



# Linguaggi per il Web: XML

Piero Fraternali – Sara Comai  
Politecnico di Milano

## XML

<http://w3c.org/XML/> – <http://www.microsoft.com/xml/>

- **eXtended Markup Language**
- **Formato di file proposto dal W3C per distribuire documenti elettronici sul World Wide Web**
- **Esempi di documenti elettronici: libri, manuali, cataloghi di prodotti, moduli d'ordine, giornali, formule matematiche, messaggi, ...**

# Evoluzione

- **1986: Standard Generalized Markup Language (SGML) ISO 8879-1986**
- **Novembre 1995: HTML 2.0**
- **Gennaio 1997: HTML 3.2**
- **Agosto 1997: XML Working Draft**
- **Dicembre 1997: XML 1.0 Proposed Recommendation**  
**HTML 4.0 Recommendation**
- **Febbraio 1998: Standard**

# XML

- **HTML: insieme fisso di tag**
- **XML: standard per creare linguaggi di markup con tag personalizzati (erede di SGML); possono essere usati in qualunque dominio**
- **HTML vs XML**

<code>&lt;h1&gt; The Idea Methodology &lt;/h1&gt;</code>	<code>&lt;book&gt;</code>
<code>&lt;ul&gt;</code>	<code>&lt;title&gt;The Idea</code>
<code>&lt;li&gt; di S. Ceri, P. Fraternali</code>	<code>Methodology &lt;/title&gt;</code>
<code>&lt;li&gt; Addison-Wesley</code>	<code>&lt;author&gt; S. Ceri &lt;/author&gt;</code>
<code>&lt;li&gt; US\$ 49</code>	<code>&lt;author&gt; P. Fraternali &lt;/author&gt;</code>
<code>&lt;/ul&gt;</code>	<code>&lt;ed&gt; Addison-Wesley &lt;/ed&gt;</code>
	<code>&lt;price&gt; US\$ 49 &lt;/price&gt;</code>
	<code>&lt;/book&gt;</code>

## **XML vs HTML**

- **XML non rimpiazza HTML!**

**XML e HTML sono nati con scopi diversi:**

- **XML progettato per descrivere DATI**  
→ cosa sono i dati
- **HTML progettato per visualizzare i dati**  
→ come appaiono i dati

## **Uso di XML**

- **Separare i dati dal modo con cui vengono visualizzati**
- **Scambiare i dati tra sistemi incompatibili**
- **Scambiare informazioni in sistemi B2B**
- **Condividere dati**
- **Memorizzare dati**
- **Creare nuovi linguaggi (WML, MathML...)**

## Esempio di documento XML

```
<?xml version="1.0"?>
<elenco>
  <prodotto codice="123kl14">
    <descrizione> Forno </descrizione>
    <prezzo> 1040000 </prezzo>
  </prodotto>
  <prodotto codice="432sd35">
    <descrizione> Frigo </descrizione>
  </prodotto>
</elenco>
```

Attributi

Tag con contenuti

## Sintassi XML

- Tutti gli elementi hanno un tag di chiusura
- I tag sono "case sensitive"
- I tag devono essere annidati correttamente
- Un documento XML deve avere un tag radice
- Gli spazi nel documento vengono preservati
- è un documento di testo → il software che tratta documenti testuali tratta anche XML

## Elementi

```
<prodotto codice="123kl14">  
  <descrizione> Forno </descrizione>  
  <prezzo> 1040000 </prezzo>  
</prodotto>
```

- Si possono estendere
- Hanno relazioni (padre-figlio)
- Hanno contenuto

## Attributi

```
<prodotto codice="123kl14">  
  <descrizione> Forno </descrizione>  
  <prezzo> 1040000 </prezzo>  
</prodotto>
```

- Gli elementi possono avere degli attributi
- Vanno racchiusi tra “ “
- Differiscono dagli elementi perchè non possono contenere elementi figli

# Attributi

## Problemi nell'uso di attributi

- Non possono contenere valori multipli
- Non sono facilmente estendibili
- Non possono descrivere strutture
- Sono difficilmente manipolabili da programmi

Vanno bene per memorizzare metadati

# Namespace

- Metodo per evitare conflitti di nome

```
<table>  
  <tr>...</tr>  
</table>
```

```
<table>  
  <product>...</product>  
</table>
```

# Namespace

- Si introduce un prefisso

```
<h:table>  
  <h:tr>...</h:tr>  
</h:table>
```

```
<my:table>  
  <my:product>...</my:product>  
</my:table>
```

# Tipi di marcature

- Elementi: <prodotto>
- Entità: &lt; (sta per <), &#8478; (Unicode)
- Commenti: <!-- qualsiasi testo -->
- Istruzioni: <? Nome-istruzione dati ?>
- Sezioni CDATA (character data)

```
<![CDATA[  
  *p = &q;  
  b = (i <= 3);  
]]>
```

## **Validazione di un documento XML**

- **Un documento XML la cui sintassi è corretta è detto “well-formed”**
- **Un documento validato rispetto ad un DTD è detto valido**

## **Document Type Definition (DTD)**

- **Definisce il tipo di un documento, cioè:**
  - i tag ammessi
  - le regole di annidamento dei tag
- **Esempio di dichiarazione di un elemento:**  
**<!ELEMENT PRODOTTO (DESCRIZIONE, PREZZO)>**
- **L'elemento prodotto contiene al suo interno un elemento descrizione seguito da un elemento prezzo**

## Modello di contenuto

- Elementi contenenti elementi figli

<!ELEMENT PRODOTTO (DESCRIZIONE)>

– <prodotto> <descrizione>...</descrizione></prodotto>

- Elementi con PCDATA (parsed character data = brano di testo qualunque)

<!ELEMENT DESCRIZIONE #PCDATA>

– <descrizione> testo </descrizione>

- Elementi vuoti

<!ELEMENT ARTICOLO EMPTY>

– <articolo/>

## Modello di contenuto

- Contenuto misto

<!ELEMENT ARTICOLO (#PCDATA | PRODOTTO)>

– <articolo> testo </articolo>

– <articolo><prodotto>..</prodotto><articolo>

- Qualsiasi contenuto

<!ELEMENT PARTE ANY>

– <parte><sottoparte></sottoparte><parte>

– <parte><prodotto></prodotto></parte>

## Occorrenze di un elemento

- **1 volta**

```
<!ELEMENT PRODOTTO (DESCRIZIONE)>  
  <prodotto>  
    <descrizione>...</descrizione>  
  </prodotto>
```

- **1 o più volte**

```
<!ELEMENT LISTA (PRODOTTO+)>  
  <lista>  
    <prodotto>..</prodotto>  
    <prodotto>..</prodotto>  
  </lista>
```

## Occorrenze di un elemento

- **0 o più volte**

```
<!ELEMENT LISTA (PRODOTTO*)>
```

```
- <lista>  
  <prodotto>..</prodotto>  
  <prodotto>..</prodotto>  
  </lista>
```

```
- <lista></lista>
```

- **0 o 1 volta**

```
<!ELEMENT PRODOTTO (DESCRIZIONE?)>  
- <prodotto>    <descrizione>...</descrizione></prodotto>  
- <prodotto></prodotto>
```

## Esempio di DTD

```
<!ELEMENT ELENCO (PRODOTTO+)>
<!ELEMENT PRODOTTO (DESCRIZIONE, PREZZO?)>
<!ELEMENT DESCRIZIONE #PCDATA>
<!ELEMENT PREZZO #PCDATA>
```

```
<elenco>
  <prodotto codice="123kl14">
    <descrizione> Forno </descrizione>
    <prezzo> 1040000 </prezzo>
  </prodotto>
  <prodotto codice="432sd35">
    <descrizione> Frigo </descrizione>
  </prodotto>
</elenco>
```

## Dichiarazioni di attributi

- Per ogni elemento dice:
  - quali attributi può avere il tag
  - che valori può assumere ciascun attributo
  - qual è il valore di default
- Esempio di dichiarazione di attributo:

```
<!ATTLIST PRODOTTO
  codice ID #REQUIRED
  label CDATA #IMPLIED
  status (disponibile | terminato) 'disponibile'>
```
- Il tag PRODOTTO può contenere 3 attributi

## Tipi di attributi

- **CDATA:** stringa
- **ID:** identificatore
- **IDREF, IDREFS:** valore di un attributo di tipo ID nel documento (o insieme di valori)
- **ENTITY, ENTITIES:** nome (nomi) di entità
- **NMTOKEN, NMTOKENS:** caso ristretto di CDATA (una sola parola o insieme di parole)

codice	ID	#REQUIRED
label	CDATA	#IMPLIED
status	(disponibile   terminato)	'disponibile'

## Vincoli sugli attributi

- **#REQUIRED:** il valore deve essere specificato
- **#IMPLIED:** il valore può mancare
- **#FIXED “valore”:** se presente deve coincidere con “valore”
- **“valore”:** il valore può non essere specificato, nel qual caso si assume “valore” come default

codice	ID	#REQUIRED
label	CDATA	#IMPLIED
status	(disponibile   terminato)	'disponibile'

## Dichiarazioni di entità

- Analoghe alle dichiarazioni di macro con **#define in C** – esempio:

```
<!ENTITY ATI "ArborText, Inc.">
<!ENTITY boilerplate SYSTEM "/standard/legalnotice.xml">
<!ENTITY ATIllogo SYSTEM "/standard/logo.gif"
    NDATA GIF87A>
```

- Le entità possono essere
  - interne (&ATI;)
  - esterne (&boilerplate; &ATIllogo;)
  - parametriche (utilizzabili solo nei DTD)

## Documenti con DTD

```
<?XML version="1.0" standalone="no"?>
```

```
<!DOCTYPE capitolo SYSTEM "libro.dtd" [
  <!ENTITY %alink.module "IGNORE">
  <!ELEMENT ulink (#PCDATA)*>
  <!ATTLIST ulink
    xml:link      CDATA #FIXED "SIMPLE"
    xml-attributes CDATA #FIXED "HREF URL"
    URL           CDATA #REQUIRED>
```

```
>
```

```
<capitolo>...</capitolo>
```

DTD esterno

DTD  
interno

## Perchè i DTD

- **Con i DTD i documenti contengono anche la descrizione della loro struttura**
- **Diverse persone possono accordarsi sul formato dei documenti tramite il DTD**
- **Utile per la validazione di un documento XML (ricevuto da terzi oppure proprio)**

## HTML e XML

- **HTML è un caso particolare di XML**
  - è possibile scrivere un DTD per XML
  - lo stile sintattico è leggermente diverso
  - è in uscita una revisione di HTML per uniformarsi a XML (XHTML)
- **Il concetto URL di HTML diventa URI (Uniform Resource Identifier)**
  - un documento XML può far riferimento ad oggetti diversi (DTD, documenti XML, etc.)
  - è più importante identificare un oggetto che localizzarlo

# **XML e i browser**

- **Supporto da Internet Explorer 5.0 in poi**
  - **Visualizzazione**
  - **Supporto DTD**
  - **Embedded in documenti HTML**
  - **Trasformazione con XSL**
  - **Visualizzazione con CSS**
- **Supporto completo in futuro anche in Netscape**