

Lab: XML

NetDB

CS, NTHU,

Fall, 2013

Outline

- What is XML?
- XML Elements and Attributes
- Entity References & Comments
- XML Namespaces
- Well Formed and Valid Documents
 - DTD
 - Schema

Outline

- What is XML?
- XML Elements and Attributes
- Entity References & Comments
- XML Namespaces
- Well Formed and Valid Documents
 - DTD
 - Schema

What is XML?

- XML stands for e**X**tensible **M**arkup **L**anguage
- Designed to transport or store data
 - Commonly used in a configuration file (like pom.xml), *.doc**x**, *.ppt**x**, etc.
 - Messages exchanged between computers (like **X**HTML web pages)
- Why is XML so popular?
 - Human readable
 - Very easy!

A Sample XML Document

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <note id="6501">
3.   <to>Tove</to>
4.   <from>Jani</from>
5.   <heading>Reminder</heading>
6.   <body>Dear John,<br />Study hard!</body>
7. </note>
```

- A document that stores a note, which is described in XML
- The first line is the XML declaration
 - Defines version and encoding
 - Not part of the XML
- The rests consist of a hierarchy of elements (note, to, from, etc.)
 - Elements can have children
 - An XML document must have a root element

Outline

- What is XML?
- XML Elements and Attributes
- Entity References & Comments
- XML Namespaces
- Well Formed and Valid Documents
 - DTD
 - Schema

XML Elements and Attributes

- Must have a start and closing tag
 - E.g., `<to> . . . </to>`, or `
`
 - Tags are case sensitive
 - Naming: `<firstName>`, `<group-id>`, etc.
- Content can be text or (child) elements
- Can have attribute
 - E.g., `<note id="6501"> . . .`
 - Value must be quoted

A Child Element or Attribute?

- Which is the better way to add a date?

```
1. ...
2. <note id="6501">
3.   <date>11/02/2010</date>
4.   <to>Tove</to>
5.   <from>Jani</from>
6.   <heading>Reminder</heading>
7.   <body>...</body>
8. </note>
```

```
1. ...
2. <note id="6501" date="11/02/2010">
3.   <to>Tove</to>
4.   <from>Jani</from>
5.   <heading>Reminder</heading>
6.   <body>...</body>
7. </note>
```

- Rule of thumb: use attribute for metadata only
 - As an element if date is part of a note
 - As an attribute if date helps (the parser) identify a note

Outline

- What is XML?
- XML Elements and Attributes
- **Entity References & Comments**
- XML Namespaces
- Well Formed and Valid Documents
 - DTD
 - Schema

Entity References & Comments

- What if you want to specify “<” in a text content or attribute value?
 - Replace “<” with entity reference
 - E.g., `<body>2 > 1</body>`

Character	Entity Reference
<	<
>	>
&	&
'	'
“	"

- Comments: `<!-- A comment -->`

Outline

- What is XML?
- XML Elements and Attributes
- Entity References & Comments
- **XML Namespaces**
- Well Formed and Valid Documents
 - DTD
 - Schema

XML Namespaces (1/2)

- What if the element names conflict?

```
1. <table>
2.   <tr>
3.     <td>Apples</td>
4.     <td>Bananas</td>
5.   </tr>
6. </table>
7. ...
8. <table>
9.   <name>Office Desk</name>
10.  <width>80</width>
11.  <length>120</length>
12. </table>
13. ....
```

- Remember “package” in Java?
 - XML has something similar

XML Namespaces

(2/2)

- Syntax:
 - xmlns:prefix="URI"
- Uniform Resource Identifier (URI) identifies an Internet Resource
 - The most common URI is the Uniform Resource Locator (URL)
 - Can provide namespace information there
- The point: give a namespace a globally unique name
- Namespace without prefix is called default namespace
 - All elements reside in the default namespace unless prefixed

```
1. <root
    xmlns:html="http://www.w3.org/TR/html4/"
    xmlns="http://www.studio.cs.nthu.edu/lab">
2.   <html:table>
3.     <html:tr>
4.       <html:td>Apples</html:td>
5.       <html:td>Bananas</html:td>
6.     </html:tr>
7.   </html:table>
8.   ...
9.   <table>
10.    <name>Office Desk</name>
11.    <width>80</width>
12.    <length>120</length>
13.  </table>
14.  ...
15.</root>
```

Outline

- What is XML?
- XML Elements and Attributes
- Entity References & Comments
- XML Namespaces
- **Well Formed and Valid Documents**
 - DTD
 - Schema

Well Formed and Valid Documents (1/3)

- A XML document is **well formed** if it has correct XML syntax mentioned above
- But how can we enforce that a `note` must contain `to`, `from`, `heading`, and `body`?
 - There can be an external **DTD** or **Schema** file that defines the element hierarchy of a document
- XML DTD (Document Type Definition):

```
1. <!DOCTYPE note
2. [
3. <!ELEMENT note (to,from,heading,body)>
4. <!ELEMENT to (#PCDATA)>
5. <!ELEMENT from (#PCDATA)>
6. <!ELEMENT heading (#PCDATA)>
7. <!ELEMENT body (#PCDATA)>
8. ]>
```

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE note SYSTEM "Note.dtd">
3. <note id="6501">
4.   <to>Tove</to>
5.   <from>Jani</from>
6.   <heading>Reminder</heading>
7.   <body>...</body>
8. </note>
```

Well Formed and Valid Documents (2/3)

- XML Schema:
 - Also an XML document, preferred to DTD
 - Defines the element structure **in a namespace**

```
1. <?xml version="1.0"?>
2. <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
   targetNamespace="http://www.studio.cs.nthu.edu"
   xmlns="http://www.studio.cs.nthu.edu"
   elementFormDefault="qualified">
3.   <xs:element name="note">
4.     <xs:complexType>
5.       <xs:sequence>
6.         <xs:element name="to" type="xs:string"/>
7.         <xs:element name="from" type="xs:string"/>
8.         <xs:element name="heading" type="xs:string"/>
9.         <xs:element name="body" type="xs:string"/>
10.      </xs:sequence>
11.    </xs:complexType>
12.  </xs:element>
13.</xs:schema>
```


Well Formed and Valid Documents (3/3)

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <note id="6501"
    xmlns="http://www.studio.cs.nthu.edu"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.studio.cs.nthu.edu http://.../note.xsd">
3.   <to>Tove</to>
4.   <from>Jani</from>
5.   <heading>Reminder</heading>
6.   <body>...</body>
7. </note>
```

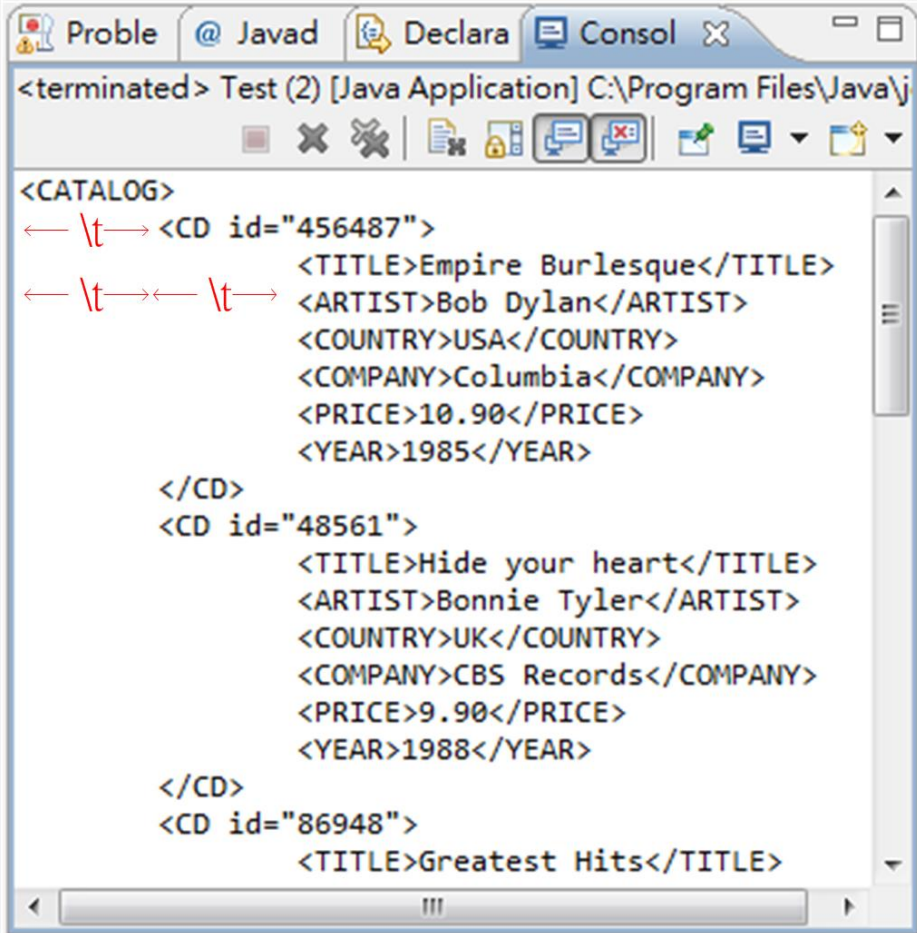
- The “http://www.w3.org/2001/XMLSchema-instance” namespace is known by any XML parser/validator
 - No need to specify its schema location
- But the XML parser/validator doesn’t understand our custom “http://www.studio.cs.nthu.edu” namespace
 - The xsi:schemaLocation attribute tells where the schema file for each custom namespace can be found
 - The value can have multiple pairs (separated by space)
- A **valid** XML document is:
 1. A well formed XML document
 2. Conforms to rules of all referenced DTD/Schemas

More XML?

- See tutorials from W3C School
 - [XML Tutorial](#)
 - [XML Schema Tutorial](#)

Exercise: Printing a Formatted XML (1/3)

1. Checkout the project "xmlparser" from https://netdb.cs.nthu.edu.tw/svn/courses/software_studio/2013_fall/samples
2. We already have a simple xml parser to parse a xml file into a tree structure.
 - To do:
 - 1) Modify the code in `XmlParser.java` to make it be able to parse the attributes.
 - 2) Override the method `toString()` in `XmlTree.java` and `XmlNode.java` to print the formatted version in the console.



The screenshot shows a Java IDE window with a console tab active. The console displays the output of a Java application, which is a formatted XML document. The XML is a catalog of CDs. The first CD has the title "Empire Burlesque" by Bob Dylan, from the USA, on Columbia, priced at 10.90, released in 1985. The second CD is "Hide your heart" by Bonnie Tyler, from the UK, on CBS Records, priced at 9.90, released in 1988. The third CD is "Greatest Hits" (title only is visible). The XML is formatted with indentation and line breaks, and the console window has a scrollbar on the right.

```
<terminated> Test (2) [Java Application] C:\Program Files\Java\j
<CATALOG>
< CD id="456487">
  <TITLE>Empire Burlesque</TITLE>
  <ARTIST>Bob Dylan</ARTIST>
  <COUNTRY>USA</COUNTRY>
  <COMPANY>Columbia</COMPANY>
  <PRICE>10.90</PRICE>
  <YEAR>1985</YEAR>
</CD>
<CD id="48561">
  <TITLE>Hide your heart</TITLE>
  <ARTIST>Bonnie Tyler</ARTIST>
  <COUNTRY>UK</COUNTRY>
  <COMPANY>CBS Records</COMPANY>
  <PRICE>9.90</PRICE>
  <YEAR>1988</YEAR>
</CD>
<CD id="86948">
  <TITLE>Greatest Hits</TITLE>
```

Exercise: Printing a Formatted XML (2/3)

- Example

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <note id="6501">
3.   <to>Tove</to>
4.   <from>Jani</from>
5.   <heading>Reminder</heading>
6.   <body>Dear John, Study hard!</body>
7. </note>
```

– In the root node, we have

- root.tagName: "note"
- root.content: "<to>...</to>...</body>"
- root.attributes: <"id", "6501">
- root.children: to, from, heading, body

Exercise: Printing a Formatted XML (3/3)

- Notify TA when you are done
- Hint:
 - You may need to consult [Java API](#) to find out how to get the elements in `ArrayList` and `TreeMap`.
 - Write a recursive method in `XmlNode`.