# Positionnement flottant

* A l'origine, le **flottement** sert à pousser, à droite ou à gauche, du texte ou une image, et à remplir l'espace libre avec le reste du contenu
* Cet usage a été **détourné pour positionner** des éléments, mais il peut varier selon le navigateur d'où des erreurs d'affichage
* Un élément flottant est sorti du flux et placé à l'extrême gauche (**float:left**) ou droite (**float:right**) de son conteneur tout en restant sur sa hauteur de ligne initiale, le reste s'écoule autour.

# float:left – float:right

* L'élément défini avec **float:left** est fixé à gauche et la suite du contenu occupe tout le reste à droite
* Avec **float:right**, on a le comportement inverse
* **float:none**, par défaut

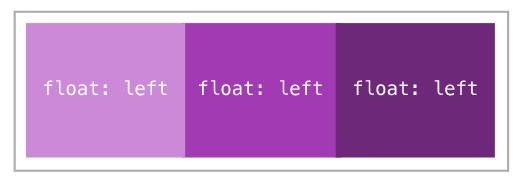
# Des blocs côte à côte

* Un élément **flottant** prend par défaut la **largeur de son contenu**. Si le contenu est important, il vaut mieux fixer avec width ou max-width
* S'il y a un **2ème élément flottant**, il reste sur le même plan et donc se place à ses côtés.
* Attention, les flottants sortent du flux, donc le parent sera vide et le contenu va dépasser. Il faut donc **fixer la hauteur** du parent ou placer

un **élément non flottant** après.

# Plusieurs float

* On peut positionner plusieurs éléments avec la propriété **float:left** ce qui permet d'afficher des blocs successifs sous forme de colonnes.
* Si l'addition des largeurs des différents blocs dépasse la largeur totale, le dernier bloc passe sous les autres.



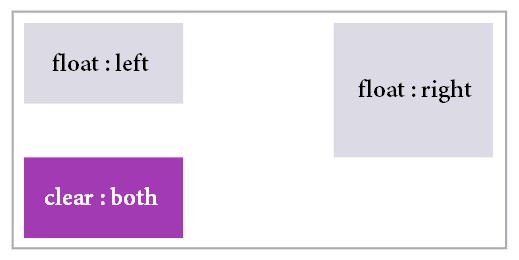
# Annuler le flottement

* La propriété **clear** interdit à un élément de se placer sur la même ligne qu'un bloc flottant et le force à passer en-dessous
* On annule les flottants à gauche (**clear:left**) ou à droite (**clear:right**) ou les deux (**clear:both**)
* **clear** permet, d'**empêcher les dépassements** de flottants, ou de placer un élément **toujours en bas** du flottant le plus long

# clear:both

* Avec la propriété **clear:both**, le dernier élément ne tient plus compte des positionnements flottants définis pour les éléments précédents
* Il reprend donc la position par défaut

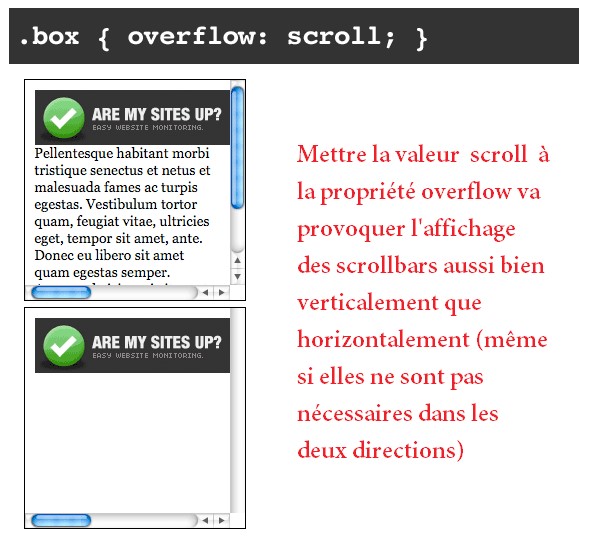
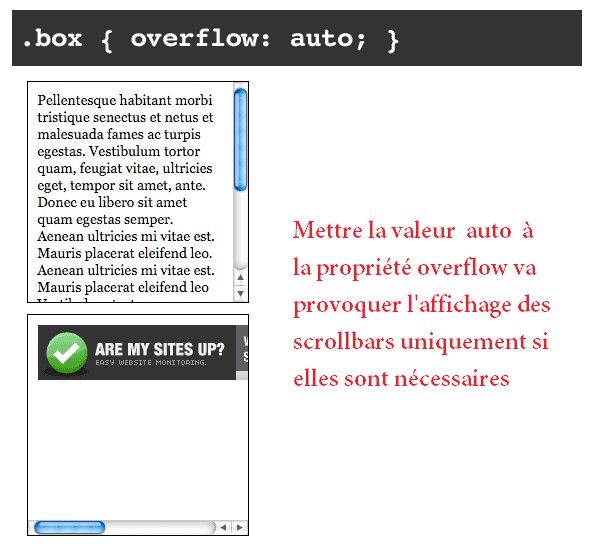
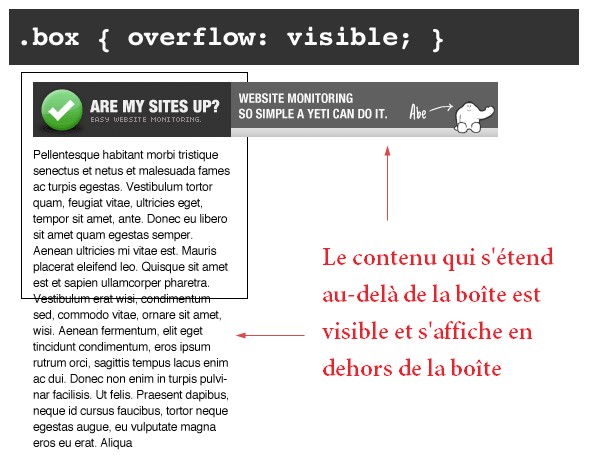
clear:both se met d’office **sous** le dernier float



# Dépassement

* La propriété **overflow** définit la méthode de dépassement des textes de leur conteneur.
  + **visible** : le contenu qui sort est visible.
  + **hidden** : le contenu est rogné aux limites du conteneur, et pas de barre de défilement
  + **scroll** : le contenu est rogné aux limites du conteneur, et des barres de défilement
* **auto** : le choix est laissé aux navigateurs.CSS3 étend ce mécanisme en proposant les propriétés, **overflow-x** et **overflow-y**

# Valeurs pour overflow



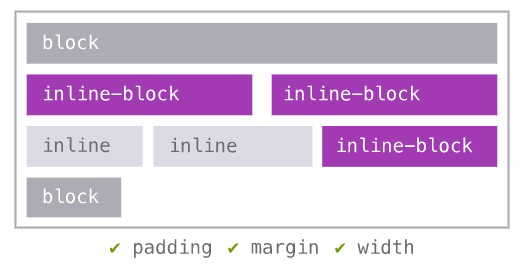
# Découpage

* Si une image est plus grande que l'élément qui la contient, il est possible de la **couper**.
* Cela s'applique aux éléments positionnés avec **position:absolute**
* On définit un rectangle et ses coordonnées avec la propriété **clip**

– Exemple  **clip: rect(0px,60px,200px,0px);**

* La valeur par défaut est **auto** clip va être **déprécié** etremplacé par **clip-path**

# Inline-block

* Avec la propriété **display:inline-block;** on peut combiner les avantages des 2 types principaux
  + Inline : des blocs peuvent se placer l'un à côté de l'autre et rester verticalement alignés
  + Block : pouvoir être dimensionnés
* Avec la propriété **vertical-align,** on peut régler l'alignement des blocs selon la valeur (top, baseline, bottom, middle,...)

# Rendu de tableau en CSS

* Il est possible de faire une **mise en page** sousforme de **tableau**.
* Pas avec <table>, <tr>, <td> !
* Mais avec des valeurs spéciales pour **display**
  + **table, table-row, table-cell**
  + **table-header-group, table-footer-group** – **table-caption,...**