

## Symbols

---

# operator 135, 146, 166  
  in Map literals 162  
  multiple uses 162  
  projection, and 162  
\${*expression*} 140–141  
%{ ... } 152  
%{*expression*} 104, 141, 178  
  and i18n 291  
/\*.*action* 347  
/\*.*do* 347  
@SkipValidation 373

## A

---

AbstractInterceptor 322  
.action 21  
.action extension 182  
  configuring 21  
action 203  
  as central figure of  
  framework 9  
  as encapsulation of work 44  
  basic validation, and 59  
  business logic, and 44  
  choosing a result 46  
  contract 34  
  cooperating with  
  interceptors 54  
  declaring 53  
  declaring in XML 33  
  default implementation 53  
  demonstrated in  
  HelloWorld 34

execute() 34  
flexibility 210  
in classic web application 204  
invocation 16  
invocation process 76  
mappings 349–350  
obligations to framework 44  
properties files, and 60  
result agnostic 210  
returning control string 46  
role in type conversion 106  
tag 148–149  
targeted by forms 182  
to URL resolution 78  
two roles of 14  
versus action implementation  
  class 112  
action attribute 181, 183  
  of form tag 182–183  
action extension 32, 222  
Action interface 34, 37,  
  52–54, 72  
  annotations, and 37  
  introduction 53  
  result name constants 53  
ActionClass.properties 292  
ActionClassName-aliasName-  
  validation.xml 275  
ActionClass-validations.xml 263  
ActionContext 98, 131–133,  
  149, 166  
  accessing from Velocity 224  
  application map 134  
  attr map 134  
  data, and 132  
  dispatcher result, and 215

i18n interceptor, and 303  
in Struts 2 architecture 16  
IteratorStatus, and 151  
keeper of all data 132  
locale determination,  
  and 302  
managing contents 146  
MVC, and 133  
named objects 143  
OGNL, and 132–135  
parameters map 134  
redirects, and 219  
relationship to  
  ValueStack 133  
request map 134  
results, and 207, 228  
role in architecture 17  
Servlet API, and 133  
servlets, and 216  
session map 134  
setting object in 143  
storing a ResourceBundle  
  in 155  
storing beans 147  
storing objects in 145  
tags, and 131  
  ValueStack, and 135–137  
ActionContextCleanUp 311  
ActionForm 62, 340–343  
  migrating to Struts 2 341–343  
ActionForward 340–341  
ActionInvocation 76, 78, 97  
  as parameter to invoke() 80  
  interceptor firing, and 78  
  interceptors, and 76  
  invoke() 79

- ActionInvocation (*continued*)
    - invoking interceptors 79
    - results, and 206
    - role in framework 100
    - state 79
    - workflow interceptor, and 85
  - ActionMapping 340
  - actionMethod 377
  - ActionName-
    - conversion.properties 124
  - action-oriented framework 9, 43
  - actionPackages 29, 36
  - ActionProxy 329
  - ActionRedirect 220
  - actions 13, 43–46, 72–73
    - Action interface, and 53
    - advanced usage 361–362
    - Ajax, and 210
    - as locus of data transfer 45–46
    - as MVC model 13, 45–46
    - associating interceptors
      - with 93
    - autowiring, and 242
    - configuring 22
    - contract with framework 52
    - creating with Spring 239–242
    - creation of 233
    - CRUD operations, and
      - 371–379
    - declaring 36
    - form prepopulation, and 171
    - forms, and 350–352
    - implementing 24, 52–62
    - in HelloWorld 30, 32
    - in Struts 2 Portfolio 52
    - intelligent defaults, and 25
    - keeping them clean 45
    - mapping interceptors to 93
    - mapping to URLs 49
    - migrating from Struts 1
      - 340–341
    - ModelDriven 64–66
    - multiple entry methods 274
    - packaging 46–52
    - role in framework 44
    - selecting result 46
    - Struts 2 vs. Struts 1 46
    - thread safety, and 46
    - unit-testing 328–332
    - wiring for validation 261–267
    - working with interceptors 67
    - writing 52
  - action-scoped error message 58
  - ActionSupport 38, 54–62, 72, 351
    - Action interface, and 54
    - basic internationalization, and 61
    - basic validation, and 258
    - locale, and 304
    - LocaleProvider, and 61
    - localization, and 60
    - ResourceBundles, and 60
    - role in basic validation 54
    - TextProvider, and 60
    - validation, and 262
    - ValidationAware, and 58, 262
    - workflow interceptor, and 54
  - add 364
  - addActionError() 58
  - addFieldError() 57–58
  - advanced validation 256
  - Ajax 168, 228, 356
    - applications, building 204–205
    - artist browser 212
    - client example 205, 209
    - demands on architecture 14
    - migrating to 211
    - plug-in 212
    - results, and 202, 204
    - Struts 2 tips 212
    - tag theme 212
    - tags 195
    - web applications 203
    - with XML
      - communication 212
  - Ajax in Action 209
  - alias-mapped actions and validation framework 271
  - alternative method
    - invocation 361–362
  - alwaysInvokePrepare 87
  - anatomy of URL 371
  - anchor tag 154
  - annotations 12, 22, 356
    - declarative architecture, and 13, 22
    - declaring validators 278
    - determining namespaces for annotated actions 37
    - for results 38
    - HelloWorld 36–38
    - identifying actions with 37
    - in the source code 37
    - location 36
    - mapping classes to tables with 250
  - scanning for 23
  - setting actionPackages init param 36
  - telling Struts 2 where to look 29
  - tool support 23
  - used in HelloWorld 36
  - using with JPA 250
  - validation framework, and 257, 278, 281
  - zero configuration, and 13, 25
- Ant 26, 328
    - Tomcat targets 26
  - AOP 254
  - Apache 11
  - Apache Tiles 313–315
  - Apache Tomcat 6, 26
  - appendPrefix parameter 274
  - application 134
    - HelloWorld 25–36
  - applicationContext.xml 235–236
    - auto-wiring, and 242
    - Hibernate, and 246
    - JPA, and 246
  - applyInterceptor 323
  - architectural components, declaring 47
  - architectural imperative of a framework 10
  - architecture
    - declarative 21–25
    - flexibility 14
    - in HelloWorld 30
    - MVC 4
    - of Struts 2 13
    - of UI component tag API 174
  - array properties and OGNL mapping 112
  - Arrays 108, 127
  - arrays 342
    - data transfer, and 112
    - OGNL, and 159
    - receiving data transfer with 113
    - type conversion, and 112
  - arsenalist 331
  - aspect-oriented programming 244
  - assertions 329
  - associative array 208
  - asterisk 364
  - asynchronous request 209
  - attr 134

- attribute types and UI tags 178
- attributes 134
  - common 178–180
  - non-String 140–141
  - OGNL, and 139–142
  - String 140–141
- authenticate 356
- authentication 95
- AuthenticationInterceptor
  - 95, 171
  - building 95, 99
- automatic data transfer 44, 51
  - uploading, and 68
- automation 10
- autowiring 243, 254
  - by auto 243
  - by constructor 243
  - by name 242
  - by type 243
- changing the method 243
- declarative architecture,
  - and 243
- flavors 242
- interceptor 83
- potential targets 242
- struts.properties, and 243
- with Spring 242–244

Aware interfaces 238

## B

---

- Back button 89
- banking transaction 366
- base class 373
- baseLayout 382–383
- BaseStrutsTestCase 331
- basic validation 258–261, 280
  - implementing 57
  - in Struts 2 Portfolio 57
  - renewing 258
  - using with the Validation Framework 260
- Beans 311, 317
- beans 146
  - postprocessor 247
  - properties, referencing 158
  - tag 145–148
  - type conversion, and 102
- beautiful code 371
- best practices 11, 231
  - web application deployment 30
- binding request parameters 8

- bloated struts-config.xml
  - file 349
- Bob Lee 317
- bonus components 198–201
- Boolean 108
  - request params 190
  - values in forms 189
- breadcrumb plug-in 321–325
- BreadCrumbInterceptor 323
- browser 168
  - as application 206
  - locale determination 289
- build 30
  - methodology 28
  - tool 26
- building Ajax applications
  - 204–205
- built-in interceptors 78, 81–90
- built-in validators 267
- BundleName\_languageCode.
  - properties 285
- bundles
  - retrieving messages from 295–299
  - specifying with i18n tag 299
- business logic 8–9, 43, 72, 107, 203
  - actions, and 44
  - calls 9
  - in HelloWorld 33

## C

---

- calls 9
- cascading style sheets 353
- chain of properties 158
- chapterFive.xml 111
- chapterSeven.xml 172
- chapterThree.xml 47
  - basic validation, and 58
- charts 315
- checkbox component 189
- checkboxlist component
  - 189, 196
- Circles, converting to
  - Strings 123–124
- classes 28
  - ActionSupport 54–62
  - Locale 284–286
  - mapping to database tables 250
  - wrapper 109–111
- classic web application 203–204
- ClassName-aliasName-validation.xml 276
- ClassName-conversion.
  - properties 117, 342
- ClassName-validation.xml 273
- classpath resources 28
- client/server 4–5
- closing tags 148
- code coverage 331
- coding
  - JPA 249
  - JSON result 206
- Collection and OGNL 162
- collection-backed
  - components 190–198
  - prepopulation, and 197–198
- Collections 114, 127, 342
  - as targets of data transfer 114
  - data-transfer, and 111
  - OGNL, and 159
  - select boxes, and 191
  - type conversion, and 111
  - type specification, and 117
  - UI component tags, and 191
- comma-separated expressions in
  - OGNL 164
- comments in source code 32
- common plug-ins 311–316
  - JFreeChart 315–316
  - SiteMesh 311–312
  - Tiles 313–315
- Commons Validator 355
  - migrating to validation framework 355–357
- component tag 333–334
- comprehensive test 331
- conditional rendering 150
- confidence 340
- configuration 12, 21
  - avoiding 25
  - of framework itself 21
  - of type converters 124–126
  - redirect result 220
  - setting framework properties 177
  - two kinds 21–22
  - types of in Struts 2 39
  - with Java annotations 23–24
  - XML, and 23
- constants 311, 318
  - declaring in XML 31
- constructor, autowiring 243
- ContextLoaderListener 239
- control string 72
  - returned by action 46, 76
- control tags 150–151
- control-flow tags 138

- controller 12
  - in MVC 13
- convention over configuration 3, 25, 169, 186
- conversion error 111
- conversionError interceptor and validation 259
- converting types 102
- core components of Struts 2 43
- counter bean 146
- coupling 233
- creating with Spring 239–242
- cross-cutting concerns 75
- cross-cutting tasks 16
  - factoring out of business logic 54
- CRUD 171, 370
  - actions, and 371–379
  - prepare interceptor, and 87
  - workflow interceptor, and 86
- crumb tracking 321
- CSS 9, 169, 185
  - for layout 177
  - tag themes, and 176
- css\_xhtml 175
  - theme 176, 187
- currency 283
- current locale 294
- custom tags 333
- custom templates, writing 337
- custom themes 337
- custom validators 267, 269–271
  - checking password strength 267–269
- customizing type
  - conversion 122–126

## D

- data 43
  - accessing 17
    - from actions 34
    - with OGNL 104
  - ActionContext, and 132
  - auto transfer 46
  - binding to Java types 103
  - carried by action 45
  - carrying in domain objects 62
  - coming into framework 105–107
  - domain data 256
  - flow through framework 102, 105
  - incoming 35
  - into framework 105

- leaving framework 107
- location 35
- location in framework 131
- moving through Struts 2 30
- outgoing 35
- path through framework 35
- persistence 232, 245
- police 355
- referencing from results 64
- source 247
- storage 17, 132
- storage in framework 166
- transferred to action 45
- transferring 102
- type 104
- validation of 10, 255
- ValueStack, and 132
- data binding 6
  - name and value attributes 173
- data layer 9
- data tags 138, 142–149
- data tier 72, 203
- data transfer 46, 73, 102, 126, 168
  - Collections, and 111
  - forms, and 104
  - key attribute, and 297
  - key interceptors 82
  - Lists, and 101, 118
  - Maps, and 101
  - methods 62
  - multi-valued request parameters, and 111
  - object instantiation 64
  - OGNL, and 103
  - onto array-backed properties 112
  - onto domain objects 67
  - onto Lists 116
  - params interceptor, and 83
  - security issues 67
  - Struts 2 Portfolio, and 112
  - tags, and 131
  - type conversion, and 63, 101
  - UI Components, and 173
  - UI tags, and 170
  - validation, and 259
  - with domain objects 62–67
  - with object-backed JavaBeans properties 62–63
- data transfer object 72
  - action as 46
- databases, choosing 246
- Date 108

- dates 342
  - formatting 283, 301
  - localizing 301
- DB abstraction 376
- DB4o 376
- declarative architecture 21, 23–25, 383–385
  - ActionInvocation, and 79
  - actions, and 43
  - annotation-based 21, 23–24
  - choosing a mechanism 24
  - declaring interceptors 90
  - declaring results 46
  - in XML-based HelloWorld 30
  - Spring object creation, and 241
  - with annotations 37
  - XML elements 51
  - XML or annotations? 24, 38
  - XML-based 22–23
- declarative mapping 361
- declaring
  - architecture 22–25
  - interceptors 90–94
  - JSON results 211
  - validation metadata 262–265
- decorator 311
- default bundles 294
- default interceptor stack 50
- default namespace 49
- default.properties 177
- default-interceptor-ref
  - in struts-default 94
  - XML element 92
- defaultStack 74, 82, 93
- execution 76
- fileUpload 70
- i18n, and 303
- parameters, and 94
- validation framework, and 259
- DefaultTypeConverter 342
- definitions 382
- department.project 343
- dependencies
  - management of 233
  - managing with Spring 246–248
- dependency injection 45, 231–235, 254, 329
  - autowiring with Spring 242–244
  - how it works in code 238
  - strategies 238–244

- Dependency Injection
  - library 317
- deploying 26
- deployment descriptor 28
- design by difference 338
- design patterns 11–12, 18, 357
- destroy 364
- developing interceptors 78
- DHTML 206, 210
- distinct mappings 361
- dispatcher result 203
  - declaring 218
- dispatcher result type 214
  - classic web application, and 204
  - dispatching to another servlet 215
  - includes, and 218
- dispatcher results
  - configuring 218
  - dispatching to a JSP 215
- DispatcherResult 214
- divs 177
- .do 21
- document root 27
- DOM 334
- domain 8–9
- domain data 203–204, 256
  - exposing via ModelDriven actions 64
  - on actions 34
  - validating 256
- domain model 341
- domain model objects and validation framework 271
- domain objects
  - data transfer, and 62
  - exposed to direct data transfer 73
  - ModelDriven, and 64
  - prepopulation, and 171
  - used to receive data transfer 64, 67
  - validation, and 271
- double validator 260
- doubleList 200
- doubleselect component 199–201
  - appearance on page 199
- dropCrumb 324
- duplicate 367
  - form posts 89
  - form submits, preventing 366–369

- requests 89
- token 367
- DynaForm 341
- dynamic method
  - invocation 362–366
- dynamic table 376
- dynamic workflow 365

## E

- Eclipse 327
- element type arrays and OGNL 114
- elements, specifying type 116
- else tag 151
- embedded OGNL, escape sequence 264
- empty action components 32
- entity annotation 251
- EntityManager 247
  - coding with 251–253
  - injecting into service object 251
  - injecting with Spring 247
- EntityManagerFactory 247, 252
- Entry 194
- environment
  - of Struts 2 4
  - setting up 327–328
- error messages
  - i18n, and 298
  - in user interface 59
  - in validation 258
  - UI tags, and 173
  - validation, and 58
- escape sequence 104, 141
- eval() 210
- exception interceptor 88
- ExceptionHandler 89
- exceptionStack 89
- excludeMethods 86, 94, 368
- execAndWait 369
- execute() 44, 203
  - accessing domain data from 35
  - introduction 53
  - ModelDriven actions, and 65
- explicit namespaces 49
- expression language 17, 102–103, 126, 131, 157, 166
  - for maps 119
  - OGNL power tools 135
  - pluggability 135
  - primer 157–165
  - purpose 104

- reference 157–165
  - syntax 135
  - ValueStack, and 157–163
- expression validator 264
- extension points, internal 319
- externalized text 187

## F

- factor 363
- factored out 373
- factories 234
- field element in validation XML 270
- field validators 263
  - annotations, and 280
  - short circuiting, and 278
- field-validator elements
  - in validation XML 270
  - in XML meta-data 263
- FieldValidator interface 267
- FieldValidatorSupport 267
  - extending 268
- files, uploading 44, 67–72
- fileUpload interceptor 67–69, 84
  - automatic transfer of File 68
  - exposes new params 68
  - multiple files 71
  - params interceptor, and 69
  - pre- and postprocessing 69
  - retrieving File in action 71
  - Struts 2 Portfolio example 69
  - tweaking 71
- filter 29
  - OpenEntityManagerInView 248
- FilterDispatcher 12–13
  - as MVC controller 13
  - declaring in web.xml 29
  - in Struts 2 architecture 16
- Filtering 162
- filter-mapping 29
- findForward 340
- Firefox 357
- forcing OGNL resolution 141
- form bean 352
- form component 180–185
- form fields
  - binding to properties 170–173
  - names, mapping to properties 109
  - OGNL, and 158

forms 168–174, 182  
 actions, and 350–352  
 building a form with Struts 2 tags 70  
 prepopulation 173  
 preventing duplicate submits 366–369  
 submission 182  
 tags, <s:token> 366–368

forward 350

frameworks 9–11, 18  
 alternatives 10–11  
 architectural decisions, and 4  
 architecture, and 10  
 automation 10  
 for web applications 3  
 roll your own 11  
 Struts 2 11–18

FreeMarker 27, 156, 166, 311–312, 332, 354  
 accessing Struts 2 data 226  
 accessing ValueStack 227  
 as basis for UI tags 168  
 as view layer choice 15  
 documentation 226  
 embedding OGNL 226  
 in UI tags 175  
 native expression language 226  
 result type 204, 223–227  
 Struts 2 tags, and 175  
 syntax 139  
 tags, and 138  
 UI component templates, and 175  
 usage 226

FreemarkerResult 213, 225–227

front controller 13

front end 14

## G

---

generics and type conversion 117

getBundle() 286

getFieldName() 269

getFieldValue() 269

getLocale() 304

getModel() 65

getText() 290, 296

global actions 31

global result 98, 227

global-messages.properties 187

global-results element of declarative architecture 227

Google Guice 317

graphs 315

Guice 310

## H

---

hard-coding 365

hasErrors() 260

head 361

head component 180

heavy lifting 362

HelloWorld 13, 18, 20, 25–36  
 annotations, and 24–25  
 demonstrating Struts 2 architecture 30  
 introduction 25  
 JavaBeans, and 35  
 OGNL, and 35  
 purpose 20  
 UI Component tags, and 33  
 ValueStack, and 35  
 with annotations 36–38  
 workflow 30  
 XML configuration, and 23  
 xml declarations 32  
 XML, and 25

Hibernate 232, 237, 244  
 using with JPA 245–249

hidden component 199

hidden fields 199, 366

hidden request parameters 199

href 154

HTML 5, 9, 168, 334  
 data, and 103  
 forms 168–174  
 fragments 149, 203, 205  
 generated by UI component templates 176  
 goo 185  
 options 197  
 pages 205, 228  
 tables 177, 185  
 tag themes, and 176

HTML DOM 182

HTML form elements 168

HTML forms  
 data transfer, and 105  
 name attribute of fields 36  
 OGNL, and 104  
 tags, and 167  
 UI component 181

HTML markup  
 generated by UI tags 169  
 generating 169–170

HTTP response 228  
 in servlets 216

HTTP. *See* Hypertext Transfer Protocol (HTTP)

HttpServletRequest 83, 105, 329, 340

HttpServletResponse 83, 340

human intervention 343

Hypertext Transfer Protocol (HTTP) 5–6, 102, 157, 302  
 converting to Java types from 108  
 GET types 321  
 headers and locale 287, 296  
 hurdles for web applications 5  
 in Ajax web apps 205  
 in servlet API 8

## I

---

i18n 282–283, 305  
 ActionSupport, and 290  
 currency, and 301  
 dates, and 301  
 dynamic text 299–300  
 global bundles 295  
 how the framework finds bundles 291  
 key attribute, and 297  
 locale determination 296, 302, 305  
 Locale, and 284–286  
 native Java parameterization of text 300  
 numbers, and 301  
 parameterizing text 299  
 quick demo 287–291  
 ResourceBundles, and 284–286  
 retrieving localized text 295  
 specifying an ad hoc bundle 299  
 Struts 2 internals 290–291  
 Struts 2 way 286  
 TextProvider 290  
 type conversion, and 298  
 validation framework, and 297  
 ValueStack, and 290

i18n interceptor 303–305

i18n tag 154–155, 299  
 attributes 155  
 ValueStack, and 299

IDE 26, 327

- if tag 151
- if/else tag 151
- impatient users 369
- implementation
  - changing with Spring 237
  - hiding with interfaces 236–238
- implementing
  - actions 52–62
  - components 22
  - type converters 122
- include tag 152–153
- includeMethods 86, 368
- includeParams 154
- incoming request data
  - validation 8
- indexing
  - OGNL, and 113, 115
  - properties 113
- inheritance
  - Struts 2 packages, and 50
  - validation inheritance 277
- injected 329
- injection 318
- input
  - attribute 349
  - result for validation 58
- installation of sample code 26
- intelligent defaults 22, 25, 316
  - versus flexibility 53
- intercept() 79, 97, 100
- interceptor 12, 311
  - control flow 80
  - entry method 79
  - how they get fired 79
  - interface 78
  - params 51, 76
  - pre- and postprocessing
    - example 69
    - sequencing 55
  - XML element 92
- Interceptor interface 95
- interceptor stack
  - declaration of 51, 90–93
  - default 50
- interceptor-ref
  - parameters, and 94
  - XML element 92
- interceptors 15–16, 44, 74, 99–100, 372–377
  - ActionInvocation, and 76, 78
  - alteration of workflow 81
  - annotations, and 90
  - architectural purpose 75
  - associating with actions 90
  - AuthenticationInterceptor
    - 95, 99
  - autowiring 83, 242
  - benefits 77
  - building block stacks 79
  - building stacks 77, 90, 99
  - building your own 95–99
  - built-in 81–90, 100
  - code reuse, and 77
  - commonly used 82
  - concepts 76
  - configuring 22
  - control of workflow, and 59, 261
  - control string, and 80
  - coordinating with actions 54
  - core framework tasks, and 74
  - creating with Spring 239–242
  - cross-cutting concerns, and 75
  - data transfer interceptors 82–84
  - declarative architecture, and 90
  - declaring 90–94
  - declaring your own
    - interceptor 98
  - default stack 36, 74, 78
  - developing your own 78
  - example code 80
  - exception 88
  - execAndWait 89
  - execution 80
  - fileUpload 68–69, 84
  - firing order 78
  - handling common tasks 67
  - i18n 303–305
  - implementing core functionality of framework 51
  - importance 51
  - in HelloWorld 30, 36
  - in request processing 16
  - intelligent defaults, and 25
  - layering power 77
  - lifecycle 80
  - life-cycle methods 95
  - logger 82
  - mapping to actions 93
  - modelDriven 88
  - OGNL, and 159
  - param tag, and 85
  - parameterizing 85, 94
  - params interceptor 82
  - per-action mappings 93
  - performance 90
  - phases 81
  - postprocessing 75, 96
  - prepare 87
  - preprocessing 75, 96
  - reuse 69
  - role in architecture 16
  - role in framework 100
  - rolling your own 95
  - scoped-modelDriven 89
  - separation of concerns, and 75
  - sequencing 99
  - servlet-config 83
  - static-params 83
  - timer 82
  - token and token-session 89
  - tokenSession 368–369
  - utility interceptors 82
  - validation framework, and 258–261
  - validation, and 86, 259
  - when they fire 16
  - workflow 77, 84
  - workflow, and 59, 76
  - working with actions 60
  - XML declaration, and 90
  - XML element 92
- interceptors element in declarative architecture 93
- interceptor-stack, XML element 92
- interface
  - decouple from
    - implementations 235
    - hide implementations 236
- interfaces 372–377
  - Action 52–54
  - autowiring by type, and 244
  - hiding implementation with 236–238
  - Interceptor 95
  - ResourceBundles, and 292
- internal component
  - system 316–319
- internal extension points 319
- internationalization 9–10, 154, 168, 344
  - basic 61
  - UI components, and 187
  - UI tags, and 173
  - via ActionSupport 61
  - See also* i18n
- Internet 4
- invalid.fieldvalue.fieldname 299
- invalid.token 367

invocation of actions 14  
 invoke methods from  
   OGNL 164  
 invoke() 81  
 IoC. *See* dependency injection  
 iterator tag 150–151  
   attributes 150  
 iterators 195  
 IteratorStatus 150–151

## J

---

J2EE 232  
 JAR files  
   collecting 245  
   required by Struts 2 28  
   web applications, and 7  
 Java 3  
   annotations 23  
   data types 6, 8  
   generics 331  
   i18n 283  
   native i18n support 285  
   types 157  
   web application  
     development 26  
   web applications, and 4  
 Java annotations,  
   configuration 23–24  
 Java Collections 159  
 Java EE 3  
 Java Persistence API 244–253  
   why to use 244–253  
 Java Servlet API 5  
   *See also* Servlets  
 Java types 102  
   converting to 122  
   data transfer, and 104  
   tag attributes, and 140  
 java.lang.Exception 88  
 JavaBeans properties 62–64,  
   147, 172, 257  
   as receivers of data  
     transfer 110  
   carrying data on actions 57  
   carrying data on  
     properties 44  
   custom results, and 211  
   data transfer, and 82  
   on action 45  
   on actions 34  
   parameters for validators,  
     and 268  
   setter injection, and 236  
   uploading, and 68

JavaBeans specification 159  
 JavaScript 9, 168, 182, 356  
   Ajax client 206  
   client 209  
   functions 180  
   hidden fields, and 199  
   in head component 180  
   JSON, and 205  
   with UI Components 178  
 Jettison 208  
 JFreeChart 315–316  
 journey of a Struts 2 request 378  
 JPA 231, 234, 244, 253  
   annotations 250, 254  
   coding 249  
   configuration with  
     Spring 246–248  
   data operations 253  
   database configuration 246  
   EntityManager 247, 251–253  
   persistence unit 247, 249  
   required JARs 245  
   Spring-managed 246  
   using with Hibernate 245–249  
   writing the code 251  
 JSON 203, 228  
   response 205, 209  
   result 206  
   result type 205–213  
   serializing objects 208  
   syntax 208  
 JSP 15, 27, 228  
   as view layer option 14  
   in result 202  
   including with include  
     tag 152  
   result type 203  
   results 213  
   Struts 2 tags, and 175  
   tags, syntax 138  
   using 204  
   .jsp extension 183  
 JSTL 141, 156, 344  
   and ValueStack 156  
 JUnit 329–332

## K

---

key attribute 174, 186  
   i18n, and 174, 296  
   ResourceBundles, and 186  
 key objects, specifying type  
   for 121

## L

---

l10n 282–283  
 label 170  
 label attribute 169, 186, 188  
 label component 198  
   usage 199  
 languages 283  
 layering and interceptors 77  
 layout 379–386  
   consistency 379–383  
 layout-related markup 186  
 lazy loading 248  
 lib 28  
 List literals in OGNL 160  
 List, implementing properties  
   as 115  
 listKey 193  
   in pre-selection 198  
 Lists 108, 127  
   as targets of data transfer 114  
   data transfer, and 102  
   element types, and 114  
   initializing 117  
   OGNL, and 115, 159  
   pointing to data in 108  
   specifying type for type  
     conversion 116  
   UI components, and 191  
 listValue 193  
   in preselection 198  
 literals of OGNL 163  
 Locale 154, 284–286  
   coding against directly 286  
   ResourceBundles, and 284  
 locale 283, 305  
   automatic determination 302  
   detection 302  
   determination 287  
   determination with  
     ActionSupport 61  
   extensions 294  
   how the framework  
     chooses 296  
   HTTP headers, and 296  
   letting the user choose  
     302–305  
   message retrieval, and 294  
   overriding 302–305  
   setting programmatically 305  
   specificity 294  
   storing in database 305  
   Struts 1, and 354  
 LocaleProvider 61, 304  
   implementing 305

locales, supporting 285  
 locale-sensitive 284, 345  
 localization, provided by  
   ActionSupport 60  
   *See also* 110n  
 localized error messages 174  
 location parameter 219, 221  
   OGNL 221  
 logger interceptor 82  
 long requests 89  
 look and feel 311, 360, 379–386  
   consistency 379–383  
 loop 150  
 loose coupling 238

## M

---

managed life cycle 7  
 Map  
   elements, specifying type  
     for 120  
     OGNL, and 160, 162  
 mapping 340  
   form field names to  
     properties 109–122  
   interceptors to actions 93  
   with OGNL expressions  
     109–122  
 Maps 108, 127  
   as targets of data transfer 119  
   data transfer, and 102  
   initialization 120  
   OGNL, and 119  
   pointing to data in 108  
   select component, and 194  
   specifying element type 120  
   specifying key type 121  
   type conversion, and 119  
   UI components, and 191  
 maps 342  
 marketing department 379  
 markup 184, 354  
 master page 382  
 Maven 26  
 message element 273  
   i18n, and 298  
   localized text, and 265  
   XML validation  
     metadata 263–264  
 message key 357  
 message resources, breaking  
   up 345–346  
 message text, resource bundles  
   as 60–62  
 MessageFormat 291, 302, 306  
 message-resources  
   parameter 354  
 messages  
   reading from properties  
     files 291–302  
   retrieving 294  
   retrieving from bundles  
     295–299  
 meta tag 370  
 metadata 38  
   declarative architecture,  
     and 23  
   validation metadata 257, 281  
   with annotations 23, 37  
 MethodFilterInterceptor 95,  
   322, 368  
 methods  
   alternative method  
     invocation 361–362  
   dynamic invocation 362–366  
   filtering 95  
   name at runtime 362  
   wildcard selection 362–365  
 migrating 346  
   ActionForms 341–343  
   actions 340–341  
   from Struts 1 339–359  
   piecemeal approach 346–359  
   to Ajax 211  
   web pages 352–354  
 migration 12, 339, 360  
 miniature S2 context 310  
 minimum set of libraries 327  
 mini-MVC 174, 201  
   of UI components 175  
 miscellaneous tags 138, 152–156  
 mock service object 237  
 model 12, 45  
   in MVC 13  
 Model 2 12  
 model component 34  
 ModelDriven 64–66, 73, 136,  
   171–172, 256, 346  
   actions 64–66  
   receiving data on model  
     objects 62  
   validation framework,  
     and 271, 273  
   interface 273  
   params interceptor, and 83  
 modelDriven 372  
   interceptor 88  
 Model-View-Controller 4  
   pattern 12–15

modularization of Struts 2  
   applications 31  
 modules in XML-based declara-  
   tive architecture 31  
 multipart request 68  
 multipart/form-data 70  
 multiple selection and  
   preselection 198  
 multiple selections 196  
 multi-valued request  
   parameters 111  
 MVC 11–12  
   Ajax, and 210  
   cleaner implementation 12  
   cleaning up 75–77  
   data persistence, and 248  
   lazy loading in view, and 248  
   pattern 12–15  
   result as view 202  
   results 228  
 MyInterface.properties 292  
 mySensitiveStack 369  
 MySQL 245  
 MySuperClass.properties 292  
 MyWebBundle.properties 345

## N

---

naked POJO 329  
 name attribute 169, 173, 186,  
   191, 349  
   form fields, and 185  
   key attribute, and 297  
   prepopulation, and 197  
 namespace 72, 148, 183, 311  
   attribute 181, 183  
   default 49  
   explicit 49  
   how they work 32  
   in Struts 2 package 32, 47  
   root 49  
   with annotations 37  
 naming convention for  
   annotations 37  
 nanoTime 377  
 new operator 233  
 no-argument constructor 159  
   OGNL, and 121  
 nonfield annotation 280  
 non-String attributes 140–141  
 nonstring attributes 141  
 now.ftl 334  
 null property  
   access 121  
   OGNL, and 158

numbers  
 formatting 301  
 localizing 301  
 numeronyms 282

## O

---

object creation 232  
 with Spring 233, 235  
 object database 376  
 object instantiation 232  
 ObjectFactory 233, 253  
 Spring version 238  
 Object-Graph Navigation Language (OGNL) 12, 15, 30, 126, 131, 166, 274  
 # operator 135, 143, 145, 162  
 accessing static methods and fields 165  
 ActionContext, and 132–135  
 advanced expression language features 163–165  
 advanced features 135  
 array backed properties, and 113  
 array syntax 112  
 as form field names 109  
 as glue technology 103  
 binding form fields to Java properties 103  
 Booleans, and 190  
 data binding, and 104  
 data validation, and 264  
 embedded 221  
 escape sequence 104  
 expression language 103–104, 133, 135, 157  
 filtering collections 162  
 forcing resolution 141  
 forcing tag attributes to be parsed for OGNL 141  
 form field names, and 108  
 FreeMarker, and 224  
 full power 157  
 HTML forms, and 18  
 i18n, and 287, 290, 300  
 in HelloWorld 35  
 in parameters 219  
 in select component 194  
 in tag attributes 139–142  
 in the view 108  
 initializing arrays 114  
 instantiation of null references 121

integration into Struts 2 108  
 introduction to expression language 158  
 list literal in UI component 191  
 List literals 160  
 Lists, and 115, 117  
 literals 163  
 locale determination, and 304  
 Map literals 161  
 Map syntax 121  
 mapping form fields to Java properties 109  
 Maps, and 119–120, 160  
 method invocation 147, 164  
 null properties, and 158  
 operators 159, 164  
 parameterizing localized text 300  
 params interceptor, and 83  
 pointing to data 106  
 power tools 163  
 prepopulation, and 172  
 projecting collections 162  
 reference 157, 165  
 referencing data in arrays 114  
 resolution 141  
 resolution of expressions 133, 137  
 ResourceBundles, and 187  
 role  
 in architecture 16–17  
 in data retrieval 36  
 in data transfer 36  
 in framework 105–107  
 root object 132, 134–135  
 Struts 2 tags, and 102  
 Struts 2, and 157  
 syntax 157  
 tags, and 108, 131  
 targeting JavaBeans properties 111  
 targeting properties 125  
 type converters 103–104, 122  
 type conversion, and 107  
 UI components, and 201  
 UI tags, and 178  
 value attribute, and 103  
 ValueStack, and 106, 132, 137, 157–163  
 Velocity, and 224  
 with Collections 159  
 objects  
 managing with Spring 235

service object 251–253  
 tightly coupled 233–235  
 OGNL expression 141  
 mapping with 109–122  
 simplifying 143  
 OGNL expression language 103  
 OGNL. *See* Object-Graph Navigation Language (OGNL)  
 OGNL-to-Java mapping 112, 121  
 Open Session In View 245  
 open source servlet containers 6  
 open-closed principle 357  
 OpenEntityManagerInView 248, 254  
 filter 248  
 opening tags 148  
 OpenSessionInView 248  
 OpenSymphony 11  
 opensymphony.com 312  
 operators of OGNL 159, 163  
 option-backed properties 62–67  
 options in forms fields 191  
 organizing packages 47–50  
 ORM 232  
 overflow the kernel 363  
 overridden by plug-ins 318  
 Override framework constants 383  
 overriding  
 locale 302–305  
 parameters 94  
 templates 336

## P

---

package element  
 in declarative architecture 93  
 in XML declarative architecture 32  
 package tag 350  
 packages 46–52, 72, 371  
 attributes of 48  
 comparison to Java packages 47  
 containers for framework components 47  
 default result type for 212  
 extending attribute 48  
 factoring actions into 47  
 inheritance 49  
 name attribute 48  
 namespace attribute 48  
 organizational strategy 48

- packages (*continued*)
    - organizing 47–50
    - Struts 2 Portfolio packages 47
    - struts-default 50, 67–68
    - URLs to actions, and 48
  - page refresh 205
  - page, as view 14
  - pageable list 330
  - paid by the keystroke 327, 349
  - param elements and
    - validation 264
  - param tag 154, 156
    - attributes 156
    - parameterizing text, and 300
    - results, and 207, 219
  - parameterize 365
  - parameters 134
    - default parameters 208
    - overriding 94
    - setting 94
    - to interceptors 85, 94
    - xml tag 94
  - params interceptor 62, 106
    - in action 111
    - modelDriven interceptor, and 88
    - preprocessing, and 76
    - validation, and 55, 259
    - ValueStack, and 136
  - paramsPrepareParamsStack 372
  - parse parameter 219, 227
    - OGNL, and 221
  - parsing to Java types 102
  - pass-through action 33, 184
    - with annotations 38
  - password component 187
  - password strength,
    - checking 267–269
  - password tag 187
  - password validator 267
  - PasswordIntegrityValidator
    - 268, 270
  - path attribute 349
  - pattern 363
  - performance 223, 328
  - persistence 373
  - persistence unit 249
  - persistence.xml 249
  - PersistenceAnnotationBeanPostProcessor 252
  - PersistenceContext
    - annotation 252
  - persistent entity 247
    - mapping to database 251
    - writing Java classes 251
  - plug-in architecture
    - breadcrumbs 321–322, 324–325
    - classpath 310
    - Eclipse 309
    - Firefox 309
  - plug-in registry 311
  - plug-ins 357–359
    - breadcrumb plug-in 321–325
    - common 311–316
    - finding 311
    - overview 310–311
  - POJOs 328
  - polymorphic factory
    - method 331
  - PortfolioService,
    - introduction 57
  - PortfolioServiceInterface 236
  - PortfolioServiceJPAImpl
    - 247, 251
  - postprocessing 75, 81
  - Preparable 87
  - prepare 372
  - prepare interceptor 87
  - prepareInput() 87
  - prepopulation 304
    - of collection-backed
      - components 197–198
    - of forms 170
    - UI components, and 173
  - preprocessing 75, 80
  - preselection 197–198
  - presentation layer 14
  - presentation tier 9
  - prework 385
  - primitives 108–111, 127
    - as targets of data transfer 109
  - PrincipalAware 84
  - processing requests 11
  - production 328
  - professional 379
  - progress bar 369
  - project management 26
  - Projection 162
  - properties 158
    - binding to form fields
      - 170–173
    - JavaBeans-compliant 62–64
    - mapping from form field
      - names 109–122
    - option-backed 62–67
  - properties files 28, 154, 306
    - default 288
    - for localized text 60
    - global 292
  - i18n, and 283
  - location 60, 287
  - ModelDriven actions, and 292
  - package level 292
  - reading messages from
    - 291–302
  - superclasses, and 292
  - type conversion
    - configuration, and 116
  - UI components, and 187
  - where to put 291–302
- property chain 158
- property tag 103, 142, 154
  - attributes 142
  - i18n, and 290
  - in HelloWorld 35
- PropertyResourceBundle 284
- push tag 144–145
  - attributes 144
  - ValueStack, and 137
- pushed 343
- put-attribute 381
- ## Q
- 
- querystring parameters 6
    - dynamically building 219
    - redirectAction results, and 222
    - redirects, and 220
- ## R
- 
- radio component 195–196
    - Collections, and 196
  - readOnly 377
  - read-only form fields 199
  - recursion 77
  - redirect result
    - configuration 220
    - type 219–221
  - redirect to other actions
    - 222–223
  - redirectAction 221
    - result and persisting request
      - params 222
  - redirection, contrast with a
    - dispatch 214
  - refactor 357
  - refactoring 326
  - related methods 362
  - relative paths 184
  - reload 328

- rendering 336
  - request 6, 134
    - attribute 216
    - data 45
    - life cycle 357
    - routing of 13
  - request parameters 6, 8, 45, 84, 105, 108, 131
    - Ajax, and 210
    - binding to Java types 8
    - converting to Java type 125
    - created by fileUpload
      - interceptor 68
    - data transfer, and 110
    - handling 111
    - i18n interceptor, and 303
    - include tag, and 152
    - multi-valued 111, 113
  - request processing 11
    - by Struts 2 15
    - customizing with
      - interceptors 77
  - request\_locale 303
  - RequestAware 217, 330
  - RequestDispatcher 213–219
    - dispatcher result type 214
    - forward() 214
    - include() 214
    - integrating other servlets, and 217
  - requiredstring validator 263, 265
  - resource acquisition 234
  - resource bundles 60
    - for message text 60–62
  - resource management 232
  - ResourceBundle 154, 284, 305
    - class-backed 286
    - creation 287
    - fundamentals 284
    - how they work 284–285
    - i18n tag, and 155
    - key attribute, and 186
    - subclassing 284
    - text tag, and 155, 289
    - validation, and 265
  - ResourceBundles
    - and i18n 283–287
    - associated with actions 288
    - coding against native Java
      - i18n 285–286
    - lookup 291–295
    - naming conventions 288
    - with Struts 2 286
  - resources
    - message resources 345–346
    - storing 284–285
  - response 6
    - JSON response 207
  - RESTful 364
  - result 12, 202, 311
    - accessing data from 45
    - accessing ValueStack 107
    - annotation location 38
    - as MVC view component 14
    - chosen by action 46, 76
    - declarative architecture, and 211
    - declaring a new type 211
    - declaring your own 210
    - default result types 212
    - definition of default
      - names 53
    - flexibility 212
    - JSON 205–206
    - JSPs in HelloWorld 34
    - pulling data from
      - framework 104, 107
    - selection 32, 43
    - selection with String
      - constants 53
    - type attribute 226
    - usage 211
    - used for Ajax 202
  - Result interface,
    - implementing 206
  - result types
    - commonly used 213–223
    - FreeMarker 223–227
    - intelligent defaults, and 25
    - parameterizing 219
    - Velocity 223–227
  - results
    - as MVC views 14–15
    - built-in 203
    - built-in types 213–223
    - configuration 227
    - configuring 22
    - creating with Spring 239–242
    - custom 204–205
    - declaration 224
    - declaring 211
    - different types 202
    - dispatcher result type
      - 213–219
    - FreeMarker 225–227
    - global declarations 227
    - global results 88
  - implementing to receive
    - parameters 207
  - in HelloWorld 30
  - intelligent defaults, and 25
  - JSON type 205–213
  - JSP 203, 214
    - lookup by name 227
    - parameterizing 207
  - redirect type 219–221
  - redirectAction type 221–223
  - role in framework 203–213
  - scope 227
    - type conversion, and 102
    - using 211
  - velocity type 224–225
  - working with actions 203
  - result-type element 221
    - in declarative
      - architecture 211, 218, 224
  - reuse 4
    - maximizing 332–336
    - of interceptors 77
  - reverse order 372
  - rich-client 168
  - root namespace 49
  - root object 134–135
  - rules of the business 365
  - runtime
    - components 23
    - configuration plug-in 310
    - creation of framework
      - components 21
  - rxception 311
- ## S
- 
- <s:token> 366, 368
  - sample application 20
    - deploying 26, 30
    - download 26
    - installing 26
    - source code 28
    - structure 26–27
  - scanning for actions 38
  - scope 350
  - scope attribute 240
  - scopedModelDriven 358
    - interceptor 89
  - select box 191–195
    - generating options 193
    - linking two 199
  - select component 191–195
    - arrays, and 195
    - backed by a Map 194

- select component (*continued*)
  - contrasted to radio component 195
  - demonstrated in Struts 2 Portfolio 193
  - HTML output 194
  - iterators, and 195
  - prepopulation, and 193
  - preselection, and 197
- sensitive methods 368
- separation of concerns 12
  - business logic, and 59
  - in basic validation 59
  - interceptors, and 75
- service locators 234
- service objects 233–253
  - implementing 251
  - injecting 235
  - interfaces, and 236
- Servlet 5
  - request parameters 105
  - session 8
- servlet 6–8, 152
  - filters 28
  - includes 218
  - integrating Struts 2 with 216
  - other than Struts 2 29
  - request 17
  - response
    - dispatcher result type, and 216
    - results, and 207
  - session 17
- Servlet API 5
  - accessing from actions 83
  - decoupling from 233
  - how to access 134
  - keeping away from 133
  - redirects, and 219
  - working with 216
- servlet container 6–7
  - installing 26
- servlet context in Struts 2
  - URLs 32
- Servlet Specification 6, 26, 152
  - dispatcher results, and 214
  - importance to Struts 2 developers 27
- ServletActionContext 207
- ServletActionRedirectResult 22
  - 2–223
- ServletConfigInterceptor 83, 97
  - setter injection, and 235
- ServletContext 83, 183
- ServletRedirectResult 219–221
- session 134
- SessionAware 97
  - setter injection, and 235
- set tag 143–144
- setter injection 172, 235
- setting parameters 94
- setting up environment 327–328
- shopping cart 366
- short-circuit attribute 278
- short-circuiting validation 271, 277
- simple 176
- simple components 180–190
- singletons 240, 341
- SiteMesh 310–312
- sizing machinery 360
- skeleton application 27
- smart defaults 350
- software, structural 10
- spaghetti code 362
- Spanish 345
- Spring 45, 231, 253, 310, 329, 348
  - AOP 241
  - applicationContext.xml 239
  - autowiring 242–244
  - autowiring interceptor 83
  - bean post processor 247
  - bean scope 240
  - bean XML element 240
  - beans 239
  - configuration 239
  - configuring JPA 246–248
  - container 239
  - creating framework objects with 239–242
  - declaring Spring beans 240
  - dependency injection 232, 235, 244
  - EntityManagerFactory 247
  - hooking into declarative architecture 241
  - in the Struts 2 Portfolio 238–244
  - integration with Struts 2 apps 238–244
  - introduction 232–235
  - JARs 239
  - managing dependencies with 246–248
  - managing objects with 235
  - namespace 240
  - object factory 238
  - plug-in 238, 253
  - prototype scope 240
  - schema 240
  - singleton beans 240
  - support for JPA 246
  - transactions 248
  - ways to inject 238
  - web application framework 232
  - why use with Struts 2 232–238
- Spring bean, id attribute 241
- Spring transaction management 253
- spring.jar 239
- Spring-managed transactions 254
- SpringObjectFactory 241
- standards 168
- stateless 369
  - protocols 5
- static resource 184
- static-params interceptor 83
- String attributes 140–141
- stringlength validator 258, 264
- Strings, converting to
  - Circles 123–124
- strings, converting to Java types 102
- structural software 10
- Struts 1 11, 62
  - actions compared to Struts 2 44
  - differences 17
  - legacy 12
  - locale, and 354
  - migrating from 12, 339–359
  - plug-in 347
  - Servlet API, and 133
- Struts 2
  - application, basic layout 25
  - built-in type converters 108
  - community 26
  - data storage, and 133
  - development path 53
  - expression language 157
  - flexible architecture 78
  - framework 11–18
  - history 11–12
  - how it works 15–18
  - interceptors 74
  - internal extension points 319
  - migrating from Struts 1 12
  - OGNL, and 104–105
  - plug-in registry 323
  - project 327
  - Servlet API, and 133

- Struts 2 (*continued*)
    - servlets, and 105
    - setting configuration
      - properties 177
    - tag library 131
    - technological context 5
    - type conversion 103
    - UI component tags 167–201
    - validation 259
    - validation options 86
    - ValueStack 106
    - web site 72
    - workflow 256–261
  - Struts 2 actions
    - contract with framework 52
    - implementing 52
  - Struts 2 components
    - declaring 21
    - default set 50
  - Struts 2 Portfolio 20, 44
    - ActionSupport, and 85
    - Ajax client example 205
    - authentication 48
    - authentication interceptor 96
    - basic validation 56
    - building the actions 52
    - custom interceptor 75, 95–96
    - custom validator example 256
    - data transfer, and 62, 109
    - default interceptor stack,
      - and 82
    - dispatching to another
      - servlet 215
    - file upload example 67
    - form prepopulation
      - demo 171
    - i18n example 287
    - introduction 47
    - iterator tag demo 150
    - ModelDriven, and 65
    - organization 49
    - packages 47
    - packaging, and 47
    - properties files 287
    - redirectAction example 222
    - tag demos 142
    - textfield demo 186
    - UI Component demo 171
    - uploading files 69–71
    - use of OGNL 103
    - user-selected locale 303
    - validation framework
      - example 261
    - versions 27
  - Struts 2 tag API 175
    - overview 137–142
    - syntax 138–139
  - Struts 2 tags
    - arrays, and 114
    - attribute types 141
    - cross technology usage 138
    - in image upload form 70
    - OGNL, and 103
    - syntax 138
  - Struts 2.1 319
  - Struts Classic 339
    - migrating from 339–359
  - struts element
    - in declarative architecture 93
    - in XML declarative
      - architecture 32
  - Struts guru 340
  - struts.custom.i18n.resources
    - 295, 345, 354
  - struts.properties 177, 328
    - auto-wiring strategy, and 243
    - i18n, and 295
    - properties files, and 187
  - struts.ui.templateDir 337
  - struts.xml 23, 28, 310, 328
    - from HelloWorld 30
    - i18n, and 295
    - modularizing with
      - includes 31
  - struts-2.0.dtd 92
  - struts2-core.jar 25, 50
  - Struts2InAction.war 26
  - struts2-spring-plugin-2.0.9.jar 239
  - struts-config.xml 349
  - struts-default
    - built-in interceptors 74
    - default interceptors, and 90
    - extending 50–51
    - importance of 51
    - package 25
    - using components from
      - 50–52
  - struts-default package 67–68,
    - 72, 78
    - heart and soul of Struts 2 51
    - interceptor declarations,
      - and 92
    - interceptor stacks, and 90
    - interceptors, and 82
    - results, and 213
  - struts-default.xml 25, 50, 310
    - basic validation, and 55
    - file uploads, and 67
    - interceptor stacks, and 90
    - results, and 213
    - the redirect result, and 220
    - validation framework,
      - and 259
  - struts-plugin.xml 310, 383
  - StrutsTypeConverter 122, 342
  - Strutter 377
  - stylesheet and head
    - component 180
  - substitution marker 364
  - SUCCESS 351
  - Sun 6
  - symbolic name 341
  - synchronized 341, 352
  - syntax
    - FreeMarker 139
    - JSP tags 138
    - OGNL 157
    - Velocity 138
- ## T
- 
- table markup 169, 187
  - tag libraries
    - migrating from Struts 1
      - 343–344
    - switching 343–344
  - taglib directive 138
  - tags 131, 166, 174–178
    - 366–368
    - action tag 148–149
    - ActionContext, and 131
    - attribute types 140, 178
    - attribute usage 141
    - bean tag 145–148
    - categories of 137–142
    - component 333–334
    - control tags 150–151
    - data tags 142–149
    - for FreeMarker 139
    - for JSP 138
    - for Velocity 138
    - i18n tag 154–155
    - id attribute 146
    - if/else tags 151
    - in HelloWorld 33
    - include tag 152–153
    - iterator tag 146, 150–151
    - miscellaneous tags 152–156
    - non-String attributes 140
    - overview 137–142
    - param tag 145, 154, 156
    - parameterizing 148, 156
    - property 104, 140

- tags (*continued*)
  - property tag 142, 147
  - pulling data from
    - ValueStack 107
  - push 137
  - push tag 144–145
  - set tag 143–144
  - setting attributes with
    - OGNL 178
  - String attributes 140
  - syntax 138–139
  - templated 334–335
  - text tag 154–155
  - UI component
    - reference 178–201
  - UI components 167, 175, 201
  - URL tag 153–154
  - usage 138–139
  - ValueStack, and 131
  - var attribute 145–146
  - view layer technology
    - options 138
  - with bodies 144, 150
  - with different results 225
- tail 361
- technological context of
  - Struts 2 3
- technology stack 4–8
- template 176, 335
- template engines 224
- templated tags, leveraging 334–335
- templates 174–178, 335
  - custom 337
  - customizing 176
  - overriding 336
- test-driven development 328–332
- testing 77, 235, 237
- text
  - encoding 8
  - input field 185
  - localizing 287
- text tag 154–155, 289
  - attributes 155
  - retrieving localized text 295
- textarea component 188
- textfield component
  - 169, 185–187
- TextProvider 60, 290–295
  - bundle lookup hierarchy 293
  - getText() 60
  - how it finds bundles 293
  - OGNL, and 290
- ResourceBundle search
  - path 291
  - validation, and 265
- TextProviders and properties
  - files 291
- theme.properties 338
- themes 169, 174, 176–178
  - ajax 177
  - changing 177
  - creating from scratch 337
  - css\_xhtml 177
  - custom 337
  - extending 338
  - simple 177
  - wrapping 337
  - xhtml 177
- third-party frameworks 325
- thread safety and actions 46
- ThreadLocal 98, 215, 312
  - ActionContext
    - implementation 17
- tight coupling 233–235
- tile regions 380
- Tiles 313–315, 360, 379–386
  - as the default result type 384
  - bootstrap 314
  - context parameter 314
  - controllers 383, 385–386
  - inherits 314
  - insertAttribute 382
  - integration 314
  - plug-in 383
  - struts-default 314
- tilesContext.getRequestScope 385
- TimerInterceptor 80, 82
- title 382
- token interceptor 89, 368–369
- tokens 368
  - preventing duplicate form
    - submits 366–369
- tokenSession interceptor 89, 367–369
- tokenStack 367
- tracking web hits 360
- transactional boundaries 248
- transactional classes 253
- transactions
  - duplication 366
  - letting Spring handle it 253
  - manager 248
  - with annotations 253
- transferring data 102
- translation 342
- traverse 346
- type
  - autowiring 243
  - converting 102
  - specifying for map
    - elements 120
  - type converters 104
  - type conversion 16, 63, 101–102, 106, 126, 131, 157
    - Collections, and 111
    - conversions supported out of
      - the box 108
    - customizing 122, 124–126
    - from Java to HTTP 125
    - handling errors 111
    - i18n, and 287, 298
    - introduction 104
    - Java 5 generics, and 117
    - Lists, and 114
    - localized error messages 299
    - Maps, and 119
    - multi-valued request parameters, and 111
    - OGNL, and 103, 122
    - out-of-the-box 108–109
    - primitives, and 109
    - properties file 116
    - rolling your own
      - converters 122
    - specifying types for
      - Collections 116, 120
    - the Struts 2 Portfolio, and 112
    - typed elements 117
    - UI tags, and 173
    - validation, and 111, 259
    - wrappers, and 109
  - type converters 102–104, 107, 126, 342
    - built-in 107–122
    - configuring 124–126
    - custom 102
    - default converters 108
    - global 126
    - how they work 124
    - implementing 122
    - OGNL, and 158
    - property tag, and 143
    - property-specific 124–126
    - tags, and 142
    - wiring a custom
      - converter 125
  - type specification of Collection
    - elements 115
  - TypeConverter interface 122
  - typed elements 116
  - type-specific conversion 118

**U**


---

UI component 167, 176, 182  
 architecture 201  
 common attributes 178  
 convention over  
 configuration 186  
 customization 201  
 customizing 170  
 data binding 170–173  
 functional range 201  
 HTML form elements,  
 and 168–174  
 HTML markup, and 169–170  
 i18n, and 296  
 JavaScript attributes 180  
 label component 198  
 layout markup 184  
 mini-MVC 170  
 name attribute 70, 172  
 OGNL, and 162, 172  
 prepopulation 170, 172  
 prepopulation of  
 checkboxes 190  
 relationship of name and  
 value attributes 173  
 tag reference 178–201  
 tags 175  
 templates 176  
 textfield 169–170  
 underlying FreeMarker  
 templates 175  
 usage 178  
 ValueStack, and 170–173, 195

UI component tags 167–201  
 in HelloWorld 33  
 validation errors, and 59

UI components 12, 201  
 architecture 174  
 bonus components 198–201  
 changing themes 177  
 checkbox component 189  
 collection-backed  
 components 190–198  
 customizing templates 176  
 data transfer 171  
 doubleselect  
 component 199–201  
 error messages, and 173  
 example of preselection of  
 collection backed  
 components 197  
 form component 181–185  
 functional roles 168  
 head component 180

integration with  
 framework 170

internationalization, and 173

introduction 167

key attribute 174

label component 198

layout, and 201

locale, and 304

mini-MVC 168, 174

password component 187

radio component 195–196

ResourceBundles, and 174

reusable customization 174

select component 191–195

simple components 180–190

tags 174

templates 174

textarea component 188

textfield component 185–187

themes 174, 176

type conversion, and 173

underlying FreeMarker  
 templates 168

validation, and 173

UI tags 138  
 advanced usage 336–338  
 common attributes 178–180

underscore 364

unique token 366

unit-testing actions 328–332

uploading 67–72

uploading files 73

URL 154, 182, 364, 371  
 changing in browser 220  
 extension 21  
 generation by form tag 184  
 mapping to Struts 2  
 actions 49  
 pattern for Struts 2 29  
 relative 184  
 servlets, and 8  
 targeted by HTML form 182

URL tag 153–154  
 attributes 153

user authentication in Struts 2  
 Portfolio 48

user interface 9  
 building with tags 167

UserAware 98  
 interface 171

user-defined types and type  
 conversion 102

utilitarian 325

utility interceptors 82

**V**


---

validate() 55, 77, 255

Validateable 55, 77, 85, 258, 280  
 contrast to Validation  
 Framework 86  
 interface 255

validating domain objects  
 271–274

validation 8, 16, 255, 332, 355  
 actions, and 261–267  
 basic 54–60, 258–261  
 declaring 278  
 failure 57  
 i18n, and 298  
 in Struts 2 Portfolio 55  
 redefining 274–277  
 UI tags, and 173

validation context 274–277  
 domain object local  
 metadata 276

validation error 55, 111

validation files, testing 332

validation framework 255,  
 280, 355  
 advanced topics 271–280  
 annotation based  
 metadata 278  
 annotations sample code 280  
 architecture 256–261  
 built-in validators 265–267  
 compared to basic  
 validation 86  
 demonstrated in Struts 2  
 Portfolio 261  
 field validators 263  
 how it works 258  
 i18n, and 265, 287, 297  
 implementing your own  
 validator 269  
 in Struts 2 workflow 258–261  
 metadata 257, 260  
 migrating to Commons  
 Validator 355–357

ModelDriven actions,  
 and 271, 273–280

nonfield validators 264

UI components, and 201

using with basic  
 validation 260

validating domain  
 objects 271–274

validation context, and  
 274–277

- validation framework (*continued*)
    - validation interceptor, and 86
    - validation metadata, and 256, 262
    - validator inheritance, and 277
    - validator short-circuiting 277
    - workflow 258
    - XML metadata for context 275
    - XWork, and 267
    - zero configuration, and 280
  - validation interceptor 86
    - workflow interceptor 86
  - validation logic in basic validation 57
  - validation metadata 256–257, 259
    - annotations 257
    - declaring 262–265
    - domain objects, and 271
    - in action local XML 271
    - in domain object local XML 271
    - XML 257
    - XML example 263
  - validation rules 355
  - ValidationAware 58, 85, 258
    - storing error messages 58
    - validation framework 258
    - validation interceptor, and 86
    - workflow interceptor, and 85
  - Validators 280
    - custom 256
    - interface 267
  - validators 256, 265–267
    - built-in 258, 265–267
    - class level annotations 279
    - convenience classes 268
    - custom 267–271
    - declaring 267, 269–271
    - field and nonfield 263–264, 267
    - inheritance 271
    - inheritance chain 277
    - introduction 257
    - mapping 273
    - parameterizing 267
    - wiring 263
  - validators element in XML
    - meta-data 263
  - validators.xml 269
  - ValidatorSupport 267
  - value attribute 173, 186
    - form fields, and 185
    - key attribute, and 297
    - prepopulation, and 197
  - ValueStack 15, 30, 131, 166
    - accessed by control tags 151
    - accessing from Velocity 224
    - ActionContext, and 135–137
    - as target of OGNL 109
    - as view of domain data 136
    - as virtual object 106, 135–137
    - collection-backed
      - components, and 198
    - data binding, and 172
    - data tags, and 142
    - doubleselect component, and 200
    - dynamic localized text, and 300
    - expression language, and 157–163
    - form prepopulation, and 172
    - framework usage 195
    - i18n tag, and 299
    - i18n, and 287
    - in classic web application 204
    - in data transfer 106
    - in HelloWorld 35
    - include tag, and 153
    - key attribute, and 186
    - managing 144
    - manipulating 148
    - mechanics 35
    - modelDriven interceptor, and 88
    - OGNL, and 132, 137, 157–163
    - param tags, and 264
    - params interceptor, and 83
    - placing objects onto 137
    - pre-selection, and 198
    - programmatically access 207, 217
    - property resolution 106
    - pulling data from 107
    - pulling values from 140
    - pushing objects onto 144–145
    - relationship to
      - ActionContext 132
    - repository of data 132
    - results, and 203, 206, 228
    - role as default root object for OGNL 134
    - role in architecture 16–17
    - storing beans 147
    - tag attributes, and 140
    - tags, and 107, 131
    - TextProvider, and 290
    - UI components, and 170, 191
    - UI tags, and 167
    - validation, and 259, 264
    - working with action 203
  - var attribute 147, 154
  - Velocity 15, 27, 203, 228, 311
    - accessing Struts 2 data 224
    - as view layer option 14
    - documentation 225
    - JAR file 224
    - native expression language 224
    - OGNL, and 224
    - result type 204, 223–227
    - Struts 2 tags, and 175
    - syntax 138
    - tags, and 138
    - using 224
  - VelocityResult 224–225
  - view 12, 202
    - of MVC 14
  - view layer and ValueStack 137
  - view-layer technology 223
    - choices 166
  - views 14–15
  - visitor validator 273
    - context, and 275
- 
- ## W
- wait pages 369–371
  - waiting 369
  - waitPage 370
  - .war file 7
  - WAR, deploying 26
  - web application framework
    - architectural solution 10
    - automation of common tasks 10
    - reasons to use 11
  - web applications 4, 6–9, 215
    - building 4
    - deploying 26
    - directory structure 27
    - domain tasks 8
    - frameworks 9–11
    - namespace 7
    - requirements 27
    - technological context 4
    - zero-configuration 13
  - web browser 5
  - web features 363

- web pages 352
    - creating with tiles
      - controllers 385–386
      - migrating from Struts 1 352–354
  - web.xml 28, 217
    - annotations, and 36
    - configuring Spring 239
    - for sample application 28
    - Spring, and 249
  - WEB-INF 27
  - website 379
  - WebWork 11
  - widget 168
  - wildcards 361, 368
    - mapping 366
    - substitution 365
  - wiring your application
    - components 21
  - wizards 89
  - WML 354
  - workflow interceptor 77, 84, 258
    - checking for errors 260
    - checking for validation errors 58
    - declaration 55
    - parameters 85
    - returning user to input page 59
    - role in validation 54
    - source code 85
    - two phases of 260
  - workflow navigation 343
  - workflows 9, 365
    - dynamic 365
  - wrapper classes 109–111
  - wrapper types 127
  - WW\_TRANS\_I18N\_LOCALE 304
  - www.ognl.org 344
  - www.strutsschool.com 361
- X**
- 
- XHTML 169
  - xhtml theme 176, 181, 185
  - XML 12
    - Ajax applications, and 203, 205
    - configuration, and 22–23
    - declarative architecture, and 22, 43
    - document structure 92
    - for declarative architecture 13
    - reduction of 24
    - result type for Ajax 212
    - Spring, and 239
    - validation framework, and 257
    - validation, and 281
  - XML-based configuration 13
  - XML-based declarative architecture 31
  - XMLHttpRequest 209, 211
  - XSLT 15, 354
  - XStream 208
  - XWork 355
    - validation, and 267
  - xwork.default.invalid.fieldvalue 299
  - xwork-conversion.properties 126
- Z**
- 
- zero configuration 13, 25
    - validation annotations, and 280