XML

Moreno Marzolla
INFN Sezione di Padova
moreno.marzolla@pd.infn.it
http://www.dsi.unive.it/~marzolla

Testo di riferimento

- Erik T. Ray, *Learning XML*, O'Reilly, First Edition, January 2001 ISBN: 0-59600-046-4, 368 pages

Cos'è XML?

- Extensible Markup Language (XML) è una notazione per rappresentare informazioni in maniera strutturata e platform-independent
  - Standard aperto
  - Supporta il set di caratteri Unicode
  - Sintassi chiara, semplice e non ambigua
  - XML può essere combinato con stylesheet per produrre documenti con stili arbitrari
- XML non è un linguaggio di markup, ma una notazione per definire linguaggi di markup

Markup

- Col termine *markup* si intende qualsiasi informazione aggiunta ad un documento con lo scopo di migliorarne il significato
  - Il *markup* identifica parti del documento, e come queste sono in relazione con altre
- Esempio

```
<message>
   <exclamation>Hello, world!</exclamation>
   <paragraph>XML is <emphasis>fun</emphasis> and
   <emphasis>easy</emphasis> to use.
   <graphic fileref="smiley_face.pict"/></paragraph>
</message>
```

Concetti fondamentali / 1

- Boundaries
  - I tag <message> e </message> definiscono l'inizio e la fine di una collezione di testo e markup
- Roles
  - Quale è il ruolo di una regione di testo? Nell'esempio i tag <paragraph> e </paragraph> identificano del testo come paragrafo, anziché una lista, un titolo o altro
- Positions
  - Esiste un ordinamento (totale) tra frammenti di documento. Ad esempio, il paragrafo appare dopo il testo etichettato come <exclamation>, quindi verrà probabilmente stampato o visualizzato in quest'ordine

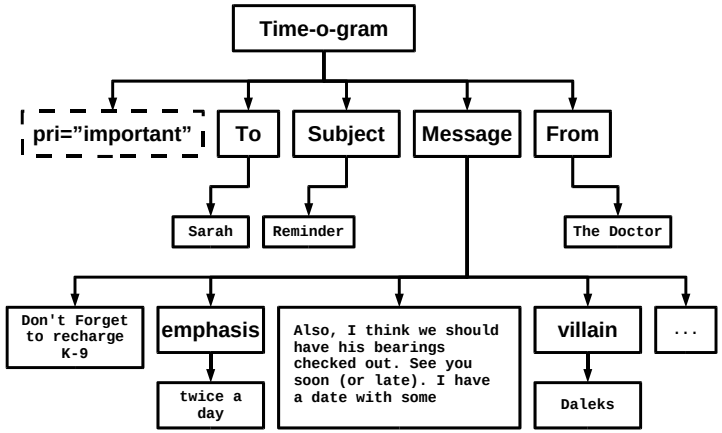
Concetti fondamentali / 2

- Containment
  - Il testo “fun” si trova in un elemento <emphasis>, il quale si trova dentro ad un <paragraph>, che è dentro a <message>. L'annidamento degli elementi può essere utilizzato dal software che processa il frammento XML, che può subire elaborazioni diverse in base a dove si trova
- Relationships
  - Un frammento di testo può essere collegato ad una risorsa che si trova in una locazione diversa. Ad esempio, il tag <graphic fileref="smiley_face.pict"/> potrebbe definire una relazione tra un frammento XML e il file chiamato smiley_face.pict.

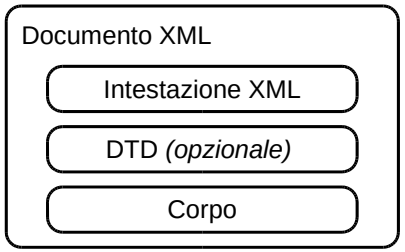
Anatomia di un documento XML

```
<?xml version="1.0"?>
<time-o-gram pri="important">
   <to>Sarah</to>
   <subject>Reminder</subject>
   <message>Don't forget to recharge K-9
<emphasis>twice a day</emphasis>. Also, I think we
should have his bearings checked out. See you soon
(or late). I have a date with some
<villain>Daleks</villain>...
   </message>
   <from>The Doctor</from>
</time-o-gram>
```


Diagramma ad albero di un documento XML



Sintassi generale



Dichiarazione XML

```
<?xml [ name1="val1" [ name2="val2" ...] ] ?>
```

- Proprietà che è possibile settare:
  - **`version`**
    - Numero di versione (al momento "1.0")
  - **`encoding`**
    - Definisce il set di caratteri utilizzato nel documento, ad esempio "US-ASCII" o "iso-8859-1", "UTF-8" (set standard di caratteri latini)
  - **`standalone`**
    - Se standalone="yes" non ci sono file esterni da includere. Altrimenti, standalone="no"
- Esempi:

```
<?xml version="1.0" standalone="no" encoding="UTF-8"?>
<?xml version="1.0"?>
```


Dichiarazione DTD

```
<!DOCTYPE root-element uri-of-dtd
[
   internal-subset
]>
```

- Esempio:

```
<!DOCTYPE time-o-gram
     PUBLIC "-//LordsOfTime//DTD TimeOGram 1.8//EN"
     "http://www.lordsoftime.org/DTDs/timeogram.dtd"
[
     <!ENTITY sj "Sarah Jane">
     <!ENTITY me "Doctor Who">
]>
```


Elementi XML

- Elemento generico

```
<name attr1="val1" attr2="val2" >
   contenuto
</name>
```

- Elemento vuoto

```
<name attr1="val1" attr2="val2" />
```

Entità interne

```
<?xml version="1.0"?>
<!DOCTYPE message
   SYSTEM "/xmlstuff/dtds/message.dtd"
[
   <!ENTITY client "Mr. Rufus Xavier Sasperilla">
   <!ENTITY agent "Ms. Sally Tashuns">
   <!ENTITY phone "<number>617-555-1299</number>">
]>
<message>
   <opening>Dear &client;</opening>
   <body>
       We have an exciting opportunity for you! A set of
       ocean-front cliff dwellings in Pi&#241;ata, Mexico
       have been renovated as time-share vacation homes.
       They're going fast! To reserve a place for your
       holiday, call &agent; at &phone;. Hurry, &client;.
       Time is running out!
   </body>
</message>
```

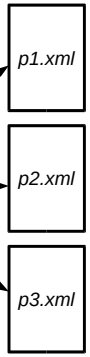
Entità predefinite

- & &
- ' '
- > >
- < <
- " “
- Numbered character entities
  - { indica il carattere di posizione decimale 123 nel set di caratteri Unicode;   indica il carattere di posizione esadecimale 20 nel set di caratteri Unicode
- Named character entities
  - è é ecc.

Entità esterne

```
<?xml version="1.0"?>
<!DOCTYPE longdoc
   SYSTEM "http://www.dtds-r-us.com/generic.dtd"
[
   <!ENTITY part1 SYSTEM "p1.xml">
   <!ENTITY part2 SYSTEM "p2.xml">
   <!ENTITY part3 SYSTEM "p3.xml">
]>
<longdoc>
   &part1;
   &part2;
   &part3;
</longdoc>
```

```
<longdoc>
&part1;
&part2;
&part3;
</longdoc>
```



Entità esterne

- La parola chiave SYSTEM è seguita da un URI

```
<!ENTITY catalog SYSTEM
   "http://www.bobsbolts.com/catalog.xml">
```

  - Però la risorsa può essere spostata altrove, rendendo non valida l'entità
- Per risolvere il problema, è possibile utilizzare un identificatore PUBLIC

```
<!ENTITY faraway PUBLIC "-//BOB//FILE Catalog//EN"
   "http://www.bobsbolts.com/catalog.xml">
```

  - Però il processore XML deve sapere come usare l'identificatore PUBLIC; per tale ragione è opportuno inserire anche un URI come fallback

Commenti e CDATA

- Commenti

```
<!-- testo del commento -->
```

- CDATA (Character Data)
  - Utile per visualizzare blocchi di testo così come scritto, senza che il processore XML interpreti i caratteri speciali

```
<![ CDATA [ testo ]]>
```

  - Esempio

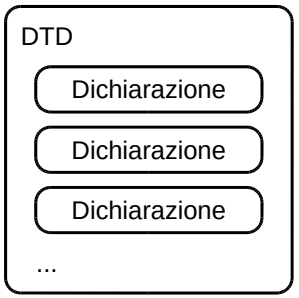
```
<para>&Egrave; possibile visualizzare frammenti di
codice come <![CDATA[if (&x < &y)]]> in documenti
XML.</para>
```


Documenti ben formati (well-formed documents)

- Sono i documenti XML che rispettano le regole sintattiche dei documenti XML
  - Ogni elemento non vuoto deve avere un tag di apertura e uno di chiusura
  - Ogni elemento vuoto deve avere una barra (/) prima della parentesi angolata di chiusura
  - Tutti i valori degli attributi vanno racchiusi tra virgolette (“)
  - I tag di apertura e chiusura devono essere bilanciati
  - Elementi di markup (<,]]> ecc) non possono apparire isolati
  - I nomi di elementi iniziano con lettere o underscore, e contengono solo lettere, numeri, punto, underscore (_), hyphens (-)

DTD

- Un DTD definisce la “grammatica” che un documento XML deve rispettare
- Formato generale:



Element Declaration / 1

```
<!ELEMENT name content-model>
```

- Elementi vuoti

```
<!ELEMENT graphic EMPTY>
```

- Elementi senza restrizioni sul contenuto

```
<!ELEMENT contain-anything ALL>
```

- Elementi che contengono solo caratteri

```
<!ELEMENT emphasis (#PCDATA)>
```

Element Declaration / 2

- Elementi che contengono solo altri elementi

```
<!ELEMENT article (para+)>
<!ELEMENT article (title, (para | sect1)+)>
<!ELEMENT article (title, subtitle?,
                    ((para+, sect1*) | sect1+))>
```

- Elementi con contenuto misto

```
<!ELEMENT para (#PCDATA | emphasis | xref)*>
```


Simboli usati

- , (virgola)
  - Indica una sequenza obbligatoria di elementi
- | (barra)
  - Indica una alternativa
- (content)
  - Raggruppa il contenuto in modo tale che l'operatore che segue si applichi al tutto
- ?
  - Rende l'elemento che precede opzionale
- +
  - Richiede una o più istanze dell'elemento che precede
- *
  - Richiede zero o più istanze dell'elemento che precede

Dichiarazione di liste di attributi

```
<!ATTLIST name
   attname1 atttype1 attdsc1
   attname2 atttype2 attdsc2
>
```

- Specifica:
  - Nome del tag cui si applica l'attributo
  - Nome dell'attributo
  - Tipo dell'attributo
  - Comportamento dell'attributo (obbligatorio/opzionale, valore di default ecc.)

```
<!ATTLIST memo
    id ID #REQUIRED
    security (high | low) "high"
    keywords NMTOKENS #IMPLIED
>
```


Attribute Datatypes / 1

- CDATA
  - Character data: rappresenta una generica sequenza di caratteri

```
<!ATTLIST circle radius CDATA "12 inches">
```

  - Esempi

```
dimensions="35x12x9 mm"
company="O'Reilly &amp; Associates"
text=" 5 + 7 = 3 * 4 "
```

Attribute Datatypes / 2

- NMTOKEN
  - Name Token: Stringa di caratteri che inizia con una lettera e può contenere numeri, lettere e certi simboli di punteggiatura

```
<!ATTLIST part number NMTOKEN #REQUIRED>
```

  - Esempi

```
skin="reptilian"
file="README.txt"
version="v3.4-b"
```


Attribute Datatypes / 3

- NMTOKENS
  - Named Tokens List: lista di token separati da spazi

```
<!ATTLIST article keywords NMTOKENS #IMPLIED>
```

  - Esempi

```
name="Greg Travis"
format="thin blue border"
```


Attribute Datatypes / 4

- ID
  - Identificatore univoco: non possono esistere elementi diversi nel documento XML con lo stesso ID

```
<!ATTLIST record num ID #REQUIRED>
```


Attribute Datatypes / 5

- IDREF
  - Riferimento ad un ID

```
<!ATTLIST relatedword ref IDREF #REQUIRED>
```

- IDREFS
  - Lista di riferimenti ad IDs

```
<!ATTLIST bookset refs IDREFS #REQUIRED>
```

Attribute Datatypes / 6

- ENTITY
  - Questo tipo accetta un nome di entità come argomento

```
<!ATTLIST bulletlist icon ENTITY #IMPLIED>
<!ENTITY bluedot SYSTEM "icons/bluedot.png" NDATA GIF>
```

  - Esempio

```
<bulletlist icon="bluedot">
```

- ENTITIES
  - Stessa cosa, con una lista di entità

Attribute Datatypes / 7

- Enumerate Value List

```
<!ATTLIST part instock ( true | false ) #IMPLIED>
<!ATTLIST schedule day
   ( mon | tue | wed | thu | fri | sat | sun )
   #REQUIRED >
<!ATTLIST shape type
   (circle | square | triangle) "square">
```

  - Esempio

```
<schedule day="fri">...</schedule>
```


Attribute Behavior

- Valore di default
  - Se l'utente non specifica un valore, viene messo quello di default, che deve essere specificato nella dichiarazione del'attributo
- Attributo opzionale (**#IMPLIED**)
  - Non c'è un valore di default, se l'utente non indica l'attributo, questo viene ritenuto assente
- Attributo obbligatorio (**#REQUIRED**)
  - L'attributo non può essere omesso, e non c'è alcun valore di default predefinito.
- Attributo con valore fisso non modificabile (**#FIXED**)

Tipi di entità / 1

- General Entity
  - **<!ENTITY abc "The ABC Group">**
  - Richiamato come &abc;
- External General Entity

```
<!ENTITY name [PUBLIC "public-id"] SYSTEM "system-id">
```

  - **<!ENTITY man SYSTEM "/pub/docs/manuals/prod23.htm">**
  - Richiamato come &man;
- Nonparsed External Entity
  - **<!ENTITY logo SYSTEM "images/logo.gif" NDATA gif>**
  - Possono essere usate solo come valori di entità

Tipi di entità / 2

- Parameter Entity
  - `<!ENTITY % paratext "(#PCDATA | emph | acronym)*">`
  - Richiamato come %paramtext;
  - Individua un frammento di DTD
- External Parameter Entity
  - `<!ENTITY % tables SYSTEM "http://www.xmljunk.org/dtds/tables2.1.dtd">`
  - Richiamato come %tables;
  - Individua un frammento esterno di DTD

Esempio Parameter Entities

```
<!ENTITY % content "para | note | warning">
<!ENTITY % id.att "id ID #REQUIRED">
<!ELEMENT chapter (title, epigraph, (%content;)+)>
<!ATTLIST chapter %id.att;>
<!ELEMENT appendix (title, (%content;)+)>
<!ATTLIST appendix %id.att;>
```

Equivale a

```
<!ELEMENT chapter
   (title, epigraph, (para | node | warning)+)>
<!ATTLIST chapter id ID #REQUIRED>
<!ELEMENT appendix
   (title, (para | node | warning)+)>
<!ATTLIST appendix id ID #REQUIRED>
```


Esempio
Versione super-semplificata di XHTML

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html SYSTEM "html.dtd">
<html>
   <h1>Esempio</h1>

   <p>Questo documento rappresenta un esempio
   molto semplice di file <u>XHTML</u>.</p>

   <p><b>Attenzione:</b> oltre a cambiare la
   formattazione dei caratteri, dobbiamo dare
   la possibilit&egrave; di inserire
   <a href="home.html">link</a></p>

   <p>Di tutti gli elementi &egrave; possibile cambiare
   lo <span style="color: blue">stile</span></p>
</html>
```


Esempio
Versione super-semplificata di XHTML

```
<!ELEMENT html ( h1 | p )* >
<!ENTITY % testo "#PCDATA">
<!ELEMENT h1 ( %testo; | a | span )* >
<!ELEMENT p ( %testo; | a | b | u | span )* >
<!ELEMENT a ( %testo; ) >
<!ELEMENT u ( %testo; ) >
<!ELEMENT b ( %testo; ) >
<!ATTLIST a href CDATA #REQUIRED>
<!ELEMENT span ( %testo; ) >
<!ATTLIST span style CDATA #IMPLIED>
<!ENTITY egrave "&#232;" >
<!ENTITY agrave "&#224;" >
```

Documenti XML Validi / Ben Formati

- Documenti XML Ben Formati (well-formed)
  - Rispettano la sintassi di XML
- Documenti XML Validi
  - Sono Ben Formati
  - In più, rispettano la struttura sintattica descritta dal proprio DTD
- Documenti Ben Formati possono *non* essere Validi