

Inhaltsverzeichnis

Vorwort	XI
----------------	-----------

1 Java 6 Basics	1
1.1 Enumerationen	2
1.2 Import statischer Member	4
1.2.1 Das Ende eines Anti-Patterns	4
1.3 Interfaces vs. Klassen	5
1.3.1 Typ vs. Implementierung	7
1.4 for-each Schleife	8
1.5 Variable Anzahl von Argumente	10
1.6 Autoboxing	12
1.7 Überblick über Mustang-Features	13
1.8 Neue I/O-Methoden	14
1.8.1 File	15
1.8.2 Console-I/O	17
1.9 Floating-Point Essentials	19
1.9.1 Floating-Points vs. Reelle Zahlen	19
1.9.2 Neue Konstante und Methoden in Java 6	24
1.9.3 Floating-Point Constraints	27
1.10 Typsichere Array-Kopien	29
1.11 Erweiterung des Collection-APIs	32
1.11.1 API-Überblick	32
1.11.2 Utility-Klasse Collections	36
1.11.3 Queue, Deque und ConcurrentStack	38
1.11.4 Konsistenz von hashCode(), equals() und compare()	43
1.11.5 Navigierbare Container	49
1.11.6 ConcurrentSkipListSet	53
1.11.7 ConcurrentMap	55
1.11.8 CopyOnWrite	56

1.12	Fazit.....	58
1.13	Referenzen.....	58
2	Generische Techniken.....	59
2.1	Generics Grundlagen.....	61
2.1.1	Generisches Anti-Pattern.....	61
2.1.2	Klassische Design-Prinzipien	63
2.1.3	Generics vs. Object.....	65
2.1.4	Eine einfacher Typ-Variable <T>.....	65
2.1.5	Einschränkungen für <T>.....	68
2.1.6	Generische Methoden.....	71
2.1.7	Typ-Inferenz	73
2.2	Arrays und Typ-Hierarchien mit Generics.....	75
2.2.1	Arrays vs. generischen Listen.....	75
2.2.2	Invarianz.....	77
2.3	Wichtige generische Begriffe.....	80
2.4	Restriktionen zu Generics.....	84
2.4.1	Unerlaubte generische Ausdrücke	84
2.5	Finden von Typ-Einschränkungen.....	88
2.5.1	Generics, ein statisches Konzept.....	92
2.6	Wildcard-Ausdrücke.....	94
2.6.1	Syntax und Eigenschaften von Wildcards.....	94
2.6.2	Ungebundene Wildcards.....	96
2.6.3	Wildcards mit extends.....	100
2.6.4	Wildcards mit super.....	102
2.6.5	Finden aktueller Typen mit Wildcards.....	104
2.6.6	Wildcard Capture.....	106
2.7	Covariant Return beim Overriding.....	107
2.8	Overriding mit covarianten Parameter-Typen.....	109
2.9	Overloading bei gleicher Signatur.....	113
2.10	Type-Token Varianten.....	114
2.10.1	Class-Token Idiom.....	115
2.10.2	Anlage eines Arrays zum aktuellen Typ.....	118
2.11	Das Self-Type Idiom.....	120
2.11.1	Self-Types im Einsatz.....	120
2.12	Enumerationen.....	126
2.12.1	EnumSet.....	129
2.12.2	EnumMap.....	130

2.12.3 Anonyme Klassen in enum.....	134
2.13 FAQ.....	137
2.14 Fazit.....	147
2.15 Referenzen.....	148
3 Reflektive Techniken.....	149
3.1 Überblick.....	149
3.1.1 Compilieren & Linken: Java vs. C/C++.....	150
3.1.2 Laden via ClassLoader.....	151
3.1.3 Sicherheit und Performanz.....	153
3.1.4 Zentrale Aufgaben von Reflexion.....	153
3.1.5 Reflexion-API ist read-only.....	154
3.2 Klassen-basiertes Reflection-API.....	155
3.2.1 Klassen-Modell.....	155
3.2.2 Ermitteln einer Class<T>-Instanz.....	156
3.3 Class & Member.....	158
3.3.1 Warnungen unterdrücken: asSubClass() und cast().....	159
3.3.2 Neue Class-Methoden isX().....	162
3.3.3 Namenskonventionen zu Class-Methoden getX().....	165
3.3.4 Getter für Klassen/Interface-Namen	166
3.3.5 Getter für Klassen/Package-Beziehungen.....	167
3.3.6 Getter für Felder, enum-Konstante.....	170
3.3.7 Getter für Methoden.....	172
3.3.8 Getter für Konstruktoren & statische Methoden.....	174
3.3.9 Modifier, Member, AccessibleObject.....	176
3.3.10 Klasse Field.....	178
3.3.11 Klasse Constructor, Initialisierung.....	181
3.3.12 Klasse Method.....	184
3.3.13 Klasse Array.....	188
3.3.14 Problemfall primitiven Typen.....	192
3.4 Interface-basiertes Reflexion-API.....	194
3.4.1 Type	196
3.4.2 GenericDeclaration, TypeVariable, Field.....	196
3.4.3 Generische Methoden in Class, Method & Constructor.....	199
3.4.4 ParameterizedType, WildcardType, GenericArrayType.....	202
3.4.5 Super Type Token.....	206
3.5 Fazit.....	211
3.6 Referenzen.....	211

4	Annotations-Techniken.....	213
4.1	Überblick.....	214
4.1.1	Standard-Annotationen in Package java.lang.....	216
4.2	Deklaration und Einsatz.....	219
4.2.1	Restriktionen & Beispiele.....	221
4.3	Meta-Annotationen.....	227
4.4	Allgemeine Standards.....	233
4.4.1	Richtlinien für Annotations-Gültigkeit.....	234
4.5	Core-Annotationen.....	235
4.5.1	Defekt-sensitive Annotationen.....	235
4.5.2	Standard-Annotationen in Java 6.....	238
4.6	Modifizieren bestehender Annotationen	240
4.7	Laufzeit-Zugriff auf Annotationen.....	241
4.8	Runtime-Auswertung von Annotationen.....	243
4.9	Validieren mit Hilfe von Annotationen.....	246
4.9.1	Validierung, Test-Frameworks & AOP.....	246
4.9.2	Ein minimales Validierungs-Framework.....	247
4.9.3	Test und Analyse der Validierung.....	252
4.10	Implementierung von Annotationen.....	254
4.11	Annotations-Prozessoren.....	259
4.11.1	Annotation-Prozessor-API.....	260
4.11.2	Zugriff auf Programm-Elemente.....	265
4.11.3	Restriktionen für Prozessoren.....	267
4.11.4	Decorator-Style.....	268
4.11.5	Factory-Einsatz.....	269
4.11.6	Prozessor-generierte Validierung.....	270
4.12	Annotation vs. Konfiguration.....	276
4.13	Fazit.....	276
4.14	Referenzen.....	277
5	Service & Component-Techniken.....	279
5.1	ClassLoader-Techniken.....	280
5.1.1	Namensräume.....	280
5.1.2	Struktur der Klasse ClassLoader.....	282
5.1.3	ClassLoader Delegation & Hierarchie.....	283
5.1.4	SecureClassLoader & Berechtigungen.....	286
5.1.5	URLClassLoader.....	289

5.1.6 Implizites/explizites Laden & Versions-Management.....	293
5.1.7 Thread Context-ClassLoaders.....	297
5.1.8 Child-First ClassLoader.....	302
5.1.9 Spezielle ClassLoader.....	307
5.1.10 Super-Packages und Module (Java 7).....	310
5.2 Services – ein erster Überblick.....	313
5.3 Persistenz & JDBC 4.....	314
5.3.1 Persistenz-Alternativen.....	314
5.3.2 Persistenz mittels einer RDBMS.....	315
5.3.3 JDBC 4.....	317
5.3.4 Derby.....	318
5.3.5 Neue JDBC 4 Features.....	320
5.3.6 Query, DataSet<T>, @Select und @Update.....	321
5.4 Das DAO-Pattern am Beispiel.....	327
5.4.1 Beispiel Student-Kurs-Beziehung.....	327
5.4.2 DAO Design-Pattern.....	330
5.4.3 Generisches DAO-Interface.....	331
5.4.4 EoD mit generischen DAO.....	333
5.5 Automatisches Laden von Services.....	344
5.5.1 Klasse ServiceLoader.....	345
5.6 Dynamisches Proxy.....	350
5.6.1 Proxy API.....	350
5.6.2 Proxies als Adapter.....	356
5.6.3 Das Decorator-Pattern	363
5.6.4 Proxy als Decorator.....	365
5.6.5 Duck Typing – in Java.....	371
5.7 JMX – Java Management Extensions.....	376
5.7.1 JMX-Überblick.....	376
5.7.2 Arten von MBeans.....	377
5.7.3 Standard MBean.....	378
5.7.4 MXBeans.....	385
5.8 Fazit.....	392
5.9 Referenzen.....	393
 6 Compiler & Scripting Techniken.....	 395
6.1 Das Compiler-API.....	396
6.1.1 Javac Mimikry.....	396
6.1.2 Überblick über das Package javax.tools.....	398
6.1.3 Compilierung mit Fehlern und Diagnose.....	401
6.1.4 Setzen der Pfade mit StandardLocation.....	403

6.1.5	DiagnosticCollector und String-Source.....	404
6.1.6	Der mühsame Weg zur In-Memory Class.....	407
6.2	Das Scripting API.....	415
6.2.1	Dynamisch vs. Statisch vs. Funktional.....	415
6.2.2	API Überblick.....	416
6.2.3	Eingebundene Skript-Sprachen.....	418
6.2.4	Einbinden von Skript-Sprachen.....	420
6.2.5	Ausführung von Skripts aus Java.....	421
6.2.6	Aufruf von Funktionen aus Java.....	425
6.2.7	Das Invocable Interface.....	427
6.2.8	Invocable an Beispielen.....	428
6.2.9	Das Compilable Interface.....	430
6.2.10	Java aus Skript-Sicht.....	432
6.2.11	JavaScript Anbindung an Java.....	432
6.2.12	JRuby Anbindung an Java.....	437
6.2.13	JavaFX Anbindung an Java.....	442
6.3	Fazit.....	444
6.4	Referenzen.....	445
	Index.....	447