

XML and XSL Basics.

By

Amit Kumar

Graduate School of Library and Information

Science

UIUC



Webmasters Forum 2005

Outline 1/3

- XML Basics
 - What is XML
 - Valid and Well Formed XML
 - Grammar
 - Uses of XML
 - Some Well Known encoding grammars
 - EAD, TEI, METS, HTML (SGML), XHTML



Outline 2/3

- XSL Basics
 - What is XSL.
 - Structure of XSL Style sheet.
 - XSL and CSS.
 - Processing XSLT with XML.
 - XPATH and functions.



Outline 3/3

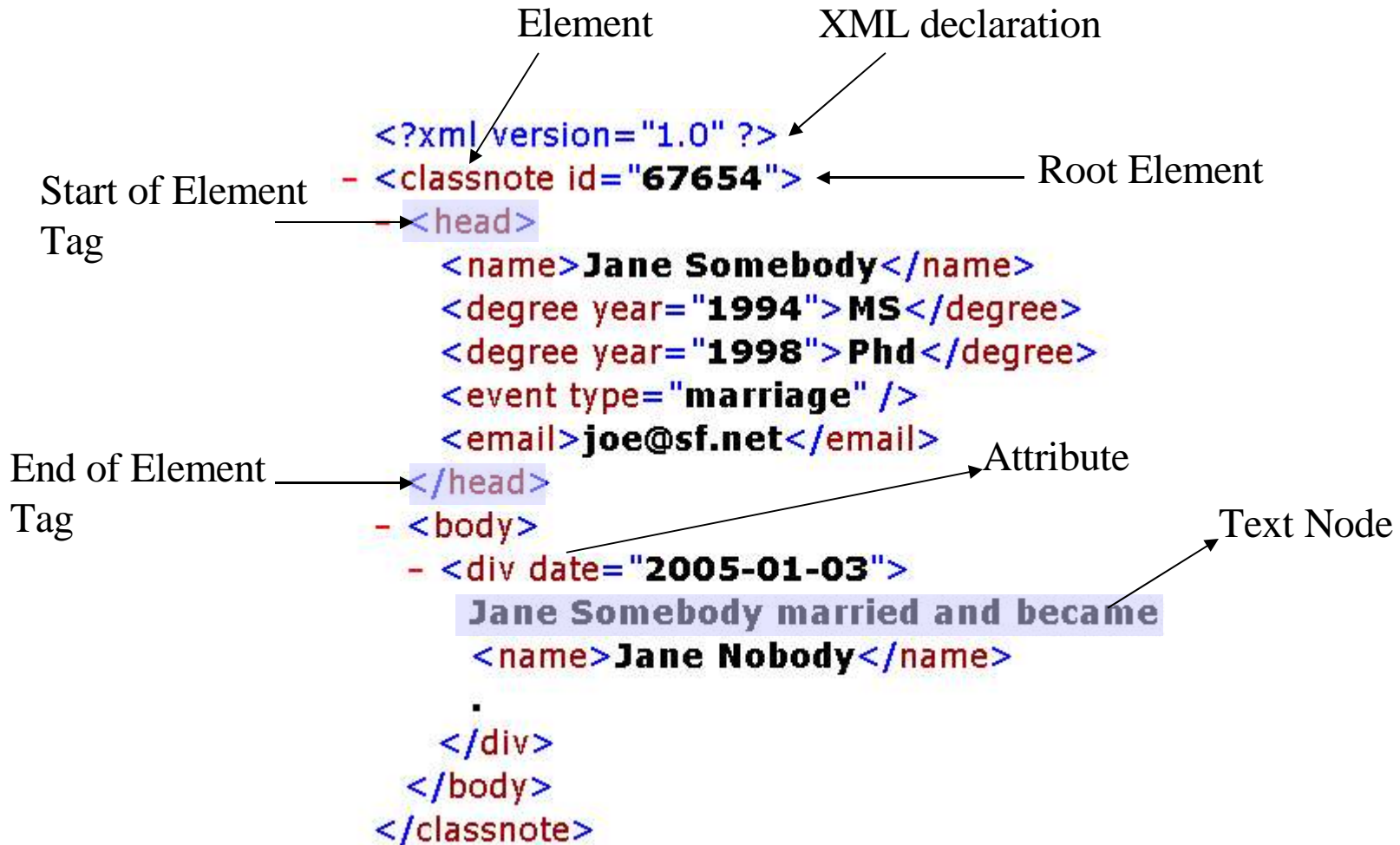
- Practical
 - XML and Websites
 - Document and Data centric aspects.
 - XML Editing Tools
 - jEdit and oXygen
 - Web Publishing Tools
 - Cocoon
 - AxKit
 - teiPublisher
 - Sample Websites
 - www.lib.umd.edu/archivesum
 - www.macgreevy.org
- Questions



XML

- eXtensible Markup Language
 - A specification developed by W3C, it is a pared down version of SGML, designed for web documents.
 - It is used to create customized tag sets for markup.
- Used for
 - Electronic Data Exchange.
 - Data Modeling.
 - Document Modeling.
 - Creating custom languages.





Valid and Well Formed XML

- An XML document is well formed if
 - Every start tag has end tag & attribute values quoted.
 - Elements are closed in the order they are opened.
 - One Root Element.
 - Entities are declared and used correctly.
- An XML document is Valid if
 - It follows a Grammar.



Grammar

- Formats
 - DTD
 - Order of the elements
 - Nesting, default attribute values
 - Modern Schema Languages
 - All the above
 - Data typing (String, Date, Boolean)



Sample DTD

```
<!ELEMENT classnote (head,body)>
<!ATTLIST classnote id ID #REQUIRED>
<!ELEMENT head (name,degree*,event*,website?,email?)>
....
....
<!ELEMENT body (div*)>
<!ELEMENT div (#PCDATA | title | institution)*>
<!ATTLIST div date CDATA #REQUIRED>

<!ELEMENT title (#PCDATA)>
<!ATTLIST title type (book|article) "book">

<!ELEMENT institution (#PCDATA)>
<!ATTLIST institution type (work|award) "work">
```



How Is XML Used ?

- Data Exchange
 - Electronic Data Exchange.
- Separation of Data and Presentation
 - Data stored in XML format.
 - Presentation stored separately.
- Programming Languages
 - XSLT.
 - Java Server Pages XML syntax.



Well Know DTDs

- Encoded Archival Description (EAD)
 - For encoding Archival Finding Aids.
 - <http://www.loc.gov/ead/>
- Text Encoding Initiative
 - For research, teaching and preservation of Literary and Linguistic texts.
 - <http://www.tei-c.org>
- Metadata Encoding & Transmission Standard (METS)
 - Metadata about a Digital Object
 - <http://www.loc.gov/standards/mets/>
- HTML (SGML) / XHTML
 - <http://www.w3.org/TR/REC-html40/sgml/dtd.html>
 - http://www.w3.org/TR/xhtml1/dtds.html#a_dtd_XHTML-1.0-S



XML and HTML

- HTML is a mark up language based on SGML.
- Some element end tags are optional like `
` and `<p>`.
- HTML is case insensitive.
- HTML is presentation specific.
`On Shakespear` by
`<i>John Milton</i>`
- XML is used to create markup languages.
- All element start tags have corresponding end tags.
- XML is case sensitive
`
 !=
`.
- XML tags and attributes describe what the individual piece of information is and the relationship between them.



What Is XSL ?

- A family of W3C recommendations for XML transformation and presentation.
 - XSL Transformation (XSLT): a language for transforming XML documents.
 - An expression language used to access or refer to parts of the XML document (XPath).
 - An XML vocabulary for specifying formatting syntax (XSL-FO)
- All XSLT/FO documents are well formed XML documents.



XSLT

- For handling/formatting/manipulating XML content
- For creating other documents from a XML document
 - Other xml docs
 - HTML docs
 - RTF



XSLT

- XSLT 1.0 was recommended by W3C on 16 Nov 1999
- XSLT 2.0 is a Working Draft latest version- 4 April 2004
- several XSLT implementations available
 - David Clarke's **XT** <http://www.jclark.com/xml/xt.html>
 - can be used for server side processing
 - Can use IE/Mozilla/FireFox
 - for client side processing
 - Saxon XSLT processor by Michael Kay
 - <http://saxon.sourceforge.net/>
 - Xalan XSLT processor by Apache
 - <http://xml.apache.org/xalan-j/>



Trees & Nodes

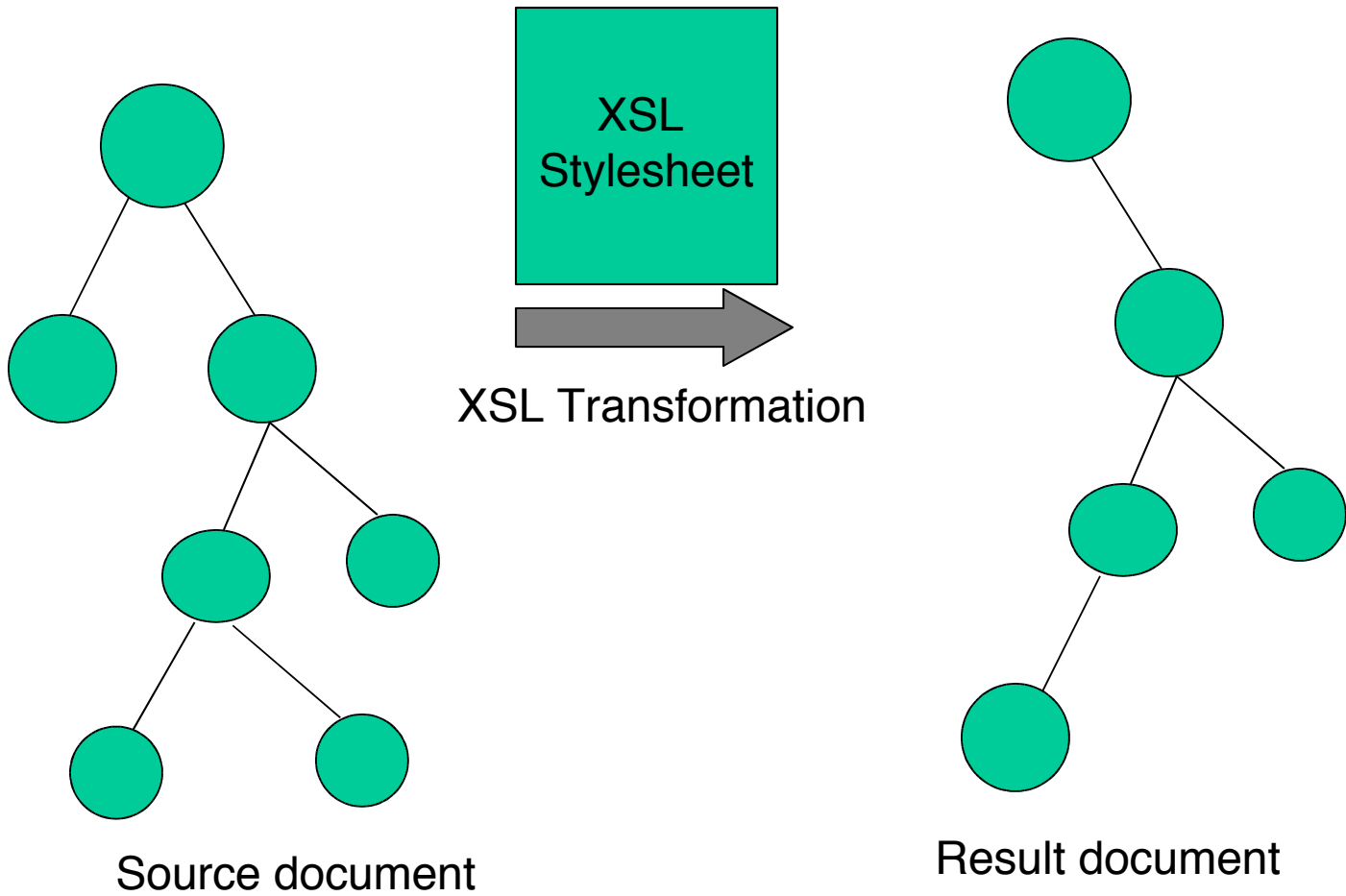
- With XSLT one does not think in terms of documents, but in terms of trees & nodes
- A tree represents the data in a document as a set of nodes
- Nodes are elements, attributes, comments, etc. in a hierarchy



XSLT consists of two parts

- A source document
 - Original XML document.
- A result tree
 - Newly-created XML/HTML/RTF or plain text document.





Structure of an XSLT Document

- Top level Elements

- xsl:output
- xsl:template
- xsl:import

.....

```
<?xml version="1.0" ?>
- <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <!-- OUTPUT METHOD -->
  <xsl:output method="html" />
  <!-- TEMPLATE -->
+ <xsl:template match="/">
  <!-- TEMPLATE MATCHING CLASSNOTE -->
+ <xsl:template match="classnote">
</xsl:stylesheet>
```



XSL Document

```

<?xml version="1.0" ?>
- <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  <!-- OUTPUT METHOD -->
  <xsl:output method="html" />
  <!-- TEMPLATE -->
- <xsl:template match="/">
- <html>
+ <head>
- <body>
  <xsl:apply-templates />
  </body>
</html>
</xsl:template>
<!-- TEMPLATE MATCHING CLASSNOTE -->
+ <xsl:template match="classnote">
  .....
+ <xsl:template match="other elements">
  .....

</xsl:stylesheet>

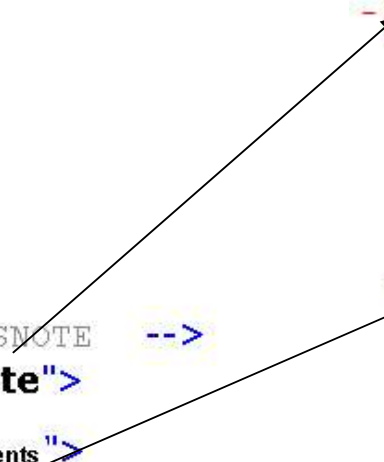
```

XML document

```

<?xml version="1.0" ?>
- <classnote id="67654">
- <head>
  <name>Jane Somebody</name>
  <degree year="1994">MS</degree>
  <degree year="1998">Phd</degree>
  <event type="marriage" />
  <email>joe@sf.net</email>
</head>
- <body>
- <div date="2005-01-03">
  <name>Jane Somebody</name>
  married and became
  <name>Jane Nobody</name>
  .
</div>
</body>
</classnote>

```



```

<?xml version="1.0" ?>
- <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <!-- OUTPUT METHOD -->
  <xsl:output method="html" />
  <!-- TEMPLATE -->
- <xsl:template match="/">
  - <html>
    - <head>
      <link href="admin-styles.css" media="screen" type="text/css" rel="stylesheet" />
      <title>Simple XSL Output</title>
    </head>
    - <body>
      <xsl:apply-templates />
    </body>
  </html>
</xsl:template>
  <!-- TEMPLATE MATCHING CLASSNOTE -->
+ <xsl:template match="classnote">
</xsl:stylesheet>

```

HTML



```

<?xml version="1.0" ?>
- <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <!-- OUTPUT METHOD -->
  <xsl:output method="html" />
  <!-- TEMPLATE -->
- <xsl:template match="/">
  - <html>
    - <head>
      <link href="admin-styles.css" media="screen" type="text/css" rel="stylesheet" />
      <title>Simple XSL Output</title>
    </head>
    - <body>
      <xsl:apply-templates />
    </body>
  </html>
</xsl:template>
  <!-- TEMPLATE MATCHING CLASSNOTE -->
- <xsl:template match="classnote">
  <b>Name:</b>
  <xsl:value-of select="head/name" />
  <br />
  <b>Degree Year:</b>
  <xsl:value-of select="head/degree/@year" />
  <br />
  - <p>
    <b>Notes:</b>
    <br />
    <xsl:value-of select="body/div" />
  </p>
</xsl:template>
</xsl:stylesheet>

```

HTML

XPATH

XSLT Processing Model.

- The parser goes to the root template

“<xsl:template match="/">”

- Processes the HTML literals.
- When it comes across

“<xsl:apply-templates />”

It goes to the XML document to find the matching nodes.



```

<?xml version="1.0" ?>
- <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <!-- OUTPUT METHOD -->
  <xsl:output method="html" />
  <!-- TEMPLATE -->
- <xsl:template match="/"> Root Template
  - <html>
    - <head>
      <link href="admin-styles.css" media="screen" type="text/css" rel="stylesheet" />
      <title>Simple XSL Output</title>
    </head>
    - <body>
      <xsl:apply-templates />
    </body>
  </html>
</xsl:template>
<!-- TEMPLATE MATCHING CLASSNOTE -->
- <xsl:template match="classnote">
  <b>Name:</b>
  <xsl:value-of select="head/name" />
  <br />
  <b>Degree Year:</b>
  <xsl:value-of select="head/degree/@year" />
  <br />
  - <p>
    <b>Notes:</b>
    <br />
    <xsl:value-of select="body/div" />
  </p>
</xsl:template>
</xsl:stylesheet>

```

```

<?xml version="1.0" ?>
<classnote id="67654">
- <head>
  <name>Jane Somebody</name>
  <degree year="1994">MS</degree>
  <degree year="1998">Phd</degree>
  <event type="marriage" />
  <email>joe@sf.net</email>
</head>
- <body>
  - <div date="2005-01-03">
    <name>Jane Somebody</name>
    <name>married and became</name>
    <name>Jane Nobody</name>
    .
  </div>
</body>
</classnote>

```

```
<?xml version="1.0" ?>
- <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <!-- OUTPUT METHOD -->
  <xsl:output method="html" />
  <!-- TEMPLATE -->
- <xsl:template match="/">
- <html>
  - <head>
    <link href="admin-styles.css" media="screen" type="text/css" rel="stylesheet" />
    <title>Simple XSL Output</title>
  </head>
  - <body>
    <xsl:apply-templates />
  </body>
</html>
</xsl:template>
<!-- TEMPLATE MATCHING CLASSNOTE -->
- <xsl:template match="classnote">
  <b>Name:</b>
  <xsl:value-of select="head/name" />
  <br />
  <b>Degree Year:</b>
  <xsl:value-of select="head/degree/@year" />
  <br />
  - <p>
    <b>Notes:</b>
    <br />
    <xsl:value-of select="body/div" />
  </p>
</xsl:template>
</xsl:stylesheet>
```

Start of the Style sheet

Output method

Template

End of the Style sheet

XSL Document

Source Document

```
<?xml version="1.0" ?>
- <classnote id="67654">
- <head>
  <name>Jane Somebody</name>
  <degree year="1994">MS</degree>
  <degree year="1998">Phd</degree>
  <event type="marriage" />
  <email>joe@sf.net</email>
</head>
- <body>
- <div date="2005-01-03">
  <name>Jane Somebody</name>
  married and became
  <name>Jane Nobody</name>
  .
</div>
</body>
</classnote>
```

```
<?xml version="1.0" ?>
- <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <!-- OUTPUT METHOD -->
  <xsl:output method="html" />
  <!-- TEMPLATE -->
+ <xsl:template match="/">
  <!-- TEMPLATE MATCHING CLASSNOTE -->
+ <xsl:template match="classnote">
</xsl:stylesheet>
```

Browser Display

Name: Jane Somebody
Degree Year: 1994

Notes:
Jane Somebody married and became Jane Nobody.



Result Document

```
<html>
  <head>
    <META http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <link rel="stylesheet" type="text/css" media="screen" href="admin-styles.css">
    <title>Simple XSL output</title>
  </head>
  <body>
    <b>Name:</b>Jane Somebody<br>
    <b>Degree Year:</b>1994<br>
    <p>
      <b>Notes:</b>
      <br>
      Jane Somebody married and became Jane Nobody.
    </p>
  </body>
</html>
```



XPATH

- Some Example XPATH
 - `/classnote` Return classnote
 - `/classnote/@id` Return id of the classnote
 - `/classnote/head` Return head child of the classnote
 - `/classnote/head[./degree/@year="1994"]` Return head child of the classnote, the head node should have a degree child with an attribute year and further the value of the attribute should be 1994
- jEdit Demo
 - Structure Browser
 - DTD elements display
 - XPATH queries



XSLT and CSS in Tandem

- CSS
 - Used in tandem with XSLT to style font, background, location and size of an element
 - Can not change the order of elements in a document; style or process on the basis of the content.
- XSLT
 - Transform one type of document to another; uses CSS or html style elements for presentation.



XML and Websites

- Data Centric
 - List of Students who have taken LIS 590 DM
 - List of All the Professors interested in Numerical Analysis
 - List of All the 500 level courses being taught in the current semester
- Easily mapped to Relational Database Model
- XML and XSLT provide alternative for data storage, presentation and query mechanism



DEMO OF STUDENTS AND COURSE XML DATA MODELS -- GSLIS



XML and Websites

- Document Centric
 - Find the paragraph in a the paper published by a Faculty “Joe Rack” that has name “Jack quack”
 - Display the abstract of all the research papers published by NCSA
 - Compendium of Literary documents; Collection of Finding aids
- Difficult to map to Relational Database
- Ideal for XML and XSLT based solution



DEMO

<http://www.lib.umd.edu/archivesum>

-- Admin Website

<http://www.macgreevy.org>

-- Administrative Interface



Open Source and Free XML Tools 1/2

- Editing Tools:
 - jEdit with XML XSLT Plugins: www.jedit.org.
 - oXygen: \$ XML XSLT Editor Debugger.



Open Source and Free XML Tools 2/2

- Publishing Tools and Databases.
 - Apache AxKit : Perl based.
 - Apache Cocoon : Java based.
 - eXist and Xindice Native XML Database : XPATH and XQUERY Search across multiple XML documents
 - <teiPublisher> : An application to develop and publish document centric XML website with search and browse facilities.



Thanks.
Questions ?

