

CSS : Le positionnement

Contenu

| | |
|--|----|
| Le flux et la position relative | 2 |
| Boîte de type bloc (« block ») en flux normal | 2 |
| Boîte de type en-ligne (« inline ») | 3 |
| La position relative | 3 |
| Positions absolue et fixe (« absolute » et « fixed »)..... | 5 |
| Le fonctionnement de la position absolue..... | 5 |
| Ménager un espace pour la boîte en position absolue..... | 6 |
| Utilisation : une mise en page à trois colonnes..... | 7 |
| La position fixe..... | 8 |
| Utilisation : un menu fixe en colonne | 8 |
| Utilisation : une barre de menu fixe..... | 9 |
| Combinaisons des positions fixe et absolue..... | 10 |
| Le positionnement flottant (« float »)..... | 12 |
| Le fonctionnement des flottants..... | 12 |
| Première utilisation : une mise en page à deux colonnes..... | 12 |
| Seconde utilisation : une mise en page à trois colonnes et plus..... | 13 |
| Débordement | 15 |
| Le problème du débordement | 15 |
| Rétablissement du flux normal | 15 |
| A retenir..... | 17 |
| Exercices | 18 |

Le flux et la position relative

Par défaut, les navigateurs affichent les boîtes issues du document html dans l'ordre du flux normal. En position relative, un élément peut être décalé verticalement et/ou horizontalement.

Pour représenter le positionnement en flux normal, on peut imaginer le navigateur parcourant (logiquement) la page de code HTML du début à la fin et retranscrivant son contenu au fur et à mesure des balises rencontrées. Comme dans la lecture d'un texte, il procède verticalement, commençant en « haut » de l'écran pour aller jusqu'en « bas », et horizontalement de gauche à droite, sur la totalité de l'espace disponible et nécessaire en largeur comme en hauteur.

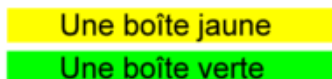
Boîte de type bloc (« block ») en flux normal

Par défaut, les boîtes de type bloc seront affichées dans une succession verticale. Prenons par exemple deux blocs différenciés par leur couleur :

```
<!-- HTML -->
<p class="jaune">Une boîte jaune</p>
<p class="verte">Une boîte verte</p>
```

```
/* CSS */
.jaune {
  background-color: #ffff00;
}
.verte {
  background-color: #00ff00;
}
```

Résultat :



Le navigateur traite successivement les deux éléments rencontrés. Comme il s'agit d'éléments de type bloc, il aligne la marge gauche de chaque élément sur la marge gauche de l'élément conteneur, c'est à dire ici le bloc conteneur initial.

Les principaux éléments créant des boîtes bloc sont :

- l'élément `div` ;
- les titres `h1`, `h2`, `h3`, `h4`, `h5`, `h6` ;
- le paragraphe `p` ;
- Les listes et éléments de liste `ul`, `ol`, `li`, `dl`, `dd` ;
- Le bloc de citation `blockquote` ;
- Le texte pré-formaté `pre` ;
- L'adresse `address`.

Boîte de type en-ligne (« inline »)

Par défaut, les boîtes de type en ligne sont affichées dans une succession horizontale. Prenons par exemple deux portions de texte différenciées par leur couleur :

```
<!-- HTML -->
<p>
  <span class="jaune">Une boîte jaune</span>
  <span class="verte">Une boîte verte</span>
</p>
```

```
/* CSS */
.jaune {
  background-color: #ffff00;
}
.verte {
  background-color: #00ff00;
}
```

Résultat

Une boîte jauneUne boîte verte

Les principaux éléments créant des boîtes en ligne sont :

- l'élément `span` ;
- le lien `a` ;
- L'image `img` ;
- Les emphases `em` et `strong` ;
- La citation `q` ;
- La citation `cite` ;
- L'élément `code` ;
- Le texte entré par l'utilisateur `kbd` ;
- L'exemple `samp` ;
- La variable `var` ;
- Les abréviations et acronymes `abbr`, `acronym` ;
- Le texte inséré `ins` ;
- Le texte supprimé `del`.

Pour résumer le flux normal : c'est un traitement linéaire du contenu de la page. L'alignement des boîtes bloc est vertical ; l'alignement des éléments en ligne dans les boîtes bloc est horizontal.

La position relative

Le positionnement relatif permet d'inscrire un contenu en flux normal, puis de le décaler horizontalement et/ou verticalement. Le contenu suivant n'est pas affecté par ce déplacement, qui peut donc entraîner des chevauchements.

Soit par exemple un positionnement relatif indice qui stipule un décalage vers le haut de 5 pixels et un arrière-plan jaune :

```
<!-- HTML -->
<p>
  Lorem
  <span class="jaune">
    boîte en position relative
  </span>
  ipsum dolor.
</p>
```

```
/* CSS */
.jaune {
  position: relative;
  bottom: 5px;
  background-color: #ffff00;
}
```

Résultat :

Lorem boîte en position relative ipsum dolor

Pour illustrer le risque de chevauchement, ajoutons un décalage vers la droite :

```
/* CSS */
.jaune {
  position: relative;
  bottom: 5px;
  left: 3em;
  background-color: #ffff00;
}
```

Résultat

Lorem boîte en position relative ipsum dolor

Positions absolue et fixe (« absolute » et « fixed »)

La position absolue et la position fixe permettent de placer une boîte par rapport aux limites de la zone d’affichage ou du conteneur. Comment les utiliser dans une mise en page CSS ?

Une boîte en positionnement absolu peut être placée n'importe-où dans le code HTML et s'afficher à l'endroit de votre choix. Ceci s'avère très utile en particulier pour :

- placer les menus de navigation en fin de page, pour améliorer l'accessibilité de votre site en donnant un accès immédiat à son contenu dans les navigateurs textes, tout en les faisant apparaître en haut de page ou encore dans une colonne pour les navigateurs graphiques
- créer plusieurs colonnes au positionnement indépendant de l'ordre dans lequel elles se trouvent en HTML

Le fonctionnement de la position absolue

Le positionnement absolu « retire » totalement du flux le contenu concerné : sa position est déterminée par référence aux limites du conteneur. Celui-ci peut-être :

- une boîte elle-même positionnée (position relative ou absolue) ;
- le bloc conteneur initial, à défaut de boîte positionnée, c'est à dire en pratique le plus souvent la fenêtre du navigateur.

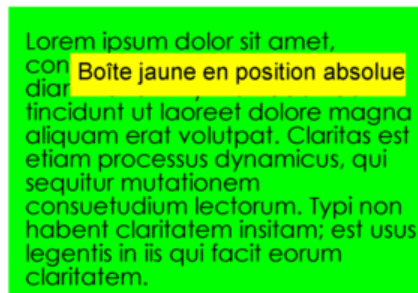
Définissons un conteneur « verte » en position relative :

```
/* CSS */
.verte {
  position: relative;
  background-color: #00ff00;
  width: 15em;
}
```

Et un positionnement absolu « jaune » :

```
/* CSS */
.jaune {
  position: absolute;
  top: 1em;
  right: 1em;
  background-color: #ffff00;
}
Et appliquons ces styles :
<div class="verte">
  <p>
    ...
  </p>
  <p class="jaune">
    Boîte jaune en position absolue
  </p>
</div>
```

Résultat

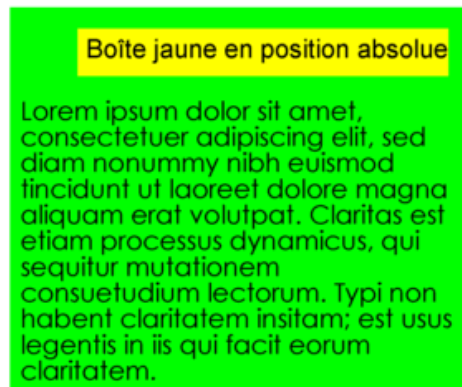


Ménager un espace pour la boîte en position absolue

Inévitablement, la boîte en position absolue recouvre le contenu de notre paragraphe. Pour l'éviter, dotons la boîte conteneur verte d'un remplissage supérieur suffisant :

```
/* CSS */
.verte {
  padding-top: 5em;
}
```

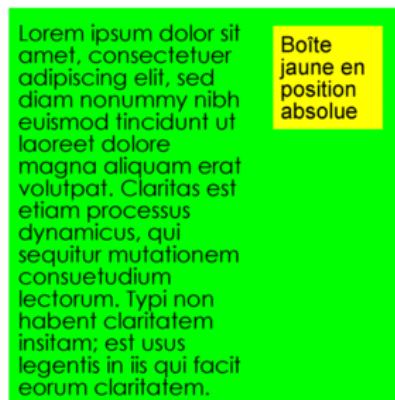
Le résultat, illustré par la figure ci-dessous, peut être observé également dans l'exemple 2



Nous pourrions aussi bien fixer la largeur de la boîte jaune et doter le texte de la boîte verte d'un remplissage à droite :

```
/* CSS */
.verte {
  padding-right: 7em;
}
.jaune {
  width: 4em;
}
```

Résultat



Utilisation : une mise en page à trois colonnes

La position absolue offre donc une alternative aux flottants pour réaliser des mises en pages à plusieurs colonnes. Le cas typique est celui des trois colonnes dont voici le code :

```
/* CSS */
body
{
    background-color: blue;
}

#centre
{
    margin: 1em 25%;
    background-color: #FFF;
}

#gauche
{
    position: absolute;
    top: 1em;
    left: 1%;
    width: 18%;
}

#droite
{
    position: absolute;
    top: 1em;
    right: 1%;
    width: 18%;
}

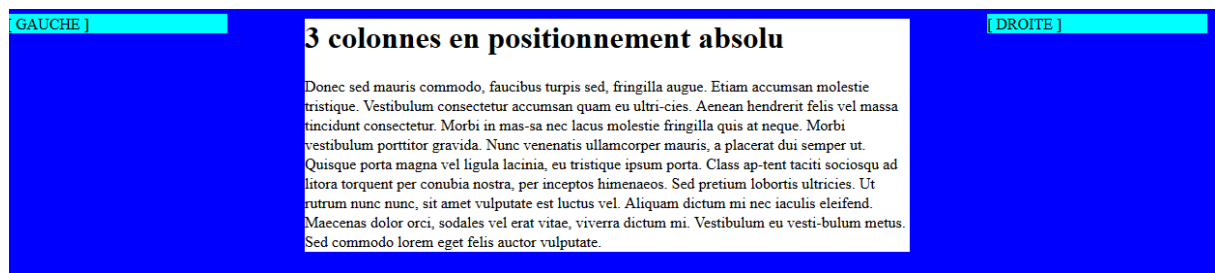
.aqua
{
    background-color: #00FFFF; /* aqua */
}
```

```

<!-- HTML -->
<body>
  <div id="centre">
    <h1>3 colonnes en positionnement absolu</h1>
    <p>Donec sed mauris commodo, faucibus turpis sed, fringilla augue. Etiam
    accumsan molestie tristique. Vestibulum consectetur accumsan quam eu ultricies.
    Aenean hendrerit felis vel massa tincidunt consectetur. Morbi in massa nec lacus
    molestie fringilla quis at neque. Morbi vestibulum porttitor gravida. Nunc
    venenatis ullamcorper mauris, a placerat dui semper ut. Quisque porta magna
    vel ligula lacinia, eu tristique ipsum porta. Class aptent taciti sociosqu ad
    litora torquent per conubia nostra, per inceptos himenaeos. Sed pretium lobortis
    ultricies. Ut rutrum nunc nunc, sit amet vulputate est luctus vel. Aliquam
    dictum mi nec iaculis eleifend. Maecenas dolor orci, sodales vel erat vitae,
    viverra dictum mi. Vestibulum eu vestibulum metus. Sed commodo lorem eget felis
    auctor vulputate.</p>
  </div>
  <div id="gauche" class="aqua">
    [ GAUCHE ]
  </div>
  <div id="droite" class="aqua">
    [ DROITE ]
  </div>
</body>

```

Résultat



La position fixe

Comme dans un positionnement absolu, le contenu concerné est retiré totalement du flux. Mais il est cette fois positionné uniquement par rapport aux limites de la zone de visualisation, autrement dit la fenêtre du navigateur. Le défilement de la page n'a aucun effet sur un contenu en position fixe.

Tout comme le positionnement absolu, le positionnement fixe est susceptible de provoquer des chevauchements. On emploiera donc des méthodes similaires pour l'exploiter.

Utilisation : un menu fixe en colonne

Pour fixer un menu de navigation dans une colonne vide à gauche du contenu, on définira les largeurs du contenu et du menu :

```

/* CSS */
.content {
  width: 65%;
  border: 1px solid #000000;
}

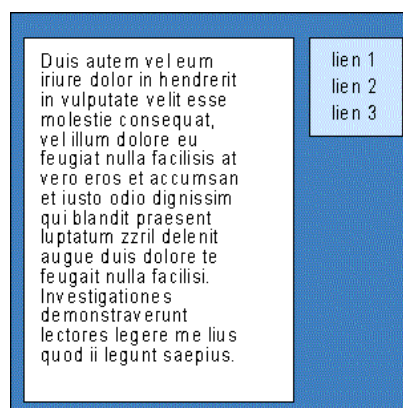
```



```
padding: 1em 1%;
}
.menu {
position: absolute;
top: 1em;
right: 1%;
border: 1px solid #000000;
padding: 1em;
width: 20%;
}
html>body .menu {
position: fixed;
}
```

```
<!-- HTML -->
<div class="content">
...
</div>
<div class="menu">
...
</div>
</body>
```

Résultat



Utilisation : une barre de menu fixe

Lorsqu'il s'agit de créer une mise en page avec contenu et menus, la position fixe a l'inconvénient apparent de sacrifier une partie de la largeur d'affichage disponible, pour éviter les chevauchements.

En fait, cet inconvénient disparaît sitôt qu'on tire profit du chevauchement au lieu de chercher à l'éviter. Imaginons une barre de menu horizontale, fixée en permanence en haut de la fenêtre d'affichage :

```
/* CSS */
.menufixe {
top: 0;
left: 0;
width: 100%;
border: 1px solid #000000;
position: absolute;
```

```

    z-index: 2;
    text-align: center;
    background-color: #ffffff;
    line-height: 2em;
}

html>body .menufixe {
    position: fixed
}

.content {
    z-index: 1;
    padding-top: 3em;
}

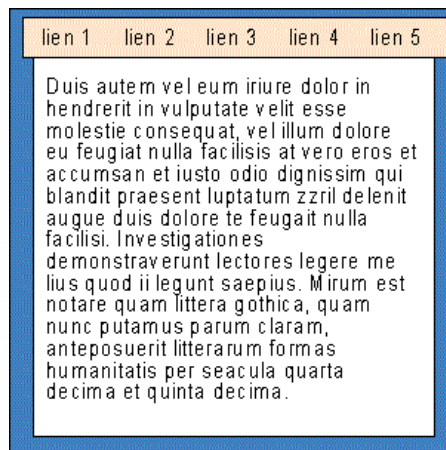
```

```

<!-- HTML -->
<body>
  <div class="content">
    ...
  </div>
  <p class="menufixe">
    ...
  </p>
</body>

```

Résultat



Combinaisons des positions fixe et absolue

Imaginons à présent deux menus différents, tous deux placés dans la même colonne :

- le premier sera fixe en haut de la colonne ;
- le second sera en position absolue plus bas dans la même colonne.

La propriété CSS `z-index` permet de spécifier le *niveau d'empilement* de chaque menu : avec un `z-index` supérieur à celui du menu fixe et un arrière-plan opaque, le menu en position absolue masquera temporairement celui-ci lors du défilement de la page. Le code nécessaire est :

```

/* CSS */
.content {

```

```

width: 65%;
border: 1px solid #000000;
padding: 1em 1%;
}
.menufixe {
position: absolute;
top: 1em;
right: 2%;
border: 1px solid #000000;
padding: 1em;
width: 18%;
z-index: 2;
}
html>body .menufixe {
position: fixed;
}
.menuabsolu {
position: absolute;
top: 15em;
right: 1%;
border: 1px solid #000000;
padding: 1em;
width: 20%;
background-color: #ffffff;
z-index: 3;
}

```

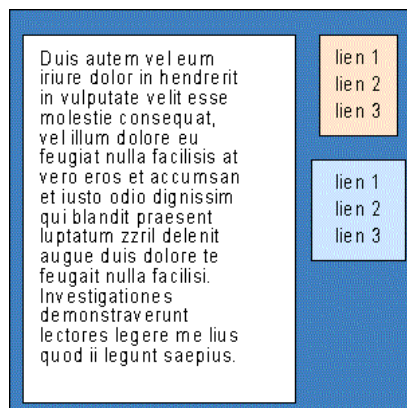
```

<!-- HTML -->
<body>
  <div class="content">
    ...
  </div>
  <div class="menu">
    ...
  </div>
</body>

```

Le point délicat est, bien-sûr, le choix de la valeur donnée à la propriété `top` du menu en position absolue, afin d'être sûr qu'il n'empiète pas sur l'espace du menu en position fixe.

Résultat



Le positionnement flottant (« float »)

Une boîte flottante est retirée du flux normal, et placée le plus à droite (float: right) ou le plus à gauche (float: left) possible dans son conteneur. Le contenu suivant cette boîte flottante s'écoule le long de celle-ci, dans l'espace laissé libre.

Le fonctionnement des flottants est complexe (problème du dépassement) et pose souvent difficulté si on n'en maîtrise pas bien les règles.

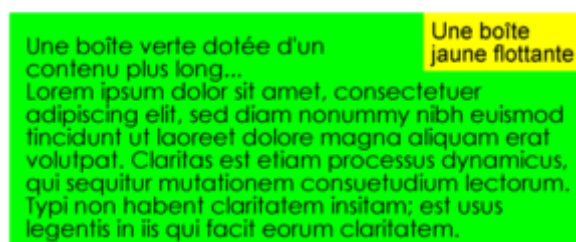
Le fonctionnement des flottants

Ajoutons aux styles précédents une règle de positionnement flottant à droite et une mesure de largeur.

```
<!-- HTML -->
<p class="jaune">Une boîte jaune flottant</p>
<p class="verte">Une boîte verte doté d'un contenu plus long...</p>
```

```
/* CSS */
.jaune {
  background-color: #ffff00;
  float: right;
  width: 100px;
  margin: 0;
}
.verte {
  background-color: #00ff00;
}
```

Résultat :



Le navigateur place tout d'abord la boîte jaune, alignée sur le bord droit de cette partie de la page, puis reprend le cours normal du flux dans l'espace laissé libre à sa gauche pour afficher la boîte verte. Le flux occupe tout l'espace disponible : la boîte verte reprend donc toute la largeur de la page sitôt *passée* la limite inférieure de la boîte flottante jaune.

Première utilisation : une mise en page à deux colonnes

La combinaison de la position flottante et du flux permet de réaliser aisément une mise en page à deux colonnes (un contenu et un menu) :

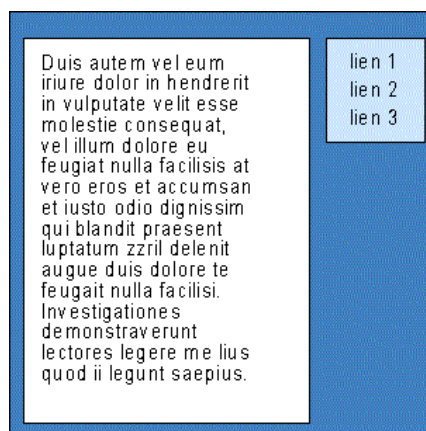
```
/* CSS */
```

```
.content {
  float: left;
  width: 70%;
}
.menu {
  margin-left: 80%;
  border: 1px solid #000000;
  padding: 1em;
}
```

```
<!-- HTML -->
<body>
  <div class="content">
    ...
  </div>

  <div class="menu">
    <ul>
      <li>lien_1</li>
      <li>lien_2</li>
      <li>lien_3</li>
    </ul>
  </div>
</body>
```

Résultat



Seconde utilisation : une mise en page à trois colonnes et plus

On peut tout aussi bien multiplier les flottants pour créer autant de colonnes que souhaité, mais on rentre ici dans un problème un peu plus complexe car il faudra tenir compte des marges et padding afin d'attribuer une largeur correcte aux blocs.

Exemple avec le code ci-dessous :

- Largeur totale = 100%
- On aurait donc $100\% / 3 = 33.33\%$ pour chaque bloc

- Cependant, comme on souhaite des marges et que les marges viennent s'ajouter à la largeur des div, on dépasserait 100% et le dernier bloc serait "poussé" sur la ligne suivante

Il faut donc tenir compte des marges des blocs flottants :

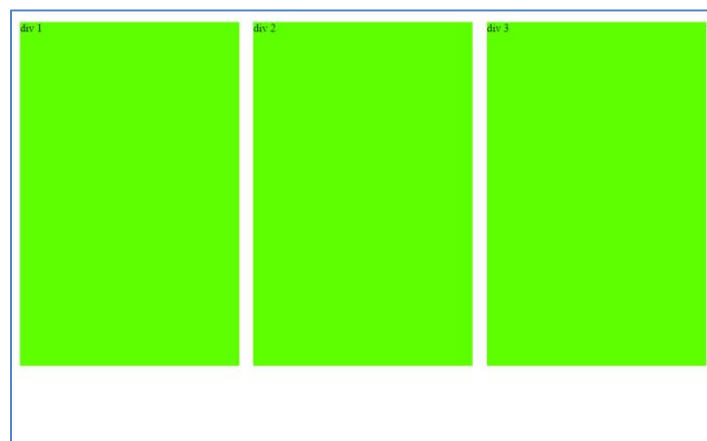
- On a trois blocs, avec une marge gauche et une marge droite, donc $3 \times 2 = 6$ marges, donc $6 \times 1\% = 6\%$
- Il faut donc retirer les 6% de marges de la largeur totale : $100 - 6 = 94\%$
- On divise donc 94% par 3 blocs = 31.33%

On constate qu'il est nécessaire de sortir la calculatrice et cela peut vite devenir quelque peu rébarbatif !

```
/* CSS */
.left
{
    float: left;
    width: 31.33%;
    background-color: #00ff00;
    margin: 1%;
    height: 500px;
}
```

```
<!-- HTML -->
<body>
  <div class="left">div 1</div>
  <div class="left">div 2</div>
  <div class="left">div 3</div>
</body>
```

Résultat



Débordement

Le problème du débordement

Avec le positionnement flottant, mais aussi dès lors qu'un bloc a une hauteur et/ou une largeur imposées, il se peut que la place manque pour afficher son contenu.

Exemple :

La propriété `overflow` permet de définir le comportement qu'adoptera le bloc dans ce cas :

| Valeur de la propriété <code>overflow</code> | Rôle |
|--|---|
| auto | Ajoute des barres de défilement horizontale et verticale au bloc si nécessaire |
| hidden | Cache tout ce qui dépasse |
| inherit | Le bloc se comporte de la même manière que son bloc parent |
| scroll | Ajoute une barre de défilement (ascenseur) horizontale et verticale au bloc |
| visible | rend visible tout ce qui dépasse (quitte à ce que le texte qui déborde chevauche le reste le page...) |

Rétablissement du flux normal

Dès qu'il y a présence d'un bloc flottant, il est nécessaire de rétablir le flux afin que les éléments HTML suivants se positionnent de façon normale.

Pour cela on utilise la propriété `clear` qui indique si un élément peut être situé à côté d'éléments flottants qui le précèdent ou s'il doit être déplacé vers le bas pour être en dessous de ces éléments. La propriété `clear` s'applique aux éléments flottants comme aux éléments non-flottants.

| Valeur de la propriété <code>clear</code> | Rôle |
|---|--|
| none | l'élément n'est pas déplacé vers le bas pour dégager le flottement. |
| left | l'élément est déplacé vers le bas afin de dégager les flottements à gauche. |
| right | l'élément est déplacé vers le bas afin de dégager les flottements à droite. |
| both | Un mot-clé qui indique que l'élément est déplacé vers le bas afin de dégager les flottements à gauche et à droite. |
| inline-start | l'élément est déplacé vers le bas pour dégager le contenu vers le début du bloc englobant. |
| inline-end | l'élément est déplacé vers le bas pour dégager le contenu du côté de la fin du bloc englobant. |

En général, sur les flottants, on utilise la valeur `both` de la propriété `clear`.

Reprenez le dernier exercice sur les flottants. Ajouter un élément censé se positionner sous les flottants :

```
/* CSS */
#suivant
{
    background-color: yellow;
}
```

```
<!-- HTML -->
<body>
    <div class="left">div 1</div>
    <div class="left">div 2</div>
    <div class="left">div 3</div>

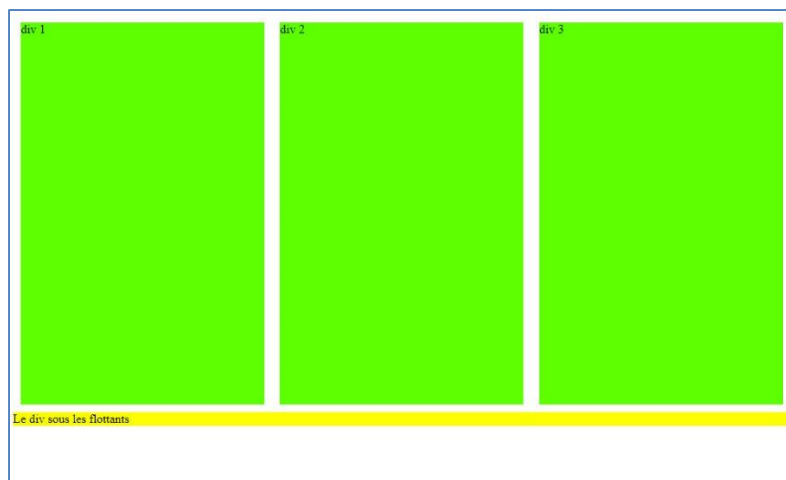
    <div id="suivant">Le div sous les flottants</div>
</body>
```

Observez ce qui se passe : vous devriez avoir le nouvel élément sous les 3 colonnes flottantes vertes (en réalité ce sont plutôt les flottants qui sont au-dessus), or ce n'est pas ce qu'on souhaite.

Ajoutez la ligne suivante dans le CSS du nouveau bloc :

```
/* CSS */
#suivant
{
    background-color: yellow;
    clear : both;
}
```

Cette fois, le bloc `#suivant` est bien positionné en dessous des flottants :



A retenir

Il existe 5 types de positionnements en CSS :

| Positionnement | Rôle |
|----------------|---|
| flux normal | positionnement naturel des éléments, établi par le navigateur en l'absence d'instructions CSS |
| relatif | inscrit le contenu en flux normal, puis le décale horizontalement et/ou verticalement. Le contenu suivant n'est pas affecté par ce déplacement, qui peut donc entraîner des chevauchements. |
| absolu | <ul style="list-style-type: none">▪ Retrait du flux normal▪ Position déterminée par référence aux limites du conteneur |
| fixe | <ul style="list-style-type: none">▪ Retrait du flux normal▪ Position déterminée par référence aux limites de la fenêtre▪ L'élément reste immobile en cas de défilement de la page |
| flottant | <ul style="list-style-type: none">▪ Retrait du flux normal▪ Placement le plus à droite (float: right) ou le plus à gauche (float: left) du conteneur |

Exercices

Reproduisez les exemples donnés dans ce cours.