# Managing Information (CSCU9T4)
# Lecture 1: Introduction to XML

Gabriela Ochoa
http://www.cs.stir.ac.uk/~goc/

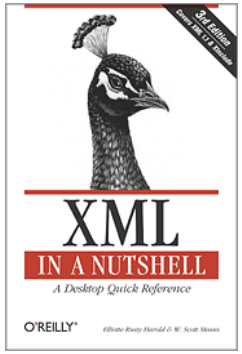
# Outline

- Preliminaries
  - Welcome and introductions
  - Module overview
  - Resources
- What is XML?
- Uses of XML
- XML Structure
- XML Syntax
- Overview of XML supporting technologies
- Summary & What's next?

# Module Overview

- 6 Lectures
  - XML Introduction
  - XML structure (DTD and Schema)
  - Java and XML 1 (SAX)
  - Java and XML 2 (DOM)
  - XML style
  - XML Applications
- 3 Labs

# Resources





- Books
  - **XML in a Nutshell** (2004) by Elliotte Rusty Harold, W. Scott Means, O'Reilly
  - **Beginning XML, 5th Edition** (2012) by Joe Fawcett, Danny Ayers, Liam R. E. Quin
    - Chapter 1 online
- Links and websites
  - XML Tutorial: W3Schools
  - XML.COM O'REALLY
  - XML Validation

# What is XML?

- Designed to describe data, not to display data
- EXtensible Markup Language.
  - **Extensible**: It lets you define your own tags.
  - **Markup**: contains *tags* or elements to provide additional information about the text (similar to HTML tags, but not fixed).
  - **Language**: It is really a *meta-language:* a language that allows us to create or define other languages.
- XML documents are self-describing, and is readable by both humans and software.
- Is a software- and hardware-independent tool for carrying information.
- Became a W3C Recommendation on Feb, 1998.

# What XML is not?

- The XML hype has gotten so extreme that some people expect XML to do everything
- XML is **NOT** a:
  1. **Programming language**: There's no such thing as an XML compiler that reads XML files and produces executable code.
  2. **Network transport protocol**: XML won't send data across the network, any more than HTML will.
  3. **Database**: You're not going to replace an Oracle or MySQL server with XML.
  4. **HTML:** XML is not the replacement for HTML.

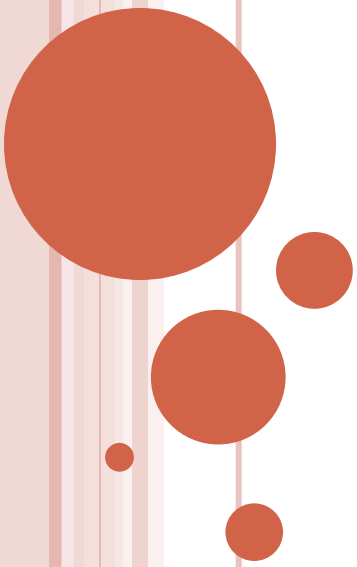
# XML vs HTML

## XML

- EXtensible ML
- Designed to *describe* data
- Focus on what data is
- Carrying information
- Other uses (not only related to web browsing) Such as web services to send requests and responses back and forth

## HTML

- Hyper-Text ML
- Designed to *display* data
- Focus on how data looks
- Displaying information
- Used by Web browsers only (interaction with human)



# Uses/Benefits of XML

- Separate data from HTML
- Simplify data sharing and transport
- Simplify platform changes

# XML Use: Separate data from HTML

- Display dynamic data in an HTML document
- Store data in separate XML files
- Use HTML/CSS for display and layout. So, changes in the underlying data will not require any changes to the HTML.

**Relevant Definitions**

- **Static website:** contains information that does not change. It remains the same, or static, for every viewer of the site.
- **Dynamic website:** contains information that changes, depending on the viewer of the site, the time of the day, the time zone, the native language of the country the viewer, and other factors.
  - **Example**: http://www.bbc.co.uk/weather/2636910

# XML Uses

- Simplify data sharing and transport
  - In practice, computer systems and databases contain data in incompatible formats.
  - Easier to create data that can be shared by different applications.
- Simplify platform changes
  - SW or HW upgrades require converting large amounts of data. Incompatible data is often lost
  - XML is stored in text format. Easier to expand or upgrade to new operating systems, applications or browsers without losing data.
- Makes data more available
  - Different applications can access XML files
  - Data can be read by machines such as handheld computers, voice machines, news feed.

# Simple example

```
<?xml version="1.0"?>
<product barcode="2394287410">
  <manufacturer>Verbatim</manufacturer>
  <name>DataLife MF 2HD</name>
  <quantity>10</quantity>
  <size>3.5"</size>
  <color>black</color>
<description>floppy disks</description>
</product>
```

- XML document of an inventory-control system or a stock database
- Marks up the data with tags describing the colour, size, etc.
- The document is text and can be stored in a text file
- Can be edited with any standard text editor. Although special XML editors do exist.

# XML STRUCTURE

- XML documents form a tree structure that starts at "the root" and branches to "the leaves".

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

**Lines**

1. XML declaration. Defines the XML version (1.0).
2. Root element (saying: "this document is a note"):

3. to 6. Child elements

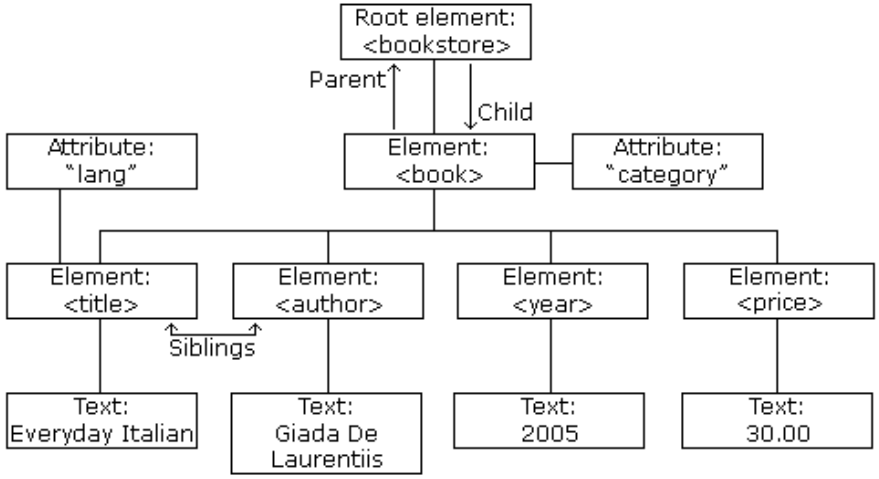
7. End of the root element

# XML TREE STRUCTURE

```
<root>
  <child>
    <subchild>.....</subchild>
  </child>
</root>
```

```
<bookstore>
  <book category="COOKING">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="CHILDREN">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="WEB">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore
```

- Root element:<bookstore>
- The <book> element has 4 children: <title>,< author>, <year>, <price>




http://www.w3schools.com/xml/nodetree.gif

# The XML Declaration

```
<?xml version="1.0" encoding="character encoding" standalone="yes|no"?>
```

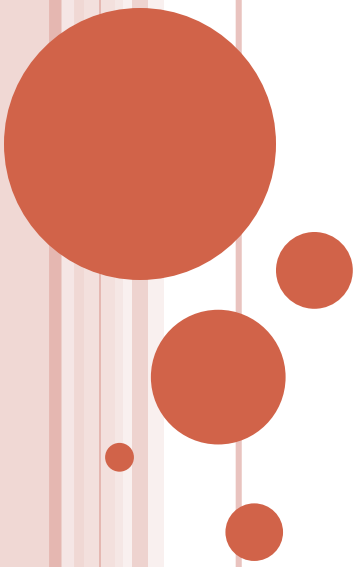
- Declaration is optional, but if provided then must be the first line.
- `version:` Mandatory
- `encoding:` represent the character set,
  - ISO-8859-1 is a standard for plain text much like ASCII
  - UTF-8 is variable-length Unicode that includes plain ASCII, so it is a safe choice for most XML documents
- `standalone:` specifies that an XML document can be read with (or without) reference to external sources

# The XML element

- Everything from (including) the element's start tag to (including) the element's end tag.
- An element can contain:
  - Other elements
  - Text
  - Attributes (provide additional information about an element)
  - A mix of all of the above...

```
<bookstore>
  <book category="CHILDREN">
    <title>Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="WEB">
    <title>Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

- <bookstore> and <book> have element contents
- <book> also has an attribute (category="CHILDREN")
- <title>, <author>, <year>, and <price> have text content



# Well-Formed XML

- XML syntax rules
- XML naming rules
- XML namespaces

# XML SYNTAX RULES

- Documents must have a root element
- All elements must have a closing tag
- Elements must be properly nested
  - Incorrect: `<b><i>This text is bold and italic</b></i>`
  - Correct: `<b><i>This text is bold and italic</i></b>`
- Attribute values must be quoted
  - Incorrect: `<note date=12/11/2007>`
  - Correct: `<note date="12/11/2007">`
- Tags are case sensitive
  - The tag `<Letter>` is different from the tag `<letter>`
- Comments in XML
  - Similar to HTML: `<!-- This is a comment -->`
- XML documents that conform to the syntax rules above are said to be "Well Formed" XML documents.

# XML SYNTAX RULES: ENTITY REFERENCES

- Some characters have a special meaning in XML.
- If you place a character like "<" inside an XML element, it will generate an error because the parser interprets it as the start of a new element.
  - Incorrect: `<message>if salary < 1000 then</message>`
  - Correct: `<message>if salary < 1000 then</message>`
- There are 5 pre-defined entity references in XML:
  - < < less than
  - > > greater than
  - & & ampersand
  - ' ' apostrophe
  - " " quotation mark

# XML NAMING RULES

Element names:

- Are case-sensitive
- Must start with a letter or underscore
- Cannot start with the letters xml (or XML, or Xml, etc.)
- Any other name can be used (except XML)
- Can contain letters, digits, hyphens, underscores, and periods
- Cannot contain spaces

# Best naming practices

- Create descriptive names, like this: <person>, <firstname>, <lastname>.
- Create short and simple names, like this: <book_title> not like this: <the_title_of_the_book>.
- Avoid "-". If you name something "first-name", some software may think you want to subtract "name" from "first".
- Avoid ".". If you name something "first.name", some software may think that "name" is a property of the object "first".
- Avoid ":". Colons are reserved for namespaces (more later).

# Naming styles

There are no naming styles defined for XML elements. But here are some commonly used:

| Style | Example | Description |
|---|---|---|
| Lower case | <firstname> | All letters lower case |
| Upper case | <FIRSTNAME> | All letters upper case |
| Underscore | <first_name> | Underscore separates words |
| Pascal case | <FirstName> | Uppercase first letter in each word |
| Camel case | <firstName> | Uppercase first letter in each word except the first |

If you choose a naming style, it is good to be consistent!

# How XML documents are used


- Input


- Read XML document
- Divides it individual elements, attributes, and other pieces


- Rules about exactly which elements and attributes are allowed where.

# How XML documents are used

The application that receives data from the parser may be:

- A web browser that displays the document
- A word processor, such as Word, loads the document for editing
- A database, such as Microsoft SQL Server, that stores the XML data in a new record
- A drawing program, such as Adobe Illustrator, that interprets the XML as two-dimensional coordinates for the contents of a picture
- A spreadsheet, such as Gnumeric, that parses the XML to find numbers and functions used in a calculation
- A personal finance program, such as Microsoft Money, that sees the XML as a bank statement
- A syndication program that reads the XML document and extracts the head- lines for today's news
- A program that you yourself wrote in Java, C, Python, or some other language that does exactly what you want it to do

# XML Scenarios

- Configuration files: e.g.. Visual Studio project files
- Web services: XML used to serialise objects in a cross platform manner.
- Web content: There's also a lot of content stored as plain XML, which is transformed either server-side or client-side when needed.
- Document management: XML is also used heavily in document-management systems to store and keep track of documents and manage metadata, usually in conjunction with a traditional relational database system.
- Database systems: Most modern high-end database systems, such as Oracle and SQL Server, can store XML documents.

# XML Scenarios

- Image Representation: Vector images can be represented with XML, the SVG format being the most popular. Advantage over a traditional bitmap: images can be manipulated far more easily. Scaling and other changes become transformations of the XML rather than complex intensive calculations
- Business Interoperability: Hundreds of industries now have standard XML formats to describe the different entities that are used in day-to-day transactions, which is one of the biggest uses of XML
  - Medical data
  - Financial transactions such as purchasing stocks and shares and exchanging currency
  - Commercial and residential properties
  - Legal and court records
  - Mathematical and scientific formulas

# XML Technologies

- XML Parsers
  - Before any work can be done with an XML document it needs to be parsed; that is, broken down into its constituent parts
- The Document Object Model (DOM)
  - Language independent Application programming interface (API).
  - Once an XML parser has done its work, it produces an in-memory representation of the XML.
  - To read and manipulate XML document programmatically.
  - Tree-like representation of an XML document. You can start at the tree's root and move to its different branches, extracting or inserting data as you go.
- DTDs and XML Schemas
  - (DTDs) and XML Schemas serve to describe the XML document structure, and what data is allowed where.
  - Used for validation

# XML Technologies

- XML Namespaces
  - Serve as a way of grouping XML names. If one or two different formats need to be used together
- Xpath
  - Used for accessing specific elements or attributes in the document.
  - It works similar to how paths in a file system work, starting at the root and progressing through the various layers until the target is found.
- XLST
  - *Extensible Stylesheet Language Transformations* (XSLT) is powerful way to transform files from one format to another.
  - XSLT is often used to transform XML to (X)HTML, either server-side or in the browser.

# Example XML-Based Languages

- MathML (Mathematics Markup Language) for mathematical expressions. See examples.
- CML (Chemical Markup Language) for describing molecules
- Legal XML for court records
- SVG - Scalable Vector Graphics
- MusicXML
- VoiceXML
- XML format used by MS Office 2007 onwards ('.docx', etc.)
- XHTML – Extensible Hypertext Markup Language (XML-Based HTML) for web pages

# Summary and Next Lectures

- What is XML?
- Uses of XML
- XML Structure
- XML Syntax
- XML Associated technologies
- A document may be well-formed, but is not valid unless its structure conforms to some specification:
  - a DTD (Document Type Definition)
  - an XML Schema