

# Tutorial 6 - XML DTD

---

CSC343 - Introduction to Databases  
Fall 2008

TA: Lei Jiang

## XML DOCTYPE Internal DTD

---

**XMLDocument Type Declaration - define your own XML markup language**

**(1) <!DOCTYPE root [DTD]>**

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE person [
  <!ELEMENT person (name)+>
  <!ELEMENT name (first, last)>
  <!ELEMENT first (#PCDATA)>
  <!ELEMENT last (#PCDATA)>
]>
<person>
  <name>
    <first>John</first>
    <last>Smith</last>
  </name>
</person>
```

# XML DOCTYPE External DTD

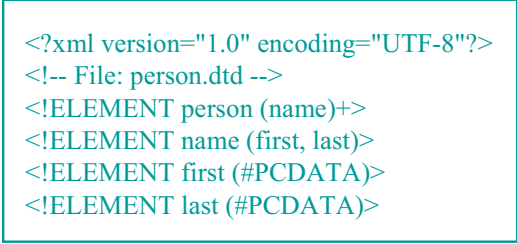
---

## (2) <!DOCTYPE root SYSTEM URL>

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?>
<!DOCTYPE person SYSTEM "person.dtd">
<person>
  <name>
    <first>John</first>
    <last>Smith</last>
  </name>
</person>
```

or using absolute URL

```
<!DOCTYPE person SYSTEM "http://www. .../person.dtd">
```



```
<?xml version="1.0" encoding="UTF-8"?>
<!-- File: person.dtd -->
<!ELEMENT person (name)+>
<!ELEMENT name (first, last)>
<!ELEMENT first (#PCDATA)>
<!ELEMENT last (#PCDATA)>
```

# XML DOCTYPE External & Internal

---

## (4) <!DOCTYPE root SYSTEM URL [DTD]

>

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE person SYSTEM "person.dtd"
  [<!ATTLIST person sex (male|female) "male">]
>
<person>
  <name>
    <first>John</first>
    <last>Smith</last>
  </name>
</person>
```

# Element Declarations

---

**<!ELEMENT name content>**

**content:** EMPTY, ANY, (#PCDATA), child element, mixed,

**(1) EMPTY:** The element is declared to be an empty one - no content allowed.

```
<!ELEMENT empty_element EMPTY>
```

```
e.g., <empty_element></empty_element>
```

```
e.g., <empty_element />
```

**(2) ANY:** The element can have any element or character data.

```
<!ELEMENT any_element ANY>
```

```
e.g., <any_element>
```

```
    This is a line of characters.
```

```
    <other_element> ... </other_element>
```

```
</any_element>
```

## Element Declarations (cont'd)

---

**(3) (#PCDATA):** The content can be only character data.

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

```
e.g., <first> John </first>
```

**(4) mixed:** The content may contain character data and/or child elements.

```
<!ELEMENT mix_element (#PCDATA | first | last)>
```

```
e.g., <mix_element>
```

```
    My first name is
```

```
    <first>John</first>
```

```
    and my last name is
```

```
    <last>smith </last>
```

```
</mix_element>
```

# Element Declarations (cont'd)

---

**(5) child elements:** The content can only contain the child elements (no character data). The sequence, alternative, and cardinality can be expressed using commas(,), pipes(|) and modifiers.

e.g. `<!ELEMENT a (x,y,z)>`: element a must have an element x, followed by y, followed by z.

e.g. `<!ELEMENT b (x|y|z)>`: element b must have an element x, or y, or z

Modifier: asterisk(\*) stands for zero or more.  
plus sign(+) stands for one or more.  
question mark(?) stands for zero or one.

e.g. `<!ELEMENT c (x*, y+, z?) >`

# Attribute List Declarations

---

**Define which attributes may be associated with a particular element. An attribute can have a name, a type, whether optional, required or fixed, and possibly a default.**

**`<!ATTLIST element_name  
attribute_name attribute_type default_declaration>`**

**attribute\_type**

**String:** CDATA

e.g. `<!ATTLIST student id CDATA #IMPLIED>`

**Enumerated:** (value1 | value2 | value3)

e.g. `<!ATTLIST car color (red|black|blue|white) #REQUIRED>`

**ID:** An ID type attribute must contain a value which is unique within the XML document. e.g., student id shown above.

**IDREF:** An IDREF type attribute refers to the ID type attribute of another element in the document. It is often used to create relationships between elements.

e.g. `<!ATTLIST student sid ID #REQUIRED>  
<!ATTLIST course cid ID #REQUIRED>  
<!ATTLIST enrolment sid IDREF #REQUIRED  
cid IDREF #REQUIRED ... >`

# Attribute List Declarations `attribute_type`

---

## default\_declaration

`#REQUIRED`, `#IMPLIED`, `#FIXED`, default value

**#REQUIRED:** Every occurrence of element must have this attribute.

```
<!ATTLIST person id ID #REQUIRED>
```

**#IMPLIED:** The attribute is optional.

```
<!ATTLIST person salary CDATA #IMPLIED>
```

**#FIXED:** The attribute must always have this value.

```
<!ATTLIST person language CDATA #FIXED "EN">
```

- The document is not valid if attribute language contains a value different from "EN".
- If element doesn't contain the attribute, the default value "EN" will be used.

**default value:** The attribute may or may not appear in the element. If not, using this value.

```
<!ATTLIST person contract (true|false) 'false'>
```

# Entity Declarations `Internal Entities`

---

**Internal Entities:** associates a name with a text string.

e.g.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!DOCTYPE person [
  <!ELEMENT person ((name)+, university)>
  <!ELEMENT name (first, last)>
  <!ELEMENT first (#PCDATA)>
  <!ELEMENT last (#PCDATA)>
  <!ELEMENT university (#PCDATA)>
  <!ENTITY ut "university of toronto">
]>
<person>
  <name>
    <first>John</first>
    <last>Smith</last>
  </name>
  <university>&ut;</university>
</person>
```

# Entity Declarations External Entities

---


**External Entities:** associates a name with the content of another file.

e.g.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE person [
  <!ELEMENT person ((name)+, university)>
  <!ELEMENT name (first, last)>
  <!ELEMENT first (#PCDATA)>
  <!ELEMENT last (#PCDATA)>
  <!ELEMENT university (#PCDATA)>
  <!ENTITY UofT SYSTEM "ut.xml">
]>
<person>
  <name>
    <first>John</first>
    <last>Smith</last>
  </name>
  &UofT;
</person>
```

File: ut.xml

```
<university>
  University of Toronto
</university>
```



# Entity Declarations Parameter Entities

---

**Parameter Entities can only be used within the DTD.**

Declaration: **<!ENTITY %name "replacement text">**

Usage: **%name;**

e.g.,

```
<!ENTITY %week_attr "(Mon|Tue|Wed|Thu|Fri|Sat|Sun)">
<!ELEMENT day EMPTY>
<!ATTLIST day today %week_attr; "Mon">
<!ENTITY anyday %week_attr;>
```

# Entity Declarations

## Predefined Entities

---

**&lt;** produces a less than <  
**&gt;** produces a greater than >  
**&amp;** produces an ampersand &  
**&apos;** produces an apostrophe '  
**&quot;** produces a quotation mark "

## CDATA

---

**Content in a CDATA section is not processed by the XML parser.**

**<![CDATA [ content ] ]>**

e.g.,

```
<![CDATA[
  if (this->getValue() < 3 && value[1] !=3 )
    cout << "error message"
]]>
```

If we don't use CDATA, we have to write as following:

```
if (this->getValue() &lt; 3 && value[1] != 3)
  cout &lt;&lt; &quot;error message&quot;
```

# Stylesheet Processing Instruction

---

(1) Associate CSS (Cascading Style Sheets):

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<?xml-stylesheet type="text/css" href="book.css"?>
<!DOCTYPE bookDatabase SYSTEM "book.dtd">
<bookDatabase>
  ...
</bookDatabase>
```

(2) Associate XSL (eXtensible Stylesheet Language)

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<?xml-stylesheet type="text/xsl" href="book.xsl"?>
<!DOCTYPE bookDatabase SYSTEM "book.dtd">
<bookDatabase>
  ...
</bookDatabase>
```

```
<?xmlversion="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE bookDatabaseSYSTEM "book.dtd">
<bookDatabase>
  <book bookID="ISBN-0-13-xxx">
    <author>Elliote Rusty Harold</author>
    <content>
      <chapter number="1">XML Structure, Syntax</chapter>
      <description> <section>XML Fundamentals</section> </description>
    </content>
    <content>
      <chapter number="2">Document Type Definitions</chapter>
      <description>
        <section>Element Declarations</section>
        <section>Attributes Declarations</section>
        <summary>summary of the chapter 2</summary>
      </description>
    </content>
  </book>
  <book bookID="ISBN-0-586-xxx">
    <author>Ian Foster</author> <author>Carl Kesselman</author>
    <image>grid architecture</image>
    <content>
      <chapter number="1">Grids in Context</chapter>
      <description />
    </content>
    <newchapters>
      <chapter number="20">Network structure</chapter>
      <description> <section>The Future(1998-2005)</section></description>
    </newchapters>
  </book>
</bookDatabase>
```

*Exercise: write the book DTD for which this is a valid XML document*



# A Complete Example DTD

---

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- file: book.dtd -->
<!ELEMENT bookDatabase (book+)>
<!ENTITY %CH "(chapter, description)">
<!ELEMENT book (author+, image*, content+, newchapters*)>
<!ATTLIST book bookID ID #REQUIRED>
<!ELEMENT author (#PCDATA)>
<!ELEMENT image (#PCDATA)>
<!ELEMENT content %CH;>
<!ELEMENT newchapters %CH;>
<!ATTLIST newchapters added (true|false) "false">
<!ELEMENT chapter (#PCDATA)>
<!ATTLIST chapter number CDATA #REQUIRED>
<!ELEMENT description (section*, summary?)>
<!ELEMENT section (#PCDATA)>
<!ELEMENT resources (#PCDATA)>
<!ELEMENT summary (#PCDATA)>
<!ENTITY EH "Elliotte Rusty Harold">
<!ENTITY IF "Ian Foster">
<!ENTITY CK "Carl Kesselman">
```