



XML Step by Step

微软公司
核心技术书库

(美) Michael J. Young 著 前导工作室 译

XML

学习指南



机械工业出版社
China Machine Press

Microsoft Press

本书是XML语言的入门书籍。它介绍了到目前为止最重要、最实用的XML技术，并包含实用的例子和教程式的练习。全书分为三部分：第一部分简要介绍XML，并给出了一个简单的例子；第二部分重点介绍创建XML文档的规则和技术；第三部分介绍目前最重要的在Web浏览器中显示XML文档的技术。最后的附录提供了一些有关XML的Web站点地址。

Michael J. Young: XML Step by Step.

Copyright © 2001 by Microsoft Corporation.

Original English language edition copyright © 2000 by Michael J. Young.

Published by arrangement with the original publisher, Microsoft Press, a division of Microsoft Corporation, Redmond, Washington, U.S.A. All rights reserved.

本书中文简体字版由美国微软出版社授权机械工业出版社出版。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。

本书版权登记号：图字：01-2000-3120

图书在版编目(CIP)数据

XML学习指南/ (美) 杨 (Young, M. J.) 著；前导工作室译。 - 北京：机械工业出版社，2001.2

(微软公司核心技术书库)

书名原文：XML Step by Step

ISBN 7-111-08622-8

I. X… II. ①杨…②前… III. 可扩充语言 - 程序设计 - 指南 IV. TP312-62

中国版本图书馆CIP数据核字 (2000) 第79985号

机械工业出版社 (北京市西城区百万庄大街22号 邮政编码 100037)

责任编辑：李云静

北京忠信诚胶印厂印刷·新华书店北京发行所发行

2001年2月第1版第1次印刷

787mm×1092mm1/16·16.5印张

印数：0 001-5 000册

定价：40.00元(附光盘)

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

前 言

扩展标记语言（Extensible Markup Language，XML），是目前最有前途的在World Wide Web上保存和传递信息的语言。

尽管超文本标记语言（Hypertext Markup Language，HTML）目前是创建Web页的最常用语言，但是它在保存信息方面的能力很有限。与之相反，XML却有着极其灵活的语法，允许真实地描述各种类型的信息（从简单的菜谱到复杂的数据库）。XML文档（与样式表或传统的HTML页一起）可以很容易地在浏览器中显示。因为XML文档能如此有效地构造和标识它所包含的信息，以致于浏览器可以以非常灵活的方式发现、抽取、排序、过滤、整理以及管理信息。

总之，XML提供了一种理想的方法，使用它可以处理Web上数量和复杂性快速增长的信息。

为什么不阅读另一本XML书籍

XML可能会让人感到混乱。XML应用程序以令人吃惊的速度不断出现，而且XML与用来格式化、显示、处理和增强XML文档的大量相关标准和技术紧密联系在一起。许多这样的相关标准和技术还处在“幼儿”阶段，正在快速地变化和发展。

大部分我读过的XML书籍都试图全面地介绍这些技术，但是却让人有一种步入迷宫的感觉。我认为，这种典型的XML书籍过于肤浅地讨论了过多的XML技术，没有把实用与非实用、现在和将来相区别。

本书回答了大部分基本的XML问题——什么是XML，为什么需要它，以及怎样使用它——并且介绍了到目前为止最重要、最实用的XML技术。

本书的主题均是经过精心筛选的，对每一种主题的介绍都很有深度，并且避免了不完整的解决方案。（例如，因为在第二部分中介绍了怎样定义XML属性，所以在第三部分接着介绍了在显示文档时怎样访问这些属性。）

直到我亲自动手编写并显示XML文档时，才真正理解了XML。因此，我在本书中也采用了这种方法，包括了许多循序渐进的指导、实用的例子和教程式的练习；避免对XML这种难于理解的主题进行理论和抽象的讨论。

本书及附带的光盘还提供了一个完整的XML学习软件包。这个软件包提供了学习创建和显示XML文档所需基础知识的所有信息、指导和软件。本书还包括了指向Web上大量XML信息的详尽链接集，如果你想要了解基础知识以外的信息，则可以利用这些链接。

在本书中将学到什么

本书的第一部分（第1章和第2章）提供了对XML的简要介绍，并为学习后续的详细信息做准备。第1章回答了我前面提到的基本问题——什么是XML，为什么需要它，以及怎样使用它来解决现实世界的问题。第2章提供了一个手把手的练习，使你对创建XML文档并在Web浏览器中

显示的全过程有一个大致印象。

第二部分（第3章到第6章）重点介绍创建XML文档的规则和技术。第3章和第4章介绍怎样创建格式正确的XML文档——符合最基本的XML需求的文档。第5章和第6章介绍怎样创建有效的XML文档——符合可选和更严格的规则集的文档。第二部分中的章节基于W3C（World Wide Web协会）的正式XML规范。

第三部分（第7章到第10章）介绍目前最重要的在Web浏览器中显示XML文档的技术。第7章和第10章解释怎样通过附加一个为浏览器提供格式指令的样式表来显示XML文档。第7章讨论级联样式表（cascading style sheet）——最简单，但是最受限的样式表类型——而第10章介绍使用XSL（扩展样式表语言，Extensible Stylesheet Language），一种用XML编写的更高级、更灵活的样式表类型。

第8章和第9章介绍怎样通过把文档链接到访问、处理和表示XML数据的HTML Web页来显示XML文档。第8章介绍怎样使用数据绑定（data binding，一种只适用于某种XML文档的简单技术）来做到这一点。第9章介绍怎样通过可用于任意XML文档类型且高度灵活的编程对象，即众所周知的XML文档对象模型（XML Document Object Model），编写脚本来以便HTML页中显示XML文档。

注意 在本书中，使用术语“页面（page）”来表示HTML源文件，使用术语“文档（document）”来表示XML源文件，以便清楚地区别这两种经常一起使用的标记语言。

第三部分重点讨论使用Microsoft Internet Explorer 5 Web浏览器显示XML文档。

最后，附录提供了一些Web站点的地址，这些站点包含了大量有关本书中所讨论主题的更深入信息。我还在各章节中包括了这些地址，每一个地址都适用于各章节讨论的内容。在本书附带光盘的“Resource Links”文件夹中名为Appendix.htm的文件中，会找到附录的拷贝。（要了解在计算机上安装Appendix.htm的信息，参见光盘上的Readme文件。）你可以通过在Web浏览器中打开Appendix.htm文件并单击链接来访问任意一个Web站点，而不必在浏览器中输入地址。

提示 你可能想要在Web上访问下列XML信息站点以便补充从本书中学到的内容。W3C在www.w3.org/中为Web创作人员提供了各种信息、标准和服务。Microsoft Developer Network（微软开发商网络，MSDN）在<http://msdn.microsoft.com/xml/default.asp>中提供了Microsoft产品所支持的、有关XML的大量信息。<http://www.oasis-open.org/cover/>中的在线参考书《The XML Cover Page》包括了有关XML和其他标记语言的综合信息。

循序渐进学习XML和Internet Explorer

尽管本书中提供的大部分技术都基于公开的标准，并且能够与大量浏览器一起协同工作，但是本书是专门为Microsoft Internet Explorer 5.0 ~ 5.5编写的。本书附带的光盘包括Internet Explorer 5.01。因此，你可以使用这张光盘上提供的浏览器软件显示本书中使用这些技术创建的任意XML文档。

提示 可以从<http://www.microsoft.com/windows/ie>下载最新的Internet Explorer 5.5。

在整本书中, Internet Explorer 5指的是Microsoft Internet Explorer 5.0 ~ 5.5。当某些信息只适用于某个Internet Explorer版本时, 会及时指出, 例如Internet Explorer 5.01或Internet Explorer 5.5。

本书中介绍的所有特性和技术都可以用于Internet Explorer 5.5。少数几种特性是第一次出现在版本5.5中, 因此无法用于以前的版本。我将清楚地标出只适用于Internet Explorer 5.5的那些特性。

环境需求

下面是使用本书及其附带光盘的基本硬件和软件需求:

- 要访问本书附带的光盘, 并安装包括在光盘中的Internet Explorer软件, 需要一台运行Microsoft Windows并安装有一台CD-ROM驱动器的计算机。可以使用Windows 95、Windows 98、Windows NT 4.0 (带有Service Pack3或更高版本)、Windows 2000, 或者任意更高版本的Windows。
- 选择“Install Example Code And Resource Links”安装选项大约需要1MB硬盘空间。
- 要查看本书中引用的Web站点, 需要连接到Internet。不过, 成功地使用本书不需要查看这些站点, 因此Internet连接是可选的。

本书的目的是介绍XML, 因此不需要事先对XML本身有所了解。不过, 本书中介绍的几种有关显示XML文档的技术需要使用下列一种或多种Web创作语言: HTML、动态HTML (DHTML) 和Microsoft JScript (普通JavaScript脚本语言的Microsoft版本)。尽管本书在例子中介绍了这些语言的特性, 但是了解这些语言的一些基本知识仍然有用。

提示 要了解在Internet Explorer 5中使用HTML和DHML的信息, 请访问MSDN提供的Web站点: <http://msdn.microsoft.com/workshop/author/default.asp>。要阅读正式的HTML规范, 参见W3C提供的Web站点: <http://www.w3.org/TR/REC-html40/>。在也是由MSDN提供的站点<http://msdn.microsoft.com/workshop/c-frame.htm#/workshop/languages/jscript/handling.asp>和<http://msdn.microsoft.com/scripting/default.htm?scripting/jscript/default.htm>中, 包括完整的JScript信息, 也包括初学者的教程。

使用附带的光盘

附带的光盘提供了下列有价值的资源, 以便对本书的信息进行补充:

- 本书中带有编号的程序清单的源文件拷贝。这些程序清单 (例如, 第2章中的程序清单2-1) 提供了XML文档、样式表和用来显示XML文档的HTML页的例子。在引入带有编号的程序清单时, 也会同时给出它在光盘中对应文件的名称。(例如, 程序清单2-1在文件Inventory.xml中。)所有这些文件都在附带光盘的“Example Code”文件夹中。
- 所有在XML文档示例中显示的图形文件。这些文件包含在与程序清单文件相同的光盘文件夹(Example Code)中。

- Web页文件Appendix.htm中有附录的拷贝。该文件在光盘上的“Resource Link”文件夹中。
- Internet Explorer 5.01版。

当把CD-ROM插入驱动器时，附带的光盘会自动启动安装。如果计算机不支持自动启动，就使用Windows资源管理器显示光盘的内容，并双击根文件夹中的StartCD.exe。这两种方法都可以打开欢迎屏幕。

欢迎屏幕上显示了阅读Readme文件、安装示例代码和资源链接，安装Microsoft Internet Explorer 5.0，浏览光盘内容，或链接到Microsoft Press支持的Web站点等选项。一旦选择了一个选项，只要按照屏幕上显示的指示操作即可。

怎样与作者联系

你可以访问我的Web站点：<http://ourworld.compuserve.com/homepages/mjy>。那里有对本书的更正、读者的问题以及答案，获取有关XML进一步帮助的资源及相关技术，我的一些其他书籍的介绍和附加的信息。

你可以给我发e-mail，地址是mjy@compuserve.com。欢迎你的意见和反馈信息。尽管我很少有时间回答那些需要深入研究的问题，但是如果你的问题我能够回答，那么我将会尽力回复。

Microsoft Press支持信息

我尽了最大的努力保证本书和附带光盘中内容的准确性。Microsoft Press通过万维网上的<http://mspress.microsoft.com/support>提供了对书籍的更正。

如果你对本书或者它所附带的光盘有意见、问题或想法，请通过e-mail把它们发送到Microsoft 出版社：

MSPinput@Microsoft.com

或者把信邮寄到：

Microsoft Press

Attn: XML Step by Step Editor

One Microsoft Way

Redmond, WA 98052-6399

请注意，产品支持不是通过上述地址提供的。

英文原书书号：ISBN0-7356-1020-7

英文原书名：XML Step by Step

目 录

前言

第一部分 XML概览

第1章 为什么要使用XML	1
1.1 XML的需要	1
1.1.1 XML解决方案	4
1.1.2 SGML、HTML和XML	6
1.2 XML的正式目标	7
1.3 标准XML应用	8
1.4 XML的实际应用	9
第2章 创建并显示第一个XML文档	12
2.1 创建一个XML文档	12
2.1.1 创建XML文档	12
2.1.2 XML文档剖析	14
2.1.3 一些基本的XML规则	16
2.2 显示XML文档	16
2.2.1 显示没有样式表的XML文档	17
2.2.2 在Internet Explorer 5中获取XML 错误	18
2.2.3 使用级联样式表显示XML文档	19

第二部分 创建XML文档

第3章 创建格式正确的XML文档	27
3.1 一个格式正确的XML文档的组成部分	27
3.2 给文档添加元素	31
3.2.1 元素的剖析	32
3.2.2 创建不同的元素类型	35
3.3 给元素添加属性	37
3.3.1 创建属性的规则	38
3.3.2 合法属性值的规则	38
第4章 添加注释、处理指令和CDATA 节	42

4.1 插入注释	42
4.1.1 注释的形式	42
4.1.2 在哪里放置注释	42
4.2 使用处理指令	43
4.2.1 处理指令的形式	43
4.2.2 怎样使用处理指令	44
4.2.3 处理指令可以放在什么地方	44
4.3 包括CDATA节	45
4.3.1 CDATA节的形式	45
4.3.2 CDATA节可以放在什么地方	46
第5章 创建有效的XML文档	47
5.1 有效XML文档的基本标准	47
5.2 添加DTD	48
5.2.1 DTD的格式	49
5.2.2 创建DTD	49
5.3 声明元素类型	50
5.3.1 元素类型声明的格式	51
5.3.2 元素内容规范	51
5.4 声明属性	57
5.4.1 属性列表声明的格式	57
5.4.2 属性类型	58
5.4.3 缺省的声明	63
5.5 使用外部DTD子集	64
5.5.1 只使用外部DTD子集	64
5.5.2 同时使用外部DTD子集和内部 DTD子集	65
5.5.3 有条件地忽略外部DTD子集的 一部分	66
5.6 把格式正确的文档转换为有效文档	67
第6章 定义和使用实体	71
6.1 实体定义和分类	71
6.2 声明通用实体	73

6.2.1 声明内部通用可析型实体	73
6.2.2 声明外部通用可析型实体	75
6.2.3 声明外部通用不可析型实体	76
6.2.4 声明符号	77
6.3 声明参数实体	78
6.3.1 声明内部参数可析型实体	78
6.3.2 声明外部参数可析型实体	79
6.4 插入实体引用	81
6.4.1 实体引用示例1	82
6.4.2 实体引用示例2	82
6.5 插入字符引用	83
6.6 使用预定义实体	84
6.7 给文档添加实体	85

第三部分 在Web浏览器中显示XML文档

第7章 使用级联样式表显示XML文档	91
7.1 使用级联样式表的基本步骤	92
7.1.1 第1步: 创建样式表文件	92
7.1.2 第2步: 链接样式表到XML文档	100
7.2 级联样式表中的级联	101
7.3 设置display属性	103
7.4 设置字体属性	105
7.4.1 设置font-family属性	105
7.4.2 设置font-size属性	107
7.4.3 设置font-style属性	110
7.4.4 设置font-weight属性	110
7.4.5 设置font-variant属性	112
7.5 设置color属性	112
7.6 设置背景属性	114
7.6.1 设置background-color属性	114
7.6.2 设置background-image属性	115
7.6.3 设置background-repeat属性	117
7.6.4 设置background-position属性	119
7.7 设置文本间距和对齐属性	123
7.7.1 设置letter-spacing属性	123
7.7.2 设置vertical-align属性	125

7.7.3 设置text-align属性	125
7.7.4 设置text-indent属性	127
7.7.5 设置inline-height属性	128
7.7.6 设置text-transform属性	129
7.7.7 设置text-decoration属性	129
7.8 设置方框属性	130
7.8.1 设置页边距属性	131
7.8.2 设置边界属性	134
7.8.3 设置填充符属性	137
7.8.4 设置尺寸属性	138
7.8.5 设置定位属性	139
7.9 在XML文档中插入HTML元素并使用名称空间	147
7.10 创建并使用一个功能完善的级联样式表	150
7.10.1 创建文档	150
7.10.2 创建样式表	153
第8章 使用数据绑定显示XML文档	155
8.1 主要步骤	155
8.2 第一步: 链接XML文档到HTML页	156
8.2.1 XML数据是怎样存储的	157
8.2.2 检查XML的错误	158
8.3 第二步: 绑定HTML元素到XML元素	158
8.3.1 使用表格数据绑定	158
8.3.2 使用单记录数据绑定	173
8.3.3 其他数据绑定技术	178
8.3.4 把数据绑定与DTD一起使用	184
8.3.5 绑定HTML元素到XML属性	189
8.4 把脚本与DSO一起使用	193
第9章 使用文档对象模式脚本显示XML文档	199
9.1 把XML文档链接到HTML页	199
9.2 DOM的结构	200
9.3 访问和显示XML文档元素	204
9.3.1 使用NodeList对象	208
9.3.2 检索元素的字符数据	209
9.3.3 显示可变数目的XML元素	210

9.3.4 使用其他方法访问元素	213	10.2 使用单个XSL模板	231
9.4 访问并显示XML文档的属性值	216	10.3 使用多个模板	243
9.5 访问XML实体和记号	218	10.4 过滤并排序XML数据	245
9.6 遍历整个XML文档	222	10.4.1 过滤	245
9.7 检查XML文档的有效性	226	10.4.2 排序	246
9.7.1 怎样使用有效性测试页面	226	10.4.3 过滤和排序样式表的例子	246
9.7.2 有效性测试页面是怎样工作的	227	10.5 访问XML属性	249
第10章 使用XSL样式表显示XML文档	230	附录 获取更多信息的Web地址	252
10.1 使用XSL样式表的基本步骤	230		

第一部分 XML概览

第1章 为什么要使用XML

XML代表扩展标记语言（Extensible Markup Language），是由World Wide Web Consortium（W3C）的XML工作组定义的。这个工作组是这样描述该语言的：

“扩展标记语言（XML）是SGML的子集，其目标是允许普通的SGML在Web上以目前HTML的方式被服务、接收和处理。XML被设计成易于实现，且可在SGML和HTML之间互相操作。”

这段话是从正式的XML规范1.0版本中引述的，该规范是XML工作组在1998年2月完成的。你可以在W3C位于<http://www.w3c.org/TR/REC-xml>的Web站点上阅读整篇文档。

正如所看到的，XML是一种专门在World Wide Web上传递信息的语言，就像HTML（超文本标记语言）一样（自从Web出现以来，HTML已经成为了创建Web页的标准语言）。因为我们已经有了HTML，而且它已发展成一种似乎可以满足任何需要的语言，所以，你可能会疑惑：为什么我们还需要在Web上采用一种全新的语言呢？XML有什么新特性和不同之处？它有什么特有的优点和长处呢？它与HTML有什么关系？它是要替换还是增强HTML？最后，这个以XML作为其子集的SGML是什么，为什么我们不就用SGML来创建Web页呢？本章将试图回答所有这些问题。

1.1 XML的需要

HTML提供了固定的预定义元素集，可以使用它来标记一个典型、通用的Web页的各个组成部分。预定义元素的例子有：标题（heading）、图片（paragraph）、列表（list）、表格（table）、图像（image）和链接（link）。例如，HTML很适合用来创建个人主页（home page），如下面的HTML页所示：

```
<HTML>
<HEAD>
<TITLE>Home Page</TITLE>
</HEAD>

<BODY>
<H1><IMG SRC="MainLogo.gif">Michael Young 's Home Page</H1>
<P><EM>Welcome to my Web site!</EM></P>

<H2>Web Site Contents</H2>
<P>Please choose one of the following topics:</P>
```

```

<UL>
  <LI><A HREF="Writing.htm"><B>Writing</B></A></LI>
  <LI><A HREF="Family.htm"><B>Family</B></A></LI>
  <LI><A HREF="Photos.htm"><B>Photo Gallery</B></A></LI>
</UL>

<H2>Other Interesting Web Sites</H2>
<P>Click one of the following to explore another Web site:</P>
<UL>
  <LI>
    <A HREF="http://www.yahoo.com/">Yahoo Search Engine</A>
  </LI>
  <LI>
    <A HREF="http://www.amazon.com/">Amazon Bookstore</A>
  </LI>
  <LI>
    <A HREF="http://mspress.microsoft.com/">Microsoft Press</A>
  </LI>
</UL>
</BODY>
</HTML>

```

如图1-1所示，Microsoft Internet Explorer 5显示了该页面。

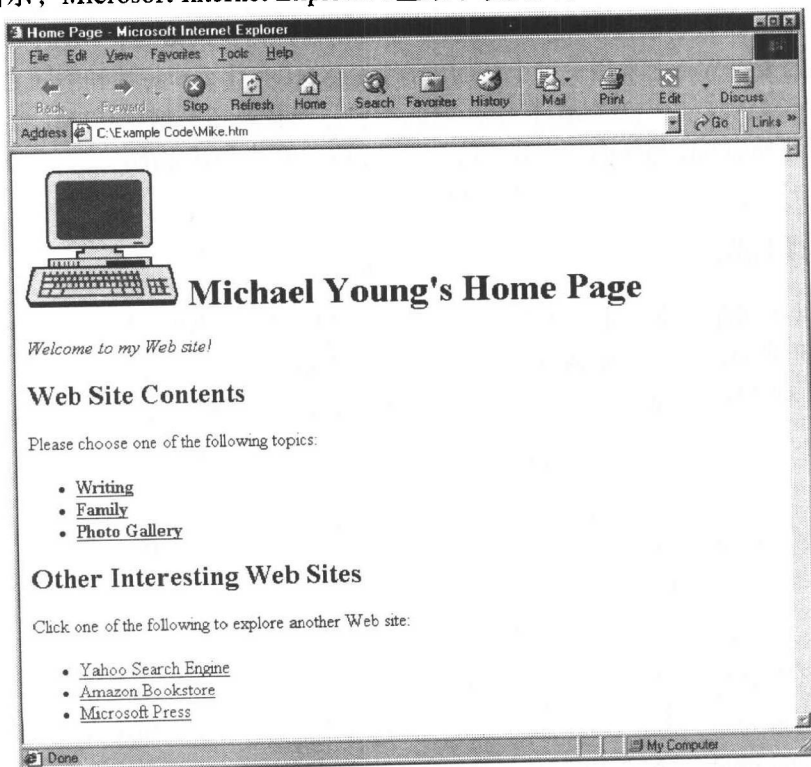


图 1-1

每一个元素都以起始标签（start-tag）开头：起始标签是前面带有左尖括号（<），后面带有右尖括号，并且包含元素名称和其他信息的文本块。大部分元素都以结束标签（end-tag）结尾，该标签很像它所对应的起始标签，不过其元素名称在斜杠（/）后面。元素content是位于起始标签和结束标签之间的文本（如果有的话）。注意，上一个示例页面中的许多元素包含嵌套元素。图1-2显示了一个HTML元素的组成部分。



图 1-2

示例HTML页面包含下列元素：

HTML元素	所标记的页面组成部分
HTML	整个页面
HEAD	标题信息，例如页面的标题
TITLE	页面的标题，它显示在浏览器的标题栏
BODY	浏览器显示的文本主体
H1	顶级的标题
H2	第二级的标题
P	文本段落
UL	项目符号列表（未排序的列表）
LI	列表中的一个项目（列表项）
IMG	图像
A	链接到另一个位置或页面（一个锚元素）
EM	一块斜体字（强调的）文本
B	一块粗体字文本

显示HTML页的浏览器可以识别这些标准元素中的每一个元素，并且知道怎样格式化和显示它们。例如，浏览器通常用最大的字体显示H1标题，H2标题用一个较小的字体，元素P表示更小的字体。它在未排序的列表中把一个LI元素显示成一个带有项目符号的、缩进的段落。此外，它把元素A转换成带有下划线的链接，用户可以单击它访问一个不同的地方或页面。

尽管自从第一个HTML版本后，预定义的HTML元素已经得到了极大的扩展，但是HTML仍然无法适用于已定义的众多文档类型。下面就是无法使用HTML充分描述的文档示例：

- 不是由典型的组成部分（标题、段落、列表、表格等等）组成的文档。例如，HTML缺乏标记音乐乐谱或算术等式集的元素。
- 数据库，例如书籍的详细目录。你可以使用HTML页保存和显示静态的数据库信息（例如书籍描述的列表）。但是，如果你想要排序、过滤、查找，以及通过其他方法使用这些信息，那么每一块信息都需要有标号（就好像在数据库程序，例如Microsoft Access 中一样）。而HTML缺乏实现它所需的典型元素。

- 想要组成树型分层结构的文档。例如，你正在写一本书，而且想要把它分为部分、章、A小节、B小节、C小节等等。接着，一个程序可以使用这个结构化的文档生成目录，各种不同级别细节信息的架构，抽取特定的小节，以及通过其他方法使用这些信息。不过，HTML标题元素只标记标题文本本身。例如：

```
<H2>Web Site Content</H2>
```

因为没有在标题元素中嵌套一个属于文档部分的实际文本和元素，所以这些元素不能用来清楚地指出文档的层次结构。

解决这些限制的方法是XML。

1.1.1 XML解决方案

XML的定义只是由框架语法组成。当创建一个XML文档时，不必使用有限的预定义元素集，而是创建自己的元素，并赋予任意你喜欢的名称——这就是扩展标记语言（Extensible Markup Language）中术语“扩展”的意义。因此，你可以使用XML描述任意类型的文档，从音乐乐谱到数据库。例如，可以描述书籍列表，如下XML文档所示：

```
<?xml version="1.0" ?>

<INVENTORY>
  <BOOK>
    <TITLE>The Adventures of Huckleberry Finn</TITLE>
    <AUTHOR>Mark Twain</AUTHOR>
    <BINDING>mass market paperback</BINDING>
    <PAGES>298</PAGES>
    <PRICE>$5.49</PRICE>
  </BOOK>
  <BOOK>
    <TITLE>Moby-Dick</TITLE>
    <AUTHOR>Herman Melville</AUTHOR>
    <BINDING>trade paperback</BINDING>
    <PAGES>605</PAGES>
    <PRICE>$4.95</PRICE>
  </BOOK>
  <BOOK>
    <TITLE>The Scarlet Letter</TITLE>
    <AUTHOR>Nathaniel Hawthorne</AUTHOR>
    <BINDING>trade paperback</BINDING>
    <PAGES>253</PAGES>
    <PRICE>$4.25</PRICE>
  </BOOK>
</INVENTORY>
```

注意 当用来描述一个数据库时，使用XML比使用专用格式（例如Access的.mdb和dBase的.dbf格式）多了两个好处：XML是可供人阅读的，而且它基于一种公共的开发标准。

了解XML文档中的元素名称（例如这个例子中的INVENTORY、BOOK和TITLE）不是XML定义的一部分是很重要的。这些名称是创建某个文档时构造的。而且你可以选择任何有效的名称作为元素名称（例如可以使用LIST代替INVENTORY，ITEM代替BOOK）。

提示 当你在XML文档中命名元素时，应尽量选择描述性的名称——例如，应该使用BOOK或ITEM，而不是FOO或BAR。XML文档的一个优点就是能够自我描述——也就是说，每一段信息可以附带一个描述性的标签。

正如从上面的例子中看到的，XML文档类似于树型分层结构，元素可以完全嵌套在另一个元素中，且只有单个顶层元素（在这个例子中是INVENTORY）——即文档元素（document element）或根元素（root element）——它包含了所有其他元素。示例XML文档结构如图1-3所示。

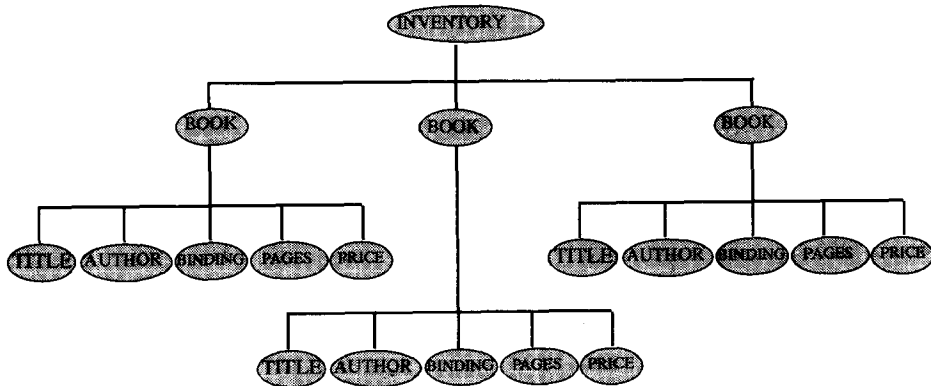


图 1-3

因此，你可以很容易地使用XML定义分层结构文档，例如上面提到的可分为部分、章、和各种级别小节的书。

1. 编写XML文档

因为XML文档没有包括预定义的元素，所以看起来像一个相当随意的标准。然而，实际上XML有着严格定义的语法。例如，与HTML不同，每一个XML元素必须同时有一个起始标签和一个结束标签（或者一个特殊的空元素标签，我将在本章稍后介绍）。任何被嵌套的元素必须完全包含在包围它的元素中。

实际上，非常灵活地创建自己的元素要求有严格的语法。这是因为XML这种定制本性（custom nature）要求定制程序（custom program）处理和显示这些文档包含的信息。严格的XML语法为XML文档提供了一种可确定的形式，并且使程序更容易编写。回忆一下本章开头所引用的话：“易于实现”是该语言的主要目标之一。

本书的第二部分讨论创建符合语法规则的XML文档。正如将学习到的，你可以编写符合两种不同的语法级别的XML文档。文档语法级别有“格式正确的”和“有效的”两种。

2. 显示XML文档

例如，在HTML页中，浏览器知道H1元素是顶级的标题，并根据这一点格式化和显示它。这样做是可能的，因为该元素是HTML标准的一部分。但是，浏览器和其他程序怎样得知如何处理和显示你在XML文档中创建的元素（例如示例文档中的BOOK或BINDING）呢？这些元素可是你自己创建的。

有三种基本方法可以告诉浏览器（尤其是Microsoft Internet Explorer 5）怎样处理和显示每一个XML元素。（我将在本书的第三部分详细地介绍这些技术。）

- 样式表链接（style sheet linking）。使用这种技术，你可以把一个样式表链接到XML文档。样式表是一个包含格式化每一种XML元素的指令的独立文件。你可以使用级联样式表（cascading style sheet, CSS）（可用于HTML页），或扩展样式表语言（Extensible StyleSheet Language, XSL）编写的样式表（比CSS的功能强大得多，并且是专门为XML文档设计的）。我将在第2、7和10章介绍这些技术。
- 数据绑定（data binding）。该选项需要你创建一个HTML页，把XML文档链接到它，并绑定该页面中的标准HTML元素，例如SPAN或TABLE元素，到XML元素。接着，HTML元素自动显示来自它们绑定的XML元素的信息。你将在第8章学习这种技术。
- 脚本（Scripting）。使用这种技术，可以创建一个HTML页，把一个XML文档链接到它，并通过编写脚本代码（JavaScript或Microsoft Visual Basic Scripting Edition (VBScript)）来访问和显示每个XML元素。浏览器把XML文档暴露为文档对象模型（Document Object Model, DOM），它提供了许多对象、属性和方法集，以便脚本代码使用它们访问、管理并显示XML元素。我将在第9章讨论这种技术。

1.1.2 SGML、HTML和XML

SGML代表结构化通用标记语言（Structured Generalized Markup Language），是所有标记语言的母语言。HTML和XML都派生自SGML（尽管方法不同）。SGML定义了一种基本的语法，但是它允许你创建自己的元素（因此使用了术语“通用”）。要使用SGML描述一个特定的文档，必须创建一个恰当的元素集和文档结构。例如，要描述一本书，可能会使用被命名为BOOK、PART、CHAPTER、INTRODUCTION、A-SECTION、B-SECTION、C-SECTION的元素等等。

用来描述一个特定文档类型的通用元素集就是众所周知的SGML应用程序。（一个SGML应用程序还包括使用我在第5章中讨论的技术指定元素以及其他特性排列方式的规则。）你可以定义自己的SGML应用程序来描述所使用的特定文档类型，或者一个标准体（standard body）可以定义一个SGML应用程序来描述广泛使用的文档类型。后一种应用程序类型的最著名例子是HTML，它是1991年开发的，用来描述Web页的SGML应用程序。

SGML看起来可能是用来描述Web文档的最佳扩展语言。但是，W3C的成员认为SGML太复杂、太麻烦，无法有效地在Web上传递信息。SGML所提供特性的灵活和冗余使得编写需要在Web浏览器中处理和显示SGML信息的软件变得困难。我们所需要的是专门为在Web上传递信息而设计的经过改进的（streamlined）SGML子集。1996年，W3C的XML工作组（XML Working Group）开发该子集，并把它命名为扩展标记语言。正如前面所述的那样，XML被设计为“易于实现的”，显然，该特性是SGML所缺少的。

因此, XML是为Web优化的SGML的简化版本。与SGML类似, XML允许在描述一个特定的文档时设计自己的元素。另一个与SGML的类似之处是, 某人或者一个标准体可以定义一个XML应用(也就是众所周知的词汇表), 它是一个可以用来描述特定文档类型(例如, 包含算术公式或矢量图形)的通用元素集和文档结构。你将在本章稍后学到更多有关XML应用的知识。

XML语法提供了比SGML更少的选项, 使得人们阅读XML文档和程序员编写用来访问、显示文档信息的浏览器、脚本和Web页更容易。

XML可以代替HTML吗

目前, 这个问题的答案是否定的。HTML仍然是让浏览器了解怎样显示Web上信息的主要语言。

在Internet Explorer 5中, 你可以在浏览器中直接打开一个附带了样式表的XML文档, 而不必使用HTML页。但是, 另外两种显示XML文档的主要方法——数据绑定和DOM脚本——使用HTML Web页作为载体来显示XML文档。(即使是使用样式表的方法, 如果你使用了XSL, 那么最后仍要使用HTML让浏览器了解怎样格式化XML数据。)

目前, XML不但没有代替HTML, 还与HTML一起使用, 并且极大地扩展了Web页的能力, 使Web页可以:

- 传递任意类型的文档。
- 用其他方法排序、过滤、重新排列、查找以及管理信息。
- 显示高度结构化的信息。

总之, XML可以与HTML进行互操作。

1.2 XML的正式目标

下面是W3C Web站点 (<http://www.w3.org/TR/REC-xml>) 上的XML正式规范中阐述的XML的10个设计目标:

1) XML应该可以直接用于Internet。

正如已经在本章中看到的, XML的主要设计目标是在Web上保存并传递信息。

2) XML应该支持各种应用程序。

尽管XML的主要目的是通过服务器和浏览器程序在Web上传递信息, 但是它还可以被其他类型的程序使用。例如, XML已被用于在财经程序之间交互信息, 用来发布和更新软件, 以及为通过电话传递信息编写语音脚本。

3) XML应该与SGML兼容。

正如我解释的, XML是SGML的专用子集。这种特性的一个好处是SGML软件工具可以很容易地适用于XML。

4) 编写处理XML文档的应用程序应该很简单。

如果希望XML有实用性, 那么它必须很容易编写出能够处理XML文档的浏览器和其他程序。实际上, 从SGML中派生出XML子集的主要原因是, 编写处理SGML文档的程序很笨拙。

下列设计目标主要是为了支持这个基本目标而服务的。

5) XML中可选特性的数目应该尽可能地少, 理想情况是零。

XML中可选特性的数目最少使得编写处理XML文档的程序更容易。SGML中有大量冗余的可选特性是它为什么被认为对于定义Web文档来说不实用的主要原因。可选的SGML特性包括重定义标签中的定界字符（通常为<和>），并且当处理程序知道元素结束的位置时忽略结束标签。一个处理SGML文档的健壮程序应该考虑所有的可选特性，即使这些特性很少使用。

6) XML文档应该便于人阅读而且相当清晰。

XML被设计为混合语（*lingua franca*），以便在用户和程序之间交换信息。可读性可通过允许人们（以及特殊的软件程序）编写和阅读XML文档来实现目标。这种便于人阅读的特性使XML区别于大部分被数据库和字处理文档所使用的专用格式。

人们可以很容易地阅读XML文档，因为它是用纯文本编写的，而且具有类似树型的逻辑结构。你可以通过为文档元素、属性和实体选择有意义的名字，并且增加有用的注释来增强XML的可读性。（我将在本章稍后部分解释元素、属性、实体和注释。）

7) XML设计应该很快地准备好。

当然，只有当程序员和用户团体都采纳XML时，XML才是一种可行的标准。因此，在这个团体开始采纳另一个标准之前，这个标准还需要完善，软件公司应该以很快的速度生成该标准。

8) XML的设计应该正式而且简洁。

XML规范是用一种定义计算机语言的正式语言编写的，这种语言就是众所周知的扩展巴斯克范式记法（Extended Backus-Naur Form, EBNF）。这种正式语言尽管看起来很难理解，但是由于解决了二义性的问题，使得它更容易编写XML文档，尤其是XML处理软件，这就进一步地鼓励了人们采纳XML。

9) XML文档应该易于创建。

要让XML成为一种适用于Web文档的实用标记语言，不仅要求XML处理程序必须很容易编写，而且要求XML文档本身必须很容易创建。

10) XML标记的简洁是最不重要的。

为了满足前面目标6的要求（XML文档应该便于人阅读且相当清晰），XML标记不应过于简洁，以致于含义模糊。

1.3 标准XML应用

不仅你可以使用XML来描述单个文档，而且任意一个人、公司或标准委员会都可以定义一个通用的XML元素集和文档结构，以便用于一种特殊的文档类。通用元素集和文档结构就是众所周知的XML应用或XML词汇表。

例如，某个组织可以定义一个XML应用来创建定义分子结构的文档，描述人力资源的文档，设计舞蹈动作多媒体演示的文档，或保存矢量图形的文档。在本章结尾部分，我将列出一些已创建或建议使用的公用XML应用。

一个XML应用通常通过创建文档类型定义（document type definition, DTD）来定义，DTD是XML文档的可选组件。DTD类似一个数据库概要信息：它定义和命名了可以在文档中使用的元素、元素显示的顺序、可用的元素属性以及其他文档特性。要使用一个特定的XML应用，通常需要在XML文档中包括其DTD；在文档中包含DTD限制了可以使用的元素和结构，这样文档