DTD

ALBERTO BELUSSI

ANNO ACCADEMICO 2012/2013

Document Type Definition (DTD)

- Un DTD è il linguaggio usato per descrive la struttura di un documento XML disponibile prima che fosse introdotto XMLSchema
- I DTD vengono utilizzati per la validazione di un documento XML (come accade per i file XSD)

Esempio di DTD

```
<!ELEMENT BookStore (Book+)>
<!ELEMENT Book (Title, Author, Date, ISBN, Publisher)>
<!ATTLIST Book
   Category (autobiography|non-fiction|fiction) #REQUIRED
   InStock (true | false) "false"
   Reviewer CDATA " ">
<!ELEMENT Title (#PCDATA)>
<!ELEMENT Author (#PCDATA)>
<!ELEMENT Date (#PCDATA)>
<!ELEMENT ISBN (#PCDATA)>
<!ELEMENT Publisher (#PCDATA)>
```

BookStore.dtd

Ogni riga rappresenta la dichiarazione di un tipo di elemento:

- L'elemento *BookStore* contiene uno o più elementi *Book*
- L'elemento *Book* contiene una lista di elementi (struttura): *Title, Author, Date, ISBN, Publisher*
- L'elemento *Book* contiene anche attributi: *Category, InStock* e *Reviewer*
- *Title, Author, Date, ISBN, Publisher* possono contenere solo testo

Dichiarazione di elementi

```
<!ELEMENT nome_elemento (modello_di_contenuto)>
```

Il modello di contenuto può essere:

- `#PCDATA`
- Elementi figli
- Sequenze di elementi figli
- Misto
- `EMPTY`
- `ANY`

Modello di contenuto: #PCDATA

Specifica che l'elemento deve contenere solamente dati di tipo carattere:

L'elemento non può contenere elementi figli di alcun tipo

Esempio: L'elemento `Title` deve contenere solo testo

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

Modello di contenuto: Elementi Figli

Specifica che un elemento deve contenere esattamente un elemento figlio di un determinato tipo

Esempio: L'elemento `Book` deve contenere esattamente un elemento `Title`

```
<!ELEMENT Book(Title)>
```

Modello di contenuto: Sequenze di elementi

Specifica che un elemento deve contenere più elementi figli:

- I figli vengono elencati in una sequenza separati da virgole
- Gli elementi figli devono apparire all'interno dell'elemento padre nell'ordine specificato

Esempio: L'elemento `Book` deve contenere un elemento `Title` e un elemento `Author`

```
<!ELEMENT Book (Title, Author)>
```

Il numero di figli

Per indicare quante istanze di un elemento possono apparire nella struttura di un altro elemento si usano i seguenti metasimboli:

- ? : zero o una istanza
- * : zero o più istanze
- + : una o più istanze

Esempio: L'elemento `Book` deve contenere un elemento `Title`, uno o più elementi `Author` e zero o un elemento `Date`:

```
<!ELEMENT Book (Title, Author+, Date?)>
```

Scelte

Una scelta è un elenco di nomi di elementi (due o più) che possono apparire nell'elemento padre

- Gli elementi della scelta vengono separati da barre verticali
- L'elemento padre non può contenere entrambi gli elementi elencati nella scelta

Esempio: L'elemento `Contatto` può contenere o un elemento `telefono_casa` o un elemento `telefono_ufficio` e non entrambi.

```
<!ELEMENT contatto (telefono_casa |
          telefono_ufficio)>
```

Uso delle parentesi

Per combinare scelte e sequenze si possono usare le parentesi

Esempi

- L'elemento `indirizzo` deve contenere un elemento a scelta tra `via` e `piazza` e a seguire un elemento `civico`

```
<!ELEMENT indirizzo ((via | piazza),
                     civico)>
```

- L'elemento `persona` può contenere un elemento `nome` e un elemento `cognome` oppure un elemento `cognome` e un elemento `nome`

```
<!ELEMENT persona ((nome, cognome) |
                   (cognome, nome))>
```

N.B.: conta l'ordine!

Modello di contenuto: Misto

Specifica che un elemento deve contenere sia dati di tipo carattere che elementi figli. Non è possibile specificare:

- l'ordine in cui appariranno
- quante istanze di essi appariranno
- l'elemento #PCDATA deve essere il primo della lista

Esempio: L'elemento `libro` può contenere dati di tipo carattere e elementi figli `titolo` e `prezzo`

```
<!ELEMENT libro
(#PCDATA|titolo|prezzo)*>
```

Modello di contenuto: EMPTY

Specifica che un elemento deve essere vuoto e quindi senza nessun tipo di contenuto

Esempio: L'elemento immagine deve essere un elemento vuoto

```
<!ELEMENT immagine EMPTY>
```

Modello di contenuto: ANY

Specifica che un elemento può contenere qualsiasi cosa:

- Testo
- Elementi figli
- Contenuto misto

Gli elementi che appaiono come figli devono comunque esser stati dichiarati

Esempio: L'elemento pagina può contenere qualsiasi cosa

```
<!ELEMENT pagina ANY>
```

Dichiarazione di attributi

```
<!ATTLIST nome_elemento
  nome_attributo1 CDATA #REQUIRED
  nome_attributo2 CDATA #IMPLIED
  nome_attributo3 CDATA #FIXED valore
  nome_attributo4 CDATA Literal>
```

Esempio: L'elemento immagine ha un attributo `codice` obbligatorio ed un attributo `titolo` opzionale.

```
<!ATTLIST immagine
codice CDATA #REQUIRED
titolo CDATA #IMPLIED
Category CDATA "Fiction">
```

Cardinalità degli attributi e altre varianti

- `#IMPLIED`: il valore dell'attributo è opzionale
- `#REQUIRED`: il valore dell'attributo è obbligatorio
- `#FIXED`: il valore dell'attributo è costante e immutabile
- `Literal`: indica il valore di default sotto forma di stringa tra apici

Tipi degli attributi

- CDATA
- NMTOKEN
- MTOKENS
- Enumerazione
- ID
- IDREF
- IDREFS
- ENTITY
- ENTITIES
- NOTATION

Tipi di Attributi (1)

- CDATA: può contenere qualsiasi tipo di stringa accettabile in un documento XML ben formato:

```
<!ATTLIST immagine titolo CDATA #IMPLIED>
    <immagine titolo="tramonto"/>
```

- NMTOKEN: può iniziare con qualsiasi carattere:

```
<!ATTLIST libro anno_pubblicazione NMTOKEN
#REQUIRED>
    <libro anno_pubblicazione="1950 d.c."/>
```

- NMTOKENS: può contenere uno o più token

```
<!ATTLIST esibizione date NMTOKENS
#IMPLIED>
    <esibizione date="10-07-2004 17-07-2004 24-07-2004"/>
```

Tipi di Attributi (2)

Enumerazione: lista di tutti i possibili valori assegnabili all'attributo:

```
<!ATTLIST data
  giorno
  (1|2|3|4|5|6|7|8|9|10|11|12|13|14|15|16|17|18|19|20|
  21|22|23|24|25|26|27|28|29|30|31) #REQUIRED
  mese
  (gennaio|febbraio|marzo|aprile|maggio|giugno|luglio|
  agosto|settembre|ottobre|novembre|dicembre)
  #REQUIRED
  anno (2003|2004|2005|2006|2007) #REQUIRED>


<data giorno="15" mese="agosto" anno="2007"/>
```

Tipi di Attributi (3)

- `ID`: contiene un nome XML che ha valore univoco all'interno del documento

```
<!ATTLIST persona cod_fisc ID
#REQUIRED>
```

```
<persona cod_fisc="RSSMRA65E25L781T"/>
```

- `IDREF`: riferimento all'attributo di tipo ID di un elemento del documento

<!ATTLIST persona cod_fisc ID #REQUIRED>

<!ATTLIST docente persona IDREF #REQUIRED>

```
<persona cod_fisc="RSSMRA65E25L781T"/>
<docente persona="RSSMRA65E25L781T"/>
```

Tipi di Attributi (4)

- `IDREFS`: contiene una lista di nomi XML ognuno dei quali deve essere un ID valido di un elemento del documento

```
<!ATTLIST persona cod_fisc ID #REQUIRED>
<!ATTLIST sposi persone IDREFS #REQUIRED>
```

```
<persona cod_fisc="RSSMRA65E25L781T"/>
<persona cod_fisc="BNCFRN63D45L781T"/>
<sposi persone="RSSMRA65E25L781T
                BNCFRN63D45L781T"/>
```

Validazione

Un documento XML per il quale è richiesta la validazione deve includere un riferimento al DTD con cui deve essere messo a confronto

- Il riferimento deve essere fornito nella dichiarazione del tipo di documento

```
<!DOCTYPE elenco SYSTEM
"http://ibiblio.org/xml/dtds/elenco.dtd">
```

- Questa dichiarazione afferma che `elenco` è la radice del documento e il DTD si trova all'URL http://ibiblio.org/xml/dtds/elenco.dtd
- La dichiarazione del tipo di documento si trova dopo la dichiarazione XML