléments `<title>` et `<base />` ne doivent figurer qu'une seule et unique fois, les autres n'ayant pas de limites.

```
<head>
  <title>Titre de la page</title>
  <meta http-equiv="Content-type" content="text/html;charset=UTF-8" />
  <base href="http://www.monsite.com" />
  <link rel="shortcut icon" type="images/x-icon" href="../images/favicon.ico" />
  <meta name="Author" content="Jean ENGELS" />
  <script type="text/javascript">
    <!-- Scripts JavaScript -->
  </script>
  <style type="text/css">
    <!-- Styles CSS -->
  </style>
</head>
```

Remarquons d'emblée que seuls les éléments `<title>`, `<script>` et `<style>` ont un contenu, et donc une balise fermante.

Nous allons maintenant détailler le rôle de chacun des éléments inclus dans `<head>`.

L'adresse de base : l'élément `<base />`

Il s'agit d'un élément vide et n'a donc pas de balise de fermeture. L'information qu'il contient est donnée dans son unique attribut `href` dont l'utilisation est obligatoire. Le

contenu de cet attribut est une URL qui fournit l'adresse de base de tous les fichiers utilisés dans la page quand leur adresse est transmise de manière relative. Si nous écrivons le code suivant :

```
<base href="http://www.funhtml.com/" />
```

le navigateur ira chercher une image dont l'URL est indiquée sur le serveur par /html/images/monimage.gif à l'adresse :

```
http://www.funhtml.com/html/images/monimage.gif
```

L'élément `<base />` possède également l'attribut commun `id`, qui ne peut servir qu'à modifier la valeur de l'attribut `href` au moyen d'un script JavaScript, selon la syntaxe suivante :

```
document.getElementById(id_element).href='valeur'
```

Les documents liés : l'élément `<link />`

Comme le précédent élément, il s'agit d'un élément vide dont l'information est contenue dans ses attributs. Il permet d'établir un lien entre la page HTML 5 en cours et un autre document externe nécessaire à la page. Nous l'utiliserons particulièrement pour lier la page à une feuille de style CSS contenue dans un fichier ayant l'extension `.css` ou un script JavaScript contenu dans un fichier sous l'extension `.js`.

L'utilisation de l'élément `<link />` crée l'incorporation virtuelle de ces documents dans le code de la page web. On parle d'incorporation virtuelle car la page se comportera comme si le code des fichiers externes faisait partie de la page, le contenu de ces fichiers n'étant pas visible, même en affichant le code source de la page.

La liaison avec les fichiers externes est déterminée par les attributs `rel`, `type`, `href`, `hreflang`, `media` et `sizes`.

- L'attribut `rel` indique le nom de la relation à établir avec le fichier externe. Il peut prendre les valeurs suivantes :
 - `rel="stylesheet"` si le fichier externe est une feuille de style.
 - `rel="alternate"` si le fichier est une page alternative (de rechange, proposée aux visiteurs dans les navigateurs).
 - `rel="shortcut icon"` ou `"icon"` pour faire référence à l'icône identifiant le site et qui s'affiche devant l'adresse dans les navigateurs les plus conformes.
 - `rel="previous"` ou `rel="prev"` si le fichier désigné est la page précédente dans l'ordre normal de consultation du site.
 - `rel="next"` si le fichier est la page suivante dans l'ordre normal de consultation du site.
 - `rel="help"` si le fichier est la page d'aide.

- L'attribut `type` précise le type de contenu du fichier externe. Il peut par exemple prendre les valeurs suivantes :

```
type = "text/css" pour une feuille de style CSS
type = "text/javascript" pour un script JavaScript
type = "text/html" ou "text/xml"
type = "images/x-icon" pour créer une icône
```

- L'attribut `href` contient l'adresse relative ou absolue de la ressource externe associée.
- L'attribut `hreflang` – qui, comme `lang`, prend pour valeur un code de langue – précise la langue utilisée dans le document cible.
- L'attribut `media` indique le type de média concerné par le document externe. Nous l'utiliserons en particulier pour lier une feuille de style en précisant le type de média visé par les styles du document CSS. Les valeurs de l'attribut `media` sont `screen` (écran d'ordinateur), `print` (imprimante), `tty` (télétype), `tv` (téléviseur), `projections` (rétro ou vidéoprojecteur), `handheld` (PDA, téléphone), `braille` (terminal braille), `aural` (navigateurs oraux) et `all` (tous les médias).
- L'attribut `size` indique la taille des icônes en particulier quand l'attribut `rel` vaut `icon`. Sa valeur doit comporter deux nombres entiers séparés par les caractères `x` ou `X`.

À titre d'exemple, nous pouvons écrire les codes suivants pour l'élément `<link/>` :

- Pour lier une feuille de style :

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

- Pour lier plusieurs feuilles de styles en précisant le média concerné :

```
<link rel="stylesheet" type="text/css" href="styleWeb1.css" media="screen"/>
<link rel="stylesheet" type="text/css" href="styleWeb2.css" media="print"/>
```

- Pour lier un script JavaScript :

```
<link rel="script" type="text/javascript" href="code.js"/>
```

- Pour créer une icône dans la barre d'adresse :

```
<link rel="icon" type="images/x-icon" href="/fashion.icon"/>
```

- Pour créer un lien de dépendance vers un document :

```
<link rel="next" type="text/html" href="page3.html" />
<link rev="prev" type="text/html" href="page1.html" />
```

- Pour créer un lien vers la page d'accueil :

```
<link rel="start" type="text/html" href="index.html" />
```

Les commentaires

Il est toujours utile de commenter votre code HTML 5, comme tout code informatique d'ailleurs, pour en permettre une meilleure compréhension, en particulier pour le relire un certain temps après l'avoir écrit. Tous les commentaires que vous écrirez seront ignorés par le navigateur, et vous pouvez donc vous exprimer librement. Pour placer un commentaire, vous devez l'ouvrir avec les symboles `<!--`, et le fermer avec les symboles `-->`. N'oubliez pas de fermer vos commentaires, sinon tout ce qui suit sera encore interprété en tant que tel, provoquant une erreur. Par exemple, on aura :

```
<!-- Voici un commentaire HTML  
qui peut comporter plusieurs lignes  
sans problème -->
```

Les méta-informations : l'élément `<meta />`

L'élément `<meta />` est également un élément vide pour lequel l'information est contenue dans ses attributs. Ces informations ne sont donc pas visibles dans la page mais elles sont destinées au serveur web, aux navigateurs et aux moteurs de recherche. Chaque information est identifiée par un nom et un contenu. Le nom de l'information est défini dans les attributs `name` ou `http-equiv` dont les rôles sont similaires, et la valeur associée est contenue dans l'attribut `content` sous la forme suivante :

```
<meta name="nom1" content="valeur1" />  
<meta http-equiv="nom2" content="valeur2" />
```

Si nous utilisons l'attribut `http-equiv`, l'information indiquée dans l'attribut `content` sera présente dans les en-têtes HTTP envoyés par le serveur au navigateur sous la forme de couple nom/valeur.

La plupart des valeurs des attributs `name` et `http-equiv` sont des mots-clés. Nous allons nous arrêter sur la signification et l'utilité des plus courants d'entre eux.

- `name="author"` désigne le nom de l'auteur de la page sans pour autant créer un copyright.

```
<meta name="author" content="Jean Engels" />
```

- `name="keywords"` : cette valeur est très importante pour le créateur de site car son incorporation dans un document sert à l'indexation des pages web par les moteurs de recherche et les annuaires. L'attribut `content` associé à `name` contient la liste des mots-clés que vous allez choisir comme les plus représentatifs du contenu du site. Chaque mot ou expression est séparé des autres par une virgule. Il n'est pas rare d'utiliser plusieurs lignes de mots-clés dans l'attribut `content`. L'utilisation de l'élément `<meta />` à cette fin est donc des plus utile car il va permettre une mise en valeur de votre site qui apparaîtra dans les réponses fournies par les moteurs de recherche si vos mots-clés correspondent à la demande formulée par un internaute. Il est important de bien choisir ses mots-clés pour qu'ils correspondent vraiment au contenu du site et d'en multiplier les variantes dans la liste de l'attribut `content`. On pourra retrouver le même

mot au singulier et au pluriel, au masculin et au féminin. En revanche, il serait contre-productif d'utiliser les mots les plus demandés dans les moteurs de recherche sous prétexte d'attirer du public, alors qu'ils ne correspondent pas au contenu réel de votre site. Exemple de code :

```
<meta name="keywords" content="HTML 4, XHTML, HTML 5, CSS 2, CSS 3, design web,  
↳ création de sites />
```

- `name="description"`. Dans le même ordre d'idée que la valeur précédente, il indique une brève description de l'information contenue dans le site. C'est cette description qui apparaît dans le moteur de recherche et il est donc essentiel qu'elle soit courte et correcte. Inutile de donner une description de 10 lignes alors que Google par exemple n'en affiche que deux. Il est également fortement recommandé d'utiliser cet élément `<meta />` car si vous ne fournissez pas vous-même une description de la page, Google et les autres moteurs de recherche utilisent les premières lignes de votre page qui ne sont pas nécessairement les plus explicites. Exemple de code :

```
<meta name="Description" content="Le site du livre &raquo; HTML 5 et CSS 3  
↳ &raquo; de Jean Engels Editions Eyrolles" />
```

- `http-equiv="refresh"`. Quand il est associé à l'attribut `content` qui a pour valeur un nombre de `N` secondes, son utilisation permet de forcer le navigateur à recharger la page toutes les `N` secondes. On procédera ainsi pour un site aux informations renouvelées très fréquemment, par exemple un site de cotation boursière, ou pour afficher l'heure régulièrement par exemple à l'aide d'un script JavaScript ou PHP. Vous pouvez également utiliser cet élément pour rediriger automatiquement le visiteur vers une autre page du même site ou encore d'un autre site. On appliquera cette technique si le contenu du site a changé de nom de domaine. Pour cela, l'attribut `content` doit contenir le nombre `N` de secondes avant lequel la redirection sera effectuée, suivi d'un point-virgule (;) et de l'URL absolue de la nouvelle page. Par exemple, pour rediriger vers une autre page au bout de dix secondes, écrivez le code suivant :

```
<meta http-equiv="refresh" content="10; http://www.funhtml.com/index/>
```

- `http-equiv="Content-type"`. Cette valeur de l'attribut permet de définir simultanément le type du document et le jeu de caractères employé dans la page. Comme nous l'avons déjà signalé, il faut l'utiliser impérativement si le jeu de caractères n'a pas été précisé, ou si cette information est absente, comme dans le cas des pages PHP. Si cette déclaration n'est pas effectuée, l'utilisation de cet élément `<meta />` est indispensable sous peine de voir le document non validé. L'attribut `content` contient alors le type du document suivi d'un point-virgule puis le code du jeu de caractères. Ce qui donne par exemple le code suivant :

```
<meta http-equiv="Content-type" content="text/html;charset=UTF-8" />
```

Les scripts internes : l'élément <script>

Nous avons vu qu'il était possible de lier la page HTML 5 avec un fichier externe contenant du code JavaScript au moyen de l'élément <link />. Si cette solution correspond bien au concept de séparation des fichiers ayant chacun un rôle différent, il est également possible de réaliser la même opération avec l'élément <script> qui sera alors vide. Le type de langage utilisé est précisé dans l'attribut `type` qui est obligatoire et contient généralement la valeur `text/javascript` ou `application/javascript` pour JavaScript. Dans le cas d'un fichier externe, il faut employer l'attribut `src` qui donne l'URL du fichier externe du script qui possède l'extension `.js` pour JavaScript. Nous aurions par exemple le code suivant :

```
<script type="text/javascript" src="http://www.funhtml/html/fichiercode.js">
</script>
```

Il est toujours possible d'incorporer du code JavaScript dans une page de code au moyen de l'élément <script> mais, à la différence de l'utilisation précédente, le code du script est inclus entre les balises <script> et </script>. Dans ce cas, seul l'attribut `type` est requis. Il est d'usage courant d'inclure tout le code JavaScript dans un commentaire situé à l'intérieur de l'élément <script> pour qu'il ne soit pas interprété par les navigateurs dépourvus d'interpréteur (s'il en existait encore !) mais aussi et surtout si le visiteur a volontairement interdit l'exécution des scripts JavaScript.

Nous pouvons par exemple écrire le code suivant :

```
<script type="text/javascript">
  <!--
    function debut()
    {alert('Bonjour HTML 5');}
  -->
</script>
```

En plus de l'attribut `id` commun à la plupart des éléments, l'élément <script> possède également les attributs suivants dont le rôle est annexe par rapport aux précédents.

- `charset` pour préciser le jeu de caractères utilisés dans l'élément.
- `defer` dont la seule valeur autorisée est `defer`. S'il est utilisé, l'interpréteur JavaScript du navigateur n'interprète pas le code contenu dans l'élément avant l'affichage de la page, ce qui rend l'affichage plus rapide. Cet attribut ne sera utilisé que si le code ne contient pas d'instructions provoquant un affichage direct dans la page du type de celles réalisées au moyen de la fonction `write ()`.

À la différence des autres éléments présents dans l'en-tête du document, l'élément <script> peut être utilisé également dans le corps du document, directement inclus entre les balises <body> et </body>, ou indirectement dans de nombreux autres éléments inclus eux-mêmes dans <body> et dont la liste figure dans le tableau 2-1.

Tableau 2-1

Tous les éléments de la catégorie Phrasing (voir le tableau 2-3) ainsi que <head>.

Le code suivant utilise l'élément <script> et tous ses attributs :

```
<script type="text/javascript" charset="UTF-8" defer="defer">
<!-- Code JavaScript -- >
</script>
```

L'incorporation des styles : l'élément <style>

L'utilisation des styles CSS est étroitement liée à HTML 5 comme elle l'était déjà avec XHTML. Nous avons vu qu'ils pouvaient le plus souvent être écrits dans un fichier externe lié à la page avec l'élément <link />. Cependant, comme pour les scripts, il est possible de les placer directement dans le code HTML en tant que contenu de l'élément <style>. Il n'est pas interdit d'écrire tous les styles dans cet élément mais cette possibilité sera réservée aux cas où les styles sont peu nombreux. Il est aussi envisageable de lier un fichier externe contenant les styles communs à toutes les pages d'un site avec <link /> et d'ajouter des styles particuliers à une page dans l'élément <style> de celle-ci.

En plus des attributs `id`, `lang`, `dir` et `title` que nous avons déjà rencontrés, et peu utiles ici, l'élément <style> possède les attributs suivants.

- L'attribut `type`, dont la présence est obligatoire, précise le type de feuilles de styles utilisées. Pour les documents HTML, il prend toujours la valeur `text/css`.
- L'attribut `media` précise le type de média concerné par la feuille de style. Il est donc possible d'incorporer plusieurs éléments <style> dans l'en-tête de la page, chacun étant destiné à un média différent (voir l'exemple 2-4). Il prend les valeurs suivantes : `screen` (écran d'ordinateur), `print` (imprimante), `tty` (télétype), `tv` (téléviseur), `projection` (rétro ou vidéoprojecteur), `handheld` (PDA, téléphone), `braille` (terminal braille), `aural` (navigateur oral) et `all` (tous les médias).

Le code suivant définit une couleur de fond jaune pour la page et une couleur bleue pour le texte pour le media écran :

```
<style type="text/css" media="screen">
  body{background-color:yellow;color:blue;}
</style>
```

Nous reviendrons en détail sur l'écriture des styles dans la seconde partie de cet ouvrage. Il nous faut simplement retenir ici la localisation des styles.

Le titre de la page : l'élément <title>

Chacun a pu remarquer dans son navigateur qu'avant l'affichage complet d'une page web, un titre apparaît dans la barre de titre située en haut de la fenêtre du navigateur. Ce texte est défini dans l'élément <title> qui est l'un des deux seuls dont la présence est obligatoire

dans l'élément `<head>`. Son contenu est un simple texte qui doit résumer le contenu de la page en une ligne maximum. Il est important de bien réfléchir à ce contenu car c'est aussi lui qui apparaîtra comme titre principal du site dans les moteurs de recherche. Il doit donc être accrocheur et bien correspondre à l'esprit de la page. Nous aurons par exemple le code suivant :

```
<title>Le site de HTML 5 et CSS 3 </title>
```

L'élément `<title>` possède les attributs globaux dont `lang`, `dir` et `id` que nous avons déjà rencontrés.

L'exemple 2-4 crée une page en utilisant la plupart des éléments possibles de l'en-tête `<head>` dont `<title>` (repère ❶), `<base />` (repère ❷), un grand nombre d'éléments `<meta />` (repère ❸) et `<link />` (repère ❹), les éléments `<style>` (repère ❺) et `<script>` (repère ❻). L'élément `<body>` (repère ❼) contient un minimum d'éléments pour obtenir un affichage, à savoir un titre (repère ❸) et une image (repère ❾). Nous reviendrons dans la section suivante sur cet élément important et sur ce qu'il peut contenir.

Exemple 2-4 Les éléments de l'en-tête du document

```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-type" content="text/html;charset=UTF-8" />
    <!-- Element <title> --> ❶
    <title> HTML 5 et CSS 3 </title>
    <!-- Element <base /> --> ❷
    <base href="http://www.funhtml.com/html/" />
    <!-- Elements <meta /> --> ❸
    <meta name="Author" content="Jean ENGELS"/>
    <meta name="Keywords" content="HTML 5, CSS 3, Web" />
    <meta name="Description" content="Le site du livre &raquo; HTML 5 et CSS 3
    &raquo; de Jean Engels Editions Eyrolles" />
    <meta http-equiv="default-style" content="text/css" />
    <meta http-equiv="Refresh" content="1250; URL=http://www.funhtml.com/xhtml">
    <!-- Elements <link /> --> ❹
    <link rel="shortcut icon" type="images/x-icon" href="/html/images/
    &raquo; favicon.ico" />
    <link rel="stylesheet" type="text/css" href="/xhtml/C2/messtyles.css" />
    <link rel="next" title="PHP 5" type="text/html" href="http://www.funhtml.com/
    &raquo; php5" />
    <link rel="previous" title="Document HTML" type="text/html" href="/html/C2/
    &raquo; exemple2-1.xml" />
    <link rel="alternate href="/en/html" hreflang=en type=text/html title="English
    &raquo; HTML">
    <!-- Element <style> --> ❺
    <style type="text/css" media="screen">
      body {background-color:yellow;color:blue;font-size:40px}
    </style>
    <style type="text/css" media="print">
```

```
    body {background-color:white;color:green;}
</style>
<!-- Element <script> --> ⑥
<script type="text/javascript">
    function debut()
    {alert('Bonjour HTML 5');}
</script>
<script type="text/javascript" src="http://www.funhtml/html/fichiercode.js">
</script>
</head>
<body onload="debut()"> ⑦
<!-- Tout le contenu de la page -->
<h1>Le corps de la page</h1> ⑧
<p>
     ⑨
</p>
</body>
</html>
```

La figure 2-1 montre le résultat minimal obtenu.

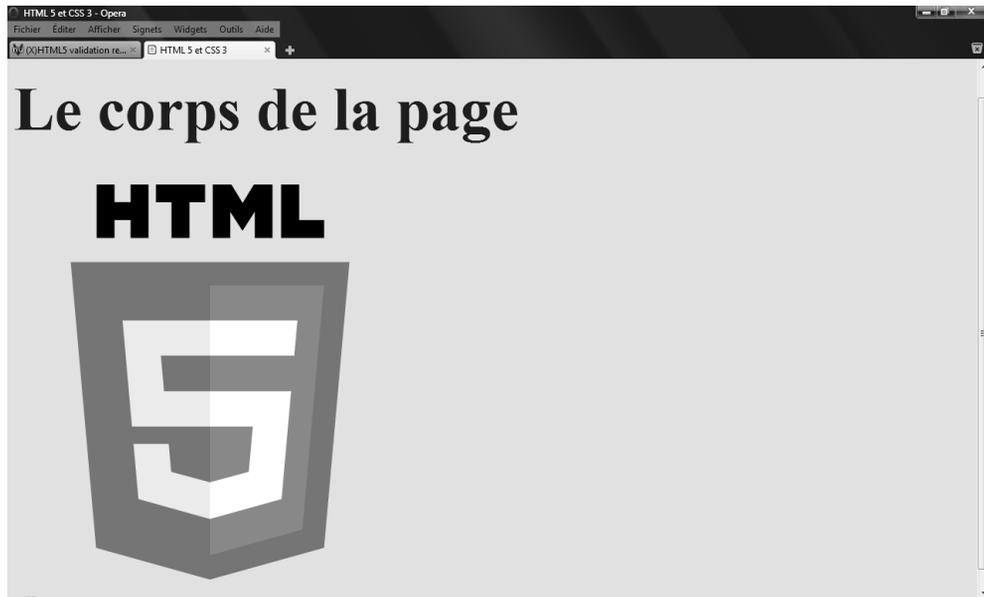


Figure 2-1

L'utilisation des éléments de l'en-tête

Le corps du document : l'élément <body>

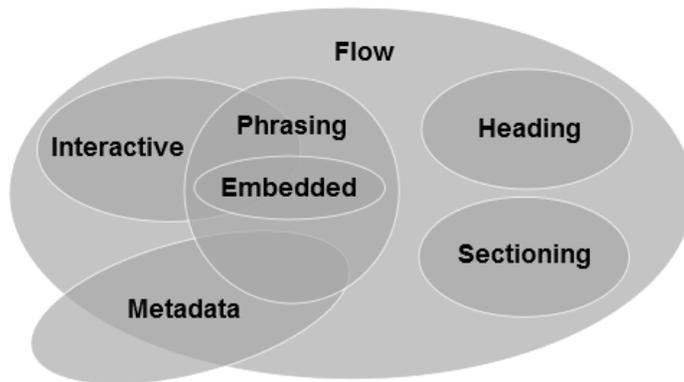
L'essentiel des éléments de l'en-tête <head> que nous venons d'aborder ont un contenu invisible dans le navigateur. L'élément <body> au contraire est le conteneur de l'ensemble des éléments textuels et graphiques d'une page web.

Les catégories d'éléments

La presque totalité des éléments HTML 5 sont désormais regroupés en catégories qui précisent leur rôle. Celles-ci ne sont pas indépendantes les unes des autres et il existe de nombreuses intersections entre ces différents ensembles comme le montre la figure 2-2.

Figure 2-2

Les catégories d'éléments



Les tableaux 2-2 à 2-8 donnent pour chaque catégorie la liste exhaustive des éléments qui la composent. Le tableau 2-9 donne la liste des éléments HTML 5 ne figurant dans aucune catégorie précise.

Tableau 2-2. L'ensemble des éléments de catégorie Flow

a, abbr, address, article, aside, audio, b, bdo, blockquote, br, button, cite, code, del, dfn, div, dl, em, embed, fieldset, figure, footer, form, h1, h2, h3, h4, h5, h6, header, hgroup, hr, i, iframe, img, input, ins, kbd, label, map, mark, math, menu, meter, nav, noscript, object, ol, output, p, pre, progress, q, s, samp, script, section, select, small, span, strong, sub, sup, svg, table, textarea, u, ul, var, video, wbr, texte brut

Tableau 2-3. L'ensemble des éléments de catégorie Phrasing

abbr, audio, b, bdo, br, button, cite, code, dfn, em, embed, i, iframe, img, input, kbd, label, mark, math, meter, noscript, object, output, progress, q, s, samp, script, select, small, span, strong, sub, sup, svg, textarea, u, var, video, wbr, texte brut

Tableau 2-4. L'ensemble des éléments de catégorie Interactive

a, button, embed, iframe, label, select, textarea, audio (si l'attribut controls est présent), img (si l'attribut usemap est présent), input (sauf pour un composant hidden), object (si l'attribut usemap est présent), video (si l'attribut controls est présent)

Tableau 2-5. L'ensemble des éléments de catégorie Heading

h1, h2, h3, h4, h5, h6, hgroup

Tableau 2-6. L'ensemble des éléments de catégorie Sectioning

article, aside, nav, section

Tableau 2-7. L'ensemble des éléments de catégorie Embedded

audio, embed, iframe, img, object, svg, video

Tableau 2-8. L'ensemble des éléments de catégorie Metadata

base, link, meta, noscript, script, style, title

Tableau 2-9. L'ensemble des éléments hors catégorie

caption, col, colgroup, dd, dt, figcaption, head, html, legend, li, optgroup, option, param, source, tbody, tfoot, th, thead, tr

Au fur et à mesure de leur étude, nous préciserons la catégorie des éléments qui peuvent être inclus dans tel élément. Ceci est utile pour savoir, comme en XHTML, quel peut être le contenu (les enfants) d'un élément et dans quels éléments il peut être inclus (ses parents). Il est important pour être conforme aux prescriptions HTML 5 de respecter les différentes inclusions. C'est donc le premier contrôle auquel vous devez procéder. Par exemple, l'élément `<body>` peut contenir tous les éléments de la catégorie Flow mais l'élément `<caption>` ne peut avoir comme parent que l'élément `<table>` et rien d'autre. Notons malgré tout que même si la catégorie Flow contient le texte brut, il est déconseillé d'inclure directement du texte dans l'élément `<body>` ; il vaut mieux le placer dans un élément enfant de `<body>`, ne serait-ce que pour lui appliquer simplement un style particulier et aussi pour qu'il apparaisse clairement dans la hiérarchie des éléments.

Exemple 2-5 Exemple d'inclusion des éléments dans une page HTML 5

```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-type" content="text/html;charset=UTF-8" />
    <title>Les éléments du corps de la page</title>
    <meta name="Author" content="Jean ENGELS" />
    <meta name="Keywords" content="HTML&nbsp;5" />
    <meta name="Description" content="Des inclusions" />
    <link rel="shortcut icon" type="images/x-icon" href="../images/favicon.ico" />
    <style type="text/css" title="style1">
      body{background-color: #EEE;}
    </style>
  </head>
  <body>
    <div>
      <address>Contact : html@funhtml.com</address>
      <blockquote>
        <hgroup>
          <h1>titre de niveau 1</h1>
        </hgroup>
        <dl>
          <dt>Liste de d&eacutefinition</dt>
          <dd>UN</dd>
        </dl>
      </blockquote>
    </div>
    <!-- -->
    <form action="script.php">
      <fieldset>
        <legend>commentaires</legend><br />
        <textarea cols="30" rows="5">texte</textarea>
      </fieldset>
    </form>
    <!-- -->
    <table border="1">
      <tbody>
        <tr>
          <td>Cellule 1 de tableau</td>
        </tr>
        <tr>
          <td>Cellule 2 de tableau</td>
        </tr>
      </tbody>
    </table>
    <!-- The document is valid HTML5 + ARIA + SVG 1.1 + MathML 2.0 -->
  </body>
</html>
```

L'exemple 2-5 présente plusieurs exemples d'inclusions successives d'éléments qui créent une hiérarchie arborescente présentée ci-dessous. L'élément `<body>` a trois enfants directs qui ont chacun à leur tour un ou plusieurs enfants, et ainsi de suite.

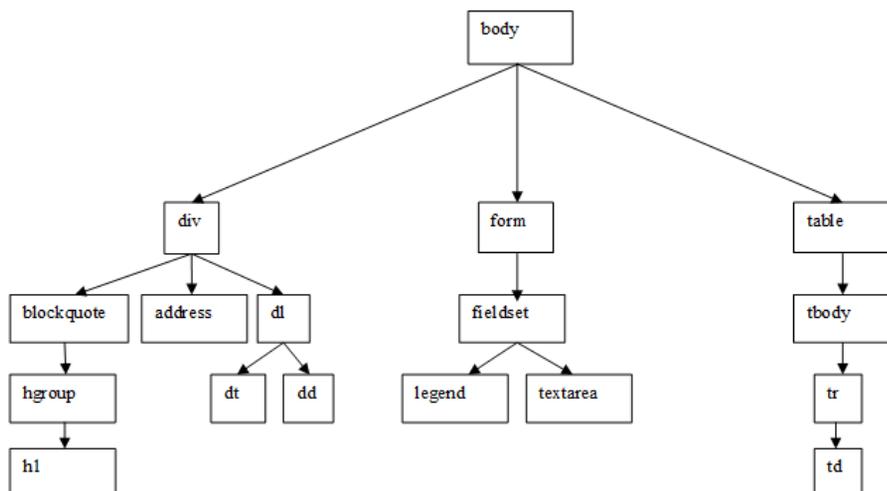


Figure 2-3

Arborescence d'un fichier HTML

Exercices

Exercice 1

La déclaration `DOCTYPE` est-elle obligatoire ? Où doit-elle être placée ? Quelles informations contient-elle ?

Exercice 2

Que faut-il faire pour que le code HTML 5 soit validé ?

Exercice 3

À quel emplacement est défini le jeu de caractères utilisé dans le document ?

Exercice 4

Où est déclaré le nom de l'élément racine du document ? Quel est son rôle et son nom ?

Exercice 5

Quels sont les éléments enfants de l'élément `<html>` ?

Exercice 6

À quoi sert l'élément `<head>` ? Quels sont ses éléments enfants ? Peuvent-ils être employés plusieurs fois dans le même document ?

Exercice 7

Quel élément est obligatoire dans l'élément `<head>` ? À quoi sert-il ?

Exercice 8

Écrire le code nécessaire à la liaison d'une feuille de style avec un document.

Exercice 9

Écrivez le code nécessaire à la liaison d'un document externe contenant des scripts JavaScript avec un document.

Exercice 10

Comment déclarer les mots-clés associés au site ? Quelle est l'utilité de cette déclaration ? Écrivez-en un exemple.

Exercice 11

Quel est le rôle de l'élément `<base />` ? Écrivez-en un exemple.

Exercice 12

Peut-on écrire le code suivant dans une page ?

```
<body>
  Bienvenue dans notre site
  <h1>Le site du HTML 5 et de CSS 3</h1>
</body>
```

Exercice 13

Peut-on inclure les éléments `<tbody>`, `<form>` et `` directement dans l'élément `<body>` ?

Exercice 14

Écrivez un script qui affiche un message d'alerte quand l'utilisateur arrive sur le site.

Exercice 15

Écrivez le code CSS suivant à l'endroit adéquat :

```
body {background-color:white;color:green;font-size :20px}
```

Incluez ensuite un texte dans la page et testez le résultat. Vous devez obtenir un fond rouge, un texte bleu avec des caractères de 20 pixels.

3

Structurer le contenu d'une page

Créer un site web, c'est d'abord créer une charte graphique que l'on va retrouver sur les différentes pages. Dans chaque page, on doit pouvoir distinguer du premier coup d'œil les grandes zones qui la composent comme l'en-tête, la zone de navigation permettant de surfer à l'intérieur du site, la zone principale, dans laquelle est situé le contenu propre de la page, et le pied de page. Mais le contenu d'une page peut être encore plus diversifié, en évitant toutefois la multiplication des zones qui peut distraire l'attention et obliger le visiteur à chercher ce qu'il désire. Nous allons tout d'abord envisager les éléments anciens, ceux propres à HTML 5 qui permettent cette structuration de la page en grandes divisions, puis ceux dédiés particulièrement à l'organisation du texte, et enfin ceux destinés à l'affichage des informations sous forme de listes.

Les grandes divisions de la page

L'élément `<div>`

L'élément `<div>` peut être directement inclus dans le corps du document `<body>`. Il crée une division de la page. Ce type de division permet de grouper dans un seul bloc un ensemble composé soit de textes, soit d'éléments inclus, auxquels on pourra appliquer globalement des styles particuliers. Une division créée avec `<div>` permet d'inclure une très grande variété d'éléments HTML 5, comme du texte brut, les éléments en ligne et la totalité des éléments de la catégorie Flow comme l'autorise `<body>`, dont par exemple les titres, les listes et les formulaires, ce qui est interdit dans un paragraphe. C'est donc un élément très riche qui se prêterait bien à la création de la structure entière d'une page avant HTML 5, et