Fonctionnement de la propriété position

La propriété **`position`** est une propriété CSS très puissante qui va nous permettre de définir un type de positionnement pour nos éléments.

On va ainsi pouvoir positionner un élément relativement à partir de sa position par défaut ou de façon absolue par rapport à un point donné dans la page en utilisation **`position`** conjointement avec les propriétés **`top`**, **`left`**, **`bottom`** et **`right`**.

Dans cette leçon, nous allons découvrir les différentes valeurs qu'on va pouvoir donner à **`position`** et apprendre à les utiliser intelligemment en tentant de comprendre leurs implications.

Nous allons pouvoir gérer et modifier le type de positionnement d'une élément HTML grâce à la propriété CSS **`position`**.

La propriété `position` ne va pas nous permettre de positionner un élément en soi dans une page mais simplement de définir un type de positionnement grâce aux valeurs suivantes :

- **`position : static;`**
- **`position : relative;`**
- **`position : absolute;`**
- **`position : fixed;`**
- **`position : sticky.`**

Position : static

La valeur **static** est la valeur par défaut de la propriété **position**. Ainsi, par défaut, tous les éléments HTML sont positionnés de manière **static**. Un élément HTML positionné avec **position : static** sera positionné selon le flux normal de la page.

Notez ici que les propriétés **top, left, bottom** et **right** n'auront aucun effet sur les éléments positionnés avec **position : static**.

```
<p>Un premier paragraphe</p>
    <p class="statique">Un deuxième paragraphe</p>
    <p class="statique gauche">Un troisième
paragraphe</p>
```

```
p{
    background-color: #0CC;
  }

  .statique{
    position: static; /*Valeur par
défaut*/
  }

  .statique.gauche{
    left: 100px; /*La propriété ne
s'applique pas sur un élément static*/
  }
```

Position : relative

Attribuer une `position : relative` à un élément va positionner l'élément dans le flux normal de la page tout comme la valeur **`static`** de la propriété **`position : static.`**

Cependant, à l'inverse d'un élément HTML positionné avec **`position : static,`** un élément positionné avec **`position : relative`** va ensuite pouvoir être décalé par rapport à sa position initiale grâce aux propriétés **`top, left, bottom`** et **`right`**.

Ces propriétés vont prendre comme origine la position initiale de l'élément. Nous allons ainsi pouvoir positionner un élément relativement à sa position de départ.

Notez qu'ici l'espace occupé initialement par l'élément va continuer à lui appartenir : les autres éléments ne seront pas affectés par le décalage de notre élément et ne vont pas se repositionner en fonction de celui-ci.

Cela implique également que l'élément décalé va pouvoir être à cheval par-dessus d'autres éléments puisque a position de ces autres éléments ne va pas changer en fonction de l'élément décalé possédant une **`position : relative`**

```
  <p>Un premier paragraphe</p>
    <p class="relatif haut">Un deuxième
paragraphe</p>
    <p>Un <span class="relatif
gauche">troisième</span> paragraphe</p>
    <br><br>
    <div class="carre bleu"></div>
    <div class="carre vert relatif haut droite">
</div>
    <div class="carre bleu"></div>
```

```css
p{
    background-color: #0CC;
  }
  div{
    width: 100px;
    height: 100px;
    display: inline-block;
    border: 1px solid black;
    box-sizing: border-box;
  }
  .relatif{
    position: relative;
  }
  .haut{
    top: 60px;  /*Elément décalé de 50px vers le bas p/r à une origine*/
  }
  .gauche{
    left: 150px;  /*Element décalé de 150px vers la droite p/r à une origine*/
  }
  .droite{
    right: 30px;  /*Element décalé de 30px vers la gauchr p/r à une origine*/
  }
```

```css
.bleu{
  background-color: #0AF;
}
.vert{
  background-color: #0FA;
}
```

Position : absolute

Un élément positionné avec **`position: absolute`** va être positionné par rapport à son parent le plus proche positionné (avec une valeur de position différente de **`static`**).

Si aucun parent positionné n'est trouvé, alors l'élément sera positionné par rapport à l'élément racine représentant la page en soi.

Le point de référence pour les propriétés `top`, **`left`**, **`bottom`** et **`right`** va ainsi être le côté de l'élément parent liée à la propriété (côté gauche pour **`left`**, supérieur pour **`top`** , etc.).

De plus, un élément positionné avec **`position : absolute`** va être retiré du flux normal de la page. Cela signifie et implique que l'espace initialement attribué à un élément au positionnement absolu (espace attribué selon le flux normal de la page) va être occupé par les éléments suivants.

Un élément positionné avec **`position: absolute`** va ainsi pouvoir se placer par-dessus d'autres éléments.

Position : Fixed

Le positionnement fixe est très proche du positionnement absolu. Un élément positionné avec **`position: fixed`** va également être retiré du flux de la page et l'espace qui lui était attribué selon le flux normal de la page va également pouvoir être utilisé par d'autres éléments.

La seule différence entre **`position: fixed`** et `position: absolute` est que l'élément ne va plus être positionné par rapport à son parent le plus proche mais par rapport au viewport, c'est-à-dire par rapport à la fenêtre visible à moins que l'un de ses parents possède une propriété **`transform`**, **`filter`** ou **`perspective`** dont la valeur est différente de `none`.

En dehors de ces cas particuliers, un élément positionné avec **`position: fixed`** apparaitra toujours à la même place même dans la fenêtre si on descend ou on monte dans la page : il sera fixe par rapport à la fenêtre. En effet, sa position va être calculée par rapport à la fenêtre visible.

A noter ici une exception pour les contenus paginés : dans ce cas-là, l'élément possédant une **`position: fixed`** sera répété dans chaque page.

```css
body,div,p{
    margin: 0px;
  }
  p{
    text-align: justify;
  }
  .carre{
    width: 200px;
    height: 400px;
    border: 1px solid black;
  }
  .bleu{
    background-color: #0AF;
  }
  .vert{
    background-color: rgb(180, 244, 2);
  }
  .fixed{
    position: fixed;
  }
  .droite{
    right: 0px;
  }
```

Position : Affichage Z-Index

Les éléments HTML vont pouvoir être positionné dans une page en CSS selon 3 dimensions : selon la largeur (axe horizontal ou axe des X en math), la hauteur (axe vertical ou axe des Y) et également selon une épaisseur ou un ordre d'empilement (axe des Z).

En effet, vous avez pu remarquer dans les exemples précédents que lorsque deux éléments se chevauchaient, il y en avait toujours un au-dessus de l'autre : il y a donc une notion d'ordre d'empilement selon cet axe 3D qui est l'axe des Z.

La propriété **`z-index`** va nous permettre de choisir quel élément doit apparaitre au-dessus de quel ordre en donnant un index sous forme de nombre à un ou plusieurs éléments. Ainsi, lorsque deux éléments se chevauchent, celui possédant la plus grande valeur pour son `z-index` apparaitra au-dessus de l'autre.

Notez que la propriété CSS **`z-index`** ne va fonctionner (et n'a de sens) qu'avec des éléments HTML positionnés, c'est-à-dire qu'avec des éléments possédant une propriété **`position`** dont la valeur est différente de **`static`** en CSS.

```
<h2>Sans z-index :</h2>
<div class="carre bleu relatif"></div>
<div class="carre relatif vert decale"></div>

<h2>Avec z-index :</h2>
<div class="carre bleu relatif z100"></div>
<div class="carre relatif vert decale"></div>
```

```css
p{
    background-color: #0CC;
  }
  div{
    width: 500px;
    height: 500px;
    display: inline-block;
    border: 1px solid black;
    box-sizing: border-box;
  }
  .bleu{
    background-color: rgb(241, 7, 210);}
  .vert{
    background-color: rgb(14, 162, 203);}

  .relatif{position: relative}
  .decale{
    right: 75px;
    top: 25px;
  }

  .z100{
    z-index: 100;
  }
```