



Ressources **informatiques**



+ QUIZ

Version en ligne

OFFERTE !
pendant 1 an

Apprenez les langages **HTML5, CSS3** et **JavaScript** pour créer votre premier site web

2^e édition

En téléchargement



code des exemples

Denis MATARAZZO





Les exemples à télécharger sont disponibles à l'adresse suivante :

<http://www.editions-eni.fr>

Saisissez la référence ENI de l'ouvrage **RI2HTCSJA** dans la zone de recherche et validez. Cliquez sur le titre du livre puis sur le bouton de téléchargement.

Avant-propos

Chapitre 1

Le Web

1. Qu'est-ce que le Web ?	7
1.1 Côté serveur : HTTP, FTP, langages, SQL	7
1.2 Côté client : HTML, CSS, JavaScript	10
2. Les langages et leur utilité	10
2.1 HTML	10
2.2 CSS	13
2.3 JavaScript	16
2.4 Exemple général avec les trois langages.	20
3. Les formats d'images	26
3.1 Format Bitmap	26
3.2 Format vectoriel	31
4. Les navigateurs et leurs outils	38

Chapitre 2

Règles générales

1. Préserver la lisibilité : l'indentation, les commentaires	47
1.1 L'indentation.	47
1.2 Les commentaires	50
2. Penser au référencement	51
3. Dossiers et chemins vers les fichiers	53
4. Les éditeurs pour le code	57

2 _____ **HTML5, CSS3 et JavaScript**

Pour créer votre premier site web

5. Des raccourcis bien pratiques	65
5.1 Sauvegarder et tester une page	65
5.2 Sélectionner du texte ou se déplacer plus vite sans la souris	67

Chapitre 3

HTML

1. Création d'une page web	71
2. Code HTML obligatoire	72
3. Le doctype	73
4. La balise <head>	73
4.1 Lien avec une feuille de style	75
4.2 Lien avec un fichier JavaScript	76
5. La balise <body>	77
5.1 Méthode et balises pour structurer une page	79
5.2 Le texte dans la page HTML	87
5.3 Les caractères spéciaux	89

Chapitre 4

CSS3

1. Les trois styles de base possibles	91
1.1 Le style de balise	91
1.2 Le style de classe	95
1.3 Le style d'ID	96
1.4 Combinaison des trois méthodes	98
2. Les polices de caractères et le Web	101
3. Les sélecteurs	111
4. Les pseudoclasses	121
4.1 Pour les liens	121
4.2 Pour le texte	123
4.3 Pour les sélecteurs	124

5. Les couleurs en hexadécimal, en RGBA ou en HSLA	125
6. Les images et les bordures	130
7. Les boutons issus d'images ou de polices	131
8. Les fonds et fonds multiples	136
9. Le positionnement	137
10. Le débordement	142
11. Utilisation de padding et margin	142
12. Les variables	144
13. Des propriétés décoratives (ombre, dégradé, arrondi...)	144
14. Des colonnes dans le texte	150
15. Les transformations 3D	152
16. Les transitions et animations	156
17. Le responsive design et les media queries	158
18. Le menu Burger	161

Chapitre 5

JavaScript

1. Introduction	167
2. La liste des tâches	167
3. Variables et affectation	170
4. Les types de variables	171
4.1 Les valeurs numériques	171
4.2 Le texte et la concaténation	171
4.3 Les tableaux	173
4.4 Les booléens	174
4.5 Les objets	175
5. Les opérateurs	178
6. Les conditions	183

4 _____ HTML5, CSS3 et JavaScript

Pour créer votre premier site web

6.1 if, else et les accolades	183
6.2 switch case	189
7. Les itérations.....	191
7.1 La boucle for	192
7.2 while	195
7.3 do ... while.....	197
7.4 break et continue	197
7.5 Foreach	198
8. Déboguer un programme.....	199
9. Les fonctions.....	203
9.1 Déclaration	203
9.2 Appel.....	204
9.3 Les variables locales et globales	204
9.4 Le retour d'une fonction	206
10. Les cookies.....	207
11. Le drag and drop.....	210
12. Afficher le site HTML en plein écran	213
13. Interactions entre JavaScript, HTML et CSS	215
14. Les bases de données locales.....	217
14.1 Crédation d'une base de données.....	218
14.2 Crédation d'une table.....	218
14.3 Insertion d'enregistrements	219
14.4 Lecture d'informations	219
15. Générer des PNG en JavaScript	220
16. Ajax	225

Chapitre 6**Mise en page HTML et CSS**

1.	Les blocs et leur position à l'écran.....	231
1.1	Les tableaux.....	231
1.2	Les div et les nouvelles balises HTML5.....	234
2.	Les listes	239

Chapitre 7**Les méthodes de dessin**

1.	La balise Canvas	243
2.	La balise SVG	248
3.	Avantages et inconvénients des deux technologies	257

Chapitre 8**Le multimédia**

1.	La balise <video>.....	259
2.	Les codecs vidéo	263
3.	La balise <audio>	264
4.	Les codecs audio	266

Chapitre 9**Les formulaires**

1.	Introduction	269
2.	Fonctionnement d'un formulaire client/serveur	270
3.	Les différentes balises du formulaire	271
4.	Les expressions régulières.....	280
5.	La validation du formulaire	282

6 _____ HTML5, CSS3 et JavaScript

Pour créer votre premier site web

6. Ajout d'un script CGI sur le serveur	285
---	-----

Chapitre 10

Les liens et menus en HTML5

1. Introduction	289
2. Création de liens	290
2.1 Ouverture de page HTML	290
2.2 Ouverture d'une image	291
2.3 Navigation dans la page	292
2.4 Proposer le téléchargement d'un fichier	293
2.5 Envoyer un e-mail	293
2.6 Déclenchement d'un script JavaScript	295
3. Création d'un menu (liste + liens + CSS)	295
4. Agir sur la page grâce au menu	297
5. Ajout de « data » dans les liens	300
6. Liste complexe organisée par JavaScript et le CSS	304
Conclusion	319
Index	321

Chapitre 6

Mise en page HTML et CSS

1. Les blocs et leur position à l'écran

La mise en page par le code HTML est simplement le fait de positionner des blocs à l'endroit souhaité. Le sujet a été abordé au chapitre HTML - section Méthode et balises pour structurer une page et va être développé ici en combinant HTML et CSS.

1.1 Les tableaux

Ils ont longtemps été la méthode principale pour disposer le contenu d'une page. La méthode consistait à créer un tableau, et dans une cellule du tableau insérer un autre tableau, qui lui-même pouvait contenir un ou des tableaux.

C'était efficace mais il fallait s'y retrouver dans le code car tous ces tableaux imbriqués faisaient que parfois il était difficile de savoir sur quel tableau on était en train de travailler.

Repartons de l'exemple avec les fruits et légumes, vu au chapitre HTML - section Méthode et balises pour structurer une page, et voyons comment obtenir cette apparence pour notre tableau. Cet exemple peut être observé via le fichier **6_1_1_Tableaux.html**.

Livraison Fruits et légumes		
VENTES	Fruits et légumes	Quantité livrée
2014	Pommes	12 Tonnes
	Poires	3 Tonnes
	Raisin	10 Tonnes
	Bananes	7 Tonnes
	Oranges	12 Tonnes

```
<table id="fruitsLegumes">
    <tr>
        <td rowspan="7" class="titreGauche">V E N T E S &nbsp; 2
0 1 4</td>
        <th colspan="2" class="titre">Livraison <br />Fruits et
légumes</th>
    </tr>
    ...

```

La première colonne « VENTES 2014 » utilise la propriété HTML rowspan. L'espace occupé par ce `<td>` sera équivalent à 7 lignes, si bien que la première colonne prendra toute la hauteur du tableau. Le texte est écrit avec des espaces entre chaque lettre, pour faire qu'à l'affichage les lettres soient les unes en dessous des autres.

Le titre « Livraison Fruits et légumes » utilise la largeur de deux colonnes grâce à la propriété colspan, qui regroupe deux colonnes. Le `
` fait que le texte s'écrit sur deux lignes.

La balise `<table>` a par défaut des marges internes et un espace entre les bordures. Si aucune valeur ne vient corriger cela, il y aura de l'espace entre les cellules, et pour les lignes de détail, ce serait assez disgracieux d'avoir un espace en arrière-plan entre « Pomme » et « 12 Tonnes ». Pour cela, il faut mettre l'espace entre les cellules à 0. Cela pourrait être écrit directement dans la balise `<table>` en écrivant `cellspacing="0"`. La méthode utilisée ici passe par le CSS.

```
#fruitsLegumes, th {
    letter-spacing: 1px;
    border: 1px solid #000;
    margin: 0;
    padding: 0;
```

```
    border-spacing: 0px;
}
```

La propriété CSS `border-spacing` permet de modifier cette valeur, et elle est forcée à 0.

Les deux premières lignes du tableau, des `<th>`, avec le fond blanc, ont une bordure intérieure. La balise `<th>` (*table head*) écrit automatiquement en gras et au centre. La balise `<th>` est stylisée avec l'ID `#fruitsLegumes` et devrait avoir une bordure tout autour d'elle de 1 pixel, noire, avec un trait continu.

Pour annuler cela, un style est créé uniquement pour la balise `<th>` qui remet la bordure à 0 pour l'ensemble et en fixe une pour le bas :

```
th {
    padding: 4px;
    background-color :#fff;
    color: #000;
    border:0px solid #000;
    border-bottom: 1px solid #000;
}
```

De la même façon, le titre de gauche a une bordure blanche sur le côté droit :

```
.titreGauche {
    text-align: center;
    width: 15px;
    background-color: #1362bb;
    color:#FFF;
    font-weight: bold;
    border-right: 1px solid #FFF;
}
```

Les tableaux restent utilisables pour l'affichage de données comme dans cet exemple. Leur principal défaut est de toujours ressembler à un tableau. Pour le responsive design, il est pratique de pouvoir, uniquement avec le CSS, changer complètement l'apparence d'une page. Or, avec le tableau ce sera soit impossible soit très compliqué. C'est pour cela que les balises `<div>` ont tant de succès.

1.2 Les div et les nouvelles balises HTML5

Les div ont été abordées dans le chapitre CSS3 et des notions de débordement et de positionnement ont été vues. Avec ce qui a été abordé entre-temps, il va être possible de faire une page HTML avec une zone header (ou menu) en haut de la page qui ne bougera pas quand l'ascenseur descendra dans la page, de la même façon que le bas de page, qui pourra rester toujours en bas quelle que soit la taille de la page. Voir l'exemple : [6_1_2_lesDivs.html](#).

Pour le code HTML, il y a quatre grandes parties. Elles sont incluses dans les balises <header>, <nav>, <div> et <footer>.

```
<header id='topPage'>Un TOP qui bouge</header>
<nav id='menu'>Un menu qui bouge jusqu'en haut de la page</nav>

<div id='mainPage'></div>

<footer id="basPage">Un Footer fixe.</footer>
```

Bien que le nom de ces quatre balises soit complètement différent, elles vont fonctionner de la même façon, s'afficher de la même façon et avoir les mêmes propriétés possibles au niveau CSS. En fait, leur nom va permettre d'aider à détecter leur contenu. La balise <header> se trouve en haut de quelque chose, ici en haut de la page. La balise <nav> contient les éléments prévus pour la navigation, comme un menu. Le div a déjà été vu et ne dit rien sur son contenu, et pour finir la balise <footer> est la zone la plus basse d'un bloc.

Le code CSS permet de positionner les éléments précisément. L'utilisation de la position : fixed ainsi que de z-index permet de faire en sorte que si le contenu de la page scrollle, le bas de page reste toujours à sa place. Et un peu de code JavaScript va permettre de faire en sorte que le menu monte jusqu'en haut de la page et s'arrête à ce moment-là.

```
html, body {
    padding: 0;
    margin: 0;
}
```

Mise à zéro des différentes marges pour la partie HTML et pour le BODY, de façon à avoir 100% de la page disponible.

```
#topPage {  
    position: absolute;  
    z-index: 100;  
    top: 0;  
    background-color: #365D88;  
    width: 100%;  
    height: 100px;  
}  
  
#menu {  
    position: absolute;  
    z-index: 100;  
    top: 100px;  
    background-color: #264D78;  
    width: 100%;  
    height: 40px;  
}  
  
#basPage {  
    position: fixed;  
    z-index: 100;  
    bottom: 0;  
    background-color: #365D88;  
    width: 100%;  
    height: 50px;  
}
```

Les `#topPage`, `#menu` et `#basPage` ont leur valeur de `z-index` fixée à 100. Ils peuvent avoir la même valeur puisqu'ils ne se chevauchent jamais.

Le `#topPage` commence en étant calé en haut (`top : 0;`) et le menu le suit en ayant le top à 100 pixels, ce qui correspond à la hauteur du `#topPage`.

Le `#basPage` a une position fixe en bas, puisque `bottom` est à 0 et que `position` est sur `fixed`.

```
#mainPage {  
    position: absolute;  
    z-index: 90;  
    width: 100%;  
    height: 2000px;  
    background: rgba(248, 80, 50, 1);  
    background: -moz-linear-gradient(top, rgba(248, 80, 50, 1) 0%,
```

```
rgba(230,198,39,1) 100%);  
background: -webkit-gradient(left top, left bottom, color-  
stop(0%, rgba(248,80,50,1)), color-stop(100%, rgba(230,198,39,1)));  
background: -webkit-linear-gradient(top, rgba(248,80,50,1) 0%,  
rgba(230,198,39,1) 100%);  
background: -o-linear-gradient(top, rgba(248,80,50,1) 0%,  
rgba(230,198,39,1) 100%);  
background: -ms-linear-gradient(top, rgba(248,80,50,1) 0%,  
rgba(230,198,39,1) 100%);  
/*background: linear-gradient(to bottom, rgba(248,80,50,1)  
0%, rgba(230,198,39,1) 100%);*/  
filter:progid:DXImageTransform.Microsoft.gradient(  
startColorstr='#f85032', endColorstr='#e6c627', GradientType=0 );  
}
```

Le #mainPage correspond à la partie principale du site. Il passera derrière le menu ou sous le menu et sous les éléments fixes en règle générale.

Si ce code est exécuté, le #menu va bouger vers le haut sans s'arrêter lorsqu'il sera au maximum de la page. Il continuera de monter et disparaîtra de l'écran.

Il est possible d'interroger le navigateur pour savoir si l'ascenseur est en cours d'utilisation, autrement dit, si la page est en train de scroller.

Pour que le #menu monte et s'arrête en haut, il suffit de détecter le déplacement du #menu, et lorsqu'il arrive à la limite de la page, de changer sa propriété CSS position et de la rendre fixe de façon à ce qu'il ne bouge plus. Il faudra naturellement la remettre relative lorsque l'ascenseur redescendra pour que le #menu retrouve sa place à l'écran.

```
function init() {  
    var element = document.getElementById('menu');  
    posDepartMenu =  
    parseInt(window.getComputedStyle(element).getPropertyValue('top'))  
;  
  
    leDoc = document.documentElement;  
    leBody = document.body;  
}
```

La fonction `init()` ; déclenchée par le `onLoad` de `<body>` va récupérer dans la variable `posDepartMenu` la position de départ du menu.

Même si la position de départ du menu est connue dans le CSS et est de 100 px, le CSS pourrait changer. La valeur pourrait ne plus être 100 et si la valeur 100 était inscrite en dur dans le code JavaScript, le CSS ne s'afficherait plus correctement. Cela permet de dissocier le CSS du JavaScript.

La valeur récupérée par `getPropertyValue('top')` sera en fait « 100px ». Les deux caractères px vont être embêtants pour effectuer le calcul, JavaScript considérant cela comme du texte. L'instruction `parseInt()` permet de convertir « 100px » en une valeur entière. Nous avons donc la valeur 100 stockée dans `posDepartMenu`.

Les deux variables `leDoc` et `leBody` sont en fait juste là pour ne pas avoir à réécrire la ligne de code complète : il suffira d'écrire `leDoc` plutôt que d'écrire `document.documentElement`. De même que pour `leStyle` ci-dessous :

```
window.onscroll = function(event) {
    var leTop = (leDoc && leDoc.scrollTop || leBody &&
    leBody.scrollTop || 0);
    document.getElementById("basPage").innerHTML = leTop + " / "
+ posDepartMenu;

    var leStyle = document.getElementById('menu').style;
    if (leTop > posDepartMenu) {
        leStyle.top = '0px';
        leStyle.position = 'fixed';
    } else {
        leStyle.top = posDepartMenu + 'px';
        leStyle.position= 'relative';
    }
};

window.onscroll
```

`window.onscroll` est l'événement qui est déclenché lorsque l'utilisateur scrollle la page. La fonction ci-dessus sera donc appelée à chaque fois que l'utilisateur fera bouger l'ascenseur.