

Domain-Specific Languages

Martin Fowler

With Rebecca Parsons

AAAddison-Wesley

Upper Saddle River, NJ • Boston • Indianapolis • San Francisco
New York • Toronto • Montreal • London • Munich • Paris • Madrid
Sydney • Tokyo • Singapore • Mexico City

Contents

Preface	xix
Part I: Narratives	1
Chapter 1: An Introductory Example	3
Gothic Security.....	3
<i>Miss Grant's Controller</i>	4
The State Machine Model.....	5
Programming Miss Grant's Controller.....	9
Languages and Semantic Model.....	16
Using Code Generation.....	19
Using Language Workbenches.....	22
Visualization.....	24
Chapter 2: Using Domain-Specific Languages	27
Defining Domain-Specific Languages.....	27
<i>Boundaries of DSLs</i>	29
<i>Fragmentary and Stand-alone DSLs</i>	32
Why Use a DSL?.....	33
<i>Improving Development Productivity</i>	33
<i>Communication with Domain Experts</i>	34
<i>Change in Execution Context</i>	35
<i>Alternative Computational Model</i>	36
Problems with DSLs.....	36
<i>Language Cacophony</i>	37
<i>Cost of Building</i>	37
<i>Ghetto Language</i>	38
<i>Blinkered Abstraction</i>	39



Wider Language Processing.....	39
DSL Lifecycle.....	40
What Makes a Good DSL Design?.....	42
Chapter 3: Implementing DSLs.....	43
Architecture of DSL Processing.....	43
The Workings of a Parser.....	47
Grammars, Syntax, and Semantics.....	49
Parsing Data.....	50
Macros.....	52
Testing DSLs.....	53
<i>Testing the Semantic Model</i>	53
<i>Testing the Parser</i>	57
<i>Testing the Scripts</i>	61
Handling Errors.....	62
Migrating DSLs.....	64
Chapter 4: Implementing an Internal DSL.....	67
Fluent and Command-Query APIs.....	68
The Need for a Parsing Layer.....	71
Using Functions.....	72
Literal Collections.....	77
Using Grammars to Choose Internal Elements.....	79
Closures.....	80
Parse Tree Manipulation.....	82
Annotation.....	84
Literal Extension.....	85
Reducing the Syntactic Noise.....	85
Dynamic Reception.....	86
Providing Some Type Checking.....	87
Chapter 5: Implementing an External DSL.....	89
Syntactic Analysis Strategy.....	89
Output Production Strategy.....	92
Parsing Concepts.....	94
<i>Separated Lexing</i>	94
<i>Grammars and Languages</i>	95
<i>Regular, Context-Free, and Context-Sensitive Grammars</i>	96
<i>Top-Down and Bottom-Up Parsing</i>	98

Mixing-in Another Language.....	100
XML DSLs.....	101
Chapter 6: Choosing between Internal and External DSLs.....	105
Learning Curve.....	105
Cost of Building.....	106
Programmer Familiarity.....	107
Communication with Domain Experts.....	108
Mixing In the Host Language.....	108
Strong Expressiveness Boundary.....	109
Runtime Configuration.....	110
Sliding into Generality.....	110
Composing DSLs.....	111
Summing Up.....	111
Chapter 7: Alternative Computational Models.....	113
A Few Alternative Models.....	116
<i>Decision Table</i>	116
<i>Production Rule System</i>	117
<i>State Machine</i>	118
<i>Dependency Network</i>	119
<i>Choosing a Model</i>	120
Chapter 8: Code Generation.....	121
Choosing What to Generate.....	122
How to Generate.....	124
Mixing Generated and Handwritten Code.....	126
Generating Readable Code.....	127
Preparse Code Generation.....	128
Further Reading.....	128
Chapter 9: Language Workbenches.....	129
Elements of Language Workbenches.....	130
Schema Definition Languages and Meta-Models.....	131
Source and Projectional Editing.....	136
<i>Multiple Representations</i>	138
Illustrative Programming.....	138
Tools Tour.....	140
Language Workbenches and CASE tools.....	141
Should You Use a Language Workbench?.....	142

Part II: Common Topics	145
Chapter 10: A Zoo of DSLs	147
Graphviz.....	147
JMock.....	149
CSS.....	150
Hibernate Query Language (HQL).....	151
XAML.....	152
FIT.....	155
Make et al.....	156
Chapter 11: Semantic Model	159
How It Works.....	159
When to Use It.....	162
The Introductory Example (Java).....	163
Chapter 12: Symbol Table	165
How It Works.....	166
<i>Statically Typed Symbols</i>	167
When to Use It.....	168
Further Reading.....	168
Dependency Network in an External DSL (Java and ANTLR).....	168
Using Symbolic Keys in an Internal DSL (Ruby).....	170
Using Enums for Statically Typed Symbols (Java).....	172
Chapter 13: Context Variable	175
How It Works.....	175
When to Use It.....	176
Reading an INI File (C#).....	176
Chapter 14: Construction Builder	179
How It Works.....	179
When to Use It.....	180
Building Simple Flight Data (C#).....	180
Chapter 15: Macro	183
How It Works.....	184
<i>Textual Macros</i>	184
<i>Syntactic Macros</i>	188
When to Use It.....	192

Chapter 16: Notification	193
How It Works.....	194
When to Use It.....	194
A Very Simple Notification (C#).....	194
Parsing Notification (Java).....	195
Part III: External DSL Topics	199
Chapter 17: Delimiter-Directed Translation	201
How It Works.....	201
When to Use It.....	204
Frequent Customer Points (C#).....	205
<i>Semantic Model</i>	205
<i>The Parser</i>	207
Parsing Nonautonomous Statements with Miss Grant's Controller (Java).....	211
Chapter 18: Syntax-Directed Translation	219
How It Works.....	220
<i>The Lexer</i>	221
<i>Syntactic Analyzer</i>	223
<i>Output Production</i>	226
<i>Semantic Predicates</i>	226
When to Use It.....	227
Further Reading.....	227
Chapter 19: BNF	229
How It Works.....	229
<i>Multiplicity Symbols (Kleene Operators)</i>	231
<i>Some Other Useful Operators</i>	232
<i>Parsing Expression Grammars</i>	233
<i>Converting EBNF to Basic BNF</i>	234
<i>Code Actions</i>	236
When to Use It.....	238
Chapter 20: Regex Table Lexer (by Rebecca Parsons)	239
How It Works.....	240
When to Use It.....	241
Lexing Miss Grant's Controller (Java).....	241

Chapter 21: Recursive Descent Parser (by Rebecca Parsons)	245
How It Works.....	246
When to Use It.....	249
Further Reading.....	249
Recursive Descent and Miss Grant's Controller (Java).....	250
Chapter 22: Parser Combinator (by Rebecca Parsons)	255
How It Works.....	256
<i>Dealing with the Actions</i>	259
<i>Functional Style of Combinators</i>	260
When to Use It.....	261
Parser Combinators and Miss Grant's Controller (Java).....	261
Chapter 23: Parser Generator	269
How It Works.....	269
<i>Embedding Actions</i>	270
When to Use It.....	272
Hello World (Java and ANTLR).....	272
<i>Writing the Basic Grammar</i>	272
<i>Building the Syntactic Analyzer</i>	274
<i>Adding Code Actions to the Grammar</i>	276
<i>Using Generation Gap</i>	278
Chapter 24: Tree Construction	281
How It Works.....	281
When to Use It.....	284
Using ANTLR's Tree Construction Syntax (Java and ANTLR) . . .	284
<i>Tokenizing</i>	285
<i>Parsing</i>	286
<i>Populating the Semantic Model</i>	288
Tree Construction Using Code Actions (Java and ANTLR).....	292
Chapter 25: Embedded Translation	299
How It Works.....	299
When to Use It.....	300
Miss Grant's Controller (Java and ANTLR).....	300
Chapter 26: Embedded Interpretation	305
How It Works.....	305
When to Use It.....	306
A Calculator (ANTLR and Java).....	306



Chapter 27: Foreign Code	309
How It Works.....	309
When to Use It.....	311
Embedding Dynamic Code (ANTLR, Java, and Javascript).....	311
<i>Semantic Model</i>	312
<i>Parser</i>	315
Chapter 28: Alternative Tokenization	319
How It Works.....	319
<i>Quoting</i>	320
<i>Lexical State</i>	322
<i>Token Type Mutation</i>	324
<i>Ignoring Token Types</i>	325
When to Use It.....	326
Chapter 29: Nested Operator Expression	327
How It Works.....	327
<i>Using Bottom-Up Parsers</i>	328
<i>Top-Down Parsers</i>	329
When to Use It.....	331
Chapter 30: Newline Separators	333
How It Works.....	333
When to Use It.....	335
Chapter 31: External DSL Miscellany	337
Syntactic Indentation.....	337
Modular Grammars.....	339
Part IV: Internal DSL Topics	341
Chapter 32: Expression Builder	343
How It Works.....	344
When to Use It.....	344
A Fluent Calendar with and without a Builder (Java).....	345
Using Multiple Builders for the Calendar (Java).....	348
Chapter 33: Function Sequence	351
How It Works.....	351
When to Use It.....	352
Simple Computer Configuration (Java).....	352

Chapter 34: Nested Function	357
How It Works.....	357
When to Use It.....	359
The Simple Computer Configuration Example (Java).....	360
Handling Multiple Different Arguments with Tokens (C#).....	361
Using Subtype Tokens for IDE Support (Java).....	363
Using Object Initializers (C#).....	365
Recurring Events (C#).....	366
<i>Semantic Model</i>	366
<i>The DSL</i>	369
Chapter 35: Method Chaining	373
How It Works.....	373
<i>Builders or Values</i>	375
<i>Finishing Problem</i>	375
<i>Hierarchic Structure</i>	376
<i>Progressive Interfaces</i>	377
When to Use It.....	377
The Simple Computer Configuration Example (Java).....	378
Chaining with Properties (C#).....	381
Progressive Interfaces (C#).....	382
Chapter 36: Object Scoping	385
How It Works.....	386
When to Use It.....	386
Security Codes (C#).....	387
<i>Semantic Model</i>	387
<i>DSL</i>	390
Using Instance Evaluation (Ruby).....	392
Using an Instance Initializer (Java).....	394
Chapter 37: Closure	397
How It Works.....	397
When to Use It.....	402
Chapter 38: Nested Closure	403
How It Works.....	403
When to Use It.....	405
Wrapping a Function Sequence in a Nested Closure (Ruby).....	405
Simple C# Example (C#).....	408
Using Method Chaining (Ruby).....	409

Function Sequence with Explicit Closure Arguments (Ruby).....	411
Using Instance Evaluation (Ruby).....	412
Chapter 39: Literal List.....	417
How It Works.....	417
When to Use It.....	417
Chapter 40: Literal Map.....	419
How It Works.....	419
When to Use It.....	420
The Computer Configuration Using Lists and Maps (Ruby).....	420
Evolving to Greenspun Form (Ruby).....	422
Chapter 41: Dynamic Reception.....	427
How It Works.....	428
When to Use It.....	429
Promotion Points Using Parsed Method Names (Ruby).....	430
<i>Model</i>	431
<i>Builder</i>	433
Promotion Points Using Chaining (Ruby).....	434
<i>Model</i>	435
<i>Builder</i>	435
Removing Quoting in the Secret Panel Controller (JRuby).....	438
Chapter 42: Annotation.....	445
How It Works.....	446
<i>Defining an Annotation</i>	446
<i>Processing Annotations</i>	447
When to Use It.....	449
Custom Syntax with Runtime Processing (Java).....	449
Using a Class Method (Ruby).....	451
Dynamic Code Generation (Ruby).....	452
Chapter 43: Parse Tree Manipulation.....	455
How It Works.....	455
When to Use It.....	456
Generating IMAP Queries from C# Conditions (C#).....	457
<i>Semantic Model</i>	458
<i>Building from C#</i>	460
<i>Stepping Back</i>	465

CONTENTS

Chapter 44: Class Symbol Table	467
How It Works.....	468
When to Use It.....	469
Statically Typed Class Symbol Table (Java).....	469
Chapter 45: Textual Polishing	477
How It Works.....	477
When to Use It.....	478
Polished Discount Rules (Ruby).....	478
Chapter 46: Literal Extension	481
How It Works.....	481
When to Use It.....	482
Recipe Ingredients (C#).....	483
Part V: Alternative Computational Models	485
Chapter 47: Adaptive Model	487
How It Works.....	488
<i>Incorporating Imperative Code into an Adaptive Model</i>	489
<i>Tools</i>	491
When to Use It.....	492
Chapter 48: Decision Table	495
How It Works.....	495
When to Use It.....	497
Calculating the Fee for an Order (C#).....	497
<i>Model</i>	497
<i>The Parser</i>	502
Chapter 49: Dependency Network	505
How It Works.....	506
When to Use It.....	508
Analyzing Potions (C#).....	508
<i>Semantic Model</i>	509
<i>The Parser</i>	511
Chapter 50: Production Rule System	513
How It Works.....	514
<i>Chaining</i>	515
<i>Contradictory Inferences</i>	515
<i>Patterns in Rule Structure</i>	516
When to Use It.....	517

Validations for club membership (C#).....	517
<i>Model</i>	518
<i>Parser</i>	519
<i>Evolving the DSL</i>	520
Eligibility Rules: extending the club membership (C#).....	521
<i>The Model</i>	523
<i>The Parser</i>	525
Chapter 51: State Machine	527
How It Works.....	527
When to Use It.....	529
Secret Panel Controller (Java).....	530
Part VI: Code Generation	531
Chapter 52: Transformer Generation	533
How It Works.....	533
When to Use It.....	535
Secret Panel Controller (Java generating C).....	535
Chapter 53: Templated Generation	539
How It Works.....	539
When to Use It.....	541
Generating the Secret Panel State Machine with Nested Conditionals (Velocity and Java generating C).....	541
Chapter 54: Embedment Helper	547
How It Works.....	548
When to Use It.....	549
Secret Panel States (Java and ANTLR).....	549
Should a Helper Generate HTML? (Java and Velocity).....	552
Chapter 55: Model-Aware Generation	555
How It Works.....	556
When to Use It.....	556
Secret Panel State Machine (C).....	557
Loading the State Machine Dynamically (C).....	564
Chapter 56: Model Ignorant Generation	567
How It Works.....	567
When to Use It.....	568
Secret Panel State Machine as Nested Conditionals (C).....	568



CONTENTS

Chapter 57: Generation Gap	571
How It Works.....	571
When to Use It.....	573
Generating Classes from a Data Schema (Java and a Little Ruby) ...	573
Bibliography	579
Index	581