XML

Stefan Alfredsson

(based on material by Johan Garcia)

Agenda

- Introduction
- Concepts
- Syntax
- Examples

XML is a method for data structuring

- Structured data examples
  - Application specific: Word document, Excel, PDF
  - Usage specific: Patient journal, invoices
- XML provides rules for creating text-based formats for structured data that are / have:
  - Unambigous
  - Extensible
  - Internationalization support
  - Platform independence

XML vs. HTML

- XML looks a bit like HTML but isn't
- Similarities
  - XML uses *tags* (<tag>), just like HTML
  - XML uses attributes (name=value)
- BUT
  - HTML assigns meaning to each tag
  - XML uses tags to delimit data, meaning is given by the application context
  - (and with HTML, the application context is web browsing, that applies the meaning of tags)

XML is a family of technologies

- XML – define concept of "tag", "attribute", etc
- XSL – eXtensible Stylesheet Language
  - XSLT – A transformation language for rearranging, adding or deleting tags & attributes
  - Xpath, XSL Formatting objects
- Xlink – adding hyperlinks to an XML file (anchor href in HTML)
- XPointer – pointing to parts of an XML document (#-anchor in HTML)
- DOM – Document Object Model, a standard set of function calls for manipulating XML files
- RDF, XML Namespaces, XFragments, CSS, ...

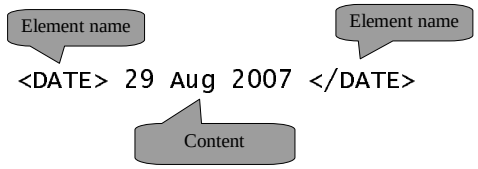
XML is new, but not that new

- Started in 1996
- Standard since feb 1998
- Idea of markup languages are not new
  - SGML (Standard Generalized Markup Language) in 70's and 80's
  - HTML (HyperText Markup Language) in 90's
- Easier to use than SGML, but not less powerful
- The design used the experiences from the large HTML userbase

Relations to other markups

- SGML → DTD → HTML
- SGML → XML
- XML → DTD → XHTML
- XML → XSLT → XML
- Semantics (meaning)
- Structure (connections)
- Style (presentation/rendering)

Element



The element is the fundamental building block in an XML document

Element, contd.

- The element must contain both start- and end tag (except empty element)
- Element can contain
  - Text data
  - Nested elements
  - Entity references or char references < >
  - CDATA sections <![CDATA [...]]> (escapes processing)
  - Processing instructions
  - Comments

Attributes

- In the start or end tag, *attributes* can be included
- An attribute is a name-value pair associated with the element
- Example:
  - <DATE CALENDAR="Julian">29 Aug 2007</DATE>
- Information can therefore be represented in two ways: as content in an element, or as value for an attribute

DTD

- DTD (Document Type Declaration) defines the document structure
- DTD can be seen as the grammar to define an application language
- DTD can define both generic languages such as XHTML and strict declarations (like a database schema)
- XML Schema is a newer alternative to DTD

Wellformed vs. valid

- An XML document can be either, Wellformed, Valid or Invalid
- Wellformed
  - Only one root element (document element)
  - Correctly nested elements
  - All elements must have start and end tags
- Valid
  - A wellformed document that adheres to a given DTD
- Invalid
  - A document that is nor well formed, nor valid

XML structure

Prolog `<?xml version="1.0"><!DOCTYPE BURGER SYSTEM "burger.dtd">`

Processing instr. `<?xml-stylesheet type="text/xsl" href="burger.xsl">`

Root element `<person> ... </person>`

Nested elements `<person> <name> ... </name> </person>`

<!-- Comments -->

DTD structure

- Prolog
  - `<?xml version="1.0"?>`
- Element list

```
<!ELEMENT name (#PCDATA)>
<!ELEMENT age (#PCDATA)>
<!ELEMENT person (name, age)>
<!ELEMENT reseller
  (info+,administrator,zone*,users?)>
```

- Attribute list

```
<!ATTLIST age unit CDATA #REQUIRED>
```

- Entity list
  - `<!ENTITY course "DVG C02">`
  - `&course;` expands to DVG C02

Burger King Example XML file

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE BURGER SYSTEM "burger.dtd">
<BURGER_KING>
 <PERSON_EATING NAME="MaryBeth">
   <BURGER AMOUNT="1">Bacon Cheeseburger</BURGER>
   <DRINK>Vanilla Milkshake</DRINK>
   <SIDE>Onion Rings</SIDE>
 </PERSON_EATING>
 <PERSON_EATING NAME="Patrick">
   <BURGER AMOUNT="2">Junior Whopper</BURGER>
   <DRINK>Root Beer</DRINK>
   <SIDE>Frys</SIDE>
 </PERSON_EATING>
</BURGER_KING>
```

Burger King Example DTD (”burger.dtd”)

```
<!ELEMENT BURGER_KING (PERSON_EATING+)>
<!ELEMENT PERSON_EATING (BURGER?,DRINK?,SIDE?)>
<!ATTLIST PERSON_EATING NAME CDATA #REQUIRED>
<!ELEMENT BURGER (#PCDATA)>
<!ATTLIST BURGER AMOUNT CDATA #REQUIRED>
<!ELEMENT DRINK (#PCDATA)>
<!ELEMENT SIDE (#PCDATA)>
```

Burger King example XSL-file

```
<?xml version="1.0" ?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl">
<xsl:template match="/">
 <HTML>
  <HEAD>
   <TITLE>Lunch at Burger King</TITLE>
  </HEAD>
  <BODY>
   <img src="burger.gif"/>
   <font size="5" color="red">Lunch at Burger King</font>
   <P/>
        <xsl:apply-templates select="BURGER_KING"/>
   <P/>
   </BODY>
  </HTML>
</xsl:template>
```

Burger King example XSL-file (contd)

```
<xsl:template match="BURGER_KING">
 <xsl:for-each select="PERSON_EATING">
  <B><xsl:value-of select="@NAME"/></B>
  <UL>
   <LI/>Burger: <xsl:value-of select="BURGER"/>
   <LI/>Drink: <xsl:value-of select="DRINK"/>
   <LI/>Side: <xsl:value-of select="SIDE"/>
  </UL>
  <P/>
 </xsl:for-each>
</xsl:template>

</xsl:stylesheet>
```

Programmatic handling of XML

- DOM: Document Object Model
  - API that builds a tree of XML elements which can be continously manipulated
- SAX: Simple API for XML
  - Event-driven API: data is manipulated as document is processed

XHTML and CSS

- XHTML is a stricter version of HTML
  - Document must be well formed
  - Extensible with namespaces
    - For example SVG, MathML

```
<?xml version="1.0" encoding="UTF-8"?>
 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns=http://www.w3.org/1999/xhtml xml:lang="sv" lang="sv">
```

- CSS: Cascading Style Sheets
  - Defines presentation separately from data
  - For example font sizes, text color, etc

Some key concepts

- Wellformed vs valid
- XML helps separate syntax, sematics and presentation of data
- A DTD specify the syntactic structure fo documents
- XSL generates the format needed for presentation
  - XHTML
  - WML
  - XML to XML transformation