

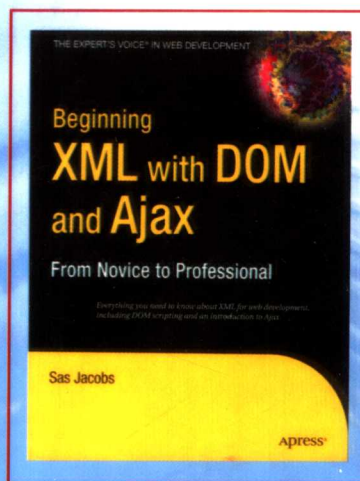
Beginning XML with DOM and Ajax

XML 基础教程

入门、DOM、Ajax 与 Flash

[美] Sas Jacobs 著
许劲松 杨波 周斌 译

- 现代 XML 教程
- 涵盖 Ajax 和 Flash 开发
- 新一代 Web 开发人员必备



TURING

图灵程序设计丛书

Web开发系列

XML基础教程

入门、DOM、Ajax与Flash

Beginning XML with DOM and Ajax

[美] Sas Jacobs 著

许劲松 杨波 周斌 译

人民邮电出版社

北京

图书在版编目 (CIP) 数据

XML 基础教程 / (美) 雅可布斯, (Jacobs, S.) 著; 许劲松, 杨波, 周斌译.

—北京: 人民邮电出版社, 2007.7

(图灵程序设计丛书)

ISBN 978-7-115-15942-7

I. X... II. ①雅...②许...③杨...④周... III. 可扩展标记语言, XML—程序设计—教材
IV. TP312

中国版本图书馆 CIP 数据核字 (2007) 第 030419 号

内 容 提 要

本书全面讲述了XML及其在Web开发领域中的作用, 同时介绍了一些特定的XML词汇以及相关的XML推荐标准。书中首先解释了XML并介绍了XML文档的不同组成部分; 其次讲解了XML应用程序客户端的处理方法, 如何使用CSS和XSLT对XML文档进行显示和转换, 如何使用JavaScript操作XML文档等内容; 然后介绍了如何在服务器端处理XML; 最后深入分析了两个XML应用程序: 新闻应用程序和社区天气门户系统。

本书适合各层次的Web开发人员阅读, 不仅为不太熟悉XML应用程序的开发者提供了一个很好的起点, 也使有经验的开发者可以更深入地理解创建XML应用程序的过程。

图灵程序设计丛书

XML 基础教程: 入门、DOM、Ajax 与 Flash

- ◆ 著 [美] Sas Jacobs
- 译 许劲松 杨波 周斌
- 责任编辑 傅志红
- ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号
邮编 100061 电子函件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
三河市海波印务有限公司印刷
新华书店总店北京发行所经销
- ◆ 开本: 800 × 1000 1/16
印张: 21.75
字数: 511 千字 2007 年 7 月第 1 版
印数: 1—5 000 册 2007 年 7 月河北第 1 次印刷

著作权合同登记号 图字: 01-2006-5094 号

ISBN 978-7-115-15942-7/TP

定价: 49.00 元

读者服务热线: (010)88593802 印装质量热线: (010)67129223

版 权 声 明

Original English language edition, entitled *Beginning XML with DOM and Ajax* by Sas Jacobs, published by Apress L.P., 2560 Ninth Street, Suite 219, Berkeley, CA 94710 USA.

Copyright © 2006 by Sas Jacobs. Simplified Chinese-language edition copyright © 2007 by Posts & Telecom Press. All rights reserved.

本书中文简体字版由 Apress L.P. 授权人民邮电出版社独家出版。未经出版者书面许可，不得以任何方式复制或抄袭本书内容。

版权所有，侵权必究。

译者序

XML 让 Web 应用程序焕然一新！

在开发者逐步趋向使用浏览器作为 Web 应用程序的通用客户端之后，浏览器所承担的任务的复杂性也在不断升级。与此同时，Web 应用程序相对于传统专用客户端应用程序而言，其用户使用体验的不足也一直为人所诟病。基于浏览器的 Web 应用程序的发展一直面临着两方面的问题：一是如何使其更适合于实现更为复杂的较大规模的应用；二是如何给用户提供更好的使用体验。

在上述两个方面，XML 都发挥了极其重要且不可替代的作用。在程序的结构方面，XML 成功地扮演了“胶水”和“信使”的角色，这使程序的层次划分更为清晰而自然。这种结构层次的细化以及层次间耦合度的降低，使 Web 应用程序成功地进入了更广泛的应用领域，并显示了实现更为复杂的应用逻辑的强大能力。而在提高用户体验方面，由于各种基于 XML 的相关 Web 词汇的出现和应用，尤其是以 XML 为重要基础的 Ajax 技术和方法的日益普及，使浏览器表现出了毫不逊色于传统专用客户端的表现和交互能力。

可以肯定地说，今天的 Web 应用程序已经完全离不开 XML。熟悉 XML 的概念和规则，并将其完美地应用于各种技术，已经成为对 Web 开发者非常重要的基本要求。本书对于 XML 的讲解全面而深入，并且收集了大量对开发工作非常有价值的资源。对于有志于从事 Web 开发的人们来说，本书既是一本优秀的 XML 教材，同时也是一本较全面的 XML 参考手册。

许劲松

2007 年春

前 言

本书旨在为 XML（可扩展标记语言）应用程序的开发者提供“一站式服务”。书中解释了 XML 及其在 Web 开发领域中的作用，同时介绍了一些特定的 XML 词汇以及相关的 XML 推荐标准。

本书适用于所有层次的 Web 开发者。对于还不太熟悉 XML 应用程序的开发者来说，本书提供了一个很好的起点，并介绍了一些重要的客户端和服务器端开发技术。较有经验的开发者可以通过深入了解书中重要的编程技术，理解创建 XML 应用程序的过程。

本书开始部分对 XML 进行了解释并介绍了 XML 文档的不同组成部分。这部分介绍了一些相关的推荐标准，包括 DTD（文档类型定义）、XML 模式、CSS（层叠样式表）、XSLT（可扩展样式表语言转换）、XPath、XLink 和 XPointer 等，以及一些常用的 XML 词汇¹，如 XHTML（可扩展超文本标记语言）、MathML（数学标记语言）和 SVG（可伸缩矢量图形）。

本书中间部分讲解了在客户端处理 XML 应用程序的方法，展示了如何使用 CSS 和 XSLT 显示和转换 XML 文档，分析了当前的浏览器对 XML 的支持，以及如何使用 JavaScript 操作 XML 文档。这部分还介绍了一些 Ajax（异步 JavaScript 和 XML）方法。

本书最后部分说明了如何在服务器端处理 XML。其中，介绍了两种服务器端语言：PHP 5 和 .NET 2.0。最后两章深入分析了两个 XML 应用程序：新闻应用程序和社区天气门户系统。

本书包含了大量可供开发者在日常工作中使用的实例。这些示例代码可以从 Apress 网站 <http://www.apress.com> 的源代码区下载²。我希望本书能成为一部有价值的 XML 工具书，使读者认识到 XML 提供给 Web 开发者的巨大功能和灵活性。

致谢

我要感谢 Apress 出版社每个人对我的帮助和支持，他们在本书的写作过程中提出了许多好的建议。另外，我还要感谢我的家人在这个过程中对我的有力支持和给予的深深的爱。

1. “词汇（vocabulary）”是一个 XML 术语，指基于 XML 定义的各种语言。——编者注

2. 本书示例代码也可从图灵网站下载。——编者注

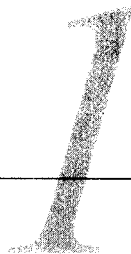
目 录

第 1 章 XML 简介.....	1	2.4 XPath.....	35
1.1 XML 是什么.....	1	2.4.1 XPath 表达式.....	36
1.1.1 XML 简史.....	2	2.4.2 标识特定节点.....	37
1.1.2 XML 的目标.....	2	2.4.3 包含计算和函数.....	37
1.2 XML 语法.....	3	2.4.4 XPath 小结.....	37
1.2.1 合式的文档.....	4	2.5 XML 链接.....	38
1.2.2 标签和元素的不同.....	5	2.5.1 简单链接.....	38
1.3 查看完整的 XML 文档.....	5	2.5.2 扩展链接.....	39
1.3.1 XML 文档的结构.....	6	2.5.3 XPointer.....	40
1.3.2 XML 的命名规则.....	7	2.5.4 XML 链接小结.....	41
1.3.3 XML 文档序言.....	7	2.6 小结.....	41
1.3.4 XML 文档元素中的段.....	9	第 3 章 Web 词汇.....	42
1.4 XML 的处理模型.....	13	3.1 XHTML.....	42
1.4.1 XML 处理方式.....	13	3.1.1 内容与表现的分离.....	43
1.4.2 DOM 解析.....	13	3.1.2 XHTML 构造规则.....	44
1.4.3 SAX 解析.....	14	3.1.3 XHTML 工具.....	53
1.4.4 为什么有两种处理模型.....	14	3.1.4 合式与有效的 XML 文档.....	54
1.5 一些 XML 工具.....	15	3.1.5 XHTML 的模块化.....	58
1.6 小结.....	15	3.2 MathML.....	58
第 2 章 XML 相关推荐标准.....	16	3.2.1 表现型 MathML.....	59
2.1 XML 命名空间的作用.....	16	3.2.2 内容型 MathML.....	61
2.1.1 在 XML 文档中加入命名空间.....	17	3.3 SVG.....	62
2.1.2 添加默认命名空间.....	18	3.3.1 矢量图形的形状.....	62
2.2 定义 XML 词汇.....	19	3.3.2 图像.....	64
2.2.1 文档类型定义.....	20	3.3.3 文本.....	65
2.2.2 XML 模式.....	23	3.3.4 汇总.....	66
2.2.3 DTD 与模式的比较.....	29	3.4 Web 服务.....	69
2.2.4 其他模式类型.....	29	3.4.1 WSDL.....	69
2.2.5 XML 词汇.....	30	3.4.2 SOAP.....	74
2.3 显示 XML.....	31	3.5 其他 Web 词汇.....	78
2.3.1 XML 和 CSS.....	31	3.5.1 RSS 和新闻提要.....	78
2.3.2 XSL.....	31	3.5.2 VoiceXML.....	78

3.5.3	SMIL	78	5.6.2	使用 HTML 命名空间来 定义链接	126
3.5.4	数据库输出格式	79	5.7	在 XML 文档中添加图像	127
3.6	小结	79	5.7.1	添加图像并在 Netscape 和 Firefox 中显示	127
第 4 章	客户端 XML	80	5.7.2	使用 CSS 添加图像	128
4.1	为什么使用客户端 XML	80	5.8	使用 CSS 添加内容	129
4.1.1	处理客户端 XML 内容	80	5.9	处理属性内容	131
4.1.2	在浏览器中为内容添加样式	81	5.9.1	在选择器中使用属性	131
4.1.3	在浏览器中处理 XML 内容	81	5.9.2	在文档中使用属性值	132
4.1.4	在 Flash 中处理 XML	83	5.10	小结	133
4.2	主流浏览器对 XML 的支持	83	第 6 章	XSLT 简介	135
4.2.1	W3C DOM	84	6.1	支持 XSLT 的浏览器	135
4.2.2	XSD	84	6.2	使用 XSLT 创建页眉和页脚	136
4.2.3	XSLT	84	6.2.1	XHTML、XSLT 和命名空间	138
4.2.4	微软 IE	85	6.2.2	创建 XSLT 样式表	138
4.2.5	Mozilla	90	6.2.3	样式表	138
4.2.6	Opera	91	6.2.4	转换<body>元素	139
4.2.7	Adobe Flash	92	6.2.5	应用转换	140
4.3	客户端和服务端之间的选择	93	6.2.6	添加页脚	140
4.3.1	使用客户端 XML	93	6.2.7	保持不变的转换	141
4.3.2	使用服务器端 XML	94	6.3	创建目录	141
4.4	小结	96	6.3.1	使用<xsl:for-each>选择每一个 行星	143
第 5 章	使用 CSS 显示 XML	97	6.3.2	增加新的行星	144
5.1	CSS 介绍	98	6.4	使用 XSLT 表现 XML	146
5.1.1	为什么使用 CSS	98	6.4.1	从 XHTML 转到 XML	146
5.1.2	CSS 规则	98	6.4.2	使用 XSLT 样式化 XML	147
5.2	使用 CSS 样式化 XHTML 文档	99	6.4.3	使用 XSLT 删除内容	149
5.3	使用 CSS 样式化 XML 文档	104	6.4.4	XSLT 中 XPath 的作用	149
5.3.1	附加样式表	105	6.5	包含图像	150
5.3.2	选择器	105	6.5.1	导入模板	151
5.4	使用 CSS 布局 XML	106	6.5.2	包含模板	152
5.4.1	W3C 框模型	106	6.6	XSLT 开发工具	152
5.4.2	使用 CSS 定位	109	6.7	小结	154
5.5	显示表格数据	120	第 7 章	高级客户端 XSLT 技术	155
5.5.1	使用 display 属性	121	7.1	XML 文档中的数据排序	155
5.5.2	使用浮动元素	122	7.2	使用 JavaScript 动态排序	159
5.5.3	表格行的合并	124			
5.6	XML 文档之间的链接	124			
5.6.1	Netscape 和 Firefox 中的 XLink	125			

7.3 添加扩展函数 (IE)	165	9.1.1 解释 Ajax 组件的作用	215
7.3.1 深入理解命名空间	166	9.1.2 XMLHttpRequest 对象	216
7.3.2 向样式表中添加扩展函数	167	9.2 汇总	223
7.3.3 支持 IE 之外的其他浏览器	169	9.2.1 使用 XMLHttpRequest 对象进行 用户名验证	223
7.4 使用命名模板	170	9.2.2 使用 Ajax 方法的通信录	225
7.5 使用 XSLT 生成 JavaScript	172	9.3 使用跨浏览器的库	230
7.5.1 XSLT 参数	174	9.4 其他 Ajax 框架和工具包	232
7.5.2 空白和模式	174	9.4.1 Backbase	232
7.5.3 onelinehtml 模板	176	9.4.2 Bindows	232
7.5.4 完成页面	177	9.4.3 Dojo	232
7.5.5 在 Mozilla 中生成 JavaScript	178	9.4.4 Interactive Website Framework	232
7.6 XSLT 提示和问题解决方法	179	9.4.5 qooxdoo	232
7.6.1 处理空白	179	9.5 对 Ajax 的批评	232
7.6.2 在 XSLT 中使用 HTML 实体	181	9.5.1 提供视觉提示	233
7.6.3 检查浏览器类型	181	9.5.2 更新界面	233
7.6.4 利用现有成果	182	9.5.3 预加载数据	233
7.6.5 XSLT 最佳实践	182	9.5.4 提供状态链接和启用返回按钮	234
7.7 小结	182	9.6 Ajax 最佳实践和设计原则	234
第 8 章 在浏览器中编写脚本	183	9.6.1 最小化服务器流量	234
8.1 W3C XML DOM	183	9.6.2 使用标准界面方法	235
8.1.1 主要的 DOM 接口	184	9.6.3 使用包装或库	235
8.1.2 MSXML 提供的额外功能	192	9.6.4 合理使用 Ajax	235
8.2 浏览器对 W3C DOM 的支持	194	9.7 小结	235
8.2.1 使用 xDOM 封装库	195	第 10 章 用 Flash 显示 XML	236
8.2.2 xDOM 注意事项	198	10.1 XML 类	237
8.3 使用 JavaScript 操作 DOM	199	10.1.1 加载 XML 文档	237
8.3.1 创建 DOM 文档对象和 装载 XML	200	10.1.2 XML 类	239
8.3.2 XSLT 操作	202	10.1.3 XMLNode 类	240
8.3.3 获取原始 XML 内容	204	10.2 在 Flash 中加载和显示 XML 内容	243
8.3.4 操作 DOM	204	10.2.1 在 Flash 中更新 XML 内容	245
8.4 实际应用	207	10.2.2 从 Flash 中发送 XML 内容	249
8.4.1 理解应用程序	207	10.3 使用 XMLConnector 组件	249
8.4.2 代码详细介绍	208	10.3.1 加载 XML 文档	250
8.4.3 处理大 XML 文档	212	10.3.2 数据绑定	251
8.5 小结	213	10.3.3 使用数据组件更新 XML 内容	253
第 9 章 在浏览器脚本中使用 Ajax 方法	214	10.4 Flash 安全	254
9.1 理解 Ajax	214	10.5 小结	254

第 11 章 服务器端 XML.....	255	第 12 章 实例：用 .NET 开发 XML 应用程序.....	282
11.1 服务器端与客户端的 XML 处理过程.....	255	12.1 理解应用程序.....	282
11.2 服务器端语言.....	256	12.1.1 设置环境.....	283
11.2.1 ASP.NET.....	256	12.1.2 新闻应用程序的组件.....	284
11.2.2 PHP.....	258	12.2 小结.....	308
11.3 简单范例.....	260	第 13 章 实例：使用 PHP 开发 XML 应用程序.....	309
11.3.1 XML 文档.....	260	13.1 理解应用程序.....	309
11.3.2 转换 XML.....	260	13.1.1 设置环境.....	309
11.3.3 添加新的 DVD.....	267	13.1.2 天气网站应用程序的各个部分.....	314
11.3.4 修改已有的 DVD.....	273	13.2 小结.....	337
11.3.5 删除 DVD.....	279		
11.4 小结.....	281		



本章将介绍XML（可扩展标记语言），并解释它的一些基本概念。如果你对XML还比较陌生，这将是一个理想的起点。这里介绍的概念将在本书后续部分做更详细的阐述。

Web开发者往往对XHTML（可扩展超文本标记语言）很熟悉，但却经常搞不清它与XML的关系，也不清楚自己为什么需要同时学习XML。但是可以肯定的是，对于开发者来说，这两种技术都是重要的。

XML是用来编写其他语言（称为XML词汇）的元语言。XHTML是这些词汇中的一种，所以如果理解了XML，你也就能理解作为XHTML基础的一些规则。XHTML是遵循XML规则的HTML语言，后面我们将会介绍更多有关的内容。

XHTML有很多的局限性。虽然它擅长于组织Web浏览器中的信息并进行显示，但它的主要任务并不是对数据进行标记。XHTML无法提供一些高级功能，如不能对内容进行排序和过滤等。描述XHTML文档内容的标签不能自己进行定义。XHTML的标签是固定的，它们通常无法与其所包含内容的类型进行关联。例如，段落标签是可用于任何类型的内容的通用容器。

XML解决了这些HTML常见的局限性问题。它比HTML灵活性更高，因为它可以与其他一些支持表现、组织、转换和导航的标准一起工作。XML文档是自描述的，在它们的文档结构中可使用可描述的标签对其所标记的内容进行识别。

本章将更详细地讨论这些问题，对XML做更多的解释并说明为什么需要你的工作中使用XML。本章将包含以下内容：

- XML的定义及其简史；
- 如何编写XML文档；
- 如何处理XML内容。

本章将首先介绍XML到底是什么，以及它适合于Web开发的哪些领域。

1.1 XML 是什么

对XML第一点也是最重要的一点认识就是，它本身并不是一个编程语言，而是一个用于构造其他语言也就是所谓词汇的元语言。XML描述了创造这些词汇的规则。每种语言都互不相同，但都使用标签来标记内容。标签的名称和结构可自由选择。对于认同标准XML词汇的组织来说，

这是通用的，这样他们就可对信息进行共享。

XML语言的一个例子就是XHTML。XHTML描述了一套标准的标签，这些标签只能以一种特定的方式使用。每个XHTML页面包含两个部分，分别使用<head>和<body>标签描述。每个部分都只能包含特定的标签。例如，不能将<meta>标签置于<body>部分中。世界各地的Web开发者都有一个共同的标准方法，这样浏览器就知道如何来呈现XHTML标签了。

XML是W3C（万维网联盟）的推荐标准，这意味着它是可以免费使用的。W3C在它的术语表（<http://www.w3.org/TR/DOM-Level-2-Core/glossary.html>）中提供了一个关于XML的更正式的定义，内容如下：

XML（可扩展标记语言）是SGML中的一个简单子集。它的目标是使通用的SGML可以如同当前的HTML一样，在网络上进行服务、接收和处理。XML设计得非常便于实现，且与SGML和HTML都具有互操作性。

1.1.1 XML 简史

XML于1998年问世，它是基于SGML（标准通用标记语言）的。SGML是一个国际标准，可以将其理解为是定义其他文档标记语言的语言。HTML是基于SGML的。SGML最主要的问题是它难于使用，而XML的目标就是要变得更简单。

XML也从HTML中获益良多。HTML专注于内容的显示；你不能用它来完成诸如排序和过滤等更高级的功能。HTML不是一个非常精确的语言，它不区分大小写。即使不正确的HTML内容也可能被某个浏览器正确地进行显示。

XML纠正了HTML中的很多缺陷。1999年，HTML被使用XML语言构造规则改写为XHTML。XHTML文档的构造规则比HTML的要精确得多。这些规则的严格程度取决于在XHTML页面中所指定的文档类型声明（DOCTYPE）。第3章将对DOCTYPE进行更多的讨论。

从1998年起，XML已明显成为管理信息的一个强有力的方法。XML文档使数据易于共享。一系列相关的W3C推荐标准解决了在XML文档内进行转换、显示和导航的问题。第2章将会详细介绍这些推荐标准。

XML的要点总结如下：

- XML不是一个语言，它的规则用来构造其他语言；
- XML创建了用来标记内容的基于标签的语言；
- XHTML是由XML创建的语言的一种，也是对HTML的重新构造；
- XML是基于SGML的。

1.1.2 XML 的目标

SGML的异常复杂使W3C非常清楚地看到XML所要达到的目标。这些目标可参见<http://www.w3.org/TR/REC-xml/#sec-origin-goals>：

- (1) XML应该在因特网上易用；
- (2) XML应支持多种不同的应用；

- (3) XML应与SGML兼容;
- (4) 处理XML文档的程序应易于编写;
- (5) XML中的可选特性将减到最小, 理想状态为零;
- (6) XML文档应是清晰、易读的;
- (7) XML的设计应该比较快捷;
- (8) XML的设计应该是形式化而精练的;
- (9) XML文档应该易于创建;
- (10) XML不追求标记的简洁。

这些目标中有几点值得注意。第一, W3C希望XML是非常简单的; 实际上, 好几个目标都包含了“容易”和“清晰”的含义。

第二, W3C赋予了XML两个目标: 供人阅读和供XML处理程序处理。XML处理程序或者解析器是用来处理XML文档的软件包。处理程序可以识别XML文档的内容, 也可以读、写或修改已存在的文档, 或是直接创建新的文档。

其目的是通过保持开发的简单性来打开XML处理程序的市场。严格的构造规则意味着所需要进行的处理会比较少。因而也意味着XML文档可应用于移动设备, 诸如手机和PDA。

通过使文档可供人阅读, 你可以更方便地使用数据, 也可以更容易地开发和调试应用程序。Unicode的应用程序开发者可以使用很多不同的语言创建XML文档。然而, 其产生的问题是XML文档可能变得比较罗嗦, 用XML来描述数据也会比采用其他方式更冗长。

Unicode

XML支持Unicode字符集以保证对多语言的支持。Unicode提供对 2^{31} 字符的支持。它包含了你可能需要的每一个字符, 同时也有很多是你永远也用不上的。

可以使用8位Unicode转换格式 (UTF-8) 来编码Unicode字符, 从而使得这些字符与它们在ASCII中的编码保持一致。显然, 这提供了与老系统更好的兼容性。中文和日语这样的语言需要UTF-16编码。可以在<http://www.unicode.org>中了解更多关于Unicode的内容。

第三, 请注意术语XML文档 (document)。这个术语的含义比传统意义上的物理文档更为丰富。一些XML文档以物理形式存在, 但另一些是作为遵循XML构造规则的信息流的形式创建的。例如, Web 服务以及内容以XML形式返回的数据库调用结果。

在理解了XML是什么以后, 现在来深入地研究XML语言的构造规则。

1.2 XML 语法

XML语言使用标签来标记文本。作为Web开发者, 你应该很熟悉标记文本的含义:

```
<p>Here is an introduction to XML.</p>
```

上面一行是XHTML, 同时也是XML。在XHTML中, <p>标签表示一段文本。XHTML中的

所有标签都有预先定义好的含义。

XML允许你创建自己的标签，因而你可以将原先的标记改写为：

```
<intro>Here is an introduction to XML.</intro>
```

在本例中，<intro>标签说明了标记了的文本的含义。XML的一大优势就是它的标签可以描述其内容——这也就是为什么XML语言会经常称为自描述的（self-describing）。

XML足够灵活，可以容许创建多种类型的语言以描述数据。对这些XML词汇的唯一限制就是，它们必须是合式的（well-formed）。

1.2.1 合式的文档

XML文档如果满足下列条件就是合式的：

- ❑ 该文档包含一个或多个元素；
- ❑ 该文档只包含一个文档元素，用于包含其他元素；
- ❑ 每个元素都应正确地结束；
- ❑ 元素都是区分大小写的；
- ❑ 属性值都括于引号中并且不为空。

本章将讨论所有这些条件。现在有必要强调其中的一些要点。XML是区分大小写的，这就意味着<intro>标签与<Intro>或<INTRO>标签并不相同。在XML中，这是三个不同的标签。在XHTML之前，HTML是不区分大小写的，因而<body>和<BODY>是相同的标签。

所有的XML标签都需要一个对应的以同样大小写书写的结束标签。因此<intro>标签必须有一个匹配的</intro>标签。如果在开始和结束标签中没有内容存在，那么可以将之缩写为一个标签<intro/>。在HTML中，有可能只写一个单独的<p>标签就添加一个段落。

在XML中，标签的顺序是重要的。最先开始的标签必须最后结束：

```
<chapter><intro>Here is an introduction to XML.</intro></chapter>
```

HTML页面没有这样的要求。下面的例子在HTML中是正确的，但是在XML中却是不可接受的：

```
<p><strong>Paragraph text</p></strong>
```

在XML中，属性中的值总是用引号括起来的：

```
<intro type="chapter">
```

使用单引号或双引号都没有关系，但是必须使用引号。这在HTML中并不是必需的。同样地，一些HTML属性，例如<td>标签中的nowrap属性，并不需要包含一个属性的名称和值对：

```
<td nowrap>A table cell</td>
```

这种形式的标签书写方式在XML中是不行的。必须将它换成如下所示的样子：

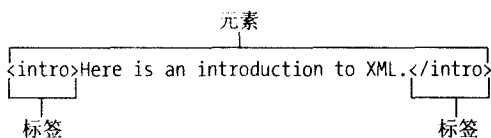
```
<td nowrap="true">A table cell</td>
```

1.2.2 标签和元素的不同

你可能已注意到，我在谈论XML文档时使用了标签（tag）和元素（element）这些术语。乍一看，它们似乎是可以互换的，但其实二者之间是有些不同的。

元素描述的是一个开始标签和一个结束标签以及其中的内容；而标签则是元素的一部分。标签以一个左尖括号开始，以一个右尖括号结束。元素通常包含开始标签和结束标签以及二者之间的内容。

下一行表示一个包含<intro>标签的完整元素：



在理解了构造规则后，下面认识一下完整的XML文档。

1.3 查看完整的XML文档

一段完整的XML称为文档。不管是用于标记文本的XML，或者是从服务器端请求回来的信息，再或是从数据库返回的记录，所有这些都是文档。

每个XML文档都是由标记和字符数据组成的。总地说来，字符数据包含开始标签和结束标签之间的文本，其他的都是标记。而标记可以进一步划分为元素、属性、文本、实体、注释、CDATA和处理指令。

下面的文档列举了XML文档的不同组成部分。该文件和其他资源文件可以从Apress网站（<http://www.apress.com>）的源代码区中下载¹。这个文档名为dvd.xml，它描述了一个小型DVD库的内容：

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- This XML document describes a DVD library -->
<library>
  <DVD id="1">
    <title>Breakfast at Tiffany's</title>
    <format>Movie</format>
    <genre>Classic</genre>
  </DVD>
  <DVD id="2">
    <title>Contact</title>
    <format>Movie</format>
    <genre>Science fiction</genre>
  </DVD>
  <DVD id="3">
    <title>Little Britain</title>
    <format>TV Series</format>
    <genre>Comedy</genre>
  </DVD>
</library>

```

1. 这些内容也可从图灵网站下载。——编者注

下面来浏览这个文档。首先，它由XML声明开始：

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

该声明是可选的，它包含一些属性，稍后会加以介绍。

这个XML文档还包含一个注释以说明它的用途：

```
<!-- This XML document describes a DVD library -->
```

这个注释是为所有阅读文档的读者提供的指南。对于XHTML来说，开发人员通常使用注释来添加一些说明。

文档元素（或称根元素）起名为<library>。值得注意的是，本文档内所有的元素都包含在一个开始和结束<library>标签中。

文档元素包含几个<DVD>元素，每个<DVD>元素包括<title>、<format>和<genre>元素。<DVD>元素中还包含一个id属性：

```
<DVD id="1">
  <title>Breakfast at Tiffany's</title>
  <format>Movie</format>
  <genre>Classic</genre>
</DVD>
```

每个<title>、<format>和<genre>元素中都含有文本。

通过标签的名字可以很容易地理解该文档的结构和内容。很显然，即使不用注释也能明白该文档描述的是一组DVD，而且也很容易推断出文档中各元素之间的关系。

1.3.1 XML 文档的结构

每个XML文档都分为两个部分：序言（prolog）和文档元素（或称根元素）。序言出现在XML文档的顶部，其中包含关于该文档的一些信息。它有点像XHTML文档中的<head>部分。在上面的XML文档中，序言包含了一个XML声明和一个注释。它也可以包含其他的元素，如处理指令或是DTD（文档类型定义），详细内容可参见1.3.3节的“处理指令”和“DTD和XML模式”部分。

合式的XML文档必须有一个文档元素用以包含可能的其他内容。XML文档中的所有内容都应该出现在文档元素或根元素的内部。在XML文档示例中，文档元素就是<library>，它包含了所有其他元素。

你可能对在XML文档中为元素所起的名字感到疑惑。在遵守XML命名规则的前提下，可以为元素和属性选择任何名字。

图1-1显示了XML文档的结构。

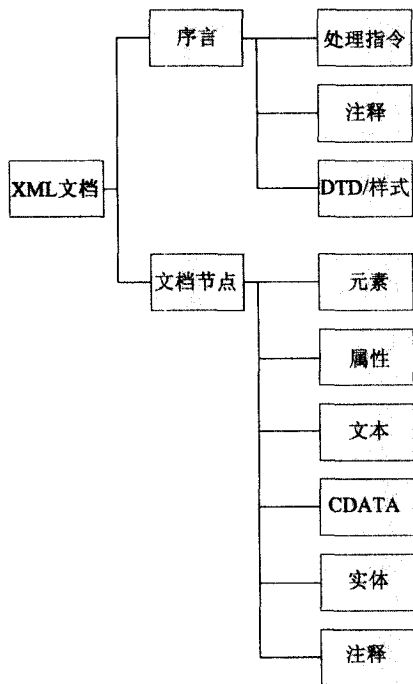


图1-1 XML文档的结构

1.3.2 XML 的命名规则

在XML文档中，元素、属性和一些其他结构是有名字的。名字是由一个起始字符加上名字字符组成。别忘了XML名字是区分大小写的。

起始字符必须是一个字母或下划线，不能是数字。而名字字符可以是除了空格和冒号以外的所有字符。冒号在XML中表示命名空间，所以不能将之用于名字中。第2章将更详细介绍命名空间。为了确保使用合法的字符，最好严格使用罗马字母（大小写均可）、数字和标点，冒号除外。

在自己书写而不是自动产生XML的内容时，最好采用标准的命名规范，而且应该使用具有描述性的名字。

我习惯以CamelCase方式命名并且以小写字母开始，除非在有些情况下元素名字通常就是大写的：

```
<camelCaseElementName>Here is an element name</camelCaseElementName>
```

通常会避免在名字中使用下划线字符，因为我觉得这会让名字更难读。

使用具有描述性的名字可以使人更易于理解。想像一下读到下面内容时你会遇到的困惑：

```
<zyxtr>Some content</zyxtr>
```

下面来总结一下XML的命名规则：

- ❑ XML名字不能以数字或标点符号开头；
- ❑ XML名字不能包含空格；
- ❑ 如果不是表示命名空间，就不要在名字中包含冒号；
- ❑ XML名字是区分大小写的。

下面将更详细地描述XML文档的内容。首先从可能出现在序言中的元素开始。

1.3.3 XML 文档序言

XML文档的序言里包含了关于文档本身的元信息，而不是文档的内容。它可能包含XML声明、处理指令、注释以及嵌入的DTD或模式。

1. XML声明

XML文档通常都以一个XML声明开始，虽然这并不是必需的：

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

在XML文档中，包含一个声明是好的做法，因为它可以告诉应用程序或人随后的是一段XML内容。它还能向处理程序提供关于该文档的额外信息，比如它的字符编码类型。

如果在XML中包含声明，就必须将它置于文档的第一行。任何内容都不能放在XML声明前面，空格也不行。如果你不小心在声明前面放了一个空格，XML处理程序就将无法正确解析文档的内容，并且会产生错误消息。

XML声明中还可以包含一些属性，这些属性提供关于版本、编码以及文档是否独立的信息：

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