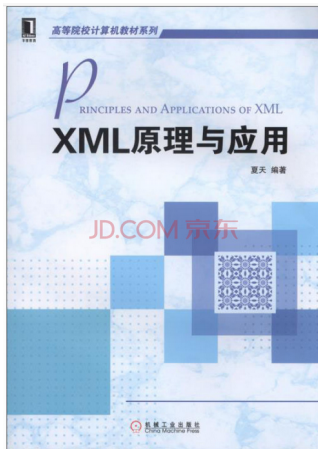


# XML 原理与应用

夏天

中国人民大学

# 幻灯片内容基于《XML 原理与应用》一书



夏天. XML 原理与应用. 机械工业出版社, 2015-03.

## CH2 XML 基本语法



# 本章学习目标

- 掌握 XML 的文档组成结构
- 掌握 XML 声明、处理指令、注释、元素、属性、命名空间等基本概念
- 能够编写格式良好的 XML 文档

- XML 文档结构
- 元素
  - 元素和标记
  - 元素的内容
  - 元素的嵌套
- 属性
  - 属性的语法形式
  - 属性的使用场景
  - 属性的命名规范
  - 属性值
- 命名空间
- XML 文档规范级别

## 2.1 XML 文档结构

XML 本身侧重于数据的表示，并通过 CSS、XSLT 等技术把数据以指定的格式进行呈现，实现数据表示与呈现的分离。

# 示例 XML 文档 I

```
1 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2 <?xml-stylesheet type="text/css" href="2-1.css"?>
3 <!-- 个人通讯录 -->
4 <addressList>
5   <group name="同学">
6     <person>
7       <name>吴泽林</name>
8       <birthday>1985-09-11</birthday>
9       <mobile>135-1234-5678</mobile>
10      <telephone>68689999</telephone>
11      <email>wuzelin@163.com</email>
12      <address>漳州</address>
13    </person>
14  </group>
15  <group name="网友">
16    <person>
```

## 示例 XML 文档 II

```
17      <name>罗中华</name>
18      <mobile>136-1111-1118</mobile>
19      <telephone>22339999</telephone>
20      <email>luozh@163.com</email>
21      <address>北京</address>
22  </person>
23 </group>
24 </addressList>
25
26 <!-- 处于篇幅考虑，书中源代码内容有删减，读者可以通过附带源码
27 查看更多信息 -->
```



# 文档构成

一个 XML 文档由序言、主体和尾声三部分构成

- 序言 (Prolog): 从 XML 声明到文档根元素开始前的部分, 包括 XML 声明、注释和处理指令等, 序言是可选的
- 文档主体 (Body): 文档根元素及其所包含的内容, 每一个 XML 文档有且仅有一个根元素
- 尾声 (Epilog): 文档根元素后面的部分, 可以包含注释、处理指令以及空白信息, 一般不出现
- 一个 XML 文档最基本的语法要素包括: XML 声明、处理指令、注释和 XML 元素.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

- version
- encoding
- standalone

# 处理指令

语法形式：<? 处理指令名称处理指令信息?>

E.g. <?topdf path="/pdf-files" ?>

- 处理指令 PI ( Processing Instruction ) 是 XML 文档中为 XML 处理程序提供必要的处理信息的指令描述。XML 解析器会把它原封不动地传递给 XML 应用程序，由应用程序来根据该指令进行必要处理，或者再把它原封不动地传递给下一个应用程序。
- 处理指令如何解释完全由外部应用程序决定
- xml-stylesheet 问题

语法形式：<!-- 注释正文 -->

- 注释本身不能放入到标记之内
- 注释正文中不能出现连续的“-”符号
- 注释不能以“->”符号串结尾，也不能放到 XML 文档声明之前
- 可以对标记进行注释
- 注释不能嵌套使用

# 注释本身不能放入到标记之内

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <document>
3   <head <!-- This is the heading element -->>
4     注释测试例子
5   </head>
6   <body>正文内容</body>
7 </message>
8 </document>
```

# 注释正文中不能出现连续的“-”符号

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <document>
3   <head>注释测试例子</head>
4   <!--这是正文内容--夏天-->
5   <body>正文内容</body>
6 </message>
7 </document>
```

# 注释不能以“—>”结尾，不能放到文档声明之前

```
1 <!-- 该注释后面是XML声明 -->
2 <?xml version="1.0" encoding="UTF-8"?>
3 <document>
4   <head>注释测试例子</head>
5   <!--这是正文内容-夏天--->
6   <body>正文内容</body>
7   </message>
8 </document>
```

# 可以对标记进行注释

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <document>
3   <!--
4   <head>注释测试例子</head>
5   <body>正文内容</body>
6   </message>
7   -->
8 </document>
```



# 注释不能嵌套使用

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <document>
3   <!--
4   <head>注释测试例子</head>
5   <!-- 这是正文内容-夏天 -->
6   <body>正文内容</body>
7   </message>
8   -->
9 </document>
```

## 2.2 元素

元素 (Element) 是 XML 文档的基本构成单元，它表示了文档的结构和文档中包含的数据，格式良好的 XML 文档必须拥有一个唯一的根元素。

- 元素和标记
- 元素内容
- 元素的嵌套

# 元素和标记

元素的基本语法形式：`< 标记 > 元素内容 </标记 >`

标记的基本语法形式：`< 标记名 [[属性名 1=" 属性值 1"] [属性名 2=" 属性值 2"] ...]>`

- 标记区分大小写
- 标记必须配对出现
- 标记首字符以字母、下划线“\_”、冒号“:”以及 Unicode 字符集中的某一部分开始，支持汉字；其他部分可以是字母、数字、下划线“\_”、连字符“-”、句点“.”、冒号“:”，以及 Unicode 字符集中的某一部分
- 标记不能包含空格符号或斜线符号“/”

# 标记示例

- 有效的非空标记示例：`<address> 漳州 </address>`
- 有效的空标记示例：``
- 无效标记示例：`<address> 漳州 </Address>`
- 无效标记示例：`<p> 段落内容之一 <p> 段落内容之二`

## 2.2.2 元素的内容

元素内容可以包括被解析字符数据 (Parsed Character Data)、字符数据 CDATA 段、以及处理指令和注释

# 被解析字符数据

被解析字符数据部分可以是任意合法的 Unicode 字符，但是不能包含被用作特殊用途的字符，例如标记的开始符号“<”。

通过实体转义的方式解决特殊字符问题

字符	引用方式
<	&lt;
>	&gt;
&	&amp;
'	&quot;
”	&apos;

# 给书名加上书名号

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <books>
3   <book>
4     <title>XML原理与应用</title>
5     <authro>夏天</authro>
6   </book>
7 </books>
```

# CDATA 段

基本语法形式：`<![CDATA[ 文本内容]]>`

有利于处理包含大量特殊符号的文本当作普通文本处理的情况



# 采用实体转义方式

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <demo>
3   <content>
4     &lt;group name="同学"&gt;
5       &lt;person&gt;
6         &lt;name&gt;吴泽林&lt;/name&gt;
7         &lt;birthday&gt;1985-09-11&lt;/birthday&gt;
8         &lt;mobile&gt;135-1234-5678&lt;/mobile&gt;
9         &lt;telephone&gt;68689999&lt;/telephone&gt;
10        &lt;email&gt;wuzelin@163.com&lt;/email&gt;
11        &lt;address&gt;漳州&lt;/address&gt;
12      &lt;/person&gt;
13    &lt;/group&gt;
14  </content>
15 </demo>
```

# 采用 CDATA 方式

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <demo>
3   <content>
4     <![CDATA[
5       <group name="同学">
6         <person>
7           <name>吴泽林</name>
8           <birthday>1985-09-11</birthday>
9           <mobile>135-1234-5678</mobile>
10          <telephone>68689999</telephone>
11          <email>wuzelin@163.com</email>
12          <address>漳州</address>
13        </person>
14      </group>
15    ]]>
16  </content>
17 </demo>
```

## 2.2.3 元素的嵌套

- 元素描述了 XML 文档的逻辑结构，对于一个复杂的文档来说，单纯用一组并列的元素是无法准确刻画其结构关系的，通过在元素中嵌套子元素可以更好地描述 XML 文档数据之间的语义关联性。同时，也可以根据元素之间的嵌套关系把整个 XML 文档看成是一棵具有严格层次关系的树，方便数据的查询、定位和处理。
- XML 的元素之间虽然可以嵌套使用，但不能交叉重叠嵌套，例如以下写法是错误的：
- `<h1> 中国 <b> 古代文明 </h1></b>`

## 2.3 属性

属性是元素的可选组成部分，用于对元素及其内容的附加信息进行说明

- 属性的语法形式
- 属性的使用场景
- 属性的命名规则
- 属性值

# 属性的语法形式

- 非空元素如下：

- `< 标记名 属性名 1=" 属性值 1" 属性名 2=" 属性值 2" ... ] > 元素内容 </标记名 >`
- `< 标记名 属性名 1=' 属性值 1' 属性名 2=' 属性值 2' ... ] /> 元素内容 </标记名 >`

- 空元素如下：

- `< 标记名 属性名 1=" 属性值 1" 属性名 2=" 属性值 2" ... />`
- `< 标记名 属性名 1=' 属性值 1' 属性名 2=' 属性值 2' ... />`

- E.g. `<person group=" 同学"></person>`

# 属性的使用场景

- 属性的使用

- 与 XML 文档读者无关的简单信息建议使用属性
- 与 XML 文档有关但是与 XML 文档的内容无关的简单信息建议使用属性
- 元素与属性的具体使用有争论，需要灵活掌握

- 属性的不足

- 属性不能包含多个值（元素可以）
- 属性不容易被扩充（方便将来的修改）
- 属性不能描述结构（子元素可以）
- 属性更难被程序代码处理
- 属性值不容易进行 DTD 测试

# 属性的命名规则

- 属性的命名规则和元素相似
- 同一个元素内不能包含多个同名属性，例如以下反例：  
`<todo date="2050-05-05" date="2150-05-05" />`

# 属性值

- 属性值必须用单引号或双引号括起来，不能混用
- 双引号括起来的属性值中可以出现单引号，反之亦然
- 属性值不能直接包括"<"符号，但可以包含">"
- 属性值不能直接包括"&"符号，除非引用实体



# 非法属性值示例

- `<todo status="important("special")"/>`  
<!-- 用双引号括起来的字符串中不能直接包含双引号 -->
- `<todo status="important > urgent"/>`  
<!-- 属性值中不能直接包含小于号 -->
- `<todo status="important & urgent"/>`  
<!-- 属性值中不能直接包含"&" -->
- `<todo status="important'"/>`  
<!-- 属性值不能用一个单引号和一个双引号括起来 -->
- `<todo status=important/>`  
<!-- 属性值必须用引号括起来 -->

# 命名空间问题

人力资源维护的员工信息：

```
1 <?xml version="1.0"?>
2 <employees>
3   <employee>
4     <name>金成</name>
5     <birthday>1982-08</birthday>
6     <hiredate>2005-09</hiredate>
7     <major>软件开发</major>
8     <projects>
9       <project name="项目A" role="工程师" year="2006"/>
10      <project name="项目B" role="架构师" year="2007"/>
11    </projects>
12  </employee>
13 </employees>
```

# 命名空间问题

你认为该员工表现优秀，应该增加薪水以示奖励，此时，可在 XML 文件中增加一条评论：“表现优秀，增加 10% 薪水”

```
1  <?xml version="1.0"?>
2  <employees>
3    <employee>
4      <name>金成</name>
5      <birthday>1982-08</birthday>
6      <hiredate>2005-09</hiredate>
7      <major>软件开发</major>
8      <comment>表现优秀，增加10%薪水</comment>
9      <projects>
10       <project name="项目A" role="工程师" year="2006"/>
11       <project name="项目B" role="架构师" year="2007"/>
12     </projects>
13   </employee>
14 </employees>
```

# 命名空间问题

人力资源管理人员根据老板的评论，加入了自己的注释

```
1 <?xml version="1.0"?>
2 <employees>
3   <employee>
4     <name>金成</name>
5     <birthday>1982-08</birthday>
6     <hiredate>2005-09</hiredate>
7     <major>软件开发</major>
8     <comment>表现优秀，增加10%薪水</comment>
9     <comment>薪水已调整</comment>
10    <projects>
11      <project name="项目A" role="工程师" year="2006"/>
12      <project name="项目B" role="架构师" year="2007"/>
13    </projects>
14  </employee>
15 </employees>
```

# 问题

由于老板和人力资源管理人员都可能向原 XML 文件中增加同样的标记 `<comment>`，日后再来阅读该文件时，便极有可能增加语义混淆：

- 哪一些评论是老板增加的？
- 哪一些是人力部门工作人员增加的？

# 使用命名空间 I

语法形式：xmlns:命名空间前缀="命名空间名"

```
1 <?xml version="1.0"?>
2 <hr:employees xmlns:hr="http://www.demo.org/human_resources"
3   xmlns:boss="http://www.demo.org/big_boss">
4   <hr:employee>
5     <hr:name>金成</hr:name>
6     <hr:birthday>1982-08</hr:birthday>
7     <hr:hiredate>2005-09</hr:hiredate>
8     <hr:major>软件开发</hr:major>
9     <boss:comment>表现优秀，增加10%薪水</boss:comment>
10    <hr:comment>薪水已调整</hr:comment>
11    <hr:projects>
```

## 使用命名空间 II

```
12      <hr:project name="项目A" role="工程师" year="2006"/>
13      <hr:project name="项目B" role="架构师" year="2007"/>
14  </hr:projects>
15 </hr:employee>
16 </hr:employees>
```

# 使用命名空间

根据教材进一步了解默认命名空间和命名空间的作用域



## 2.5 XML 文档的规范级别

- 格式良好的 XML 文档
  - 符合 XML 语法要求
- 有效的 XML 文档
  - 格式良好，满足约束条件
- 规范化的 XML 文档
  - 判定两个 XML 文件在信息集合的角度看是否一致的方法

# END

