



JSTL

Practical Guide for JSP Programmers

Sue Spielman

JSTL

Practical Guide for JSP Programmers

The Morgan Kaufmann Practical Guides Series

Series Editor, Michael J. Donahoo

JSTL: Practical Guide for JSP Programmers

Sue Spielman

JSP: Practical Guide for Java Programmers

Robert J. Brunner

Java: Practical Guide for Programmers

Zbigniew M. Sikora

The Struts Framework: Practical Guide for Java Programmers

Sue Spielman

Multicast Sockets: Practical Guide for Programmers

David Makofske and Kevin Almeroth

TCP/IP Sockets in Java: Practical Guide for Programmers

Kenneth L. Calvert and Michael J. Donahoo

TCP/IP Sockets in C: Practical Guide for Programmers

Michael J. Donahoo and Kenneth L. Calvert

JDBC: Practical Guide for Java Programmers

Gregory D. Speegle

For further information on these books and for a list of forthcoming titles, please visit our website at <http://www.mkp.com/practical>

JSTL

Practical Guide for JSP Programmers

Sue Spielman

Switchback Software LLC



ELSEVIER

AMSTERDAM • BOSTON • HEIDELBERG • LONDON
NEW YORK • OXFORD • PARIS • SAN DIEGO
SAN FRANCISCO • SINGAPORE • SYDNEY • TOKYO

Morgan Kaufmann is an imprint of Elsevier



MORGAN KAUFMANN PUBLISHERS

This book is printed on acid-free paper.

Senior Editor:	Rick Adams
Developmental Editor:	Karyn Johnson
Publishing Services Manager:	Simon Crump
Senior Project Manager:	Angela G. Dooley
Project Management:	Keyword
Composition:	CEPHA
Cover Design:	Cate Barr
Printer:	Maple-Vail

Copyright 2004, Elsevier Science (USA)

All rights reserved.

No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopy, recording, or any information storage and retrieval system, without permission in writing from the publisher.

Designations used by companies to distinguish their products are often claimed as trademarks or registered trademarks. In all instances in which Morgan Kaufmann Publishers is aware of a claim, the product names appear in initial capital or all capital letters. Readers, however, should contact the appropriate companies for more complete information regarding trademarks and registration.

In no event shall the publisher or the author be liable for any direct, indirect, special, consequential, or inconsequential damages. No warranties are expressed or implied, including warranties or merchantability or fitness for a particular purpose.

Permissions may be sought directly from Elsevier's Science & Technology Rights Department in Oxford, UK: phone: (+44) 1865 843830, fax: (+44) 1865 853333, e-mail: permissions@elsevier.com.uk. You may also complete your request on-line via the Elsevier homepage (<http://elsevier.com>), by selecting "Customer Support" and then "Obtaining Permissions."

Morgan Kaufmann

An imprint of Elsevier Science

340 Pine Street, Sixth Floor, San Francisco, California 94104-3205, USA

<http://www.mkp.com>

Library of Congress Catalog Card Number: 2003107479

International Standard Book Number: 0-12-656755-7

Printed in the United States of America

03 1 2 3 4 5

Act mindfully
Accept entirely
Move strongly
Think softly
Speak beautifully
Live simply
Love completely

Anonymous

This Page Intentionally Left Blank

Dedicated to my family

Mommasita, Papar, Liz, Ken, Randi, Ryan, Christopher, Bev, and Sid

This Page Intentionally Left Blank

Contents

Preface xvii

1 Introduction 1

- 1.1 What Exactly Is the JSTL? 1
- 1.2 Why a JSP Standard Tag Library? 2
- 1.3 Why Now? 2
- 1.4 Why You *Really* Want to Use the JSTL 3
- 1.5 The Need for Encapsulation 3
- 1.6 Functional Overview 3
- 1.7 JSTL Tag Libraries 4
- 1.8 Getting Ready to Use the JSTL 6
- 1.9 The Road to the JSTL 6
 - 1.9.1 Dynamic vs. Static Content 6
 - 1.9.2 Using Dynamic Content 7
 - 1.9.3 Using Dynamic Web Features 7
 - 1.9.4 Server-Side Processing 7
- 1.10 Servlets to the Rescue 8
- 1.11 Hello My Friend Servlet 9
- 1.12 JavaServer Pages 12
- 1.13 When a JSP, When a Servlet? 14
- 1.14 Evolving JSP 14
- 1.15 Custom Actions in Action 15
 - 1.15.1 Why Use a Custom Action 15

- 1.15.2 Hello My Friend Using Custom Actions 16
- 1.15.3 The TLD File 16
- 1.15.4 The Tag Handler 18
- 1.16 The Power of Tag Libraries 19
 - 1.16.1 Need for a Tag Library 20
- 1.17 Making Life Easier, JSTL in Action 20

2 JSTL Basics 23

- 2.1 Environment Setup 23
- 2.2 Using the Book Examples 24
- 2.3 JSP Scopes 25
- 2.4 JSTL Scoped Variables 27
 - 2.4.1 Var and Scope Attributes 27
 - 2.4.2 Variable Visibility 28
- 2.5 Dynamic and Static Attributes 29
- 2.6 Handling Errors and Exceptions 29
- 2.7 Action Body Content 31
- 2.8 Configuration Settings 32
- 2.9 The Config Class 33
- 2.10 Summary 34

3 Using the Expression Language 35

- 3.1 Implicit Objects Available in the EL 36
- 3.2 Accessing Data Structures 37
- 3.3 EL Operators 38
 - 3.3.1 Relational Operators 38
 - 3.3.2 Other Operators 38
 - 3.3.3 Using Operators 38
- 3.4 Automatic Type Conversion 40
- 3.5 Default Values 41
- 3.6 Summary 42

4 Working with the Core Actions 43

- 4.1 Writing Output to the JspWriter 43
- 4.2 Setting Variables 45
- 4.3 Removing Variables 49

- 4.4 Using `<c:catch>` 50
 - 4.4.1 Handling Exceptions 51
- 4.5 Decisions, Decisions, Decisions—Conditional Actions 52
 - 4.5.1 Simple Conditional 53
 - 4.5.2 Mutually Exclusive Conditionals 54
 - 4.5.3 Creating Custom Logic Actions 56
- 4.6 Handling Iterators 56
 - 4.6.1 `<c:forEach>` 57
 - 4.6.2 Paging through a Large Collection 59
 - 4.6.3 Looping with a Collection 63
 - 4.6.4 Tokenizing Data Using `<c:forTokens>` 67
- 4.7 URL-Related Actions 71
 - 4.7.1 `<c:import>` 71
 - 4.7.2 `<c:url>` 73
 - 4.7.3 Creating and Using Dynamic Links 73
 - 4.7.4 `<c:param>` 74
 - 4.7.5 `<c:redirect>` 75
- 4.8 Untangling the Web We Weave 76
 - 4.8.1 The Power of `<c:import>` 76
 - 4.8.2 `<c:import>` and the Composite View Pattern 76
 - 4.8.3 Storing Imported Content 80
 - 4.8.4 Using Character Encoding 82
- 4.9 Redirecting 83
- 4.10 Summary 84

5 Working with the XML Actions 85

- 5.1 Overview of Supporting Technologies 85
- 5.2 eXtensible Markup Language (XML) 86
 - 5.2.1 Using XML Files for Data Storage 87
 - 5.2.2 XML APIs 87
- 5.3 eXtensible Stylesheet Language (XSL) 88
 - 5.3.1 Allowing for Transformation 88
 - 5.3.2 XSL Languages 89
- 5.4 XML Path Language (XPath) 90
 - 5.4.1 Library Functions 91
- 5.5 Variable Mappings 91
- 5.6 Using the Select Attribute 93
- 5.7 Accessing Resources 93
 - 5.7.1 Node Types 94
 - 5.7.2 Node Functions 94

- 5.8 eXtensible Stylesheet Language Transformation (XSLT) 95
 - 5.8.1 XSLT Namespace 95
- 5.9 Parsing XML Documents 96
 - 5.9.1 <x:parse> Action 96
 - 5.9.2 Filtering 97
 - 5.9.3 Using the Advanced <x:parse> Attributes 99
- 5.10 Using <x:out> and <x:set> 100
 - 5.10.1 <x:out> Action 100
 - 5.10.2 <x:set> Action 101
- 5.11 <x:set> and <x:out> in Action 101
- 5.12 Using XML Documents to Determine Flow Control 102
 - 5.12.1 <x:if> Action 103
 - 5.12.2 Using <x:if> 104
 - 5.12.3 <x:choose>, <x:when>, and <x:otherwise> Actions 105
- 5.13 Going Loopy with <x:forEach> 106
 - 5.13.1 Nested forEach Loops 107
- 5.14 XML Transformation Actions 110
 - 5.14.1 <x:transform> Action 110
- 5.15 Transforming Content 111
- 5.16 Providing Parameters to Transformations 113
 - 5.16.1 <x:param> Action 113
 - 5.16.2 Performing Multiple Transformations 115
- 5.17 Summary 115

6 Working with the Internationalization and Formatting Actions 117

- 6.1 Locales 118
 - 6.1.1 Internationalization vs. Localization 118
- 6.2 Why be Language Independent? 119
- 6.3 Localizing an Application Using Resource Bundles 119
- 6.4 Types of I18N Architectures 120
- 6.5 First, the <fmt:message> Action 121
- 6.6 Localization Context 122
- 6.7 Localization Context Sample 123
- 6.8 Preferred Locales 123
 - 6.8.1 Setting the Preferred Locales 124
- 6.9 Formatting Locales 124
- 6.10 How Resource Bundles are Decided 125
 - 6.10.1 Resource Bundle Lookup Differences 127