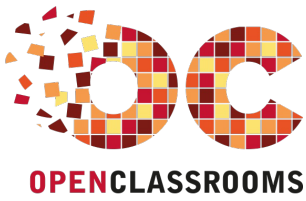
# Des techniques modernes pour l'agencement en colonnes

**Par Mattéo DELABRE (m@tteo78)**




www.openclassrooms.com

*Licence Creative Commons 4 2.0*
*Dernière mise à jour le 10/02/2012*

# Sommaire

# Des techniques modernes pour l'agencement en colonnes

Par  Mattéo DELABRE (m@tteo78)

Mise à jour : 10/02/2012
Difficulté : Intermédiaire Durée d'étude : 1 heure, 20 minutes

BY-NC-ND

Il a longtemps été impossible d'obtenir des colonnes de même taille sans effectuer de bidouillages : avec des images, du JavaScript ou d'autres techniques qui ne touchent parfois pas toutes les personnes ou ne sont pas facilement adaptables (certaines techniques vous demanderont de recommencer toute la procédure si vous désirez écarter un peu plus les colonnes). Ce tutoriel présente deux techniques rendues possibles par l'utilisation de CSS3, l'une d'entre elles vous démontrera qu'il est possible de simuler le comportement des tableaux pour parvenir à nos fins, tandis que l'autre utilisera une fonctionnalité prévue uniquement à cet effet.

Avant l'arrivée et la popularisation de CSS, les tableaux remplissaient malgré eux cette fonction ; il a, par ailleurs, été démontré sous beaucoup d'angles que cette option était à proscrire. Notamment au niveau de la sémantique : un tableau est fait pour représenter *des données*. À la lecture de ces lignes, vous devez sûrement vous demander pourquoi je vous explique de nouveau cela (car je suis sûr que vous le saviez déjà auparavant) ; nous allons exploiter ces fonctionnalités pour produire l'effet escompté.

Un autre moyen d'atteindre notre but, de façon plus claire mais compatible avec moins de navigateurs, est d'utiliser les nouvelles propriétés définies avec la norme CSS niveau 3 qui permettent de créer des colonnes de texte dans n'importe quel élément. Nous aborderons ce sujet, accompagné d'un sujet de travaux pratiques, à la fin de ce tutoriel.

La première de ces techniques est fondée sur CSS 2.1 et l'autre sur CSS niveau 3.
En résumé, et généralement, les techniques présentées **ne fonctionneront pas** sur Internet Explorer (hormis la première qui est supportée depuis la version 8 du navigateur précédemment cité). De plus, les codes d'exemple de ce cours sont rédigés en HTML5, encore bogué sur I.E. ; si vous utilisez une autre version de HTML (HTML4, xHTML…), il vous faudra adapter les codes en conséquence.


Sommaire du tutoriel :



## Copier à la perfection un tableau

*Non, il ne s'agit pas là de peinture mais d'affichage d'éléments en tant que tableaux.*

Revenons sur ce que je vous ai dit tout à l'heure à propos des tableaux. La principale lacune des tableaux est le fait qu'ils ne sont pas conçus pour s'occuper du design d'un site, mais pour stocker des données. Un design « en tableaux » compliquera la lecture du code source. Le code suivant met en application cette technique démodée, il n'est là qu'à titre d'information :

**Code : HTML**

```
<table>
    <tr>
        <td>Colonne 1</td>
        <td>Colonne 2, avec du contenu supplémentaire<br />…<br
/>…</td>
    </tr>
</table>
```

L'utilisation de ce code est désormais, et pour de bonnes raisons, fortement déconseillée.

Ce qui nous intéresse, nous, c'est ce qui se passe avec le tableau, c'est-à-dire sa capacité à avoir des cellules de même taille. Avec la norme CSS apparue en juin 2011 (CSS 2.1), il est possible de simuler le comportement des cellules de tableaux, sans pour autant créer des tableaux. Autrement dit, les propriétés réservées aux tableaux peuvent désormais s'appliquer à n'importe quel élément.

En résumé, pour cette méthode, nous créerons un code tout à fait banal, qui ne contiendra pas de tableau. Du côté CSS, on active la représentation sous forme de tableau. La présentation est séparée du contenu, la sémantique est respectée : la technique est parfaitement envisageable.

L'affichage en tableaux est supporté par près de 90 % des utilisateurs (selon *Caniuse*).

Mettons, dès à présent, tout cela en pratique ! 

## Un cadre parfait pour nos colonnes-tableaux

Dans cette première partie pratique, nous allons créer pas à pas, avec des explications détaillées, une base de contenu en HTML, en laissant de côté le CSS. Nos deux colonnes contiennent du texte, elles peuvent donc être représentées par des éléments de paragraphe comme suit :

**Code : HTML**

```
<p class="colonne">Contenu de la colonne</p>
<p class="contenu">Contenu</p>
```

Ce code, cependant, n'est pas suffisant au niveau de la présentation CSS. Réfléchissons à un élément de tableau : nos deux paragraphes seront des cellules. En l'occurrence, ces cellules sont « abandonnées » en plein milieu du code, au contraire d'un tableau où elles sont contenues dans l'élément de même nom. En conclusion, nous devons créer un élément de division qui contiendra nos colonnes et qui sera affiché en tant qu'élément principal de tableau.

Voici donc la structure HTML que nous avons choisie :

**Code : HTML**

```
<div class="colonnes">
    <p class="colonne">Contenu de la colonne</p>
    <p class="contenu">Contenu principal</p>
</div>
```

Prévisualiser

Étant donné que nous n'allons pas modifier cette structure, nous pouvons conclure que quelqu'un ayant désactivé CSS pourra parfaitement lire nos colonnes, sans être dérangé par des éléments superflus en rapport avec le design.

Étudions à présent la partie la plus intéressante et la moins banale : le CSS.

# Dresser la base de nos colonnes, grâce à CSS

Nous allons pour cette seconde partie pratique créer ensemble un style CSS minimal et suffisant pour que s'affichent nos colonnes, tout cela en explications.

### *La propriété `display`*

La propriété **`display`** a le plus souvent été connue sous trois valeurs possibles :

**Code : CSS - Les valeurs possibles pour display**

```
display: block; /* Affiche en tant qu'élément block */
display: inline; /* Affiche en tant qu'élément en-ligne */
display: none; /* N'affiche pas l'élément */
```

La révision première de CSS2 introduit bien des nouvelles fonctionnalités, mais en complète aussi d'autres en ajoutant des nouvelles possibilités. La propriété **`display`** n'échappe pas à la règle et introduit beaucoup de valeurs intéressantes dont des valeurs pour l'affichage en tant que tableaux. Les deux valeurs suivantes vont nous intéresser particulièrement :

**Code : CSS - De nouvelles valeurs possibles pour display en CSS3**

```
display: table; /* Affiche en tant qu'élément principal de tableau
*/
display: table-cell; /* Affiche en tant que cellule de tableau */
```

### *Affichage des éléments de colonnes*

Lors de la mise en place de notre structure HTML, nous avons présenté cette division comme nécessaire. En réalité, elle ne l'est pas vraiment puisqu'il ne s'agit pas de vraies cellules, mais, en créant un élément de tableau, nous pourrons profiter des propriétés qui s'y appliquent. Voici donc notre début de codage :

**Code : CSS**

```
div.colonnes {
    display: table;
}

p.colonne, p.contenu {
    display: table-cell;

    padding: 5px;
    border: 1px solid black;
}
```

Prévisualisation

Nous pouvons remarquer, grâce aux bordures que nous avons définies, que les deux colonnes possèdent bien la même hauteur. Nous n'avons pas défini de largeurs spécifiques, elles s'adaptent donc en conséquence de leur contenu. Cela dit, vous pouvez les spécifier vous-même.

# Le dernier coup de peinture : finaliser en un design

Vous pouvez remarquer que le design est très minimaliste et vous vous dites sûrement qu'il est très difficile de styler des cellules de tableau…

Eh bien, vous vous trompez fortement, dans ce chapitre seront présentées les propriétés utiles pour rendre plus jolies nos colonnes.

### Les avantages de `display: table`

Le plus important de cette balise est le fait de pouvoir définir l'espacement entre nos cellules. En effet, étant donné que nous héritons, en plus du comportement des tableaux, de ses propriétés, nous pouvons utiliser `border-spacing`.

**Code : CSS**

```
div.colonnes {
    display: table;

    border-collapse: separate; /* Pour s'assurer que les cellules ne
sont pas collées */
    border-spacing: 5px; /* 5 pixels d'écartement entre les colonnes
*/
}
```

La propriété `empty-cells: [show | hide]` pourrait aussi vous être utile, bien que je ne lui aie pas encore trouvé d'utilité. 

### Les avantages de `display: table-cell`

Ici, le plus gros avantage est la propriété `vertical-align` qui permet de centrer un texte verticalement dans votre colonne. Cela peut se révéler utile pour une colonne qui ne concerne pas la navigation (pour que l'utilisateur ne soit pas obligé de descendre pour atteindre les liens) et qui s'adapte à un contenu afin de « combler le vide » occasionné par ce redimensionnement.

**Code : CSS**

```
p.colonne {
    display: table-cell;

    text-align: center; /* Centrage horizontal */
    vertical-align: middle; /* Centrage vertical */
}
```

### Un design possible pour nos colonnes

En résumé de tous les agréments présentés dans cette partie, j'ai créé à titre d'exemple un design possible en colonnes. Voici le lien pour le visualiser directement, tandis que vous pouvez regarder la source CSS ici. Ce n'est qu'une possibilité de design, vous pouvez en créer bien d'autres, sous beaucoup de formes différentes.

### D'autres valeurs pour la simulation de tableaux

La propriété `display` est capable d'afficher, en plus des cellules, tous les éléments concernant les tableaux. Voici une liste à laquelle vous pouvez vous référer :

| Valeur | Description |
|---|---|
| `table` | Élément principal de tableau |
| `table-inline` | Tableau en-ligne |

| | |
|---|---|
| `table-row` | Ligne de tableau |
| `table-column` | Colonne de tableau |
| `table-row-group` | Groupement de lignes de tableau |
| `table-column-group` | Groupement de colonnes de tableau |
| `table-cell` | Cellule de tableau |
| `table-header-group` | En-tête de tableau |
| `table-footer-group` | Pied de tableau |
| `table-caption` | Titre de tableau |

*Source des valeurs : W3C - Liste des valeurs pour la propriété `display` (anglais)*

## Sans tableaux ? Des propriétés CSS3 prévues à cet effet

Il y a peu de temps, en lisant votre journal aux pages 2 et 3, vous avez sûrement remarqué du texte disposé en colonnes. Depuis lors, vous rêvez peut-être d'obtenir le même résultat sur votre site. Le W3C a pensé à vous en introduisant dans la nouvelle norme CSS niveau 3 des propriétés simples pour créer des colonnes de texte. Nous allons donc nous intéresser au « working draft » de ces propriétés, voici le texte en question.

L'idée est en travail depuis avril 2011, c'est donc relativement récent, ce qui veut dire que peu de navigateurs l'implémentent. Néanmoins, ici n'est pas le sujet, poursuivons pour trouver ce qui nous intéresse. Voilà, nous y sommes. Intéressons-nous de plus près à ces propriétés, si vous le voulez bien.

### *Taille et nombre des colonnes*

Sur les navigateurs dans leurs versions récentes (Firefox 3.5, Opera 11.1 et, à l'avenir, I.E. 10), chaque élément HTML, chaque texte est disposé en colonnes dans son élément parent ! Simplement, ils sont disposés dans une seule colonne, ce qui peut se régler facilement grâce à plusieurs des propriétés imaginées par le W3C, implémentées sur les navigateurs récents.

Durant la présentation des propriétés d'affichage en colonnes, je les écrirai sans les préfixes propriétaires (`-moz-`, `-webkit-` et autres) dans un souci de lisibilité. Cependant, pour qu'elles fonctionnent sur Chrome (et dérivés) et Firefox, il faut les préfixer avec ceux qui ont été présentés plus haut.

- La propriété `column-count` permet de spécifier le nombre de colonnes de texte présentes dans l'élément. Par défaut, la valeur est `auto`, ce qui signifie que si vous spécifiez une largeur pour les colonnes, le nombre de colonnes sera calculé par le navigateur. Dans le cas contraire, une seule colonne sera utilisée. La valeur à spécifier pour ces propriétés est simplement un nombre, raisonnable (ne demandez pas cent colonnes sur un élément de 100 pixels de largeur : le tout serait illisible) si possible.

- La propriété `column-width` permet de spécifier une largeur pour chacune des colonnes qui seront créées. Cette propriété est actuellement prise en compte par les navigateurs uniquement si la propriété `column-count` est définie à `auto`, dans le cas contraire sa valeur est ignorée.

  La valeur que peut prendre cette propriété est une longueur comme vous avez l'habitude de les spécifier. Cependant, vous ne pouvez pas utiliser les pourcentages. Mais les autres unités sont utilisables, bien que je vous déconseille l'utilisation des centimètres qui sont très mal interprétés.

- La « méga-propriété » de colonnes (`columns`) rassemble les deux propriétés citées ci-dessus. Personnellement, je ne lui ai pas trouvé d'utilité puisque la valeur de `column-width` est ignorée quand l'autre a une valeur. De plus, Firefox ne reconnaît pas cette propriété, ne l'utilisez donc pas avant un peu de temps.

Exemple :

**Code : CSS - Exemple d'utilisation des propriétés citées**

```
body {
  column-count: 2; /* Disposition de la page en deux colonnes */
}

body {
  column-width: 10em; /* Disposition de la page en colonnes de
taille dix fois supérieure à la normale */
}
```

### *Améliorer la visibilité des colonnes*

Vous voudriez sûrement ajouter une touche personnelle à l'apparence des colonnes ainsi produites. Des propriétés ont encore une fois été imaginées à cet effet. Nous pourrions, par exemple, régler l'espacement entre nos colonnes qui est réglé à une valeur qui diffère actuellement sur certains navigateurs ; ou bien, afficher un trait entre chaque colonne… Petite présentation avec ces propriétés simples mais utiles.

- La propriété **`column-gap`** permet de définir l'espacement entre les colonnes crées par le navigateur. La valeur par défaut varie entre *0px* et *1em*, selon les navigateurs, bien que la seconde soit, elle, recommandée par la spécification. Toutes les unités sont acceptées en tant que valeur, sauf, encore une fois, les pourcentages.

- La propriété **`column-rule-style`** définit le style de la bordure séparant les colonnes, et elle est définie par défaut à **`none`**. Les valeurs possibles sont les mêmes que celles proposées pour les bordures d'éléments divers :
  - **`none`** (pas de bordure) ;
  - **`dotted`** (bordure « en points ») ;
  - **`dashed`** (série de petits traits) ;
  - **`solid`** (une simple bordure continue) ;
  - **`double`** (une double bordure continue) ;
  - **`groove`** (effet de profondeur, vers l'intérieur) ;
  - **`ridge`** (effet de profondeur, vers l'extérieur).

- La propriété **`column-rule-width`** permet de définir la largeur de la bordure séparant les colonnes définies. Par défaut, **`none`**. Vous pouvez définir cette propriété avec n'importe quelle unité, sauf les pourcentages.

- La propriété **`column-rule-color`**, quant à elle, définit la couleur de cette dernière bordure.

- Toutes ces propriétés peuvent être définies grâce à la « méga-propriété » **`column-rule`**.
  Exemple : **`column-rule: 3px inset #330000`**.

Exemple :

**Code : CSS - Code d'exemple résumé des propriétés**

```
body {
  column-width: 150px; /* Colonnes de 150 pixels de largeur */
  column-gap: 30px; /* Espacement */
  column-rule: 1px solid black; /* Un simple trait noir pour bordure
de séparation */
}
```

Il existe d'autres propriétés pour mettre en forme les cellules. Cependant, la plupart des autres ne sont pas implémentées dans les

navigateurs actuels, puisque leur fonction n'est pas principale.
On trouve d'autres propriétés, rarement implémentées sur les navigateurs actuels, qui introduisent des fonctionnalités supplémentaires. Par exemple, pour réguler la distribution du texte parmi les colonnes ou pour répartir un élément sur plusieurs colonnes en largeur. En outre, Mozilla a initié l'utilisation de la propriété **height** pour les conteneurs de colonnes, qui permet de spécifier la hauteur maximale que les colonnes peuvent prendre (cependant, cette propriété reste hors de la spécification actuelle).

Néanmoins, cela peut poser quelques problèmes, comme nous allons le voir.

### *Propriétés peu souvent interprétées*

La propriété **column-span** n'est reconnue ni par Mozilla Firefox ni par Chrome, qui sont deux navigateurs possédant une part importante du marché. Cette propriété permet de spécifier si les éléments HTML de type *block* contenus dans les colonnes prennent, en largeur, la place de toutes les colonnes ou se limitent à celle où ils se trouvent. La valeur par défaut étant **none** (les éléments prennent la place de la colonne où ils se trouvent uniquement), nous pouvons sans problème insérer des éléments *block* dans nos colonnes.

Cependant, impossible d'utiliser cette propriété actuellement car aucune des autres valeurs n'est acceptée. Nous devrons nous contenter de ce genre de choses jusqu'à l'interprétation complète :

*Un élément `h1` impossible à placer sur 2 ou 3 colonnes (cliquez pour agrandir)*
Prévisualisation

Restez à l'écoute des news pour être au courant de l'évolution de la situation.

Vous avez désormais tous les outils nécessaires pour créer un design en colonnes. Sachant que cette méthode ne concerne que, globalement, le texte et qu'elle est encore à l'état de test, vous pouvez tout de même l'utiliser dans des buts simples. Désormais, pourquoi pas un exercice de travaux pratiques, afin d'assimiler toute cette théorie ?

## [T.P] Reconstitution d'une première page de journal

Le but de cet exercice est de reconstituer, à l'aide de HTML5 et CSS3 uniquement, une première page de journal. Il est évident que vous devrez disposer les morceaux de texte en colonnes, tout cela dans des alignements justifiés.

Cet exercice ne vous fera pas uniquement travailler les colonnes CSS3, ce sera aussi l'occasion pour vous de revoir certaines propriétés peut-être oubliées. Il ne vous est pas interdit de remonter plus haut dans ce tutoriel ou de rechercher des informations supplémentaires pour vous rafraîchir la mémoire, c'est même le but principal de l'exercice.

Au niveau de la mise en page du journal, je vous laisse libre choix. Personnellement, j'ai choisi de placer le titre de journal en haut, des gros titres pour décrire le contenu du journal, avec des annotations de page avec le texte « à la une » juste au-dessous. J'ai aussi ajouté un texte latéral sur la gauche du journal. Bien évidemment, je ne vous impose pas cette mise en page, laissez libre cours à votre imagination.

### *Correction*

Dans cette correction, que je vous propose parmi tant d'autres disponibles, j'ai utilisé d'autres agréments CSS3 (dégradés, bordures arrondies et d'autres) qui ne sont pas absolument nécessaires. Cependant, cela rend le tout plus réel.

**Code : HTML**

```
<div role="main" class="news-body">
  <h1>Lorem Ipsum</h1> <!-- Titre -->

  <div class="news-left"> <!-- Manchette gauche -->
    <h1>Dolor sit amet</h1>
    <div>
      Plusieurs paragraphes de texte Lorem Ipsum…
    </div>
  </div>
```

```
  <div class="news-titles"> <!-- Gros titres -->
    <h2>In hac habitasse platea dictumst. Curabitur tellus.
<span>page 2</span></h2>
    <h2>Praesent at metus ac eros accumsan pulvinar. <span>page
3</span></h2>
    <h3>Phasellus tincidunt quam quis tortor mattis volutpat. Donec
risus orci. <span>page 5</span></h3>
  </div>

  <div class="news-first"> <!-- A la une -->
    <h1>Aliquam in neque nunc. Donec.</h2>
    <div>
      Plusieurs paragraphes de texte Lorem Ipsum…
    </div>
  </div>

  <div class="news-page">page 1</div> <!-- Page -->
</div>
```

Cette structure est celle que j'ai trouvé la plus appropriée, cependant, vous pourriez en avoir choisi une totalement différente. Le tout est que le résultat soit lisible facilement. Passons désormais à la feuille de style adaptée au code que je viens de vous présenter.

**Code : CSS**

```
body { /* Arrière plan de la page */
  background-color: #888;
  font-family: "Times New Roman", "Arial Black", serif;
}

div.news-body { /* Conteneur du journal */
  background-image: -moz-linear-gradient(right bottom,
rgb(205,205,205) 70%, rgb(227,227,227) 90%); /* Dégradé de fond */
  background-image: -webkit-linear-gradient(right bottom,
rgb(205,205,205) 70%, rgb(227,227,227) 90%);

  padding: 5px;

  border-radius: 0 10px 0 0; /* Coin haut droit corné */
  -moz-border-radius: 0 10px 0 0;
  -webkit-border-radius: 0 10px 0 0;
  -khtml-border-radius: 0 10px 0 0;
  -ms-border-radius: 0 10px 0 0;

  border-top: 5px double #AAA; /* Bordures bas, gauche simples ;
haut, droit doubles */
  border-right: 3px double #AAA;
  border-bottom: 1px solid #999;
  border-left: 1px solid #AAA;
}

div.news-body > h1 { /* Titre principal */
  font-size: 3em;
  text-align: center;

  font-style: italic;

  margin-top: 0;

  border-bottom: 1px solid black;
}

div.news-body div.news-left { /* Manchette gauche */
  width: 300px;
  float: left;
```

```
  text-align: center;

  padding-right: 5px;
  border-right: 2px solid black;
}

div.news-body div.news-left div { /* Texte de la manchette */
  text-align: justify;

  column-count: 2; /* Deux colonnes */
  column-gap: 1.5em;
  column-rule: 1px solid black;

  -moz-column-count: 2;
  -moz-column-gap: 1.5em;
  -moz-column-rule: 1px solid black;

  -webkit-column-count: 2;
  -webkit-column-gap: 1.5em;
  -webkit-column-rule: 1px solid black;
}

div.news-body div.news-titles { /* Titres */
  text-align: center;
}

div.news-body div.news-titles h2 span, div.news-body div.news-titles
h3 span { /* Titres (annotations de page) */
  font-style: oblique;
  font-size: 0.7em;
}

div.news-body div.news-first { /* A la une */
  text-align: center;
}

div.news-body div.news-first div { /* Texte de la une */
  text-align: justify;

  column-count: 3; /* Trois colonnes */
  column-gap: 1.5em;
  column-rule: 1px solid black;

  -moz-column-count: 3;
  -moz-column-gap: 1.5em;
  -moz-column-rule: 1px solid black;

  -webkit-column-count: 3;
  -webkit-column-gap: 1.5em;
  -webkit-column-rule: 1px solid black;

  padding-left: 5px;
}

div.news-body div.news-page { /* Informations de bas de page (numéro
de page) */
  clear: both; /* En dessous de la manchette */

  text-align: right;
  font-style: oblique;

  margin-top: 5px;
  margin-bottom: 5px;
}
```

Prévisualisation

J'insiste sur le point que le code, dans son état actuel, n'est pas compatible avec I.E. Notamment sur le point des colonnes ; sur ce navigateur, les deux principales colonnes seront affichées (puisqu'elles subissent un simple `float`) cependant, les colonnes de texte ne le seront pas avant la version 10 de ce dernier (prévu). De plus, le fond du « journal » ne s'affichera pas, étant un dégradé CSS3. La feuille de style présentée est donc minime dans sa compatibilité et nécessite de créer une feuille spécialisée pour I.E. utilisant un groupe de propriétés alternatif (comme, par exemple, la technique présentée en début de tutoriel). Nous en venons par cette phrase au paragraphe de conclusion.

*Affichage sur I.E. 8 (cliquez pour agrandir)*

### *En conclusion : que choisir ?*

Parmi les deux méthodes présentées dans ce tutoriel, laquelle choisir ? Tout dépend de votre site et de votre point de vue. Si vous vous axez sur la compatibilité, la première méthode sera, sans doute, celle que vous choisirez. En effet, pour l'affichage en tableaux, seul I.E. peut poser un problème, car il est compatible avec cela depuis sa version 8 uniquement. Tandis que pour les colonnes CSS3, il ne le sera qu'en version 10 ! Globalement, les autres navigateurs supportent assez bien les deux techniques.

Il faut surtout comprendre que les deux techniques ne sont pas utilisées dans le même but. Alors que la première sera utilisée pour mettre en page tout un site, la seconde ne sera utilisée, la plupart du temps, que sur des portions de texte délimitées. Ainsi, vous pourrez même utiliser les deux méthodes simultanément sur la même page d'un site.

En conclusion, le choix dépend toujours de vos besoins.

## Q.C.M.

Le premier QCM de ce cours vous est offert en libre accès.
Pour accéder aux suivants

Connectez-vous Inscrivez-vous

La technique simulant l'affichage d'un tableau modifie-t-elle la structure HTML ?

- ◯ Oui, plusieurs  doivent être ajoutées.
- ◯ Oui, pour une structure simple, le code sera changé, mais le seul grand changement est l'ajout d'une unique balise globale.
- ◯ Non, le code reste inchangé.

La sémantique est-elle respectée, pour la première méthode, malgré l'utilisation de tableaux ?

- ◯ Oui, il s'agit d'ailleurs de la force principale de cette technique : nous ne créons pas vraiment de tableaux.
- ◯ Oui, bien que des tableaux soient utilisés dans le code HTML pour produire des colonnes.
- ◯ Non, l'utilisation de tableaux pour ce genre d'utilisation n'est pas conseillée et cette technique est à proscrire.

L'utilisation des propriétés CSS3 pour la création de colonnes de texte peut-elle être utilisée avec le nom original des propriétés ?

- ◯ Oui, les préfixes propriétaires sont inutiles.
- ◯ Oui, sur certains navigateurs.
- ◯ Non, étant donné la compatibilité de cette technique, l'utilisation des préfixes propriétaires est nécessaire sur tous les navigateurs.

Les colonnes CSS3 sont-elles supportées entièrement par tous les navigateurs, connus, actuels ?

- ◯ Oui et de nouvelles propriétés devraient voir le jour sous peu.
- ◯ Oui.
- ◯ Non, certaines propriétés ne sont pas encore implémentées par beaucoup de navigateurs actuels.
- ◯ Non, beaucoup de navigateurs n'implémentent pas du tout les colonnes CSS3

Correction !

Statistiques de réponses au QCM

Ainsi s'achève cette présentation des principales techniques actuelles et disponibles afin d'afficher du texte en colonnes. Sachez qu'il est possible de réaliser bien d'autres choses avec les simulations de tableaux (première fonctionnalité présentée / propriétés CSS display: table-[X] ). En effet, leur usage n'est pas limité à la mise en page par colonnes.

Les propriétés CSS3 disponibles à cet effet, elles, doivent exclusivement être utilisées pour l'affichage de colonnes. Toutefois, si l'utilisation d'un tableau se révèle utile (présentation de données), n'utilisez pas la simulation de tableaux, mais utilisez la balise `<table>` !

Merci d'avoir lu ce tutoriel et n'hésitez pas à rapporter vos commentaires ou signaler d'éventuelles erreurs !

### *À consulter également :*

*Logo CSS3*

- Mise en page CSS avancée avec la propriété display (Alsacreations - Français)
- D'autres valeurs possibles pour la propriété display en CSS3 (Alsacreations - Français)
- Référence CSS de la propriété display (MDN Docs - Français)
- Spécification Tableaux - Propriété display pour les tableaux (Spécification W3C - Anglais)
- Informations et démonstrations relatives à l'affichage de colonnes en CSS3 (CSS3.info - Anglais)
- Spécification des propriétés pour les colonnes (W3C - Anglais)
- Table de compatibilité de l'affichage en tableaux (Caniuse - Anglais)
- Table de compatibilité de l'affichage en colonnes CSS3 (Caniuse - Anglais)

Partager