The XML Metalanguage

Mika Raento

mika.raento@cs.helsinki.fi

University of Helsinki – Department of Computer Science
Preliminaries
Preliminaries

- Motivation
- Practicalities
- Course Overview
Motivation

XML is used to

- Publish documents (Linux documentation in DocBook, Reference works such as Dictionaries) in several formats from the same contents
- Publish news items on the web via RDF (for example: Slashdot, CNN, Mozillazine) that can be incorporated to other web sites or client software
- Store program settings and preferences (Gnome)
Motivation

- XML is used to
  - Exchange business documents, such as invoices and inventories (ebXML – new standard for EDI, for example Finnish customs documents)
  - Make remote procedure calls over the Internet (SOAP=Web Services allows calling Google from your code)
  - Build message-based large-scale software (SAP R/3 integration, Ascade Cockpit application)
Motivation

XML is

- Fairly easy to learn
- Human and machine-readable
- Lightweight for processing
- Easy to find software for
Practicalities

- 2 study weeks – 8 × 2 hours lectures – 8 × 2 hours exercises
- Lecturer Mika Raento, D419, available Thu 16–17
- Lectures in Finnish, material in English, one exercise group in English
- One course exam on Nov 10th
- Literature: The XML Companion, 3rd edition by Neil Bradley — or use the web
- Course web site on http://www.cs.helsinki.fi/u/mraento/teaching/xml_s03/
- Newsgroup news:hy.opiskelu.tk.tkl.xml
Practicalities

- Exam + project work

- Two ways to complete project work:
  - Exercises + smaller project that will be partially done at the exercises. You are allowed to miss at most two exercises. OR
  - Larger standalone project work (details at the course web page)

- So: cancel your registration at an exercise group, if you don’t think you’ll be able to attend
Practicalities

- Grading maximum 60 points
- Exam 30 points, 15 points minimum to pass
- Project work 30 points, 15 points minimum to pass (10 points from exercise attendance, 20 points from work OR 30 points from larger project work)
- 3 extra points (above 60) for exercises attended after the minimum six
- No exercise points taken into account if you take the separate exam
Course Overview

1. Introduction. History. Motivation
2. From HTML to XML. Well-formedness and validity.
4. DTD limitations. Alternatives
5. Namespaces. XML processing.
7. FO. Basics, mind set, more advanced topics.
Introduction
Introduction

- What is XML?
- What does it look like?
- What does 'metalanguage' mean?
- XML-processors
- DTDs
- Transformations
- Style definitions
eXtensible Markup Language

- W3C recommendation
  - Version 1.0 (1.1 candidate recommendation)
  - An agreed-upon textual format for representing tree-structured data
  - For storing, combining, exchanging and publishing information
  - Human- and machine-readable
XML document instance

<!-- Example document instance -->

<university>
    <department>
        <name>
            Department of Computer Science
        </name>
        <address>
            Teollisuuskatu 23
        </address>
    </department>
</university>
XML document instance

<!-- Example document instance -->

<university>

<department> Department of Computer Science
</department>

<address> Teollisuuskatu 23
</address>

</university>
XML document instance

<!-- Example document instance -->

<university>
  <department>
    <name>Department of Computer Science</name>
    <address>Teollisuuskatu 23</address>
  </department>
</university>
XML document instance

<!-- Example document instance -->

<university>
  <department>
    <name>
      Department of Computer Science
    </name>
    <address>
      Teollisuuskatu 23
    </address>
  </department>
</university>
XML document instance

- Actual document contents, that have been *marked up* in an agreed way
- Self-describing (for humans) *tags*
- *Elements* and nested elements, meaningful units of information
- *Text* within elements
- *Comments*
Logical vs. physical structure

- Logical structure
  - Logical relationships and constraints
  - Describes the structure of the information content

- Physical structure
  - Entities
  - Characters and character set
  - Files
XML Processors

- XML parser
  - Finds errors
  - Provides information to applications

- Entity (document part) management
  - Combines entities to documents
  - Combines physical files
Metalanguage XML

- XML provides a general syntax for tree structured data
- Users provide an application-specific grammar for this syntax
  - Element names
  - Element order and nesting
  - Certain reserved words
- This grammar is called a *Document Type Definition, DTD*
Example DTD

<!-- Document Type Definition (DTD) example -->
<!ELEMENT university (department+)>
<!ELEMENT department (name, address)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT address (#PCDATA)>
DTD

- Defines the type of the document, or its structure
- One rule per element
  - Name of the element
  - Allowed content
- Grammar for document instances
- "Regular expressions" for element content (may be recursive)
- not required in XML
Advantages of using DTDs

- Allows a *validating* parser
  - Checks that the document instance corresponds to the DTD
- Consistent use of the tags
- Standard DTDs for specific applications
  - A common vocabulary
Descriptive (declarative) markup

- Generalized markup (no formatting (information necessarily))
- syntactic form without semantics
- however includes element names that describe content
- In addition we need a way to describe formatting and e.g. links so that we can *present* the information to humans
Descriptive vs. procedural markup

- Descriptive
  - *Categorises* the document into parts
  - logical (logical parts and their relations)
  - self-describing
  - content and format separated
  - E.g. XML
Descriptive vs. procedural markup

- **Procedural**
  - Defines what *processing* is to be carried out on the document
  - Visible (e.g. \(\LaTeX\)) or invisible (e.g. Word)
  - Formatting information
  - Content and format mixed

- This distinction is neither clear-cut (e.g. \(\LaTeX\) with only \section, \subsection etc., or applications of XML such as XSL) nor all-encompassing, but provides a useful starting point
Stylesheets

- Defines the presentation (output) format
- Possibly several per DTD and/or document instance
- Cascading Style Sheets (CSS)
- XML Stylesheet Language (XSL)
- (DSSSL, not covered in this course)
Other applications of XML

- Data transfer
  - Subsets (views) of relational databases
  - EDI (Electronic data interchange)
  - Message-based applications
  - Coarse-grained RPC

- Publishing
  - Documents
  - Metadata

- Etc.
Publishing process

XML document → Formatting → Formatted document

XSL/CSS stylesheet
History of XML — SGML

- Standard Generalised Markup Language
- Based (in part) on IBM’s GML (1969)
- Introduced in 1974, ISO standard 1986
- Large and complicated
- Tools correspondingly large and complicated ← few and expensive
- XML has basically the same expressive power (almost a proper subset)
- Still widely used in publishing of very large documents/document collections
History of XML — HTML

- Huge success, basis for the web
- Non-standard extensions problematic (although nowadays mainly in javascript/DOM)
- Lots of tools available
History of XML — HTML

- An SGML DTD + predefined semantics
  - Practical when the only purpose is to present information
  - Easy and pleasing presentation in a browser
  - Focuses on tags for book-like document structure, presentation and linking
XML — SGML — HTML

- XML combines good features from both SGML (expressiveness, extensibility) and HTML (simple, easy to understand)

- Lots of tools available

- XHTML is HTML cast into an XML DTD (instead of SGML)

- SGML may still be the best format for very large documents

- XML does not solve all problems

- All three languages are needed
XML Design principles

1. XML shall be straightforwardly usable over the Internet
2. XML shall support a wide variety of applications
3. XML shall be compatible with SGML
4. It shall be easy to write programs which process XML documents
5. The number of optional features in XML is to be kept to the absolute minimum, ideally zero
XML Design principles

6. XML documents should be human-legible and reasonably clear
7. The XML design should be prepared quickly
8. The design of XML shall be formal and concise
9. XML documents shall be easy to create
10. Terseness is of minimal importance
Related standards

- XLink — hyperlinking for XML
- XPath — locating and selecting XML document parts
- XPointer — The reference language for XLink
- XSL — XML stylesheet language
- XSLT — XSL transformations
- SAX — stream-oriented XML API
- DOM — tree-oriented XML API
On this course

- Introduction, motivation, background
- XML documents (XML)
- XML DTDs (XML)
- XML transformations (XSLT)
- Stylesheets (CSS, XSL)
- Some tools
- Related information (XML Schema, XPath, XML Namespaces)
Literature

- Bradley: The XML Companion
- http://www.w3.org/XML/
- http://www.xml.org
- http://www.xml.com
- http://www.xmlsoftware.com
- http://www.xslinfo.com
- http://xml.coverpages.org
- http://xml.apache.org
This lecture in literature

- Bradley: 2, 3, 31
- or