

Extracted from:

# Language Implementation Patterns

---

Create Your Own Domain-Specific and  
General Programming Languages

This PDF file contains pages extracted from Language Implementation Patterns, published by the Pragmatic Bookshelf. For more information or to purchase a paperback or PDF copy, please visit <http://www.pragprog.com>.

**Note:** This extract contains some colored text (particularly in code listing). This is available only in online versions of the books. The printed versions are black and white. Pagination might vary between the online and printer versions; the content is otherwise identical.

Copyright © 2010 The Pragmatic Programmers, LLC.

All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the publisher.

The  
Pragmatic  
Programmers

# Language Implementation Patterns

Create Your Own Domain-  
Specific and General  
Programming Languages

Edited by Susannah Davidson Pfäzler

*Terence Parr*





Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and The Pragmatic Programmers, LLC was aware of a trademark claim, the designations have been printed in initial capital letters or in all capitals. The Pragmatic Starter Kit, The Pragmatic Programmer, Pragmatic Programming, Pragmatic Bookshelf and the linking *g* device are trademarks of The Pragmatic Programmers, LLC.

With permission of the creator we hereby publish the chess images in Chapter 11 under the following licenses:

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License"

([http://commons.wikimedia.org/wiki/Commons:GNU\\_Free\\_Documentation\\_License](http://commons.wikimedia.org/wiki/Commons:GNU_Free_Documentation_License)).

Every precaution was taken in the preparation of this book. However, the publisher assumes no responsibility for errors or omissions, or for damages that may result from the use of information (including program listings) contained herein.

Our Pragmatic courses, workshops, and other products can help you and your team create better software and have more fun. For more information, as well as the latest Pragmatic titles, please visit us at

<http://www.pragprog.com>

Copyright © 2010 Terence Parr.

All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the publisher.

Printed in the United States of America.

ISBN-10: 1-934356-45-X

ISBN-13: 978-1-934356-45-6

Printed on acid-free paper.

P1.0 printing, December 2009

Version: 2010-1-13

# Contents

---

<b>Acknowledgments</b>	<b>13</b>
<b>Preface</b>	<b>14</b>
What to Expect from This Book . . . . .	15
How This Book Is Organized . . . . .	16
What You'll Find in the Patterns . . . . .	17
Who Should Read This Book . . . . .	17
How to Read This Book . . . . .	18
Languages and Tools Used in This Book . . . . .	19
<b>I Getting Started with Parsing</b>	<b>21</b>
<b>1 Language Applications Cracked Open</b>	<b>22</b>
1.1 The Big Picture . . . . .	22
1.2 A Tour of the Patterns . . . . .	24
1.3 Dissecting a Few Applications . . . . .	28
1.4 Choosing Patterns and Assembling Applications . . . . .	36
<b>2 Basic Parsing Patterns</b>	<b>39</b>
2.1 Identifying Phrase Structure . . . . .	40
2.2 Building Recursive-Descent Parsers . . . . .	42
2.3 Parser Construction Using a Grammar DSL . . . . .	44
2.4 Tokenizing Sentences . . . . .	45
P.1. Mapping Grammars to Recursive-Descent Recognizers	47
P.2. <i>LL(1)</i> Recursive-Descent Lexer . . . . .	51
P.3. <i>LL(1)</i> Recursive-Descent Parser . . . . .	56
P.4. <i>LL(k)</i> Recursive-Descent Parser . . . . .	61

<b>3</b>	<b>Enhanced Parsing Patterns</b>	<b>67</b>
3.1	Parsing with Arbitrary Lookahead . . . . .	68
3.2	Parsing like a Pack Rat . . . . .	70
3.3	Directing the Parse with Semantic Information . . . . .	70
	P.5. Backtracking Parser . . . . .	73
	P.6. Memoizing Parser . . . . .	80
	P.7. Predicated Parser . . . . .	86
<b>II</b>	<b>Analyzing Languages</b>	<b>89</b>
<b>4</b>	<b>Building Intermediate Form Trees</b>	<b>90</b>
4.1	Why We Build Trees . . . . .	92
4.2	Building Abstract Syntax Trees . . . . .	94
4.3	Quick Introduction to ANTLR . . . . .	101
4.4	Constructing ASTs with ANTLR Grammars . . . . .	103
	P.8. Parse Tree . . . . .	107
	P.9. Homogeneous AST . . . . .	111
	P.10. Normalized Heterogeneous AST . . . . .	113
	P.11. Irregular Heterogeneous AST . . . . .	116
<b>5</b>	<b>Walking and Rewriting Trees</b>	<b>118</b>
5.1	Walking Trees and Visitation Order . . . . .	119
5.2	Encapsulating Node Visitation Code . . . . .	122
5.3	Automatically Generating Visitors from Grammars . . . . .	124
5.4	Decoupling Tree Traversal from Pattern Matching . . . . .	127
	P.12. Embedded Heterogeneous Tree Walker . . . . .	130
	P.13. External Tree Visitor . . . . .	133
	P.14. Tree Grammar . . . . .	136
	P.15. Tree Pattern Matcher . . . . .	140
<b>6</b>	<b>Tracking and Identifying Program Symbols</b>	<b>148</b>
6.1	Collecting Information About Program Entities . . . . .	149
6.2	Grouping Symbols into Scopes . . . . .	151
6.3	Resolving Symbols . . . . .	156
	P.16. Symbol Table for Monolithic Scope . . . . .	158
	P.17. Symbol Table for Nested Scopes . . . . .	163
<b>7</b>	<b>Managing Symbol Tables for Data Aggregates</b>	<b>172</b>
7.1	Building Scope Trees for Structs . . . . .	173
7.2	Building Scope Trees for Classes . . . . .	175
	P.18. Symbol Table for Data Aggregates . . . . .	178
	P.19. Symbol Table for Classes . . . . .	184

<b>8</b>	<b>Enforcing Static Typing Rules</b>	<b>198</b>
	P.20. Computing Static Expression Types . . . . .	201
	P.21. Automatic Type Promotion . . . . .	210
	P.22. Enforcing Static Type Safety . . . . .	218
	P.23. Enforcing Polymorphic Type Safety . . . . .	225
<b>III</b>	<b>Building Interpreters</b>	<b>233</b>
<b>9</b>	<b>Building High-Level Interpreters</b>	<b>234</b>
	9.1 Designing High-Level Interpreter Memory Systems . .	235
	9.2 Tracking Symbols in High-Level Interpreters . . . . .	237
	9.3 Processing Instructions . . . . .	239
	P.24. Syntax-Directed Interpreter . . . . .	240
	P.25. Tree-Based Interpreter . . . . .	245
<b>10</b>	<b>Building Bytecode Interpreters</b>	<b>254</b>
	10.1 Programming Bytecode Interpreters . . . . .	256
	10.2 Defining an Assembly Language Syntax . . . . .	258
	10.3 Bytecode Machine Architecture . . . . .	260
	10.4 Where to Go from Here . . . . .	265
	P.26. Bytecode Assembler . . . . .	267
	P.27. Stack-Based Bytecode Interpreter . . . . .	274
	P.28. Register-Based Bytecode Interpreter . . . . .	282
<b>IV</b>	<b>Translating and Generating Languages</b>	<b>291</b>
<b>11</b>	<b>Translating Computer Languages</b>	<b>292</b>
	11.1 Syntax-Directed Translation . . . . .	294
	11.2 Rule-Based Translation . . . . .	295
	11.3 Model-Driven Translation . . . . .	297
	11.4 Constructing a Nested Output Model . . . . .	305
	P.29. Syntax-Directed Translator . . . . .	309
	P.30. Rule-Based Translator . . . . .	315
	P.31. Target-Specific Generator Classes . . . . .	321
<b>12</b>	<b>Generating DSLs with Templates</b>	<b>325</b>
	12.1 Getting Started with StringTemplate . . . . .	326
	12.2 Characterizing StringTemplate . . . . .	329
	12.3 Generating Templates from a Simple Input Model . . .	330
	12.4 Reusing Templates with a Different Input Model . . . .	333

12.5	Using a Tree Grammar to Create Templates . . . . .	336
12.6	Applying Templates to Lists of Data . . . . .	343
12.7	Building Retargetable Translators . . . . .	349
<b>13</b>	<b>Putting It All Together</b>	<b>360</b>
13.1	Finding Patterns in Protein Structures . . . . .	360
13.2	Using a Script to Build 3D Scenes . . . . .	361
13.3	Processing XML . . . . .	362
13.4	Reading Generic Configuration Files . . . . .	364
13.5	Tweaking Source Code . . . . .	365
13.6	Adding a New Type to Java . . . . .	366
13.7	Pretty Printing Source Code . . . . .	367
13.8	Compiling to Machine Code . . . . .	368
<b>A</b>	<b>Bibliography</b>	<b>370</b>
	<b>Index</b>	<b>372</b>

# The Pragmatic Bookshelf

---

The Pragmatic Bookshelf features books written by developers for developers. The titles continue the well-known Pragmatic Programmer style and continue to garner awards and rave reviews. As development gets more and more difficult, the Pragmatic Programmers will be there with more titles and products to help you stay on top of your game.

## Visit Us Online

---

### Language Implementation Patterns' Home Page

<http://pragprog.com/titles/tpdsl>

Source code from this book, errata, and other resources. Come give us feedback, too!

### Register for Updates

<http://pragprog.com/updates>

Be notified when updates and new books become available.

### Join the Community

<http://pragprog.com/community>

Read our weblogs, join our online discussions, participate in our mailing list, interact with our wiki, and benefit from the experience of other Pragmatic Programmers.

### New and Noteworthy

<http://pragprog.com/news>

Check out the latest pragmatic developments, new titles and other offerings.

## Buy the Book

---

If you liked this eBook, perhaps you'd like to have a paper copy of the book. It's available for purchase at our store: [pragprog.com/titles/tpdsl](http://pragprog.com/titles/tpdsl).

## Contact Us

---

Online Orders:	<a href="http://www.pragprog.com/catalog">www.pragprog.com/catalog</a>
Customer Service:	<a href="mailto:support@pragprog.com">support@pragprog.com</a>
Non-English Versions:	<a href="mailto:translations@pragprog.com">translations@pragprog.com</a>
Pragmatic Teaching:	<a href="mailto:academic@pragprog.com">academic@pragprog.com</a>
Author Proposals:	<a href="mailto:proposals@pragprog.com">proposals@pragprog.com</a>
Contact us:	1-800-699-PROG (+1 919 847 3884)