

XML Processing with Perl, Python, and PHP

XML与Perl、Python和 PHP编程指南

〔美〕 Martin C. Brown 著

邱仲潘 等译

電子工業出版社

Publishing House of Electronics Industry

北京 • BEIJING

内 容 提 要

XML即可扩展标记语言,由于它所具有的灵活性,用户可将一切存放成XML,直接访问、处理和更新信息而又不损失任何功能。本书介绍使用各种脚本语言分析、处理与使用XML。简要论述XML之后,将深入这些语言并介绍它们如何提供利用XML功能的方案,同时介绍一些重要协议(如SOAP与XML-RPC)如何实现无缝数据传输。阅读完本书后,读者就可以充分利用XML的功能,在不断增长的电子商务世界中自由施展。



Copyright©2002 SYBEX Inc., 1151 Marina Village Parkway, Alameda, CA 94501. World rights reserved. No part of this publication may be stored in a retrieval system, transmitted, or reproduced in any way, including but not limited to photocopy, photograph, magnetic or other record, without the prior agreement and written permission of the publisher.

本书英文版由美国SYBEX公司出版,SYBEX公司已将中文版独家版权授予中国电子工业出版社及北京美迪亚电子信息有限公司。未经许可,不得以任何形式和手段复制或抄袭本书内容。

版权贸易合同登记号: 01-2001-5173

图书在版编目(CIP)数据

XML与Perl、Python和PHP编程指南/(美)布朗(Brown, M. C.)著;邱仲潘等译.-北京:电子工业出版社,2002.4

书名原文:XML Processing with Perl, Python and PHP

ISBN 7-5053-7559-8

I. X… II. ①布… ②邱… III. ①可扩展语言,XML—程序设计②主页制作—应用软件,Perl③主页制作—应用软件,Python④主页制作—应用软件,PHP IV. TP393.092

中国版本图书馆CIP数据核字(2002)第020559号

责任编辑:和 敬

印 刷:北京天竺颖华印刷厂

出版发行:电子工业出版社 <http://www.phei.com.cn>

北京市海淀区万寿路173信箱 邮编:100036

北京市海淀区翠微东里甲2号 邮编:100036

经 销:各地新华书店

开 本:787×1092 1/16 印张:19.75 字数:500千字

版 次:2002年4月第1版 2002年4月第1次印刷

定 价:30.00元

凡购买电子工业出版社的图书,如有缺损问题,请向购买书店调换,若书店售缺,请与本社发行部联系。联系电话:(010) 68279077

致 谢

首先感谢Tom Cirtin在年初向我提供了出书思路。还要感谢Jennifer Campbell在后面阶段接手和管理这个项目。这是个漫长的过程，经常遇到各种困难，但Sybex人员一起完成了出色的工作。

还要感谢Gene Redding对本书进行编辑，虽然我有五年的写作经历，但仍然不尽如人意。感谢Charles Hornberger在技术上的支持，指出书中的缺漏和例子中的缺陷。

感谢建立本书所有模块、扩展和例子的幕后英雄们，包括James Clarke、Late Night Software公司人员、Apple Computer公司、Scriptics、Larry Wall人员和Perl小组、Guido van Rossum、Fredrik Lundh和Python与PyXML小组的其他人员，以及开发PHP、Ruby与Rebol的人们。

感谢在我编写本书期间处理“9.11”事件的人们。我深深地同情在此事件中受到影响的家庭。虽然我和我的亲友都没有受到“9.11”事件影响，但我认识和共处过的许多人受到了影响。特别感谢和祝愿Wendy Rinaldi、Rikke Jørgensen与Aharon Robbins。

感谢我的代理人Neil Salkind与Vicki Harding和StudioB工作室的其余人员使我的联系、协商和检查工作井然有序。

最后，感谢我的妻子。感谢她默默倾听我的许多抱怨和困惑。

前 言

大约十年前，我进入商界时，处理的是英国政府机构的数据库。我们用自由文本检索系统存储有关软件产品、教员和学员的信息。尽管我们具有这些信息的字段，但进行搜索时，搜索的是整个文档，而不只是一个或一组字段。

我的工作之一是编写程序，处理这个信息、导出字段、取出复合地址和电话号码以及整理结果，放进新版数据库中。

数据库通过Sun公司的Unix系统访问，网络上的PC机和Mac机器不能相互通信，但可以通过Telnet与Sun数据库服务器通信。你可以进行搜索、编辑信息、剪切和粘贴，但不能控制信息的外观，除非手工调整复制的文本。

十年后

十年后，我的工作依然差不多，但不在同一公司，也不处理同一信息。但是，我终究还是在处理需要管理和使用大量文本的数据库和存储系统，但使用有一定智能的结构化方式，知道要存储什么，但又足够灵活，不会限制要存储的内容。

我所处理的信息要从几种不同平台访问。仅仅在我的家庭办公室中，就有Macs、Unix、Linux至少五种不同版本的Windows和运行EPOC32与PalmOS的掌中宝。它们支持不同字符集，我要将信息转换成更有用的格式，如用HTML显示，或存储在更严格的系统中，如关系型数据库管理系统（RDBMS）。

用什么工具呢？是否还要用自由文本检索系统？

如果使用数据库系统，如何将联系人信息从桌面Mac机传输到手提Windows笔记本或Palm中？如果要联机浏览信息，能否方便地转换？如果建立的应用程序可以访问信息，如何存储选项？如何以所有需要的机器都能访问的格式在网络上提供信息？

方案

解决的方案就是使用可扩展标记语言（XML，Extensible Markup Language）。我得到需要的所有灵活性，同时又不损失任何功能。我可以在信息中增加字段、结构和布局而又不破坏任何现有工具。可以用相当标准的应用程序将XML信息变成更合适的格式。事实上，可以方便地将XML文档变成结构型数据库，可以用SQL查询数据库和将记录导出到XML格式。

也可以将一切存放成XML，直接访问、处理和更新信息。如果需要，甚至可以用XQL查询XML文档。可以用其在不同平台之间交换信息，由于所有信息都是标准的、很容易处理的格式，因此可以在能够访问的任何平台上使用这些信息。

工具

我认为，任何工作都可以选择多种工具。每个编程作业各不相同，可以用多种不同方案，用多种工具简化过程。但是，脚本语言提供了最快的开发环境，许多脚本语言具有比基于Java或C/C++的传统XML处理工具更丰富的支持平台（和更多访问方法）。

例如，Python可以在MacOS、Unix、Windows与PalmOS中运行，Rebol支持更多平台。AppleScript是8.0以上MacOS版本的标准组件，甚至放在MacOS X中。Perl是大多数Linux版本的标准，甚至有些商业化Unix安装也把Perl作为标准组件。

事实上，我能更方便地访问比任何Java或C/C++开发环境更丰富的开发工具。此外，其中有些很容易使用，为什么还要找别的语言呢？例如，MacOS中可以用AppleScript和任何应用程序通信。

想想看，只要告诉Perl把XML文档转换成Word文档进行编辑，转换成HTML在Web上浏览，转换成SQL表格在数据库中存储，能干多少工作！

本书

本书介绍用各种脚本语言分析、处理与使用XML。简要介绍XML之后，我轮流介绍这些语言，看看它们如何提供利用XML功能的方案，同时介绍一些重要协议（如SOAP与XML-RPC）如何实现无缝数据传输。本书有许多样本脚本，可以到www.sybex.com站点找到本书链接，下载这些脚本的完整版本。

XML已经无处不在，特别是在不断增长的电子商务世界中，更是红红火火。本书帮你充分利用XML的功能，希望你经常复习书中内容，巧妙利用XML。

译 者 的 话

本书翻译过程中得到了周阳生、刘文红、邹能东、彭振庆、黄志坚、李耀平、江文清等同志的大力帮助，刘文琼、邱冬金等同志完成了本书的录入工作，刘云昌、刘联昌兄弟帮助进行了书稿与打印稿的校对，在此深表感谢。

目 录

第一部分 运用XML	1
第1章 XML简介	1
专属数据格式	2
XML——使数据可移植	2
XML目标	4
XML特性	4
XML: 过去、现在与未来	4
XML与脚本语言	5
下一步工作	6
第2章 XML基础	7
XML结构	7
良构XML文档	10
实体引用	11
XML处理指令	12
XML声明	13
小结	13
第3章 文档类型定义 (DTD)	14
DTD语法	14
用DTD建立数据模型	20
何时使用DTD	20
标准DTD	21
小结	21
第4章 XML与脚本语言	22
为何使用脚本语言	22
脚本语言的问题	27
小结	28
第5章 XML与数据交换	29
分析XML	29
Unicode	33
远程数据交换	35
小结	36

第二部分 XML与Perl	39
第6章 Perl中的XML方案	39
使用XML::Parser	39
用SAX进行XML处理	48
用DOM进行XML处理	50
生成XML	53
其他XML模块	54
小结	57
第7章 Perl与Unicode	59
内核支持	59
使用Unicode数据	60
Unicode字符转换	65
小结	65
第8章 用Perl生成与分析XML文档	67
使用SAX分析器	67
使用DOM分析器	75
小结	82
第9章 用Perl转换XML文档	84
数据库管理	84
将数据库内容转储到XML	102
小结	107
第10章 在Perl中采用SOAP/XML-RPC	108
SOAP::Lite简介	108
SOAP客户端编程	111
生成SOAP服务器	114
调试SOAP::Lite	117
使用XML-RPC	119
SOAP::Lite与XML-RPC的方向	121
小结	121
第三部分 XML与Python	123
第11章 Python中的XML方案	123
xmllib模块	123
用Expat分析	127
用SAX分析	129
用DOM分析	131

小结	136
第12章 Python与Unicode	137
生成Unicode字符串	137
转换Unicode	138
访问Unicode数据库	141
小结	144
第13章 用Python生成与分析XML文档	145
用SAX分析	145
用xmlproc验证	151
小结	153
第14章 用Python转换XML文档	154
将XML转换成内部结构	155
将XML变成内部类表示	157
小结	166
第15章 在Python中采用SOAP/XML-RPC	167
使用SOAP	167
XML-RPC方案	174
小结	179
第16章 Zope与XML文档	180
XML导出格式	181
组合DTML与XML资源	184
Zope与XML-RPC	192
小结	193
 第四部分 XML与PHP	 195
第17章 XML与PHP	195
建立简单XML分析器	195
XML分析器内幕	200
将XML转换成HTML	206
小结	211
第18章 用PHP开发XML应用程序	212
RSS格式	212
建立累积器	213
编写RSS文档	222
小结	224

第19章 PHP与XML-RPC	226
编写XML-RPC客户机	227
编写XML-RPC服务器	230
XML-RPC数据转换	232
PHP中XML-RPC的好处	235
小结	236
 第五部分 XML与其他语言	237
第20章 XML与REBOL	237
在REBOL中分析XML信息	237
XML-RPC和REBOL	241
小结	242
 第21章 XML与Ruby	244
分析XML	244
Ruby与XML-RPC	250
小结	253
 第22章 XML与Tcl	254
TclXML分析器	254
用Tk浏览XML	261
使用XML-RPC	265
小结	267
 第23章 AppleScript与XML	268
用AppleScript分析XML	268
XML-RPC与AppleScript	279
XML与MacOS X	279
小结	281
 附录A Unicode速查手册	282
基本字符集	282
XML字符集名	296
 附录B 资源指南	298
一般性资源	298
XML资源	299
Perl资源	301
Python资源	301
PHP资源	302

REBOL资源	302
Ruby资源	302
Tcl资源	303
AppleScript资源	303
XML软件	304

第一部分 运用XML

第1章 XML简介

- XML目标
- 使数据可移植
- XML内幕
- 过去、现在与未来

自从计算机出现以来，一直存在数据存储和交换的问题。实际上，文件可以分成两大类：基本文本或二进制文件。

文本文件的兼容性最好，用ASCII系统和标准8位字符存储信息。ASCII系统是普遍接受的，无论是Sinclair ZX81还是PDP-11/73、Commodore 64、Atari ST、现代PC机、Mac和Unix工作站，都能读取和写入ASCII数据。当然，ASCII系统也不是没有问题，例如不同机器用不同字符表示行末符，但这些问题不是无法克服的。

但是，ASCII系统作为字母、数字和基本标点符号以外内容的存储格式时，则会遇到问题。ASCII文本的一个基本问题是，它们实际上只限于128个不同字符，包括主要字母（大小写）、数字和基本标点符号，如逗号、美元号与数学符号。

对于标准ASCII系统，只能表示这些标准字符，因此无法表达音调符和其他币值字符。我们甚至无法访问音调符，因此无法在文本中标识，使Word之类的程序能够理解其含义。

表达问题是普通文本文件的主要被抱怨的一点。顾名思义，普通文本是存储信息的无格式和无结构解决方案。虽然存在逗号分隔值（CSV，Comma Separated Values）和制表符分隔字段（TDF，Tab Delimited Fields）之类的方案，但它们只适合表格式数据，不适合任何其他内容。

假设要存储的标识文档使用黑体和斜体、不同字体、特殊字符和嵌入图形、电影与声音呢？最简单的方法是建立自己的专属二进制格式。你不受7位数据的限制，而是用8位的完全宽度字符，不是靠文本表达，而是根据需要对文档采用格式和结构。别人的应用程序读不懂你的文档也没关系，如果别人要读懂你的文档，就要买你的应用程序，不是吗？

专属数据格式

如果使用你的应用程序，这些专属格式自然不错，但如果要与别人交换文档呢？如果用E-mail传输文档，则可能要将其编码成ASCII格式，通常可以由E-mail软件自动处理，然后将它译码回二进制格式。

接收方收到文档之后，仍然需要生成文档的应用程序，至少需要能够导入或读取专属二进制格式的应用程序。这样就会有问题。世界上有许多不同的字处理程序，如果你发送的信件用Word编写，而接收方使用AppleWorks，怎么办？

可以保存成兼容格式。两个应用程序都支持多功能文本格式（RTF，Rich Text Format），这实际上是保持大多数文档格式的结构化文本格式，但它并不是万无一失的。祝贺你，你终于解决了第一个数据交换问题！

现在要对最新的数据库应用程序做这个工作。第一个问题是，没有直接对应于RTF格式的信息交换方法。显然，可以将数据导出为DIF、SYLK或前面介绍的CSV与TDF格式。我们要对数据库中的所有表格做这个工作，需要在另一端设置数据库，保存要导入的信息。

举一个具体例子，如联系人数据库，则可以说得更具体些。在系统间交换整个表格不成问题，但如果只交换单个记录，则会遇到问题。如果数据库中有三个不同表格，包含联系人姓名、地址和联系号码，则一个联系人只要从每个表中取几行记录。你要分别导入每个表格，如果记录号不相符，则更要命！

尽管在建立的两个数据库系统之间传输信息并不难，但要对两个不一致的数据库做这个工作则不那么简单，例如一个是E-mail软件中的联系人数据库，一个是掌中宝内的联系人数据库。字段名不符，字段数和类型通常也不符。

修改导出时生成的原始文本数据可以解决问题，但这个过程中可能丢失一些数据。此外，还要手工增加一些元素，这本该是自动化的，计算机应该简化我们的工作，不是吗？

XML——使数据可移植

现在我们必须顺应潮流。应用程序间的数据交换绝非易事，即使自己编写的应用程序间的数据交换也是这样。事实上，这常常是运行应用程序时最恼人的过程，也是用户论坛和技术支持中心经理最热门的话题。

数据交换无处不在，无论是最新的信用卡购物还是单击E-mail消息中的URL，都会触发某种形式的数据交换。再一步看，与朋友交换文档、在快报和产品目录导入图形、进行邮件合并、甚至在桌面电脑和掌中宝之间共享数据，都要利用数据交换。

每种情形中的关键领域是如何以最容易移植的格式建模数据，同时尽量保持数据结构。RTF、CSV、TDF和各种其他格式都在这方面跃跃欲试，问题是每种格式都只能解决某些问题，实际上都使用自己的专属格式。我们要返回到出发点。

1974年，Charles F. Goldfarb发明了标准化通用标记语言（SGML，Standardized General Markup Language）。这个系统用标准文本表示复杂文档的内容，用标签（tag）描述

文本的内容和格式，使其可以转换成原始SGML文档和从原始SGML文档取得数据，生成最后文档或取得某个元素。无论是一本书还是一个速查卡，都可以从原始SGML文档取得，都不需要修改或复制内容。

1991年，Tim Berners-Lee利用SGML提供的基本机制创建了一种方式，可以标识屏幕上显示的文本，称为超文本标记语言（HTML，Hypertext Markup Language）。尽管Internet在当时已经不是新东西，但HTML使Internet的用法焕然一新，使浏览与交换信息的方法焕然一新。

大约1997年，人们发现，许多适用于SGML与HTML的原则也同样适用于构造数据模型。如果用SGML声明文档元素，使我们可以选取各个段落、章节和其他特定段，而HTML定义文本格式，则可以用可扩展标记语言（XML，Extensible Markup Language）按结构化格式存储数据。

XML可以标识文本文档，以便识别信息的不同部分。例如，可以标识联系人记录如下：

```
<contact>
  <name>Martin Brown</name>
  <address>The House</address>
  <town>Sometown</town>
  <postcode>AB12 3CD</postcode>
  <contact_numbers>
    <phone>01234 567890</phone>
    <mobile>09876 543210</mobile>
    <email>mc@mcs1p.com</email>
  </contact_numbers>
</contact>
```

现在就可以从这个文档中选取联系人姓名、地址和电话清单。

整个文档都在文本中，因此我们不必考虑对专属格式处理和编程阅读程序。字段很容易标识，我们不必怀疑哪个字段是我的电话号码，整个记录可以来自数据库中的多个不同表格。而且这个转换中没有丢失任何信息。

回到原先的问题：在应用程序之间交换数据时，可以看到，我们已经解决了所有问题，不管是二进制格式还是CSV、RTF或其他文本和数据特定格式。

利用我们的联系人文档，如果从桌面上的E-mail应用程序导出，然后传输到掌中宝中，则可以方便有效地复制信息，不必进行任何手工干预。如果掌中宝无法对付移动电话号码，则可以忽略这个字段。如果它用单个字段保存地址、市镇和邮政编码信息，则可以在记录时将这些信息组合到一起。

这就是XML：以结构化方式构造数据模型，使我们能够方便地在应用程序之间交换数据。XML方案使数据变得可移植。

XML目标

可扩展标记语言（XML，Extensible Markup Language）是SGML的旁集。尽管它符合SGML系统的大多数基本原则，但删除了其中一些复杂细节，更容易在Web上显示和格式化信息。万维网联盟（W3C）最初开发XML时的设计目标如下：

1. XML应可在Internet中直接使用。
2. XML应支持各种应用程序。
3. XML应与SGML兼容。
4. 易于编写分析XML文档的程序。
5. XML标准的可选特性数应尽量减少（最好为0）。
6. XML文档应易读易懂。
7. XML设计应迅速准备。
8. XML设计应正式和简洁。
9. XML文档应易于生成。
10. XML标志的简洁与否并不重要。

总的来说，W3C取得了成功。XML易于使用、生成、分析和理解，即使原始格式也很容易阅读。XML 1.0规范已经确定，正式发布于1998年2月10日。

XML特性

XML特性主要可以用六句话描述：

- XML可以根据需要调整存储和组织信息的方法，而不是由生成信息的应用程序控制。
- XML用Unicode字符集，因此不限于ASCII或任何语言的任何字符集。XML文档可以写成英语、中文、（印度）古吉拉特语、希腊语或梵语。
- XML是开放标准，即不属于某个人或组织，不依赖于一家公司，不属于或依赖于一个应用程序的特性。
- XML文档可以开放也可以严格，可以通过检查语法、数据内容或文档结构而检查文档质量。
- XML清晰易读，人们可以阅读和编写XML文档，可以用标准文本编辑器修改与编写XML文档。
- XML是构造数据模型的系统。可以不用将数据变成另一种格式而用样式单将数据转换成格式化文档。

XML：过去、现在与未来

XML尽管已经有一定时间，但仍然是一门较新的技术。到本书编写时（2001年7月），XML已经有3年半的历史，但XML理想中的许多特性、应用和承诺还没有实现。

这是无可指责的。**HTML**已经有十个年头，但至今仍然刚刚开始渐入佳境。大多数人每天使用**HTML**，有些人每天编写**HTML**，但如何更好地利用**HTML**仍然是个问题。兼容性问题（不同浏览器用不同方式显示相同**HTML**）、标志和**HTML**在整个文档格式机制中的地位仍然是悬而未决的问题。

XML实际上是一组技术。**XML**标准本身定义如何在**XML**文档中指定元素及其属性，幕后的文档类型定义（**DTD**, **Document Type Definition**）是个可选元素，定义**XML**文档中可以加入的字段与数据结构、布局和验证。

XML标准的扩展可以定义和指定**XML**文档中的其他元素，如**XLink**增加超链接，**XPointer**与**XFragments**指向和引用**XML**文档区。要将**XML**文档转换成**HTML**在**Web**页面上显示，可以用级联样式单（**CSS**, **Cascading Style Sheets**）、可扩展样式单语言（**XSL**, **Extensible Stylesheet Language**）和**XSL**转换（**XSLT**, **XSL Transformations**）。

最后，还有读取**XML**文档的技术，如**XML**简单API（**SAX**, **Simple API for XML**）和文档对象模型（**DOM**, **Document Object Model**）。还有使用**XML**的技术，如资源描述格式（**RDF**, **Resource Description Format**）建立元数据模型，**RDF**站点汇总（**RSS**, **RDF Site Summary**）将新闻信息流化成结构化格式，而**XML-RPC**与**SOAP**之类的远程技术用**XML**与远程服务器交换请求和响应，可以远程执行功能。

目前，**XML**仍然处于“演示式”阶段，正在讨论和确定标准，尽管许多方面大有进展，但要真正进入尽善尽美，还有待时日。

将来，**XML**将成为计算经历的主要部分。你是否意识到这点，就看它的广告做得怎样。

有些公司正在建立工作组，确定系统间通信的标准。已经有一些工作组在解决上面介绍的联系人数据库和桌面/掌中宝数据问题。还有一些公司在开发电子数据交换（**EDI**, **Electronic Data Interchange**）解决方案，这个系统需要定义成百上千个字段，保存订单信息。

早晨起来，你会通过一组**RSS**阅读新闻，向朋友发送**XML**格式**E-mail**，与银行交换**XML**文档以查验最新结余，用**XML**文档下订单并向供应商和客户索要发票，而不是通过打印、传真或邮寄。

XML与脚本语言

前面介绍了**XML**的内容、来源及其要解决的问题，这与脚本语言有什么关系呢？显然，如果要读取或写入**XML**文档，则需要用编写应用程序的语言读取或写入。

许多人在快速应用程序开发（**RAD**, **rapid application development**）环境中使用**XML**文档，许多人使用**Perl**、**Python**和**Tcl**之类的脚本语言，而有些人则在**Web**和**Internet**应用程序中使用**XML**。

大多数情况下，即使在编写非**XML**脚本时，脚本语言也比**C/C++**有一些优点。脚本语言更快写成，更易使用，通常具有更好的文本处理特性。例如，本书介绍的许多脚本语言支持更好的数据输入系统、灵活的数组处理功能和复杂的结构，是组合字符串、数组和散列/字典而建成的。

既然XML的目的是处理文本信息和将文本信息构造成为更有用的格式，为什么不用脚本语言呢？

下一步工作

本书的目的是介绍如何用Perl、Python、PHP、REBOL、Ruby、Tcl和AppleScript通过XML读取、写入、格式化和构造信息，而不是想比较哪个语言更好，但我们主要介绍最重要的三种脚本语言。

本书是用每种脚本语言进行各种XML处理与操纵任务的实用指南。因此，你不必按顺序阅读，而可以转入所要的章节：

- 如果不熟悉XML标准或如何编写与使用XML，则可以看看第2章和第3章，其中分别介绍编写XML文档和DTD的基本知识。
- 不管喜欢哪种脚本语言，第4章是重要的背景知识，指出脚本语言比C/C++有哪些优点。
- 第5章介绍编写XML文档时使用的不同技术及如何在各种脚本语言中使用这些特性。具体地说，我们介绍Unicode及其对XML文档的影响，如何用XML建立用任何语言编写的应用程序之间的桥梁。
- 如果你是Perl编程人员，已经知道XML知识，则可以直接转入第6章。
- 如果你是Python编程人员，已经知道XML知识，则可以直接转入第7章。
- 如果你是PHP编程人员，已经知道XML知识，则可以直接转入第17章。
- 如果你是REBOL编程人员，已经知道XML知识，则可以直接转入第20章。
- 如果你是Ruby编程人员，已经知道XML知识，则可以直接转入第21章。
- 如果你是Tcl编程人员，已经知道XML知识，则可以直接转入第22章。
- 如果你是AppleScript编程人员，已经知道XML知识，则可以直接转入第23章。