YTU

Web技术原理及应用
Web系统与技术

第7章 XML简介

烟台大学计算机学院
陈智育

内容

- 7.1 简介
- 7.2 XML语法
- 7.3 XML文档结构
- 7.4 文档类型定义[DTD]
- 7.5 名称空间
- 7.6 XML构架[Schema]
- 7.7 显示没有格式化的XML文档
- 7.8 通过CSS显示XML文档
- 7.9 XSLT样式表
- 7.10 XML处理器
- 7.11 Web服务(Service)

7.1 简介

- **元标记[meta-markup]语言**
  - 是指一种可以定义标记语言的语言
- **SGML是一种元标记语言**
  - 1980s初开发, 1986成为ISO标准
  - 1990s初, SGML->HTML, 专用于Web文档
- **XML也是一种元标记语言**
  - 1998发布1.0版, 2004发布1.1版

7.1 简介

❖HTML的两个问题

- 虽描述了信息显示的布局和外观,但没有体现这些信息的含义(标签集固定)
- 对标签在文档中的排放位置和顺序没有限制(如:不正确嵌套)

❖解决思路

- 基于元标记语言, 为特殊用户专门定义一种新的标记语言; 并要求严格语言规则

7.1 简介

- 一种解决方案
  - 基于SGML, 为不同用户群定义新的标记语言
  - 但SGML过于庞大和复杂, 难以实现方案
- 另一种解决方案
  - 定义一个SGML简化版本-XML
  - 基于XML定义新的标记语言
  - XML语法严格, 可避免第二个问题

7.1 简介

- XHTML是基于XML的HTML版本
- XML是一种简单通用的数据存储和交换的方式
- XML没有预定义任何标签
- 一种基于XML的标记语言称为一个标签集(XML应用)
- 使用基于XML的标记语言编写的文档称为XML文档

7.2 XML语法

- **XML语法可分为两种不同的级别**
  - 一般性的初级XML语法, 所有XML文档都遵循
  - 由文档类型定义[DTD]或XML构架[Schema]规定, 针对特定的XML标签集
- **XML文档组成:**
  - 数据元素
  - 标记声明(XML解析器的指令)
  - 处理指令(提供给处理文档中数据的应用程序)

7.2 XML语法

- **所有XML文档都从一个XML声明开始**
  - <?xml version = "1.0" encoding = "utf-8"?>
- **XML名称(元素或属性)**
  - 需以字母或下划线开头
  - 可以包含数字,连字符和句点
  - 无长度限制
  - 大小写敏感(body和Body不同)
- **XML语法和XHTML类似(注释)**
- **XML文档都定义一个根元素(如: html)**

7.2 XML语法

- 结构良好的(well-formed)文档

```
<?xml version = "1.0" encoding = "utf-8" ?>
 <ad>
   <year> 1960 </year>
   <make> Cessna </make>
   <model> Centurian </model>
   <color> Yellow with white trim </color>
   <location>
     <city> Gulfport </city>
     <state> Mississippi </state>
   </location>
 </ad>
```

7.2 XML语法

- **编写XML文档会碰到的两种选择**
  - 为一个元素增加一种新属性
  - 定义一个嵌套的元素
- **有时，嵌套标签要比属性好**
  - 属性不能描述结构，而嵌套标签可以描述结构和未来的扩展(日期: 年->年月日)
- **有时，应使用属性**
  - 为了标识元素的编号和名称，如XHTML中id和name

7.2 XML语法

```
<!-- A tag with one attribute -->
<patient name = "Maggie Dee Magpie">
  ...
</patient>

<!-- A tag with one nested tag -->
<patient>
  <name> Maggie Dee Magpie </name>
  ...
</patient>

<!-- A tag with one nested tag, which contains three nested tags -->
<patient>
  <name>
    <first> Maggie </first>
    <middle> Dee </middle>
    <last> Magpie </last>
  </name>
  ...
</patient>
```

第3种较好

7.3 XML文档结构

- **一个XML文档常伴有两个辅助性文件**
  - 一个用来规定标签集和结构的语法规则
    - 文档类型定义[DTD]或XML构架[Schema]
  - 一个包含一个样式表用于显示
    - 层叠样式表[CSS]或XSLT样式表
  - 后缀: *.dtd, *.xsd, *.css, *.xsl

- **XML文档由一个或多个实体组成，这些实体是一些逻辑上相关的信息集合.**

7.3 XML文档结构

- **文档实体结构的原因**
  - 大文档可分解为小文档以方便管理
  - 相同数据可重复使用, 同时避免数据不一致
  - 二进制实体
- **非二进制实体引用,用其值代替; 二进制实体引用,用相应的应用程序来处理**
- **实体名称规则**
- **实体引用格式: &entity_name;**
- **预定义的实体: (P39 表2-1)**
  - < > & " '

7.4 文档类型定义

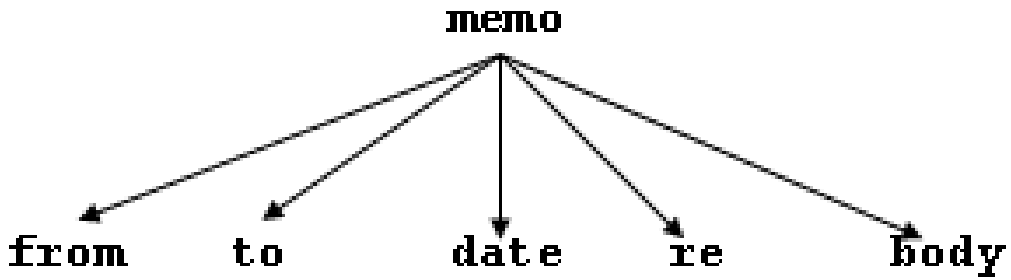
- DTD是一个结构规则的集合，这些规则称为声明
- DTD定义了文档可用的元素和出现规则
- 目的：为一组XML文档定义标记语言，提供标准格式规范
- 内部DTD和外部DTD
- DTD声明格式：<!keyword …>
- 4种关键字：ELEMENT, ATTLIST, ENTITY, NOTATION

7.4 文档类型定义

❖元素声明

- 定义元素的名称和元素间的结构
- 内部节点: 描述其子元素
- 叶节点: 描述其字符形式(内容类型)

<!ELEMENT memo (from, to, date, re, body)>



7.4 文档类型定义

❖元素声明

- 内部节点: 修改符: +, *, ?(P230 表7-1)

<!ELEMENT person (parent+, age, spouse?, sibling*)>

- 叶节点: 内容类型: PCDATA, EMPTY, ANY

<!ELEMENT name (#PCDATA)>

7.4 文档类型定义

❖属性声明

- 一般格式:

  <!ATTLIST el_name at_name at_type [default]>

- 类型: 10种, CDATA
- 默认值: 具体值或要求, P231 表7-2
  - 具体值
  - #FIXED 具体值
  - #REQUIRED
  - #IMPLIED

```
<!ATTLIST car doors CDATA "4">
<!ATTLIST car engine_type CDATA #REQUIRED>
<!ATTLIST car price CDATA #IMPLIED>
<!ATTLIST car make CDATA #FIXED "Ford">

<car doors = "2" engine_type = "V8">
  ...
</car>
```

7.4 文档类型定义

❖实体声明

- 普通实体: XML文档中可用
- 参数实体: DTD中可用[%]
- 一般格式:

```
<!ENTITY [%] entity_name "entity_value">
```

- 例: &jfk;

```
<!ENTITY jfk "John Fitzgerald Kennedy">
```

- 外部文本实体

```
<!ENTITY entity_name SYSTEM "file_location">
```

7.4 文档类型定义

❖一个DTD的示例: planes.dtd

```
1    <?xml version = "1.0"  encoding = "utf-8" ?>
2
3    <!-- planes.dtd - a document type definition for
4                         the planes.xml document, which specifies
5                         a list of used airplanes for sale  -->
6
7    <!ELEMENT planes_for_sale (ad+)>
8    <!ELEMENT ad (year, make, model, color, description,
9                  price?, seller, location)>
10   <!ELEMENT year (#PCDATA)>
11   <!ELEMENT make (#PCDATA)>
12   <!ELEMENT model (#PCDATA)>
13   <!ELEMENT color (#PCDATA)>
14   <!ELEMENT description (#PCDATA)>
15   <!ELEMENT price (#PCDATA)>
16   <!ELEMENT seller (#PCDATA)>
17   <!ELEMENT location (city, state)>
18   <!ELEMENT city (#PCDATA)>
19   <!ELEMENT state (#PCDATA)>
```

一次或多次

零次或多次

?叶节点,内部节点,树状图

7.4 文档类型定义

❖一个DTD的示例

```
21      <!ATTLIST seller phone CDATA #REQUIRED>
22      <!ATTLIST seller email CDATA #IMPLIED>
23
24      <!ENTITY c "Cessna">
25      <!ENTITY p "Piper">
26      <!ENTITY b "Beechcraft">
```

7.4 文档类型定义

- **内部和外部DTD(使用DOCTYPE声明)**
  - 内部DTD: DTD包含在XML代码中

```
<!DOCTYPE root_name [
  ...
]>
```

  - 外部DTD: DTD在一个外部文件中定义

<!DOCTYPE *XML_doc_root_name* SYSTEM *"DTD_file_name"*>

7.4 文档类型定义

planes.xml

```
1  <?xml version = "1.0"  encoding = "utf-8"?>
2
3  <!-- planes.xml - A document that lists ads for
4         used airplanes -->
5
6  <!DOCTYPE planes_for_sale SYSTEM "planes.dtd">
7
8  <planes_for_sale>
9         <ad>
10           <year> 1977 </year>
11           <make> cessna </make>
12           <model> Skyhawk </model>
13           <color> Light blue and white </color>
14           <description> New paint, nearly new interior,
15         685 hours SMOH, full IFR King avionics
16               </description>
17           <seller phone = "555-222-3333"> Skyway Aircraft </seller>
18           <location>
19              <city> Rapid City, </city>
20              <state> South Dakota </state>
21           </location>
22        </ad>
```

7.5 名称空间

- 同一XML文档可用多种标记语言(标签集)编写
  - XHTML的<table>与家具标记语言<table>
- 为了避免标签冲突，使用名称空间来区分不同
  - 家具标记语言<my:table>
  - XHTML的<html:table>或<table>
- 声明名称空间通常在文档的根元素中完成

```
<html xmlns:html = "http://www.w3.org/1999/xhtml"
      xmlns:my   = "http://www.my.com/names" >
```

默认名称空间

```
<html xmlns      = "http://www.w3.org/1999/xhtml"
      xmlns:my   = "http://www.my.com/names" >
```

7.6 XML构架

- **DTD的缺点:**
  - 语法与XML不同, 容易引起混乱
  - 不能对数据格式做限制
- **XML构架[Schema]是DTD的替代技术**
- **构架标签集对应的名称空间(xmlns:xsd)**
  - http://www.w3.org/2001/XMLSchema
- **构架文档根元素为schema**
- **构架文档定义的名称空间(标签集)的名称指定**

```
targetNamespace =
    "http://cs.uccs.edu/planeSchema"
```

7.6 XML构架

- 一个完整的构架起始标签的例子

```
<xsd:schema

<!-- Namespace for the schema itself(构架标签) -->
  xmlns:xsd = "http://www.w3.org/2001/XMLSchema"

<!-- Namespace where elements defined here will be placed -->
  targetNamespace ="http://cs.uccs.edu/planeSchema"

<!-- Default namespace for this document(新定义的标签) -->
  xmlns = "http://cs.uccs.edu/planeSchema"

<!-- Next, specify non-top-level elements to
    be in the target namespace -->
  elementFormDefault = "qualified">
```

7.6 XML构架

- 定义构架实例(使用构架定义的标签集创建文档)
  - 文档根元素须指定的属性
    - 缺省名称空间(targetNamesapce, 新标签集的名称)
    - 实例的标准名称空间(XMLSchema-instance)
    - 指明所使用的构架文档(schemaLocation,两个值)

```
<planes
  xmlns = "http://cs.uccs.edu/planeSchema"
  xmlns:xsi = http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation = "http://cs.uccs.edu/planeSchema
                                planes.xsd" >
```

7.6 XML构架

❖数据类型概述

- XML构架定义了44种数据类型
  - 原始(primitive): string, Boolean, float, …
  - 派生(derived): byte, long, decimal, …
- 用户自定义类型(派生类)=基类型+约束
- 自定义类型
  - 简单: 无属性, 无子元素
  - 复杂: 可包含属性和子元素
- 类型命名或匿名
- 元素局部和全局

7.6 XML构架

❖定义简单类型

- 定义元素
  - <xsd:element name = "engine" type = "xsd:string" />
  - 使用: <engine> … </engine>
  - 默认值: default; 常量值: fixed

- 用户自定义类型firstName

```
<xsd:simpleType name = “firstName" >
  <xsd:restriction base = "xsd:string" >
    <xsd:maxLength value = “10" />
  </xsd:restriction>
</xsd:simpleType>
```

7.6 XML构架

❖ 复杂类型

- 多种类别, 仅纯元素元素(element-only element)
- 使用complexType标签定义
- 子元素在有序组(sequence)或无序组(all)中

```
<xsd:complexType name = "sports_car" >
  <xsd:sequence>
    <xsd:element  name = "make" type = "xsd:string" />
    <xsd:element  name = "model " type = "xsd:string" />
    <xsd:element  name = "engine" type = "xsd:string" />
    <xsd:element  name = "year" type = "xsd:string" />
  </xsd:sequence>
</xsd:complexType>
```

7.6 XML构架

❖planes.xsd

```
1  <?xml version = "1.0"  encoding = "utf-8" ?>
2  <!-- planes.xsd   A simple schema for planes.xml -->
3  <xsd:schema
4      xmlns:xsd = "http://www.w3.org/2001/XMLSchema"
5      targetNamespace = "http://cs.uccs.edu/planeSchema"
6      xmlns = "http://cs.uccs.edu/planeSchema"
7      elementFormDefault = "qualified">
8
9      <xsd:element name = "planes">
10         <xsd:complexType>
11             <xsd:all>
12                 <xsd:element name = "make"
13                              type = "xsd:string"
14                              minOccurs = "1"
15                              maxOccurs = "unbounded"/>
16             </xsd:all>
17         </xsd:complexType>
18     </xsd:element>
19 </xsd:schema>
```

7.6 XML构架

❖planes1.xml

```
1  <?xml version = "1.0"  encoding = "utf-8"?>
2
3  <!-- planes1.xml
4       A simple XML document for illustrating a schema
5       The scema is in planes.xsd
6       -->
7
8  <planes
9    xmlns = "http://cs.uccs.edu/planeSchema"
10   xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
11   xsi:schemaLocation = "http://cs.uccs.edu/planeSchema
12                         planes.xsd">
13     <make> Cessna </make>
14     <make> Piper </make>
15     <make> eechcraft </make>
16 </planes>
```

7.7 显示没有格式化的XML文档

- 未格式XML文档,浏览器仅能显示代码

该 XML 文件并未包含任何关联的样式信息。文档树显示如下。

```
- <!--
     planes.xml - A document that lists ads for
          used airplanes
  -->
- <!--
     <?xml-stylesheet  type = "text/css"  href = "planes.css" ?>
  -->
- <planes_for_sale>
  + <ad></ad>
  - <ad>
      <year> 1965 </year>
      <make> Piper </make>
      <model> Cherokee </model>
      <color> Gold </color>
    - <description>
        240 hours SMOH, dual NAVCOMs, DME, new Cleveland brakes, great shape
      </description>
      <seller phone="555-333-2222"> John Seller </seller>
    - <location>
        <city> St. Joseph, </city>
        <state> Missouri </state>
      </location>
    </ad>
  </planes_for_sale>
```

7.8 通过CSS显示XML文档

- 针对**XML**文档的**CSS**样式表由元素名称和相应样式属性组成
- 属性**display:**
  - 行显示(inline,默认)或换行块显示(block)

```
<!-- planes.css - a style sheet for the planes.xml document -->

ad { display: block; margin-top: 15px; color: blue;}
year, make, model { color: red; font-size: 16pt;}
color {display: block; margin-left: 20px; font-size: 12pt;}
description {display: block; margin-left: 20px; font-size: 12pt;}
seller { display: block; margin-left: 15px; font-size: 14pt;}
location {display: block; margin-left: 40px; }
city {font-size: 12pt;}
state {font-size: 12pt;}
```

7.8 通过CSS显示XML文档

❖样式表引用格式

```
<!DOCTYPE planes_for_sale SYSTEM "planes.dtd">
<?xml-stylesheet  type = "text/css"  href = "planes.css" ?>
<planes_for_sale>
        <ad>
            <year> 1977 </year>
```

```
1977 cessna Skyhawk
  Light blue and white
  New paint, nearly new interior, 685 hours SMOH, full IFR King avionics
 Skyway Aircraft
    Rapid City, South Dakota
1965 Piper Cherokee
  Gold
  240 hours SMOH, dual NAVCOMs, DME, new Cleveland brakes, great shape
 John Seller
    St. Joseph, Missouri
```

```
<link rel = "stylesheet"  type = "text/css"
 href = "http://www.wherever.org/termpaper.css">
</link>
```

7.9 XSLT样式表

- 可扩展样式语言**XSL**用于**XML**文档转换和表现
  - XSLT – Transformations
  - XPath – XML路径语言
  - XSL-FO – 格式对象
- **XSLT**样式表将**XML**文档转为各种格式**(XHTML)**
- 连接**XSLT**样式表和**XML**文档的处理指令

```
<?xml-stylesheet type = "text/xsl" href = "XSLT style sheet"?>
```

7.9 XSLT样式表

- **XSLT样式表为XML文件, 根元素为stylesheet**

```
<xsl:stylesheet
    xmlns:xsl = "http://www.w3.org/1999/XSL/Format"
    xmlns = "http://www.w3.org/1999/xhtml" >
```

- **模板定义(根元素必须):**
  - <xsl:template match = “元素名称">
- **两种XSLT元素类型**
  - 一种类型: 包含实际内容(XHTML元素)

```
<span style = "font-size: 14">Happy Easter!</span>
```

  - 另种类型: 从XML文档中复制内容

```
<xsl:value-of select = "year" />
```

7.9 XSLT样式表

❖xslplane.xml

```
1  <?xml version = "1.0"  encoding = "utf-8" ?>
2  <!-- xslplane.xml -->
3  <?xml-stylesheet type = "text/xsl" href = "xslplane.xsl" ?>
4  <plane>
5     <year> 1977 </year>
6     <make> Cessna </make>
7     <model> Skyhawk </model>
8     <color> Light blue and white </color>
9  </plane>
```

Airplane Description

Year: 1977
Make: Cessna
Model: Skyhawk
Color: Light blue and white

7.9 XSLT样式表

❖xslplane.xsl

```
<?xml version = "1.0"  encoding = "utf-8" ?>
<xsl:stylesheet version = "1.0"
                xmlns:xsl = "http://www.w3.org/1999/XSL/Transform"
                xmlns = "http://www.w3.org/1999/xhtml">
   <xsl:template match = "plane">
     <html><head><title> Style sheet for xslplane.xml </title>
     </head><body>
     <h2> Airplane Description </h2>
     <span style = "font-style: italic; color: blue;"> Year: </span>
     <xsl:value-of select = "year" /> <br />
     <span style = "font-style: italic; color: blue;"> Make: </span>
     <xsl:value-of select = "make" /> <br />
     <span style = "font-style: italic; color: blue;"> Model: </span>
     <xsl:value-of select = "model" /> <br />
     <span style = "font-style: italic; color: blue;"> Color: </span>
     <xsl:value-of select = "color" /> <br />
     </body></html>
   </xsl:template>
</xsl:stylesheet>
```

7.10 XML处理器

- 用途
  - 检测文档是否符合基本语法(结构良好)
  - 替换文档中对实体的引用
  - 从DTD和构架中复制默认值到文档
  - 验证文档是否符合DTD和构架规范(解析器)
- **SAX(Simple API for XML)**
  - 基于事件处理机制
- **DOM**
  - 基于DOM树结构

7.11 Web服务

❖Web Service

- 不同软件之间互相连接和交互操作的技术
- 提供的是软件服务而不是文档
- 类似概念: RPC, DCOM, CORBA
- SOAP: 基于HTTP, 是一个XML标签集, 定义了消息和RPC格式

小结

- **简介: 元标记语言, SGML, XML**
- **XML语法: 两种级别**
  - 基本语法: 开头,名称,注释,根元素,标签or属性
- **XML文档结构: 辅助文件, 实体(引用,二进制)**
- **名称空间: xmlns, 根元素**
- **DTD: 关键字(元素,属性,实体), 外部**
- **XML构架:**
  - 构架:xmlns:xsd, schema, targetNamespace
  - 实例:xmlns, xmlns:xsi, xsi:schemaLocation
  - 数据类型, 简单类型, 复杂类型
- **CSS显示: display属性, 引用**
- **XSLT样式表: 引用, stylesheet, 模板,元素类型**
- **XML处理器和Web服务**

课后

- 仔细阅读课本
- 阅读和测试示例代码
- 复习题(书面)
  - 复习题4: XML的主要目标是什么?
  - 复习题12: DTD的主要作用是什么?
  - 复习题26: XML名称空间是什么?
  - 复习题27: 构架相对与DTD的两个优点?
  - 复习题37: 如何用XSLT样式表处理XML文档?

YTU

Thank You !

烟台大学计算机学院
陈智育