

contents

preface xvii
acknowledgments xx
about this book xxiii

PART 1 SPRING ESSENTIALS 1

- 1** ***A Spring jump start*** 3
- 1.1 Why Spring? 5
 - A day in the life of a J2EE developer* 5
 - *Spring's pledge* 6
 - 1.2 What is Spring? 8
 - Spring modules* 9
 - 1.3 Spring jump start 12
 - 1.4 Understanding inversion of control 15
 - Injecting dependencies* 16
 - *IoC in action* 16
 - IoC in enterprise applications* 23
 - 1.5 Applying aspect-oriented programming 25
 - Introducing AOP* 25
 - *AOP in action* 27
 - *AOP in the enterprise* 30

- 1.6 Spring alternatives 33
 - Comparing Spring to EJB* 33
 - *Considering other lightweight containers* 36
 - *Web frameworks* 38
 - *Persistence frameworks* 40
- 1.7 Summary 40

2 **Wiring beans** 42

- 2.1 Containing your beans 44
 - Introducing the BeanFactory* 44
 - *Working with an application context* 46
 - *A bean's life* 47
- 2.2 Basic wiring 50
 - Wiring with XML* 54
 - *Adding a bean* 55
 - *Injecting dependencies via setter methods* 58
 - *Injecting dependencies via constructor* 65
- 2.3 Autowiring 69
 - Handling ambiguities of autowiring* 71
 - *Mixing auto and explicit wiring* 72
 - *Autowiring by default* 72
 - *To autowire or not to autowire* 72
- 2.4 Working with Spring's special beans 73
 - Postprocessing beans* 74
 - *Postprocessing the bean factory* 76
 - Externalizing the configuration* 78
 - *Customizing property editors* 80
 - *Resolving text messages* 83
 - *Listening for events* 85
 - *Publishing events* 86
 - *Making beans aware* 87
- 2.5 Summary 90

3 **Creating aspects** 91

- 3.1 Introducing AOP 92
 - Defining AOP terminology* 93
 - *Spring's AOP implementation* 95
- 3.2 Creating advice 97
 - Before advice* 99
 - *After advice* 101
 - *Around advice* 102
 - Throws advice* 104
 - *Introduction advice* 105
- 3.3 Defining pointcuts 105
 - Defining a pointcut in Spring* 105
 - *Understanding advisors* 107
 - *Using Spring's static pointcuts* 107
 - Using dynamic pointcuts* 111
 - *Pointcut operations* 113

- 3.4 Creating introductions 115
 - Implementing IntroductionInterceptor* 115
 - *Creating an IntroductionAdvisor* 119
 - *Using introduction advice carefully* 120
- 3.5 Using ProxyFactoryBean 122
- 3.6 Autoproxying 124
 - BeanNameAutoProxyCreator* 124
 - *DefaultAdvisorAutoProxyCreator* 126
 - *Metadata autoproxying* 128
- 3.7 Summary 128

PART 2 SPRING IN THE BUSINESS LAYER..... 131

4 *Hitting the database* 133

- 4.1 Learning Spring's DAO philosophy 134
 - Understanding Spring's DataAccessException* 135
 - Working with DataSources* 137
 - *Consistent DAO support* 139
- 4.2 Using JDBC with Spring 141
 - The problem with JDBC code* 142
 - *Using JdbcTemplate* 144
 - Creating operations as objects* 152
 - *Auto-incrementing keys* 155
- 4.3 Introducing Spring's ORM framework support 156
- 4.4 Integrating Hibernate with Spring 157
 - Hibernate overview* 157
 - *Managing Hibernate resources* 159
 - *Accessing Hibernate through HibernateTemplate* 162
 - *Subclassing HibernateDaoSupport* 163
- 4.5 Spring and JDO 164
 - Configuring JDO* 164
 - *Accessing data with JdoTemplate* 165
- 4.6 Spring and iBATIS 166
 - Setting up SQL Maps* 167
 - Using SqlMapClientTemplate* 168
- 4.7 Spring and OJB 169
 - Setting up OJB's PersistenceBroker* 169
- 4.8 Summary 171

- ## 5 **Managing transactions 173**
- 5.1 Understanding transactions 174
 - Explaining transactions in only four words* 176
 - *Understanding Spring's transaction management support* 177
 - *Introducing Spring's transaction manager* 178
 - 5.2 Programming transactions in Spring 181
 - 5.3 Declaring transactions 183
 - Understanding transaction attributes* 185
 - *Declaring a simple transaction policy* 189
 - 5.4 Declaring transactions by method name 191
 - Using NameMatchTransactionAttributeSource* 191
 - Shortcutting name-matched transactions* 194
 - 5.5 Declaring transactions with metadata 195
 - Sourcing transaction attributes from metadata* 196
 - Declaring transactions with Commons Attributes* 197
 - 5.6 Trimming down transaction declarations 201
 - Inheriting from a parent TransactionProxyFactoryBean* 202
 - Autoproxying transactions* 203
 - 5.7 Summary 206
- ## 6 **Remoting 207**
- 6.1 Spring remoting overview 208
 - 6.2 Working with RMI 212
 - Wiring RMI services* 212
 - *Exporting RMI services* 214
 - 6.3 Remoting with Hessian and Burlap 218
 - Accessing Hessian/Burlap services* 219
 - *Exposing bean functionality with Hessian/Burlap* 220
 - 6.4 Using Http invoker 223
 - Accessing services via HTTP* 224
 - *Exposing beans as HTTP Services* 225
 - 6.5 Working with EJBs 226
 - Accessing EJBs* 227
 - *Developing Spring-enabled EJBs* 231
 - 6.6 Using JAX-RPC web services 233
 - Referencing a web service with JAX-RPC* 234
 - *Wiring a web service in Spring* 236
 - 6.7 Summary 238

7	<i>Accessing enterprise services</i>	240
7.1	Retrieving objects from JNDI	241
	<i>Working with conventional JNDI</i>	241
	<i>Proxying JNDI objects</i>	243
7.2	Sending e-mail	244
7.3	Scheduling tasks	248
	<i>Scheduling with Java's Timer</i>	248
	<i>Using the Quartz scheduler</i>	250
	<i>Invoking methods on a schedule</i>	254
7.4	Sending messages with JMS	256
	<i>Sending messages with JMS templates</i>	257
	<i>Consuming messages</i>	261
	<i>Converting messages</i>	263
7.5	Summary	266
PART 3 SPRING IN THE WEB LAYER.....		267

8	<i>Building the web layer</i>	269
8.1	Getting started with Spring MVC	270
	<i>A day in the life of a request</i>	271
	<i>Configuring DispatcherServlet</i>	272
	<i>Spring MVC in a nutshell</i>	275
8.2	Mapping requests to controllers	279
	<i>Mapping URLs to bean names</i>	280
	<i>Using SimpleUrlHandlerMapping</i>	281
	<i>Using metadata to map controllers</i>	281
	<i>Working with multiple handler mappings</i>	282
8.3	Handling requests with controllers	283
	<i>Writing a simple controller</i>	285
	<i>Processing commands</i>	287
	<i>Processing form submissions</i>	289
	<i>Processing complex forms with wizards</i>	294
	<i>Handling multiple actions in one controller</i>	301
	<i>Working with Throwaway controllers</i>	305
8.4	Resolving views	307
	<i>Using template views</i>	308
	<i>Resolving view beans</i>	310
	<i>Choosing a view resolver</i>	313
8.5	Using Spring's bind tag	314
8.6	Handling exceptions	317
8.7	Summary	317

9 **View layer alternatives 319**

- 9.1 Using Velocity templates 321
 - Defining the Velocity view 321* ▪ *Configuring the Velocity engine 322* ▪ *Resolving Velocity views 323* ▪ *Formatting dates and numbers 324* ▪ *Exposing request and session attributes 325* ▪ *Binding form fields in Velocity 326*
- 9.2 Working with FreeMarker 327
 - Constructing a FreeMarker view 328* ▪ *Configuring the FreeMarker engine 329* ▪ *Resolving FreeMarker views 330* ▪ *Binding form fields in FreeMarker 330*
- 9.3 Designing page layout with Tiles 332
 - Tile views 332* ▪ *Tile controllers 335*
- 9.4 Generating non-HTML output 337
 - Producing Excel spreadsheets 338* ▪ *Generating PDF documents 340* ▪ *Generating other non-HTML files 343*
- 9.5 Summary 344

10 **Working with other web frameworks 346**

- 10.1 Working with Jakarta Struts 347
 - Registering the Spring plug-in 348* ▪ *Implementing Spring-aware Struts actions 348* ▪ *Delegating actions 350*
- 10.2 Working with Tapestry 352
 - Replacing the Tapestry Engine 353* ▪ *Loading Spring beans into Tapestry pages 355*
- 10.3 Integrating with JavaServer Faces 357
 - Resolving variables 357* ▪ *Publishing request handled events 361*
- 10.4 Integrating with WebWork 362
 - WebWork 1 363* ▪ *XWork/WebWork2 364*
- 10.5 Summary 365

11 **Securing Spring applications 367**

- 11.1 Introducing the Acegi Security System 368
 - Security interceptors 369* ▪ *Authentication managers 370* ▪ *Access decisions managers 370* ▪ *Run-as managers 370*

11.2	Managing authentication	371
	<i>Configuring a provider manager</i>	371
	▪ <i>Authenticating against a database</i>	373
	▪ <i>Authenticating against an LDAP repository</i>	382
	▪ <i>Enabling Single Sign-On with Acegi and Yale CAS</i>	384
11.3	Controlling access	389
	<i>Voting access decisions</i>	389
	▪ <i>Deciding how to vote</i>	390
	<i>Handling voter abstinence</i>	392
11.4	Securing web applications	392
	<i>Proxying Acegi's filters</i>	394
	▪ <i>Enforcing web security</i>	397
	<i>Processing a login</i>	400
	▪ <i>Setting up the security context</i>	406
	<i>Ensuring a secure channel</i>	407
	▪ <i>Using the Acegi tag library</i>	411
11.5	Securing method invocations	412
	<i>Creating a security aspect</i>	412
	▪ <i>Securing methods using metadata</i>	414
11.6	Summary	416
<i>appendix A:</i>	<i>Spring setup</i>	417
	A.1 Downloading Spring	418
	A.2 Choosing a distribution	418
	A.3 Setting up your project	419
	A.4 Building with Ant	420
<i>appendix B:</i>	<i>Spring-related projects</i>	422
	B.1 AppFuse	423
	B.2 Rich Client Project	424
	B.3 Spring.NET	424
	<i>index</i>	427