Les sélecteurs en CSS3

par debray jerome (Dji programmation web2 et design) (Blog)

Date de publication :

Dernière mise à jour :

Les sélecteurs permettent par le biais d'une "requête CSS" d'atteindre un ensemble de noeud dans un document HTML et de lui donner un style.

Ces "requêtes" sont des règles de reconnaissance de motifs qui déterminent les règles de style qui s'appliquent aux éléments du DOM.

En CSS3, il y a donc des nouveautés au niveau des sélecteurs afin d'atteindre des noeuds dans le DOM de manière encore plus précise.

I - Compatibilité

Chrome, Safari, Opera, Firefox 4, Internet Explorer 9

II - Rappel

Voici un tableau sur la syntaxe des sélecteurs en CSS2 (référence W3C traduit sur yoyodesign : **Les sélecteurs**) :

Motif	Signification
[illegible]	[illegible]

III - Les nouveautés CSS3

Les nouveautés incluent entre autres les **sélecteurs d'attributs**, le **combinateur d'adjacence directe**, les **pseudo-classes** et les **pseudo-éléments**.

IV - Les sélecteurs d'attributs

Il y a 3 nouveaux sélecteurs

[attr^="stringValue"]

Ce sélecteur permet de sélectionner un élément DOM dont l'attribut "attr" commence exactement par la valeur "stringValue".

Exemple :

```
p.example{
 margin:0;
 padding:10px;
 color:#000;
}
p.example[title^="ess"]{
 color:#fff;
 background:#333;
}
```

```
<p class="example"> je n'ai pas d'attribut title</p>
<p class="example" title="comment"> j'ai un attribut title mais il ne commence pas par "ess"</p>
<p class="example" title="essai"> j'ai un attribut title commençant par "ess"</p>
<p class="example" title="esson"> j'ai un attribut title commençant par "ess" également</p>
```

Démo

[attr$="stringValue"]

Ce sélecteur permet de sélectionner un élément DOM dont l'attribut "attr" finit exactement par la valeur "stringValue".

Exemple :

```
p.example2{
 margin:0;
 padding:10px;
 color:#000;
}
p.example2[title$="sai"]{
 color:#fff;
 background:#045FB4;
}
```

```
<p class="example2"> je n'ai pas d'attribut title</p>
<p class="example2" title="comment"> j'ai un attribut title mais il ne commence pas par "ess"</p>
<p class="example2" title="essai"> j'ai un attribut title commençant par "ess"</p>
<p class="example2" title="esson"> j'ai un attribut title commençant par "ess" également</p>
```

Démo

[attr*="stringValue"]

Ce sélecteur permet de sélectionner un élément DOM dont l'attribut "attr" comporte au moins une fois la valeur "stringValue".

Exemple :

```
p.example3{
 margin:0;
 padding:10px;
 color:#000;
}
p.example3[title*="val"]{
 color:#fff;
 background:#990000;
}
```

```
<p class="example3"> je n'ai pas d'attribut title</p>
<p class="example3" title="comment"> j'ai un attribut title mais il ne contient pas "val"</p>
<p class="example3" title="val"> j'ai un attribut title contenant au moins "val"</p>
<p class="example3" title="evaluer"> j'ai un attribut title contenant au moins "val" également</p>
<p class="example3" title="eval"> j'ai un attribut title contenant au moins "val" également</p>
```

Démo

V - Le combinateur d'adjacence directe

Permet d'ajouter un style à tous les éléments qui suivent un élément particulier.

Exemple :

```
.example4 div{
 margin:0;
 padding:10px;
 color:#000;
}
.example4 div~p{
 color:#fff;
 margin:20px;
 width:200px;
 padding:5px;
 border:1px solid #333;
 background:#006644;
}
```

```
<div class="example4">

 <div>je suis l'élément particulier div</div>
 <p> je suis un p qui suit le div (l'élement particulier)</p>
 <p>je suis un p qui suit le div (l'élement particulier)</p>
 <span>je suis un span</p>
 <p>je suis un p qui ne suit pas le div (l'élement particulier)</p>

</div>
```

Démo

VI - Les pseudo-classes

Plusieurs pseudo-classes ont été ajouté :

:root

Ce sélecteur représente un élément qui est la racine d'un document. Par exemple, en HTML 4, l'élément est *html*.

:nth-child(*expression*)

Ce sélecteur permet de cibler tous les éléments en se basant sur leur position dans la liste des enfants de leur parent.

expression peut être un nombre, une expression numérique ou un mot clé tel que "odd" ou "even".

Exemple :

```
.exampleTable{
 width:100%;
 border:1px solid #444;
}
.exampleTable tr:nth-child(even){ //tous les enfants aux numéros pairs
 background:#999999;
 text-shadow: 2px 2px 5px #111;
 color:#fff;
}
.exampleTable tr:nth-child(odd){ //tous les enfants aux numéros impairs
 background:#990000;
 color:#fff;
}
.exampleTable tr:nth-child(3n){ //tous les 3 enfants
 background:#045FB4;
 color:#fff;
}
.exampleTable tr:nth-child(7){ //l'enfant numéro 7
 background:#006400;
 text-shadow: 2px 2px 2px #fff;
 color:#000;
}
```

```
<table class="exampleTable">
 <tr>
  <td>1ere ligne</td>
 </tr>
 <tr>
  <td>2eme ligne</td>
 </tr>
 <tr>
  <td>3eme ligne</td>
 </tr>
 <tr>
  <td>4eme ligne</td>
 </tr>
 <tr>
  <td>5ere ligne</td>
 </tr>
 <tr>
  <td>6eme ligne</td>
 </tr>
 <tr>
  <td>7eme ligne</td>
 </tr>
 <tr>
  <td>8eme ligne</td>
 </tr>
</table>
```

Démo

:nth-last-child(*expression*)

Ce sélecteur accepte les mêmes arguments que :nth-child() et correspond au dernier enfant d'un élément parent.

C'est le même principe que le :nth-child() sauf que l'on part de la fin.

Exemple :

```
.exampleTable2{
 width:100%;
 border:1px solid #444;
}
.exampleTable2 tr:nth-last-child(odd){ /*tous les enfants aux numéros impairs depuis la fin.*/
 background:#990000;
 color:#fff;
}
.exampleTable2 tr:nth-last-child(-n+2){ /*les 2 derniers enfants.*/
 background:#045FB4;
 color:#fff;
}
.exampleTable2 tr:nth-last-child(7){ /
*l'enfant numéro 7 en partant de la fin donc la 2ème ligne du tableau.*/
 background:#006400;
 text-shadow: 2px 2px 2px #fff;
 color:#000;
}
```

```
<table class="exampleTable2">
 <tr>
  <td>1ere ligne</td>
 </tr>
 <tr>
  <td>2eme ligne</td>
 </tr>
 <tr>
  <td>3eme ligne</td>
 </tr>
 <tr>
  <td>4eme ligne</td>
 </tr>
 <tr>
  <td>5ere ligne</td>
 </tr>
 <tr>
  <td>6eme ligne</td>
 </tr>
 <tr>
  <td>7eme ligne</td>
 </tr>
 <tr>
  <td>8eme ligne</td>
 </tr>
</table>
```

Démo

:last-child

Correspond à ***:nth-last-child(1)*** .

:first-child

Correspond à ***:nth-child(1)*** .

:nth-of-type(*expression*)

Ce sélecteur représente un élément qui a *expression* frères du même type **devant** lui dans l'arbre DOM.

```
img:nth-of-type(2n+1) { float: right; }
img:nth-of-type(2n) { float: left; }
```

alternance de la position des images en flottant.

:nth-last-of-type(*expression*)

Ce sélecteur représente un élément qui a *expression - 1* frères du même type après lui dans l'arbre DOM.

```
body > h2:nth-of-type(n+2):nth-last-of-type(n+2)
```

représente tous les h2 fils d'un élément XHTML body sauf le premier et le dernier.

:first-of-type

Correspond à :nth-of-type(1). :first-of-type représente un élément qui est le **premier** enfant de son type dans la liste des enfants de son élément parent.

:last-of-type

Correspond à :nth-last-of-type(1). Représente un élément qui est le **dernier** enfant de son type dans la liste des enfants de son élément parent.

:only-child

Correspond à un élément qui n'a **aucun frère**. Cette pseudo-classe correspond à ***:first-of-type:last-of-type*** ou ***:nth-of-type(1):nth-last-of-type(1)*** .

:checked

Correspond aux éléments cochés d'un formulaire.

:contains(*value*)

Correspond aux éléments dont le contenu textuel contient la sous-chaîne donnée en argument.

Exemple :

```
p:contains('essai'){
 background:#900;
}
```

signifie que tous les éléments "p" contenant la sous chaine "essai" auront pour couleur d'arrière plan, la valeur '#900'.

L'usage de la pseudo-classe de contenu (:contain) est restreint aux médias statiques.

:empty

Correspond aux éléments n'ayant pas d'enfant.

:not([*expression*])

Représente un élement qui n'est pas représenté par l'expression donnée en argument.

Exemple:

```
button:not([DISABLED]){
 ...
}
a:not(:visited){
 ...
}
```

:target

Représente un élément qui est la cible de l'URI.

Exemple :

```
p:target { background:#900}
```

Tout élement p qui sera la cible de l'URI (via l'ID # en tant que ancre) aura pour couleur d'arrière plan la valeur "#900".

VII - Les pseudo-élements

::first-line

Applique la règle de style à la premiere ligne du texte de l'élément.

```
p::first-line { text-transform: uppercase }
```

La 1ère ligne de ou des éléments "p" est mise en majuscule.

::first-letter

Applique la règle de style à la première lettre du texte de l'élément.

```
p::first-letter { font-size: 2em }
```

La 1ère lettre de ou des éléments "p" a une taille de police de 2em.

::selection

Applique la règle de style à la sélection du texte de l'élément faite par l'utilisateur.

```
p::selection { background:#006644 }
```

A la sélection, le texte sélectionné aura une couleur d'arrière plan de valeur '#006644'.

::before et ::after

Génère un contenu avant ou après un contenu d'un élément.

Ces pseudo-elements existent en CSS2.1 sous forme **:before** et **:after**.

VIII - Remerciements

Merci à **NarcissX** pour sa relecture.