Architecture et langages Web
(NFA040)

CSS : positionnement (tutoriel)

Remarque. Ce tutoriel est extrait de la section de MDN intitulée : Introduction au positionnement. Les sources de l'exemple, rappelés ci-dessous, accompagnent ce sujet.

Le positionnement permet de sortir les éléments du flux normal de la composition du document, et de les faire se comporter différemment, par exemple en plaçant un élément sur un autre ou en occupant toujours la même place dans la zone d'affichage du navigateur. Cet article explique les diverses valeurs de `position`, et comment les utiliser.

Style CSS

```css
body {
  width: 500px;
  margin: 0 auto;
}

p {
  background: aqua;
  border: 3px solid blue;
  padding: 10px;
  margin: 10px;
}

span {
  background: red;
  border: 1px solid black;
}
```

Body HTML

```html
<h1>Basic document flow</h1>

    <p>I am a basic block level element. My adjacent block level elements sit
on new lines below me.</p>

    <p>By default we span 100% of the width of our parent element, and our
height is as tall as our child content. Our total width and height is our
content + padding + border width/height.</p>

    <p>We are separated by our margins. Because of margin collapsing, we are
separated by the width of one of our margins, not both.</p>

    <p>inline elements <span>like this one</span> and <span>this one</span>
sit on the same line as one another, and adjacent text nodes, if there is
space on the same line. Overflowing inline elements <span>wrap onto a new line
if possible - like this one containing text</span>, or just go on to a new
line if not, much like this image will do: <img src="long.jpg" alt="a wide but
short section of a photo of several fabrics"></p>
```

Introduction au positionnement

Le positionnement permet de modifier le cours classique de la mise en page pour produire des effets intéressants. Vous souhaitez modifier légèrement le placement de boîtes par rapport à leur position par défaut dans la mise en page, et donner ainsi une touche d'originalité à votre page ? Vous souhaitez créer un élément d'interface utilisateur flottant au-dessus d'autres parties de la page, et/ou que cet élément reste fixé à la même place dans la fenêtre du navigateur, quel que soit le point de défilement de la page ? Le positionnement est l'outil qu'il vous faut, il rend de tels agencements possibles.

Il y a différents types de positionnement que vous pouvez appliquer à des éléments HTML. Pour utiliser un type particulier de positionnement sur un élément, nous utilisons la propriété `position`.

1 Positionnement statique

Le positionnement statique est celui reçu par défaut par chaque élément. Cela veut tout simplement dire « positionner l'élément selon le flux normal, rien de spécial à voir ici ».

Pour illustrer ce positionnement (et disposer d'exemple qui nous servira pour les prochaines sections), ajoutez tout d'abord une classe `positioned` pour le deuxième `<p>` dans le HTML :

```
<p class="positioned"> ... </p>
```

Puis ajoutez la règle suivante au bas de votre CSS :

```
.positioned {
  position: static;
  background: yellow;
}
```

Si vous sauvegardez et actualisez, vous verrez qu'il n'y a aucune différence, à l'exception de la mise à jour de la couleur de l'arrière-plan du deuxième paragraphe. C'est correct, comme nous l'avons vu plus tôt, le positionnement statique est le comportement par défaut !

Exercice. Tester la feuille de style décrite ci-dessus, et comparez votre résultat à celui-ci : `1_static-positioning.html`

2 Positionnement relatif

Le positionnement relatif est le premier type de positionnement que nous allons étudier. Il est très similaire au positionnement statique. Cependant, une fois que l'élément positionné occupe une place dans le cours normal de la mise en page, vous pourrez modifier sa position finale. Vous pourrez par exemple le faire chevaucher d'autres éléments de la page. Poursuivons : mettez à jour la déclaration de `position` dans le code :

```
position: relative;
```

Si vous sauvegardez et actualisez à ce stade, vous ne verrez aucun changement dans le résultat. Alors, comment modifier la position de l'élément ? Vous avez besoin d'employer les propriétés `top`, `bottom`, `left` et `right` dont nous parlerons dans le prochain paragraphe.

Présentation de `top`, `bottom`, `left` et `right`

`top`, `bottom`, `left` et `right` sont utilisés conjointement à `position` pour définir exactement là où placer l'élément positionné. Pour le tester, ajoutez les déclarations suivantes à la règle `.positioned` dans le CSS :

```
top: 30px;
left: 30px;
```

Exercice. Tester la feuille de style décrite ci-dessus, et comparez votre résultat à celui-ci : `2_relative-positioning.html`

Cool, n'est-ce pas ? Oui, mais ce n'était probablement pas ce à quoi vous vous attendiez. Pourquoi le déplacement s'est-il effectué vers le bas et à droite si nous avons défini `top` (haut) et `left` (gauche) ? Même si cela peut paraître illogique, c'est la façon dont fonctionne le positionnement relatif. Songez à une force invisible poussant le côté spécifié de l'élément à positionner, le déplaçant ainsi dans la direction opposée. Par exemple, si nous spécifions `top: 30px;`, une force pousse le haut de la boîte, entraînant son déplacement vers le bas de 30px.

3 Positionnement absolu

Le positionnement absolu nous apporte des résultats bien différents.

Appliquer `position: absolute`

Modifions la déclaration de `position` dans le code :

```
position: absolute;
```

Exercice. Tester la feuille de style décrite ci-dessus, et comparez votre résultat à celui-ci : `3_absolute-positioning.html`

Tout d'abord, notez que l'emplacement où l'élément à positionner aurait dû se trouver dans le cours normal de la mise en page du document ne s'y trouve plus. Le premier élément et le troisième sont l'un à côté de l'autre comme si le second n'existait plus ! Dans un sens, c'est le cas. Un élément positionné de manière absolue ne fait plus partie du cours normal de la mise en page. Il se trouve maintenant sur un plan qui lui est propre, séparé de tout le reste. C'est très utile : cela signifie que nous pouvons créer une fonctionnalité d'interface graphique isolée qui n'interfère pas avec la position des autres éléments sur la page. Par exemple, des boîtes d'informations contextuelles (*popup*), des menus de contrôle, des panneaux déroulants (*rollover panels*), des fonctionnalités d'interface utilisateur que l'on peut glisser et déposer n'importe où sur la page, et bien plus encore.

Ensuite, notez que la position de l'élément a changé. `top`, `bottom`, `left` et `right` se comportent différemment avec le positionnement absolu. Au lieu de positionner l'élément en fonction de sa position relative dans la mise en page du document, ils définissent la distance à laquelle l'élément doit se situer par rapport aux côtés de l'élément contenant. Dans ce cas, nous indiquons que l'élément à positionner de manière absolue doit se placer à 30px du haut et à 30px de la gauche de « l'élément conteneur » (il s'agit dans ce cas, l'élément conteneur est le bloc conteneur initial, voir la section ci-dessous pour plus d'informations).

Contextes de positionnement

Quel élément est « le conteneur » d'un élément positionné de manière absolue ? Cela dépend en grande partie de la propriété `position` des éléments qui sont les ancêtres de l'élément positionné (voir Identifier le bloc englobant).

Si aucun élément ancêtre ne voit sa propriété `position` explicitement définie, par défaut, tous les éléments ancêtres auront une position statique et par conséquent, l'élément positionné de façon absolue sera contenu dans **le bloc englobant initial**. Ce bloc englobant initial a les dimensions de la zone d'affichage (*viewport*) et est aussi le bloc qui contient l'élément `<html>`. Autrement dit, l'élément positionné de façon absolue sera affiché en dehors de l'élément `<html>` et positionné relativement à la zone d'affichage.

Dans la structure HTML, l'élément positionné est imbriqué dans l'élément `<body>`, mais pour la disposition finale, il est situé à 30px du bord haut et du bord gauche de la page. Vous pouvez modifier **le contexte de positionnement**, c'est-à-dire l'élément par rapport auquel l'élément est positionné de façon absolue. Pour cela, on définira le positionnement d'un des éléments ancêtres. Pour voir cet effet, ajoutez la déclaration suivante dans la règle ciblant `body` :

```
position: relative;
```

Exercice. Tester la feuille de style décrite ci-dessus, et comparez votre résultat à celui-ci : `4_positioning-context.html`

4 Positionnement fixe

Voyons maintenant le positionnement fixe. Cela fonctionne exactement de la même manière que le positionnement absolu, avec une différence essentielle : alors que le positionnement absolu fixe un élément en place par rapport à l'élément `<html>` ou son parent positionné le plus proche, le positionnement fixe fige un élément en place par rapport à la vue par la fenêtre du navigateur elle-même. Cela signifie que vous pouvez créer des éléments d'interface utilisateur utiles qui sont fixés en place, comme des menus de navigation persistants.

Voici un exemple simple pour montrer ce que nous voulons dire. D'abord, supprimez la règle `.positioned` de la CSS.

Maintenant, mettez à jour la règle `body`. Supprimez la déclaration `position: relative;` et ajoutez une hauteur fixe, ainsi :

```
body {
  width: 500px;
  height: 1400px;
  margin: 0 auto;
}
```

Maintenant, donnez la position `fixed` à l'élément `<h1>` et centrez-le en haut de la fenêtre. Ajoutez la règle suivante à la CSS :

```
h1 {
  position: fixed;
  top: 0;
  width: 500px;
  margin-top: 0;
  background: white;
  padding: 10px;
}
```

`top: 0;` est requis pour que l'élément soit collé au haut de l'écran. Ensuite, nous donnons au titre d'en-tête la même largeur que la colonne de contenu. Nous mettons ensuite un fond blanc et un peu de remplissage pour que le contenu ne soit pas visible sous lui.

Si vous enregistrez et actualisez maintenant, vous verrez un petit effet amusant par lequel le titre reste fixe et le contenu semble défiler vers le haut et disparaître en dessous. Mais nous pouvons l'améliorer davantage — actuellement, une partie du contenu commence sous l'en-tête. En effet, l'en-tête positionné n'apparaît plus dans le cours du document, de sorte que le reste du contenu se déplace vers le haut. Nous devons tout baisser un peu ; nous pouvons le faire en fixant une marge supérieure sur le premier paragraphe. Ajoutez ceci maintenant :

```
p:nth-of-type(1) {
  margin-top: 60px;
```

```
}
```

Exercice. Tester la feuille de style décrite ci-dessus, et comparez votre résultat à celui-ci : `6_fixed-positioning.html`

5 Positionnement adhérent (*sticky*)

Il existe une autre valeur de positionnement disponible : `position: sticky`. Elle est un peu plus récente que les autres. Il s'agit essentiellement d'un hybride entre position relative et position fixe : l'élément à positionner est en positionnement relatif jusqu'à un certain seuil (par exemple, 10px du haut de la fenêtre), seuil au-delà duquel il est en positionnement fixe.

Index déroulant

Une utilisation courante et pleine d'intérêt de `position: sticky` consiste à créer une page d'index déroulante dans laquelle les divers en-têtes restent collés en haut de la page une fois qu'ils l'ont atteint. Le balisage d'un exemple de ce type ressemble à ceci :

Body HTML

```
<h1>Positionnement adhérent</h1>

<dl>
    <dt>A</dt>
    <dd>Abeille</dd>
    <dd>Abricot</dd>
    <dd>Altimètre</dd>
    <dd>Avion</dd>
    <dt>B</dt>
    <dd>Banane</dd>
    <dd>Betterave</dd>
    <dd>Bœuf</dd>
    <dd>Bouvreuil</dd>
    <dd>Buzzard</dd>
    <dt>C</dt>
    <dd>Calculateur</dd>
    <dd>Camera</dd>
    <dd>Cane</dd>
    <dd>Chameau</dd>
    <dt>D</dt>
    <dd>Dime</dd>
    <dd>Dindon</dd>
    <dd>Drapeau</dd>
    <dd>Drone</dd>
    <dt>E</dt>
    <dd>Eau</dd>
    <dd>Éléphant</dd>
    <dd>Escadrille</dd>
</dl>
```

Le CSS pourrait ressembler à ce qui suit. Dans le flux normal, les éléments `<dt>` défilent avec le contenu. Quand on ajoute `position: sticky` à l'élément `<dt>` avec une valeur `top` de 0, les navigateurs prenant en charge ce positionnement colleront les titres au sommet de la vue de la fenêtre au fur et à mesure qu'ils atteignent cette position. Chaque en-tête suivant remplacera l'en-tête précédent au fur et à mesure que le contenu défile.

```css
dt {
  background-color: black;
  color: white;
  padding: 10px;
  position: sticky;
  top: 0;
  left: 0;
  margin: 1em 0;
}
```

Exercice. Tester la feuille de style décrite ci-dessus, et comparez votre résultat à celui-ci : 7_sticky-positioning.html