XML Introduction

Prof. B.D. Chaudhary

July 2012

Table of contents

XML

- SGML, HTML, XML, XHTML, ...
- Standard Generalized Markup Language
- HTML: HyperText Markup Language
- XML: eXtensible Markup Language
- Both HTML and XML being subset of SGML

XML (Contd..)

- An open and evolving standard
- Not a Markup language
- A Meta Language
- Managed by World Wide Web Consortium
- W3C: www.w3.org/XML/

XML Applications

- XHTML: An XML compatible HTML
- MathML: A mathematical equation language
- CaXML: For chess data
- VoiceXML: For structuring conversation

XML Applications (Contd..)

- WML: Wireless Markup Language
- QML: Question Markup Language
- WSDL: Web Service Description Language
- ....

Related Standards

- Torrent of acronyms, standards, and rules
- Document Modeling
- DTD: Document Type Definition
- XML Schema: For extending DTD
- RDF: Resource Description Framework
- DOM: Document Object Model

Related Standards (Contd...)

- Addressing, Querying, and Transformation
- XLink: To describe links between resources
- XBase: To provide base URI services
- XInclude: To embed XML documents

Related Standards (Contd...)

- XPointer: To specify paths in URI
- XPath: To locate XML objects
- XSLT: To transform an XML documents

SGML

- Standard Generalized Markup Language
- All-encompassing coding scheme
- ANSI (1986) and ISO (1992)
- DSSSL: Document Style Semantics and Specification language
- HyTime: Hypermedia/Time-based Structuring Language

Motivations for XML

- Convergence of technology cultures
- Different cultures
- Everything-is-a-relation
- Everything-is-an-object
- Everything-is-a-document
- Everything-is-an-agent

Motivations for XML (Contd...)

- Separation of features of data
- Content
- Structure
- Presentation
- Consumer: Human being and m/c

Motivations for XML (Contd...)

- Data model for middle-ware and Web resources
- Meta language to define markup language
- Protocol for document exchange and processing
- Interoperability and integration

XML Features

- Simple and clear syntax
- Unambiguous structure
- Document validation
- Supports staggering number of writing systems and symbols with Unicode
- Easily combined with style-sheets to create formatted documents

XML Core Concepts

- Elements
- Attributes
- Entities
- Processing Instructions

XML Document: Physical View

- XML document or document only
- Two views : Physical and Logical
- Physical: Either Markup or Character Data
- To store contents in one or more les
- Collectively they represent an XML document

XML Document: Logical View

- Logical: Structure of document
- XML document is viewed as tree
- Elements divide the document into its constituent parts
- Elements are organized into a hierarchy

Document Structure (Contd...)

- Root element: Top element of the hierarchy
- Also called Document Element
- Root element defines boundary
- It encloses all the other elements

Document Structure (Contd...)

- Only one root element in a document
- XML documents are commonly stored as text files
- Any text editor may be used to create XML document
- Use extension *.xml* for clarity
- XML document does not contain formatting information

An Example XML Document

```
<!-- "Prolog" -->
<?xml version="1.0"?>
<!DOCTYPE Greeting SYSTEM "wel.dtd">
<!-- End of Prolog -->
<!-- This is the first XML example -->
<Greeting>
    <from>Instructor</from>
    <to>CS 181 Class</to>
    <myGreetings>
        <greetings>Welcome to XML</greetings>
    </myGreetings>
</Greeting>
```

Comments

- Tags: Enclosed between $<>$
- Tags demarcate and label the parts of documents
- Comment start tag: $<!--$
- Comment end tag: $-->$
- May extend several lines

Document Prolog

- XML document: Prolog and Body
- It may hold additional information
- Text encoding, Processing Instructions, and Document Type Definition being used
- Ordering between prolog and body significant

XML Declaration

- Syntax: <?xml name1 = “value1” name2 = “value2” ...
- Three properties can be set
- version, encoding, and standalone
- <?xml version=“1.0” encoding=“US-ASCII” standalone=“yes”?>
- Note single and double quotes enclosing values

XML Declaration (Contd...)

- All properties are optional
- Property names must be in lowercase
- Values must be quoted: single or double
- Desirable to include version

XML Character Set

- ASCII: 7 bit coding
- ISO: 8 bit coding
- ISO-8859-1,-2,-3,..
- XML document may contain: carriage return, line feed, and Unicode characters

XML Character Set (Contd...)

- The Unicode consortium: www.unicode.org
- Unicode: 16 bit code
- Encodes major scripts of world
- Universal Character System(UCS): 32 bits
- ISO-10646: Lower 2 bytes are Unicode

Character Encoding

- Each subset for a script
- This subset for a script is mapped to 8 bit code
- The characters are in same order, but starts at lower number
- The mapped subset is known as character encoding
- Most common encoding scheme: UTF-8
- UTF-8: Efficient to encode documents having ASCII characters

Character Encoding (Contd...)

- UTF-8 is default XML character encoding
- Some common character encodings:
  - US-ASCII, ISO-8859-1, ISO-8859-n
  - ISO-8859-1-Windows-3.1-Latin-1
  - UTF-7, UTF-8, UTF-16
  - ISO-10646-UCS-2, ISO-10646-UCS-4
  - UCS-2 is same as Unicode

Element

- Elements are building blocks
- Elements are organized in an hierarchy
- Hierarchy defines the logical structure
- Elements acts as containers and labels
- Types of the elements are differentiated by tags
- Start-tag, End-tag, Empty-tag

Element (Contd...)

- Empty-element tag: <name/>
- Start-tag must have a matching end tag
- XML IS CASE SENSITIVE
- <message> ....</Message> : Wrong

Attributes

- Attributes describe elements
- An element may have zero or more attributes
- Attributes are placed within element's start tag
- Only one occurrence of each attribute
- Example: <car doors = "4"/>
- Attribute doors has value "4"

Elements and Attributes Names

- Can be of any length
- Must begin with a letter or an underscore
- May contain letters, digits, underscores, hyphens, and periods
- Names are case sensitive
- Example:
  <instructor Designation="Professor"> Chaudhary
  </instructor>

Reserved Attributes

- xml:lang: Classifies an element by language
- xml:lang="en"
- xml:space: Specifies whether whitespace should be preserved
- xml:space="default"
- xml:link and xml:attribute

White Space Characters

- Spaces, tabs, line feeds and carriage return
- An XML parser is required to pass all characters in a document
- Application need to decide the significance
- Insignificant white spaces may be collapsed into single white space character or removed
- This process is called normalization

Entity References

- Reserved characters: $<$, $>$, & , ‘, “, etc.
- To use these characters in content: Entity References
- Entity references begin with & and ends with ;
- Unicode may be used in document
- &#1583; &#1571; ..

Built-in Entities

- XML provides built-in entities
- &amp;, &lt;, &gt;, &apos;, and &quote;
- Example:<message>&lt;&gt;&quote;</messae>

Entities

- An entity is a placeholder for content
- Declared once and may be used several times
- Does not add anything semantically
- Convenient to read and write XML document
- It represents physical containers such as files or URLs

Entities (Contd...)

- Entities are classified as: Parameter and General
- Parameter entities are used in only DTDs
- General entities: Character, Unparsed, and mixed-content
- Mixed-content: Internal and External
- Character: Predefined, Numbered, and Named

Entities (Contd...)

- An entity consists of a name and a value
- The value may be anything
- Two syntax for entity reference:
- General entities: &name;
- Parameter entities: %name;

An Example of Entities

```
<?xml version="1.0"?>
<!DOCTYPE message SYSTEM "xmldtds/message.dtd" [
<!ENTITY client "Mr. John">
<!ENTITY agent "Ms. Sally">
<!ENTITY phone "<number> 617-555-1299</number>"
]>
<message>
    <opening> Dear &client;</opening>
    <body> We have an exciting .... Pi &#241; ata:..</body>
</message>
```

Character Entities

- Character entities: Contain single character
- Classified into several groups
- Predefined character entities: amp, apos, lt,....
- Numbered character entities: &#231, &#xe7
- Hexadecimal version is distinguished with x

Character Entities (Contd...)

- Named character entities
- They have to be declared
- Easy to remember mnemonic
- Large numbers have been defined in DTDs
- ISO-8879: Standardized set of named character entities

Mixed Content Entities

- Unlimited length
- May include markup as well as text
- Categories: Internal and External
- Internal: Replacement text is defined in the declaration
- &client; &agent; &phone;

Mixed-content Entities (Contd...)

- Predefined character entities must not be used in markup if they are used in entity definition
- Exceptions: quote and apos
- Entities can contain entity reference if it has been declared
- Recursive declaration not permitted

Mixed-content Entities (Contd...)

- External: Replacement text in another file
- Useful for sharing the contents
- Breaks a document into multiple physical parts
- A linking mechanism

Mixed-content Entities (Contd...)

An example:

```
<?xml version="1.0"?>
<!DOCTYPE doc SYSTEM "http://www.dtds.com/generic.dtd" [
<!ENTITY part1 SYSTEM "p1.xml">
<!ENTITY part2 SYSTEM "p2.xml">
<!ENTITY part3 SYSTEM "p3.xml">
]>
<longdoc>
   &part1;
   &part2;
   &part3;
</longdoc>
```

External Entities

- Replacement text is inserted at the time of parsing
- &part1;, &part2;, and &part3; are external entities
- Keyword SYSTEM is used.
- File names are under single or double quotes.
- System identifier: To identify resource location.

External Entities (Contd...)

- Quoted string may be URL
- Alternative is to use keyword PUBLIC
- XML processor will locate the resource
- Public identifier may be accompanied with a system identifier

Unparsed Entities

- Replacement text/content not parsed
- Used to import graphic and sound files
- Keyword NDATA used after system identifier
- NDATA: Notation for data
- It is followed by notation identifier

Unparsed Entities (Contd...)

An Example:

```
<!DOCTYPE doc [
<!ENTITY mypic SYSTEM
"photos/erick.gif" NDATA GIF
]>
< doc >
   &mypic;
</doc>
```

NOTATION

- Notation defines the application to process NDATA
- <!NOTATION Formatname SYSTEM "Appl. Identifier" >
- <!NOTATION TIFF SYSTEM "/program/showtiff.exe" >
- <!NOTATION JPEG SYSTEM "JPG" >
- <!NOTATION " " >

CDATA Section

- CDATA sections are helpful for XML authors
- It contains literal codes
- It is set off by <![CDATA[and]]>
- Everything between $\underline{\text{<![CDATA}}$ and the $\underline{\text{]]>}}$ is treated as raw data
- In CDATA < is not start tag

CDATA Section (Contd...)

- An acronym for “character data”
- An example:
  <para> Then you may say <![CDATA[if(&x < &y)]]> and be done with it.</para>

Processing Instructions

- Presentational information should be kept out of document
- It is container of data
- Targeted toward a specific XML processor
- Contains keyword and target data

Processing Instructions (Contd...)

- <? name data ?>
- Examples:<? flubber pg = 9 recto ?>
  <? things ?>
- <title> The Introduction to the XML <? lb ?> portability <? lb ?> and integratin</title>
- <? lb ?> forces line break

Well-Formed Documents

- An XML parser or processor
- Parses XML documents
- Produce parse tree
- Document is not well-formed if parser is not able to generate tree
- Every XML document must be well formed

Well-Formed Documents (Contd...)

- Some of the syntactic rules checked
  - Case sensitive: message and Message are different tags
  - Single root element
  - A start and end tag for each element
  - Properly nested tags
  - Quoted attribute values
  - Comments and processing instructions may not appear inside tags

Document Validation

- A valid document includes a Document Type Declaration
- The DTD defines rules for the documents
- These are additional constraints
- Validating parser compares documents to their DTDs
- The DTD lists all elements, attributes, and entities the document use and their contexts.