·自学经典·

XML 实用技术

《自学经典》

伍 逸 编著

- 内容细致，知识全面，适合初、中级读者学习使用
- 案例丰富，所有技术要点均采用示例程序的方式讲解
- 实例代码中注释详细，方便读者理解代码的具体含义
- 由浅入深，循序渐进，强调理论和实践的结合

清华大学出版社

自学经典

XML 实用技术自学经典

石 磊 编著

清华大学出版社
北 京

内 容 简 介

本书主要讲述 XML 及其相关技术，全书共 10 章，分别介绍 XML 基础语法、XML 命名空间、文档类型定义、利用应用文档对象模型操作 XML 文档、JavaScript 语言、应用 XPath 操作 XML 文档、利用 CSS 和 XSLT 转换 XML、可缩放矢量图形相关知识、C#的基础知识和语法及在 C#中应用文档对象模型读写 XML 文档。

本书适合于对 XML 感兴趣，想更深入学习 XML 及其相关技术的读者。

图书在版编目（CIP）数据

XML 实用技术自学经典 / 伍逸编著. —北京：清华大学出版社，2016
（自学经典）
ISBN 978-7-302-41253-3

Ⅰ. ①X…　Ⅱ. ①伍…　Ⅲ. ①可扩充语言－程序设计　Ⅳ. ①TP312

中国版本图书馆 CIP 数据核字（2015）第 186471 号

责任编辑： 袁金敏
封面设计： 刘新新
责任校对： 胡伟民
责任印制： 李红英

出版发行： 清华大学出版社
网　　址：http://www.tup.com.cn, http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者： 清华大学印刷厂
装 订 者： 北京市密云县京文制本装订厂
经　　销： 全国新华书店
开　　本： 185mm×260mm　　**印　　张：** 19.5　　**字　　数：** 490 千字
版　　次： 2016 年 1 月第 1 版　　**印　　次：** 2016 年 1 月第 1 次印刷
印　　数： 1～3000
定　　价： 49.00 元

产品编号：066012-01

前　言

由于在实际工作中需要大量用到 XML 及其相关技术，比如用于存储、传输数据，格式化显示 XML 数据等，因而笔者对 XML 的应用还算有所心得，于是产生了写一本关于 XML 技术方面书籍的想法，经过一段时间的努力，终于完稿。希望本书能够起到抛砖引玉的作用，引领读者加快学习 XML 技术的步伐。

本书内容

本书以实例为中心，全面介绍了 XML 应用的方法和技术。全书共 10 章，这 10 章既相互关联又各独自为篇。第 1～4 章分别介绍了 XML 的格式和语法、XML 命名空间、文档类型定义，以及 XML 模式。第 5 章介绍了利用应用文档对象模型操作 XML 文档和 JavaScript 语言。第 6 章讲解了应用 XPath 操作 XML 文档。第 7 章讲述了 CSS 和 XSLT 转换 XML 的方法。第 8 章介绍了可缩放矢量图形的相关知识。第 9 章讲述了 C# 的基础知识和语法。第 10 章介绍了在 C# 中应用文档对象模型、XmlReader 和 XmlWriter 读写 XML 文档的知识。

本书以实例来讲述各种与 XML 相关的知识和技术，对涉及的各种技术，如 CSS、JavaScript、C# 的类及函数给出了详细的解释。具体特点如下：

- 涵盖了 XML 技术的各个方面，既适合初学者，也适合于具备一定 XML 知识的读者。
- 所有的技术要点均采用示例程序的方式加以讲解，避免了枯燥的理论解释。
- 每一章都有相应的练习习题，使读者能更进一步地掌握学习的知识点。
- 在较为复杂的 C# 综合示例程序中，展示了在 C# 编程中比较常用但又不易掌握的技术难点，如自定义控件、控件间的互动等，并在书中详细地加以解释。

读者对象

本书适合两类读者阅读。一类是从未接触过 XML，希望通过阅读相关书籍掌握 XML 这门技术的读者。建议这类读者按照目录安排，循序渐进地阅读本书。另一类是具备一定的 XML 技术基础，希望实现技术升级和掌握新技术的读者。如果这类读者对于 XML 技术具有兴趣，同时又具有一定的编程经验，或想更深入地了解，建议先粗略地阅读前 3 章，然后将时间和精力放在书中感兴趣的部分。

由于时间仓促，加之作者水平有限，书中肯定会有不少缺点和疏漏，敬请读者批评指正，编者会在适当的时间再作修订补充，以跟上 XML 技术的发展。

编　者

目　录

第1章 XML简介

XML是Extensible Markup Language（可扩展置标语言）的缩写。XML是一种类似于HTML的标记语言，用于组织、存储和发送数据信息。XML没有固定的标记，它允许用户定义数量不限的标记来描述文档中的资料。XML作为一种数据传输方式在计算机信息领域中得到了广泛的支持。本章主要内容：

- 标记语言的发展历史
- XML的特性
- XML的语法规则
- XML相关技术的简介，如DTD、XSLT、XPath等

1.1 标记语言的发展简史

XML同HTML一样都来自SGML（Standard Generalized Markup Language，即标准通用标记语言）。SGML包含了一系列的文档类型定义（简称DTD，DTD中定义了标记的含义），SGML的语法是可以扩展的。SGML的内容十分庞大，既不易学也不易用，实现起来也非常困难。鉴于这些原因，Web的发明者——欧洲核子物理研究中心的研究人员，根据当时（1989年）的计算机技术，开发了HTML。HTML抛弃了SGML复杂、庞大的缺点，继承了SGML很多优点。HTML最大的特点是简单和跨平台。HTML是一种界面技术，它只使用了SGML中很少的一部分标记，例如HTML 4.0中只定义了70余种标记。为了更容易地实现，HTML规定的标记是固定的，即HTML语法是不可扩展的。HTML这种固定的语法使它易学易用，在计算机上为HTML开发浏览器也十分容易。正是由于HTML的简单，使得基于HTML的Web应用得到了极大的发展。

然而近年来，随着Web应用的不断发展，HTML的局限性也越来越明显地体现了出来，如HTML无法描述数据、可读性差、搜索时间长等，人们又把目光转向了SGML。但是庞大的SGML学、用复杂，于是人们自然会想到仅使用SGML的子集，以使新的语言既方便使用又容易实现。正是在这种形势下，Web标准化组织W3C建议使用一种精简的SGML版本——XML应运而生了。

XML最初的设计目的是为了实现EDI（Electronic Data Interchange，电子数据交换），确切地说是为电子数据交换提供一个统一的标准数据格式。

在EDI应用过程中，XML展现了如下的优势。

- 低成本。XML不需要支付VAN（Value-Added Network，增值网络）的高额费用，中小企业也负担得起。
- XML允许用户创建自己的商业规则和格式。

- 容易解释。XML 通过免费下载的解析器就可以很容易地解释。
- 平台独立。XML 是跨平台的语言，不管是什么平台，都能进行数据交换。用户可开发各种各样的 XML 扩展，比如数学标记语言 MathML、化学标记语言 CML 等。

此外，一些著名的 IT 公司，如 Oracle、IBM 以及微软等都积极地投入人力与财力来研发 XML 相关的软件与服务支持，这无疑确立了 XML 在 IT 产业中的重要地位。

1.2 什么是 XML

XML 是一种用于描述数据的标记语言，它不提供固定的标记，而是允许用户自定义数量不限的标记来描述数据，且允许使用嵌套的信息结构。不同于 HTML，XML 的重点在于表示数据，它提供了一个直接处理数据的通用方法。HTML 着重描述数据的显示格式，而 XML 着重描述的是数据内容。XML 文档以.xml 为后缀。编写 XML 文档不需要特别的软件，只要有一个文本编辑器就可以，比如“记事本”程序。先看一个简单的 XML 文档。

```
<?xml version="1.0" encoding="UTF-8"?>
  <books ISBN ="9787544238212">
    <title>The Book Thief</title>
    <price>25</price>
    <quantity>10</quantity>
  </books>
```

XML 文档的第 1 行是 XML 声明，定义了 XML 的版本和使用的字符编码。在这个例子中，代码的第 1 行 XML 声明定义了 XML 的版本（目前发布的是 1.0 版本），使用的字符编码是 UTF-8 字符集。代码的第 2 行定义了文档的根元素<books>，是 XML 文档必须声明的元素。代码的第 3～5 行定义了根元素的子元素（在这里有 3 个子元素<title>、<price>和<quantity>）。最后一行的代码则定义了根元素的结束。

每个 XML 元素都以一个起始标记（opening tag）“<”开始 ，以一个结束标记（closing tag）“</”收尾，比如 <title> 就是一个起始标记，</title> 就是一个结束标记。XML 元素可以带有属性，属性值要加引号，比如例子中的 ISBN 就是<books>的属性，属性值为“9787544238212”。XML 的标记（tag）是可以自定义的，用来描述数据，比如例子中的<title>标记表示这个元素内的数据是书名，The Book Thief 就是一个具体书名。 用户可以修改标记，比如写成下面的形式。

```
<booktitle>The Book Thief</booktitle>
```

由于 XML 的标记可以自定义，因而可以用 XML 语句来描述和存储各种内容的数据，比如有关电影或者家具的数据。也就是说，各种内容的数据，都可以通过 XML 描述和存储起来。从结构上说，XML 文档是一棵节点树。一个 XML 文档只有一个根节点，但可以包括数量不限的子节点。

根据上面的例子，可以对 XML 总结如下。

- XML 是一种可扩展的标记语言。

- XML 没有固定的标记，用户可以自行定义标记来描述数据。
- XML 主要用来描述和存储数据。
- XML 具有自我描述性。
- XML 是树状结构的文档，是个结构化的文档。
- XML 文档使用的是文本格式。

1.3　使用 XML 的好处

使用 XML 具有如下好处。

- 易于携带和传输。
- XML 文档不依赖于特殊的软件，只要有文本编辑器，就可以编写 XML 文档，而且保存格式也是文本格式。一个 XML 文档就是一个小小的文本文件，易于携带和传输。
- 易于共享和跨平台。
- XML 文档本身是个文本文件，而且采用结构化的数据，很容易被各系统读取。
- 易于查询。
- 因为 XML 的结构是树状结构，因此易于查询。
- 易于展示。

可使用任何文本软件或任何浏览器打开并查看 XML 文档内容。

1.4　用浏览器浏览 XML 文档

可使用任何浏览器来浏览 XML 文档的内容。

假设用文本编辑软件如 Notepad 创建了一个文本文件，将 1.2 节中的 XML 文档内容粘贴到文本文件中，并将该文本文件存储为 book.xml。则要在浏览器中浏览 XML 文档，只需要在浏览器中打开 book.xml 文件，即可看到如图 1-1 所示的结果。

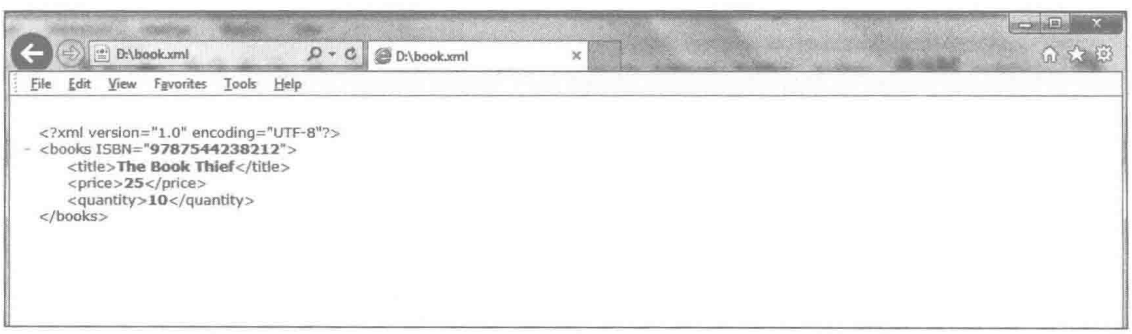

图 1-1　浏览 XML 文档

1.5 XML 语法

编写 XML 文档必须遵循一些语法规则，但在介绍这些规则前，首先要了解标记、元素和属性这 3 个术语。

1.5.1 XML 的标记、元素和属性

XML 最主要的术语有标记、元素和属性 3 个。

1. 标记（起始标记，结束标记）

在上面的 XML 文档中，已经看到有这样一些特征相同的字符串：<title>、<price>、<quantity>、</books> 等。它们都是由小于号“<”开始，由大于号“>”结束，在 XML 规范里，将其称为 XML 标记。标记又有起始标记和结束标记之分。起始标记由“<”开始，由“>”结束。比如<title>、<price>、<quantity>。结束标记由“</”开始，由“>”结束。比如</title>、</price>、</quantity>。

2. 元素

XML 元素是指从一个起始标记到它的结束标记间的一段内容。比如<title>The book thief</title> 就是一个元素。元素是 XML 文档的基本单位，一个 XML 文档可以由一个或者多个元素构成。XML 元素可以扩展，以携带更多的信息。假设有如下的 XML 文档。

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
  <to>Frank</to>
  <from>John</from>
  <body>Don't forget the meeting!</body>
</note>
```

假设已经创建了一个应用程序，可将 <to>、<from> 以及 <body> 元素从文档中提取出来，并输出以下信息。

```
MESSAGE
To: Frank
From: John
Don't forget the meeting!
```

如果这个 XML 文档作者又向这个文档中添加了一些信息，此时的文档内容如下。

```
<note>
 <date>2008-08-08</date>
 <to>Frank</to>
 <from>John</from>
 <heading>Reminder</heading>
 <body>Don't forget the meeting!</body>
 <phone>12345678</phone>
</note>
```

那么这个应用程序会中断或崩溃吗？不会。这个应用程序仍然可以找到 XML 文档中的 <to>、<from> 以及 <body> 元素，并产生同样的输出。XML 的优势之一，就是可以经常在不中断应用程序的情况进行扩展。

3. 属性 (attribute)

一个元素可以带有属性，属性写在起始标签里，位于元素名称的后面。比如：

```
<books ISBN ="9787544238212">
```

其中 ISBN ="9787544238212" 就是 books 元素的一个属性。其中 ISBN 是属性的名称，9787544238212 是属性的值，在语句中属性值必须加引号。

1.5.2　XML 的语法规则

XML 主要的语法规则如下。

1. 每个起始标签必须有对应的结束标记

例如前面的例子中，起始标记<price>必须有相应的结束标记</price>。

2. 一个 XML 文档只能有一个根元素

XML 文档是树状结构的，像一棵节点树。比如上面例子中，<books>就是根元素，而<title>、<price>、<quantity>则是<books>元素的子节点。

如果一个文档中有两个<books>根元素，就会出错，比如下面的文档运行时就会出错。

```
<?xml version="1.0" encoding="UTF-8"?>
<books ISBN ="9787544238212">
 <title>偷书贼</title>
 <price>25</price>
 <quantity>10</quantity>
</books>
<books ISBN ="978758225">
 <title>香水</title>
 <price>100</price>
 <quantity>12</quantity>
</books>
```

3. 所有的XML元素必须正确嵌套

正确的嵌套如下。

```
<books><title>香水</title></books>
```

错误的嵌套如下。

```
<books><title>香水</books></title>
```

4. 标记区分大小写

XML 标记是区分大小写的，起始标记与相应的结束标记的大小写必须一致。

下面的两行代码中，第 1 行是错误的，第 2 行是正确的。

```
<title>我的故事</Title>
<title>我的故事</title>
```

XML 元素是 XML 文档的基本单位。一个 XML 文档由一个或者多个 XML 元素构成。比如<site>woyouxian.net</site> 就是一个 XML 元素，也可以是一个最简单的 XML 文档。一个 XML 元素从一个起始标记开始，到对应的结束标记结束。起始标记和结束标记之间的内容，称为 XML 元素的内容，比如，woyouxian.net 就是元素<site>woyouxian.net</site>的内容。

如果一个 XML 元素没有内容，比如，<site></site>则称其为空元素。空元素有一种特殊的写法，即以“<”开始，然后是元素名称，在以“/>”结束，比如<site /> 。

XML 语法中标记是区分大小写的。比如<SITE>和<site>就表示两个不同的元素。这一点在编写 XML 文档时要特别注意。

5. XML元素间有父子、同级关系

XML 文档是树状结构的，它只有一个根元素，其他元素都是根元素的后代。比如下面的例子：

```
<?xml version="1.0" encoding="UTF-8"?>
<father>Tom Smith
 <son>John Smith
  <grandson>Hans Smith</grandson>
 </son>
 <daughter>Jane Smith</daughter>
</father>
```

根元素是<father>，<father>下面有 2 个子节点，即 son 和 daughter 元素，而 son 元素下面又有一个子节点，即<grandson>元素。

XML 元素之间的关系，主要有：

- 子节点(child)
- 父节点(parent)
- 并列节点又称兄弟姐妹关系(sibling)

以上面的例子来解释这些关系。<son>元素和<daughter>元素是<father>元素的子节点。<father>元素是<son>元素和<daughter>元素的父节点。<daughter>元素和<son>元素的关系是并列节点关系。

6. 属性的值必须加引号

XML 元素可以带有属性。作为 XML 元素的附加信息，属性写在起始标记里，位于元素名称的后面。比如 <book ISBN ="9787544238212"> 就是一个带有属性的 XML 元素。XML 属性是以名称和值的形式配对出现的，XML 属性名称是区分大小写的，比如 Name

和 name 就表示两个不同的属性。XML 属性的值应用引号围起来，可以用双引号，也可以用单引号。以下两种写法都是正确的，不过通常来说，多采用双引号。

```
<book ISBN ="9787544238212">
<book ISBN ='9787544238212'>
```

如果属性的值里包含双引号，就用单引号包围属性值，比如下面的例子。

```
<site info ='wo"you"xian.net'>
```

如果属性值里包含单引号，就用双引号包含属性值，比如下面的例子。

```
<site info ="wo'you'xian.net">
```

一个 XML 元素可以有一个或者多个属性，每个属性都以空格分开。比如下面的例子。

```
<site name="woyouxian.net" author="me">
```

7. 一个XML元素不能有相同的属性

下面的写法是错误的，因为一个 XML 元素不能有两个相同的属性名称，即使属性值不同也不允许。

```
<books ISBN ="9787544238212" ISBN ="97875442dr">
```

不过，如果将其中一个 ISBN 改为小写就是对的，比如下面的写法就是正确的。

```
<books ISBN ="9787544238212" isbn ="97875442dr">
```

这是因为 XML 是区分大小写的，ISBN 和 isbn 表示两个不同的属性。

关于属性的应用，还需要详细说明一下。

在描述数据时应该使用 XML 元素还是属性呢？没有硬性规定说哪些数据应该使用元素，哪些数据应该使用属性，比如以下这两种写法都是对的。

第一种写法，使用属性：

```
<site name="woyouxian.net">
```

第二种写法，使用元素：

```
<site>
 <name>woyouxian.net</name>
</site>
```

通常来说，表达描述元数据（metadata）时，应使用属性，而描述数据本身时应当使用元素。元数据意为描述数据的数据。比如一篇文章，文章关键词就是元数据，而文章的内容就是数据本身，示例如下。

```
<article keywords="XML,属性" >
```

<content>XML 可以带有属性，作为 XML 元素的附加信息。XML 属性是以名称和值的形式配对出现的。XML 属性应写在起始标签里面，位于起始标记的名称之后。

```
</content>
</article>
```

另外，ID 索引大都使用属性，比如：

```
<employee ID="6699">
```

在 XML 中，使用属性值通常能够简化文档的书写，但不要太频繁地应用属性值，毕竟 XML 是用来储存和传送数据信息的，它的可扩展性更为重要，这是因为，用户可能随时需要向 XML 文件中添加数据，虽然使用属性值可以方便地为元素添加额外的信息说明，但是这样做非常不利于日后的维护和更新。对比下面的两个例子。

示例 1：使用属性值

```
<?xml version="1.0"encoding="UTF-8"?>
<Me Name="jsper"Gender="male" Job="No" Email="jsper@371.net">
</Me>
```

示例 2：不使用属性值

```
<?xml version="1.0" encoding="UTF-8"?>
<Me>
 <Name>jsper</Name>
 <Gender>male</Gender>
 <Job>No</Job>
 <Email>jsper@371.net</Email>
</Me>
```

显然，示例 2 比示例 1 更易于扩展、维护和更新。此外，使用属性值还有以下一些缺点：

- 属性值不可以包含多重数值。
- 属性值难于扩展。
- 属性值不能够用于描述结构内容（子元素则可以）。
- 属性值很难通过 DTD 来进行测试。

在 XML 规范中，空白包括空格、制表符和空行。在编辑 XML 文档时，常常使用空白来分隔标记，以获得较好的可读性。XML 的空白字符有两类:有效空白字符和无效空白字符。有效空白字符是文档的一部分，应当保留。有效空白字符是指，在编辑 XML 文档时，为了提高可读性而添加的字符。在 XML 文档中，可以在元素中使用一个特殊的属性 xml:space，来通知应用程序保留此元素中的空白。在有效的文档中，这个属性和其他任何属性一样，在使用时必须声明。xml:space 属性必须被声明为 Enumerated（枚举）类型，它的值必须是“default”和“preserve”两者之一，也可两个都取。

“default”允许应用程序根据需要处理空白。如果不包含 xml:space 属性，结果与使用“default” 值相同。“preserve”指示应用程序按原样保留空白，暗示空白可能有含义。xml:space 属性的值应用于包含该属性的元素的所有子代，但由一个子元素覆盖时除外。

例如，下列文档指定的是相同的空白行为。

```
<?xml version="1.0" encoding="UTF-8"?>
<poem xml:space="default">
  <author>
    <givenName>Alix</givenName>
    <familyName>Krakowski</familyName>
```