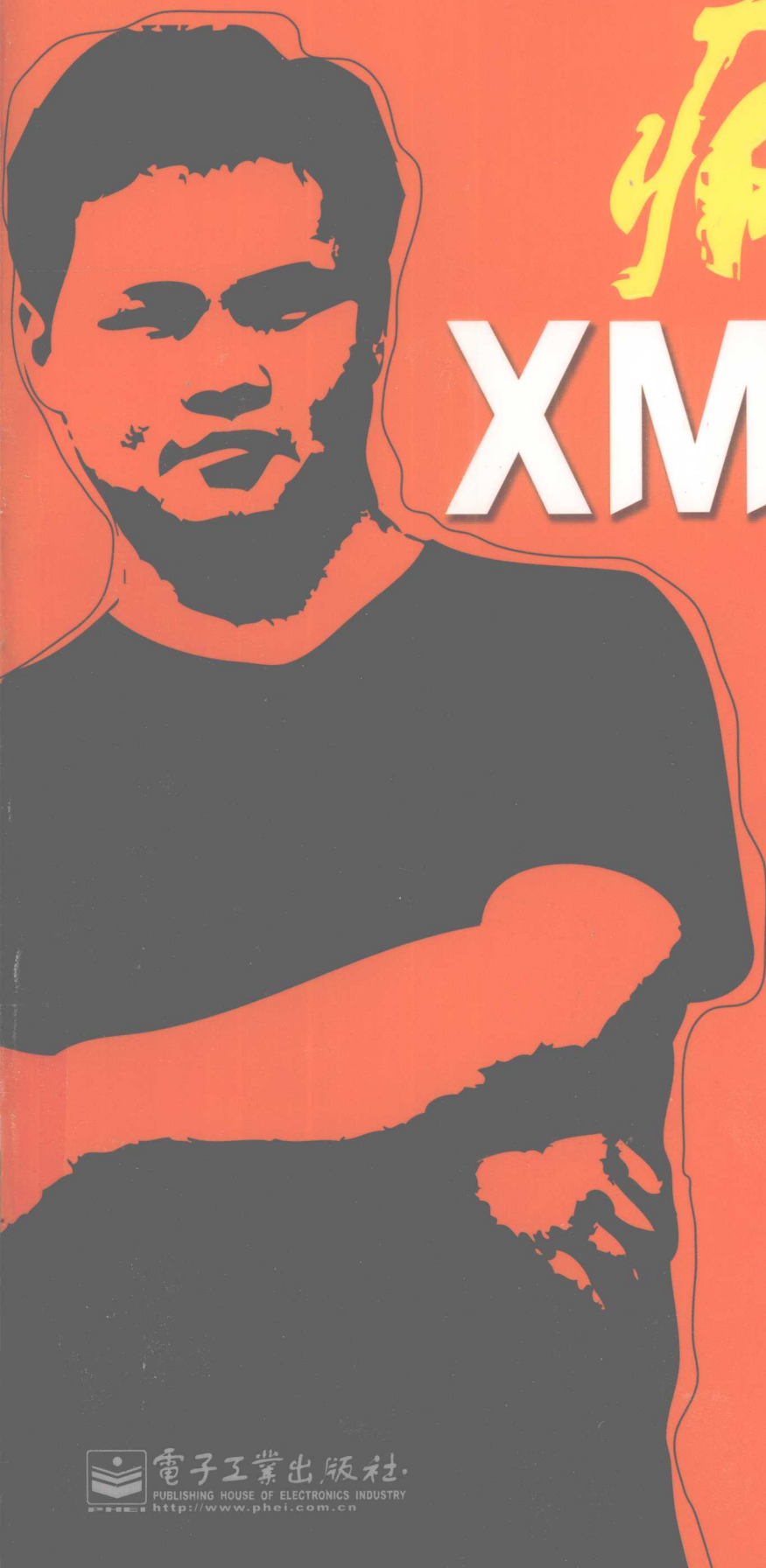
Broadview
www.broadview.com.cn

疯狂 XML讲义

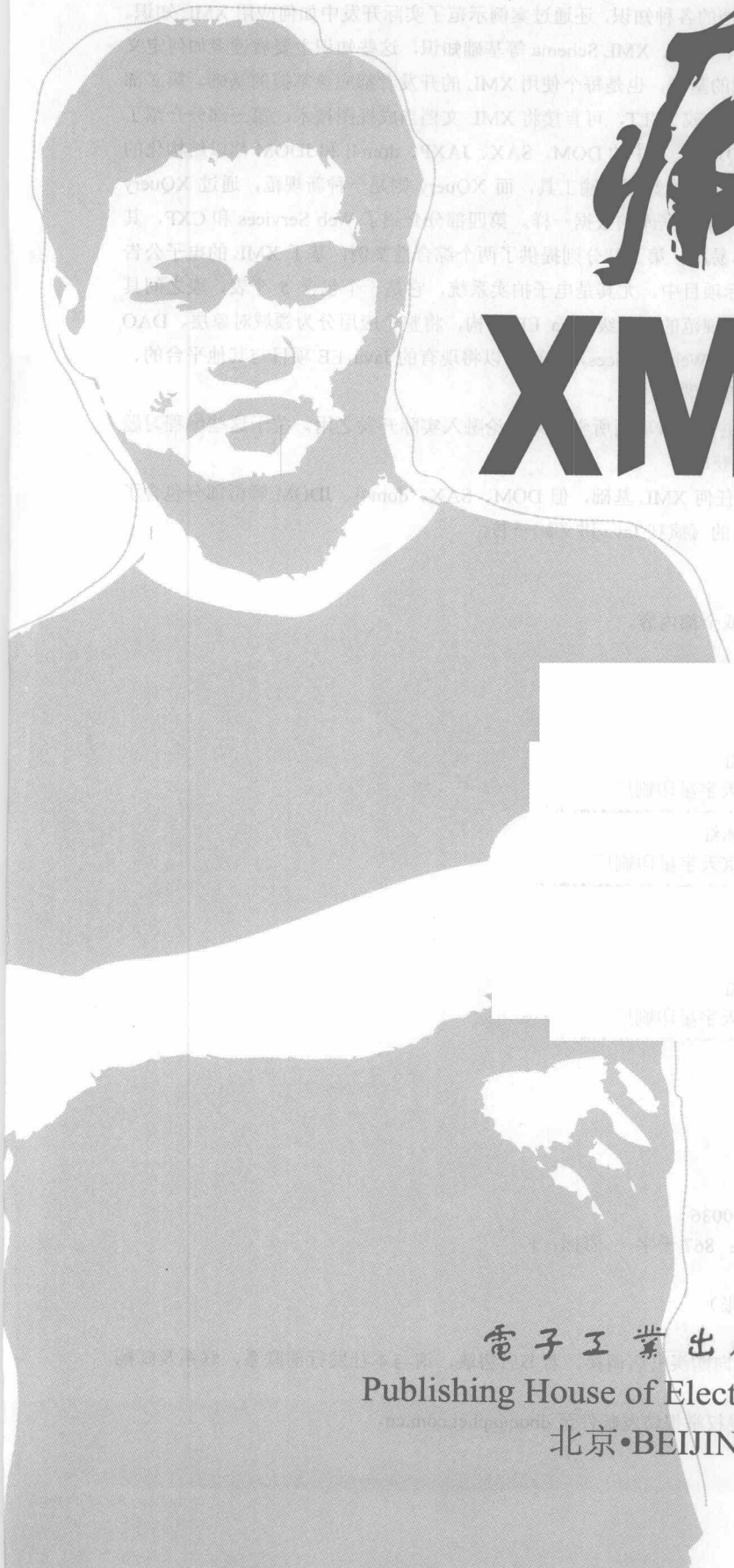
李刚 编著



疯狂源自梦想
技术成就辉煌

疯狂源自梦想
技术成就辉煌

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn



疯狂XML讲义

李刚 编著

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内 容 简 介

本书主要以 XML 为核心，深入地介绍了 XML 的各种相关知识。本书作为疯狂 Java 体系图书之一，依然保持该体系图书系统、全面的特点：不仅详细介绍了 XML 文档的各种知识，还通过案例示范了实际开发中如何应用 XML 知识。

本书主要分为五个部分。第一部分介绍了 XML、DTD、XML Schema 等基础知识，这些知识主要教读者如何定义有效的 XML 文档，这部分内容是深入学习后面知识的基础，也是每个使用 XML 的开发者都应该掌握的基础。第二部分介绍了 CSS、XSLT 和 XPath 等知识，通过使用 CSS 或 XSLT，可直接将 XML 文档当成视图技术。第三部分介绍了 DOM、SAX、JAXP、dom4j、JDOM、XQuery 和 XQJ 等，其中的 DOM、SAX、JAXP、dom4j 和 JDOM 都以结构化的方式来创建、解析 XML 文档，从而可以将 XML 文档作为数据传输工具，而 XQuery 则是一种新规范，通过 XQuery 可以查询 XML 文档中的数据，就像使用 SQL 查询关系数据库的数据一样。第四部分介绍了 Web Services 和 CXF，其中 CXF 是 Java 领域的主流 Web Services 框架，简单易用。第五部分则提供了两个综合性案例：基于 XML 的电子公告系统和电子拍卖系统，让读者将前面所学应用到实际项目中。尤其是电子拍卖系统，它是一个包含 5 个表，表之间具有复杂关联映射的系统，该案例采用目前最流行、最规范的轻量级 Java EE 架构，将整个应用分为领域对象层、DAO 层和业务逻辑层，然后用 CXF 将业务逻辑组件包装成 Web Services，从而可以将现有的 Java EE 项目与其他平台的、其他语言的异构项目进行整合，具有极好的指导价值和借鉴意义。

本书大部分章节后都提供了相应的编程习题，供开发者巩固所学，将理论融入实际开发之用。关于这些编程习题的解题思路和参考答案可登录 http://www.crazyit.org 获取。

本书是疯狂 Java 体系丛书之一，学习本书无须任何 XML 基础，但 DOM、SAX、dom4j、JDOM 解析部分包含了大量 Java 编程，因此建议先认真阅读疯狂 Java 体系的《疯狂 Java 讲义》一书。

图书在版编目（CIP）数据

疯狂 XML 讲义 / 李刚编著. —北京：电子工业出版社，2009.11
ISBN 978-7-121-09755-3

I. 疯… II. 李… III. 可扩充语言，XML—程序设计 IV. TP312

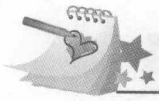
中国版本图书馆 CIP 数据核字（2009）第 194427 号

责任编辑：朱沭红
印　　刷：北京天宇星印刷厂
装　　订：三河市鑫金马印装有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编 100036
开　　本：850×1168　1/16　印张：31.75　字数：867 千字　彩插：1
印　　次：2009 年 11 月第 1 次印刷
印　　数：3500 册　　定价：65.00 元（含光盘 1 张）

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。



前　言

自从 1998 年问世以来，XML 已经广泛应用于各种开发平台、各种编程语言中，并衍生出大量新的标记语言：如 SVG（Scalable Vector Graphics）、SMIL（Synchronized Multimedia Integration Language）等，这些都充分说明了 XML 强大的生命力和广泛的引用前景。

对于大部分 Java 开发者而言，使用 XML 最多的地方还是配置文件，不管是 Java Web 应用的初学者，还是 Struts、Spring 等开源框架的学习者，几乎每天都会接触到大量的 XML 文档。因此大部分开发者都会把 XML 文档和配置文件等同起来，但实际上 XML 的应用是非常广泛的。本书系统而深入地介绍了 XML 以及 XML 相关方面的内容，并通过实际案例示范了 XML 在实际开发中的应用，本书将带领读者系统而深入地掌握 XML 的相关知识。

本书大部分章节后都提供了相应的编程习题，供学习者巩固所学、将理论融入实际开发之用。关于这些编程习题的解题思路和参考答案可登录 http://www.crazyit.org 获取。

本书有什么特点

本书写作过程中大量参考了 http://www.w3c.org 站点中关于 DTD、XML Schema、XSLT、XPath、XQuery 等技术的最新规范。也参考了微软 MSDN 里关于 XML 的内容，主要由 http://msdn.microsoft.com/zh-cn/library/ms256177.aspx 和 http://msdn.microsoft.com/zh-cn/library/ms256177(VS.80).aspx 两个页面提供。

本书旨在系统、深入地介绍 XML 以及相关技术，希望读者能通过本书更全面地掌握、使用 XML 的相关知识。归纳起来，本书大致有如下 3 个特点：

1. 知识全面、专业性强

本书并不是一本关于 XML 基础的图书，而是系统且深入地介绍 XML 相关知识，不仅包括 DTD、XML Schema 等基础内容，也包括 XSLT、XPath、XQuery、DOM、SAX、JDOM 和 dom4j 等高级内容。除此之外，还介绍了 XML 在实际企业开发中一个重要的应用方向：Web Services，并介绍了 Java 领域的 Web Services 框架 CXF 的用法。这些知识点覆盖了 XML 相关的绝大部分内容，专业性非常强。

2. 案例实际、实用性强

本书前面各章讲解各小知识点时，不仅介绍了它们的基本语法，还提供了与之对应的小范例，希望读者能理论结合实际，边读边练，通过编码来掌握相应的知识。本书除了介绍 XML 及其相关知识之外，更重点介绍了如何将 XML 知识融入实际应用开发，例如书中最后两个案例，都是 XML 在实际企业开发中极好的范例。

3. 深入浅出、易读性强

本书与疯狂 Java 体系的其他图书一样，即使在介绍专业性极强的知识时，也尽量使用深入浅出的类比来帮助读者理解它们，然后配合针对性很强、重点突出的小示例来说明这些知识，降低初学者的学习难度。通过这种案例驱动的讲解方式，让读者阅读时更容易理解各知识点的功能；接下来又能迅速将所学知识应用于实际开发。

本书创作感言

动手写作本书之前，博文视点的朱沭红老师已经善意地提醒笔者：关于 XML 的书都很难卖，可能大家都觉得 XML 太简单了。想一想也是，几乎所有学习 Java 开发的读者都应该接触过 XML 文档：当你开始学习 JSP 开发时，你就得写 web.xml 配置文件；当你学习 Struts 时，你又得写 struts-config.xml 配置文件。几乎所有 Java 学习者都接触过 XML 文档，所以大家可能觉得 XML 如此简单，怎么可能需要一本书来介绍呢？

很多人可能觉得 XML 非常简单，因为他们每天都会接触 XML 文档，但如果再深入问一下：XML 除了做配置文件之外，还能干什么？他们往往一无所知。

他们知道 XML 可以做配置文件，但可能不知道 XML 配置文件的结构是由 DTD、Schema 控制的——因为我经常看到有人问 web.xml 文件能包含哪些元素？如何将 JSP 文件配置成一个 Servlet？等等诸如此类的问题，如果他知道阅读 web.xml 文件对应的 DTD、Schema，那他就不会再有这些疑问了。

他们可能不知道 XML 还可以作为一种轻量级的持久化解决方案，就像数据库一样，应用程序可以用类似于 SQL 的 XQuery 来查询 XML 文档中的信息。他们可能不知道 XML 可以直接作为表现层来代替传统的 HTML……

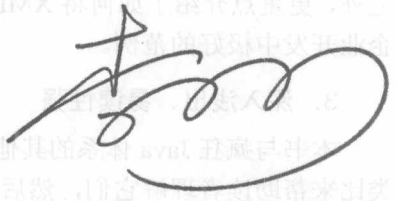
后来我觉得，如果确实存在上面这些情况，那我就更应该写这样一本书：全面而深入地来介绍 XML 的相关知识，希望可以让大家掌握 XML 的全貌。作为配置文件只是 XML 的功能之一，而 XML 的作用远不止这些。

最后我和朱沭红老师达成一个共识：出版一本图书除了考虑经济上的回报之外，更多地应该考虑一本书对社会价值的作用。就我以一个老程序员的眼光来看，XML 不管是对于 Java 平台、还是其他如.NET 平台都具有非常重要的作用。就像做 B/S 开发的程序员，你可以选择 Java、PHP 或者其他，但你无法回避 JavaScript；XML 也是如此，它不仅对于 Java 平台是有用的，而且对于其他开发平台也是有用的。

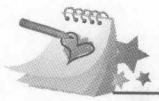
由于 XML 技术有着广泛的用途，而广大 Java 学习者、开发者又对其重视不够，因而我们觉得出版这样一本书也许可以让大家对 XML 投以更多的目光，那这本书的价值也就体现出来了。

本书写给谁看

本书是疯狂 Java 体系丛书之一，学习本书无须任何 XML 基础，但 DOM、SAX、dom4j、JDOM 解析部分包含了大量 Java 编程，因此建议先阅读疯狂 Java 体系的《疯狂 Java 讲义》一书。如果时间允许，建议读者按照本书所附的学习线路图，遵循学习规律进行学习。



2009-10-12



光盘说明

一、光盘内容

本光盘是《疯狂 XML 讲义》一书的配书光盘，书中的代码按章、节存放，即第 2 章、第 2 节所使用的代码放在 codes 文件夹的 02\2.2 文件夹下，依次类推。

另：书中每份源代码也给出与光盘源文件的对应关系，方便读者查找。

本光盘 codes 目录下有 16 个文件夹,其内容和含义说明如下：

01～16 个文件夹名分别对应于《疯狂 XML 讲义》中的章名，即第 2 章所使用的代码放在 codes 文件夹的 02 文件夹下，依次类推。

其中 16 文件夹下有 auction 和 auctionClient 两个文件夹，其中 auction 文件夹包含的是应用服务端，部署后该应用将对外提供 Web Services，而 auctionClient 是用测试 Web Services 的客户端程序。

两个文件夹下都是与 IDE 平台无关的项目，使用 Ant 来编译即可。

二、运行环境

本书中的程序在以下环境调试通过：

安装 jdk-6u14-windows-i586-p.exe，安装完成后，添加 CLASSPATH 环境变量，该环境变量的值为；%JAVA_HOME%/lib/tools.jar;;%JAVA_HOME%/lib/dt.jar。如果为了可以编译和运行 Java 程序，还应该在 PATH 环境变量中增加%JAVA_HOME%/bin。其中 JAVA_HOME 代表 JDK（不是 JRE）的安装路径。

安装 Apache 的 Tomcat6.0.20，不要使用安装文件安装，而是采用解压缩的安装方式。

安装 Tomcat 请参看《轻量级 Java EE 企业应用实战》第 1 章。安装完成后，将 Tomcat 安装路径的 lib 下的 jsp-api.jar 和 servlet-api.jar 两个 JAR 文件添加到 CLASSPATH 环境变量之后。

安装 apache-ant-1.7.1。

将下载的 Ant 压缩文件解压缩到任意路径，然后增加 ANT_HOME 的环境变量，让变量的值

为 Ant 的解压缩路径。

并在 PATH 环境变量中增加%ANT_HOME%/bin 环境变量。

安装 MySQL5.0 或更高版本，安装 MysSQL 时候选择 GBK 的编码方式。

安装 XMLSpy 2008 或者更高版本，关于 XMLSpy 2008 的安装方式请参考本书第 1 章内容。

安装 Stylus Studio 2009 或者更高版本，关于 Stylus Studio 2009 的安装方式请参考本书第 1 章内容。

三、注意事项

独立应用程序的代码中都包括 build.xml 文件，在 Dos 或 Shell 下进入 build.xml 文件所在路径，执行如下命令：

ant build -- 编译程序

ant run --运行程序

对于 Web 应用，将该应用复制到%TOMCAT_HOME%/webapps 路径下，然后进入 build.xml 所在路径，执行如下命令：ant build -- 编译应用启动 Tomcat 服务器，使用浏览器即可访问该应用。

代码中有些项目需要连接数据库，读者应修改数据库 URL 以及用户名、密码让这些代码与读者运行环境一致。如果项目下有 SQL 脚本，导入 SQL 脚本即可，如果没有 SQL 脚本，系统将在运行时自动建表，读者只需创建对应数据库即可。

在使用本光盘的程序时，请将程序拷贝到硬盘上，并去除文件的只读属性。

四、技术支持

如果您使用本光盘中遇到不懂的技术问题，您可以登录如下网站与作者联系：

网站：http://www.crazyit.org

目录 CONTENTS

第 1 章 XML 概述 …… 1
1.1 XML 的起源 …… 2
1.1.1 标记语言 …… 2
1.1.2 XML 的基本概念 …… 3
1.1.3 XML 和 HTML …… 3
1.2 XML 的优势 …… 4
1.2.1 简单易用的标记语言 …… 4
1.2.2 严格的格式 …… 5
1.2.3 数据逻辑和显示逻辑分离 …… 5
1.3 XML 和 Java EE …… 6
1.3.1 配置描述 …… 6
学生提问 指定 Web 组件的配置信息是不是只能采用 web.xml 文件呢？ …… 7
1.3.2 简化的数据交换 …… 7
1.3.3 Web Services …… 8
1.4 XML 的编辑工具 …… 8
1.4.1 普通文本编辑工具 …… 8
1.4.2 XMLSpy 简介 …… 9
1.4.3 使用 XMLSpy 编辑 XML 文档 …… 9
1.4.4 Stylus Studio 简介 …… 11
1.4.5 使用 Stylus Studio 编辑 XML 文档 …… 11
1.5 XML 的竞争对手 …… 12
1.5.1 Java 的 Annotation …… 12
1.5.2 轻量级的数据交换格式——JSON …… 15
1.6 本章小结 …… 17
第 2 章 XML 文档规则 …… 18
2.1 XML 文档的分类 …… 19
2.1.1 格式不良好的 XML 文档 …… 19
2.1.2 格式良好但无效的 XML 文档 …… 20
2.1.3 有效的 XML 文档 …… 22
2.2 XML 文档的整体结构 …… 22
2.2.1 有且仅有一个根元素 …… 22
2.2.2 元素必须合理结束 …… 23
2.2.3 元素之间必须合理嵌套 …… 24
2.2.4 元素的属性必须有值 …… 24
2.3 XML 声明 …… 26
学生提问 UTF-8 不是兼容 Unicode 吗？那 UTF-8 也应该支持中文啊，为什么上面的 XML 文档会出现字符集错误的情况呢？ …… 27
2.4 XML 元素的基本规则 …… 28
2.4.1 合法的标签名 …… 28
2.4.2 嵌套子元素 …… 29
2.4.3 空元素 …… 29
2.5 字符数据 …… 30
2.5.1 使用实体引用 …… 31
2.5.2 使用 CDATA 标记 …… 32
2.6 注释 …… 33
2.7 处理指令 …… 33
2.8 W3C 对于属性的使用建议 …… 35
2.9 换行处理 …… 36
2.10 本章小结 …… 36
本章练习 …… 36
第 3 章 DTD 详解 …… 37
3.1 XML 语义约束 …… 38
学生提问 我一直有一个疑问：XML 太“随意”了，我们想怎样定义元素都可以，想怎样嵌套子元素也行，想怎样定义属性也行，真的是这样吗？ …… 38
3.2 引入 DTD …… 39
3.2.1 内部 DTD …… 39
3.2.2 外部 DTD …… 40
3.2.3 公用 DTD …… 41
3.3 DTD 文档的结构 …… 41
3.3.1 验证 XML 文档的有效性 …… 42
3.4 定义元素 …… 42
3.4.1 定义任意类型的元素 …… 43
3.4.2 定义空元素 …… 44
3.4.3 定义字符串内容的元素 …… 45
3.4.4 定义混合内容 …… 45
3.5 定义子元素 …… 47
3.5.1 有序的子元素 …… 47
3.5.2 互斥的子元素 …… 48
3.5.3 子元素出现的频率 …… 48
3.5.4 组合子元素 …… 49
3.5.5 无序的子元素 …… 51

学生提问 上面的 DTD 中明明定义了<书名.../>、<价格.../>、<作者.../>和<简要介绍.../>4 个元素之间具有互斥关系，它们怎么可以同时出现呢？ 52
3.6 定义元素属性 53
3.6.1 对属性的约束规则 53
3.6.2 定义属性类型 55
3.7 定义实体 57
3.7.1 定义实体 57
3.7.2 定义参数实体 59
3.7.3 外部实体 60
3.7.4 外部参数实体 61
3.8 定义符号 61
3.8.1 未解析实体 62
学生提问 XML 文档里的内容不是应该由 XML 解析器负责处理吗？如果 XML 解析器不负责处理，那这些数据岂不是就失去作用了？ 62
3.8.2 ENTITY 和 ENTITIES 类型的属性 63
3.8.3 NOTATION 类型的属性 63
3.9 使用 XMLSpy 创建 DTD 64
3.10 本章小结 65
本章练习 65
第 4 章 XML Schema 基本语法 66
4.1 DTD 和 Schema 67
4.1.1 Schema 概述及其优势 67
4.1.2 Schema 在 Java EE 里的应用 68
4.2 XML Schema 入门 69
4.2.1 XML Schema 根元素 69
4.2.2 在 XML 中引用无命名空间的 Schema 70
4.2.3 在 XML 中引用有命名空间的 Schema 71
4.3 Schema 中的注释 72
4.4 理解 Schema 的数据类型 72
4.5 Schema 内置类型 74
4.5.1 字符串及相关类型 76
4.5.2 数值类型 77
4.5.3 日期、时间类型 78
4.5.4 boolean 类型 80
4.5.5 anyURI 类型 80
4.5.6 二进制数据 81
4.6 使用限制派生新类型 81
4.6.1 指定基类型的两种方式 82
4.6.2 指定类型的两种方式 84
4.6.3 范围约束 86
4.6.4 长度约束 86
4.6.5 精度约束 86
4.6.6 枚举约束 87
4.6.7 正则表达式约束 88
4.6.8 空白处理 88
4.7 使用<list.../>派生列表类型 91
4.7.1 限制列表类型 92
4.8 使用<union.../>派生联合类型 94
4.8.1 限制联合类型 96
4.9 列表和联合结合使用 97
4.10 阻止派生新的简单类型 98
4.10.1 使用 final 属性 98
4.10.2 为约束指定 fixed 属性 99
4.11 合并多个 Schema 100
4.11.1 使用 include 元素 100
4.11.2 使用 redefine 元素 101
4.11.3 使用 import 元素 102
4.12 XMLSpy 中关于 Schema 的操作 103
4.12.1 创建新的 Schema 103
4.12.2 为 XML 创建 Schema 104
4.12.3 为 XML 分配 Schema 104
4.13 本章小结 105
第 5 章 XML Schema 高级知识 106
5.1 使用 anyType 定义任意类型 107
5.2 定义复杂类型 108
5.2.1 定义复杂类型的方式 109
5.2.2 扩展简单类型 109
学生提问 既然派生复杂类型的方式有两种，那接下来是不是应该介绍“限制简单类型来派生复杂类型”了？ 110
5.2.3 包含属性的两种方式 111
5.2.4 扩展包含简单内容的复杂类型 112
5.2.5 使用派生类型的另一种方式 114
学生提问 在上面的 Schema 中先定义了一个 book_ Type 类型，然后扩展该类型增加了一个 price 属性，为何不在定义 book_Type 类型时一次性添加 isbn、name 和 price 三个属性，而非要一次一次地添加呢？这样做有实际意义吗？ 114

5.2.6 限制包含简单内容的复杂类型 ……115
5.2.7 限制 anyType 派生新类型……116
学生提问 为什么没有通过扩展 anyType 来派生新类型呢？ ……117
5.2.8 包含子元素的两种方式 …… 120
5.2.8 空元素类型 …… 123
5.2.9 混合内容类型 …… 124
5.3 复杂类型的进一步派生 …… 125
5.3.1 限制空元素类型 …… 125
5.3.2 扩展空元素类型 …… 126
5.3.3 限制包含子元素的类型 …… 127
5.3.4 扩展包含子元素的类型 …… 128
5.3.5 限制混合内容类型 …… 131
5.3.6 扩展混合内容类型 …… 133
5.4 阻止派生新的类型 …… 133
5.5 通配符 …… 135
5.5.1 元素通配符 …… 135
5.5.2 属性通配符 …… 137
5.6 元素替换 …… 138
5.6.1 阻止自己被替换 …… 139
5.6.2 阻止指定派生类型的替换 …… 141
5.7 抽象元素和抽象类型 …… 143
5.7.1 抽象元素 …… 143
5.7.2 抽象类型 …… 144
5.8 一致性约束 …… 145
5.8.1 key 约束 …… 147
5.8.2 unique 约束 …… 149
5.8.3 keyref 约束 …… 150
5.9 元素组与属性组 …… 152
5.10 定义符号 …… 153
5.11 本章小结 …… 154
本章练习 …… 155
学生提问 如何确定一个 XML 元素所在的命名空间呢？ …… 157
第 6 章 命名空间详解 …… 156
6.1 使用命名空间 …… 157
6.2 Schema 的命名空间支持 …… 159
6.2.1 在 Schema 中使用命名空间 …… 159
6.2.2 命名空间对 XML 文档的作用 …… 162
6.2.3 为属性使用命名空间限定 …… 166
6.2.4 命名空间对一致性约束的影响 …… 168
6.2.5 局部元素和局部属性的强制限定 … 170
6.3 命名空间和 DTD …… 174
6.4 本章小结 …… 176
第 7 章 使用 CSS 显示 XML 文件 …… 177
7.1 样式单简介 …… 178
7.1.1 显示 XML 的两种常用样式单 …… 178
7.1.2 样式单的优势 …… 178
7.2 CSS 的基本用法 …… 179
7.2.1 CSS 基本语法 …… 179
7.2.2 引入外部样式文件 …… 180
7.2.3 使用内部 CSS 样式 …… 181
7.2.4 使用内联样式 …… 183
7.3 对 XML 文档有效的 CSS 选择器 …… 184
7.4 使用 CSS 显示 XML …… 186
7.5 本章小结 …… 187
第 8 章 使用 XSLT 显示 XML …… 188
8.1 XSL 概述 …… 189
8.2 XSLT 入门 …… 189
8.2.1 XSLT 转换入门 …… 189
学生提问 XSLT 为什么不直接将 XML 根元素当成根元素呢？ …… 192
8.2.2 使用 template 元素定义模板 …… 192
8.2.3 使用 apply-templates 处理子节点 … 193
8.2.4 使用 value-of 输出节点内容 …… 196
8.2.5 匹配节点的模式 …… 198
8.2.6 mode 属性 …… 200
8.3 XSLT 转换分类 …… 202
8.3.1 客户端转换和服务器端转换 …… 202
8.3.2 Xalan 处理器 …… 202
8.3.3 Saxon 处理器 …… 205
学生提问 我发现用 Saxon 进行实时转换和用 Xalan 进行实时转换的 JSP 页面代码完全相同，这是为什么呢？ …… 207
8.4 XSLT 的内置模板规则 …… 208
8.5 流程控制元素 …… 209
8.5.1 分支处理 …… 209
8.5.2 循环控制 …… 211
8.5.3 排序控制 …… 213
8.6 控制空白的处理方式 …… 215
8.7 创建结果树 …… 215
8.7.1 创建元素和属性 …… 216
8.7.2 创建文本 …… 221
8.7.3 创建处理指令 …… 222
8.7.4 创建注释 …… 222
8.7.5 复制 …… 223
8.7.6 输出格式化数值 …… 225

8.8 变量和参数 …… 230
8.8.1 为变量和参数指定值 …… 231
8.8.2 全局和局部的变量和参数 …… 232
8.8.3 改变参数值 …… 234
8.9 使用命名模板 …… 235
8.10 包含和导入 …… 238
8.10.1 使用 import 导入 …… 238
8.10.2 使用 include 包含 …… 239
8.10.3 解决模板定义冲突 …… 241
8.11 XSLT 1.1 的 fallback 支持 …… 241
8.12 指定输出格式 …… 242
8.12.1 转换 XML 文档的相关格式 …… 242
8.12.2 转换 HTML 文档的相关格式 …… 244
8.13 XSLT 的内置函数 …… 245
8.13.1 使用 system-property 函数 …… 245
8.13.2 使用 current 函数返回当前节点集 …… 246
8.13.3 使用 element-available 和 function-available 函数 …… 246
8.13.4 使用 unparsed-entity-uri 函数 …… 247
8.13.5 使用 document 函数处理多个源 XML 文档 …… 248
8.13.6 使用 format-number 函数 …… 250
8.13.7 使用 key 函数 …… 251
8.13.8 使用 generate-id 函数 …… 254
8.14 XSLT 2.0 的常用新功能 …… 255
8.14.1 分组 …… 255
8.14.2 多文档输出 …… 257
8.14.3 字符映射 …… 259
8.14.4 <xsl:value-of>元素的改进 …… 260
8.14.5 数据类型绑定 …… 260
8.14.6 正则表达式支持 …… 261
8.14.7 用户自定义函数 …… 263
8.15 使用 XMLSpy 管理 XSLT 操作 …… 264
8.16 本章小结 …… 265
本章练习 …… 265
第 9 章 XPath 语言详解 …… 266
9.1 XPath 语言简介 …… 267
9.1.1 XPath 节点 …… 267
9.1.2 XPath 基本概念 …… 268
9.1.3 节点关系 …… 268
9.2 绝对路径和相对路径 …… 269
9.2.1 XPath 基路径 …… 269
9.3 XPath 基础语法 …… 270
9.3.1 轴（axis） …… 271
9.3.2 节点测试（node-test） …… 271
9.3.3 限定谓语（predicate） …… 272
9.3.4 简化写法 …… 272
9.4 运算符和表达式 …… 273
9.4.1 算术运算符 …… 273
9.4.2 比较运算符 …… 274
9.4.3 逻辑运算符 …… 274
9.4.4 组合多个路径的运算符 …… 275
9.5 XPath 2.0 新增的表达式 …… 275
9.5.1 for 表达式 …… 275
9.5.2 if 表达式 …… 278
9.5.3 some/every 判断表达式 …… 279
9.6 XPath 2.0 的类型支持 …… 280
9.7 内置函数库 …… 281
9.7.1 字符串相关函数 …… 281
9.7.2 数值相关函数 …… 283
9.7.3 日期和时间相关函数 …… 284
9.7.4 boolean 值相关函数 …… 287
9.7.5 节点相关函数 …… 287
9.7.6 序列相关函数 …… 288
9.8 本章小结 …… 290
第 10 章 DOM、SAX 和 JAXP 解析 …… 291
10.1 DOM、SAX 和 JAXP 概述 …… 292
学生提问 接口的实现不是类吗，怎么接口的实现还是接口啊？ …… 294
10.2 JAXP 的 DOM 支持 …… 297
10.2.1 XML 文档和 DOM 模型 …… 297
10.2.2 DOM 树中的对象类型 …… 298
10.2.3 DOM 解析器 …… 299
10.2.4 使用 DTD 验证 XML 文档 …… 300
10.2.5 使用 DOM 解析 XML 文档 …… 302
10.2.6 使用 DOM 创建 XML 文档 …… 304
10.2.7 使用 DOM 修改 XML 文档 …… 307
10.2.8 解析 DTD 信息 …… 308
学生提问 如果我想获取 DTD 中的元素定义、属性定义等信息该怎么办？ …… 311
10.2.9 DOM 和命名空间 …… 310
10.3 JAXP 的 SAX 支持 …… 312
10.3.1 SAX 的处理机制 …… 312
10.3.2 SAX 解析器和监听器 …… 313
学生提问 为何 XMLReader 在调用时不需要传入 SAX 解析事件的监听器呢？SAX 解析不是总是基于事件机制的吗？ …… 314

学生提问 XMLReader 和 SAXParser 到底什么关系？我们到底应该用哪个呢？ …… 316
10.3.3 使用 DTD 验证 XML 的有效性 … 316
10.3.4 使用 SAX 解析 XML 文档 …… 317
10.3.5 SAX 和命名空间 …… 320
10.4 DOM 和 SAX 的比较 …… 323
10.5 使用 XML Schema 验证 XML 文档 …… 324
10.5.1 SchemaFactory 和验证 …… 324
学生提问 上面的 SchemaFactory 的 newInstance() 方法中怎么还有一个 schemaLanguage 参数，它代表什么呢？ …… 326
10.5.2 获取节点的类型信息 …… 328
10.6 浏览器对 DOM 的支持 …… 330
10.7 本章小结 …… 334
本章练习 …… 334
第 11 章 使用 dom4j 处理 XML 文档 …… 335
11.1 dom4j 简介 …… 336
11.1.1 dom4j 的封装和优势 …… 336
11.1.2 下载和安装 dom4j …… 337
11.2 dom4j 常用 API …… 338
11.3 使用 dom4j 访问 XML 文档 …… 339
11.3.1 验证 XML 文档 …… 339
11.3.2 使用 dom4j 解析 XML 文档 …… 340
11.3.3 使用访问者模式遍历 XML 文档 … 342
11.4 使用 dom4j 创建 XML 文档 …… 344
11.5 修改 XML 文档 …… 345
11.6 使用 dom4j 获取命名空间信息 … 346
11.7 本章小结 …… 349
本章练习 …… 349
第 12 章 使用 JDOM 处理 XML 文档 …… 350
12.1 JDOM 简介 …… 351
12.1.1 JDOM 常用 API …… 351
12.1.2 下载和安装 JDOM …… 352
12.2 使用 JDOM 访问 XML 文档的实例 …… 352
12.2.1 验证 XML 文档的有效性 …… 353
12.2.2 使用 JDOM 解析 XML 文档 …… 354
12.3 使用 JDOM 创建 XML 文档 …… 356
12.4 修改 XML 文档 …… 357
12.5 使用 JDOM 获取命名空间信息 … 358
12.6 选择 dom4j 还是 JDOM …… 361
12.7 本章小结 …… 361
本章练习 …… 361
第 13 章 XQuery 详解 …… 362
13.1 XQuery 简介 …… 363
13.1.1 XQuery 和 XPath 的关系 …… 363
13.1.2 XQuery 和 XSLT 的关系 …… 363
13.2 使用 XQuery …… 364
13.2.1 使用 XMLSpy 查看 XQuery 结果 …… 365
13.2.2 使用 Saxon 执行 XQuery 查询 …… 366
13.3 基本表达式（Primary Expressions） …… 367
13.3.1 直接量（Literal） …… 367
13.3.2 变量引用 …… 367
13.3.3 圆括号表达式 …… 367
13.3.4 上下文项表达式 …… 367
13.3.5 函数调用 …… 367
13.4 序列表达式 …… 368
13.4.1 构造序列 …… 368
13.4.2 过滤表达式 …… 368
13.4.3 组合节点序列 …… 369
13.5 算术表达式 …… 370
13.6 比较表达式 …… 370
13.6.1 值比较 …… 371
13.6.2 通用比较 …… 371
13.6.3 节点比较 …… 371
13.7 逻辑表达式 …… 372
13.8 构造器 …… 373
13.8.1 直接构造 …… 373
13.8.2 计算构造 …… 373
13.9 FLWOR 表达式 …… 373
13.9.1 for、let 和 return 子句 …… 374
13.9.2 where 子句 …… 375
13.9.3 order by 子句 …… 376
13.10 if 表达式 …… 376
13.11 some/every 判断表达式 …… 377
13.12 与序列类型有关的表达式 …… 378
13.12.1 instance of …… 378
13.12.2 typeswitch …… 378
13.12.3 cast …… 378
13.12.4 castable …… 379
13.13 模块和序言 …… 379
13.13.1 序言 …… 380
13.13.2 版本声明 …… 380
13.13.3 模块声明 …… 380
13.13.4 设置器（setter） …… 381

13.13.5 导入设置 …… 382
13.13.6 命名空间声明 …… 384
13.13.7 默认命名空间声明 …… 384
13.13.8 变量声明 …… 385
13.13.9 函数声明 …… 386
13.13.10 选项声明 …… 387
13.14 使用 XQJ 执行 XQuery …… 388
13.14.1 XQJ 和 JDBC 的类比性 …… 388
学生提问 JDBC API 和接口我都知道，但 JDBC 实现好像很少听说啊？ …… 389
13.14.2 XQJ 的编程步骤 …… 389
13.15 本章小结 …… 392
第 14 章 Web Services 详解 …… 393
14.1 XML 和 Web Services …… 394
14.1.1 Web Services 概述 …… 394
14.1.2 Web Services 平台概述 …… 395
14.1.3 Web Services 的广泛应用 …… 396
14.2 SOA 和云计算 …… 398
14.2.1 SOA 和 Web Services …… 398
14.2.2 云计算和 Web Services …… 399
14.3 XML 和 SOAP 协议 …… 400
14.3.1 SOAP 基本语法 …… 400
14.3.2 Header 元素 …… 401
14.3.3 Body 元素 …… 403
14.3.4 Fault 元素 …… 403
14.4 XML 和 WSDL …… 404
14.4.1 WSDL 基本语法 …… 404
14.4.2 portType 元素 …… 409
14.4.3 binding 元素 …… 411
14.4.4 service 元素 …… 412
14.4.5 WSDL 和 UDDI …… 413
14.5 使用 CXF 开发 Web Services …… 413
14.5.1 CXF 概述 …… 414
14.5.2 下载和安装 CXF …… 414
14.5.3 使用 CXF 开发 Web Services …… 415
学生提问 是否必须将 Web Services 服务接口的.class 文件复制到客户端呢？ …… 422
14.5.4 动态客户端 …… 421
14.5.5 整合 Spring 开发 Web Services …… 423
4.5.6 为 Web Services 增加权限控制 …… 425
14.6 本章小结 …… 429
第 15 章 基于 XML 的电子公告系统 …… 430
15.1 系统设计 …… 431
15.1.1 系统组件的交互关系 …… 431
15.2 设计 XML 文档 …… 431
15.2.1 保存状态的 XML 文档 …… 432
15.2.2 定义 XML Schema …… 434
15.3 定义 XSLT 样式单 …… 436
15.3.1 为公告列表定义样式单 …… 436
15.3.2 为用户评论设计样式单 …… 438
15.4 实现控制器 …… 441
15.4.1 添加公告 …… 441
15.4.2 查看评论列表 …… 444
学生提问 为什么不直接将 XML 文档保存在 Web 应用根路径下呢？这样不是就可以让浏览者直接访问这些 XML 文档了么 …… 446
15.4.3 添加评论 …… 445
15.5 本章小结 …… 447
本章练习 …… 447
第 16 章 Web Services 案例——电子拍卖系统 …… 448
16.1 系统功能简介和架构设计 …… 449
16.1.1 系统功能简介 …… 449
16.1.2 系统架构设计 …… 450
16.2 持久层设计 …… 450
16.2.1 系统实体 …… 451
16.2.2 系统 E-R 图和数据表 …… 451
16.2.3 实现 Hibernate PO …… 453
16.2.4 管理 SessionFactory …… 461
16.3 实现系统 DAO 层 …… 462
16.3.1 DAO 的基础配置 …… 464
16.3.2 实现系统 DAO 组件 …… 464
16.3.3 配置系统 DAO 组件 …… 470
16.4 实现业务逻辑层 …… 471
16.4.1 定义业务逻辑组件接口 …… 471
16.4.2 依赖注入 DAO 组件 …… 473
16.4.3 业务逻辑组件中的异常处理 …… 474
16.4.4 处理用户竞价 …… 476
16.4.5 判断拍卖物品状态 …… 478
16.4.6 事务管理 …… 480
16.4.7 配置业务层组件 …… 481
16.5 使用 CXF 提供 Web Services …… 483
16.5.1 启用 CXF 支持 …… 484
16.5.2 实现 Web Services …… 485
16.5.3 测试 Web Services …… 490
16.6 本章小结 …… 491
本章练习 …… 492

CHAPTER 1

第1章 XML概述

本章要点

- 标记语言的历史和作用
- XML 的历史和作用
- XML 和 HTML 的联系和区别
- XML 和 HTML 相互之间的差异
- XML 的几个优势
- XML 作为配置描述文件
- XML 作为数据交换工具
- XML 和 Web Services
- 使用普通文本编辑器编写 XML 文档
- 使用 XMLSpy 编写和验证 XML 文档
- 使用 Stylus Studio 编写和验证 XML 文档
- XML 配置描述文件和 Java Annotation 的对应关系
- XML 数据交换和 JSON 数据交换的对应关系

XML 的全称是 Extensible Markup Language，即可扩展标记语言，它由 SGML（Standard Generalized Markup Language）发展而来，允许开发者自由定义标签，可以将标签和内容有效分离。不同于 HTML，XML 不再侧重于数据如何表现，而是更多地侧重于关注数据如何存储和传输。因此它逐渐演变成为了一种跨平台的数据交换格式，一种轻量级的持久化方案。通过使用 XML，开发者可以在不同平台、不同系统之间进行数据交换，还可以将程序状态保存到 XML 文件中，而无须使用关系数据库。

XML 广泛应用于 Java EE 开发的各个方面，绝大部分 Java 应用和框架都在使用 XML 作为配置文件来管理各 Java 组件，Java 作为最流行的跨平台编程语言，不可避免地需要在不同平台之间进行数据交换，而 XML 则为这种数据交换提供了支持。除此之外，XML 还是 Web Services 技术的重要基础，而 Web Services 技术则是异构系统整合的重要手段，甚至为 SOA（面向服务的架构）也提供了底层的技术实现。由此可见，XML 对于实际企业开发具有举足轻重的作用。

1.1 XML 的起源

XML 是一种应用非常广泛的标记语言，与另一种广为人知的标记语言 HTML 相似，它只是通过在文本文件中添加一些额外的标记，来传递更多附加信息。与所有标记语言一样，XML 本身并无任何“动作行为”。

➤➤1.1.1 标记语言

标记语言不是像 Java、C 一样的编程语言，它本身并无任何“动作行为”。标记语言只是用一系列约定好的标记来对电子文档进行标记，从而为电子文档额外增加语义、结构和格式等各种方面的信息。

简而言之，标记语言专为信息增加额外的标记，也就是增加一些特殊标识。例如有些阅读过的书，在某一段文字下画有一道线，这道线就是一种标记。该标记用于传递一种附加的信息：这段文字很重要。

实际上，所有的标记语言都用于为被标记内容附加一些额外的信息：例如最常见的标记语言 HTML。假设在 HTML 页面中有如下代码：

```
<font color="red"/>被标记的内容</font>
```

上述代码中的<font.../>标记用于向浏览器传递额外的信息：被<font.../>标记的文字需要采用红色显示。由此可见，HTML 页面中的标记主要用于向浏览器传递各种显示信息。

标记语言可传递非常丰富、全面的信息，不仅可传递各种显示信息，还可传递其他的各种额外信息。标记语言家族有两个先驱：GML 和 SGML，其中 GML（Generalized Markup Language，通用标记语言）由 IBM 的研究人员于 20 世纪 60 年代创建。创建 GML 的初衷是建立一种通用的文档格式，以提高系统的可移植性。显然，这种通用的文档格式必须遵守特定的标记规则。

再后来，IBM 的研究人员进一步完善并规范了 GML，将其称为 SGML（Standard Generalized Markup Language，标准通用标记语言）。1986 年，国际标准化组织（ISO）采纳 SGML 作为工业标准。今天，SGML 主要用来定义文献模型的逻辑和物理类结构等。

GML 和 SGML 是标记语言的两个先驱，但这两个先驱都没有得到广泛应用，应用最广的是另外两个标记语言：HTML 和 XML。

图 1.1 大致显示了标记语言的发展历史。

通过上面的介绍可以看出，标记语言比编程语言简单得多，只需按规定为文本文件添加一些特殊标记即可，这些特殊的标记用于传递更多额外的信息。

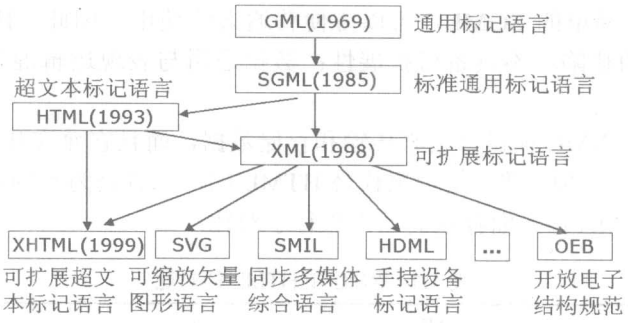


图 1.1　标记语言的发展历史

➤➤1.1.2　XML 的基本概念

XML（Extensible Markup Language，可扩展标记语言）由 SGML 简化而来，它不仅具备 SGML 的各种优势，同时还摈弃了其复杂、难于使用的缺点。

SGML 从 20 世纪 80 年代初开始应用，其主要作用是：用于创建其他的标记语言。它为语法标记提供了强大的工具，同时还具备极好的可扩展性，因此在分类和索引数据中非常有用。SGML 非常复杂，因而对于网络上的日常应用难以适应。最重要的是，包括 Netscape 在内的一些浏览器厂商明确拒绝支持 SGML，这样就妨碍了 SGML 的传播。

XML 对 SGML 进行了简化，但并没有弱化它的功能，SGML 能实现的功能，XML 几乎都可以实现，这也是 XML 最吸引人的地方。

1998 年 2 月 10 日，XML 正式成为 W3C 的推荐标准。

与 HTML 不同的是，XML 被设计用于传输和存储数据，而不是用来显示数据。

XML 是可扩展标记语言，因此没有预定义任何标签，开发者可以自行定义任意的标签。

例如如下代码定义了疯狂 Java 实训营的教材：

```
<疯狂 Java 实训营教材>
        <第 1 本教材>疯狂 Java 讲义</第 1 本教材>
        <第 2 本教材>疯狂 Ajax 讲义</第 2 本教材>
        <第 3 本教材>轻量级 Java EE 企业应用实战</第 3 本教材>
        <第 4 本教材>疯狂 XML 讲义</第 4 本教材>
        <第 5 本教材>经典 Java EE 企业应用实战</第 5 本教材>
</疯狂 Java 实训营教材>
```

上面这份 XML 代码列出了 5 本图书的书名，该文件中的各种标记都是自定义的，开发者完全可以自定义符合约定的标记。

需要指出的是，XML 文件只是一份静态的文本文件，它可以对外提供一些信息，但不能完成任何“动态行为”。我们必须自行编写软件或程序，才能传送、接收和显示出这个文档。

XML 没有任何特别之处，它只是纯文本而已，任何有能力处理文本文件的软件都可以编辑 XML 文件。当然，如果希望程序能读懂 XML 文件，并有针对性地处理其中的标签，就需要编写处理它的应用程序了。

由此可见，XML 并不是传统意义上的编程语言，而只是一种独立于软件和硬件的信息传输工具。

现在，XML 在企业应用中的作用甚至超过了传统的 HTML，大批的软件开发商采用它作为数据交换的标准。XML 是各种应用程序之间进行数据传输的最常用工具，它在信息存储和描述领域正变得越来越流行。

➤➤1.1.3　XML 和 HTML

HTML 具有免费、简单的特点。几乎所有的浏览器都支持 HTML 标记。HTML 最初于 1990 年由

CERN 设计，它是种非常简单的 SGML，可以方便普通人的使用。因此，HTML 得到了广泛应用。但它同时具有与生俱来的缺陷：不具备可扩展性；数据逻辑与表现逻辑混杂，导致难以阅读，难以维护。

与 HTML 不同的是，XML 被设计用来传输和存储数据，而且它允许开发者自定义标记，因此功能更强大。值得指出的是，XML 并不是用来代替 HTML 的，二者是为不同目的而设计的。

表 1.1 对 XML 和 HTML 各方面存在的差别进行了对比：

表 1.1 XML 和 HTML 的对比

比较内容	HTML	XML
是否预置标签	预置大量标签	未预置任何标签
可扩展性	不具有可扩展性	是元标记语言，可用于定义新的标记语言，具有很好的可扩展性
侧重点	侧重于如何表现信息	侧重于传输和存储数据，其焦点是数据本身
语法要求	不要求标记的嵌套	严格要求嵌套、配对，并遵守 DTD 或 Schema 定义的语义约束
可读性及可维护性	难以阅读，难以维护	结构清晰，便于阅读，便于维护
数据和显示的关系	内容描述与显示混为一体，难以分离	数据逻辑与显示逻辑分离
结构描述	不支持深层的结构描述	文件结构嵌套可以复杂到任何程度
与数据库的关系	与数据库没有关系	与关系型数据库的数据表对应，可进行转换
是否区分大小写	大部分浏览器不区分大小写	严格区分大小写
编辑工具	文本编辑工具，大量所见即所得的编辑器（如 Dreamweaver 等）	文本编辑工具，大量 XML 编辑器（如 XMLSpy 等）
处理工具	任何浏览器都可	需要专门的程序进行处理

1.2 XML 的优势

XML 能够广泛应用于企业应用开发的各个领域，是由于其本身具有几大优势。简单地说，XML 具有如下 3 个优势。

➤➤1.2.1 简单易用的标记语言

从某种程度上讲，XML 比 HTML 更简单。由于 XML 不像 HTML 那样提供了大量预置标签，因此开发者甚至不需要记住什么标记，而可以使用任何标记来定义 XML 文档，只需遵守 XML 基本规则即可。

例如要定义一份我们喜欢的游戏的列表，可以定义如下 XML 文件：

程序清单：codes\01\1.2\games.xml

```
<喜欢的游戏>
    <游戏 1>
        <游戏名称>侍魂</游戏名称>
        <游戏类型>动作格斗</游戏类型>
    </游戏 1>
    <游戏 2>
        <游戏名称>雷电</游戏名称>
        <游戏类型>飞行射击</游戏类型>
    </游戏 2>
    <游戏 3>
        <游戏名称>StarCraft</游戏名称>
        <游戏类型>即时战略</游戏类型>
```