125

TP312XS
516

Internet 新技术丛书

XSL 技术实践
Just XSL

（美） John E. Simpson 著

彭仕安 郭漫雪 周林明 等译

机械工业出版社
China Machine Press

本书介绍了 XSL 技术，包括 XSLT 和 XSL-FO 以及其他 W3C（World Wide Web Consortium）相关标准。本书代码丰富，实用性强，用实例讲解了问题的解决方案，还介绍了一些流行的用于处理 XSLT 和 XSL-FO 的工具软件及其工作原理。

Simplified Chinese edition copyright © 2002 by PEARSON EDUCATION NORTH ASIA LIMITED and China Machine Press.

Original English language title: Just XSL, 1E, by John E. Simpson, Copyright©2002. All Rights Reserved.

Published by arrangement with the original publisher, Pearson Education, Inc., publishing as PH PTR.

This edition is authorized for sale only in the People's Republic of China (excluding the Special Administrative Region of Hong Kong and Macau).

本书封面贴有 Pearson Education 培生教育出版集团激光防伪标签，无标签者不得销售。
版权所有，侵权必究。

本书版权登记号：图字：

图书在版编目（CIP）数据

XSL 技术实践——Just XSL/(美)森普逊(Simpson,J.E.)著;彭仕安等译．－北京：机械工业出版社，2002.7
(Internet 新技术丛书)
书名原文：Just XSL
ISBN 7-111-10452-8

Ⅰ.X… Ⅱ.①森…②彭… Ⅲ.可扩充语言，XSL－程序设计 Ⅳ.TP312

中国版本图书馆 CIP 数据核字（2002）第 040726 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：宋燕红 张鸿斌
北京市密云县印刷厂印刷·新华书店北京发行所发行
2002 年 7 月第 1 版第 1 次印刷
787mm×1092mm 1/16·27 印张
印数：0 001-4 000 册
定价：48.00 元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

前　言

如果你正在所喜欢的当地书店或网上电子书店的书架前面徘徊，寻找一本关于可扩展样式单语言（Extensible Stylesheet Language，XSL）的书。那么你必须了解以下内容：

首先，你应该了解可扩展标记语言（Extensible Markup Language，XML）。知道它是由标准化的通用标记语言（Standardized General Markup Language，SGML）派生出来的，尽管它的名字表明它是一种标记语言，但它又不仅仅是一种简单的标记语言，而是一种创建和管理标记的语言。XML 文档由文字组成，除此之外，没有其他的内容。XML 与 HTML 有些相似，它们都使用尖括号、& 符号等等，但它却比 HTML 更为强大。在 XML 中，定义元素的标记必须成对匹配（起始标记/结束标记对）。或许你有时候会感到迷惑，DTD 表示的是“Document Type Definition”（文档类型定义）还是“Document Type Declaration”（文档类型声明），但你很清楚 DTD 的作用。诸如此类的东西都是你必须了解的。

其次，即使你对 XML 有一定的了解，你还是会有些迷惑。它显得那么的不顺手。当需要在文档和数据中使用 XML 标记时，应该怎么办呢？你已经听说过 XSL（你想买一本这方面的书，并在寻找它，对吗?），而且可能你会认为在 XSL 中可以找到如何使用 XML 这个问题的答案。

再次，也是最重要的一点，你希望找到一本关于 XSL 的书，它能够帮助你理解其中的概念，而又不是简单地将标准中的有关概念再解释一遍（无论它解释得多么清楚）。你肯定希望能够反复地参考这本书中所学过的主题。在你更新知识之前，需要将这些内容牢记在心。

如果前面三段的描述符合你的情况，我认为本书很适合你。以下是我在本书中将要介绍的内容，以及计划不予介绍的内容。

本书要介绍的内容

第 1 章为后面所有的章节做铺垫，介绍一些基本概念。如果你已经对 XSL 有些了解，则可以跳过这章中的大部分内容。

第 1 章显得有些独立，在它之后的内容分为两大部分，第一部分介绍 XSL 转换（XSLT），第二部分介绍 XSL 格式对象（XSL-FO）。在这两个主要部分中，分别有一个很详细的关于 W3C（World Wide Web Consortium）相关标准的解释。如果标准中的内容模棱两可、容易搞混或是显得非常的奇怪，我都会明确地将它们指出来。但是，要理解这些内容，你需要一些耐性以及一定的引导。我将通过解释、例子和比喻的方式引导你。（至于耐性，那就看你自己的了。）

除了大量的例子片段之外，本书这两部分中的各章都包括一个解决问题的例子。某些章中

的问题可能极为简单，很容易回答（虽然答案可能不太明显）。而有些章中的问题则极为复杂，要用好几页的 XSLT 或 XSL-FO 代码才能很好地解决。

对 XSLT 的有效使用取决于你对 XPath 表达式的熟悉程度。可以很放心地假设，当你选用本书时，你已经对 XML 有了一定的了解，但却不能大胆地假设你已经对 XPath 有了一定的了解。因此 XSLT 部分的第 1 章中还将详细地介绍 XPath。

最后，我还将演示一些流行的用于处理 XSLT 和 XSL-FO 的工具软件，并解释它们的原理。

本书没有介绍的内容

按巴特·辛普生（与本书无关）的开场方式，将本书中不介绍的内容列在下面：

我不会详细讲述 XML。我认为你已经了解它了。

我不会教你 XLink、XPointer、XML 方案以及其他以带 X 缩写的标准（显著，XSLT、XPath 以及 XSL-FO 除外）。在某些地方会引用这些其他的标准，如果你受到其他方面的影响，感到迷惑，我会给出一些提示，帮助你走出这种困境，但我不会在它们上面花太多的时间。

我将不会花太多的时间和精力来向大家演示如何安装、运行及使用这些我所使用的工具软件（可以肯定，介绍这些信息的这种技术书很容易就过时）。另外，你很快就会发现，这些工具一般都各有其独特的东西，我会在遇到它们时给出一些提示。

最后，在本书中，几乎看不到关于“如何创建一个 Java Servlet/编写一个 CGI 程序/编写一个 ASP 页面来处理 XSL”。在关于服务端 XSL 处理一章的某些简单例子中，可以找到一些这方面的信息；但即使有这种信息，它们也只是关于如何使用的现有的服务端工具，而不是关于如何建立自己的工具。

关于作者的一些信息

我作为应用开发人员已经 20 多年了，编写技术书籍也快有 10 年了。但我并不是从大学毕业后马上就进入计算机世界的；我教过高中（教授英语和新闻），还从事过其他的一些短期工作。

所有的这些都是要使大家相信（也可能，很可能只是为了使我自己相信）：当使用和思考技术时，我倾向于不是出于技术本身的原故。生命实在短暂，一定要热情地投入到这个电子潮流中去，力争很好地控制并操纵它。

因此，我还有一个希望：我希望大家学习本书的时候，不要对我们感到厌烦。可以肯定，我想向你演示如何使用 XSL。但我一直在想，我们需要更多的例子来演示应用严肃技术中的乐趣，而不是如何使用它们的例子。

如果你已经看过了我的上一本书《Just XML》的话，就不会奇怪我的工作方式。在那本书中，我向大家介绍了能让大家感到奇怪的 XML 应用：使用 FlixML（一个 XML 词汇表）来

构建老式的、投资低廉的 B 片影评[⊖]。本书中还要用到 FlixML，不过已经做了一些小修改，这样可以更好地演示 XSLT 和 XSL-FO 的特征。你会发现很少的例子能帮助你使用 XSL 来创建公司的年度报表；但请相信我，如果你能使用它来生成一个 B 片电影节的节目单，或者你至少能明白我是怎样创建的，则如何创建一个公司年度报表就是小事一桩了。

我还将偶尔偏离主题，进入无关的 B 片领域。这四个所谓的“B 片”框将向大家介绍四部 B 片，我认为这四部片子很值得一看。我保证这些离题都只是短暂的。

如果你想了解 FlixML（或者 B 片）的更多信息，可以访问我的网站www.flixml.org 。在那里可以找到 FlixML 影评例子，各种不同版本的 DTD，以及本书的勘误表（请上帝原谅，如果有错误的话），有用的 Web 站点，等等。

⊖ 《Just XML》一书的大部分读者似乎很喜欢 FlixML——它能或多或少地全面演示基本原理。我感到很有趣，很多人将 FlixML 的失败归咎于对 XML 使用得过于刻板。

此为试读，需要完整PDF请访问：www.ertongbook.com

目　录

第三部分 XSL-FO

第一部分　XSL 简介

这部分仅包含一章，概要性地介绍了两个 XSL 标准：XSLT 和 XSL-FO，以及这两个标准的发展过程及产生背景。

本部分只给出了少量代码，在本书的第二、三部分的章节中，将提供详细的代码实例。

此为试读,需要完整PDF请访问: www.ertongbook.com

第1章　为什么需要XSL

XML并不是全能的，这是一个让人震惊的事实。

毫无疑问，数据的结构化文本表示格式使得应用的创建大为简化。但事实上XML无法满足表达需求。尽管XML文档内在的逻辑结构是很完美的，但对习惯于Web页面和精美输出的用户来说，XML文档简直是丑陋之极。（当然，对于不仅需要可视化显示的时候，XML提供的信息是不够的。）

而且，你或许已经听到过（可能甚至还议论过），XML真正的优势在于数据交换格式方面，而不在于数据存储格式方面。仔细想一想，你会发现标记语言本身并没有任何特征使得它特别胜任于数据交换的工作。如果将XML信息传递给某个不支持XML的应用或传递给使用其他的XML词汇表的应用，这时候需要分别为上述情况编写单独的转换程序。这样做又能带来什么好处呢?

解决这些困境的答案在于认请上述两种情况其实是同一问题的两种表现。这个问题就是：如何方便地将XML文档转换为其他的某种内容。

1.1　XSL入门

在XML1.0建议书发布6个月以后，W3C在1998年中期发布了第一个关于工作草案，Working Draft（WD）的XML词汇表可扩展样式单语言Extensible Stylesheet Language（XSL）。对于那些习惯于级联样式单（Cascading Style Sheet，CSS，用于HTML的样式化）中的stylesheet（格式化表单）的人说来，XSL显得很奇特[⊖]。这种奇特性表现在两个方面。

首先，理所当然，XSL是以XML为基础的样式单语言来表示CSS属性的。即使你很熟悉XML标记并且很了解CSS，当看到它们在同一个文档中同时出现时，你将会惶恐不安的。

其次，XSL的设计不仅受到CSS和XML的强烈影响，而且还受一种称为文档格式语义及规范语言（Document Style Semantics and Specification Language，DSSSL，用于SGML文档表示的语言）的影响。

下面的DSSSL代码可以用于样式化一个FlixML文档的部分内容。

```
(element (remarks)
  (make paragraph
    font-size: 12pt
    font-weight: bold
    line-spacing: 12pt
      (process-children)))
```

⊖ XSL引入了一些不必要的麻烦：是XSL中的stylesheet呢？还是CSS中的style sheet呢？鉴于本书是关于XSL的，在所有的情况下，都使用stylesheet；在表示CSS中的样式单时，将明确地列出style sheet来。

上面代码片断的第 1 行声明它应用于一个名为 remarks 的元素类型。其他的代码分别与该元素的某个特定字符字体相关，当然，用 CSS 也完全可以实现上述表述。(上面代码中的术语 font-size、font-weight、line-spacing 在 CSS 中都存在。)

但 DSSSL 中有些 CSS 中不存在的东西。可以将 CSS 的属性比作人类语言中的形容词（如“红”、“球状的”、“乱七八糟”、“无法想像的”等)。它们的目的都是用来改变某些底层对象的基本特征：形容词是用来修饰名词的；CSS 是用来修饰元素的内容的。

DSSSL 引入基本语法中的是 verb（动词)，如“make paragraph”和“process-children”。这种样式单语言不仅能修饰元素内容，还可以做些别的事情。例如：在上述代码中，“make paragraph”将 remarks 元素内容转换为一个显示单元——段落（paragraph)；“process-children”告诉支持 DSSSL 的应用搜索整个样式单中应用于 remarks 子元素的 DSSSL 代码，并将这些代码应用于该文档中出现的这类子元素[⊖]。

使用 XSL 草案 1.0 的规则，可以将前面的 DSSSL 代码 XSL 化，生成一个 XSL 样式单：

```
<xsl:template match="remarks">
  <fo:block
    font-size="12pt"
    font-weight="bold"
    line-spacing="12pt">
    <xsl:process-children/>
  </fo:block>
</xsl:template>
```

这些 XSL 代码与 DSSSL 代码很相似[⊖]。注意，这段代码不仅用 XSL 语法重新定义了，使用起始和结束标志表示分组关系，而不再用圆括号了；而且各种元素名称前都加了名字前缀——xsl:和 fo:(在 XSL WD1.0 版中，fo:表示 flow object（流对象)；流对象代表内容的不连续可显示单元)。在第 2 章中，将详细讲述关于 XSL 中名字空间前缀的用法。

XSL WD1.0 标准的演讲：一分为二

在 XSL WD1.0 标准发布几个月后，人们发现该标准有些繁琐及头重脚轻。因为该标准企图在短小的篇幅中定义庞大的内容。

同时，人们很快发现 XSL 中的部分规则很适合其他的用途，而不仅用于样式化及显示 XML 文档。就是用xsl:名字空间前缀来表示 XSL 元素的规则部分，这部分规则与 XML 内容的转化有关。例如，如果能将 XML 文档升级，适应新的 DTD 版本（XML 是基于 DTD 的)，或能将 XML 文档内容从一个词汇表转变到另一个词汇表中，这将会是很有用的。

而且，人们发现，XSL中另一半关于“流对象”部分的细化工作是极为艰巨的。XSL标

⊖ CSS 并不能胜任、或不能很好地实现 XSL 中提供的某些功能。在本章的 1.3 节中，将更为深入地讨论这个主题。

⊖ 这种相似性并不是巧合。如果比较一下 XSL 标准和 DSSSL 标准的制定人员名单和其他的相关信息，会发现有很多人都相同。

准的制定者们想在 XSL 身上克隆 CSS 的各种能力，包括使 XML 文档可以在 Web 页面上显示；而且他们希望 XSL 能更强大——XSL 能直接生成复杂的打印输出，而这也正是 CSS 最不能胜任的。

由于上述原因，在 XSL WD 反复地修改了几遍之后，XSL 标准分裂为两个标准：XSL Transformations（XSLT）和 XSL Formatting Objects（XSL-FO）。

词汇的演变

以前的术语 flow objects 现在被 formatting objects 所取代。我个人非常赞同这种改变，因为 flow objects 显得太专业了。

如果术语 objects 被其他的单词所代替，我想我会更高兴的。但这将会是很滑稽的，因为我们不得不接受我们所能得到的。

1.2　转变 XML

几年前，我还在从事造型软件[⊖]的研发工作。但现在我的工作使我无需再跟踪特殊用途的图像软件，而且我再也没有从事任何视频产品的创作。我现在只知道这种软件比以前功能更强大了。

在任何情况下，这种软件包要完成造型工作，需要设定原始图像和目标图像，并且指定原始图像和目标图像中的对应点。造型软件所做的工作就是填补空白——在原始图像和目标图像之间创建一系列的中间图像，使得原始图像到目标图像之间的变形过程是连续、渐进的。

上述造型原理可以由图 1-1 说明。在图 1-1 中演示了如何将一个日常生活中的手持烤肉工具的厨师形象造型为 B 级恐怖电影中的一个手持长柄斧头的疯子形象。只要将原始人物形象头部的点映射到疯子头部的点（A-〉A'），脚上的点映射到疯子脚上的点（B-〉B'），烤肉调料勺的边角点对应长柄斧的边角点（C-〉C'），其他点的映射关系与此类似。在造型过程中，选的点越多，造型软件就越清楚如何实现二者之间的变形关系，因此变形过程也更为流畅。

XSLT 转变工作的原理与上述造型软件有些相似。大体说来，样式单中的代码描述了 XML 源文档的所有内容，而这些内容也正是所要生成的结果文档中的内容。在 XSLT 转变中，没有必要像上述厨师到刽子手的变形过程中那样在相同的地方建立点到点的一一对应关系，可以将相关的内容放在任意位置。因此可以像图 1-2 中那样将对应信息任意放置。图中的 FlixML 的 title 元素放置在输出的底部，而 remarks 元素放置在顶部。在本例中，这两个元素除内容的放置位置与源文件中的位置不同外，他们还被转移成两种不同的事物。（在本例中，分别由不同的字体表示）。

⊖ 或许你还不了解“造型”这个词，它指使用拉伸或压缩等方法改变一个形体的表面，直到该形体变成另一种形状为止。现在这已经是一些电影和电视节目中的特技效果的常见手法了。例如在《终结者 2》中，Robert Patrick 扮演的那个坏机器人：他可以从一个水银状的机器人变形为任何他想模拟的人的形状。

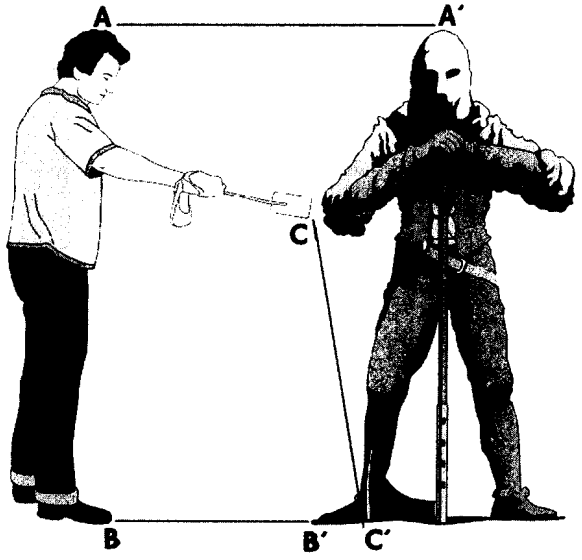

图 1-1 将一个手持烤肉工具的厨师形象造型为手持长柄斧的杀人狂形象。图中左边的点与右边的点是对应的。对于其他的对应关系不明确的点，造型软件会加以确定。注意，在本例中，厨师的形象并不是简单地变形为一个姿势相似的杀人狂魔形象；人物形象将转过 90 度，面对观众，而且单手持烤肉工具的形象也将变形为两手叠加放在斧柄顶部的形象

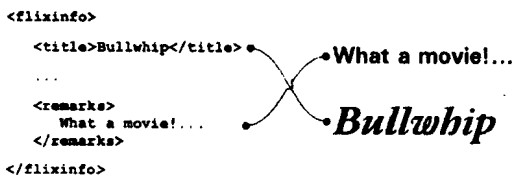

图 1-2 将 FlixML 文档转换为其他的……。与图 1-1 相同，这种“变形”能实现，因为这种软件的用户建立了原始形式与期望的结果之间的对应关系。还与图 1-1 相同的是，不存在特殊的要求，要求给定点的原始位置与最终位置相同

当然，这并不奇怪。早在几十年前，程序员们就开始编程将文本从一种形式转换为其他形式。XSLT 与这些编程努力的显著区别在于以下：

- XSLT 是解决转换问题的一种通用方案。对于各种不同的文档结构，通用解决方案是很有用的，因为我们不需要为各种不同的文档结构编写定制的程序代码。
- 虽然需要为每种结构编写定制的样式单，但这些样式单通常都比用过程化语言（如 C++ 和 Java）编写的程序要简单。XSLT 的样式单不仅容易编写，而且也容易读，不熟悉 XSLT 的人都可以读懂它。
- 因为开发的简单性和易懂性，相对于文档处理应用程序，XSLT 样式单的效率要高得多。为每种文档开发特定的程序的工作量是相当惊人的，更不用说为每个文档开发一

个应用程序，这也正是阻碍文本处理应用被广泛接受的一个主要原因。（如果没有XSLT，这种情况也可能在XML身上出现）。

简而言之，XSLT在文档处理应用的两个互相矛盾的目标，即简单性和通用性之间起着重要的平衡作用。

1.3 XML的格式化

在前几页中，使用了一段DSSSL代码并将它与CSS作比较。如我曾提到过的那样，DSSSL及其后的XSL向CSS的“样式语法”中添加了“动词”（verb）的概念。

那段DSSSL代码中还表达或暗示了一些其他的事物，在CSS找不到与这些事物对等的事物。

1.3.1 非元素内容的格式化

我再次重述一遍：开始的DSSSL介绍指出，该样式单的部分应用于名为remarks的元素类型。这里要解释的是，DSSSL不仅可以格式化元素内容，还可以格式化其他的内容，如属性值等。相反，CSS只能格式化元素内容。对于任意XML应用，尤其是对于那些内容主要由属性组成的XML文档，这一点可以说是CSS的一个跛脚之处。（例如有许多空元素的情况。）

CSS选择器中的“属性”（Attribute）

事实上，CSS选择器完全能够通过元素的属性，或根据这些属性的值选择元素。

但我要强调的是，CSS并不能实现这些属性值的自显示，只能显示那些有特殊属性的元素的值。

假设有一个完整的FlixML文档。根元素flix-info有一对属性：author属性表示该FlixML文档的作者，copyright表示该文档的创作日期。因此，一个flixinfo元素的开始标签例子可能像这样：

```
<flixinfo author="John E. Simpson" copyright="2001">
```

你可能会想在文档中的显示作者及版权日期信息。如果仅使用CSS，则这些属性就像根本不存在，因此无法显示他们。

1.3.2 内容的重新排序

在CSS中，要想给原始文档的内容排序将会是件很麻烦的事。如果在FlixML文档中，title元素是以根元素flixinfo的第一个子元素形式出现的，则CSS样式单无法使它出现在其他位置。

这个限制中隐藏着另外一个事实，那就是使用CSS单时，引用文档内容片段的频率不能比该内容片段在原文档中出现的频率高。例如，可以在文档的开始处使用大的sans-serif字体显示一部电影的名字，但不能在文档的脚注或结尾处显示它（除非标题出现在如嵌入的Plot-

summary 元素中)。

1.3.3 面向 Web 页面的显示

CSS 是为配合 HTML 的使用而发展起来的——在带滚动条的浏览器窗口中控制内容的显示。尽管 CSS 为了支持其他的媒体格式而做了某些妥协(如 CSS2.0 加入了“paged media”属性,以支持插入分页标识),但 CSS 主要还是面向 Web 应用的。

这并没有贬低 CSS 的意思。CSS 的确是为 Web 应用而制定的;我们不能因为它不支持打印及其他的媒体格式而认为它是有缺陷的,就像不能因为我的 DVD 播放器不能喂给我的猫吃而认为它是垃圾一样。

问题在于有很多用途并不是面向 Web 应用的显示模式所能胜任的。如果它可以胜任的话,那么 Adobe Acrobat 的 PDF 格式就不会取得如此的成功了。XSL-FO 就是专门为了修正 CSS 在这些缺陷而出现有,XSL-FO 不仅提供了 CSS 的标准属性集,还提供了对打印的支持,如提供 header(页眉)、footer(页脚)等打印专用元素。

1.3.4 CSS 是否适用于所有与 XML 相关的内容

的确,CSS 中的这些缺陷,即使所有的缺陷加在一起,也不足以推翻 CSS 这个旧的格式化表单标准。

继续学习及使用 CSS 的主要原因在于它是专为 Web 应用而制定的。如果只关心 XML 如何在 Web 页面上的显示问题,则 CSS 无疑是您最好的选择。

首先,CSS 可以直接格式化 XML,就像格式化 HTML 那样。所要做的仅仅是将 XML 文档中的元素作为选择器,然后将该文档连接到 CSS 格式化表单。在文档中有一个处理暗示(Processing Instruction,PI)表示这个连接,如下面的代码所示:

```
<?xml-stylesheet
    type="text/css" href="my_styles.css"?>
```

伪属性 type 必须填写为上面的值,否则 CSS 无法识别;伪属性 href 是格式化表单请求的名字(必要的话,可以包含协议、服务器名、路径等信息)。

当然,其他访问该文档的用户必须有支持 XML 的浏览器,而且该浏览器还要支持 CSS,才能正确解析该文档。

相反,在本书中的大部分例子将 XML 文档转换为 HTML 文档,然后再用 CSS 对 HTML 结果文档做相应的处理[⊖]。

虽然本书中不会过多地介绍 CSS,但如果要做一个 Web 应用的开发者,建议您至少应该了解 CSS 中的基础知识。

⊖ 严格地说,本书的许多例子是将 XML 转变成 XHTML(Extensible Hypertext Markup Language,可扩展超文本标记语言),XHTML 是 W3C 组织所推荐的。当然,XSLT 也完全适用于将 XML 转变成旧的 HTML 格式。

此为试读,需要完整PDF请访问: www.ertongbook.com