

XML (30 hours)

Introduction to XML development

- Objectives
- Quick example
- Key benefits of XML
- What is a markup language?
- SGML: some history
- Relationship of SGML to XML
- Evolution of XML
- Why is XML important?
- Uses for XML
- Quick example — revisited
- Modelling data
- Elements
- Attributes
- Nesting elements
- Structure of a document
- The <?xml...?> declaration
- What are web services?
- Benefits of web services
- Current status

XHTML

- Motivation behind XHTML
- Example document
- XHTML standards
- Differences between HTML and XHTML
- Writing XHTML for older browsers
- Including scripts and stylesheets
- Advantages of XHTML

Well-formed XML documents

- Objectives
- Valid versus well-formed XML
- Well-formed XML documents
- Valid XML documents
- Recap basic terms
- Element naming rules
- Attributes
- Comments

- XML declaration
- Processing instructions
- Non-permissible characters
- Errors and fatal errors

XML namespaces

- Objectives
- Need for namespaces
- Problem: same names different concepts
- Solution: group names in namespaces
- Problem: ensuring unique namespaces
- Using basic XML namespaces
- Example: basic XML namespaces
- Default namespaces
- Example: default namespaces
- Default namespaces: inheritance and cancellation
- Example: namespace inheritance
- Different namespace notation
- Attributes and namespaces
- Extending XML
- Namespaces and DTDs
- The use of URLs in namespaces

An introduction to XML schemas

- Objectives
- The principle of schemas
- The place of DTDs
- Writing a schema
- Example: simple memo format
- Example: schema for <memo> documents
- Sequences of elements
- <sequence>
- Complex and simple types
- Defining attributes
- Simple types
- <attribute use= > values
- Tying documents to schemas
- Instance documents which use namespaces
- Schema processing tools
- Authoring tools
- XML parsers with schema support
- Other tools supporting schemas
- Web resources

Valid XML with Document Type Definitions (DTDs)

- [Objectives](#)
- [Goals of DTDs](#)
- [Schemas and DTDs: differences](#)
- [Validation](#)
- [Example: the memo format](#)
- [Example: DTD for memos](#)
- [The document type declaration](#)
- [The PUBLIC source specifier](#)
- [The SYSTEM source specifier](#)
- [Declaring entities](#)
- [Entities jargon](#)
- [Declaring elements](#)
- [Declaring empty elements](#)
- [The ANY content model](#)
- [#PCDATA content](#)
- [Elements which contain elements](#)
- [Optional elements](#)
- [Repeated elements](#)
- [Repeated elements \(continued\)](#)
- [Mixed content](#)
- [Declaring attributes](#)
- [Attribute types](#)
- [Adding per-document entities](#)
- [Limitations of DTDs](#)

Principles of styling XML

- [Objectives](#)
- [What is styling?](#)
- [Styling XML](#)
- [CSS](#)
- [Using CSS with XML](#)
- [XSL](#)
- [Putting XSL together](#)
- [XSLT](#)
- [XSL-FO](#)
- [Associating style sheets with XML documents](#)
- [The <?xml-stylesheet?> processing instruction](#)
- [Convergence between CSS and XSL-FO](#)

Using cascading stylesheets with XML

- [Objectives](#)
- [CSS introduction](#)

- CSS in HTML
- CSS in XML
- Using CSS with XML

A brief introduction to XSLT

- Why transforming XML is useful
- Business to business transactions
- XSLT for transforming XML
- Disadvantages of XSLT
- Available software
- Using XT
- Different kinds of XSLT processors
- XSLT and XSL-FO for presentation
- Example: simplified syntax
- Input and output of stylesheet
- Example: name of root element
- Non-simplified syntax
- How stylesheet processing works
- Empty stylesheets
- Input and output of empty stylesheet
- Getting values from the input
- Example: getting a value from the input
- select="..." expressions
- What is XPath?
- Accessing the values of attributes

Further XSLT

- Example: inserting static text
- Copying elements unchanged
- Example: renaming an element
- Example: removing an element
- Conditionally including output
- Example: conditional templates
- Example: counting elements
- Inserting text into the output
- Example: explicitly inserting text
- Example: listing elements
- Current node in a template
- Example: simple arithmetic
- Example: listing URLs from links
- Example: looping
- Example: sorting

XSLT details

- Root node

- [XPath expression syntax](#)
- [Template matching with XPath](#)
- [Applying templates](#)
- [Building location paths with XPath](#)
- [Predicates](#)
- [Using XPath in XML](#)
- [XPath numeric functions](#)
- [XPath's string functions](#)
- [Axes: heading in other directions](#)
- [Changing case with translate](#)
- [XPath summary](#)
- [Choosing among alternatives](#)
- [Example of <xsl:choose>](#)
- [Whitespace stripping](#)
- [Calling templates by name](#)
- [Passing parameters to templates](#)
- [Example: changing newlines to
](#)
- [Structuring stylesheets](#)

XSL-FO

- [Objectives](#)
- [What is XSL-FO?](#)
- [XSL-FO basics](#)
- [Processing XSL-FO](#)
- [Internationalisation](#)
- [Simple page layout](#)
- [Real page layout](#)
- [XSL-FO page layout jargon](#)
- [More jargon: regions](#)
- [Creating pages](#)
- [A simple example](#)
- [A simple example \(continued\)](#)
- [Static content](#)
- [XSL-FO and XSLT](#)
- [XSL-FO and XSLT \(continued\)](#)
- [XSLT example](#)
- [Other features](#)
- [Tables](#)
- [Inheritance](#)
- [The future](#)

Processing XML

- [Objectives](#)
- [Reasons for processing](#)
- [How to process XML](#)
- [Choosing a processing method](#)

- General XML processing with XSLT
- XML parser types
- Event-based parsing
- Tree-based parsing

Processing XML events with SAX

- Objectives
- SAX — the simple API for XML
- Event-based programming with SAX
- Typical steps in using SAX
- Capturing element events
- Example of a ContentHandler
- Start and end of documents
- Capturing character data
- Example: totalling prices in an invoice
- Example: continued
- Positions of elements
- Extensions to SAX

Processing XML objects with DOM

- What is the DOM?
- DOM level 1 object model
- The XML DOM
- DOM's relationship to applications
- DOM interfaces
- Document representation by DOM
- The DOM core
- The DOM core interfaces
- Interfaces provided by the DOM core
- DOM core interfaces (contd.)
- Using the DOM in Java
- Example: print href values of <a>s
- Catching validation errors
- Error handlers in an anonymous inner class
- The future of DOM