

Dispense XML

Indice

<u>Dispense XML</u>	1
<u>Breve introduzione a XML</u>	1
<u>Analisi e Visualizzazione del documento</u>	1
<u>Supporto Microsoft per XML</u>	3
<u>XML server-side</u>	3
<u>Spazio dei nomi</u>	4
<u>Struttura di un documento XML</u>	5

Dispense XML

Breve introduzione a XML

XML è un nuovo linguaggio creato a partire dallo standard SGML al fine di garantire maggior flessibilità e generalità di utilizzo per implementare diversi tipi di dati.

È stato sviluppato dallo XML Working Group (originariamente noto come SGML Editorial Review Board) costituitosi all'interno del W3C nel 1996.

Obiettivi:

- 1. Creare un linguaggio utilizzabile con facilità in Internet ma più flessibile di HTML.
- 2. Compatibilità con applicazioni differenti.
- 3. Facilità di progettazione ed elaborazione.
- 4. Necessità di disporre di un linguaggio che permettesse di descrivere e strutturare i dati.
- 5. Comprensibilità elevata dei documenti grazie all'utilizzo di elementi aventi nomi non arbitrari o convenzionali ma che rispecchino il loro contenuto
- 6. Permettere la pubblicazione online di documenti indipendenti dal tipo di dispositivo che vi avrà accesso.

XML è un metalinguaggio che permette la definizione di markup.

È possibile definire delle classi di documenti significative (chiamate DTD – Document Type Definition ossia definizione di tipo del documento) e associarvi particolari proprietà mediante un foglio di stile esterno, che può essere realizzato in formato CSS o XSL. Si possono anche utilizzare DTD già create (come MathML–Mathematical Markup Language).

Analisi e Visualizzazione del documento

Un documento XML viene interpretato da una specifica applicazione, costituita da due parti fondamentali:

Un parser, che esegue il controllo sintattico del documento e si occupa della gestione degli errori. Il parser esegue il controllo su due livelli:

- * Sulla validità del documento, se esiste una DTD.
- * Sulla forma del documento (se è ben formato oppure no).

Un processore, che si occupa di visualizzare il documento utilizzando un apposito foglio di stile.

Per ottenere una visualizzazione della pagina web è necessario aggiungere valore semantico agli elementi dichiarati, ossia applicare al documento un foglio di stile (CSS o XSL) tramite una dichiarazione posta subito dopo l'iniziale dichiarazione del linguaggio:

```
<?xml:stylesheet href="xml.css" title="Stile" type="text/css"?>
```

Esempio di foglio di stile CSS:

Dispense XML

```
title {  
    display:block;  
    font-family: Arial, Helvetica;  
    font-weight: bold;  
    font-size: 20pt;  
    color: #9370db;  
    text-align: center;  
}  
ISBN {  
    display:block;  
    font-family: Arial, Helvetica;  
    font-weight: bold;  
    font-size: 12pt;  
    color: #c71585;  
    text-align: left;  
}  
authors {  
    display:inline;  
    font-family: Arial, Helvetica;  
    font-style: italic;  
    font-size: 10pt;  
    color: #9370db;  
    text-align: left;  
}  
description {  
    display:block;  
    font-family: Arial, Helvetica;  
    font-size: 12pt;  
    color: #ff1010;  
    text-align: left;
```

}

Supporto Microsoft per XML

- Explorer 5.0:
- Visualizzazione immediata dei documenti XML, sia in versione albero, sia formattati con un foglio di stile.
- XML engine, che garantisce il supporto rispetto alle specifiche W3C di XML 1.0 e degli spazi dei nomi.
- Supporto di XSLT, tramite Microsoft XSLT processor.
- Supporto di XML Schema. Gli schemi sono elementi di XML che definiscono in modo rigido le regole di un documento XML per quanto riguarda i tag utilizzati e i tipi di dati.
- DOM (Document Object Model) XML, interfaccia di programmazione che offre agli sviluppatori controllo su contenuto, struttura e formato dei documenti XML.

Differenze specifiche W3C – supporto Microsoft:

- Spazio dei nomi XSLT: Explorer richiede lo spazio dei nomi relativo alla versione precedente di XSL (<http://www.w3.org/TR/WD-xsl>) mentre le specifiche W3C <http://www.w3.org/1999/XSL/Transform>.
- Tipo MIME XSLT: "test/xsl" invece che "text/xml".
- Template di default: non supportato.

XML server-side

É possibile effettuare la trasformazione del codice XML mediante:

- Codice ASP.
- PHP.
- Java servlet.
- Appositi processori.

Esempio codice ASP:

```
<%  
  
'Load the XML  
  
set xml = Server.CreateObject("Microsoft.XMLDOM")  
  
xml.async = false  
  
xml.load(Server.MapPath("esami2.xml"))  
  
'Load the XSL  
  
set xsl = Server.CreateObject("Microsoft.XMLDOM")  
  
xsl.async = false  
  
xsl.load(Server.MapPath("esami2.xsl"))  
  
'Transform the file
```

Dispense XML

```
Response.Write(xml.transformNode(xsl))
```

```
%>
```

Breve spiegazione:

- Creazione oggetto DOM e caricamento file xml.
- Creazione oggetto DOM e caricamento file xsl.
- Trasformazione del file mediante metodo "write".

Processori XSLT

Programmi, realizzati generalmente in linguaggio Java o C++, che si occupano di applicare un foglio di stile XSLT a un documento XML e di produrre il documento risultante come file indipendente.

Processori piú diffusi:

- **Sablotron** (<http://www.gingerall.com>) : scritto in C++. Può essere usato da linea di comando o come modulo Perl. Sintassi:

```
sabcmd file.xsl file.xml file.htm
```
- **Xalan** (<http://xml.apache.org/xalan/overview.html>): processore creato dalla ASF (Apache Software Foundation), con tecnologia derivata da precedente prodotto IBM (LotusXSL). Disponibile in Java e in C++. Free, disponibile per numerose piattaforme. Può essere richiamato sia da linea di comando o da Java API.
- **Saxon** (<http://users.iclway.co.uk/mhkay/saxon>): programma Java. Può essere richiamato da linea di comando, da una classe Java oppure tramite servlet.

Altri prodotti simili: Oracle XSL9, XT10, iXSLT11, 12, Stylus13, MSXML314.

Spazio dei nomi

Lo spazio dei nomi (namespace) di XML é un modo per eliminare ambiguità nell'uso di elementi, marcatori e attributi.

Nome attribuito alla lista degli elementi contenuti nel documento.

Ognuno degli elementi é preceduto da un prefisso, separato dal nome dal carattere ":". es fo:, utilizzato da XSL.

XML utilizza l'attributo predefinito "xmlns" (eXtensible Markup Language Name Space) per introdurre i prefissi utilizzati. Il valore di quest'attributo é un URI, ad esempio <http://www.w3.org/1999/XSL/Format>.

Ogni spazio dei nomi deve essere dichiarato prima di poter essere usato.

La sintassi per la dichiarazione é:

```
xmlns:[prefisso] = " [URI] "
```

Struttura di un documento XML

Un documento XML deve innanzitutto essere ben formato, ossia rispettare tutte le regole di sintassi previste dal linguaggio:

- Presenza di un unico elemento radice che contiene tutti gli altri.
- Chiudere sempre tutti gli elementi aperti (a meno che siano elementi vuoti, che vanno comunque chiusi con la speciale sintassi
`
`)
- Differenza tra maiuscolo e minuscolo.

Un documento XML é valido se possiede una DTD a cui corrisponde. A indicare se é provvisto o meno di una DTD interna o esterna é l'attributo standalone nella dichiarazione iniziale:

Encoding permette di definire il tipo di codifica utilizzato per presentare i dati.

```
<?XML version="1.0" standalone="yes"
      encoding="UTF-8"?>
```

Struttura **gerarchica** basata su elemento radice e figli.

Forma degli elementi:

```
<nome
      (attributo:valore)></nome>
```

Elementi vuoti :

```
<nome />
```

I tag possono iniziare con una lettera, un underscore (_), oppure ":". Non é possibile iniziare nessun tag con la combinazione di lettere "xml", né maiuscole né minuscole.

DTD

Un elemento XML puó avere una struttura predefinita tramite la DTD in modo da vincolarne il contenuto.

Definizione dell'elemento principale, dei figli, degli attributi (con il dominio di valori assumibili).

É possibile anche dichiarare entità, ossia riferimenti a blocchi di dati, interni o esterni al documento.

Una DTD descrive lo "schema" del documento.

Codifica per elementi e attributi :

```
'
```

Indica che l'ordinamento dato deve essere rispettato

```
|
```

Indica che é possibile inserire gli elementi in qualsiasi ordine

Dispense XML

+

Indica che deve essere presente almeno un elemento di quel tipo

*

Indica che può essere presente un elemento di quel tipo

?

Indica che l'elemento é opzionale

()

Indica un raggruppamento

I tipi di dati possono essere:

PCDATA, caratteri interpretati.

CDATA, caratteri non interpretati.

Per ogni elemento possono poi essere indicati una serie di attributi (**ATTLIST**), definiti in base al tipo di carattere (PCDATA o CDATA) e alla obbligatorietà o meno della loro presenza.

Tre casi base:

- **#REQUIRED**, la presenza dell'attributo é obbligatoria.
- **#FIXED**, il valore dell'attributo é fisso.
- **#IMPLIED**, l'attributo é previsto ma non obbligatorio e non ha un valore di default.

Esempio di DTD

```
<!DOCTYPE books [  
<!ELEMENT books (book+)>  
<!ELEMENT book (title, coll, ISBN, authors, description?, price+)>  
<!ELEMENT title (#PCDATA)>  
<!ELEMENT ISBN (#PCDATA)>  
  
<!ELEMENT authors (author+)>  
  
<!ELEMENT author (#PCDATA)>  
  
<!ELEMENT description (#PCDATA)>
```

Esempio di documento XML:

Dispense XML

```
<!DOCTYPE books SYSTEM "books.dtd">

  <books>

    <book>

      <title>Titolo del libro</title>

      <ISBN>1234567</ISBN>

      <authors>Nomi degli autori

        <author>Primo autore</author>

        <author>Secondo autore</author>

      </authors>

      <description>Breve descrizione del contenuto del libro</description>

    </book>

  </books>
```

Last Modified Date: 03/20/2001 09:42:34