INTERNET & WEB APPLICATION DEVELOPMENT SWE 444

Fall Semester 2008-2009 (081)

Module 4 (II): Data Description and Transformation, XML

Dr. El-Sayed El-Alfy
Computer Science Department
King Fahd University of Petroleum and Minerals
alfy@kfupm.edu.sa

Objectives/Outline

- Objectives
  - Learn about the role of XML DTD and Schema
  - Learn how to view XML
  - Learn about embedding XML data in HTML documents
  - Learn about MSXML parser
  - Learn how to convert a relational data into XML
  - Identify XML related technologies
- Outline
  - XML DTD & Schema
  - Viewing XML Files
  - Displaying XML with CSS
  - Displaying XML with XSL
  - XML Data Islands
  - MSXML
  - Namespaces
  - Relational Data to XML
  - XML Related Technologies

XML DTD & Schema

- A "Valid" XML document is a "Well Formed" XML document, which also conforms to the rules of a Document Type Definition (DTD)
- A DTD defines the documents structure with a a list of legal elements
- XML Schema is an XML based alternative to DTD
- Errors in XML documents will stop the XML program
  - Unlike HTML browsers, the W3C XML specification states that a program should stop processing an XML document if it finds an error.
    - Because XML software should be small, fast, and compatible.
- XML Validator helps check the syntax of XML files
  - http://www.w3schools.com/xml/xml_validator.asp

XML DTD & Schema (cont.)

- An XML file with an embedded DTD

```
<?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>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<message>Don't forget me this weekend!</message>
</note>
```

XML DTD & Schema (cont.)

The XML file

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE note SYSTEM "Note.dtd">
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

Note.dtd

```
<!DOCTYPE note [
    <!ELEMENT note (to, from, heading, body)>
    <!ELEMENT to (#PCDATA)>
    <!ELEMENT from (#PCDATA)>
    <!ELEMENT heading (#PCDATA)>
    <!ELEMENT body (#PCDATA)>
]>
```


XML DTD & Schema (cont.)

XML Schema

```
<xs:element name="note">

<xs:complexType>
    <xs:sequence>
        <xs:element name="to" type="xs:string"/>
        <xs:element name="from" type="xs:string"/>
        <xs:element name="heading" type="xs:string"/>
        <xs:element name="body" type="xs:string"/>
    </xs:sequence>
 </xs:complexType>

</xs:element>
```

Viewing XML Files

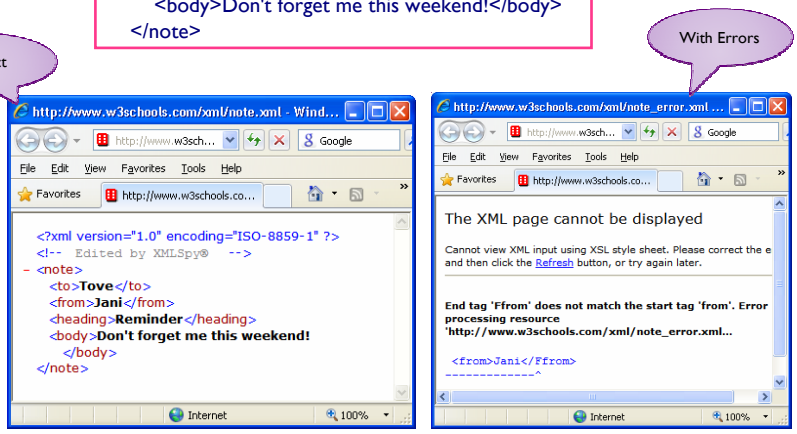
- Raw XML files can be viewed in all major browsers such as IE 5.0 +, Netscape 6.0 +
  - Don't expect XML files to be displayed as HTML pages.
  - To make it display like a web page, you have to add some display information
- XML documents do not carry information about how to display the data
- Different solutions to the display problem
  - using CSS, XSL, JavaScript, and XML Data Islands

Viewing XML Files (cont.)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
- <note>
      <to>Tove</to>
      <from>Jani</from>
      <heading>Reminder</heading>
      <body>Don't forget me this weekend!</body>
  </note>
```

Corect

With Errors



Browsers Support for XML

- Netscape 6 supports XML
- IE 5.0 supports the XML 1.0 standard; it has the following XML support:
  - Viewing of XML documents
  - Displaying XML with CSS
  - Transforming and displaying XML with XSL
  - XML embedded in HTML as Data Islands
  - Binding XML data to HTML elements
  - Access to the XML DOM
  - Full support for W3C DTD standards

Displaying XML with CSS

- With CSS, you can add display information to an XML document
- Formatting XML with CSS is NOT the future of the Web
- Formatting with XSL will be the new standard

Example

the xml file

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/css" href="cd_catalog.css"?>
<CATALOG>
    <CD>
        <TITLE>Empire Burlesque</TITLE>
        <ARTIST>Bob Dylan</ARTIST>
        <COUNTRY>USA</COUNTRY>
        <COMPANY>Columbia</COMPANY>
        <PRICE>10.90</PRICE>
        <YEAR>1985</YEAR>
    </CD>
    <CD>
        <TITLE>Hide your heart</TITLE>
        <ARTIST>Bonnie Tyler</ARTIST>
        <COUNTRY>UK</COUNTRY>
        <COMPANY>CBS Records</COMPANY>
        <PRICE>9.90</PRICE>
        <YEAR>1988</YEAR> </CD>
. . . .
</CATALOG>
```

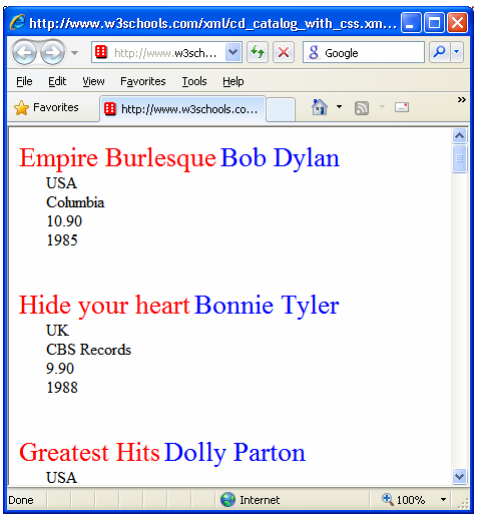

Example (cont.)

the css file

```
CATALOG
        { background-color: #ffffff; width: 100%; }

CD
        { display: block; margin-bottom: 30pt; margin-left: 0; }

TITLE
        { color: #FF0000; font-size: 20pt; }

ARTIST
        { color: #0000FF; font-size: 20pt; }

COUNTRY,PRICE,YEAR,COMPANY
        { Display: block; color: #000000; margin-left: 20pt; }
```

Example (cont.)

- Viewed by the browser



Displaying XML with XSL

- XSL stands for eXtensible Stylesheet Language
- With XSL you can add display information to your XML document
- XSLT is the recommended style sheet language of XML
- XSL is far more sophisticated than CSS
- One way to use XSL is to transform XML into HTML before it is displayed by the browser
  - Either by the browser
  - Or at the server

Example

the xml file

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/xsl" href="simple.xsl" ?>
<breakfast_menu>
      <food>
            <name>Belgian Waffles</name>
            <price>$5.95</price>
            <description>two of our famous Belgian Waffles with plenty of real maple syrup</description>
            <calories>650</calories>
      </food>
      <food>
            <name>Strawberry Belgian Waffles</name>
            <price>$7.95</price>
            <description>light Belgian waffles covered with strawberries and whipped cream</description>
            <calories>900</calories>
      </food>
      …
      </breakfast_menu>
</breakfast_menu>
```

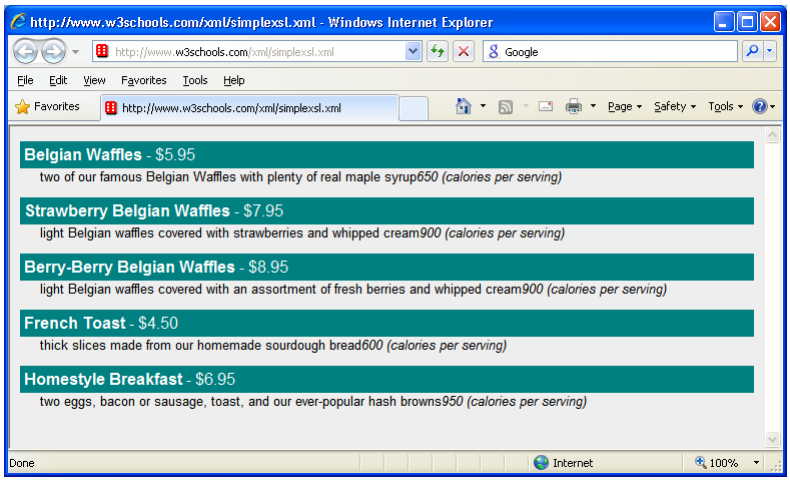

Example (cont.)

the xsl file

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<html xsl:version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns="http://www.w3.org/TR/xhtml1/strict">
  <body style="font-family:Arial,helvetica,sans-serif;font-size:12pt; background-color:#EEEEEE">
    <xsl:for-each select="breakfast_menu/food">
      <div style="background-color:teal;color:white;padding:4px">
        <span style="font-weight:bold;color:white">
        <xsl:value-of select="name"/></span>
        - <xsl:value-of select="price"/>
      </div>
      <div style="margin-left:20px;margin-bottom:1em;font-size:10pt">
        <xsl:value-of select="description"/>
        <span style="font-style:italic">
          (<xsl:value-of select="calories"/> calories per serving)
        </span>
      </div>
    </xsl:for-each>
  </body>
</html>
```

Example (Cont.)

- View the result in IE 8



XML Embedded in HTML

- XML can be embedded within HTML pages in Data Islands
  - The unofficial <xml> tag is used to embed XML data within HTML

```
<html>
    <body>

        <xml id="note" src="note.xml"></xml>

    </body>
</html>
```

- Manipulated via client side script or data binding
- The next step is to format and display the data in the data island by binding it to HTML elements.

Internet Explorer - XML Data Islands

- Data Islands can be bound to HTML elements (like HTML tables)

```
<html>
<body>
<xml id="cdcat" src="cd_catalog.xml"></xml>
<table border="1" datasrc="#cdcat">
<tr>
<td> <span datafld="ARTIST"> </span> </td>
<td> <span datafld="TITLE"> </span> </td>
</tr>
</table>
</body>
</html>
```

- An XML data island with ID "cdcat" is loaded from an external file XML file
- An HTML table is bound to the data Island with a data source attribute
- The table data elements are bound to the XML data with a data field attribute inside a span

Viewed by IE

Bob Dylan	Empire Burlesque
Bonnie Tyler	Hide your heart
Dolly Parton	Greatest Hits
Gary Moore	Still got the blues
Eros Ramazzotti	Eros
Bee Gees	One night only
Dr.Hook	Sylvias Mother
Rod Stewart	Maggie May
Andrea Bocelli	Romanza
Percy Sledge	When a man loves a woman
Savage Rose	Black angel
Many	1999 Grammy Nominees
Kenny Rogers	For the good times
Will Smith	Big Willie style
Van Morrison	Tupelo Honey

The Microsoft XML Parser (MSXML))

- To read and update an XML document, you need an XML parser
- The MSXML parser comes with Microsoft IE 5.0
- Once you have installed IE 5.0, the parser is available to scripts inside HTML documents & ASP files
- The parser features a language-neutral programming model that supports:
  - JavaScript, VBScript, Perl, VB, Java, C++ and more
  - W3C XML 1.0 and XML DOM
  - DTD and validation
- You can create an XML document object with the following code:

```
var xmlDoc=new ActiveXObject("Microsoft.XMLDOM")
```


Loading an XML file into the parser

- The following code loads an XML document (note.xml) into the XML parser:

```
<script type="text/javascript">
var xmlDoc = new
ActiveXObject("Microsoft.XMLDOM")
xmlDoc.async="false"
xmlDoc.load("note.xml")
// ... processing the document goes here
</script>
```

  - The second line in the code above creates an instance of the MSXML parser
  - The third line turns off asynchronous loading, to make sure that the parser will not continue execution before the document is fully loaded
  - The fourth line tells the parser to load the XML document called note.xml

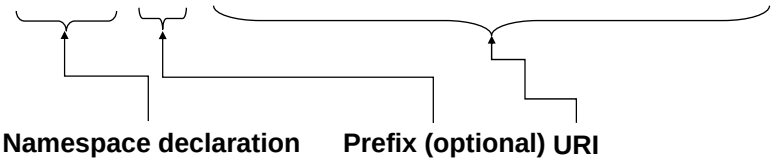
Namespaces: Overview

- Part of XML's extensibility
- Allow authors to differentiate between tags of the same name (using a prefix)
  - Frees author to focus on the data and decide how to best describe it
  - Allows multiple XML documents from multiple authors to be merged
- Defined by a W3C recommendation called Namespaces in XML
  - http://www.w3.org/TR/REC-xml-names/
- W3Schools Tutorial
  - http://www.w3schools.com/XML/xml_namespaces.asp

Namespaces: Declaration

Namespace declaration examples:

```
xmlns: bk = "http://www.example.com/bookinfo/"

xmlns: bk = "urn:mybookstuff.org:bookinfo"

xmlns: bk = "http://www.example.com/bookinfo/"
```



any element or attribute names that start with the prefix "bk" are considered to be in the XHTML namespace

URI is a string of characters which identifies an Internet Resource (Uniform Resource Identifier) >> (URL, URN)

Namespaces: Examples

```
<BOOK xmlns:bk="http://www.bookstuff.org/bookinfo">
   <bk:TITLE>All About XML</bk:TITLE>
   <bk:AUTHOR>Joe Developer</bk:AUTHOR>
   <bk:PRICE currency='US Dollar'>19.99</bk:PRICE>
```

```
<bk:BOOK xmlns:bk="http://www.bookstuff.org/bookinfo"
       xmlns:money="urn:finance:money">
   <bk:TITLE>All About XML</bk:TITLE>
   <bk:AUTHOR>Joe Developer</bk:AUTHOR>
   <bk:PRICE money:currency='US Dollar'>
       19.99</bk:PRICE>
```


Namespaces: Default Namespace

- An XML namespace declared without a prefix becomes the default namespace for all sub-elements
- All elements without a prefix will belong to the default namespace:

```
<BOOK xmlns="http://www.bookstuff.org/bookinfo">
   <TITLE>All About XML</TITLE>
   <AUTHOR>Joe Developer</AUTHOR>
```

Namespaces: Scope

- Unqualified elements belong to the inner-most default namespace.
  - BOOK, TITLE, and AUTHOR belong to the default book namespace
  - PUBLISHER and NAME belong to the default publisher namespace

```
<BOOK xmlns="www.bookstuff.org/bookinfo">
   <TITLE>All About XML</TITLE>
   <AUTHOR>Joe Developer</AUTHOR>
   <PUBLISHER xmlns="urn:publishers:publinfo">
      <NAME>Microsoft Press</NAME>
   </PUBLISHER>
</BOOK>
```


Namespaces: Attributes

- Unqualified attributes do NOT belong to any namespace
  - Even if there is a default namespace
- This differs from elements, which belong to the default namespace

Entities

- Entities provide a mechanism for textual substitution, e.g.

<	produces the left angle bracket	**<**
>	produces the right angle bracket	**>**
&	produces the ampersand	**&**
'	produces a single quote character	**'**
"	produces a double quote character	**"**

- You can define your own entities
- Parsed entities can contain text and markup
- Unparsed entities can contain any data
  - JPEG photos, GIF files, movies, etc.

Example

```
<!DOCTYPE videocollection [
      <!ENTITY R "Romance">
      <!ENTITY WAR "War">
      <!ENTITY COM "Comedy">
      <!ENTITY SF "Science Fiction">
      <!ENTITY ACT "Action">
  ]>
```

CDATA

- By default, all text inside an XML document is parsed
  - PCDATA - Parsed Character Data
- But text inside a CDATA section will be ignored by the parser.
  - CDATA - (Unparsed) Character Data
  - <![CDATA[…]]>
- Characters like "<" and "&" are illegal in XML elements
  - To avoid errors script code can be defined as CDATA.
- Whitespace inside a CDATA is (usually) preserved
- The only real restriction is that the character sequence]]> cannot occur inside a CDATA
- CDATA is useful when your text has a lot of illegal characters (for example, if your XML document contains some HTML text)

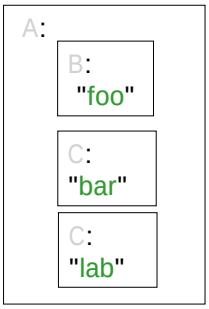
Example

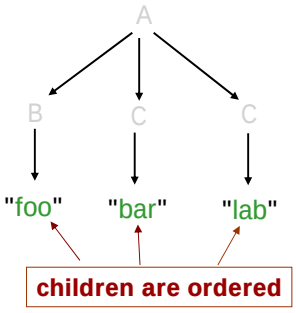
```
<script>
<![CDATA[
function matchwo(a,b)
{
    if (a < b && a < 0) then
    {
       return 1;
    }
    else
    {
       return 0;
    }
}
]]>
</script>
```

Pure XML -- Instance Model

- XML 1.0 Standard:
  - No explicit data model
  - Only syntax of well-formed and valid (wrt. a DTD) documents
- Implicit data model:
  - nested containers ("boxes within boxes")
  - labeled ordered trees (=a semistructured data model)
  - relational, object-oriented, other data: easy to encode

```
<A>
  <B>foo</B>
  <C>bar</C>
  <C>lab</C>
</A>
```

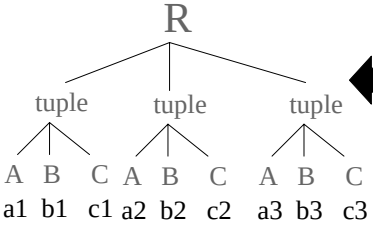




Example: Relational Data to XML

R

A	B	C
a1	b1	c1
a2	b2	c2
a3	b3	c3




```
<R>
   <tuple>
        <A> a1 </A>
        <B> b1 </B>
        <C> c1 </C>
   </tuple>
   <tuple>
        <A> a2 </A>
        <B> b2 </B>
        <C> c2 </C>
   </tuple>
   …
</R>
```

Adding Structure and Semantics

- XML Document Type Definitions (DTDs):
  - define the structure of "allowed" documents (i.e., valid wrt. a DTD)
  - ≈ database schema
  - => improve query formulation, execution, ...
- XML Schema
  - defines structure and data types
  - allows developers to build their own libraries of interchanged data types
- XML Namespaces
  - identify your vocabulary

XML Related Technologies

- XHTML - Extensible HTML
- CSS - Cascading Style Sheets
- XSL - Extensible Style Sheet Language
  - XSL consists of three parts: XML Document Transformation (renamed XSLT, see below), a pattern matching syntax (renamed XPath, see below), and a formatting object interpretation.
- XSLT - XML Transformation
  - XSLT is far more powerful than CSS. It can be used to transform XML files into many different output formats.
- XPath - XML Pattern Matching
  - XPath is a language for addressing parts of an XML document. XPath was designed to be used by both XSLT and XPointer.

XML Related Technologies (cont.)

- XLink - XML Linking Language
  - The XML Linking Language (XLink), allows elements to be inserted into XML documents in order to create links between XML resources.
- XPointer - XML Pointer Language
  - The XML Pointer Language (XPointer), supports addressing into the internal structures of XML documents, such as elements, attributes, and content.
- DTD - Document Type Definition
  - A DTD can be used to define the legal building blocks of an XML document.
- Namespaces
  - XML namespaces defines a method for defining element and attribute names used in XML by associating them with URI references.

XML Related Technologies (cont.)

- DOM - Document Object Model
  - The DOM defines interfaces, properties and methods to manipulate XML documents.
- XSD - XML Schema
  - Schemas are powerful alternatives to DTDs. Schemas are written in XML, and support namespaces and data types.
- XQL - XML Query Language
  - The XML Query Language supports query facilities to extract data from XML documents.
- SAX - Simple API for XML
  - SAX is another interface to read and manipulate XML documents

Q & A

?

References

- Some useful links with examples and other resources:
  - *Internet and World Wide Web How to Program*, 4/e, H. M. Deitel, P. J. Deitel, and A. B. Goldberg, Pearson Education Inc., 2008. Chapters 13.
  - W3 Schools XML Tutorial
    - http://www.w3schools.com/xml/default.asp
  - W3C XML page
    - http://www.w3.org/XML/
  - XML Tutorials
    - http://www.programmingtutorials.com/xml.aspx
  - Online resource for markup language technologies
    - http://xml.coverpages.org/
  - http://xmlfiles.com/