

# Pour rétablir le flux après un float, vous êtes plutôt HR, BR ou DIV ?

par Bruno Bichet ([css4design](#))

Date de publication : 28/02/2008

Cet article vous propose des méthodes pour le rétablissement du flux dans vos mises en page CSS.

- I - Introduction
- II - Un HR cross-browser cachère
  - II-A - Deux, c'est trop
  - II-B - C'était mieux avant ?
- III - Styler la balise BR
- IV - La rustine
- V - Quelques mots sur la remise à zéro des éléments HTML
  - V-A - Faire "reset", c'est rigolo tout le temps ?
- VI - En guise de conclusion
- VII - Remerciement

(1)

## I - Introduction

Dans le prolongement du billet précédent concernant une **mise en pages CSS avec la propriété float**, je me suis intéressé aux différentes manières de styler la balise *hr* pour rétablir le flux après en avoir sorti des éléments en les faisant flotter à gauche ou à droite. J'utilise généralement *hr* avec le traditionnel *clear: both* et *visibility: hidden* et je jongle ensuite avec les marges ou la hauteur pour espacer les éléments. Or, depuis que j'utilise le **reset.css** d'Eric Meyer, j'ai tendance à supprimer les marges de tout ce qui bouge, mais cette chère *règle horizontale* reste récalcitrante à la remise à zéro *cross browser*...

## II - Un HR cross-browser cachère

Il existe heureusement des solutions pour palier les insuffisances du dernier de la classe. L'une d'entre elles repose sur l'application d'un *margin-top* et *margin-bottom* négatif :

### Code CSS de la balise hr

```
hr
{
  height: 1px;
  margin: -0.5em 0;
  padding: 0;
  color: #F00;
  background-color: #F00;
  border: 0;
}
```

J'ajoute à cela les propriétés *clear: both* et *visibility: hidden*, je mets la hauteur à zéro et je supprime les couleurs pour obtenir un *spacer* de bon aloi. Les valeurs de *margin* ne convenant pas dans mon cas, j'ai tâtonné pour trouver une valeur satisfaisante :

### Code CSS de la balise hr

```
hr
{
  height: 0;
  margin: -1.2ex 0;
  padding: 0;
  border: 0;
  clear: both;
  visibility: hidden;
}
```

Une autre solution trouvée cette fois sur ultra-fluide transforme le *hr* en bloc pour mieux maîtriser son comportement dans le but de créer une ligne rouge horizontale de 1 pixel d'épaisseur sans marge :

### Code CSS de la balise hr

```
hr
{
  display: block;
  height: 1px;
  margin: 0;
  _margin: -7px 0;
  padding: 0;
  color: #F00;
  background-color: #F00;
  border: 0;
}
```

Pour mes besoins, j'applique la recette précédente :

### Code CSS de la balise hr

```
hr
{
```

#### Code CSS de la balise hr

```
display: block;
height: 0;
margin: 0;
_margin: -7px 0;
padding: 0;
border: 0;
visibility: hidden;
}
```

Il y a bien ce `_margin` qui peut être gênant, mais il reste toujours la possibilité de le caser dans une feuille de style dédiée à IE avec les commentaires conditionnels.

La différence avec la méthode précédente est qu'une fois transformé en *block* (l'auteur de l'astuce ayant habilement déterminé que le *hr* se comportait par certains côtés comme un élément en ligne) le *margin: 0* suffit à supprimer les marges pour les navigateurs modernes. Le *\_margin: -7px 0* s'adressant plus spécifiquement à IE6. Reste à savoir comment se comporte IE7 dans les mêmes circonstances.

## II-A - Deux, c'est trop

Ces deux techniques fonctionnent assez bien pour donner le même rendu aux traits horizontaux ou pour rétablir le flux. Malheureusement pour moi, j'ai deux blocs imbriqués contenant des *float: left* et *float: right*, ce qui implique deux *hr* l'un en dessous de l'autre :

```
</div>
<!-- end .pix -->
<hr class="spacer" />
</div>
<!-- end .contents -->
<hr class="spacer" />
</div>
<!-- end #container -->
</body>
```

Ces deux *hr* successifs m'ennuient (un peu) car en l'absence de prise en charge des feuilles de style, je me retrouve avec deux vilains traits de séparation. Et puis, si je me suis mis à utiliser le *reset.css*, c'est pour éviter au maximum d'utiliser des CSS spécifiques à certains navigateurs, alors, bon...

Je tiens à préciser que j'utilise généralement *clear: both* sur le bloc lui-même au lieu de créer un *spacer* superflu. Mais curieusement, les blocs en question se retrouvaient sous les colonnes latérales... La technique utilisée pour créer les colonnes latérales reposant également sur la propriété *float*, j'ai le sentiment qu'au bout d'un moment on se retrouve dans la situation où il devient interdit d'interdire à un bloc d'avoir des voisins à droite ou à gauche : les *spacers* indépendants deviennent indispensables.

## II-B - C'était mieux avant ?

Avant d'utiliser les *hr* pour rétablir le flux, j'utilisais une `<div class="spacer"></div>` qui fonctionnait furieusement bien malgré l'absence de sémantique associée

### III - Styler la balise BR

J'ai essayé d'utiliser la balise *br* en lui associant un *clear: both*. Tout a l'air de fonctionner, mais si le flux est bien rétabli sous Firefox, IE6 ne passe pas le test de la bordure : elle n'encadre pas l'ensemble de la *div* comme si un *clear: both* n'avait pas d'effet sur un *br*...

C'est dommage parce que lorsqu'il s'agit simplement de rétablir le flux et non de créer une séparation entre deux contenus, cette balise *br* aurait presque été sémantique : un retour à la ligne... Que demander de mieux (oué, que ça fonctionne sous IE6, je sais#) ?

Je garde cette solution sous le coude pour tests complémentaires. En attendant, j'ai trouvé sur [dimension-internet](#) un gabarit utilisant la balise *br* pour rétablir le flux :

#### Code CSS du br

```
br.clearfloat
{
  clear: both;
  height: 0;
  font-size: 1px;
  line-height: 0px;
}
```

## IV - La rustine

En désespoir de cause, je me suis résolu à utiliser `<div class="spacer"></div>` lorsqu'il s'agit seulement de rétablir le flux, et `<div class="spacer"><hr /></div>` lorsque j'ai besoin de séparer aussi des contenus :

### Code CSS du br

```
.spacer
{
  clear: both;
}

.spacer hr
{
  display: none;
}
```

Ce n'est pas super CHIC, mais au moins la feuille de style est allégée et ne nécessite ni hack ni commentaire conditionnel. Si j'utilise souvent ces derniers, ça m'ennuie de les mettre en place pour une ou deux déclarations seulement. Bien évidemment, j'utilise et je recommande fortement l'usage du `hr` pour rétablir le flux lorsqu'un espacement est souhaité entre deux blocs, ou mieux encore, l'application du `clear: both` sur les blocs eux-mêmes.

## V - Quelques mots sur la remise à zéro des éléments HTML

Au cours de mes recherches sur *hr*, je suis tombé sur une intervention de Florent V. qui fait part de ses réserves quant à l'utilisation d'une remise à zéro des éléments. L'utilisation d'un reset présente selon lui deux écueils. D'une part, il serait facile d'oublier de styler une balise et d'autre part, il ne serait pas souhaitable de vouloir tout maîtriser : un peu de lâcher prise serait bienvenue.

Si je partage cette dernière opinion, je ne suis pas totalement d'accord avec la première : les balises HTML ne sont pas en nombre infini, et au pire, en cas d'oubli, il est facile de palier le problème. Pour ma part, en complément de *reset.css*, j'utilise un fichier *html.css* pour donner un nouveau style "par défaut" aux éléments HTML.

Ce fichier est similaire à la partie */\* basics \*/* de l'**exemple** fourni par david larlet de *biologeek*. Notons que la partie */\* reset \*/* de ce fichier ne reprend qu'une partie du *reset.css* d'Eric Meyer.

### V-A - Faire "reset", c'est rigolo tout le temps ?

Il y a évidemment des cas où l'utilisation d'un *reset* est à prendre avec des pincettes : lorsque vous intervenez sur une partie d'une page déjà construite, ou dans une moindre mesure, lorsque d'autres personnes sont susceptibles d'intervenir. Je ne conseillerais pas non plus l'utilisation de la remise à zéro dans le cadre d'une formation : si vous commencez l'étude des CSS, il vaut peut-être mieux connaître les bases avant de faire table rase.

## VI - En guise de conclusion

En tout état de cause, si la théorie nous dicte la bonne conduite à tenir, force est de constater que la pratique nous joue parfois des tours de cochons ^\_^

## VII - Remerciement

Tous mes remerciements à **trotters213** pour sa relecture.

1 :

Cet article a été publié à l'origine le 22 June 2007 et est toujours visible à l'adresse suivante : **Pour rétablir le flux après un float, vous êtes plutôt HR, BR ou DIV ?**