

## Symbols

---

@At annotation 298  
@Autowired 199  
    annotation on constructor 34  
@EJB 199  
@Entity 301  
@Get 298  
@GuardedBy 198  
@Immutable 246, 260–261  
@Inject 199  
    @Autowired in place of 35  
    skip annotation 93  
@On 298  
@PostActivate 200  
@PostConstruct 199  
@PreDestroy 199  
@PrePassivate 200  
@Remove 199  
@Repeat 299  
@RequestScoped 298, 300, 303  
@ScopingAnnotation 163  
@SessionScoped 303  
@Singleton 132, 145  
    missing 172  
@ThreadSafe 168, 246  
@Transactional 211, 222

## A

---

Abstract Factory pattern. *See*  
    Factory pattern  
abstract identity  
    referred to as a key 37  
Acegi Security 149

ACID transactions 157  
Adapter pattern 95, 119,  
    175, 180  
Adobe Flex 320  
advice 212  
    avoid too much 219  
Algorithmic complexity 131  
Amazon 198  
annotations  
    feature of Java 26  
Ant 226  
anti-pattern 196, 271  
    black box 276–280  
    fragile base 277  
    lifecycle 276  
    rigid configuration 271  
    singleton 135, 276  
    type unsafe configuration 271  
    unwarranted constraints 273  
AOP 16, 312  
    Guice's approach 18  
    intercepting methods 211  
    interceptor 311  
    make deferred  
        modifications 234  
    use caution when using 221  
AOP injection  
    *See also* method decoration 62  
AopAlliance 18, 218  
    and Guice 214  
Apache 269  
Apache Avalon  
    earliest DI library 17  
    interface injection 61  
Apache CXF 280

Apache Excalibur  
    Fortress 17  
Apache Hadoop  
    cluster-computing 184  
Apache HiveMind 18–19, 269  
Apache James 17  
Apache OpenJpa 225  
Apache Tapestry 18, 269  
Apache Tomcat 290  
Apache Wicket 269  
application logic 51  
    defined 51  
    separate from infrastructure  
        logic 52  
application server 198, 200  
AspectJ 18  
    build-time weaving 234  
    pointcut 215  
    Springs library 214  
aspect-oriented programming.  
    *See* AOP  
Assisted Injection pattern 86  
AssistedInject. *See* Assisted  
    Injection pattern  
atomic operations 263  
atomicity 157  
autowiring  
    constructor 34

## B

---

BCS  
    instantiation modes 316  
    not an XML format 315  
Bean Validation 280

BeanFactoryAware 84  
 behaviorally consistent 109  
 Beust, Cedric 137  
 binding annotations 255  
 bindings  
   determine type 43  
   keys provide 37  
   wrong type 43  
 black box  
   anti-patterns 276  
   vs. encapsulation 276  
 boilerplate  
   removing 181  
 bootstrap  
   combined with  
     configuration 32  
 bootstrapping  
   the injector 22, 42  
 Builder pattern 16, 88  
 Butterfly Components 315  
 Butterfly Container 315  
   bean scopes 316  
 Butterfly Container Script. *See*  
   BCS

---

**C**

C# 14, 19, 46, 83  
 C++ 19  
   destructor 189  
   solves general problems 16  
 cache scoping. *See* singleton  
 caching  
   in-memory 181  
 Castle MicroKernel 19  
 CGLib  
   Guice and Spring 218  
 circular initialization  
   no injection solution 77  
 circular reference 71  
   solve with proxies 75  
   two solutions 72  
 circularity 71  
   breaking 73  
 class  
   testability 4  
 class name  
   fully qualified 273  
 classloader 217  
 classpath 274  
 client  
   definition 3  
   object is 2  
   testing 9  
   verify behavior 10

client code  
   not invoking injector 24  
 clients 198  
 Closeable 197  
   definition 197  
 cluster computing 184  
 coarse grained 157  
 code  
   behaviorally focused 22  
   modular design 22  
   separation of infrastructure  
     and application 15  
   testability 22  
   testable 100  
   thread-unsafe 113  
 combinatorial bindings  
   build in Spring.JavaConfig 50  
 combinatorial keys 47, 161  
   benefits 48  
   drawback 48  
   Guice 49  
 complexity  
   algorithmic 131  
   linear 131  
 concurrency 10, 246, 257–265  
   vs. synchronization 262  
 concurrency problems  
   from scope widening 176  
 concurrent 136  
 concurrent access  
   to counter 178  
 configuration  
   combined with bootstrap 32  
 confined to single threads. *See*  
   thread confinement  
 connection pool 195  
 connection validation 195  
 consistency 157  
 ConstraintFactory. *See* JSR-303  
 construction by hand 5, 10  
   creating injector 28, 32  
   DI enables testability 13  
   manual DI 13  
   problems 6  
   toothpaste example 124  
   why so named 6  
 constructor  
   autowiring 34  
   initialization hook 56  
   purpose 55  
 constructor injection 74, 128  
   advantage 6  
   circular reference  
     problem 71

    contextual 85  
     dominant form of idiom 65  
     first preference 81  
     leverages immutability 67  
     object validity 78  
     vs. setter injection 55, 66  
 constructor pyramid 69–71  
 constructor wiring 29  
   PicoContainer 33  
 context objects  
   set when needed 91  
 contextual injection  
   Builders 90  
   variation of partial  
     injection 84  
 contract  
   revelation of intent 110  
 cookie 154  
 Copland 19  
 crosscutting concern 210  
   logging 210  
 crosstalk 290  
   layers 290  
   modularity and services  
     coupling 295  
   presentation layer 296  
   requirements 290  
 custom annotations 283  
 custom scope 157–166  
   in Guice 160  
   registering in Guice 163  
   Spring 164

---

**D**

DAO 38, 247  
 Data Access Object. *See* DAO  
 data type  
   scalar 250  
 database 111, 123, 224, 292, 312  
   connection pool 129  
   map object to row 224  
   Oracle 111  
   pool of connections 130  
   PostgreSQL 111  
 Db4objects  
   warp-persist 222  
 decorate. *See* Decorator  
 Decorator Design pattern 62, 16  
 delayed injection. *See* partial  
   injection  
 delegator 233  
 Delegator pattern 234

- dependencies 24
  - half-constructed 67
  - identifying for injection 36
- dependency 3
- dependency graph. *See* dependency
- dependency injection framework. *See* dependency injector
- dependency injection. *See* DI
- dependency injector 13, 15
  - features 37
- dependent 3
- deployment profiles 118
- destroy lifecycle hook 196
- destruction
  - times and frequency 191
- Destructor anti-pattern 196
- discourage use of 197
- DI 1
  - DSL used in 16
  - helps loose coupling 111
  - infrastructure code 111
  - manual 13
  - not an invasive technique 28
  - not explicitly knowing 13
  - popular use 15
  - real world use 17
  - scoping 25
  - vs. IoC 15
- DI library 52, 55
  - configuring the injector 26
  - method decoration
    - support 82
- DI solutions
  - fragmentation 267–270
- direct field injection
  - difficult to test 269
- documentation 110
- domain model 296, 301
- Domain Specific Language. *See* DSL
- domain-specific scope 139
  - building web applications 140
- downcast
  - defined 28
  - returned instance 32
- downcasting
  - Provider pattern 83
- DSL 16
  - for dependency injection 315
  - injection-oriented 26
  - solves specific problems 16
- duck-typed 19

- durability 157
- Dynamic Finders 308
- dynamic proxies
  - cannot intercept private methods 231
  - See also* proxy

## E

---

- eager instantiation 129
  - vs. lazy instantiation 201
- EasyMock 114–115
- EJB 12, 114, 182
  - does not support constructor injection 199
  - lifecycle hook 200
  - stateful 198
  - stateful session 269
- Emailer
  - change behavior 5
- encapsulation 5, 14, 252
  - class member 30
  - destroyed 253
    - at package and module level 257
  - principles violated 7
  - vs. black box 276
  - warp-persist 256
- Enterprise Java Bean. *See* EJB
- EntityManager. *See* JPA
- escape
  - during construction 245
- execution context 25

## F

---

- Factory
  - equivalent of singleton
    - scoping 136
- Factory chain 316
- Factory pattern 7–11, 24
  - applied to Emailer 7
- faulty configuration 67
  - causes heartache 68
- field injection 65
- filter
  - traces requests 268
- FilterToBeanProxy 149
- final fields
  - attempt to modify 246
- finalization 189
  - domain-specific patterns 197

- finalizer
  - can come in handy 190
- finite resources
  - disposal of 189
  - freeing 196
  - modeling as Closeables 197
- Fortress 17
- Fowler, Martin 15
- fragile base class anti-pattern 277
- framework
  - lessons for designers 270–280
- framework design
  - replacement with mocks
    - essential 270
- fully qualified class name 273
- functionality
  - changing in base class 277
  - share 277

## G

---

- Gang of Four 88, 95, 128
- garbage collection 189
  - finalizer before collector
    - runs 189
    - leads to memory leaks 179
- garbage collector. *See* garbage collection
- Gin 18
- Gmail 2
- good key behavior 46
  - rules 47
- Google 12
- Google AdWords 18
- Google Guice. *See* Guice
- Google Sitebricks 269
  - configuring 294
- Google Web Toolkit. *See* GWT
- graphical user interface. *See* GUI
- grid
  - computing 183
  - Oracle Coherence 182
  - sometimes called cluster
    - cache 182
- grid scope 182
- GUI 267
- Guice 294–295, 320
  - approach to AOP 18
  - binding scopes 147
  - bindings 45
  - binds under no scope 131
  - CGLib 218
  - combinatorial keys 49
  - compact setter injection 59

Guice (*continued*)  
 create Emailer 23  
 custom scope 160  
 doesn't react to  
   @Immutable 168  
 as injector 290  
 Jabber 254  
 key as binding 29  
 key as identifier 29  
 modules defined 30  
 nomenclature 127  
 Providers out of the box 83  
 request scoping 144  
 scopes 124  
 servlet. *See* guice-servlet  
 setter injection 87  
 singletons 246  
 testing matchers 213  
 transaction scope 161  
 warp-persist 222  
 guice-servlet 25, 222, 291, 293  
 GWT 18

## H

---

Half-Life 43  
 Hammant, Paul 15  
 hashtable 241, 244  
   no locking 264  
   synchronized wrapper  
     around 264  
   thread-local 162  
 Hellesøy, Aslak 15  
 Hibernate 266, 290–291  
   JPA 225  
   warp-persist 222  
 Hibernate Query Language. *See*  
   HQL  
 Hollywood Principle 15, 138  
   scopes apply to state of  
     objects 181  
   *See also* IoC  
 HomePage  
   sign out link 304  
 HQL 309  
 HSQL  
   database 312  
   *See also* Hypersonic  
 HTML  
   web applications 140  
 HTTP  
   request scoped 251  
 HTTP filter  
   with Spring Security 225

HTTP protocol  
   stateless 154  
 HTTP request 124, 276  
   scope 141  
 HTTP session scope 149  
 Hypersonic 290, 312

## I

---

IDE  
   helps catch errors 45  
 identifier  
   combinatorial key 36  
 identifying by type 44  
   limitations 46  
 immutability 246  
   pitfalls 258  
 immutable  
   good design 167  
 immutable dependencies  
   create 66  
 in-construction  
   cannot break cycle 75  
 in-construction problem  
   in a nutshell 76  
   setter injection 76  
 infrastructure code 116  
   separating by area 102  
 infrastructure logic 15, 51  
   separate from application  
     logic 52  
 inheritance  
   replacing with delegation 279  
 initialization 196  
   called on 199  
   times and frequency 191  
 injection  
   scope widening 145, 169  
 injection idiom  
   choosing 65  
 injector  
   all-purpose 24  
   bootstrapping 22  
   configuration 26  
 injector configuration  
   changing at runtime 118  
   modifying 100  
 integration tests 112, 116  
   reveal configuration  
     quirks 118  
   for web application 117  
 IntelliJ IDEA 42, 292  
 intercepting methods 211  
 interception  
   lose 87

  modifying behavior 37  
   use cases for 221, 234  
 interceptor 214, 217  
   bound 213  
   with Guice 212  
   with Spring 214  
 interceptor chain 221  
 interface injection 60  
   pro and con 62  
 invasive technique 28  
 Inversion of Control. *See* IoC  
 IoC 1, 14–15  
   container 15  
   several meanings 15  
 isolation 157

## J

---

Jabber 253  
 Java 14, 46, 83, 197, 291  
   annotations 26  
   atomic library 263  
   birthplace of DI 17  
   C# and .NET 19  
   class member  
     encapsulation 30  
   Closeable interface 197  
   finalizer 189  
   garbage collector 179  
   hashtable 241  
   mock objects frameworks 114  
   no proxy for classes 218  
   package-local visibility 232  
   package-privacy 254  
   proxying interfaces 216  
   semantics 44  
   servlet filter 144  
   Set 38  
   statically typed 42  
   Strings immutable 261  
   volatile keyword 168  
 Java Community Process  
   Bean Validation 280  
 Java EE 17, 198  
   IoC 15  
 Java library  
   binary-tree 38  
   hash-table 38  
 Java Memory Model 79  
 Java Naming and Directory  
   Interface. *See* JNDI  
 Java Persistence API. *See* JPA  
 Java Servlet  
   registration of servlets and  
     filters 267

Java Servlet Framework 191  
   detect browser capability 154  
 Java Servlet Specification 145  
 Java Swing 196  
 Java Virtual Machine. *See* JVM  
 java.lang.reflect. *See* reflection  
 java.lang.reflect.Proxy. *See* proxy  
 java.util.HashMap. *See* hashtable  
 JavaConfig 45  
 JavaServer Faces. *See* JSF  
 JBoss Seam 19  
 JDK Proxy. *See* proxy  
 JMock 114  
 JNDI 12  
 Johnson, Rod 15, 17  
 JPA  
   warp-persist 222  
 JSF 269  
 JSR-303 280  
   Bean Validation 280  
   doesn't restrict plug-in  
     code 284  
 JUnit 137  
 JVM  
   manages threads 169

## K

---

keys  
   bound improperly 13  
   class literal 46  
   combinatorial 47  
   good behavior 40, 43, 46  
 keyword  
   final 233

## L

---

language  
   duck typed 19  
   dynamically typed 42  
   object-oriented 241  
   statically typed 42  
 latent state 124  
 lazy instantiation 201  
   vs. eager instantiation 201  
 leaking  
   of semantics 252  
 Lee, Bob 18  
 lexical ordering 242  
 libraries  
   provide pluggable  
     services 271

lifecycle 312  
   basic events 187  
   closely related to scope 187  
   customizing 202  
   customizing with  
     multicasting 205  
     customizing with  
       postprocessing 202  
   domain-specific 191  
   events 191, 205  
   hook 191  
   notification 203  
   notifying of events 37  
 lifecycle constrained anti-  
   pattern 276  
 lifecycle events  
   not universal 191  
 lifecycle hook  
   @PrePassivate 200  
   leads to unintuitive  
     designs 196  
 lifecycle scenarios  
   servlets vs. database  
     connections 191  
 linear complexity 131  
 locking 245  
 logging 212, 229–230  
 loose coupling 16, 270  
   example 107  
   with DI 111

## M

---

manual dependency injection.  
   *See* construction by hand  
 matcher 220  
 memory complexity 129, 131  
 memory leaks  
   and scope-widening 179  
 memory reclamation 179  
 meta-annotation 163  
 metadata  
   externalized 93  
   injector configuration 26  
 method decoration 82  
   DI library support 82  
   fraught with pitfalls 64  
   variation on DI 62  
 method interception 214  
   enhance 220  
 methods  
   intercepting 211  
 mock objects 8, 114  
   useful 103  
 Mockito 114

modularity 270  
 Module 30, 213  
 modules 118  
   defined 100  
   separation and  
     independence 101  
 Mort Bay Jetty 290, 313  
 Mozilla Thunderbird 2  
 multicasting 205  
   proxy 207  
 mutable singletons 168  
 mutual exclusion 178

## N

---

namespaces  
   use of 38  
 NanoContainer 18  
 NASA Mars Rover 12  
 native method 190  
 .NET 19  
 network sockets 197  
 Nintendo Wii 43  
 no scope 125–128  
   vs. singleton 133  
 no-scoped 170  
 nullary constructor 273, 275

## O

---

object  
   good citizen 66  
   out-of-scope 170  
   role of 100  
 object creation 187  
 object destruction. *See* finaliza-  
   tion  
 object graph 69  
   assembly 36  
   building 4  
   changing 13  
   complex 47  
   described in one place 111  
   description 3  
   emphasis on unit testing 21  
   Factory pattern 7  
   how to construct 6  
   offload burden 5  
   requires new factory 11  
   size and scope 130  
   structure may change 119  
   system of dependencies 3  
   tightly coupled 104

object/relational mapping. *See* ORM  
 object-oriented languages 157  
 object-oriented programming.  
*See* OOP  
 objects  
   relationship between 3  
 OOP 2  
   encapsulation 252  
   tight coupling 102  
   using constructor injection 6  
 Oracle 111, 157  
 Oracle Coherence 182, 266  
 Oracle TopLink  
   JPA 225  
 ORM tools  
   Hibernate 266  
 OSCache 271  
   uses reflection 273  
 out of scope 123, 139  
 out-of-container. *See* integration  
   test  
 OutOfMemoryError 180  
 out-of-scope 170  
   between scope contexts 175

## P

---

package-local 232, 296  
 partial injection 81  
   contextual injection 84  
 pattern 127  
 patterns  
   Abstract Factory 7  
   Adapter 95, 119, 175, 180  
   Assisted Injection 86  
   Builder 16, 88  
   Decorator 16  
   Decorator Design 62  
   Delegator 234  
   Factory 7–11, 24  
   Provider 82, 127, 161,  
     180, 308  
   *See also* individual pattern  
   names  
 persistence 157, 272–273,  
   291, 296  
 persistence.xml. *See* JPA  
 PicoContainer 14–15, 19  
   all-purpose destroy  
     method 196  
   bias toward constructor  
     injection 18  
   cache scoping 129  
   greedy 33

  identifying by type 46  
   injection 31  
   locking mode 245  
   multicasting 206  
   mutable injector 119  
   no scope 128  
   prefers constructor wiring 33  
 PicoContainer 2.0 32  
 PicoContainer Gems 206  
 pointcut 215, 226–227, 276  
   AspectJ 215  
 PostgreSQL 111, 157  
 postprocessing 202  
 power of scopes  
   leveraging 181–184  
 precompilation 19  
 presentation layer 248–249, 296  
 programmatic  
   configuration 280  
 programming to contract  
   108, 113  
 prototype scope  
   Spring's name for no  
     scope 127  
 Provider pattern 82, 127, 161,  
   180, 308  
   implemented in Spring 164  
 proxy 230, 312  
   of class can be powerful 219  
   pitfalls 228  
   powerful technique 212  
   protected methods 232  
   resolving circular  
     references 74  
 Python 19

## R

---

Rahien, Ayende 16  
 RDBMS 118  
 rebinding 118  
   closely tied to scope 119  
   with adapter 120  
 refactoring  
   possible cost 106  
 reflection 273  
 reinjection 81  
   method decoration 82  
   with Provider pattern 82  
 reinjection problem 161  
   scope widening 173  
   use mitigants 175  
 relational database 157  
 Remote Enterprise Java Bean.  
   *See* EJB

request scoped 251  
   each instance unique 150  
   objects not  
     multithreaded 149  
 Request scoping  
   in Spring 147  
 request scoping  
   in Guice 144  
 request-scoped. *See* request  
   scope  
 resource bundle 271  
 revelation of intent 108  
 rigid configuration anti-  
   pattern 271  
 role interface  
   uses 61  
 root object  
   obtained from injector 25  
 Ruby 19

## S

---

safe publication 241  
   guarantee of order 244  
   subtlety 243  
   synchronization 244  
 scalar data type 250  
 scope 79–80, 87, 124–125  
   apply Hollywood  
     Principle 181  
   closely related to lifecycle 187  
   defined 123  
   domain-specific 139  
   general purpose 123  
   HTTP request 141, 157  
   HTTP session 149, 157  
   managing state 37  
   no scope 123  
   singleton 123  
   thread-confined 158  
 scope widening 172  
   form of reinjection  
     problem 173  
   and memory leaks 179  
   with thread-safety 179  
 scope-widening 170, 179, 241  
   domestic problems 176  
 scope-widening injection 145,  
   169–180  
   not caught with unit test 173  
 scoping 270  
   annotation 153  
   functionality 156  
   powerful feature of DI 25

- sealed code
    - Guice injects 93
    - injecting objects in 92
  - security 296, 311
  - service
    - identity of by key 13
  - service clusters
    - Oracle Coherence 266
  - Service Location pattern 25
  - Service Locator
    - JNDI 12
    - pass it a key 12
    - See also* Service Locator pattern
    - typical use 12
  - Service Locator pattern 12–13
  - service-client
    - relationship 3
  - services 295, 304
    - database-specific 141
    - decomposing into objects 3
    - definition 3
    - objects as 2
    - prefixes 38
  - services-oriented architecture.
    - See* SOA
  - servlet
    - filter 144
    - lifecycle stages 191
    - restrictive structure 268
  - servlet lifecycle
    - DI in conjunction with 193
  - session bean
    - stateful 198
  - session scope
    - each instance shared 150
    - HTTP 149
    - scoping annotation 153
  - setter injection 54, 56, 69, 168
    - can't leverage
      - immutability 67
    - common idiom in use 60
    - dependencies wired 78
    - dominant form of idiom 65
    - false negative 68
    - no additional code 70
    - vs. constructor injection
      - 55, 66
  - setter method 54, 136, 250
  - Ship, Howard Lewis 18
  - single-sign-on semantics 182
  - singleton 129
    - abused as a bridge 170
    - litmus test 130
    - must be thread-safe 167
    - mutable 168
    - scoping 160
    - when reclaimed 179
  - singleton anti-pattern
    - 135, 138, 276
    - See also* singleton objects
  - singleton objects
    - make immutable 168
  - Singleton pattern 10
  - singleton scope 128
    - advantages over no scope 129
    - context is injector 128
    - vs. no scope 133
    - vs. singleton objects 135
  - singleton scoping
    - pool of database connections 130
  - singleton-scoped objects
    - cached 201
    - eager instantiation in Spring 202
  - Sitebricks
    - as web application framework 290
  - SmartyPants 320
    - supports live injections 321
  - SOA 130, 183
  - source code generation 19
  - specification of behavior 108
  - spellchecker
    - creating one in French 8
  - Spring 14, 19, 83, 269
    - @Autowired 27
    - all-purpose destroy method 196
    - autowiring 34
    - bean tag 268
    - beans are singletons 131
    - binding scopes 147
    - CGLib 218
    - check spelling 26
    - constructor injection 17
    - constructor wiring 29
    - custom scope 164
    - IntelliJ IDEA 42
    - JavaConfig 45
    - lifecycle support method 164
    - make aware of autowired classes 35
    - namespaces 39
    - prototype scope 127
    - request scoping 147
    - scopes 124
    - setter injection 57, 72, 93
    - XML configuration 27
  - Spring 2.5 34
  - Spring Framework 17
  - Spring MVC 193, 269
  - Spring Security 149, 228, 270
    - requires HTTP filter 225
    - securing methods 224
    - uses Ant-style paths 226
  - Spring XML 104
    - binding explicit 29
  - SpringMVC 270
  - SQL 195, 290
  - Stage 201
  - stateful
    - context 156
  - stateful behavior
    - a peril 174
  - static methods
    - proxies cannot intercept 230
  - string identifier. *See* string keys
  - string keys 29
    - advantage over plain type 47
    - choose wisely 40
    - flexible but unsafe 47
    - identifying by 37
    - limitations 42
  - StructureMap 14, 19
  - Struts2 18, 269
  - Swing 196
  - synchronization 244–245
    - vs. concurrency 262
  - synchronized keyword 178
- ## T
- 
- TCP connection 130
  - testability 4
    - very important 270
  - testable 104
  - testable code 100
    - crucial to writing 112
  - testing. *See* testability
  - TestNG 137
  - tests
    - automated unit 112
    - integration 112, 116
  - thread confinement 163
  - thread of execution 158
  - thread-local
    - hash table 162
  - thread-safe
    - in a class 169

- thread-safe classes
  - creating 167
- thread-safety
  - eternally difficult 179
  - scope-widening 179
- thread-unsafe 178
- tight coupling 104
  - perils 102
  - refactoring impacts 105
- time complexity 131
- Tirsen, Jon 15
- tracing interceptor
  - with Guice 212
  - with Spring 214
- transaction
  - ACID 157
  - specific context 158
- transaction completes
  - commit 162
  - rollback 162
- transaction scope 158, 181
  - apt fit 160
  - in Guice 161
- transaction scoped
  - object sought outside a
    - transaction 163
  - wired 163
- Tweet 300
- Twitter 290
- type
  - identifying by 44
  - resolution 42
  - specific class of data 42
- type keys 46
  - refers to wrong type 45
  - safe but inflexible 47
- type literals
  - analogous to string literals 44
- type unsafe configuration anti-pattern 271
- type-safety
  - disregard for 271

---

**U**


---

- unit test 109, 112
  - and interception 236
  - first line of defense 118
  - injector configuration 114
  - single concern 113
  - violated rules 117
  - writing 4
- unwarranted constraints anti-pattern 273
- URL rewriting 154
- use cases for interception 234

---

**V**


---

- variant 49
  - using spellcheckers 39
- viral injection 25
- volatile 177
  - keyword 168

---

**W**


---

- warp-persist 222, 256
  - @Transactional 222
  - Dynamic Finders 308
- weaving 212
- web applications 140
  - building practical 124
- Web Services Description Language. *See* WSDL
- WebWork 269
- wiring 245
  - absence of directive 35
  - after instance is
    - constructed 56
    - common forms 55
    - construction order 74
    - dependencies 55
    - repeat code 6
    - satisfies circularity 71
- WSDL 320

---

**X**


---

- XML 94
  - boilerplate schema 28
  - injection in Spring 27

---

**Z**


---

- zombie connection 195
- zombie objects
  - unreclaimable 179