XML

XML e Metadati

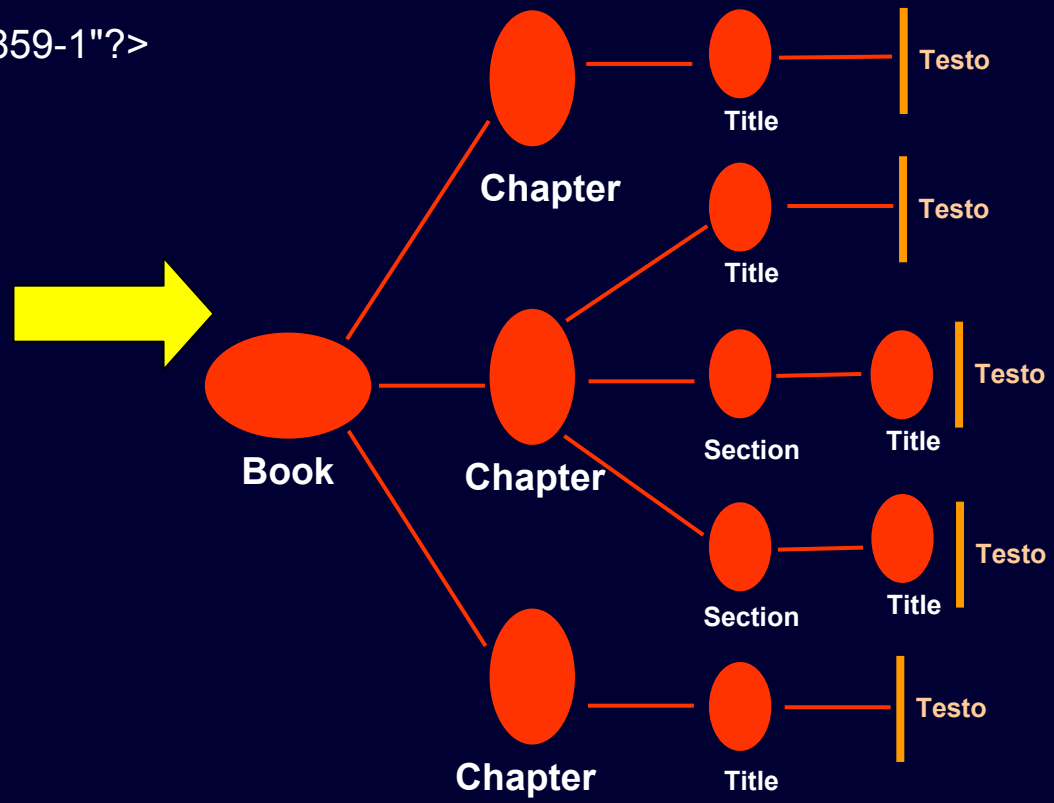
- XML (eXtensible Markup Language)
- XML viene usato per descrivere i dati
- I “tag” XML non sono predefiniti come in HTML
- XML viene spesso utilizzato come linguaggio per descivere i metadati

Un semplice esempio

XML Declaration (“questo è XML”)

Codifica dei caratteri usata

```
<?xml version="1.0" encoding="iso-8859-1"?>
<book>
    <chapter>
        <title>Introduction</title>
    </chapter>
    <chapter>
        <title>Story</title>
        <Section>
            <title>Part 1</title>
        </Section>
        <Section>
            <title>Part 2</title>
        </Section>
    </chapter>
    <chapter>
        <title>Index</title>
    </chapter
</book>
```



Organizzazione gerachica dei dati

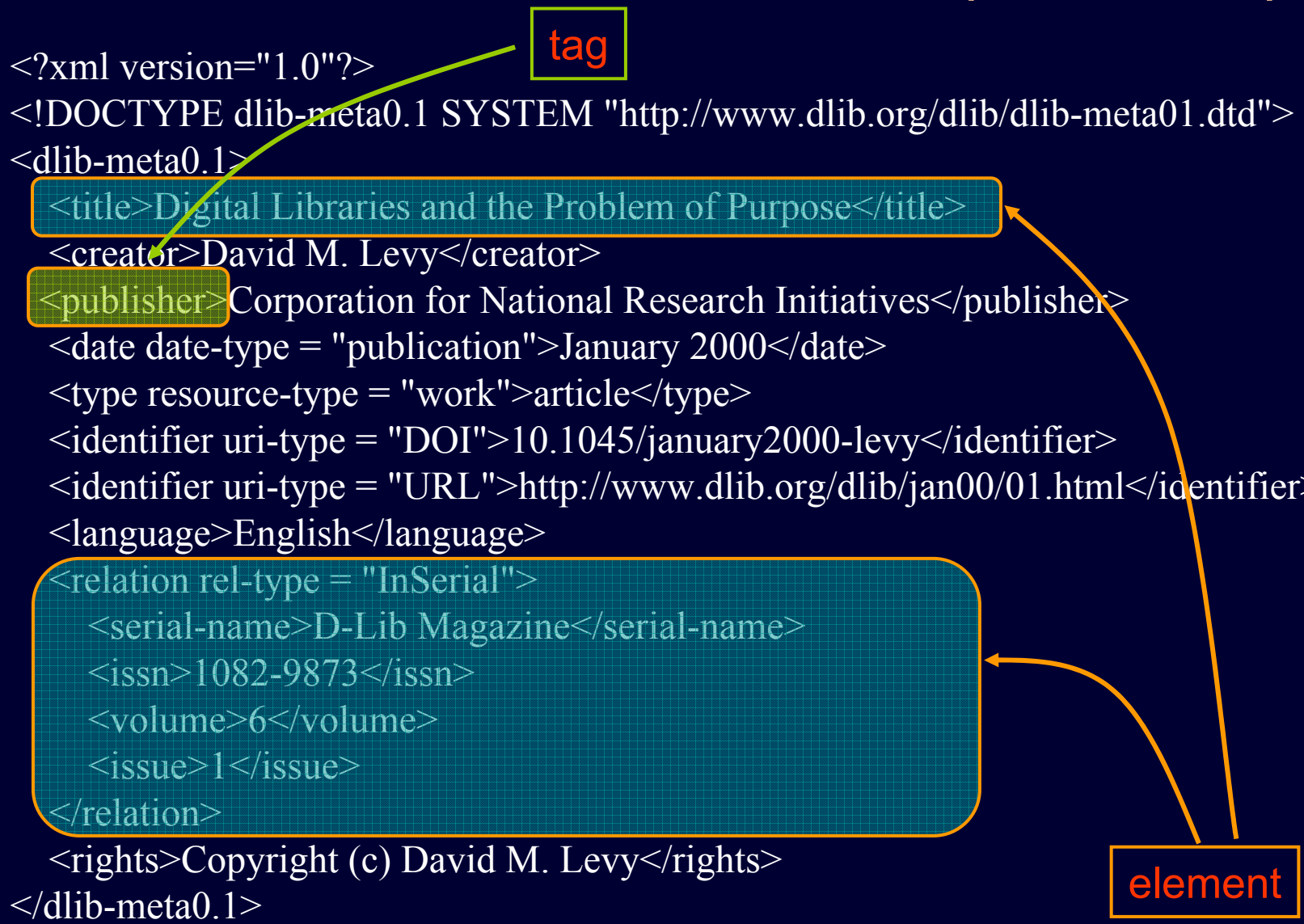
XML elements

- **Costituenti principali dei documenti XML**
- **Possono contenere testo, attributi o altri elementi**

```
<name att1 = "val1" att2 = "val2">
    contenuto
<\name>
```

- **Regole sui nomi degli Elementi:**
  - I nomi possono contenere lettere, numeri ed altri caratteri
  - Non possono iniziare con numeri o caratteri di punteggiatura
  - Non possono iniziare con "XML"
  - Non possono contenere spazi
    - ➔ **<first_name>, <last_name>.**

Esempio documento XML (Metadata)



Attributi

- **Gli attributi forniscono informazione aggiuntiva sugli elementi**

```
<date date-type = "publication">January 2000</date>
```

  - Ad es. gli attributi possono associare una label univoca all'elemento, oppure possono descrivere una proprietà dell'elemento.

- **Sintassi**

```
Name = "value"
```

- **Un elemento può avere qualunque numero di attributi, ma ogni attributo ha una sola occorrenza**

```
<team persona="Mario" persona="Anna">
<team persone="Mario Anna">
<team persona1="Mario" persona2="Anna">
<team>
   <persona>Mario<\persona>
   <persona>Anna<\persona>
<\team>
```

Forma errata

Forme corrette

Utilizzo elementi

Esempio documento XML (Metadata)

Attributo dell'elemento date

```
<?xml version="1.0"?>
<!DOCTYPE dlib-meta0.1 SYSTEM "http://www.dlib.org/dlib/dlib-meta01.dtd">
<dlib-meta0.1>
   <title>Digital Libraries and the Problem of Purpose</title>
   <creator>David M. Levy</creator>
  <publisher>Corporation for National Research Initiatives</publisher>
   <date date-type = "publication">January 2000</date>
   <type resource-type = "work">article</type>
   <identifier uri-type = "DOI">10.1045/january2000-levy</identifier>
   <identifier uri-type = "URL">http://www.dlib.org/dlib/jan00/01.html</identifier>
   <language>English</language>
   <relation rel-type = "InSerial">
      <serial-name>D-Lib Magazine</serial-name>
      <issn>1082-9873</issn>
      <volume>6</volume>
      <issue>1</issue>
   </relation>
   <rights>Copyright (c) David M. Levy</rights>
</dlib-meta0.1>
```

Namespaces [1/2]

- **Meccanismo usato per identificare diversi "spazi" dei nomi in XML e combinarli nello stesso documento**
  - Nomi di ***elementi*** o ***attributi***
- **È un modo per identificare diversi *dialetti*, che hanno una particolare semantica e/o modalità di elaborazione.**
  - Ad es. namespace dei termini matematici
  - Viene usato anche per inserire delle direttive di visualizzazione (XSLT) del documento XML
- **Ad esempio <title> può avere significati diversi in contesti diversi, oppure `<key>` può indicare una chiave di protezione (in un contesto di gestione della sicurezza) oppure una chiave d'accesso (nel contesto di un database)**
- **Un namespace consiste di un gruppo di elementi e di nomi di attributi. I nomi del namespace vengono identificati utilizzando un prefisso**
  - `ns-prefix:local-name`
- **Un namespace deve essere dichiarato prima di poterlo utilizzare**
  - `Xmlns:name ="url"`

Namespaces [2/2]

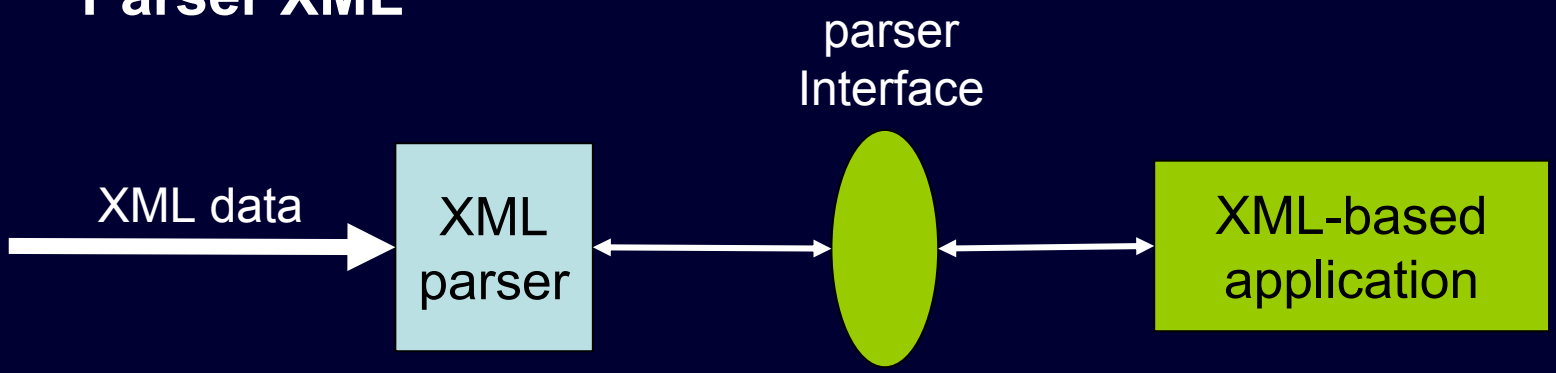
```
<?xml version= "1.0" encoding= "utf-8" ?>

<html xmlns="http://www.w3.org/1999/xhtml1"
      xmlns:mt="http://www.w3.org/1998/mathml" >
<head>
   <title> Title of XHTML Document </title>
</head><body>
<div class="myDiv">
    <h1> Heading of Page </h1>
     <mt:mathml>
          <mt:title> ... MathML markup . . .
     </mt:mathml>
     <p> more html stuff goes here </p>
</div>
</body>
</html>
```

Default 'space' è *xhtml*

mt: prefisso che indica lo 'spazio' *mathml*

Elaborazione di documenti XML

Parser XML



- **Il parser verifica che il file XML sia sintatticamente corretto**
- **Questi dati sono detti** *well-formed*
  - È la condizione minima per essere considerati in formato XML
- **Il parser deve terminare l'elaborazione se i dati non sono well-formed**
  - Per es. Termina l'elaborazione e lancia un'eccezione all'applicazione.

Definizione di modelli di documenti XML

- **In XML è possibile creare il proprio modello di markup, specificando quali devono essere gli elementi e gli attributi dei documenti**
- **Vi sono due modi per definire un modello di documento XML**
  - **XML Document Type Declaration (DTD)** – Fa parte della specifica di XML
  - **XML Schema (anche detto XSD) –** Permette di specificare vincoli più precisi sui documenti XML.
- **Il DTD e XML schema permettono di specificare quali sono gli elementi permessi ed i nomi degli attributi, le regole di composizione gerachica degli elementi, e le restrizioni di tipo o di contenuto degli elementi**

Validazione dei documenti XML

- Documenti XML "Well Formed“
  - Un documento XML "Well Formed" ha una sintassi XML corretta
- Documenti XML "Validi“
  - Un documento XML "Valido" è un documento XML "Well Formed", che è anche conforme alle regole di un **Document Type Definition** (DTD) o di un **XML Schema Definition** (XSD).
- **Il processo di elaborazione termina con un errore se il documento XML non è valido.**

XML DTD

- Il DTD definisce la struttura del documento con una lista di elementi legali
- Si può avere una dichiarazione del DTD (dichiarazione DOCTYPE)

Interna

```
<?xml version="1.0"?>
<!DOCTYPE note
[ <!ELEMENT note (to,from,heading,body)>
  <!ELEMENT to (#PCDATA)>
  <!ELEMENT from (#PCDATA)>
  <!ELEMENT heading (#PCDATA)>
  <!ELEMENT body (#PCDATA)>
]>
  <note>
        <to>Jo</to>
        <from>Mary</from>
         <heading>Reminder</heading>
         <body>Don't forget me this weekend</body>
  </note>
```

Esterna

```
<?xml version="1.0"?>
<!DOCTYPE note SYSTEM "note.dtd">
<note>
          <to>Jo</to>
          <from>Jani</from>
      <heading>Reminder</heading>
      <body>Don't forget me this
        weekend!</body>
 </note>
--- note.dtd ---
<!ELEMENT note (to,from,heading,body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
```

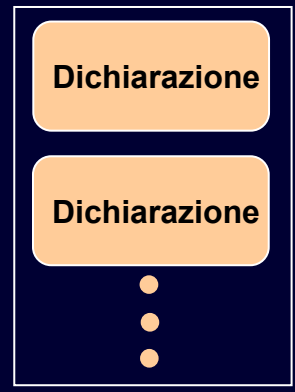
Perché usare un DTD

- **Usando i DTD ogni documento contiene una descrizione del proprio formato.**
- **Gruppi di utenti possono accordarsi sull'uso di DTD comuni per facilitare lo scambio dei documenti.**
- **Le applicazioni possono usare un DTD standard per verificare che i dati ricevuti dall'esterno sono validi.**
- **Si possono utilizzare i DTD per verificare che i dati prodotti abbiano la struttura corretta.**

Sintassi del DTD [1/8]

- **Un DTD definisce**
  - L'insieme degli elementi permessi
    - ➔ **Questo si può considerare come il "vocabolario" del linguaggio**
  - Il "modello del contenuto" di ogni elemento
    - ➔ **Specifica quali elementi o dati possono essere inclusi in un elemento, in quale ordine, in quale numero, e se sono obbligatori o opzionali.**
  - Specifica l'insieme di attributi permessi per ogni elemento
    - ➔ **Ogni dichiarazione definisce il nome, il tipo del dato, i valori di default (se ci sono) ed il comportamento dell'attributo**

Sintassi del DTD [2/8]



- **Un DTD è composto da un insieme di dichiarazioni**
- **Dichiarazioni di Elementi**
  - Per ogni elemento che si userà nel documento si **deve** avere una dichiarazione
  - `<!ELEMENT name content-model>`
  - `Name` è il nome dell'elemento (ad es. titolo, persona, ecc.)
  - Il `content-model` permette di specificare quale tipo di contenuto può essere incluso nell'elemento, quanti possono essere i suoi elementi e in che ordine vanno inseriti

Sintassi del DTD [3/8]

- **Vi sono 5 diversi tipi di content model**
  - Elementi EMPTY
    - `<!ELEMENT graphic EMPTY>`
  - Elementi con nessuna restrizione sul contenuto
    - **L'elemento può contenere qualunque altro elemento**
    - **Poco utile in un DTD reale, si usa nella fase di sviluppo**
    - `<!ELEMENT esempio ALL>`
  - Elementi che contengono solo dati di tipo carattere
    - `<!ELEMENT esempio (#PCDATA)>`
  - Elementi che contengono solo altri elementi
    - **Usa una sintassi particolare per specificare i vincoli sugli elementi**
    - `<!ELEMENT articolo (para+)>`
    - `<!ELEMENT articolo (tit, (para | sez)+)>`
    - `<!ELEMENT articolo (tit, sottotit?, ((para+,sez*) | sez+))>`
  -

+ una o più ripetizioni

*** Zero o più ripetizioni**

? opzionale

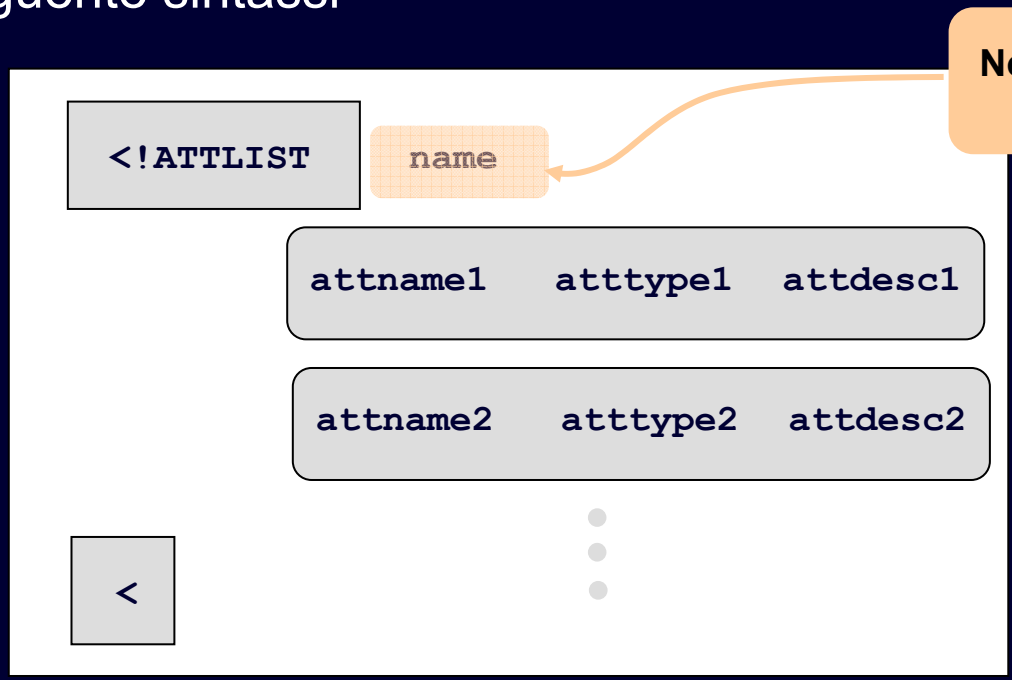
| richiede uno dei due elementi

Sintassi del DTD [4/8]

- Elementi con contenuto misto
  - ➔ **Possono contenere sia elementi che caratteri**
  - ➔ `<!ELEMENT para (#PCDATA | subpara*)>`
  - ➔ `<!ELEMENT articolo (titolo, autore*, (para | tabella | lista)+, bibliografia?>`
    - – **L'elemento `articolo` contiene**
      - > **Un `titolo`**
      - > **Zero o più `autori`**
      - > **Una o più occorrenze di `para, tabella, lista`**
      - > **Una `bibliografia` opzionale**
      - > **Ad es. `titolo, autore, autore, para, list, bibliografia` è un elemento articolo corretto**
  - ➔ **I simboli * + | ? hanno lo stesso significato del caso precedente**

Sintassi del DTD [5/8]

- **Dichiarazioni di attributi**
  - Per ogni elemento vengono dichiarati tutti i suoi attributi con la seguente sintassi



Sintassi del DTD [6/8]

- **Una dichiarazioni di attributi**
  - Da un nome all'attributo
  - Specifica il tipo di dato dell'attributo
  - Descrive il comportamento dell'attributo (ad es. se ha un valore di default, o se l'autore deve specificare un valore)

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

L'attributo id è di tipo ID ed è obbligatorio

Security può assumere due valori, "high" è di default

Keywords è opzionale e non ha valori di default

Sintassi del DTD [7/8]

- **I `datatype` degli attributi sono**
  - CDATA: caratteri
  - NMTOKEN (NMTOKENS): stringa di caratteri (lista di stringhe)
  - ID: identificatore unico
  - IDREF (IDREFS): riferimento ad un identificatore
  - ENTITY (ENTITIES): accetta un nome di ENTITA' come valore
  - Enumeration value list: lista di keywords

Sintassi del DTD [8/8]

- **Comportamenti degli attributi**
  - Assegnamento di valori di default
    - ➔ `<!ATTLIST message`
      `importance (high | medium | low) "medium" >`
  - #IMPLIED: attributo opzionale e non vi è alcun valore di dafault
  - #REQUIRED: deve essere fornito un valore
  - #FIXED: il valore è prefissato e non può essere modificato

Esempio documento XML (Metadata)

```
<?xml version="1.0"?>
<!DOCTYPE dlib-meta0.1 SYSTEM "http://www.dlib.org/dlib/dlib-meta01.dtd">
<dlib-meta0.1>
    <title>Digital Libraries and the Problem of Purpose</title>
    <creator>David M. Levy</creator>
   <publisher>Corporation for National Research Initiatives</publisher>
    <date date-type = "publication">January 2000</date>
    <type resource-type = "work">article</type>
    <identifier uri-type = "DOI">10.1045/january2000-levy</identifier>
    <identifier uri-type = "URL">http://www.dlib.org/dlib/jan00/01.html</identifier>
    <language>English</language>
    <relation rel-type = "InSerial">
        <serial-name>D-Lib Magazine</serial-name>
        <issn>1082-9873</issn>
        <volume>6</volume>
        <issue>1</issue>
    </relation>
    <rights>Copyright (c) David M. Levy</rights>
</dlib-meta0.1>
```

Il DTD del D-Lib Magazine [1/6]

<!-- DTD to mark up the metadata elements in D-Lib Magazine -->
<!-- William Y. Arms, Cathy Rey, March 8, 1999 Updated June 16, 1999 -->

<!ELEMENT dlib-meta0.1 (title, creator+, publisher, date, type, identifier+, language*, relation, rights+)>

<!-- Element names are from the Dublin Core set of 15 names. -->
<!-- Attributes are used to clarify the usage by D-Lib Magazine. -->

Continua nella prossima slide

Il DTD del D-Lib Magazine [2/6]

```
<!ELEMENT title (#PCDATA)>
<!--  Title as supplied with all punctuation -->
<!ELEMENT creator (#PCDATA)>

<!--  This element is repeated for each author or other creator -->
<!--  It contains the name of the author as provided, -->
<!--  without affiliation or contact information.  -->

<!ELEMENT publisher (#PCDATA)>

<!-- Publisher is "Corporation for National Research Initiatives" -->
```

Continua nella prossima slide

Il DTD del D-Lib Magazine [3/6]

```
<!ELEMENT date (#PCDATA)>
<!ATTLIST date
        date-type CDATA #FIXED "publication">

<!--  Issue date, e.g., "July 1995", or "July/August 1998" -->

<!ELEMENT type (#PCDATA)>
<!ATTLIST type
        resource-type CDATA #FIXED "work">

<!--  D-Lib Magazine assigns metadata to works -->
<!--  The default type is an "article" -->
```

Continua nella prossima slide

Il DTD del D-Lib Magazine [4/6]

```
<!ELEMENT identifier (#PCDATA)>
<!ATTLIST identifier
        uri-type (DOI | URL) #REQUIRED>

<!--  Every work should have a single DOI and one or more URLs. -->
```

Continua nella prossima slide

Il DTD del D-Lib Magazine [5/6]

```
<!ELEMENT relation (serial-name, (issn, volume, issue)*)>
<!ATTLIST relation
        rel-type CDATA #FIXED "InSerial">
   <!ELEMENT serial-name (#PCDATA)>
   <!ELEMENT issn (#PCDATA)>
   <!ELEMENT volume (#PCDATA)>
   <!ELEMENT issue (#PCDATA)>

<!--  The serial name is "D-Lib Magazine".  -->
<!--  The ISSN is "1082-9873".  -->
<!--  Volume corresponds to year of publication, 1995 is "1".  -->
<!--  The issue is a count of the actual issues in the volume.  -->
```

Continua nella prossima slide

Il DTD del D-Lib Magazine [6/6]

<!ELEMENT language (#PCDATA)>

<!-- The name of the language in English as: "English", "French, "Japanese" -->

<!ELEMENT rights (#PCDATA)>

<!-- The copyright statement as given on the work. -->

XML Schema

- **Un XML schema descrive la struttura di un documento XML.**
  - Definisce gli elementi che possono apparire nel documento
  - Definisce gli attributi che possono apparire nel documento
  - Definisce la struttura degli elementi
    - ➔ **Relazioni padre – figlio**
    - ➔ **Numero dei figli**
    - ➔ **Ordine dei figli**
  - Definisce se un elemento è vuoto oppure può includere del testo
  - Definisce il tipo dei dati degli elementi e degli attributi
  - Definisce i valori di default e i valori prefissati per gli elementi e gli attributi

Visualizzazione di documenti XML

- **Si utilizzano gli stylesheet**
  - XML non usa tag predefiniti per cui il significato di ogni tag non è noto: un browser non sà come presentare un documento XML
  - Il documento XML non include informazione di formattazione
  - È necessario fornire informazione aggiuntiva che specifica come il documento deve essere visualizzato
- **Il Cascading Style Sheets (CSS) è un semplice meccanismo per aggiungere gli stili (per es. Font, colori, spaziatura) ai documenti web.**
- **XSL - Extensible Stylesheet Language viene utilizzato per specificare gli stylesheet.**

Riferimenti

- **W3C http://www.w3.org/XML/**
- **Erik T. Ray, Learning XML, O'Reilly, 2001**
- **XML Tutorial**
  - http://www.w3schools.com/xml/default.asp
  - http://www.zvon.org/xxl/XMLTutorial/General/contents.html