



Le nuove tabelle: XHTML 2

DARIO DE JUDICIBUS

Usate per anni dagli sviluppatori in modo da supplire alle carenze dell'HTML per quello che riguardava la definizione della struttura di una pagina web, rese poi obsolete in tale ruolo dalle tecniche basate sulle divisioni e i fogli di stile, le tabelle sono state riscoperte dalle ultime versioni dell'XHTML e del CSS riprendendo il loro significato originale.

Per molti anni le tabelle HTML sono state una mano santa per molti sviluppatori. Grazie alla loro versatilità e facilità di utilizzo, esse permettevano di dare a una pagina web strutture anche molto articolate senza ricorrere alle *frame*, il cui controllo era ed è tuttora molto limitato, soprattutto per quello che riguarda l'ordine di caricamento delle sottopagine e la gestione del tasto *Back* del browser. Il prezzo da pagare, tuttavia, era una minore leggibilità del codice e di conseguenza una maggiore difficoltà di manutenzione dello stesso.

Negli ultimi anni il loro utilizzo in tal senso era stato osteggiato dai sostenitori della netta separazione fra struttura e stile a favore dell'utilizzo delle divisioni e dei fogli di stile. Quest'ultima tecnica, sebbene concettualmente più pulita, tuttavia, per diverso tempo non ha fatto altro che spostare la complicazione dal codice HTML a quello CSS. Il primo è infatti diventato più semplice, chiaro, mentre il secondo si è complicato parecchio a causa delle varie interpretazioni degli standard W3C da parte dei disegnatori dei vari browser. Solo recentemente, con le ultime versioni dei browser, c'è la speranza di poter fare a meno dei tanti trucchetti che tali incompatibilità ci hanno costretti a usare nei fogli di stile, mantenendo al contempo nelle pagine HTML quella semplicità che è stata raggiunta grazie alle divisioni.

Qualcuno potrà pensare che questo abbia decretato la morte delle tabelle, ma non è così. Le tabelle, come tali, sono delle componenti fondamentali di un documento HTML, perché comunque rappresentano una struttura molto comune in uno scritto, specialmente se di natura tecnica o finanziaria, per cui la loro utilità non è mai stata messa in discussione. Semmai si è tornati a utilizzarle nella loro accezione originale, ovvero come un elemento

capace di rappresentare tabulati e dati in cui l'incolonnamento e l'allineamento hanno un'importanza primaria.

Ecco quindi che sia la versione 2 dell'XHTML che la 3 del CSS dedicano il giusto spazio a questo elemento correggendone difetti e limitazioni passate e reinterpretandolo nella giusta maniera, grazie anche alla definizione di altri attributi ed elementi, come le colonne nelle divisioni ad esempio, che rendono definitivamente inutile l'utilizzo di tabelle per altri scopi se non quelli previsti originariamente.

In questo articolo vedremo come l'XHTML 2 supporta le tabelle, mentre nel prossimo vedremo le tabelle dal punto di vista del CSS 3.

Le tabelle nell'XHTML 2

Analizziamo dunque come le tabelle si sono evolute nell'XHTML 2. Assumiamo, per semplicità, un minimo di conoscenza dell'elemento `table` dell'HTML 4 e dei suoi sottoelementi.

Come si può vedere nella *tabella A* in fondo all'articolo, l'elemento `table` può contenere uno o nessun elemento `caption`, uno o nessuno elemento `summary`, nessuno o più elementi `col` piuttosto che `colgroup`, almeno un elemento `tbody`, un elemento `tr` ed eventualmente uno o nessuno elemento `thead` e uno o nessuno elemento `tfoot`.

La maggior parte di questi elementi a loro volta possono avere diversi attributi. Ovviamente ognuno di essi può avere uno dei tanti attributi cosiddetti *comuni*, ovvero tipici della maggior parte degli elementi XHTML. Gli elementi comuni sono quelli base, ovvero `id`, `class`, `title` e `layout`, quelli per la localizzazione dell'elemento, le metainformazioni, i ruoli, gli eventi e via dicendo. Alcuni sono nuovi rispetto alle versioni precedenti dell'XHTML e soprattutto dell'HTML, altri sono quelli classici che molti già conoscono. Non entreremo in dettaglio dato che non sono specifici delle sole tabelle, argomento appunto di questo articolo.

La prima cosa che salta all'occhio è che l'elemento principale, ovvero `table`, ha solo gli attributi comuni. Che fine hanno fatto `width`, `height`, `cellpadding`, `cellspacing`, `border` e `align`? La risposta è semplice: sono tutti attributi che adesso **vanno** obbligatoriamente definiti **solo** con i fogli di stile. Già nell'XHTML 1.0, nella versione *strict*, `align` era stato relegato fra i cosiddetti attributi deplorati (*deprecated*), ma gli altri avevano ancora mantenuto a pieno titolo il loro ruolo di attributo valido. Così non è più nell'XHTML 2 che intende forzare gli sviluppatori a una maggiore coerenza nell'uso dei fogli di stile.

Lo stesso vale per tutti i sottoelementi di `table`: anche se alcuni hanno altri attributi oltre a quelli comuni, lo sviluppatore può dare solo specifiche di natura concettuale, non stilistica.

Può ad esempio dire quante righe o colonne sottende una cella, ma non quanto è larga o alta la singola riga o colonna. Per quello bisogna usare i corrispondenti attributi nel foglio di stile.

Vediamo quindi i vari sottoelementi. Il primo è **caption** e serve a descrivere la natura della tabella, ce ne può essere solo uno, può essere omesso e comunque va messo subito dopo **table**. Usato relativamente poco dagli sviluppatori HTML, che spesso preferivano realizzarlo manualmente con un paragrafo posto prima o dopo la tabella, questo tipo di elementi sta acquistando sempre una maggiore importanza in un'ottica di web semantico. Utilizzare infatti un paragrafo per la didascalia di una tabella non permette a un'applicazione che analizzi il documento di comprendere se quel determinato paragrafo è realmente collegato logicamente alla tabella e in che modo. L'elemento **caption**, invece, è naturalmente parte della tabella stessa e ha un ruolo che qualunque applicazione può interpretare correttamente nell'ambito di un'analisi semantica.

Il secondo sottoelemento è **summary**. Come **caption** serve a spiegare in cosa consiste la tabella, che contenuti ha, a cosa serve, ma più in dettaglio rispetto all'elemento precedente. Lo scopo di **summary** è di fornire informazioni dettagliate sulla tabella per quei sistemi che ne permettono l'accessibilità anche con metodi alternativi alla rappresentazione grafica, come i sintetizzatori vocali e i lettori Braille per non vedenti. Questo contenuto, quindi, non è in genere visualizzato come **caption** dai browser tradizionali, sebbene debba essere obbligatoriamente messo a disposizione del visitatore tramite un qualche meccanismo come un menù contestuale, un tasto di selezione rapida e via dicendo.

Il terzo sottoelemento, o meglio, il terzo blocco di elementi, è formato da zero o più elementi **col** o **colgroup**. Questi elementi permettono di dividere una tabella in colonne. Queste colonne non rappresentano necessariamente le colonne della tabella, ma una suddivisione verticale di comodo che tiene conto di tutti i possibili posizionamenti delle varie celle della stessa. Concettualmente sono simili alle specifiche di tabulazione di un documento (*figura 1*).

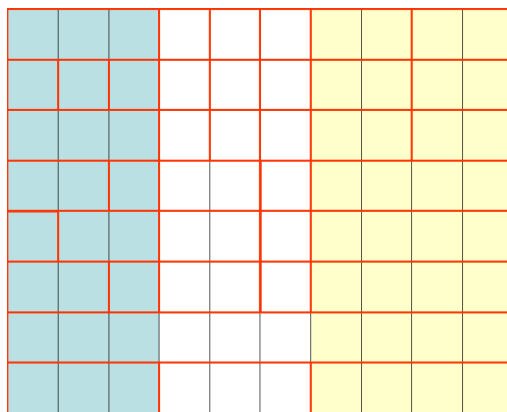


Figura 1 - Esempio di tabella

L'elemento **colgroup** serve a definire un gruppo di una o più colonne, a seconda del valore assunto dall'attributo **span** o del numero di elementi **col** presenti all'interno dell'elemento (i

due metodi sono mutuamente esclusivi). Un gruppo può rappresentare un insieme di colonne logicamente correlate, come l'indirizzo, il telefono e l'indirizzo di posta elettronica di una persona, oppure un gruppo di colonne aventi gli stessi attributi o le stesse caratteristiche nel foglio di stile. Analogamente l'elemento `col` rappresenta una singola colonna oppure una sequenza di colonne affiancate a seconda che `span` sia uno (od omissso) o maggiore di uno.

Ad esempio, la tabella in *figura 7* può essere definita nel modo seguente dal punto di vista della struttura:

```
<colgroup id="azzurro" span="3" />
<col id="bianco_1" />
<col id="bianco_2" />
<col id="bianco_3" />
<colgroup id="giallo">
<col id="giallo_1" span="2">
<col id="giallo_2" span="2">
</colgroup>
```

e in questo dal punto di vista dello stile:

```
#azzurro { width: 150px; background-color: #BBE0E3 ; }
#bianco_1 { width: 50px }
#bianco_2 { width: 50px }
#bianco_3 { width: 50px }
#giallo { background-color: #FFFFCC }
#giallo_1 { width: 100px }
#giallo_2 { width: 100px }
```

Fin qui per quello che riguarda la tabella nel suo complesso. Gli elementi successivi riguardano invece il contenuto vero e proprio della tabella, ovvero i dati. Questi possono essere strutturati in diverse sezioni. Ne esistono di tre tipi: **thead**, **tfoot** e **tbody**. Le prime due possono anche non esserci, ma se ci sono allora devono essere presenti al massimo una sola volta; della terza ce ne deve essere almeno una, ma è possibile averne anche di più. All'interno di queste sezioni ci possono essere una o più righe, ovvero non possono essere vuote.

Questa divisione in sezioni serve principalmente per il meccanismo di paginazione, là dove applicabile. Supponiamo ad esempio di stampare una tabella piuttosto lunga. Verosimilmente le varie righe si distribuiranno su più pagine. Quelle contenute nelle sezioni **thead** e **tfoot** saranno presenti in tutte le pagine, rispettivamente in testa e in fondo alle altre, mentre le varie sezioni **tbody** saranno, se possibili, distribuite in modo che le righe di

ognuna di queste sezioni vengano raggruppate insieme nelle singole pagine (*figura 2*). Da notare che **sia `thead` che `tfoot`** devono essere posti prima della prima delle sezioni **`tbody`**.

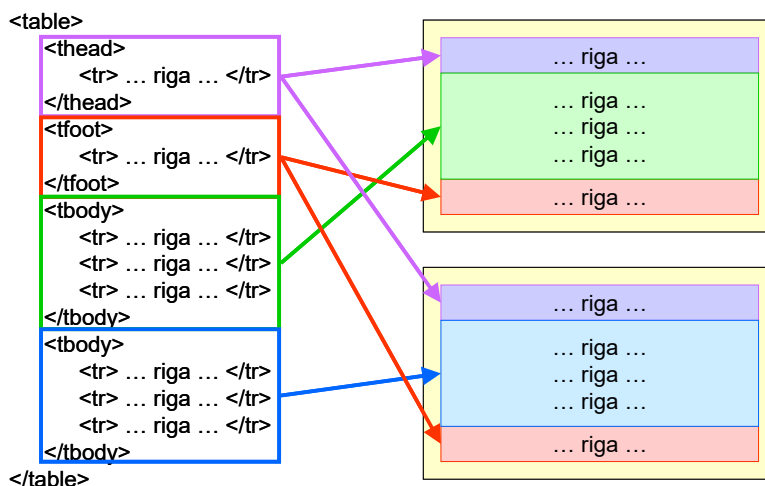


Figura 2 - Sezioni delle tabelle e paginazione

Il meccanismo di paginazione non è detto tuttavia che serva solo in caso di stampa. Anche una applicazione visuale può decidere di applicarlo: esempi classici sono le presentazioni, generalmente strutturate in fogli (*slide*) e i browser che girano su dispositivi dotati di schermo molto piccolo, come i telefonini e i palmari.

Gli ultimi tre elementi sono quelli che servono a definire le righe e le celle della tabella. Le righe sono identificate dall'elemento **`tr`**, mentre le celle possono essere di due tipi: dati (**`td`**) ed etichette (**`th`**). Le etichette servono a classificare i dati e generalmente sono poste in una riga in cima alla tabella, in una colonna alla sua sinistra, o in entrambe le posizioni. Strutture più complesse sono comunque possibili.

Nell'XHTML 2 il collegamento logico fra dati ed etichette ha acquisito una particolare rilevanza. In passato, infatti, gli elementi **`th`** e **`td`** servivano più che altro a rappresentare differenze stilistiche nella tabella. Ad esempio, le etichette erano rese in grassetto. In un'ottica di web semantico, invece, diventa sempre più importante riuscire a dare una chiave di lettura della tabella al di là dell'aspetto esteriore. Ecco allora che questi due elementi condividono tutta una serie di attributi aggiuntivi. Vediamoli più in dettaglio.

Il primo è **`abbr`**, che serve a fornire una versione abbreviata del contenuto della cella. Questa informazione viene utilizzata specialmente nel caso di sintesi vocale o di rapporto stampato, quando si desidera associare ad ogni cella la corrispondente etichetta senza tuttavia riportarla per intero.

Il secondo è **`axis`**, che permette di associare a una cella, sia che si tratti di un dato che di una etichetta, una o più categorie. Queste categorie non è detto che corrispondano a elementi

presenti esplicitamente nella tabella, ma piuttosto aggiungo un ulteriore livello di metainformazioni alla tabella. Ad esempio:

```
<th axis="località">Roma</th>
<th axis="stagione">Estate</th>
...
<td axis="data">8 luglio 1986</td>
<td axis="prezzo">&euro; 38,00</td>
...
```

Il termine *asse* corrisponde alla raffigurazione delle varie categorie in una sorta di spazio multidimensionale nel quale ogni categoria rappresenta appunto un asse o, se si preferisce, una coordinata. Una singola cella può anche corrispondere a più assi, come ad esempio:

```
<td axis="letteratura,prosa,titolo">I Promessi Sposi</td>
```

Il terzo è **headers**. Abbiamo detto che generalmente un dato è associato ad una o più etichette, ma se tale associazione può risultare abbastanza chiara osservando la tabella, è ben più difficile rilevarla se questa viene ad esempio letta da un sintetizzatore vocale. In generale, poi, accedendo il singolo dato all'interno del file XHTML, può essere complesso per un'applicazione ricostruire a quale etichetta esso sia effettivamente associato. Per superare questo problema è sufficiente dare alle varie etichette un identificativo e riportare nell'attributo **headers** del dato la lista degli identificativi corrispondenti allo stesso. Ad esempio

```
<tr>
  <th id="per">Periodo</th>
  <th id="loc">Località</a>
  <th id="sp">Spese</a>
</tr>
...
<tr>
  <th id="apr">Aprile</th>
  <td headers="loc apr">Roma</td>
  <td headers="sp apr">&euro; 38,00</td>
</tr>
```

A questo punto diventa semplice anche per un'applicazione associare i 38 euro all'etichetta identificata da **sp** e a quella identificata da **apr**, ovvero comprendere che si tratta di una spesa effettuata nel mese di Aprile.

Per strutture più semplici, in alternativa a **headers**, si può usare **scope**. Questo attributo è in genere utilizzato per le etichette, ma può essere presente sia in un elemento **th** che in un elemento **td**. Quando usato per un elemento **th**, esso indica a quali celle l'etichetta si riferisce. Se il valore è **row**, allora l'etichetta si riferisce a tutte le celle della stessa riga; quando è **col** a tutte quelle della stessa colonna; quando è **rowgroup** o **colgroup** a tutte quelle del gruppo di righe o di colonne del quale la cella che contiene l'etichetta fa parte. L'utilizzo di **scope** è permesso anche con **td** nel caso una cella sia allo stesso tempo considerata sia etichetta che dato. Ad esempio, il dato **Roma** dell'esempio precedente potrebbe anche essere pensato come un'etichetta per la spesa nel caso si vogliano analizzare tutte le spese effettuate in quella città indipendentemente dal periodo.

Infine **rowspan** e **colspan**, che indicano per quante righe o colonne, rispettivamente, una determinata cella si estende. Di norma il valore è 1, ma è possibile estendere una cella per più righe e/o colonne. In particolare il valore 0 significa: estendi la cella dal punto dove si trova fino al termine della tabella. Quindi, se una cella ha **rowspan="0"**, la cella si estenderà fino al margine destro della tabella, mentre se ha **colspan="0"**, si estenderà fino a quello inferiore.

Come si può vedere l'XHTML 2 ha semplificato parecchio gli elementi e gli attributi delle tabelle, eliminando tutto ciò che appartiene concettualmente ai fogli di stile e aggiungendo tutta una serie di elementi e attributi che ne definiscono la struttura concettuale senza per questo fissarla in termini di unità di misura, fornendo inoltre tutta una serie di metainformazioni che permettono l'interpretazione delle informazioni tabulate anche da parte di applicazioni informatiche.

Questo è quanto, per il momento. Nel prossimo articolo vedremo come si impostino le proprietà dei fogli di stile in CSS 3 per dare un particolare aspetto a una tabella.

TABELLE

Elementi	Attributi	Contenuti
table	comuni	caption?, summary?, (col* colgroup*), ((thead?, tfoot?, tbody+) (tr+))
caption	comuni	(PCDATA testo)*
summary	comuni	(PCDATA flusso)*
col	comuni, span	EMPTY
colgroup	comuni, span	col*
thead	comuni	tr+
tfoot	comuni	tr+
tbody	comuni	tr+
tr	comuni, moduli	(td th)+
td	comuni, abbr, axis, colspan, headers, rowspan, scope	(PCDATA flusso)*
th	comuni, abbr, axis, colspan, headers, rowspan, scope	(PCDATA flusso)*

Tabella A - Elementi, attributi e contenuti delle tabelle