

Symbols

@After annotations 412
@Autowired annotation 27, 325
@Before annotations 412
@Before methods 412
@BeforeClass annotation 412
@ContextConfiguration annotation 27, 426
@DirtyContext annotation 426
@ExceptionHandler annotation 325
@Header 319
@PathVariable annotation 325
@RequestBody annotation 325
@RequestMapping annotation 107, 325
@RequestParam annotation 107
@ResponseStatus annotation 107, 325
@RunWith annotation 27
@Scheduled annotation 102–103
@ServiceActivator 319
syntax 26
#expression syntax 26

Numerics

200 (OK) status code 341
202 status code 323
404 (NOT FOUND) status code 341
409 status code 323

A

abstract attribute 58, 61, 84–85
abstractions, task executor 377–378
Accenture 5
acceptance testing 409

ACID properties 252
ACME 197
Advanced Message Queuing Protocol. *See* AMQP
after method 82
afterChunk 254
AfterJob annotation 79
afterJob method 79
afterProcess method 81, 418–419
afterPropertiesSet 152–153
afterRead method 81
AfterStep annotation 82
afterStep method 80, 82
afterWrite method 81
allow-start-if-complete attribute 62, 85, 244–245
AMQP (Advanced Message Queuing Protocol) 389
AnnotatedImportProductsJobListener class 79
AnnotationMethodHandlerAdapter 343
annotations, launching batch jobs from
 Spring 102–103
AOP (Aspect-Oriented Programming) 240–242
Apache Commons IO project 21
Apache Maven 3 development environment
 439–445
 description of 440–441
 installation 439–440
 projects, adding dependencies to 442, 444–445
appendAllowed property 161
application code, retrying in with
 RetryTemplate 239–240
application components 34
application developer, stopping batch jobs
 for 113–116
 from chunk-oriented step 114–116
 from tasklet 113–114
ApplicationContext 361

- applicationContext.xml file 105
- ApplicationEvent class 362
- ApplicationEventMonitoringNotifier class 362
- ApplicationEventPublisherAware 362
- ApplicationListener 362
- applications
 - batch 4–5
 - online store 8–9, 278–279
 - web, deploying Spring MVC framework in 321–322
- ArgPreparedStatementSetter 205–206
- Aspect-Oriented Programming. *See* AOP
- assembler property 368
- Assert class 412–413
- assertEquals(expectedValue, actualValue)
 - method 412
- AssertFile class 433
- assertNotNull(value) method 412
- assertNull(value) method 412
- assertTrue(booleanValue) method 412
- assertUpdates 180–181
- asynchronous batch job launches, vs. synchronous
 - batch job launches 89–90
- attributes, of transactions 255–256
- AUTO_ACKNOWLEDGE 264

B

- BackToBackPatternClassifier class 190–192
- batch applications 3–5
- Batch class 154, 212
- batch configuration 53–86
 - advanced 75–86
 - configuration inheritance 83–86
 - listeners 78–83
 - SpEL 76–77
 - step scope 75–76
 - job repository 72–75
 - choosing 72–73
 - specifying parameters for 73–75
 - jobs 57–60
 - configuring 58–60
 - job entities hierarchy 57–58
 - steps 60–72
 - XML vocabulary 54–57
 - namespace 54–56
 - XML features 56–57
- batch domain language 33–35
 - how Spring Batch interacts with outside world 35
 - main components of 34–35
 - reasons for using 33
- batch jobs 87–116
 - embedding scheduler in container 91
 - idempotent operations in 272–273
 - launching from command line
 - handling exit codes 94–97
 - overview 90–91
 - with job parameters 94
 - without job parameters 93
 - launching from cron 98–99
 - configuring 98–99
 - jobs suited for use with 99
 - launching from Spring scheduler 99–103
 - options for 100
 - setting up 100–101
 - with annotations 102–103
 - with XML 101–102
 - launching from web application 103–109
 - embedding Spring Batch in 104–105
 - with HTTP request 105–109
 - reason for building with 9
 - Spring Batch launcher API 88–89
 - stopping gracefully 109–116
 - for application developer 113–116
 - for operator 110–113
 - synchronous vs. asynchronous launches 89–90
 - triggering jobs by external event 91–92
- batch metadata 263, 454–455
- batch namespace 39, 54–55
- batch prefix 19
- batch processes
 - reason for using 9–10
 - testing 23–28
 - leveraging SpEL for configuration 25–26
 - setting up test infrastructure 23–25
 - writing test for job 26–28
- batch processing 57
- batch status, vs. exit status 282
- BATCH_JOB_EXECUTION table 351
- BATCH_JOB_INSTANCE table 351
- BATCH_STEP_EXECUTION table 351
- batch-default.properties file 454
- Batch-Graph tab 44, 447, 449
- batch-oriented processing, Spring Batch 5
- BatchMonitoringNotifier interface 358–359
- BatchSqlUpdate class 159
- BatchStatus 282
- batchUpdate 159
- bean element 55, 76
- Bean Validation standard, validating items
 - with 216–219
- BeanFactory 147
- BeanFactoryPostProcessor interface 76
- BeanFactoryStepLocator class 401–402
- BeanPropertySqlParameterSource 180
- beans namespace prefix 54–55
- beans, holder 296–300
- BeanWrapperFieldExtractor class 164, 167, 170–171, 173

- BeanWrapperFieldSetMapper 128–130
- before method 82
- beforeChunk 254
- BeforeJob annotation 79
- beforeJob method 79
- beforeProcess method 81
- beforeRead method 81
- BeforeStep annotation 82
- beforeStep method 82
- beforeWrite method 81
- BEGIN 165
- benefits, of testing 410
- best effort pattern
 - with JMS 263–266
 - avoiding duplicate messages 266
 - avoiding losing messages with transaction synchronization 264–266
 - issues with message delivery 263–264
- BigDecimal type 14–15, 129
- binding, late
 - writing in job execution context and reading using 295–296
 - writing to holder bean and using to late binding to read 299–300
- Bitronix Transaction Manager 262
- BookProduct 169–170
- bookProductLineAggregator 170–171
- bulletproof jobs 223–250
 - description of 224
 - designing bulletproof job 224–225
 - skip, retry, and restart features in action 226–227
 - techniques for bulletproofing jobs 225–226
 - restart on error 242–250
 - completed steps 245–246
 - enabling between job executions 243–244
 - in middle of chunk-oriented step 247–250
 - limiting number of restarts 246
 - no restart option 244–245
 - retrying on error 234–242
 - configuring retryable exceptions 234–236
 - controlling with retry policy 236–238
 - listening to retries 238–239
 - RetryTemplate implementation 239–242
 - skipping instead of failing 227–234
 - configuring exceptions to be skipped 227–228
 - configuring SkipPolicy class for complete control 229–231
 - listening and logging skipped items 231–234
- Callback class 161, 174
- Callback interface 161
- case studies 8–11
 - dependencies for 445
 - import product use case 10–11
 - online store application 8–9
 - reason for building with batch jobs 9
 - reason for using batch processes 9–10
- Castor 6
- CastorMarshaller 136–137
- chaining item processors 219–222
- channel adapters, file-writing 330
- channel-based partitions, SPI using 399–402
- channels, remote chunking using 389
- character-separated fields, extracting 126–127
- checked exception 71
- chunk element 19, 43, 45, 56, 67–68
- chunk processing 7, 12–13, 33
- chunk size, choosing 19
- chunk-completion-policy attribute 64–65
- chunk-oriented architecture 193
- chunk-oriented steps
 - restarting on errors in middle of 247–250
 - stopping batch jobs from 114–116
 - transaction management in 254
- ChunkContext 291, 424–425
- ChunkListener 80, 254
- ChunkMessageChannelItemWriter class 390–392
- ChunkOrientedTasklet 45, 63, 388
- ChunkProcessor interface 388
- ChunkProcessorChunkHandler 393
- ChunkProvider interface 388
- chunks 213, 215, 220, 222, 251, 253, 263, 267
 - chunk-oriented steps 207–208
 - configuring 63–69
 - processing items in 194–195
- chunkWriter 392
- Classifier interface 190
- ClassifierCompositeItemWriter class 190–192
- classpath argument 93
- ClassPathXmlApplicationContext class 93
- clean lifecycle 440
- CleanStep 423
- CleanTasklet class 424–425
- CleanTaskletTest class 411, 424–425
- cleanup step 42
- CLIENT_ACKNOWLEDGE 264
- close method 82
- code, application 239–240
- ColumnRangePartitioner class 402–403
- comma-separated value. *See* CSV
- command line, launching batch jobs from
 - handling exit codes 94–97
 - overview 90–91
 - with job parameters 94
 - without job parameters 93

C

- cache-capacity attribute 64–65
- Callable interface 431

CommandLineJobRunner class 92–97
 commit convention, in Spring and Java Enterprise Edition 71
 commit-interval attribute 19, 64–65, 70, 85
 Commons database 456
 commons-io 445
 COMPLETED 302–304, 423–424, 436–437
 COMPLETED WITH SKIPS 281–284, 286
 CompletionPolicy interface 64
 component scanning 327
 components

- plugging in existing with ItemProcessorAdapter class 200–201
- transaction management in 253–255
 - chunk-oriented steps 254
 - listeners 254–255
 - tasklets 253

 CompositeItemProcessor 220–221, 418–419, 428–429
 CompositeItemProcessorTest class 411, 425, 428
 CompositeItemWriter 189
 compressing files 20
 computed fields, writing to files 165
 configuration files, Spring 447–449
 configuration inheritance 83–86
 configuration step 11
 configuration, leveraging SpEL for 25–26
 ConnectorServerFactoryBean class 369
 Console Server 39
 containers, embedding batch jobs scheduler in 91
 contextConfigLocation 457
 ContextLoaderListener class 104–105
 ContextLoaderListener object 109
 controllers

- configuring 327–328
- web, communicating job status from 341–344

 conversion field 168
 conversion values 168
 COUNT 281
 cron annotation attribute 102
 cron option, Spring scheduler 100
 cron, launching batch jobs from 98–99

- configuring 98–99
- jobs suited for use with 99

 CSV (comma-separated value) 14, 119
 curl tool, testing job submission with 328–329
 cursorRefPosition 144
 cursors, reading with 140–144, 148–150
 CustomEditorConfigurer class 128
 Customer object 186–187
 CustomScopeConfigurer class 76

D

DAOs (Data Access Objects) 72, 206, 348
 data partitioning, customizing 394, 402–404
 data replication 308–309
 data retrieval, using paging to manage 144–148, 150–151
 data-source attribute 74
 Database Connection Pool library. *See* DBCP
 database cursors, reading with 140–144
 database item writers

- configuring 18
- implementing 17–18

 database schema, for jobs 348–351
 databases

- configuring Spring Batch infrastructure in 37–41
 - accessing job metadata 39–40
 - job repository 38–41
 - reading XML and writing into 336
 - writing data to 179–183
 - with JDBC 179–181
 - with ORM 182–183

 DataFieldMaxValueIncrementer interface 454
 dataSource bean 456
 dataSource data source 74
 DataSource interface 18, 69
 DataSourceTransactionManager class 69–70, 75
 date job parameter 48
 Date object 94
 date parameter 94
 DBCP (Database Connection Pool) library 74
 DeadlockLoserDataAccessException class 67
 deciders, job execution 285–287
 decision logic, embedding

- in job execution decider 285–287
- in step execution listener 283–285

 declarative transactions, common pitfalls

- with 256–257

 decompress step 60–61
 decompressing, input files 20–23
 decompression step 11
 decompression tasklet, implementing 20–22
 decompressTasklet bean 77
 DecompressTasklet class 20–21
 DecompressTasklet tasklet 63
 default lifecycle 440
 DefaultFieldSet 417
 DefaultJobParametersValidator class 59–60, 421
 DefaultLineMapper class 15, 17, 120–121, 125–126
 DefaultRecordSeparatorPolicy 123, 132
 Defines property 442
 delegates 219
 delimited files, writing data to 166

- DelimitedLineAggregator class 162–166
 - DelimitedLineTokenizer class 16, 126–127
 - dependencies
 - adding to projects 443
 - dependencies for case study 445
 - logging dependencies 444–445
 - test dependencies 445
 - Spring Batch Admin, adding to web application 452–453
 - DependencyInjectTestExecutionListener class 426
 - destination files, name of 331
 - detecting, and filtering duplicates 269–271
 - detection, manual 267–272
 - development environments 439–449
 - Apache Maven 3 439–445
 - description of 440–441
 - installation 439–440
 - projects 441–445
 - STS for Eclipse 446–449
 - importing projects 446
 - Spring configuration file 447–449
 - directories, writing import files with Spring MVC framework in 329–331
 - configuring file-writing channel adapter 330
 - destination file name 331
 - DiscountService 239–242
 - DispatcherServlet class 108–109, 322, 343, 453
 - distributed services layer 366
 - distribution samples, building Spring Batch Admin console from 451–452
 - Document Object Model. *See* DOM
 - Document Type Definition configuration system. *See* DTD
 - doInStepScope 430–431
 - DOM (Document Object Model) 135
 - domain language, batch 33–35
 - how Spring Batch interacts with outside world 35
 - main components of 34–35
 - reasons for using 33
 - domain objects
 - creating with FlatFileItemReader class 14–15
 - JMS messages and 267–268
 - driving query pattern, implementing with item processor 204–208
 - configuring chunk-oriented step 207–208
 - executing with JDBC item reader 205–206
 - loading items 206–207
 - DTD (Document Type Definition) configuration system 55
 - duplicate message 263
 - duplicateImport method 325
 - DuplicateKeyException 325
 - DUPS_OK_ACKNOWLEDGE 264
- ## E
-
- EchoJobParametersTasklet 317
 - Eclipse development environment, STS for 446–449
 - importing projects 446
 - Spring configuration file 447–449
 - email messages
 - notifying of problems using 360–361
 - writing data to send 186–187
 - embedding
 - decision logic
 - in job execution decider 285–287
 - in step execution listener 283–285
 - Spring Batch Admin console 452–453
 - EmptyResultDataAccessException 180
 - encoding property 161, 174
 - enterprise integration 306–344
 - definition of 307–310
 - enterprise integration challenges 307–309
 - styles of enterprise integration 309–310
 - Spring Batch framework and 310–311
 - Spring Integration framework 312–320
 - combining with Spring Batch framework 313
 - project 312–313
 - quick-start 313–320
 - Spring MVC framework 320–331, 338–344
 - triggering jobs from file system events 332–338
 - converting file into job launch request 333–334
 - implementing import job 334–338
 - scanning input directory with Spring Integration framework 332–333
 - enterprise service bus. *See* ESB
 - entities, registered as streams 68
 - Entity 121, 149
 - EntityManagerFactory 183
 - equals() 236
 - errors, restarting on 242–250
 - completed steps 245–246
 - enabling between job executions 243–244
 - in middle of chunk-oriented step 247–250
 - limiting number of restarts 246
 - no restart option 244–245
 - ESB (enterprise service bus) 312
 - ETL (extract, transform, and load) process 10
 - events, triggering batch jobs by 91–92
 - Exception class 71, 229
 - ExceptionClassifierRetryPolicy 237
 - exceptions
 - configuring retryable 234–236
 - to be skipped, configuring 227–228
 - exclude element 67, 228, 235
 - excludeWriter 418
 - exclusion element 443

execute method 21, 45, 253, 274
 execution context, sharing data between steps
 using 289–296
 ExecutionContext 383, 402–403, 430–431
 ExecutionContextPromotionListener 294
 ExecutionListener interface 81
 executions 277–305
 job flow
 complex 278–279
 driving 280–287
 externalizing flow definitions 300–302
 sharing data between steps 287–300
 stopping job execution 302–305
 executor element, XML 90
 exit code mapper 95
 exit codes, launching batch jobs from command
 line 94–97
 exit status 282–287
 batch status vs. 282
 choosing between step execution listener and
 job execution decider 287
 embedding decision logic
 in job execution decider 285–287
 in step execution listener 283–285
 ExitCode 303, 365
 exitCode method 96
 ExitCodeMapper interface 96–97
 ExitStatus class 95–96, 282–286
 Extensible Markup Language. *See* XML
 externalizing job flow definitions 300–302
 extract, transform, and load process. *See* ETL
 ExtractorLineAggregator 160–163

F

FactoryBean 147
 fail elements 303
 FAILED 281–284, 302–304, 355–357
 failing
 skipping incorrect lines instead of 28–31
 skipping instead of 227–234
 configuring exceptions to be skipped
 227–228
 configuring SkipPolicy class for complete
 control 229–231
 listening and logging skipped items 231–234
 fetchSize 141, 143, 145, 149
 FieldExtractor class, writing data to files
 using 163–165
 FieldExtractor interface 163
 fields
 creating objects from 128–130
 extracting character-separated 126–127
 FieldSet class 126, 128–130, 163, 416–417
 FieldSet parameter 15
 FieldSetMapper 120–121, 128–129, 416–417
 FieldSetMapper interface 15, 125, 128–129
 fieldSetMappers 134
 fieldsUsedAsTargetMethodArguments 184–185
 file sets
 reading 138–139
 writing data to 178–179
 file system events, triggering jobs from 332–338
 converting file into job launch request 333–334
 implementing import job 334–338
 scanning input directory with Spring Integra-
 tion framework 332–333
 file-writing channel adapters 330
 File.listFiles method 249
 filename generation strategy 331
 filename property 396
 filename-generator attribute 330
 filename, to import from job configuration 77
 FileNameGenerator interface 331
 files
 converting into job launch request 333–334
 destination, name of 331
 import, writing in directory with Spring MVC
 framework 329–331
 input, decompressing with tasklet 20–23
 reasons for compressing 20
 writing data to 159–179
 adding header and footer 171–173
 computed fields 165
 delimited files 166
 file sets 178–179
 fixed-width files 166–169
 matching classes to LineAggregators 169–171
 using FieldExtractor class 163–165
 using FlatFileItemWriter class 161–163
 using LineAggregator Interface 163
 XML files 173–178
 FilesInDirectoryItemReader class 248–249
 FileUtils class 21
 filtering
 detecting duplicates and, with item
 processor 269–271
 items 208–211
 implementing processor for 209–211
 in item-processing phase 208–209
 FilteringProductItemProcessor 196–197
 findRunningJobExecutions method 353
 fine-grained scaling, with partitioning 394–404
 configuring 395–396
 SPI 397–404
 FINISHED constant 22
 fixed-delay option, Spring scheduler 100
 fixed-length fields 127
 fixed-rate option, Spring scheduler 100
 fixed-width files, writing data to 166–169

fixedDelay annotation attribute 102
 FixedLengthTokenizer 126–127
 fixtures 412
 flags field 167
 flat files 119–134
 DefaultLineMapper class 125–126
 fields
 creating objects from 128–130
 extracting character-separated 126–127
 FlatFileItemReader class 125
 reading
 configuration of FlatFileItemReader class
 16–17
 creating domain objects with FlatFileItem-
 Reader class 14–15
 flat file format 14
 implementing FieldSetMapper interface
 15–16
 Product domain class 14
 reading JSON 130–132
 reading records
 heterogonous 133–134
 multiline 132–133
 FlatFileFooterCallback 160–161, 171–172
 FlatFileHeaderCallback 160–161, 171, 173
 FlatFileItemReader bean 26
 FlatFileItemReader class 14, 29, 120–125
 configuration of 16–17
 creating domain objects with 14–15
 FlatFileItemWriter class 160–167, 170–173,
 418–419
 FlatFileParseException class 29–30, 228, 232–233
 FlowStep class 44
 footer, writing to files 171–173
 footerCallback property 161, 174
 Formatter class 166–168
 Formatter conversions 168
 FormatterLineAggregator class 162, 166–167,
 170–171, 173
 fragmentRootElementName 136–137
 functional testing 409, 432–437
 jobs 436–437
 steps 433–436
 using JobLauncherTestUtils 433

G

gateways
 configuring 327–328
 Spring Integration framework 326
 GeneratesJobMetaData class 39
 getCommaCount 132–133
 getExecutions 354, 371
 getExitCode() method 96
 getExitStatus 355–357

getFailureExceptions 356–357, 360
 getFailureExitDescriptions 357
 getJobExecution method 354
 getJobExecutions method 354
 getJobInstance method 354
 getJobInstances method 354
 getJobNames 353–355
 getNext method 59
 getObject 147
 getParameters 354, 420
 getProducts 152–153
 getSkipCount 357
 getStepExecution 353, 427–428, 430–431
 getStepExecutionSummaries 354, 358, 371
 getString 420
 getSummary 354, 358, 371
 global transactions 259–262
 local transactions vs. 260
 transaction managers and 260–262
 globalListener listener 86
 grid-computing frameworks 7
 gridSize 400–403

H

H2 database 454
 handlers, partition 398–399
 hashCode() 236
 HDFS (Hadoop Distributed File System) 7
 header, writing to files 171–173
 headerCallback property 161, 174
 heterogonous records 133–134
 Hibernate, SessionFactory interface 99
 hibernate.cfg.xml 150
 HibernateCursorItemReader 149–150
 HibernateItemWriter class 158, 182–183
 HibernatePagingItemReader 150–151
 HibernateTemplate 182
 hierarchies, job entities 57–58
 holder beans, sharing data between steps
 using 296–300
 horizontal scaling 374
 HTTP requests, launching batch jobs with
 105–109

I

iBATIS 6
 IbatisBatchItemWriter 183
 idempotency, handling duplicate messages
 with 272–273
 definition of idempotency 272
 idempotent operations in batch job 272–273
 IdToProductItemProcessor 207–208

- Implementation class 197
- import files, writing in directory with Spring MVC framework 329–331
 - configuring file-writing channel adapter 330
 - destination file name 331
- import jobs 334–338
 - configuring 337–338
 - mapping with job instance 334–336
 - reading XML and writing into database 336
- import product use case 10–11
- import products job, multiple runs of 47–50
- import-id attribute 331
- import-products-job.xml file 92
- import-products.jar file 93
- importId property 299
- importing projects 446
- ImportMetadata class 297
- ImportMetadataHolder class 297
- importProducts method 325
- importProductsJob 359, 364–365, 371, 386–387
- ImportProductsJobListener class 78–79
- importProductsStep-master 401
- ImportValidator 419–420, 429, 431
- ImportValidatorTest class 411–412, 419
- in-memory database 41
- in-memory job repositories 73
- include element 227–228, 233, 236, 239
- incrementer attribute 58–59
- incrementer object 59
- information portal 308
- infrastructure components 34
- infrastructure configuration file 25
- InitializingBean 152–153
- INPUT 420, 429–431, 434, 436
- input directory, scanning with Spring Integration framework 332–333
- input files, decompressing with tasklet 20–23
- input, services as 151–154
- inputResource job parameter 26
- INSERT 180–181, 187–188, 421–423
- insert statement, SQL 17
- install phase 440
- instance, and parameters 88–89
- instrumentation layer 366
- integration file 321
- integration testing 409, 425–432
 - using Spring Batch
 - StepScopeTestExecutionListener 427–432
 - for ItemProcessor 428–429
 - for ItemReader 429–432
 - using Spring TestContext Framework 426–427
- integration-test 440
- InterfaceBasedMBeanInfoAssembler class 368
- intValue method 96
- InventoryOrderWriter 268–269, 271
- IOUtils class 21
- isEndOfRecord 123, 132–133
- isolation attribute 70
- isolation-level-for-create attribute 74–75
- item listeners 78
- item processing 12
- item processors 194–197
 - chaining 219–222
 - configuring 195–197
 - detecting and filtering duplicates with 269–271
 - filtering
 - implementing 209–211
 - items in item-processing phase 208–209
 - implementing custom 199–200, 202–204
 - processing
 - in chunk-oriented step 194–195
 - use cases for 195
- item readers
 - JDBC 139–148
 - ORM 148–151
 - reading with database cursors and result sets 140–144
 - using paging to manage data retrieval 144–148
 - ORM
 - cursors 148–150
 - data retrieval 150–151
- item writers
 - custom 187–189
 - database
 - configuring 18
 - implementing 17–18
 - writing and tracking items in 268–269
- ItemJmsWriter 186
- ItemListenerSupport, testing using Mockito 418–419
- ItemPreparedStatementSetter class 180–181
- ItemPreparedStatementSetter interface 181
- itemPreparedStatementSetter property 180
- ItemProcessListener interface 80–81
- ItemProcessor class 201
- ItemProcessor interface 12, 63, 195, 197, 203
- ItemProcessorAdapter 197, 200–201, 204, 207
- ItemReader interface 151–152, 156, 382–383
- ItemReaderAdapter 152–153
- ItemReaderException 228
- ItemReadListener interface 80–81
- items 34
- ItemSkipListener 254
- ItemSqlParameterSourceProvider class 180, 421–422, 435
- itemSqlParameterSourceProvider property 180
- ItemStream interface 118–119, 248–250, 290, 383

ItemSupportListener 418
 ItemWriteListener interface 80–81
 itemWriter 393
 ItemWriter interface 17–18, 162, 174, 179, 188
 ItemWriterAdapter class 184–185
 ItemWriters 187, 192, 201, 204, 384, 421–423

J

JAR file 93, 441–442, 452–455
 java -version 439–440
 Java Architecture for XML Binding. *See* JAXB
 Java Archive file. *See* JAR file
 Java class 314–315, 333, 336
 Java database 445
 Java Enterprise Edition 71
 Java language, representation of job launch request 314–316
 Java Management Extension. *See* JMX
 Java Message Service. *See* JMS
 Java method 325
 Java Naming and Directory Interface. *See* JNDI
 Java object 316, 336, 342, 410
 Java Open Transaction Manager. *See* JOTM
 Java Persistence API. *See* JPA
 Java process 93
 Java Transaction API. *See* JTA
 java -v command 439
 Java Virtual Machine process. *See* JVM
 java.lang.Runnable 377
 java.util.concurrent package 377
 java.util.concurrent.Callable 431
 java.util.Formatter 162, 166, 169
 java.util.logging 232
 java.util.Map 130
 JavaScript Serialized Object Notation. *See* JSON
 JavaServer Faces. *See* JSF
 javax.sql.XAConnection 261
 JAXB (Java Architecture for XML Binding) 6, 173
 JConsole, monitoring with JMX 369–372
 JDBC (Java Database Connectivity) 5

- executing driving query pattern with 205–206
- initialize-database 348, 427
- item readers 139–148
 - ORM 148–151
 - reading with database cursors and result sets 140–144
 - using paging to manage data retrieval 144–148
- repository based on
 - tracking job submissions with 325–326
 - writing data to databases with 179–181

 JDBC database 454
 JDBC driver 140–141, 145
 jdbcBatchItemWriter class 179–181, 183, 385
 jdbcCursorItemReader class 140, 143–144, 206, 381, 384
 jdbcCursorItemReader sql 206
 jdbcPagingItemReader 145–146
 jdbcProductImportRepository 334
 jdbcTemplate 159, 188, 198–199, 206, 210
 JMS (Java Message Service) 35, 71, 118, 158, 257, 312, 376

- best effort pattern with 263–266
 - avoiding duplicate messages 266
 - avoiding losing messages with transaction synchronization 264–266
 - issues with message delivery 263–264
- messages, and domain objects 267–268
- reading data from 154–155
- writing data to 185–186

 JmsItemReader class 154–155
 JmsItemWriter class 185
 JmsTemplate 154–155, 185–186, 264–265
 JMX (Java Management Extension), monitoring with 366–372

- configuring 368–369
- using JConsole 369–372

 JNDI (Java Naming and Directory Interface) 456
 jndi-name 153–154
 Job bean 108
 job database schema, and monitoring 348–351
 job element 19, 56–59, 61, 73, 78
 job execution deciders

- embedding decision logic in 285–287
- enabling restarting on errors between 243–244
- step execution listeners and choosing between 287

 job exit statuses 95
 job explorer 112
 job files, deploying 455
 job flow

- complex, in online store application 278–279
- driving 280–287
 - choosing one path over another 280–281
 - exit status 282–287
 - externalizing flow definitions 300–302

 job instance 88
 Job Instances view 48
 Job interface 88
 job launcher 24, 34, 36, 112
 job listeners 78–80
 Job object 315
 job parameter 36, 88, 107
 job registry 112
 job repositories 24–25, 34, 36–37, 72–75, 112

- choosing 72–73
- configuring 38–39
- creating tables for 38
- implementations of 40–41

- job repositories (*continued*)
 - monitoring using 352–358
 - detecting problems using 355–358
 - JobExplorer interface 352–355
 - JobRepository interface 352
 - specifying parameters for 73–75
 - in-memory job repository 73
 - persistent job repository 73–75
- job-executions channel 317
- job-repository attribute 25, 58, 73
- job-repository element 39, 56, 73, 75
- JobExecution class 350, 362, 423–424, 433–434
- JobExecution getSkipCount 358
- JobExecution object 89
- jobExecutionContext object 77
- jobExecutionContext variable 296
- JobExecutionDecider interface 285, 411, 423–424
- JobExecutionListener interface 78–79
- JobExplorer interface 340, 352–356
- JobExplorer method 355
- JobExplorerFactoryBean class 353
- JobInstance class 350, 352–353, 355, 371, 423
- jobLauncher bean 456
- JobLauncher interface 35–36, 88, 92, 99
- jobLauncherTaskExecutor bean 456
- JobLauncherTestUtils class 433–437
- JobLaunchingMessageHandler class 315–316
- JobLaunchRequest class 314–315
- JobLaunchRequest object 315–317, 319
- JobOperator interface 58, 110, 354–355, 368, 372
- JobOperator method 355, 358
- JobParameters argument 88
- JobParameters class 35–36, 350
- jobParameters object 77
- jobParameters variable 26
- JobParametersBuilder class 88
- JobParametersExtractor bean 302
- JobParametersIncrementer interface 59
- JobParametersInvalidException 420
- JobParametersValidator class 411, 419–421
- JobParametersValidator interface 59
- JobRegistry interface 315
- jobRegistry property 107
- jobRepository bean 25, 58, 456
- JobRepository interface 36–38, 72, 352–353
- jobs 57–60
 - accessing metadata 39–40
 - anatomy of 41–50
 - modeling jobs with steps 42–45
 - running job instances and job executions 46–50
 - batch, idempotent operations in 272–273
 - configuring 58–60, 271–272
 - execution context of 289–290
 - using step execution context and promoting data to 293–295
 - using to share data between steps 290–293
 - writing in and reading using late binding 295–296
 - functional testing of 436–437
 - job entities hierarchy 57–58
 - launching and storing metadata 36–37
 - job launcher 36
 - job repository 36–37
 - launching with Spring Batch framework 313–320
 - and Spring Integration framework 317–320
 - Java representation of job launch request 314–316
 - simple job to display job parameters 316–317
 - monitoring with Spring MVC framework 341–344
 - REST, with Spring MVC framework 320–331, 338–344
 - status of, communicating from web controller 341–344
 - testing submission of
 - with curl tool 328–329
 - with RestTemplate class 329
 - tracking submissions with JDBC-based repository 325–326
 - triggering from file system events 332–338
 - converting file into job launch request 333–334
 - implementing import job 334–338
 - scanning input directory with Spring Integration framework 332–333
 - writing test for 26–28
 - JobStep class 44
 - JOTM (Java Open Transaction Manager) 261
 - JPA (Java Persistence API) 5, 182, 260
 - JpaItemWriter 183
 - JpaPagingReader class 150
 - JpaQueryProvider 151
 - JpaTransactionManager class 70
 - JSF (JavaServer Faces) 216
 - JSON (JavaScript Serialized Object Notation) 117, 124, 130–132
 - JSON file 121
 - JsonLineMapper class 130–131
 - JsonLineMapperWrapper class 131
 - JTA (Java Transaction API) 41, 261
 - JtaTransactionManager class 261
 - junit 445
 - JUnit framework, unit testing using 411–415
 - JUnitSampleTest 410
 - JVM (Java Virtual Machine) process 90

K

keyName 399

L

large object. *See* LOB

late binding 76

writing in job execution context and reading 295–296

writing to holder bean and using to late binding 299–300

launch method 101

LaunchDatabaseAndConsole class 39, 50

LaunchDatabaseAndConsole program 47

LaunchImportProductsJob class 47, 49

launchJob 433, 437

LaunchSpringBatchAdmin program 47

lib directory 93

lifecycles, of job instances and job executions 47

LimitCheckingItemSkipPolicy 229–230

LineAggregator Interface

matching classes to 169–171

writing data to files using 163

LineAggregator interface 160, 162–163, 166

lineAggregator property 161

LineCallbackHandler 120–122

LineMapper interface 14–15, 120–121, 123, 125

lineSeparator property 161

linesToSkip 16, 122–123

LineTokenizer 120–121, 125–128

LinkedList 415–416

List<String> 353–354, 357, 415–416

ListDirectoryItemReader 155–156

listener-container 392–393

listeners 78–83, 223, 232–233, 239, 250

job 78–80

monitoring with 358–362

implementing 358–360

notifying using emails 360–361

notifying using Spring messaging 361–362

repeat and retry 82–83

step execution, embedding decision logic in 283–285

steps 80–82

stopping from chunk-oriented step 114–115

transaction management in 254–255

listeners element 78, 82, 86

listening

to retries 238–239

to skipped items 231–234

loading items 206–207

LOB (large object) 74, 353

lob-handler attribute 74

local partitioning 396

local scaling 374, 376–377

local transactions, vs. global transactions 260

logback-test.xml file 444

logging dependencies 444–445

logging, skipped items 231–234, 444

M

M2_HOME variable 440

M2_HOME/bin 440

machine implementations

master 390–392

slave 392–394

MailSender 360–361

main method 92–93

manual detection, handling duplicate messages with 267–272

configuring job for 271–272

detecting and filtering duplicates with item processor 269–271

JMS messages and domain objects 267–268

writing and tracking items in item writer 268–269

Map 130–131

mapFieldSet method 16, 128–129, 416–417

MapJobRepositoryFactory class 73

MapJobRepositoryFactoryBean class 73

mapLine 124, 131

mapping, import jobs with job instance 334–336

mapRow 142

Marshaller class 136, 175

Marshaller interface 179

marshaller property 174

MarshallingHttpMessageConverter 343

master machine implementation 390–392

Maven command 440–441

Maven file 441

maven.compiler 442–443

max-varchar-length attribute 74

maxRows 141, 143

maxValue 404

MBean 366–367, 369–370

MBeanExporter class 368–369

Message object 333

message-based solutions 4

message-oriented middleware. *See* MOM

MessageBuilder class 319

MessageChannelPartitionHandler class 400–401

MessageListenerContainers 154

messages

avoiding duplicate 266

avoiding losing with transaction

synchronization 264–266

messages (*continued*)
 handling duplicate
 with idempotency 272–273
 with manual detection 267–272
 issues with delivery of 263–264
 JMS, and domain objects 267–268
 messagingOperations 400–401
 META-INF/persistence.xml 183
 META-INF/spring/batch/jobs directory 455
 metadata, job
 accessing in repository 338–340
 launching and storing 36–37
 MetadataInstanceFactory class 423–424, 428
 metamodels 202
 method attribute 102
 method process, ItemProcessor interface 12
 MimeMessage 186
 minValue 404
 MobilePhoneProduct 134, 169–170
 mobileProductLineAggregator 170–171
 Mockito, unit testing using 415–423
 for FieldSetMapper 416–417
 for ItemListenerSupport 418–419
 for ItemWriter 421–423
 for JobParametersValidator 419–421
 MockitoSampleTest 410
 Model-View-Controller. *See* MVC
 modeling, jobs 42–45
 MOM (message-oriented middleware)
 185, 257, 389
 monitoring 345–372
 and job database schema 348–351
 overview 346–348
 using job repository 352–358
 detecting problems using 355–358
 JobExplorer interface 352–355
 JobRepository interface 352
 with JMX 366–372
 configuring 368–369
 using JConsole 369–372
 with listeners 358–362
 implementing 358–360
 notifying using emails 360–361
 notifying using Spring messaging 361–362
 with Spring Batch Admin 362–366
 detecting problems using 365–366
 features of 363–365
 MonitoringExecutionListener class 359
 MonitoringNotifier 360, 362
 moveProcessedFiles 386–387
 multiline records 132–133
 MultiResourceItemReader 138–139, 249
 MultiResourceItemWriter class 178–179
 MultiResourcePartitioner class 398–399
 MultiResourceReader class 138

multithreaded step pattern 404
 multithreaded step strategy 376
 multithreaded steps 378–385
 configuring 379–381
 issues with 381–385
 process indicator pattern 383–385
 thread-safe item reader 382–383
 MVC (Model-View-Controller) 216
 mvn clean command 441
 mvn clean install -P bootstrap command 441
 mvn clean install command 441
 mvn clean package command 441
 mvn command 440
 mvn test command 441
 mvn -version command 440

N

name=value syntax 94
 namespaces 19, 54–56
 needsToBeFiltered 196, 210
 NEXT 423–424
 next attribute 23, 43, 61
 NextDecider 423–424
 NextDeciderTest class 411, 423
 no restart option 244–245
 no-rollback-exception-classes 71, 259
 NonTransientFlatFileException 228
 NonTransientResourceException 228, 248–249
 NOOP (NO OPERATION) 73
 null 152, 156, 253, 267, 270
 NullPointerException 429

O

objectName 369
 onError method 83, 238
 online store application 8–9, 278–279
 onProcessError method 81
 onReadError method 81
 onSkipInProcess method 81
 onSkipInRead method 81
 onSkipInWrite method 81
 onWriteError method 81
 open method 83
 Open Services Gateway initiative. *See* OSGi
 OPERATION 189–190
 operator, stopping batch jobs for 110–113
 configuring job operator 110–112
 invoking job operator 110
 understanding stop message 112–113
 optionalKeys property 60
 Oracle database 262
 Order object 268

- OrderItem 267–269
 - org.slf4j 444
 - org.springframework 444–445
 - org.springframework.batch.core package 38
 - org.springframework.validation 216
 - ORM (Object Relational Mapping) 157, 204
 - item readers 148–151
 - cursors 148–150
 - data retrieval 150–151
 - writing data to databases with 182–183
 - OS command 451
 - OSGi (Open Services Gateway initiative) 35
 - outbound-channel-adapter 391
 - overwriteOutput property 174
 - OXM class 137
- P**
-
- pageSize 146, 151
 - paging, using to manage data retrieval 144–148, 150–151
 - PagingQueryProvider 146
 - parallel step pattern 404
 - parallel step strategy 376
 - parallelizing, processing 385–387
 - parallelJobs 379
 - parameters
 - and instance, in Spring Batch launcher API 88–89
 - job, simple job to display 316–317
 - launching batch jobs from command line
 - with job parameters 94
 - without job parameters 93
 - parameters.getParameters().containsKey(key) 420
 - parent attribute 58, 61, 84–85
 - parentStep parent step 85
 - ParseException 228, 248–249
 - partition element 395
 - partition handlers, SPI using default 398–399
 - partitioner 397
 - Partitioner interface 394, 397, 402
 - PartitionHandler interface 394, 397
 - partitioning step pattern 404
 - partitioning step strategy 376
 - partitioning, fine-grained scaling with 394–404
 - configuring 395–396
 - SPI 397–404
 - partitions, channel-based 399–402
 - PartitionStep class 44
 - PartnerItemIdProcessor 199–200, 221
 - PartnerIdMapper 198–201, 221
 - PartnerProduct 201–203, 220
 - PartnerProductIdProcessor 202–204, 221
 - PartnerProductMapper 202–204
 - PassThroughFieldExtractor class 163–164
 - PassThroughLineAggregator class 160, 162–163, 435
 - PassThroughLineMapper class 124
 - PATH environment variable 440
 - paths, choosing one over another 280–281
 - PatternMatchingCompositeLineMapper class 124, 133–134
 - patterns
 - process indicator 383–385
 - scaling, comparing 404–406
 - transaction management 259–273
 - best effort pattern with JMS 263–266
 - global transactions 259–262
 - handling duplicate messages 267–273
 - shared resource transaction pattern 262–263
 - PENDING status 340
 - performance, enhancing by scaling 374–375
 - persistent job repositories 41, 73–75
 - PessimisticLockingFailureException 235
 - plain old Java objects. *See* POJO
 - PlatformTransactionManager interface 63, 70, 73, 261
 - plugging, into batch metadata 454–455
 - POJO (plain old Java objects) 80, 316, 378, 410
 - policies, retry 236–238
 - polling file system to trigger jobs task 311
 - POM (Project Object Model) 439
 - POM file 452
 - pom.xml file 439, 442
 - PositivePriceValidator 413–414, 428–429
 - PositivePriceValidatorTest class 411
 - PostgresPagingQueryProvider class 147
 - PRB 133–134
 - precision field 168
 - PreparedStatement 139, 142–143, 181
 - preparedStatementSetter 141–143, 205–206
 - prepareInputFileFlow bean 301
 - PriceMandatoryValidator 413–414, 428
 - PriceMandatoryValidatorTest class 411
 - primers, on transactions 252
 - PRM 133–134
 - procedureName 143–144
 - process indicator pattern 383–385
 - process method 209
 - process scheduling 97
 - processing
 - items
 - in chunk-oriented step 194–195
 - use cases for 195
 - parallelizing 385–387
 - processing data 193–222
 - filtering items 208–211
 - implementing processor for 209–211
 - in item-processing phase 208–209

- processing data (*continued*)
 - item processors 194–197
 - chaining 219–222
 - configuring 195–197
 - implementations 197
 - processing 194–195
 - transforming items 197–208
 - implementing driving query pattern with item processor 204–208
 - read 198–204
 - validating items 208, 211–219
 - with Bean Validation standard 216–219
 - with custom validator 213–214
 - with Valang validator from Spring Modules project 214–215
 - with ValidatingItemProcessor class 212
 - processing phase 193
 - processor attribute 63, 65, 67
 - processor element 66
 - processor-transactional 258
 - processors
 - item, detecting and filtering duplicates
 - with 269–271
 - transactional, and transactional reader 257–258
 - Product class 149–150, 182, 190, 217, 336
 - product data 11–20
 - database item writer
 - configuring 18
 - implementing 17–18
 - read-write step
 - anatomy of 11–13
 - configuring 18–20
 - reading flat file 14–17
 - configuration of FlatFileItemReader class 16–17
 - creating domain objects with FlatFileItemReader class 14–15
 - format of 14
 - implementing FieldSetMapper interface for Product objects 15–16
 - Product domain class 14
 - Product domain objects 14
 - Product ItemListener 418
 - Product method 184
 - Product objects, implementing FieldSetMapper interface for 15–16
 - Product table 383
 - product-imports-as-string channel 328
 - Product.class 134
 - ProductFieldExtractor 165
 - ProductFieldSetMapper class 15, 17, 129, 416–417
 - ProductFieldSetMapperTest class 411, 417
 - ProductFooterCallback class 172
 - ProductFooterStaxCallback class 175–177
 - ProductHeaderStaxCallback 175, 177
 - ProductImport class 338–339, 342
 - ProductImportFileNameGenerator class 331
 - ProductImportGateway interface 326
 - ProductImportToJobLaunchRequestHandler class 333
 - ProductImportUtils class 331
 - ProductItemListener class 411, 418–419
 - ProductItemListenerTest class 418
 - ProductItemWriter 408, 421–422
 - ProductItemWriterMockTest class 411
 - ProductJdbcItemWriter class 17–18
 - ProductLineAggregator 171
 - productMarshaller 137
 - ProductRouterClassifier 190–192
 - ProductRowMapper 141–142, 206–207
 - products_corrupted.zip file 49
 - products.zip file 49
 - ProductService class 184–185
 - ProductServiceAdapter class 152–153
 - ProductsLineAggregator class 170
 - productsStep 433–434
 - ProductStepTest class 432
 - productWriter 185
 - Project Object Model. *See* POM
 - projects 441–442
 - adding dependencies to 442–445
 - for case study 445
 - logging 444–445
 - test 445
 - importing 446
 - Spring Integration framework 312–313
 - propagation attribute 71
 - PROPAGATION_NEVER 253
 - property element 55
 - PropertyEditor interface 128
 - PropertyExtractingDelegatingItemWriter class 184–185
 - PropertyPlaceholderConfigurer class 76
 - prototype scope 76
 - prototypeBeanName 129–130
 - PUT requests 322
- ## Q
-
- Quartz 103
 - queryProvider 146, 149, 151
 - queryString 149–151
 - quick-start, launching Spring Batch job 313–320
 - Java representation of job launch request 314–316
 - simple job to display job parameters 316–317
 - with Spring Integration framework 317–320

R

-
- Range class 127–128
 - RangeArrayPropertyEditor class 127–128
 - RDBMS (relational database management system)
 - setup 25
 - read items
 - changing state of 198–201
 - implementing custom item processor 199–200
 - plugging in existing component with Item-ProcessorAdapter class 200–201
 - use case 198–199
 - producing new objects from 201–204
 - implementing custom item processor 202–204
 - use case 201–202
 - READ_COMMITTED 70, 255
 - READ_UNCOMMITTED 70, 255–256
 - read-write step
 - anatomy of 11–13
 - configuring 18–20
 - reader attribute 63, 65, 67
 - reader element 66
 - reader-transactional-queue 64, 72, 257–258, 265, 271
 - readers
 - implementing custom 155
 - thread-safe item 382–383
 - transactional, and transactional processor 257–258
 - ReaderWithListenerTest class 425
 - ReaderWithStepScopeTestUtilsTest class 425, 431
 - reading and writing step 11
 - reading data 117–156
 - concepts related to 118–119
 - file sets 138–139
 - flat files 119–134
 - DefaultLineMapper class 125–126
 - fields 126–130
 - FlatFileItemReader class 125
 - reading JSON 130–132
 - reading records 132–134
 - from JMS 154–155
 - from relational databases 139–151
 - implementing custom readers 155
 - services as input 151–154
 - XML files 135–137
 - reading file 138
 - readWrite step 60–61
 - readWriteProducts 23, 395–396, 399
 - readWriteProductsMultiThreadedStep 379, 381
 - readWriteProductsStep 379
 - ready-to-use components 5, 11
 - RecordSeparatorPolicy interface 120, 123, 132
 - ref attribute 60, 62–63, 102
 - refCursorPosition 143–144
 - relational database management system setup. *See* RDBMS
 - relational databases, reading data from 139–151
 - reliability, Spring Batch 5
 - remote chunking 387–394
 - description of 387–389
 - with Spring Integration framework 389–394
 - master machine implementation 390–392
 - remote chunking using channels 389
 - slave machine implementation 392–394
 - remote chunking pattern 404
 - remote chunking strategy 376
 - Remote Method Invocation. *See* RMI
 - remote partitioning 396
 - remote procedure 309–310
 - remote scaling 376–377
 - remote-slsb 153–154
 - RemoteChunkHandlerFactoryBean class 392
 - repeat and retry listeners 82–83
 - REPEATABLE_READ level 70
 - RepeatStatus 22, 424–425
 - RepeatStatus.CONTINUABLE 253
 - RepeatStatus.FINISHED 253
 - replyChannel 400–401
 - repositories
 - accessing job metadata in 338–340
 - configuring 327–328
 - JDBC-based, tracking job submissions with 325–326
 - REQUIRED 255–256
 - requiredKeys property 60
 - requirements, for Spring TestContext Framework 426–427
 - REQUIRES_NEW 254, 257
 - RESOURCE 183, 420, 429–431, 436
 - Resource interface 137
 - resource property 16, 26, 161, 174
 - ResourceAwareItemReaderItemStream
 - interface 139
 - ResourceAwareItemWriterItemStream 162, 179
 - ResourcelessTransactionManager class 73
 - resources 137–138, 154
 - ResourceServlet 453
 - ResourceSuffixCreator interface 179
 - REST (Representational State Transfer)
 - 106, 321, 443
 - monitoring jobs with 338–344
 - submitting jobs with 320–331
 - restart feature 226–227, 289, 303
 - restartable attribute 58, 244–245, 247–250
 - restarting, on error 242–250
 - completed steps 245–246
 - enabling between job executions 243–244

restarting, on error (*continued*)
 in middle of chunk-oriented step 247–250
 limiting number of restarts 246
 no restart option 244–245

RestTemplate class 329, 331

result sets, reading with 140–144

ResultSet 140–142, 144, 146, 381

retrieval, of data 144–148, 150–151

retry feature
 combining with skip feature 236
 in action with skip and restart features 226–227

retry policies, controlling retrying on error
 with 236–238

retry-limit attribute 64–65, 234–237, 239

retry-listeners element 66, 239

retry-policy attribute 64

retry-policy element 66

retryable-exception-classes 67, 234–237, 239

retrying, on error 234–242
 configuring retryable exceptions 234–236
 controlling with retry policy 236–238
 listening to retries 238–239
 RetryTemplate implementation 239–242

RetryListener interface 238

RetryListenerSupport 238

RetryOperations interface 239

RetryOperations property 240

RetryOperationsInterceptor 240–241

RetryPolicy interface 64, 237, 240

RetryTemplate implementation
 retrying in application code with 239–240
 retrying transparently with AOP and 240–242

RMI (Remote Method Invocation) 312, 369

RmiRegistryFactoryBean class 369

robustness, Spring Batch 5

rollback convention 71

rolling back, choosing whether to 258–259

root-database-context.xml file 39

rootElementAttributes property 174

rootTagName property 174–175, 177

router classifier 190

routerDelegate 191–192

RowMapper 139, 141–142, 146, 206–207

run method 36, 88–89, 92

Runnable interface 377

Runner class 426

RuntimeException class 71

S

samples, building Spring Batch Admin console
 from 451–452

SampleTask class 377

SamungStatementSetter 142–143

saveOrUpdate 182–183

saveState property 161, 174, 383–384

sbia-servlet.xml file 108

scale out approach 374

scale up approach 374

scaling 373–406
 comparing patterns 404–406
 enhancing performance by 374–375
 local and remote 376–377
 model of 375–378
 local and remote scaling 376–377
 task executor abstraction 377–378

multithreaded steps 378–385
 configuring 379–381
 issues with 381–385

parallelizing processing, configuring parallel
 steps 385–387

remote chunking 387–394
 description of 387–389
 with Spring Integration framework 389–394

scaling strategies 7–8

scaling, fine-grained 394–404

schedulers 6

scheduling jobs 115

scheduling.xml file 105

schema.sql 348

schema-[database].sql naming convention, SQL
 scripts 38

schema-drop.sql 349

scope attribute 76

SCP (Secure Copy) 20, 137

Secure Shell. *See* SSH

SELECT statement 140–143, 147–148, 434–435

SERIALIZABLE level 71, 75, 255

server connector, JMX 369

service activator 317

Service Provider Interface. *See* SPI

service-activator 319

service-interface attribute 328

service=batch,bean=jobOperator 368–369

services, as input 151–154

serviceUrl 369

servlet filters 453

servlet listener 453

[servlet-name]-servlet.xml file 108

SessionFactory interface 99, 149–150, 182

sessionTransacted 264–265, 271

setExcludeWriter 418–419

setResource 162

Shared database 309

shared resource transaction pattern 262–263

Short Message Service. *See* SMS

shouldDeleteIfExists property 162

shouldDeleteIfEmpty property 162

SimpleAsyncTaskExecutor implementation 378

SimpleChunkProcessor class 388, 393

- SimpleJdbcTemplate 408, 421–423, 434–436
- SimpleJobLauncher class 36, 38
- SimpleJobRepository class 72
- SimpleJvmExitCodeMapper class 95
- SimpleMailMessage class 186–187, 360–361
- SimpleMailMessageItemWriter 186–187
- SimpleMessageApplicationEvent class 362
- SimplePartitioner class 397
- SimpleRetryPolicy 237–238, 240, 242
- SimpleStepExecutionSplitter class 397, 402
- SingleColumnRowMapper 205–206
- singleton scope 76
- site lifecycle 440
- skip feature
 - combining retry feature with 236
 - in action with retry and restart features 226–227
- skip-limit attribute 30, 64–65, 228
- skip-limit pair 230
- skip-policy attribute 64, 229
- skip-policy element 66
- SkipListener interface 80–81, 231–232
- SkipListenerSupport 232
- skippable-exception-classes 227–230, 233, 236
- skipping 209
 - incorrect lines 28–31
 - instead of failing 227–234
 - configuring exceptions to be skipped 227–228
 - configuring SkipPolicy class for complete control 229–231
 - listening and logging skipped items 231–234
- SkipPolicy class, configuring for complete control 229–231
- SkipPolicy interface 64
- slave machine implementation 392–394
- SMS (Short Message Service) 347
- sp_product 144
- specificListener listener 86
- SpEL (Spring Expression Language) 25–26, 76–77, 427
- SPI (Service Provider Interface) 395, 397–404
 - customizing data partitioning 402–404
 - using channel-based partitions 399–402
 - using default partition handler 398–399
- split element 385
- Spring Batch 3–31
 - batch applications 4–5
 - case study 8–11
 - import product use case 10–11
 - online store application 8–9
 - reason for building with batch jobs 9
 - reason for using batch processes 9–10
 - decompressing input file with tasklet 20–23
 - implementing decompression tasklet 20–22
 - reasons for compressing files 20
 - description of 5–8
 - robustness and reliability 6–7
 - scaling strategies 7–8
 - embedding in web applications 104–105
 - infrastructure of 36–41
 - configuring in database 37–41
 - launching jobs and storing job metadata 36–37
 - interaction with outside world 35
 - product data 11–20
 - database item writer 17–18
 - read-write step 11–20
 - reading flat file 14–17
 - skipping incorrect lines instead of failing 28–31
 - testing batch process 23–28
 - leveraging SpEL for configuration 25–26
 - setting up test infrastructure 23–25
 - writing test for job 26–28
- Spring Batch Admin 40, 47, 49, 450–457
 - building from distribution sample 451–452
 - configuring 454–457
 - deploying job files 455
 - overriding Spring Batch Admin configuration 455–457
 - plugging into batch metadata 454–455
 - downloading 450–451
 - embedding in web application 452–453
 - adding Spring Batch Admin dependencies 452–453
 - declaring Spring Batch Admin console in web.xml file 453
 - monitoring with 362–366
 - detecting problems using 365–366
 - features of 363–365
- Spring Batch database 348
- Spring Batch file 313
- Spring Batch framework
 - combining with Spring Integration framework 313
 - enterprise integration and 310–311
 - launching jobs with 313–320
 - Java representation of job launch request 314–316
 - simple job to display job parameters 316–317
 - using Spring Integration framework 317–320
- Spring Batch interface 200
- Spring Batch JAR file 38
- Spring Batch launcher API 88–89, 92
- Spring Batch namespace 56
- Spring Batch StepScopeTestExecutionListener,
 - integration testing using 427–432
 - for ItemProcessor 428–429
 - for ItemReader 429–432
- Spring Batch Test module 423–425
 - for JobExecutionDecider 423–424
 - for tasklet 424–425

- Spring Batch type 315
- Spring Batch XML 19, 42, 44–45
- Spring class 329
- Spring configuration file 447–449
- Spring default namespace 56
- Spring Expression Language. *See* SpEL
- Spring file 447
- Spring Framework foundations, Spring Batch 5
- Spring Integration framework 312–320
 - gateway 326
 - project 312–313
 - quick-start, launching Spring Batch job 313–320
 - remote chunking with 389–394
 - master machine implementation 390–392
 - slave machine implementation 392–394
 - using channels 389
 - scanning input directory with 332–333
 - sending job launch request with 318–320
 - with Spring Batch framework 313, 317–318
- Spring interface 342
- Spring messaging, notifying of problems
 - using 361–362
- Spring Modules project, Valang validator
 - from 214–215
- Spring MVC controller, launching batch jobs
 - using 106–109
- Spring MVC framework
 - and REST
 - monitoring jobs with 338–344
 - submitting jobs with 320–321, 329–331
 - deploying in web application 321–322
- Spring OXM 6
- Spring scheduler, launching batch jobs from
 - 99–103
 - options for 100
 - setting up 100–101
 - with annotations 102–103
 - with XML 101–102
- Spring TestContext Framework 27, 426–427
- Spring transaction manager 70
- Spring XML Beans 55
- Spring XML schema-based configuration 55
- spring.batch.version 442–443
- SpringJUnit4ClassRunner.class 426–428, 430–431, 435–436
- SpringSource 5
- SpringSource Tool Suite. *See* STS
- SpringValidator class 215–216
- SQL database 426
- SQL parameter 180–181
- SQL scripts 38
- SQL statement 141–143, 147–148, 158–159, 179–181
- SqlPagingQueryProviderFactoryBean 146–147
- SqlParameterSource 421–423, 435
- src/main/resources directory 454
- SSH (Secure Shell) 20
- Start Date attribute 48
- start-limit attribute 62, 244, 246
- startNextInstance method 58
- stateful objects 381
- StatelessSession 149
- Statement parameter 149
- states, of read items 198–201
- statisticStep 434–436
- StatisticStepTest class 432
- Status field 365
- StAX (Streaming API for XML) 6, 135, 174
- StaxEventItemReader class 136–137
- StaxEventItemWriter class 173–174, 176–179
- StaxWriterCallback 174–176
- STDOUT class 444
- stdout-channel-adapter 329
- step element 19, 25–26, 43, 56, 60–62
- step execution context
 - sharing data between steps using 289–290
 - using and promoting data to job execution context 293–295
- step execution handling 394
- step execution listeners
 - and job execution deciders, choosing between 287
 - embedding decision logic in 283–285
- Step interface 44
- step scope 75–77, 427
- stepChunk 392
- StepExecution class 175–177, 352–353, 427–431, 434, 436
- StepExecution object 113–114
- StepExecution parameter 80
- StepExecution.setTerminateOnly() method 113
- stepExecutionContext object 77
- StepExecutionListener interface 80
- StepExecutionListenerSupport 172–173, 175–176
- StepExecutionRequestHandler 400–402
- StepExecutionSplitter interface 397, 402
- StepListener interface 80, 408, 411, 418
- stepName 400–401
- steps 60–72
 - chunk-oriented
 - configuring 207–208
 - processing items in 194–195
 - restarting on errors in middle of 247–250
 - transaction management in 254
 - completed, restarting on errors 245–246
 - configuring 60–61
 - chunks 63–69
 - tasklets 61–63
 - transactions 69–72
 - functional testing of 433–436

- steps (*continued*)
 - modeling jobs with 42–45
 - configuring 43–44
 - processing with TaskletStep implementation 44–45
 - parallel, configuring 386–387
 - sharing data between 287–300
 - using execution context 289–296
 - using holder beans 296, 300
 - using job execution context to share data between 290–293
 - StepScope class 75–76
 - StepScopeTestExecutionListener class 426–430
 - StepScopeTestUtils method 429–432
 - stopConditionsMet method 115
 - STOPPED 282, 302–303
 - stopping batch jobs 109–116
 - for application developer 113–116
 - from chunk-oriented step 114–116
 - from tasklet 113–114
 - for operator 110–113
 - configuring job operator 110–112
 - invoking job operator 110
 - understanding stop message 112–113
 - storage technologies, Spring Batch 5–6
 - StoredProcedureItemReader 143–144
 - Streaming API for XML. *See* StAX
 - streams element 67–68
 - String argument 96
 - String column 403
 - String property 292, 299
 - String table 403
 - String type 15
 - String-typed parameters 94
 - STS (SpringSource Tool Suite), for Eclipse development environment 446–449
 - importing projects 446
 - Spring configuration file 447–449
 - STS-Maven integration plug-in 446
 - synchronization, of transactions 264–266
 - synchronizer, job repository as 75
 - SynchronizingItemReader class 382–384
 - synchronous batch job launches, vs. asynchronous batch job launches 89–90
 - SyncTaskExecutor 387
 - system exit codes 95
 - system testing 409
 - System.printf() method 168
- T**
-
- table-prefix attribute 74
 - tables, creating for job repository 38
 - targetDirectory job parameter 26
 - targetFile job parameter 26
 - targetMethod 153, 184
 - targetObject 153, 184
 - targetSize 403–404
 - task 102–103
 - annotation-driven element 103
 - scheduled element 102
 - scheduled-tasks element 102
 - task executor abstraction 377–378
 - task namespace 90, 100–101
 - task-executor attribute 380
 - task:executor element 457
 - TaskExecutor interface 90, 377–378
 - TaskExecutorPartitionHandler class 396–399
 - tasklet code 291
 - tasklet element 19, 23, 62–63, 380–381
 - Tasklet interface 20–22, 44–45, 62–63
 - tasklet tag 43
 - tasklets
 - configuring 61–63
 - decompression
 - implementing 20–22
 - of input files 20–23
 - stopping batch jobs from 113–114
 - testing, using Spring Batch Test module 424–425
 - transaction management in 253
 - TaskletStep class 44
 - TaskletStep implementation, processing with 44–45
 - TDD (test-driven development) 409
 - Test class 410–411, 424–425, 429, 432
 - test dependencies 445
 - Test method 413, 430
 - test-driven development. *See* TDD
 - testAfterProcessResult 419
 - TestCase class 412
 - testEmptyProductFailure 429
 - TestExecutionListeners 426–428, 430
 - testing 407–437
 - batch process 23–28
 - leveraging SpEL for configuration 25–26
 - setting up test infrastructure 23–25
 - writing test for job 26–28
 - benefits of 410
 - defined 408–409
 - functional testing 432–437
 - jobs 436–437
 - steps 433–436
 - using JobLauncherTestUtils 433
 - integration testing 425–432
 - using Spring Batch StepScopeTestExecutionListener 427–432
 - using Spring TestContext Framework 426–427

testing (*continued*)
 types of 409–410
 unit testing 410–425
 using JUnit framework 411–415
 using Mockito 415–423
 using Spring Batch Test module 423–425
 for JobExecutionDecider 423–424
 for tasklet 424–425
 testInsertProduct 422–423
 testIntegration 434
 testJobParameters 420
 testMapFieldSetClassic 417
 testMapFieldSetMock 417
 testMissingJobParameters 420
 testNegativePrice 414–415
 testNegativePriceFailure 428–429
 testPositivePrice 414
 testUpdateProduct 423
 testValidateEmptyJobParameters 420
 testValidProduct 413–414
 testZeroPriceFailure 429
 ThreadPoolTaskExecutor 378
 thread safety 379, 382
 thread-local-channel 391
 thread-safe item reader 382–383
 Thread.currentThread().isInterrupted()
 method 112
 ThreadLocal class 382
 ThreadPoolExecutor class 378
 ThreadPoolTaskExecutor implementation 378
 throttle-limit attribute 381
 throws clause 71
 timeout attribute 71
 toArray() method 163
 tokenizers 126, 134
 toString() method 160, 162–164, 168, 360
 tracking, writing items and 268–269
 transaction attributes 62–63, 69, 74, 255–256
 transactions 260–262
 transaction synchronization, avoiding losing mes-
 sages with 264–266
 transactional processors, and transactional
 reader 257–258
 transactional property 162, 175
 transactional reader, and transactional
 processor 257–258
 TransactionDefinition class 71
 transactionManager bean 456
 transactionManager transaction manager 75
 transactionManager-ref attribute 73
 transactions 251–274
 configuration of 255–259
 choosing whether to roll back 258–259
 common pitfalls with declarative
 transactions 256–257

 transaction attributes 255–256
 transactional reader and processor 257–258
 configuring 69–72
 management in components 253–255
 chunk-oriented steps 254
 listeners 254–255
 tasklets 253
 management patterns 259–273
 best effort pattern with JMS 263–266
 global transactions 259–262
 handling duplicate messages 267–273
 shared resource transaction pattern 262–263
 primer on 252
 transforming, items 197–208
 implementing driving query pattern with item
 processor 204–208
 read 198–204
 TransientDataAccessException 235
 transparently retrying, with AOP and Retry-
 Template implementation 240–242
 try-catch block 71
 types, of testing 409–410

U

unchecked exception 71
 unit testing 410–425
 using JUnit framework 411–415
 using Mockito 415–423
 for FieldSetMapper 416–417
 for ItemListenerSupport 418–419
 for ItemWriter 421–423
 for JobParametersValidator 419–421
 unmarshaller 135–137
 UPDATE statement 188, 421, 423
 update statement 226, 232, 234, 249
 update_timestamp 205
 use cases
 for changing state of read items 198–199
 for processing items 195
 for producing new objects from read items
 201–202
 useStateless 151

V

valang property 215
 ValangValidator 215–216
 validate method 59
 validate phase 440
 validating items 208, 211–219
 with Bean Validation standard 216–219
 with custom validator 213–214
 with Valang validator from Spring Modules
 project 214–215
 with ValidatingItemProcessor class 212

ValidatingItemProcessor class 212, 215, 219, 222
 ValidationException class 71, 212–215,
 218, 414–415
 validator element 60
 Validator interface 212, 216
 Validator object 197
 values() method 163
 verify phase 440
 verifyNoMoreInteractions 415, 417, 420, 422
 version property 175
 vertical scaling 374

W

WAR file 441, 451
 web applications
 deploying Spring MVC framework in 321–322
 embedding Spring Batch Admin console
 in 452–453
 adding Spring Batch Admin
 dependencies 452–453
 declaring Spring Batch Admin console in
 web.xml file 453
 launching batch jobs from 103–109
 embedding Spring Batch in 104–105
 with HTTP request 105–109
 web container 35
 web controllers
 communicating job status from 341–344
 REST 323–329
 configuring controller, repository, and
 gateway 327–328
 implementation of 324–325
 Spring Integration gateway 326
 testing job submission 326–328
 WEB-INF directory 104, 108
 WEB-INF/batch-infrastructure.xml file 108
 web.xml file 104, 108, 451, 453
 WholeBatchTest class 432
 width field 167
 WorkManager class 378
 WorkManagerTaskExecutor implementation 378
 WrappedJsonLineMapper class 131
 writeCount 175–177
 writeHeader 171–172
 writer attribute 63, 65, 67
 writer element 66

writers, item 268–269
 writing data 157–192
 adapting existing services for reuse 183–185
 concepts for 158
 custom item writers 187–189
 to databases 179–183
 with JDBC 179–181
 with ORM 182–183
 to files 159–179
 adding header and footer 171–173
 computed fields 165
 delimited files 166
 file sets 178–179
 fixed-width files 166–169
 matching classes to LineAggregators 169–171
 using FieldExtractor class 163–165
 using FlatFileItemWriter class 161–163
 using LineAggregator Interface 163
 XML files 173–178
 to JMS 185–186
 to send email messages 186–187
 writing files 159

X

XA-compliant driver 261
 XML (Extensible Markup Language) 6, 135–137
 features of 56–57
 launching batch jobs from Spring scheduler
 with 101–102
 reading and writing into database 336
 Spring scheduler 99
 vocabulary 54–57
 namespace 54–56
 XML features 56–57
 XML files, writing data to 173–178
 XML namespaces 318
 XMLEvent 175–176
 XMLEventFactory class 175–176
 XStream 6
 XStreamMarshaller class 174, 177

Z

ZIP archive 22, 28