



CSS3 – I selettori

DARIO DE JUDICIBUS

Sono molti le aree su cui si sta lavorando nel terzo livello del CSS. Una delle più importanti riguarda le modalità per selezionare gli elementi ai quali applicare un certo stile. Il nuovo livello permetterà molta più flessibilità e possibilità dei precedenti.

Qualcuno forse si chiederà che senso abbia lavorare su un terzo livello dei *Cascading Style Sheets* quando il livello 2.1 è stato da poco consolidato e la maggior parte dei browser ancora hanno problemi nell'implementazione del livello 2 e in qualche caso anche del livello 1. In effetti uno degli obiettivi dei gruppi che stanno definendo il CSS3 è proprio quello di affrontare il problema della conformità. In pratica il problema si può ricondurre a due componenti: la prima è che, dato un livello, ci vuole un certo tempo perché i produttori di *User Agents* (UA) rendano le loro applicazioni conformi; il secondo è che comunque ci vorrà diverso tempo prima che gli utenti finali passino alle versioni più aggiornate dei browser. Entrambi questi problemi scoraggiano di fatto gli sviluppatori di documenti HTML ed XML dall'adottare le nuove specifiche nel loro codice, in quanto l'obiettivo primario di chi fa un sito è quello di renderlo visibile al maggior numero possibile di visitatori.

Per affrontare almeno in parte il problema, il W3C ha deciso di rendere modulare la terza versione dei CSS. In questo modo i vari gruppi potranno procedere separatamente e i produttori di browser potranno implementare i moduli già pronti senza aspettare che l'intero standard sia stato definito. In pratica alcuni moduli potranno entrare nello stato di *raccomandazione* prima di altri. Ovviamente questo permetterà anche ad ogni UA di essere conforme solo a determinati moduli in funzione del contesto in cui andrà ad operare.

Selettori e dichiarative

Una delle aree sulle quali si sta maggiormente lavorando sono i selettori. Per selettore intendiamo quella parte del blocco di istruzioni che identifica a quale elemento va applicato un determinato stile. Già con la versione 2.1, diventata raccomandazione solo all'inizio del 2005, la sintassi dei selettori si era arricchita di un certo numero di possibilità.

Con il CSS3 le possibilità offerte allo sviluppatore per identificare con precisione un elemento nel codice a fronte di tutta una serie di criteri si è notevolmente ampliata.

In pratica, come sanno molto bene gli sviluppatori di codice HTML e XML, un foglio di stile è una sequenza di stili, ognuno formato da un selettore e da una dichiarativa. Per esempio, in

```
p { text-align: justify; color: black; text-indent: 1em; }
```

il selettore è **p** e corrisponde all'elemento **<p>**, ovvero al paragrafo semplice, mentre lo stile corrispondente è riportato fra parentesi graffe.

CSS1

Nel primo livello del CSS, ovvero il CSS1, esiste un numero limitato di criteri per selezionare un elemento. Il più semplice è il marcatore dell'elemento stesso, come nell'esempio precedente. È anche possibile selezionare tutti gli elementi appartenenti a una certa classe o un elemento che abbia un determinato identificativo:

```
p.myclass { text-align: center; margin: 0.5em 5em 0.5em 5em; }  
p#myid { color: red; }
```

Inoltre è anche possibile selezionare un elemento solo se si trova all'interno di un altro elemento. Questo meccanismo di inclusione può utilizzare l'asterisco a indicare un elemento generico, per cui a un elemento **em** all'interno di un paragrafo **p** corrisponde il selettore **p em**, mentre se l'inclusione prevede un livello ulteriore di annidamento avremo **p * em** dove l'asterisco rappresenta appunto un elemento qualunque. Utilizzando una virgola è inoltre possibile associare a una stessa dichiarativa più elementi e la stessa dichiarativa può essere spezzata in più blocchi. Ad esempio:

```
h1, h2, h3 { font-family: Verdana; }  
h1 { font-size: 1.8em; }  
h2 { font-size: 1.4em; }  
h3 { font-size: 1.0em; }
```

Per finire CSS1 prevede alcune pseudoclassi e pseudoelementi che permettono di identificare un elemento quando si trova in un determinato stato o parte di un elemento. Una di queste pseudoclassi è **:hover**, che si applica solo all'elemento **a** e che permette di cambiarne lo stile quando il puntatore si trova sopra a un collegamento, mentre uno pseudoelemento del CSS1 è **:first-letter**, ovvero la prima lettera del testo contenuto all'interno dell'elemento. Vedremo che il CSS3 fa un uso notevole delle pseudoclassi e degli pseudoelementi differenziando quest'ultimi con l'utilizzo di una coppia di "due punti" come separatore. In CSS3, quindi, la prima lettera di un paragrafo va identificata con il selettore

```
p::first-letter { color: navy; font-weight: bold; }
```

CSS2

Nel CSS2 il sistema dei selettori si è ulteriormente arricchito, anche se purtroppo ben pochi di questi meccanismi di selezione sono attualmente utilizzati perché non tutti i browser li supportano.

L'asterisco è diventato un selettore universale, per cui `* li`, ad esempio, permette di selezionare tutti i marcatori `<li>` qualunque sia il tipo di lista nel quale compaiono. Inoltre compaiono per la prima volta gli attributi. C'è da dire che IE ha qualche problema con l'asterisco, dato che selettori come `* html` e `* * body` che, a rigor di logica non dovrebbero selezionare alcun elemento, vengono interpretati dal browser Microsoft rispettivamente come `html` e `* body`.

CSS2.1 dà anche la possibilità di selezionare solo quegli elementi in cui compare uno specifico attributo o nel quale un certo attributo abbia un determinato valore. In quest'ultimo caso è anche possibile specificare quando una certa stringa compare all'interno di una lista separata da spazi o da trattini. Quest'ultimo caso è molto utile se l'attributo ha come valori i codici lingua definiti dal RFC 1766 (ad esempio `it-ch` o `fr-be`).

```
/*
** Dichiarativa per tutti i paragrafi con allineamento specificato
*/
p[align] { ... }

/*
** Dichiarativa per tutti i paragrafi allineati a destra
*/
p[align="right"] { ... }

/*
** Dichiarativa per tutti i collegamenti che puntano a risorse in italiano
*/
a[hreflang="it"] { ... }

/*
** Dichiarativa per tutti i collegamenti relativi anche
** all'autore del documento
*/
a[rel~="author"] { ... }
```

Da notare che questi meccanismi evitano di creare inutili classi quando di fatto esse corrispondono a una qualche logica di selezione basata su determinati attributi di un elemento.

Un altro criterio permette di selezionare un elemento adiacente a un altro utilizzando il simbolo “più”. Ad esempio, per selezionare un paragrafo se e solo se viene subito dopo un’intestazione di primo livello, basterà scrivere

```
h1 + p { ... }
```

Il CSS2 ha ampliato anche la lista degli pseudoelementi e delle pseudoclassi. Ad esempio, per aggiungere la scritta “Nota: “ davanti a un paragrafo, basta scrivere

```
p#nota::before { content: "Nota: "; }
```

Infine, per i documenti scritti in più lingue è molto utile la pseudoclasse **:lang**. Ad esempio,

```
:lang(it) > p { ... }  
:lang(en) > p { ... }
```

permette di dare stili differenti ai paragrafi in italiano e a quelli in inglese all’interno di un testo.

CSS3

E veniamo finalmente al CSS3. Attualmente l’unico *browser* che sta iniziando a implementare quei selettori CSS3 che ragionevolmente diventeranno raccomandazioni è quello relativo al Progetto Mozilla. La prima area che è stata arricchita è quella dei selettori basati sugli attributi. Nel CSS3 è possibile selezionare gli elementi in base anche a una sottostringa del valore di un attributo. Ad esempio:

```
/* Seleziona i collegamenti ai sottodomini di tipo w3 */  
a[href^=http://w3.] { ... }  
  
/* Seleziona i collegamenti ai domini italiani */  
a[href$=".it"] { ... }  
  
/* Seleziona le immagini nella sottocartella foto */  
img[src*="foto/"] { ... }
```

Una seconda area riguarda la possibilità di selezionare un elemento in base alla sua posizione nell’albero degli elementi. Ad esempio, la radice di un documento XML è **:root** mentre un elemento figlio di un altro elemento viene rappresentato usando il simbolo **>**. Ad esempio

```
/* Seleziona l’ultimo paragrafo di una citazione */  
blockquote#discorso > p:last-child { ... }
```

Ricordo la distinzione fra *discendente* e *figlio*: un elemento B è discendente da A se B è figlio di A oppure se B è figlio di un discendente di A. Un figlio, invece, è un elemento contenuto direttamente in un altro, detto quindi *genitore*. Possiamo quindi affermare che

```
A B = ( A > B ) U ( A * B )
```

Ad esempio `body > p` selezionerà tutti i paragrafi di primo livello, cioè inclusi direttamente nel corpo del documento, mentre `body * p` selezionerà tutti gli altri.

Un'altra regola sintattica permette di selezionare un elemento quando viene dopo un altro ma non è necessariamente adiacente al primo. In questo caso si usa la tilde. Ad esempio, il primo paragrafo dopo un'intestazione di primo livello può essere selezionata con

```
h1 ~ p { ... }
```

anche se tra i due c'è un'intestazione di secondo livello, ovvero un `h2`.

Molto utile è anche la possibilità di selezionare il complemento a un certo criterio, ovvero di usare la pseudoclasse `:not`. Ad esempio

```
p:not[align="left"] { ... }
```

Permette di specificare lo stile di tutti i paragrafi non allineati a sinistra.

Nel CSS3 si sta dando inoltre una particolare attenzione a quegli elementi che permettono di creare moduli interattivi, ovvero quelli che si mettono all'interno dell'elemento `form`. Vedremo in un prossimo articolo tutte le novità a riguardo, soprattutto per quello che riguarda il nuovo concetto di *apparenza*. Per quanto riguarda i selettori, il CSS3 mette a disposizione un certo numero di nuove pseudoclassi per gestire i vari stati nei quali i campi e controlli di un modulo si possono venire a trovare. Ad esempio, è verosimile pensare che si voglia rendere con uno stile differente un pulsante a bottone (*radio button*) a seconda che sia abilitato o meno. Per far questo possiamo usare le seguenti pseudoclassi, ovviamente applicabili anche ad altri controlli:

```
radio.gender:enabled { ... }  
radio.gender:disabled { ... }
```

Se invece abbiamo un casella di spunta (*check box*) potremmo voler gestire il testo a lato in modo differente a seconda che sia smarcata o meno. In questo caso scriveremo

```
input:checked { ... }
```

Un discorso analogo possiamo farlo per i controlli che si trovino in uno stato indeterminato (`:indeterminant`).

Altre pseudoclassi ci danno maggiori opportunità di selezionare un elemento in base al suo posizionamento nell'albero degli elementi. Ad esempio `:empty` permette di selezionare elementi senza contenuto. In particolare ci si è focalizzati su due categorie di pseudoclassi: quelle basate sulla posizione e quelle basate sul tipo. Facciamo un esempio pratico. Se voglio dare un aspetto a tutte le righe pari di una tabella ed un altro a tutte quelle dispari scriverò

```
table#striped tr:nth-child(2n+1) { background-color: white; }
table#striped tr:nth-child(2n) { background-color: silver; }
```

Se invece voglio che le immagini in un certo documento siano posizionate alternativamente a destra e a sinistra, scriverò

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

In pratica, dove ho `child` ho una relazione diretta genitore-figlio, dove invece ho `type` ho semplicemente un certo tipo di elemento indipendentemente dalla sua posizione nell'albero. A questo punto è evidente il significato di

```
h2:only-of-type { ... }
h2:only-child { ... }
```

La prima regola stabilisce lo stile se e solo se nel documento c'è in assoluto un solo elemento `h2`, mentre la seconda stabilisce lo stile di un elemento `h2` se e solo se è l'unico all'interno di un elemento contenitore, ovvero non ci sono altri `h2` ad avere lo stesso genitore. Ovviamente, se c'è un solo elemento `h2`, la seconda regola è equivalente alla prima. Tuttavia, se in un documento ci sono due `h2`, uno direttamente nel `body` e uno all'interno di una `div`, la prima regola non agisce mentre la seconda si applica a entrambe le intestazioni.

Questo tipo di selettori sono quindi molto potenti e permettono di costruire schemi basati su modelli ripetitivi di stili. Da notare che il parametro all'interno delle parentesi tonde deve sempre essere nullo o positivo, e che il valore può essere parametrizzato con `n` oppure essere fisso. Ad esempio `(3n+1)` permette di selezionare il primo, il quarto, il settimo elemento e così via, mentre `(-n+3)` permette di selezionare solo il terzo, il secondo e il primo. Un valore nullo non seleziona alcun elemento. Un valore fisso, come `(6)`, seleziona un solo elemento, in questo caso il sesto. La conta può iniziare dall'alto, come in `:nth-of-type`, o dal basso, come in `:nth-last-of-type`. Ad esempio

```
p { text-align: left; }
p:nth-last-of-type(-n+2) { text-align: right; }
```

allineerà a destra solo gli ultimi due paragrafi di un documento. Tutti gli altri saranno allineati a sinistra.

Il CSS3 ha anche aggiunto a `:first-child` il corrispettivo dal fondo, ovvero `:last-child`, e ha previsto anche le corrispondenti pseudoclassi per i tipi, ovvero `:first-of-type` e `:last-of-type`.

Una pseudoclasse a sé è invece `:target`, che rappresenta tutti quegli elementi che sono il possibile bersaglio di un collegamento rivolto all'interno dello stesso documento. In pratica serve a identificare quelli che in gergo sono chiamati *frammenti* e che negli URL seguono il simbolo `#`. Ad esempio, se in un documento ho

```
...come si può vedere nel <a href="#report">rapporto</a> dell'Associazione
Pro Tapiri Orfani.</p>
...
...
<table id="report">
<tr>
...

```

allora è possibile dare un certo aspetto alla tabella referenziata utilizzando la regola

```
table:target { ... }
```

Ovviamente avrei potuto anche specificare la stessa tabella usando `table#report`, ma in questo caso la regola sarebbe stata applicata indipendentemente dal fatto che la tabella fosse agganciabile da un collegamento interno. Inoltre, usando `:target`, posso dare a tutte le tabelle referenziate internamente lo stesso aspetto o comunque una stessa caratteristica, che abbiano o meno un `id`. È evidente che questo discorso vale solo per i frammenti, dato che non c'è modo di sapere se un elemento è referenziato da un altro documento, a meno che il documento non sia stato caricato perché era stato richiesto il frammento in questione.

Un'altra pseudoclasse molto utile è quella che permette di selezionare un elemento se questo contiene una certa stringa di testo ma purtroppo funziona solo per periferiche di visualizzazione statiche come le stampanti (`print`), non gli schermi (`screen`). Ad esempio, per mostrare in rosso tutti i paragrafi dove è presente la parola "sesso" basta scrivere

```
p:contains("sesso") { color: red; }
```

Il CSS3 propone anche uno pseudoelemento che corrisponde al testo eventualmente selezionato dall'utente con il cursore o il puntatore. In genere questo testo viene reso al negativo rispetto al testo normale. Utilizzando `::selection` è possibile cambiare il comportamento normale del browser. Ovviamente si tratta di uno pseudoelemento e non di una pseudoclasse perché costruisce dinamicamente un elemento non codificato dallo

sviluppatore in base a quali caratteri sono stati effettivamente selezionati, sia che siano solo una parte di un paragrafo, sia che prendano due o più paragrafi.

Alcuni esempi più complessi

Per concludere vediamo alcune combinazioni avanzate che mostrano quanto sia diventato flessibile il meccanismo dei selettori nel CSS3.

Ad esempio,

```
h2 + blockquote > p:not(:first-of-type):not(:last-of-type)
```

rappresenta tutti i paragrafi all'interno di una citazione posta subito dopo un'intestazione di secondo livello, ad eccezione del primo e dell'ultimo.

```
td.high.side
```

rappresenta una cella di una tabella alla quale sia stata associata una lista di classi fra le quali high e side, ad esempio

```
<td class="high full side" nowrap>Testo</td>
```

Se un certo collegamento ha contemporaneamente il fuoco e l'utente vi posiziona sopra il puntatore, allora assumerà lo stile corrispondente al selettore

```
a:focus:hover
```

Per cambiare il tipo di simbolo utilizzato in una lista non ordinata quando la voce in questione è referenziata da un qualche collegamento all'interno dello stesso documento, basterà scrivere:

```
ul > li { list-style-type: square; }  
ul > li:target { list-style-type: disc; }
```

mentre per posizionare davanti a un paragrafo, anch'esso referenziato, una qualche immagine, si dovrà usare

```
p:target::before { content: url(images/disc.png); }
```

Infine, se si vuole dare all'ultima voce di una lista ordinata compatta uno stile differente, si dovrà scrivere qualcosa del tipo:

```
ol[compact] > li:last-child { ... }
```

Questo è quanto. Naturalmente il discorso non si esaurisce qui. Ci sono già molte altre proposte di arricchimento della sintassi per i selettori. Una proposta portata avanti dal

sottoscritto e da altri sviluppatori indipendentemente è quella di definire un selettore **B < A** che identifichi tutti gli elementi A che siano genitori di un elemento B. Ad esempio **strong < p** selezionerebbe tutti i paragrafi contenenti almeno un pezzo di testo in grassetto. Vedremo se sarà accettata. Ci aspettano comunque sicuramente molte altre novità, nella speranza che questa volta **tutti** i browser implementino gli standard così come sono stati definiti dal W3C.