Il DTD di HTML e XHTML

Fabio Vitali

Introduzione

Oggi esaminiamo in breve:

- Il DTD di HTML 4
- XHTML 1.0

Introduzione al DTD di HTML

Il linguaggio HTML è un tipo di documenti SGML (esiste un DTD di HTML).

HTML viene usato dai browser WWW per visualizzare documenti ipertestuali. Tramite HTML è possibile realizzare documenti con una semplice struttura, aspetti grafici anche sofisticati, che contengono testo, immagini, oggetti interattivi e connessioni ipertestuali ad altri documenti

HTML è esistito in varie versioni dal 1989 ad oggi, compresa una forma in XML, chiamata XHTML.

XHTML1.0: Nel 1998 parte l'iniziativa di riformulare HTML come applicazione di XML, piuttosto che di SGML.

Il 26 gennaio 2000 esce la prima recommendation del W3C, XHTML 1.0, che è una semplice riformulazione di HTML 4 in termini di XML, senza nessuna introduzione di nuove forme.

I DTD di HTML 4

Per esigenze di compatibilità, HTML 4 è composto di tre DTD alternativi:

- Transitional DTD (detto anche loose): contiene l'intero linguaggio ammesso per HTML, inclusi quegli elementi "deprecati" che vengono mantenuti per compatibilità col passato.
- Strict DTD: contiene i soli elementi di HTML che non vengono influenzati dall'uso degli style sheet (ma sono escluse le tabelle, che non sono gestite da CSS).
- Frameset DTD: un semplicissimo DTD per quei documenti in cui al posto di BODY si usano i tag dei frame.

Criteri di sviluppo (1)

HTML 4.0 estende HTML 3.2 con meccanismi per i fogli di stile, gli script, i frame, oggetti embedded, criteri di internazionalizzazione, tabelle più ricche e miglioramenti ai form.

Questi sono i criteri di sviluppo più importanti:

- **Internazionalizzazione** (*Internationalization* o *I18N*): l'adozione dei meccanismi necessari per il supporto di linguaggi e notazioni di tutto il mondo, e per la creazione di documenti contenenti linguaggi misti.
- **Accessibilità**: l'adozione dei meccanismi necessari per il supporto delle esigenze degli utenti con limitazioni fisiche (visive, uditive, etc.).

A seguire: Criteri di sviluppo (2)

Criteri di sviluppo (2)

- **Supporto per oggetti multimediali** l'adozione dei meccanismi necessari per inserire (embed) in maniera generalizzata oggetti di ogni possibile media all'interno di una pagina HTML.
- **Style sheet**: l'adozione di meccanismi per specificare in maniera precisa e sofisticata la resa tipografica di una pagina senza appesantire la gestione del contenuto.
- **Scripting**: l'adozione dei meccanismi necessari per realizzare sul client degli oggetti attivi, in grado di eseguire computazioni locali (ad esempio, per pre-verificare la correttezza delle informazioni inserite in un form).

Una visita al DTD di HTML 4: loose DTD

XHTML 1.0

XHTML 1.0 è una riformulazione di HTML 4 come un'applicazione di XML 1.0. La semantica degli elementi e degli attributi non è assolutamente cambiata da HTML 4.

XHTML è il primo di una serie di DTD che riproducono, limitano ed estendono HTML. Sono fatti per lavorare con user agent basati su XML, ma con un'esplicita strategia di transizione.

I documenti XHTML sono documenti XML, scritti per essere usati anche in ambienti HTML 4, danno accesso immediato a tulte le caratteristiche DOM.

Differenze con HTML 4 (1)

- I documenti debbono essere ben formati, in particolare l'annidamento deve essere corretto.

```
<p>Paragrafo con <em>enfasi</p></em>
```

- I nomi di elementi e attributi sono minuscoli (XML è case-sensitive)

```
<P> Oggi siamo <EM>qui</EM></P>
```

- Il tag finale è obbligatorio per elementi non vuoti

```
<UL>
    <LI> Primo elemento
    <LI> secondo elemento
</UL>
```

- Gli elementi vuoti debbo seguire la sintassi XML

```
<HR/> <BR/>
```

Differenze con HTML 4 (2)

- I valori degli attributi debbono sempre avere le virgolette

```
<input type=checkbox value=pippo>
```

- Non esiste il concetto di minimizzazione degli attributi

```
<input type="radio" checked>
<input type="radio" checked="checked">
```

- Elementi con attributi “id” e “name”

  In HTML esistono due tipi di attributi identificatori: “name” per i tag come a, applet, form, frame, iframe, img, e map, ed “id” per esprimere caratteristiche di stile per tutti gli elementi.

  In XML, solo UN attributo alla volta può essere di tipo ID. E’ stato scelto quindi che l’attributo “id” sia di tipo ID, mentre “name non lo è. Quindi documenti XHTML debbono usare “id” come attributo per l’identificazione di frammenti, e “name” è diventato deprecato.

Differenze con HTML 4 (3)

- Elementi di stile e script

  Poiché il contenuto di SCRIPT e STYLE è definito come #PCDATA, i caratteri “<“ e “&” sono significativi per XML e quindi vengono interpretati come markup. Per evitarlo, è necessario mettere gli script e gli stili esternamente, oppure usare sezioni CDATA.

```
<script>
        <![CDATA[ qualunque carattere ]]>
</script>
```

- Esclusioni SGML

  Il content model di alcuni elementi in HTML esclude esplicitamente l’annidamento ricorsivo (es., A non può contenere altri A). Questo purtroppo non è esprimibile in XML, ma è comunque non accettabile.

Compatibilità con HTML

- **Processing instructions**

  Alcuni browser rendono le processing instructions come tag.

- **Elementi vuoti**

  Affinché un elemento vuoto sia comprensibile sia da motori XML che HTML, è possibile mettere uno spazio vuoto prima del carattere di fine tag:

  ```
  <HR /><BR />
  ```

- **Stili e script embedded**

  Stili e script vanno posti fuori dal documento se contengono “<“, “&” o “]]>”. Da notare che molti processori XML rimuovono i commenti, quindi non si può più inserire gli script e gli stili in commenti SGML.

- **Identificatori di frammenti**

  E’ opportuno usare sia l’attributo id che l’attributo name, ogni volta che ve ne sia bisogno.

La modularizzazione di XHTML

La diffusione dei device non da scrivania (es. portatili, palmari, cellulari, orologi da polso, smart card, apparecchi su automobili, ecc.) nonché il fallimento di linguaggi di markup specifici (vedi il caso di WML) ha portato il W3C a ripensare completamente il senso di XHTML.

In particolare, a prevedere l'esistenza di una famiglia di linguaggi XHTML, ciascuno specificamente adattato al tipo di device su cui gira, e tuttavia *sostanzialmente* similari.

Il fondamento della modularizzazione di XHTML si basa sulla possibilità di crearsi un proprio linguaggio di markup usando parti note di XHTML, selezionando solo quelle adatte e ignorando le altre, e magari aggiungendo altre sezioni specifiche.

I moduli di XHTML (1)

XHTML è stato diviso nei seguenti moduli:

- Attribute collection
  - Core (obbligatori): class, id, title
  - I18N: xml:lang
- Core modules (obbligatori)
  - Structure module: html, head, body, title
  - Text module
    - **Heading**: h1 | h2 | h3 | h4 | h5 | h6
    - **Block**: address | blockquote | div | p | pre
    - **Inline**: abbr | acronym | br | cite | code | dfn | em | kbd | q | samp | span | strong | var
  - Hypertext module: a
  - List module: ul, ol, li, dl, dd, dt

I moduli di XHTML (2)

- Text extension module
  - Presentation module: hr, b, i, tt, big, small, sub, sup
  - Edit module: ins, del
  - Bi-directional text module: bdo
- Forms modules
  - Basic Form module:
    - form, input, select, textarea, option, label + attributi come HTML 3.0
  - Forms module
    - Basic Form + vari attributi + button, fieldset, legend, optgroup
- Table modules
  - Basic Tables Module:
    - table, tr, td, th, caption + attributi come HTML 3.0
  - Tables Module:
    - Basic Tables + vari attributi + col, colgroup, tbody, thead, tfoot

I moduli di XHTML (3)

- Images modules
  - Basic image module: img
  - Client-side image map module: @usemap, map, area,
  - Server-side image map module: @ismap
- Frames module
  - Frames module: frameset, frame, noframes
  - Iframes module: iframe
  - Target module: @target
- Object modules
  - Object module: object, param
  - Applet module: applet, param (deprecato)

I moduli di XHTML (4)

- Scripting modules
  - Scripting module: noscript, script
  - Intrinsic events module: vari attributi di eventi (@onload, @onclick, @onfocus, @onselect, ecc.)
- Stylesheet modules
  - Stylesheet module: style
  - Style attribute: @style
- Head modules
  - Metainformation module: meta
  - Link module: link
  - Base module: base
- Other modules
  - Name identification modules: @name (deprecato: si usi id)
  - Legacy modules: basefont, center, dir, font, isindex, menu, s, strike, u (deprecati)

XHTML Basic e XHTML 2.0

La modularizzazione di XHTML ha permesso la creazione di vari sottolinguaggi derivati da HTML. In particolare:

- **XHTML Basic**: pensato per device con poche capacità di calcolo, e con schermi piccoli e/o non grafici. Include obbligatoriamente i moduli core (structure, text, hypertext e list), in più usa immagini, form, basic table e object. Infine permette facoltativamente di selezionare altri moduli, come lo scripting module
- **XHTML 2.0**: ancora in draft, è la prima vera estensione di HTML dal 1997. Definisce svariati nuovi elementi, ed un certo numero di nuovi attributi. Elimina definitivamente gli elementi deprecati di HTML, appoggiando tutte le specifiche presentazionali ai fogli di stile. Fornisce elementi per funzionalità oggi realizzate con script. (es. menu ottenuto con nl - navigation list).

Conclusioni

Oggi abbiamo parlato di

- La storia di HTML
- Il DTD di HTML 4
- XHTML 1.0
- Modularizzazione di XHTML
- Evoluzione di XHTML

Riferimenti

- D. Raggett, A. Le Hors, I. Jacobs, *HTML 4.01 Specification*, W3C Recommendation 24 December 1999, **http://www.w3.org/TR/html401**
- S. Pemberton et alii, *XHTML™ 1.0: The Extensible HyperText Markup Language, A Reformulation of HTML 4 in XML 1.0*, W3C Recommendation 26 January 2000, **http://www.w3.org/TR/xhtml1**
- M. Altheim et alii, *Modularization of XHTML*, W3C Recommendation 10 April 2001, W3C Recommendation 10 April 2001, **http://www.w3.org/TR/xhtml-modularization**
- M. Altheim, S. McCarron, *XHTML™ 1.1 - Module-based XHTML*, W3C Recommendation 31 May 2001, **http://www.w3.org/TR/xhtml11**
- J. Axelsson et alii, *XHTML™ 2.0*, W3C Working Draft 31 January 2003, **http://www.w3.org/TR/2003/WD-xhtml2-20030131**