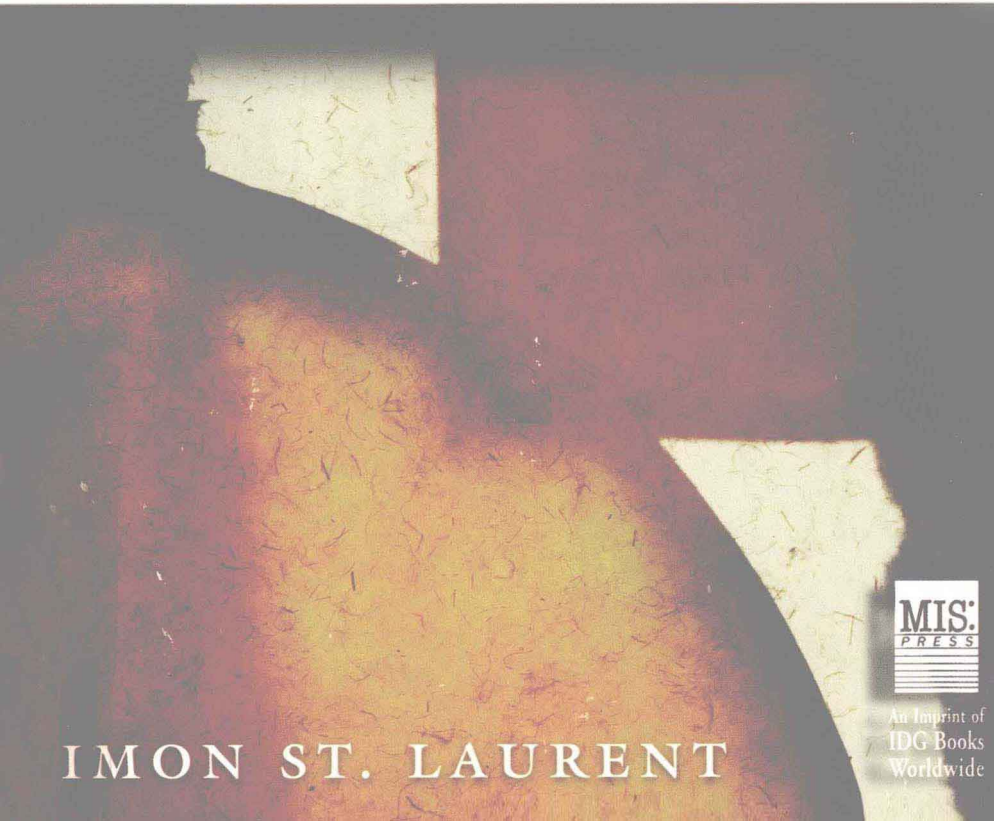
XML

A PRIMER



IMON ST. LAURENT

MIS: PRESS
An Imprint of
IDG Books
Worldwide

XML: A PRIMER

SIMON ST. LAURENT

MIS: PRESS

MIS:Press
An imprint of IDG Books Worldwide
An International Data Group Company
919 E. Hillsdale Blvd.
Suite 400
Foster City, CA 94404
www.idgbooks.com
www.mispress.com

Library of Congress Catalog Card No.: 97-45820

ISBN: 1-55828-592-X

Printed in the United States of America

10 9 8 7 6 5 4 3

1P/RZ/QW/ZY/IN

Distributed in the United States by IDG Books Worldwide, Inc.

Distributed by Macmillan Canada for Canada; by Transworld Publishers Limited in the United Kingdom; by IDG Norge Books for Norway; by IDG Sweden Books for Sweden; by Woodslane Pty. Ltd. for Australia; by Woodslane (NZ) Ltd. for New Zealand; by Addison Wesley Longman Singapore Pte Ltd. for Singapore, Malaysia, Thailand, Indonesia and Korea; by Norma Comunicaciones S.A. for Colombia; by Intersoft for South Africa; by International Thomson Publishing for Germany, Austria and Switzerland; by Toppan Company Ltd. for Japan; by Distribuidora Cuspide for Argentina; by Livraria Cultura for Brazil; by Ediciencia S.A. for Ecuador; by Ediciones ZETA S.C.R. Ltda. for Peru; by WS Computer Publishing Corporation, Inc., for the Philippines; by Unalis Corporation for Taiwan; by Contemporanea de Ediciones for Venezuela; by Computer Book & Magazine Store for Puerto Rico; by Express Computer Distributors for the Caribbean and West Indies. Authorized Sales Agent: Anthony Rudkin Associates for the Middle East and North Africa.

For general information on IDG Books Worldwide's books in the U.S., please call our Consumer Customer Service department at 800-762-2974. For reseller information, including discounts and premium sales, please call our Reseller Customer Service department at 800-434-3422.

For information on where to purchase IDG Books Worldwide's books outside the U.S., please contact our International Sales department at 650-655-3200 or fax 650-655-3297.

For information on foreign language translations, please contact our Foreign & Subsidiary Rights department at 650-655-3021 or fax 650-655-3281.

For sales inquiries and special prices for bulk quantities, please contact our Sales department at 650-655-3200 or write to the address above.

For information on using IDG Books Worldwide's books in the classroom or for ordering examination copies, please contact our Educational Sales department at 800-434-2086 or fax 317-596-5499.

For press review copies, author interviews, or other publicity information, please contact our Public Relations department at 650-655-3000 or fax 650-655-3299.

For authorization to photocopy items for corporate, personal, or educational use, please contact Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923, or fax 978-750-4470.

Associate Publisher: *Paul Farrell*

Editor: *Ann Lush*
Technical Editor: *Steven Champeon*
Managing Editor: *Shari Chappell*

Production Editor: *Kitty May*
Copy Edit Manager: *Karen Tongish*
Copy Editor: *Sara Black*

XML:

A PRIMER

For Tracey, who makes my world sparkle.

ACKNOWLEDGMENTS

I'd like to thank my technical editor, Steven Champeon, for helping me sort out the sordid details of SGML, HTML, and XML, making this book far clearer as well as more accurate. He was a much-needed partner in a number of battles. Tracey Cranston read the opening chapters to make sure they stayed intelligible, kept me excited about XML, and made the process far smoother by smiling at me regularly. Ann Lush, my editor, got me excited about XML in the first place, got the proposal through a number of obstacles, and helped keep me sane throughout the process. I'd also like to thank Sara Black for making my prose clearer and Kitty May for presenting it so well. My mother helped keep the writing moving with regular shipments of cookies, for which I'm very grateful.

INTRODUCTION

XML, to a certain extent, is HTML (Hypertext Markup Language) done right. XML (eXtensible Markup Language) offers a unique combination of flexibility, simplicity, and readability by both humans and machines. HTML developers who have spent years cursing the strange formatting quirks of HTML and the extreme difficulty of converting anything *from* HTML are in for a treat. XML gives developers the ability to create and manipulate their own tags and works smoothly with Cascading Style Sheets to allow developers to create pages that are as elegantly presented as they are structured. Programmers can build simple parsers to read XML data (or, better, reuse parsers built by others), making it an excellent format for interchanging data.

If you're an HTML developer who's interested in XML you're in the right place. This book attempts to explain XML in terms that any reasonably experienced HTML developer can understand. Although some of the concepts may be difficult, XML itself is really quite approachable. Unlike the Standard Generalized Markup Language (SGML), its behemoth predecessor, XML uses a reasonably concise syntax that can provide developers with an enormous amount of power—without the learning curve associated with SGML. Although XML is, in a sense, SGML-lite, I've done my best to avoid describing XML from an SGML perspective.

Because much of the best literature (and experience) available on creating document type definitions and using marked-up documents has come from the SGML community, I've pointed out some of the

many differences between SGML and XML. If you don't know or care about SGML, you can safely ignore all such information. Still, it won't hurt to learn a bit about SGML, if you have the time and interest.

I have great hopes for XML. XML seems to me the best tool for accomplishing great things with markup, a significant improvement on both HTML and SGML. It has more flexibility than HTML, without the mind-numbing complexity of SGML. XML holds out the promise of markup that both humans and machines can interpret, making it easy for developers to debug their documents and for programmers to build systems around them. Although the "paperless office" has been just over the horizon for the last 20 years, XML, in combination with ubiquitous networking, may finally provide the tools needed to make it a reality. (Don't hold your breath, though; old habits die very slowly.) The simplicity of XML makes it useful for small projects, whereas its clear structures make it useful for larger projects. XML can be massaged, manipulated, processed, fragmented, and rebuilt far more easily than previous formats.

Unfortunately, at press time, there aren't many tools available that work with XML. This book has been written around some of the few tools available—Tim Bray's Lark, Microsoft's MSXML, Norbert Mikula's NXP, and Peter Murray-Rust's Jumbo, all of which deserve praise as bold pioneers. James Clark's NSGMLSU (part of his SP package) deserves honorable mention as a powerful parser, albeit one from the more staid world of SGML. The two leading browsers, Microsoft's Internet Explorer and Netscape Communicator, offer feeble support for XML and no support, respectively. Nonetheless, both companies have made public commitments to providing support and hopefully will make good on those commitments in reasonably short order.

This book definitely focuses on hand-coding XML. Although I certainly hope that hand-coding will be quickly replaced by rapidly evolving tools, hand-coded XML will be around for a short while at least. It took a while for the HTML toolset to grow, and

undoubtedly XML will have its growing pains as well. Even though many SGML tools are available and can be applied to XML development, their price ranges and target market seem to stay well above the broader audience for XML. With time, prices will fall, and tools will become more powerful, just as they have in every other area of computing.

This book is a primer and not a complete guide to all things XML. The document type definitions need applications built around them for them to be useful, and most of the tools presented can give only a basic idea of XML's potential. I fully expect that "graduates" of this book will be eager to move on to the next great thing. With any luck, those graduates (and people who have read other books as well) will spread the word about XML, building an XML community as rich and varied as the HTML community is now.

CONTENTS IN BRIEF

CONTENTS

CHAPTER 1

Let Data Be Data

XML promises to transform the basic structure of the Web, moving beyond HTML and replacing it with a stronger, more extensible architecture. It promises to return the Web to content-based structures instead of the format-based structures imposed by designers frustrated by the immaturity of Web design tools. It may also free the Web from the tyranny of browser developers by ending their monopoly on element development and implementation.

The World Wide Web Consortium (W3C) has moved ahead of the commercial browser developers with a very promising new approach to markup. XML, the eXtensible Markup Language, makes it possible for developers to create their own mutually interoperable dialects of markup languages, including but not limited to HTML. This could bring about a cease-fire in the browser wars between Netscape and Microsoft as added features shift to a component model from browser code. More immediately, it allows developers to create markup structures based on logical content rather than formatting. That will make it easier for humans and computers to search for specific content-based information on pages instead of just searching the entire text of a page. XML, in concert with Cascading Style Sheets (CSS), should allow developers to create beautiful pages that are easily managed.

The WYSIWYG Disaster

The first word processor I used was a very simple text editor. I thought it was really amazing how the screen could move around my

cursor point to make my 40-column screen display most of an 80-column page, but for the most part it was only good for doing homework and writing other similarly boring documents that I printed out on my lovely dot-matrix printer. After working with computers for a few years, programming them and cursing them, I gave up and bought an electric typewriter. It let me do some pretty fancy things, like underline text without having to enter bizarre escape codes. There wasn't a good way to type boldface text, but I didn't have to worry about wasting acres of paper because of a typo in code. The typewriter gave me what-you-see-is-what-you-get (WYSIWYG) in a classical ink-on-paper kind of way.

I stuck with my typewriter for a couple of years, until I discovered the Macintosh. I'd hated the Mac when it first came out, because every magazine I got covered an expensive machine I didn't own. It didn't even have a decent programming package. But when I encountered the Mac again about four years later, I was thrilled. It was actually fun to write papers, because I could toggle all the style information, write in multiple columns, and even use 72-point type once in a while. It didn't look very good on my Imagewriter, but it was pretty amazing compared to my old dot-matrix computer text. I turned in papers with headlines, bibliographies that used proper italics, multiple columns, and even a picture or two. Writing wasn't just about spewing out sentences anymore. I could create headlines, subheads, tables, footnotes, and use all kinds of other formatting to give even a short paper a set of structures that made it look smart. Using styles made it even easier: apply a set of formatting tools once, then call it up as a named set. It seemed like magic.

Ten years later I still format my documents with headings and subheads. Fortunately, I'm not as concerned about footnotes, but I've developed a new problem: it's hard to reuse my old documents. When I was writing papers for a grade it didn't matter very much—I wrote the paper, turned it in, and never thought about it again. Now I spend my days working with piles of information written years ago by people thousands of miles away, and converting the files into the