Franck CELLIER

Formateur en Informatique

Technicien Qualité d'Organisme de Formation

http://cellierfranck.alwaysdata.net

MS LANGAGE & DEVELOPPEMENT POUR LE WEB

LA PROPRIÉTÉ CONTENT

La propriété **content** est utilisée avec les pseudo-éléments ::before et ::after afin de générer le contenu d'un élément. Les objets insérés via la propriété content sont des *éléments remplacés anonymes*.

CSS.

```
/* Des mots-clés qui ne peuvent pas être mélangés
  avec d'autres valeurs */
content: normal;
content: none;

/* Une valeur <string>, les caractères non-latin */
/* doivent être échappées par ex. \000A9 for &copy; */
content: 'prefix';

/* Valeurs décrivant une URI */
content: url(http://www.exemple.com/test.png);

/* Valeurs utilisant un compteur */
content: counter(compteur_chapitre);
content: counter(compteur_section, ".");

/* attr() lie à la valeur de l'attribut HTML */
content: attr(value string);

/* Mots-clés dépendant de langue */
/* ou de la position            */
content: open-quote;
content: close-quote;
content: no-open-quote;
content: no-close-quote;

/* Sauf avec normal et none, on peut utiliser */
/* plusieurs valeurs de façon simultanée */
content: open-quote chapter_counter;

/* Valeurs globales */
content: inherit;
content: initial;
content: unset;
```


2 rue Gabriel FAURÉ - 59150 WATTRELOS - fcellier34@gmail.com - ☎ 06 41 12 70 19 - www.linkedin.com/in/formateur-franck-cellier

Valeur initiale	normal
Applicabilité	pseudo-éléments ::before et ::after
Héritée	non
Média	tous
Valeur calculée	Sur les éléments, le résultat du calcul est toujours normal. Sur ::before et ::after, si normal est spécifié, cela donnera none. Sinon, pour les valeurs d'URI, on aura l'URI absolue ; pour les valeurs attr(), on aura la chaine résultante ; pour les autres mots-clé, ce sera comme spécifié.
Type d'animation	discrète
Ordre canonique	l'ordre unique et non-ambigu défini par la grammaire formelle

Syntaxe

Valeurs

none

Le pseudo-élément n'est pas généré.

normal

Résulte en none pour les pseudo-éléments :before et :after.

<string>

Contenu sous forme de texte. Les caractères non-latins doivent être encodés avec leur séquence d'échappement Unicode (\000A9 représentera par exemple le symbole ©).

<url>

La valeur est l'URL qui désigne une source extérieure (comme une image). Si la ressource ou l'image ne peut être affichée, soit elle est ignorée, soit un texte de remplissage prend la place.

counter() ou counters() (cf. <counter>)

Un compteur CSS, généralement un nombre, qui peut être affiché grâce à la fonction counter() ou counters().

La première possède deux formes : 'counter(*name*)' ou 'counter(*name*, *style*)'. Le texte généré est la valeur du compteur le plus profond possédant un nom donné dans ce pseudo-élément ; il est formaté selon le style indiqué (decimal par défaut).

La seconde a également deux formes : counters(*name*, *string*) ou counters(*name*, *string*, *style*). Le texte généré est la valeur de tous les compteurs d'un nom donné dans ce pseudo-élément, depuis le moins profond jusqu'au plus profond séparés par la chaîne définie. Les compteurs sont formatés selon le style indiqué (decimal par défaut). Voir la section sur les compteurs automatiques et sur la numérotation pour plus d'informations.

attr(X)

Renvoie la valeur de l'attribut X de l'élément comme une chaîne. S'il n'existe pas d'attribut X, une chaîne vide est renvoyée. La sensibilité à la casse du nom de l'attribut dépend du langage utilisé.

open-quote | close-quote

Ces valeurs sont remplacées par la chaîne appropriée de la propriété quotes.

no-open-quote | no-close-quote

N'introduit aucun contenu, mais incrémente (respectivement décrémente) le niveau d'imbrication des citations.

Syntaxe formelle

normal | none | [<content-replacement> | <content-list>] [/ <string>]?

où
<content-replacement> = <image>
<content-list> = [<string> | contents | <image> | <quote> | <target> | <leader()>]+

où
<image> = <url> | <image()> | <image-set()> | <element()> | <cross-fade()> | <gradient>
<quote> = open-quote | close-quote | no-open-quote | no-close-quote
<target> = <target-counter()> | <target-counters()> | <target-text()>
<leader()> = leader(<leader-type>)

où
<image()> = image([[<image> | <string>]? , <color>?]!)
<image-set()> = image-set(<image-set-option>#)
<element()> = element(<id-selector>)
<cross-fade()> = cross-fade(<cf-mixing-image> , <cf-final-image>?)
<gradient> = <linear-gradient()> | <repeating-linear-gradient()> | <radial-gradient()> | <repeating-radial-gradient()>
<target-counter()> = target-counter([<string> | <url>] , <custom-ident> , <counter-style>?)
<target-counters()> = target-counters([<string> | <url>] , <custom-ident> , <string> , <counter-style>?)
<target-text()> = target-text([<string> | <url>] , [content | before | after | first-letter]?)
<leader-type> = dotted | solid | space | <string>

où
<color> = <rgb()> | <rgba()> | <hsl()> | <hsla()> | <hex-color> | <named-color> | currentcolor | <deprecated-system-color>
<image-set-option> = [<image> | <string>] <resolution>
<id-selector> = <hash-token>
<cf-mixing-image> = <percentage>? && <image>
<cf-final-image> = <image> | <color>
<linear-gradient()> = linear-gradient([<angle> | to <side-or-corner>]? , <color-stop-list>)
<repeating-linear-gradient()> = repeating-linear-gradient([<angle> | to <side-or-corner>]? , <color-stop-list>)
<radial-gradient()> = radial-gradient([<ending-shape> || <size>]? [at <position>]? , <color-stop-list>)
<repeating-radial-gradient()> = repeating-radial-gradient([<ending-shape> || <size>]? [at <position>]? , <color-stop-list>)
<counter-style> = <counter-style-name> | symbols()

où
<rgb()> = rgb(<percentage>{3} [/ <alpha-value>]?) | rgb(<number>{3} [/ <alpha-value>]?) | rgb(<percentage>#{3} , <alpha-value>?) | rgb(<number>#{3} , <alpha-value>?)
<rgba()> = rgba(<percentage>{3} [/ <alpha-value>]?) | rgba(<number>{3} [/ <alpha-value>]?) | rgba(<percentage>#{3} , <alpha-value>?) | rgba(<number>#{3} , <alpha-value>?)
<hsl()> = hsl(<hue> <percentage> <percentage> [/ <alpha-value>]?) | hsl(<hue>, <percentage>, <percentage>, <alpha-value>?)
<hsla()> = hsla(<hue> <percentage> <percentage> [/ <alpha-value>]?) | hsla(<hue>, <percentage>, <percentage>, <alpha-value>?)
<side-or-corner> = [left | right] || [top | bottom]

<color-stop-list> = <color-stop>#{2,}
<ending-shape> = circle | ellipse
<size> = closest-side | farthest-side | closest-corner | farthest-corner | <length> | <length-percentage>{2}
<position> = [[left | center | right] || [top | center | bottom] | [left | center | right | <length-percentage>] [top | center | bottom | <length-percentage>]? | [[left | right] <length-percentage>] && [[top | bottom] <length-percentage>]]
<counter-style-name> = <custom-ident>

où
<alpha-value> = <number> | <percentage>
<hue> = <number> | <angle>
<color-stop> = <color> <length-percentage>?
<length-percentage> = <length> | <percentage>

Exemples

Titres et citations

HTML

```
<h1>5</h1>
<p> Commençons par une citation de Sir Tim Berners-Lee,
   <q cite="http://www.w3.org/People/Berners-Lee/FAQ.html#Internet">
      I was lucky enough to invent the Web at the time when the Internet already existed - and had for a decade and a half.</q>  We must understand that there is nothing fundamentally wrong with building on the contributions of others.
</p>

<h1>6</h1>
<p> Citons le manifeste Mozilla
   <q cite="http://www.mozilla.org/about/manifesto/">
      Internet est une ressource publique mondiale qui doit demeurer ouverte et accessible.
</p>
```

CSS

```
q {
   color: #00008B;
   font-style: italic;
}

q::before   { content: open-quote }
q::after    { content: close-quote }

h1::before  { content: "Chapitre "; }
```

Ajouter une icône avant un lien

HTML

```
<a href="http://www.mozilla.org/fr/">Accueil</a>
```

CSS

```
a::before{
   content: url(https://mozorg.cdn.mozilla.net/media/img/favicon.ico) " MOZILLA: ";
   font:    x-small Arial,freeSans,sans-serif;
   color:   gray;
}
```

Résultat

Utiliser les classes

HTML

```
<h2>Top des ventes</h2>
<ol>
   <li>Thriller politique</li>
   <li class="nouveaute">Histoires effrayantes</li>
   <li>Ma biographie</li>
   <li class="nouveaute">Bit-lit</li>
</ol>
```

CSS

```
.nouveaute::after {
   content: " Nouveau !";
   color: red;
}
```

Remplacer un élément

Dans cet exemple, on remplace le contenu d'un élément avec une image. Il est possible de remplacer le contenu d'un élément avec une valeur de type <url> ou <image>. Le contenu ajouté avec ::before ou avec ::after ne sera plus généré car l'élément sera devenu un élément remplacé.

HTML

```
<div id="replaced">Mozilla</div>
```

CSS

```
#replaced {
  content: url("https://mdn.mozillademos.org/files/12668/MDN.svg");
}

#replaced::after { /* Ceci ne sera pas affiché, */
                   /* l'élément sera un élément remplacé */
  content: " (" attr(id) ")";
}
```