

XML

Jonathan Geisler

April 18, 2008



What is XML?

- IS ...

What is XML?

- IS ...
 - Text (portable)



What is XML?

- IS ...
 - Text (portable)
 - Markup (human readable)

What is XML?

- IS ...
 - Text (portable)
 - Markup (human readable)
 - Extensible (valuable for future)

What is XML?

- IS ...
 - Text (portable)
 - Markup (human readable)
 - Extensible (valuable for future)
- is NOT a ...

What is XML?

- IS ...
 - Text (portable)
 - Markup (human readable)
 - Extensible (valuable for future)
- is NOT a ...
 - Programming language

What is XML?

- IS ...
 - Text (portable)
 - Markup (human readable)
 - Extensible (valuable for future)
- is NOT a ...
 - Programming language
 - Transport protocol

What is XML?

- IS ...
 - Text (portable)
 - Markup (human readable)
 - Extensible (valuable for future)
- is NOT a ...
 - Programming language
 - Transport protocol
 - Database

Why did they make XML?

- HTML is too restrictive
- SGML is too complicated



- Very much like HTML
- Every start tag must have an end tag
- Elements may not overlap
- Attribute values must be quoted
- No duplicate attributes
- No comments inside tags
- No unescaped `<` or `&` signs

Valid vs. Well-formed

- Well-formed means the document is acceptable XML
- Valid means the document contains acceptable tags for the application domain



- Elements
- Tags
- Character Data
- Attributes
- Entities

Document Type Definition (DTD)

DTDs define what you really want for your XML content. You describe what a valid document consists of and any ordering that might be necessary.



Since XML is extensible by any, how can we ensure that no one else uses the same tag as us? Namespaces!



Since XML is extensible by any, how can we ensure that no one else uses the same tag as us? **Namespaces!**

But DTDs are hard . . . Let's use XML to define our XML! We'll be able to be more specific with our specifications and not have to learn a new syntax. The book gives a nice intro, but to master it, you would want to get a reference that goes into more detail.

Use CSS on XML the same way you did for XHTML since it is just one type of XML document. Nothing is special here . . .

eXtensible Stylesheet Language Transforms can powerfully transform an XML document to some other document. That could be another XML document, an XHTML document, or even an MS Word document.

In order to access various parts of our document to perform transformations we need a method for referring to the different parts. Think of this like the DOM for Javascript except much more powerful.

Very basics for XPath

To specify how to find the document element, we need three basic concepts:

- 1 Direction to search (XPath has the concept of a current node in the document to search from)
- 2 Node to test for matching
- 3 Filtering of matches

The combination of all three makes an XPath look a lot like a path in a URL or file system.



① /A/B/C

XPath examples

- 1 /A/B/C
- 2 A/B/C

XPath examples

- 1 /A/B/C
- 2 A/B/C
- 3 //A//B//C

XPath examples

- 1 /A/B/C
- 2 A/B/C
- 3 //A//B//C
- 4 /A/*

XPath examples

- 1 /A/B/C
- 2 A/B/C
- 3 //A//B//C
- 4 /A/*
- 5 /A/@att

XPath examples

- 1 /A/B/C
- 2 A/B/C
- 3 //A//B//C
- 4 /A/*
- 5 /A/@att
- 6 /A[@att]

We're going to look at some online examples rather than trying to reproduce them here. They come from <http://zvon.org/xxl/XSLTutorial/Output/contents.html> .



We can either have the entire document to work with (DOM) or just the elements we need to process (SAX). Each has benefits depending on the type you want to do. The DOM is familiar to you since you've already used it with Javascript. SAX will become more familiar to you as you work with events in Javascript, since it is essentially an event-based framework for parsing the document.