

An Introduction to XML

Ramesh Sankaranarayanan

Department of Computer Science
Australian National University

COMP3410 IT in E-Commerce

Outline

- 1 Introduction
 - Module Outline
 - What is XML?
 - XML vs HTML
 - Why XML?
- 2 History
 - GML, SGML
 - HTML
 - XML
- 3 XML Basics
 - Syntax
 - XML Markup Language Examples
 - MathML
 - XML Parsers

Outline

- 1 Introduction
 - Module Outline
 - What is XML?
 - XML vs HTML
 - Why XML?
- 2 History
 - GML, SGML
 - HTML
 - XML
- 3 XML Basics
 - Syntax
 - XML Markup Language Examples
 - MathML
 - XML Parsers

Module Outline

This module covers:

- XML Basics
- XML Syntax
- XML Validation
- XML Transformation

What is XML?

XML - eXtensible Markup Language

- A language for creating other languages
- Has a well-defined structure
- Can be used to create markup languages like HTML (XHTML)

XML vs HTML

An HTML example

```
<html>
<body>
  <h2>John Doe</h2>
  <p>2 Backroads Lane<br>
  Timbuctoo<br>
  045935435<br>
  john.doe@timbuctoo.com<br>
</p>
</body>
</html>
```

XML vs HTML

This will be displayed as:

John Doe

2 Backroads Lane

Timbuctoo

045935435

john.doe@timbuctoo.com

HTML

- Specifies how the document is to be rendered, and not what information is contained in the document
- Hard for a machine to extract the embedded information. Relatively easy for a human

XML vs HTML

Now, look at the following

```
<contact>
  <name>John Doe</name>
  <address>2 Backroads Lane</address>
  <country>Timbuctoo</country>
  <phone>045935435</phone>
  <email>john.doe@timbuctoo.com</email>
</contact>
```

In this case

- The information contained is being marked, not the rendering
- Content decoupled from presentation
- Readable by both humans and machines

Why XML?

Some of its advantages:

- Open standard
- Extensible
- Transformable
- Large number of evolving tools

Outline

- 1 Introduction
 - Module Outline
 - What is XML?
 - XML vs HTML
 - Why XML?
- 2 History
 - GML, SGML
 - HTML
 - XML
- 3 XML Basics
 - Syntax
 - XML Markup Language Examples
 - MathML
 - XML Parsers

GML and SGML

Generalized Markup Language (GML)

- Charles Goldfarb, Ed Mosher and Ray Lorie, 1969
- Integrated law office information system project. Used as a means of allowing the text editing, formatting and information retrieval subsystems to share documents
- IBM now produces over 90% of its documents with it

GML and SGML

Standard Generalized Markup Language (SGML)

- Charles Goldfarb, 1974
- Extremely powerful, but complex
- Later adapted for use as an all-purpose information standard
- Established as an ISO standard in 1986
- Used quite a lot in the domain of electronic publishing

Hyper Text Markup Language (HTML)

- Tim Berners-Lee and Anders Berglund invented a tag based language for marking up technical documents that a group of scientists in Europe shared over the Internet
- Later expanded to a simplified application of SGML and called HTML
- Language of the web for the rendering of documents
- Has a fixed set of tags and is used primarily for defining how content is to be displayed
- Particular instance of a markup language
- Can't be used to define new markup languages

Extensible Markup Language (XML)

- World Wide Web Consortium (W3C) combined the power of SGML with the simplicity of HTML and came up with XML
- Latest specification standard is XML 1.0 (fifth edition), a W3C Recommendation released on 26 November 2008
- Less than a tenth of the size of the SGML specification
- Has many of SGML's features including extensibility, structure and validation
- Meant to be interoperable with both SGML and HTML

- 1 Introduction
 - Module Outline
 - What is XML?
 - XML vs HTML
 - Why XML?
- 2 History
 - GML, SGML
 - HTML
 - XML
- 3 XML Basics
 - Syntax
 - XML Markup Language Examples
 - MathML
 - XML Parsers

An XML example

```
<?xml version="1.0"?>
<!-- An XML Example -->

<contact>
  <name>John Doe</name>
  <address>2 Backroads Lane</address>
  <country>Timbuctoo</country>
  <phone>045935435</phone>
  <email>john.doe@timbuctoo.com</email>
</contact>
```

In this example:

```
<?xml version="1.0?">
```

is a processing instruction. Conveys useful information to an application. The above says that this is an XML document and uses XML specification version 1.0.

and

```
<!-- An XML Example -->
```

is a comment. You can't use a double hyphen inside a comment.

In this example:

```
<contact> ... </contact>
```

is an **element** whose **type name** is **contact**. Every element has a start tag and an end tag. You can have nested elements.

Elements can have **attributes**

```
<movie type="mystery" rating="R" year="1968">
  The Guns of Navarone
</movie>
```

Here, **type**, **rating** and **year** are attributes.

The combination of elements and attributes, as shown in the examples above, can be used to define the structure of various types of data. In effect, each such definition represents a markup language for a specific type of data.

XML Rules

- Element type names are case sensitive
- Each element must have a starting and ending tag
- Tags must maintain their order in nested elements
- <contact/> denotes an empty element

Markup language examples

Examples of XML based markup languages

MathML Mathematical Markup Language.

CML Chemical Markup Language.

SpeechML Speech Markup Language.

XFRML Extensible Financial Reporting Markup Language.

SMIL Synchronized Multimedia Interface Language.

PDML Product Data Markup Language.

A MathML example

Here's an example of MathML:

```
<html><body>
  <math>
    <msup>
      <mi>x</mi>
      <mn>2</mn>
    </msup>
    <mo>=</mo>
    <mn>2</mn>
  </math>
</body></html>
```

A MathML example

This represents the following equation:

$$x^2 = 2$$

XML Parsers

XML Parsers and Processors

XML **parsers** or **processors** are used to check if an XML document obeys the XML syntax rules. A document that obeys the rules is called a **well-formed** document. Parsers typically build tree structures from XML documents. There are many different XML parsers available. Examples are:

- The W3C XML Parser used in Amaya, which is the W3C's open source HTML and XML editor and browser
- expat, Mozilla's XML parser
- msxml, Microsoft's XML parser

XML Parsers

XML Parsers

- There normally is an underlying data (or information) model behind every XML document. This data model can be specified in different ways (will look at this later)
- An XML parser can check if the given XML document adheres to the specified data model. Such a parser is called a **validating** parser
- An XML document that is parsed successfully by a validating parser is called a **valid** document
- A data model specification is not required for every XML document, but it is advisable