
Table of Contents

Foreword	xv
Preface	xvii
1. Zero to Sixty: Introducing Scala	1
Why Scala?	1
If You Are a Java Programmer...	1
If You Are a Ruby, Python, etc. Programmer...	2
Introducing Scala	4
The Seductions of Scala	7
Installing Scala	8
For More Information	10
A Taste of Scala	10
A Taste of Concurrency	16
Recap and What's Next	21
2. Type Less, Do More	23
In This Chapter	23
Semicolons	23
Variable Declarations	24
Method Declarations	25
Method Default and Named Arguments (Scala Version 2.8)	26
Nesting Method Definitions	28
Inferring Type Information	29
Literals	36
Integer Literals	36
Floating-Point Literals	37
Boolean Literals	38
Character Literals	38
String Literals	39
Symbol Literals	39

Tuples	40
Option, Some, and None: Avoiding nulls	41
Organizing Code in Files and Namespaces	44
Importing Types and Their Members	45
Imports are Relative	46
Abstract Types And Parameterized Types	47
Reserved Words	49
Recap and What's Next	52
3. Rounding Out the Essentials	53
Operator? Operator?	53
Syntactic Sugar	54
Methods Without Parentheses and Dots	55
Precedence Rules	56
Domain-Specific Languages	57
Scala if Statements	58
Scala for Comprehensions	59
A Dog-Simple Example	59
Filtering	60
Yielding	60
Expanded Scope	61
Other Looping Constructs	61
Scala while Loops	61
Scala do-while Loops	62
Generator Expressions	62
Conditional Operators	63
Pattern Matching	63
A Simple Match	64
Variables in Matches	64
Matching on Type	65
Matching on Sequences	65
Matching on Tuples (and Guards)	66
Matching on Case Classes	67
Matching on Regular Expressions	68
Binding Nested Variables in Case Clauses	69
Using try, catch, and finally Clauses	70
Concluding Remarks on Pattern Matching	71
Enumerations	72
Recap and What's Next	74
4. Traits	75
Introducing Traits	75
Traits As Mixins	76

Stackable Traits	82
Constructing Traits	86
Class or Trait?	87
Recap and What's Next	88
5. Basic Object-Oriented Programming in Scala	89
Class and Object Basics	89
Parent Classes	91
Constructors in Scala	91
Calling Parent Class Constructors	94
Nested Classes	95
Visibility Rules	96
Public Visibility	98
Protected Visibility	99
Private Visibility	100
Scoped Private and Protected Visibility	102
Final Thoughts on Visibility	110
Recap and What's Next	110
6. Advanced Object-Oriented Programming In Scala	111
Overriding Members of Classes and Traits	111
Attempting to Override final Declarations	112
Overriding Abstract and Concrete Methods	112
Overriding Abstract and Concrete Fields	114
Overriding Abstract and Concrete Fields in Traits	114
Overriding Abstract and Concrete Fields in Classes	119
Overriding Abstract Types	120
When Accessor Methods and Fields Are Indistinguishable: The Uniform Access Principle	123
Companion Objects	126
Apply	127
Unapply	129
Apply and UnapplySeq for Collections	132
Companion Objects and Java Static Methods	133
Case Classes	136
Syntactic Sugar for Binary Operations	139
The copy Method in Scala Version 2.8	140
Case Class Inheritance	140
Equality of Objects	142
The equals Method	143
The == and != Methods	143
The ne and eq Methods	143
Array Equality and the sameElements Method	143

Recap and What's Next	144
7. The Scala Object System	145
The Predef Object	145
Classes and Objects: Where Are the Statics?	148
Package Objects	150
Sealed Class Hierarchies	151
The Scala Type Hierarchy	155
Linearization of an Object's Hierarchy	159
Recap and What's Next	164
8. Functional Programming in Scala	165
What Is Functional Programming?	165
Functions in Mathematics	166
Variables that Aren't	166
Functional Programming in Scala	167
Function Literals and Closures	169
Purity Inside Versus Outside	169
Recursion	170
Tail Calls and Tail-Call Optimization	171
Trampoline for Tail Calls	172
Functional Data Structures	172
Lists in Functional Programming	173
Maps in Functional Programming	173
Sets in Functional Programming	174
Other Data Structures in Functional Programming	174
Traversing, Mapping, Filtering, Folding, and Reducing	174
Traversal	175
Mapping	175
Filtering	178
Folding and Reducing	179
Functional Options	181
Pattern Matching	182
Partial Functions	183
Currying	184
Implicits	186
Implicit Conversions	186
Implicit Function Parameters	188
Final Thoughts on Implicits	189
Call by Name, Call by Value	189
Lazy Vals	190
Recap: Functional Component Abstractions	192

9. Robust, Scalable Concurrency with Actors	193
The Problems of Shared, Synchronized State	193
Actors	193
Actors in Abstract	194
Actors in Scala	194
Sending Messages to Actors	195
The Mailbox	196
Actors in Depth	197
Effective Actors	202
Traditional Concurrency in Scala: Threading and Events	203
One-Off Threads	203
Using <code>java.util.concurrent</code>	204
Events	204
Recap and What's Next	210
10. Herding XML in Scala	211
Reading XML	211
Exploring XML	212
Looping and Matching XML	213
Writing XML	214
A Real-World Example	215
Recap and What's Next	216
11. Domain-Specific Languages in Scala	217
Internal DSLs	218
A Payroll Internal DSL	222
Infix Operator Notation	223
Implicit Conversions and User-Defined Types	223
Apply Methods	224
Payroll Rules DSL Implementation	224
Internal DSLs: Final Thoughts	229
External DSLs with Parser Combinators	230
About Parser Combinators	230
A Payroll External DSL	230
A Scala Implementation of the External DSL Grammar	233
Generating Paychecks with the External DSL	239
Internal Versus External DSLs: Final Thoughts	244
Recap and What's Next	245
12. The Scala Type System	247
Reflecting on Types	248
Understanding Parameterized Types	249
Manifests	250

Parameterized Methods	251
Variance Under Inheritance	251
Variance of Mutable Types	255
Variance In Scala Versus Java	256
Implementation Notes	259
Type Bounds	259
Upper Type Bounds	259
Lower Type Bounds	260
A Closer Look at Lists	261
Views and View Bounds	263
Nothing and Null	267
Understanding Abstract Types	267
Parameterized Types Versus Abstract Types	270
Path-Dependent Types	272
C.this	273
C.super	273
path.x	274
Value Types	275
Type Designators	275
Tuples	275
Parameterized Types	275
Annotated Types	275
Compound Types	276
Infix Types	276
Function Types	277
Type Projections	279
Singleton Types	279
Self-Type Annotations	279
Structural Types	283
Existential Types	284
Infinite Data Structures and Laziness	285
Recap and What's Next	288

13. Application Design	289
Annotations	289
Enumerations Versus Pattern Matching	300
Thoughts On Annotations and Enumerations	304
Enumerations Versus Case Classes and Pattern Matching	304
Using Nulls Versus Options	306
Options and for Comprehensions	308
Exceptions and the Alternatives	311
Scalable Abstractions	313
Fine-Grained Visibility Rules	314

Mixin Composition	316
Self-Type Annotations and Abstract Type Members	317
Effective Design of Traits	321
Design Patterns	325
The Visitor Pattern: A Better Alternative	326
Dependency Injection in Scala: The Cake Pattern	334
Better Design with Design By Contract	340
Recap and What's Next	342
14. Scala Tools, Libraries, and IDE Support	343
Command-Line Tools	343
scalac Command-Line Tool	343
The scala Command-Line Tool	345
The scalap, javap, and jad Command-Line Tools	350
The scaladoc Command-Line Tool	352
The sbaz Command-Line Tool	352
The fsc Command-Line Tool	353
Build Tools	353
Integration with IDEs	353
Eclipse	354
IntelliJ	356
NetBeans	359
Text Editors	360
Test-Driven Development in Scala	361
ScalaTest	361
Specs	363
ScalaCheck	365
Other Notable Scala Libraries and Tools	367
Lift	367
Scalaz	368
Scalax	368
MetaScala	368
JavaRebel	368
Miscellaneous Smaller Libraries	368
Java Interoperability	369
Java and Scala Generics	369
Using Scala Functions in Java	372
JavaBean Properties	374
AnyVal Types and Java Primitives	375
Scala Names in Java Code	375
Java Library Interoperability	377
AspectJ	377
The Spring Framework	381

Terracotta	384
Hadoop	384
Recap and What's Next	385

Appendix: References	387
-----------------------------------	------------

Glossary	393
-----------------------	------------

Index	407
--------------------	------------