

SAMS

北京科海培训中心

XML 揭秘

—入门·应用·精通

[美] Michael Morrison, et al. 著
陆新年 陆新宇 等译



清华大学出版社

SAMS

北京科海培训中心

XML 揭秘

——入门·应用·精通

[美] Michael Morrison, et al. 著

陆新年 陆新宇 等译

清华大学出版社

(京)新登字 158 号

著作权合同登记号:01-2000-3937

内 容 提 要

本书由在 Web 技术领域中名声显赫的 Michael Morrison 编写。该书全面介绍了 XML 的相关技术,并对 XML 深层次的技术给出宝贵而实用的建议,远见卓识地描述了 XML 用于 Web 信息结构化的光明前景,使你可以从中获取丰富的技巧和经验,理解和拓宽知识的深度,掌握 XML 的全部潜能。作者在本书的后半部分详细地讲述了 XML 在多个领域中的应用。为了帮助读者更好地理解书中的内容,作者提供了大量简明实用且又具代表性的示例。

本书内容丰富、由浅入深,既适合初学者学习掌握 XML 语言,也适合有一定基础的读者精通 XML。

XML Unleashed

Copyright ©2000 by Sams Publishing

本书中文简体字版由美国 SAMS 公司授权清华大学出版社和北京科海培训中心出版。未经出版者书面允许不得以任何方式复制或抄袭本书内容。

版权所有,盗版必究。

本书封面贴有清华大学出版社激光防伪标签,无标签者不得进入各书店。

出版者:清华大学出版社(北京清华大学校内,邮编 100084)

印刷者:北京门头沟胶印厂

发行者:新华书店总店北京科技发行所

开 本:787×1092 1/16 印张:42.375 字数:1030 千字

版 次:2001 年 6 月第 1 版 2001 年 6 月第 1 次印刷

印 数:0001~5000

盘 号:ISBN 7-900635-28-9

定 价:69.50 元(光盘)

本书的主要作者

Michael Morrison 不但是一位作家、程序员、玩具发明家,而且还写了很多书。这些书包括 Complete Idiot's Guide to Java 2(Que Publishing, 1999), Java 1.1 Unleashed(Sams. net Publishing, 1997), How to Program JavaBeans(Ziff-Davis Press, 1997)以及 Teach Yourself Internet Game Programming with Java in 21 Days(Sams. net Publishing, 1996)。Michael 是 Gas Hound Games 的富有创造力的领导者, Gas Hound Games 的网址是 <http://www.gashound.com>。Michael 还是基于 Web 的几门课程的教师,其中包括 DigitalThink 的“Introduction to Java 2”以及“JavaBeans for Programmers”等等。平时,Michael 会给好朋友 Sarah 和 Travis 上网球课,或是和朋友 Keith 和 Josh 玩曲棍球,或是和他的妻子 Mahsheed 看电影,其他的时间里,他会坐在池塘边幻想。你可以访问 Michael 的站点 <http://www.thetribe.com>。

本书的其他作者

Frank Boumphrey 是一位医药文件顾问,他预见到了用 XML 和微软的技术来编写 Internet 和 Intranet 应用的前景。他广泛地编写和推广 XML, ASP 和 VB,同时,他还为 HTML 程序员协会(HTML Writers Guild, HWG)讲授 XML 和 ASP 的在线课程。他是 HWG Special Events 的副主席,同时他还是 XHTML 规范、扩展表单规范和 XHTML 模块化规范的编辑,这些规范都属于万维网联盟(W3C)。

他有一个美满的婚姻,五个孩子和几只小猫。

David Brownell 最近在 JavaSoft 工作。在那里他提供了 Sun 的 XML 和 DOM 软件的原始版本, SSL 和公钥结构以及 JavaServer 分页技术。他还致力于 Web 服务器和 Java Web 服务器的 Java Servlet API 的开发。

在此之前,他已经和 Internet 技术打了十几年的交道。最初是在波士顿地区,后来他搬到了旧金山地区从事 CORBA 的开发。他是 CORBA 互操作性和 IIOP 规范的主要制订者之一。

目前他是一位独立的 Web 和 Internet 顾问。

简介

在 1996 年 2 月的一次采访中, Bill Gates 说“HTML 是我们的数据类型。”尽管这句话是在不朽的 Web 时代中产生的, 但是它的确说明了 XML(可扩展标记语言)与 HTML(超文本标记语言)的相关性。XML 这项技术为我们共同创建的, 但目前依然十分混乱的 HTML 世界提供了框架结构。我们知道, 计算机是用来存储和处理数据的, 而数据最终是通过 Web 来共享的。然而, 在我们为能够如此快速和高效地共享信息而兴奋不已的时候, 我们忽略了周密思考的必要性。让我们回到正题, Bill Gates 是错误的: HTML 实际上不是一种好的数据类型。幸运的是, 一些人在几年以前已认识到这个问题, 并致力于开发解决方案。如果现在让 Bill Gates 说那句话, 他会说“XML 是我们的数据类型。”

XML 被设计为一把“铲子”, 它可以把我们在这几年的 HTML 标记以及 Web 页面中的描述标记所造成的“洞”中挖出来。XML 为正文和软件开发提供了框架结构, 而这在过去一直是几乎没有的。为了实现这一目的, XML 使软件的能力有了突飞猛进的提高。XML 显著地改变着我们所熟知的 Web(如果你认为这些都仅仅是一些广告, 请参看本书所述的 XML 功能。本书的最后一部分, 说明了现实世界中都有哪些公司和个人正在使用 XML)。

你可能和大多数人一样, 并不十分了解 XML。一个准确的定义是: “XML 是设计标记语言的规范”。换句话说, XML 是用来描述 HTML 之类的标记语言的元语言。当前形式的 HTML 不是一种 XML 标记语言, 不过已经有将 HTML 进行略微修改以符合 XML 标准的计划被启动。

XML 的伟大之处并不在于你能够通过它来创建标记语言, 而在于你能够通过它来创建高度结构化的标记语言。XML 实际上仅仅是一种标准化了的可以在 Web 上表示结构化信息的文本格式, 事实上就是这么简单。然而, 当你为像 Web 这样的巨大的信息仓库提供结构的时候, 什么事情都可能发生。新的自动化可以通过将结构化的数据在应用程序之间通过 Web 来回传递来实现。搜索引擎突然变得聪明起来而且可以进行基于上下文而不是正文的搜索。至少 Web 上的数据变得可以自我描述, 这就使应用程序可以更充分地利用它们。

说实话, XML 对 Web 影响的程度尚未知晓, 但是如果你对使用一种全新的数据驱动的 Web 应用感兴趣, 你一定会十分激动。这本书将会包括你所需要了解的 XML 并在广泛的应用模式中使用它所需要的全部内容。下面是本书包括的主要内容:

- XML 现状
- 用 DTD 来模式化数据
- 用 XML 模式来模式化数据
- 通过 XML 模式来创建文件
- 将 DTD 转化为 XML 模式
- 定义名字空间
- 样式表基础
- XSL 和 CSS 的比较

- XML 到 HTML 的桥梁
- XML 解析器揭密
- 用 DOM 和 SAX API 解析 XML
- 浏览 XML
- XHTML:XML 和 HTML 结合的产物
- 把 HTML 文件转换为 XHTML
- XLink 揭密
- XPointer 揭密
- 编写 XML 脚本
- 使用 DSO
- 理解 XSL 模式
- 用 XML 实现 EDI
- XML 编写和正文管理工具
- 生成 XML 文件
- 用 XMLNews 创建新闻故事
- 用 SMIL 集成多媒体对象
- 用 CDF 创建通道
- 用 VML 和 SVG 描述向量图形
- 用 3DML 创建 3D 世界
- 用 MathML 创建数学公式
- 用 P3P 创建私有策略
- 用 RELML 列出财产
- 用 HRMML 管理人力资源
- 用 VoxML 创建语音应答应用

本书是如何组织的

本书被设计为随时可用的最全面的 XML 开发手册。我们尽量将 XML 非常广泛的内容组织为手册的形式。这本书被分为以下几部分：

- 第 1 部分：定义 XML 文件。介绍了 XML 数据的模式化以及它和 XML 文件的关系。我们详细探讨了 XML 数据的模式化，包括在创建有效的 XML 文件时数据模式化的重要意义。对 XML 数据模式化的两个主要实现方法：DTD 和 XML 模式也进行了详细的讨论。这一部分的最后讨论了名字空间，名字空间提供了在 XML 词汇中元素和属性名的唯一性。
- 第 2 部分：格式化 XML 文件。这一部分描述了样式表以及两种主要的用来格式化 XML 文件的样式表技术：Cascading Style Sheets (CSS, 层叠样式表) 和 eXtensible Style Language (XSL, 可扩展样式语言)。有趣的话题是关于哪种技术更好的大争论。这一部分还提供了用来说明各种样式表技术细节的实际的例子。
- 第 3 部分：XML 文件的处理。这一部分讨论的是大多数 XML 应用的关键功能。在

说明了 XML 处理的基础之后,这一部分介绍了用 Java 及 C++ 来解析 XML 的方法。同时还提供了对两种主流的 XML 处理 API——文件对象模型(DOM)和 XML 的简单 API(SAX)的详细讨论。

- 第 4 部分:XML 与 Web。这一部分主要讲述了 XML 是为 Web 而设计的这一事实。Web 技术主要决定于支持它的浏览器,所以这一部分主要讨论浏览器对 XML 的支持。你将学到许多 XHTML 的许多知识,XHTML 是 HTML 和 XML 的结合,它指明了 Web 的未来。对 XML 链接技术:XLink 和 XPointer 也进行了详细的讨论,同时讲述了如何用脚本语言来处理 XML。
- 第 5 部分:使用 XML 操作数据。这一部分讨论了 XML 数据访问。XML 的关键应用之一就是在数据库之间即时地交换数据。这一部分介绍了辅助 XML 这一功能的技术,诸如 XML 数据源对象(DSO)和 XML 查询语言(XQL)。你将会看到如何使用 XML 在 Active Server Pages 中访问数据库。最后讨论了 XML 怎样适应于 WIDL(Web 接口定义语言)和 EDI(电子数据交换)的商业数据交换。
- 第 6 部分:XML 工具。这一部分介绍了一些简化 XML 文件和应用创建过程的工具。你将会学习使用 XML 编写和正文管理工具,而不是那些用来实现非常特殊的 XML 任务的特殊工具。
- 第 7 部分:XML 词表探讨。这一部分实际上是本书最有趣的部分。因为这一部分探讨了在现实世界中有哪些公司和个人使用了 XML。这一部分非常实用(你将会看到几个现成的例子)而且涉及了令人难以置信的广泛课题——执行了诸如标出新闻故事、表示多媒体数据,构建 3D 世界、通过 Web 管理隐私、生成简历和求职广告,描述语音响应的应用等等功能的 XML 词汇。

谁应该读这本书

尽管这本书的读者可以是任何对学习 XML 感兴趣的人,但是它更适合于那些想要随时获取 XML 细节的 Web 开发者。如果读者曾对 XML 有所了解或者曾经创建过 XML 文件,那么这对于阅读本书是很有帮助的。千万要记住:XML 和 HTML 密切相关,所以如果你有 Web 页面编写经验,那么你能够很快学会 XML。问题是这本书涉及到了 XML 的使用。因而你可以立即着手用 XML 做一些有趣的事情。本书还涉及了一些非常实用的工业标准 XML 应用,它们可以被用来完成从构建语音响应应用到创建 3D 世界的任何工作。

如果你非常想钻研 XML 技术并了解它能够做什么,那么这本书正适合你。你最好是顺序地一直读下去,不过你也可以直接跳到你特别感兴趣的章节。最重要的是,我鼓励你保持良好的心情并享受发现 XML 技术力量的乐趣。

本书中的约定

本书遵守下列约定:占位符可以被替换为实际的文件名、参数,或者其他任何它所表示的元素。作为揭秘系列图书,本书还包含一些注释、提示以及警告,来帮助读者更快地发现重点和有用的信息。本书中所引用的所有代码都可以在随书的 CD-ROM 中找到。

告诉我们您的想法

作为本书的读者,您是本书最重要的批评者和评论者。我们非常重视您的观点并且希望了解我们哪些做得不错,哪些还能做得更好,您希望我们出版哪些内容,以及您所愿意给予我们的任何建议。

作为 Sams 的一位出版者,我非常欢迎您的建议。您可以发传真、电子邮件或者直接给我写信,让我知道这本书中您所喜欢的和不喜欢的地方,以及我们怎样才能把书写得更好。

请注意,我无法回答您对本书中的技术问题,这是由于我已经收到了很多信件,因而无法一一回答。

在您的来信中,请务必包括本书的标题和作者,以及您的名字和电话或传真。我会认真考虑您的建议并和本书的所有作者和编辑共享。

Fax: 317-581-4770
Email: office_sams@mcp.com
Mail: Publisher
Michael Stephens
Sams
201 West 103rd Street
Indianapolis, IN 46290 USA

目 录

第 1 部分 定义 XML 文件

第 1 章 XML 新手	(1)
1.1 XML 历史简介	(1)
1.2 XML 是怎样的	(2)
1.3 XML 语法基础	(4)
1.3.1 标记	(6)
1.3.2 实体引用	(6)
1.3.3 注释	(7)
1.3.4 处理指令	(7)
1.3.5 文件类型声明	(8)
1.3.6 CDATA 片段	(8)
1.4 XML 的现状	(9)
1.4.1 XML 的显示样式	(9)
1.4.2 浏览 XML	(9)
1.4.3 解析 XML	(10)
1.5 总结	(11)
第 2 章 数据模型:文件类型定义和 Schema	(12)
2.1 XML 数据模型基础	(12)
2.2 用 DTD 来建立数据模型	(15)
2.3 用 XML Schema 建立数据模型	(17)
2.4 比较两种建立数据模型的方法	(18)
2.5 总结	(20)
第 3 章 DTD 基础	(21)
3.1 DTD 和文件结构	(21)
3.1.1 内部 DTD 和外部 DTD	(22)
3.1.2 有效的和结构良好的文件	(24)
3.2 理解元素和属性	(25)
3.3 使用元素	(27)
3.3.1 空元素	(27)
3.3.2 只含子元素的元素	(28)
3.3.3 混合元素	(29)
3.3.4 ANY 元素	(31)
3.4 使用属性	(31)
3.4.1 字符串属性	(32)

3.4.2 枚举属性	(33)
3.4.3 标记属性	(33)
3.4.4 声明多重属性	(34)
3.5 用 DTD 创建有效的文件	(34)
3.6 总结	(36)
第 4 章 进一步深入 DTD	(37)
4.1 实体和文件结构	(37)
4.2 字符和实体	(38)
4.3 使用实体	(40)
4.3.1 一般实体	(41)
4.3.2 参数实体	(42)
4.4 外部实体	(43)
4.4.1 引用非解析实体	(43)
4.4.2 使用外部参数实体	(44)
4.5 声明表示法	(45)
4.6 在文件中引用实体	(46)
4.7 使用条件段	(48)
4.8 总结	(49)
第 5 章 XML Schema 基础	(50)
5.1 了解 XML Schema	(50)
5.1.1 XML Schema 和 W3C	(50)
5.1.2 XML Schema 的好处	(51)
5.2 深入 XML Schema 词表	(52)
5.2.1 schema 元素	(54)
5.2.2 datatype 元素	(55)
5.2.3 ElementType 元素	(55)
5.2.4 element 元素	(57)
5.2.5 group 元素	(58)
5.2.6 AttributeType 元素	(59)
5.2.7 attribute 元素	(60)
5.2.8 description 元素	(60)
5.3 XML Schema 数据类型	(61)
5.4 从 XML Schema 创建文件	(62)
5.5 总结	(64)
第 6 章 XML Schema 构造技术	(65)
6.1 将 DTD 转换为 Schema	(65)
6.2 根据 Schema 验证文件	(69)
6.3 用 XML Authority 生成 Schema	(70)
6.4 理解内容模型	(75)
6.5 XML Schema 的未来	(76)
6.5.1 元素继承	(76)
6.5.2 受约束的数据类型	(76)

6.6 总结	(77)
第7章 使用XML名字空间	(78)
7.1 名字空间基础	(78)
7.2 声明名字空间	(79)
7.2.1 缺省声明	(80)
7.2.2 显式声明	(80)
7.3 将 Schema 作为名字空间引用	(81)
7.4 在 Schema 中使用名字空间	(87)
7.5 名字空间和文件对象模型	(89)
7.6 总结	(90)
 第2部分 格式化XML文件	
第8章 对比两种样式规范:可扩展样式表语言和层叠样式表	(91)
8.1 样式表基础	(91)
8.1.1 HTML 是一种非常糟糕的描述语言	(91)
8.1.2 将描述踢出 HTML	(92)
8.1.3 通过样式表显示 XML	(92)
8.2 CSS 和 XSL 发展史	(93)
8.2.1 理解 CSS	(94)
8.2.2 理解 XSL	(95)
8.3 XSL 与 CSS 的比较	(95)
8.4 XSL 和 CSS 联合使用	(96)
8.5 总结	(97)
第9章 使用层叠样式表格式化XML	(98)
9.1 CSS 初步	(98)
9.2 深入CSS样式表	(99)
9.3 CSS 样式属性	(101)
9.3.1 display 属性	(101)
9.3.2 width 和 height 属性	(102)
9.3.3 border 属性	(102)
9.3.4 margin 属性	(102)
9.3.5 color 属性	(103)
9.3.6 text 属性	(103)
9.3.7 font 属性	(103)
9.4 创建CSS样式表	(103)
9.5 总结	(111)
第10章 理解XSL	(112)
10.1 处理XSL样式表	(112)
10.2 XSL 体系结构	(113)
10.2.1 XSL 转换	(114)

10.2.2 XPath	(114)
10.2.3 XSL 格式化对象	(114)
10.3 总结	(115)
第 11 章 创建 XSL 样式表	(116)
11.1 XSL 与 Internet Explorer 5.0	(116)
11.2 深入 XSL 样式表	(116)
11.2.1 模板	(117)
11.2.2 模式	(118)
11.3 XSLT 模板结构	(118)
11.3.1 <code>xsl:value-of</code> 元素	(119)
11.3.2 <code>xsl:if</code> 元素	(119)
11.3.3 <code>xsl:for-each</code> 元素	(119)
11.3.4 <code>xsl:apply-templates</code> 元素	(120)
11.4 开发 XSL 样式表	(120)
11.5 总结	(132)
 第 3 部分 XML 文件的处理 	
第 12 章 XML 处理基础	(133)
12.1 XML 文件的处理	(133)
12.2 为什么要解析 XML	(134)
12.3 为什么要验证 XML 文件	(135)
12.3.1 文件类	(135)
12.3.2 用 DTD 来验证内容	(136)
12.3.3 在 XML EDI 中使用 DTD	(137)
12.3.4 用 DOM 来使用 DTD	(138)
12.4 深入 XML 解析器	(138)
12.4.1 解析树	(138)
12.4.2 结构良好的解析器	(140)
12.4.3 验证解析器	(142)
12.5 解析 XML 的两种方案的比较	(144)
12.5.1 将文件作为树来解析	(144)
12.5.2 将文件作为平面数据结构来解析	(147)
12.6 作为对象的解析器	(149)
12.7 总结	(149)
第 13 章 用 Java 解析 XML	(151)
13.1 Java 的 XML 解析器库剖析	(151)
13.1.1 核心功能	(152)
13.1.2 核心 API	(152)
13.1.3 这个核心没有揭示 DTD	(152)
13.2 市面上的工具	(153)
13.2.1 Sun 的 Java Project X package	(154)

13.2.2	IBM 的 XML4J v2 包	(154)
13.2.3	Oracle 的 v2 XML 包	(155)
13.2.4	DataChannel 的 XJParser 包	(155)
13.2.5	Microstar 的 Aelfred 解析器	(156)
13.2.6	James Clark 的 XP 解析器	(156)
13.2.7	其他 XML 解析器	(156)
13.3	用 Java 解析 XML 文件	(158)
13.3.1	使用 SAX 解析器	(159)
13.3.2	取得一个 DOM 文件	(160)
13.3.3	在解析之后;DOM 扩展	(167)
13.4	总结	(172)
第 14 章	用 C++ 解析 XML	(173)
14.1	为什么用 C 编写解析器	(173)
14.2	Expat 解析器	(173)
14.2.1	取得和使用 Expat	(174)
14.2.2	用 Expat 解析	(174)
14.3	IBM 的 C++ XML 解析器	(175)
14.3.1	取得和使用 IBM 解析器	(175)
14.3.2	用 IBM 解析器验证文件的有效性	(176)
14.4	Microsoft 的 C 解析器	(178)
14.4.1	Xmlint.exe 解析器	(178)
14.5	MSXML 解析器	(178)
14.5.1	获得并使用解析器	(180)
14.5.2	实现解析器	(180)
14.5.3	MS 解析器错误处理	(181)
14.5.4	MS 解析器的选项和模式	(182)
14.5.5	用 IE5 打开 XML 文件	(183)
14.6	用 MSXML 构建解析器接口	(184)
14.6.1	在解析器中详述脚本	(188)
14.6.2	使用 MSXML 解析器解析和验证 XML 文件	(189)
14.7	MSXML 解析器的错误	(190)
14.7.1	巨大 DTD 的不正确操作	(190)
14.7.2	DOM 中的不正确语法	(190)
14.8	在文件和 DTD 设计中使用 C 解析器	(190)
14.8.1	设计文件类型	(190)
14.8.2	调试模板文件	(191)
14.8.3	检查 DTD 语法	(191)
14.8.4	使用 C++ 解析器调试 DTD	(191)
14.8.5	解释错误信息	(192)
14.9	总结	(192)
第 15 章	使用文件对象模型	(193)
15.1	W3C 和 XML DOM	(193)

15.1.1 DOM 的 API 接口	(193)
15.2 文件对象模型	(194)
15.3 文件树和解析树	(195)
15.4 W3C DOM 基础	(196)
15.4.1 DOM 对象	(196)
15.5 DOM 接口	(196)
15.6 节点和对象	(197)
15.7 访问 DOM 中的节点	(198)
15.7.1 遍历树	(198)
15.7.2 用名字来访问节点	(198)
15.8 DOM 方法返回的数据类型	(198)
15.9 Document 接口	(200)
15.9.1 Document 接口的只读属性	(200)
15.9.2 Implementation 属性	(201)
15.9.3 Document 接口的方法	(201)
15.9.4 使用 Node 和 nodeList 接口	(202)
15.9.5 使用 getElementsByTagName()方法	(204)
15.10 DocumentFragment 接口	(204)
15.11 node 接口	(205)
15.11.1 node 接口的属性	(205)
15.11.2 节点属性特征的调查	(207)
15.11.3 node 接口的方法	(209)
15.11.4 使用节点方法:示例 A	(211)
15.11.5 使用节点方法:示例 B	(212)
15.12 NodeList 接口	(213)
15.12.1 item()方法	(213)
15.13 NamedNodeMap 接口	(213)
15.13.1 getNamedItem(),setNamedItem()和 removeNamedItem()方法	(213)
15.13.2 length 属性	(213)
15.14 CharacterData 接口	(215)
15.14.1 data 属性	(215)
15.14.2 length 属性	(215)
15.14.3 subStringData 方法	(215)
15.14.4 appendData()方法	(215)
15.14.5 insertData()方法	(215)
15.14.6 deleteData()方法	(216)
15.14.7 replaceData()方法	(216)
15.15 Attr 接口	(218)
15.15.1 name 属性	(218)
15.15.2 value 属性	(218)
15.15.3 specified 属性	(218)
15.16 Element 接口	(218)
15.16.1 处理属性的方法	(219)

15.16.2 其他 element 接口的属性和方法	(220)
15.17 不常用的 DOM 接口	(220)
15.17.1 text 接口	(220)
15.17.2 CDATASection 接口	(220)
15.17.3 DocumentType 接口	(220)
15.17.4 Notation 接口	(221)
15.17.5 Entity 和 EntityReference 接口	(222)
15.17.6 ProcessingInstruction 接口	(224)
15.18 实现 DOM	(224)
15.18.1 Microsoft 的 DOM 引擎	(224)
15.18.2 IBM 的 DOM 引擎	(224)
15.19 为应用程序加入 DOM 支持	(225)
15.19.1 Java	(225)
15.19.2 C++	(225)
15.19.3 Visual Basic	(225)
15.19.4 ASP	(225)
15.20 脚本和 DOM	(225)
15.20.1 使用 C++ DSO	(225)
15.20.2 使用 Microsoft XMLDOM ActiveX 数据对象(ADO)	(226)
15.21 Gecko 中的 DOM 支持	(227)
15.22 永久保存文件	(227)
15.23 总结	(228)
第 16 章 在 Java 中使用 SAX API	(229)
16.1 SAX 1.0 的结构	(229)
16.2 SAX 1.0 包基础	(230)
16.2.1 解析 XML 文本	(230)
16.2.2 HandlerBase:统一的 SAX 处理函数	(231)
16.2.3 SAX DocumentHandler 接口	(231)
16.3 SAX 应用程序的例子	(234)
16.3.1 简单的模板处理	(234)
16.3.2 构建 DOM 树	(236)
16.4 SAX 的其他核心功能	(238)
16.4.1 使用 Parser.parse()的其他方法	(238)
16.4.2 ErrorHandler 对象	(240)
16.4.3 InputSource 对象	(241)
16.4.4 其他 DocumentHandler 回调	(243)
16.4.5 不是所有的解析器都一样	(244)
16.5 SAX 的高级特征	(245)
16.5.1 对外部解析实体使用 EntityResolver	(245)
16.5.2 诊断 Locale	(247)
16.5.3 对未解析实体和注释使用 DTD Handler	(248)
16.6 SAX 2.0	(249)

16.6.1 解析器特征标志	(250)
16.6.2 解析器属性值	(250)
16.7 总结	(251)

第 4 部分 XML 与 Web

第 17 章 浏览 XML

(252)

17.1 XML 和 Web 浏览器现状	(252)
17.2 Microsoft Internet Explorer	(253)
17.2.1 XML 原始代码	(253)
17.2.2 XML 错误处理	(253)
17.2.3 CSS 和 XSL	(255)
17.2.4 XML 名字空间	(256)
17.2.5 XML Schema	(256)
17.2.6 CDF 和 VML	(257)
17.3 Netscape Navigator(Mozilla)	(257)
17.3.1 CSS 和 XSL	(258)
17.3.2 Expat 解析器	(258)
17.3.3 XUL 和 RDF	(258)
17.4 其他浏览器	(259)
17.4.1 CITEC DocZilla	(259)
17.4.2 W3C Amaya	(259)
17.4.3 Opera	(260)
17.5 总结	(260)

第 18 章 XHTML;XML 与 HTML 结合的产物

(261)

18.1 为什么要用 XHTML	(261)
18.2 XHTML 和 HTML 4.0 的差别	(262)
18.3 XHTML 和文件的有效性	(263)
18.3.1 声明 XHTML DTD	(264)
18.3.2 声明 XHTML 名字空间	(264)
18.3.3 验证 XHTML 文件	(264)
18.4 创建 XHTML 文件	(265)
18.5 将 HTML 文件转化为 XHTML	(267)
18.5.1 手工转换文件	(268)
18.5.2 使用 HTML 整理工具	(271)
18.6 总结	(273)

第 19 章 使用 XLink 和 XPointer 链接文件

(274)

19.1 链接 HTML 的方法	(274)
19.2 超越传统 HTML 链接	(275)
19.3 W3C 链接规范	(276)
19.3.1 XPath	(277)
19.3.2 XPointer	(278)

19.3.3 XLink	(278)
19.4 深入 XLink	(279)
19.4.1 链接的类别	(280)
19.4.2 XLink 属性	(280)
19.4.3 创建 XLink	(281)
19.5 深入 XPointer	(283)
19.5.1 位置路径	(283)
19.5.2 创建 XPointer	(284)
19.6 总结	(285)
第 20 章 编写 XML 脚本	(286)
20.1 为什么编写 XML 脚本	(286)
20.2 XML 脚本选择	(288)
20.2.1 JavaScript	(288)
20.2.2 VBScript	(289)
20.2.3 Perl	(289)
20.2.4 Python	(289)
20.2.5 AppleScript	(290)
20.2.6 Tcl	(290)
20.3 JavaScript 入门	(290)
20.3.1 语句	(290)
20.3.2 变量	(291)
20.3.3 表达式	(291)
20.3.4 注释	(291)
20.3.5 函数	(291)
20.3.6 对象	(292)
20.4 再谈 XML DOM	(292)
20.4.1 XMLDOMDocument 对象	(293)
20.4.2 XMLDOMNode 对象	(294)
20.4.3 XMLDOMNodeList 对象	(295)
20.4.4 XMLDOMNamedNodeMap 对象	(296)
20.4.5 XMLDOMElement 对象	(296)
20.4.6 XMLDOMAttribute 对象	(297)
20.4.7 XMLDOMText 对象	(297)
20.4.8 XMLDOMParseError 对象	(297)
20.5 开发 XML 脚本	(298)
20.6 总结	(307)

第 5 部分 使用 XML 操作数据

第 21 章 XML 数据源对象	(309)
21.1 数据库和远程数据访问	(309)
21.1.1 数据是如何组织的	(309)