

Michael Inden

Java – die Neuerungen in Version 9 bis 12

Modularisierung, Syntax- und API-Erweiterungen

Inhaltsverzeichnis

1	Einleitung	1
I	Sprach- und API-Erweiterungen in Java 9	5
2	Syntaxerweiterungen in JDK 9	7
2.1	Anonyme innere Klassen und der Diamond Operator	7
2.2	Erweiterung der <code>@Deprecated</code> -Annotation	8
2.3	Private Methoden in Interfaces	9
2.4	Verbotener Bezeichner <code>'_'</code>	11
3	Neues und Änderungen in JDK 9	13
3.1	Neue und erweiterte APIs	13
3.1.1	Das neue Process-API	13
3.1.2	Collection-Factory-Methoden	19
3.1.3	Reactive Streams und die Klasse <code>Flow</code>	23
3.1.4	Erweiterungen in der Klasse <code>InputStream</code>	33
3.1.5	Erweiterungen rund um die Klasse <code>Optional<T></code>	35
3.1.6	Erweiterungen im Stream-API	40
3.1.7	Erweiterungen in der Klasse <code>LocalDate</code>	44
3.1.8	Erweiterungen in der Klasse <code>Arrays</code>	45
3.1.9	Erweiterungen in der Klasse <code>Objects</code>	47
3.1.10	Erweiterungen in der Klasse <code>CompletableFuture<T></code>	48
3.2	Sonstige Änderungen	52
3.2.1	Optimierung bei Strings	52
3.2.2	Deprecation diverser Typen und Methoden im JDK	53
4	Änderungen in der JVM in JDK 9	55
4.1	Änderung des Versionsschemas	55
4.2	Unterstützung von Multi-Release-JARs	57
4.3	Java + REPL => <code>jshell</code>	60
4.4	HTML5 Javadoc	65
5	Übungen zu den Neuerungen in JDK 9	67

II Sprach- und API-Erweiterungen in Java 10 bis 12	75
6 Neues und Änderungen in Java 10	77
6.1 Syntaxerweiterung <code>var</code>	77
6.2 API-Neuerungen	81
6.2.1 Unveränderliche Kopien von Collections	81
6.2.2 Immutable Collections aus Streams erzeugen	83
6.2.3 Erweiterung in der Klasse <code>Optional</code>	84
6.2.4 Modifikationen in der Versionierung	85
6.2.5 Verschiedenes	87
6.3 Fazit	88
7 Neues und Änderungen in Java 11	89
7.1 Syntaxerweiterung für <code>var</code>	90
7.2 API-Neuerungen	91
7.2.1 Neue Hilfsmethoden in der Klasse <code>String</code>	91
7.2.2 Neue Hilfsmethoden in der Utility-Klasse <code>Files</code>	93
7.2.3 Erweiterung in der Klasse <code>Optional<T></code>	95
7.2.4 Erweiterung im Interface <code>Predicate<T></code>	95
7.2.5 HTTP/2-API	96
7.3 Neuerungen in der JVM	101
7.3.1 Epsilon Garbage Collector	101
7.3.2 Launch Single-File Source-Code Programs	101
7.3.3 Das Tool Flight Recorder	102
7.4 Deprecations und Entfernungen im JDK	102
7.4.1 Aufräumarbeiten in der Klasse <code>Thread</code>	102
7.4.2 Deprecation der JavaScript-Unterstützung	102
7.4.3 Ausgliederung von JavaFX	103
7.4.4 Ausgliederung von Java EE und CORBA	103
7.5 Fazit	104
8 Neues und Änderungen in Java 12	105
8.1 Switch Expressions	105
8.1.1 Einführendes Beispiel	105
8.1.2 Zuweisungen im Lambda	109
8.1.3 <code>break</code> mit Rückgabewert	109
8.2 Microbenchmark Suite	110
8.2.1 Eigene Microbenchmarks und Varianten davon	111
8.2.2 Microbenchmarks mit JMH	113
8.2.3 Fazit	118

8.3	Java 12 – notwendige Anpassungen für Build-Tools und IDEs	119
8.3.1	Java 12 mit Gradle	119
8.3.2	Java 12 mit Maven	120
8.3.3	Java 12 mit Eclipse	121
8.3.4	Java 12 mit IntelliJ	121
8.4	Fazit	121
9	Übungen zu den Neuerungen in den JDKs 10 und 11	123

III Modularisierung 131

10	Modularisierung mit Project Jigsaw	133
10.1	Grundlagen	134
10.1.1	Bisherige Varianten der Modularisierung	135
10.1.2	Warum Modularisierung wünschenswert ist	137
10.2	Modularisierung im Überblick	138
10.2.1	Grundlagen zu Project Jigsaw	138
10.2.2	Einführendes Beispiel mit zwei Modulen	146
10.2.3	Packaging	155
10.2.4	Linking	157
10.2.5	Abhängigkeiten und Modulgraphen	161
10.2.6	Module des JDKs einbinden	163
10.2.7	Arten von Modulen	168
10.3	Sichtbarkeiten und Zugriffsschutz	170
10.3.1	Sichtbarkeiten	170
10.3.2	Zugriffsschutz an Beispielen	172
10.3.3	Transitive Abhängigkeiten (Implied Readability)	177
10.4	Zusammenfassung	182
11	Weiterführende Themen zur Modularisierung	183
11.1	Empfehlenswertes Verzeichnislayout für Module	184
11.2	Modularisierung und Services	186
11.2.1	Begrifflichkeiten: API, SPI und Service Provider	186
11.2.2	Service-Ansatz in Java seit JDK 6	187
11.2.3	Services im Bereich der Modularisierung	190
11.2.4	Definition eines Service Interface	191
11.2.5	Realisierung eines Service Provider	193
11.2.6	Realisierung eines Service Consumer	194
11.2.7	Kontrolle der Abhängigkeiten	197
11.2.8	Fazit	198

11.3	Modularisierung und Reflection	199
11.3.1	Verarbeitung von Modulen mit Reflection	199
11.3.2	Tool zur Ermittlung von Modulen zu Klassen	201
11.3.3	Besonderheiten bei Reflection	203
11.4	Kompatibilität und Migration	209
11.4.1	Kompatibilitätsmodus	209
11.4.2	Migrationsszenarien	212
11.4.3	Fallstrick bei der Bottom-up-Migration	216
11.4.4	Beispiel: Migration mit Automatic Modules	218
11.4.5	Beispiel: Automatic und Unnamed Module	219
11.4.6	Beispiel: Abwandlung mit zwei Automatic Modules	222
11.4.7	Mögliche Schwierigkeiten bei Migrationen	224
11.4.8	Fazit	224
12	Übungen zur Modularisierung	225
 IV Verschiedenes		235
13	Build-Tools und IDEs mit Java 11	237
13.1	Nicht modularisierte Applikationen	237
13.1.1	Gradle	239
13.1.2	Maven	241
13.1.3	Eclipse	243
13.1.4	IntelliJ IDEA	243
13.1.5	Externe Abhängigkeiten im Kompatibilitätsmodus	244
13.2	Modularisierte Applikationen	246
13.2.1	Gradle	247
13.2.2	Maven	251
13.2.3	Eclipse	256
13.2.4	IntelliJ IDEA	258
13.3	Fazit	260
14	Zusammenfassung	261

V	Anhang	265
A	Schnelleinstieg in Java 8	267
A.1	Einstieg in Lambdas	267
A.1.1	Lambdas am Beispiel	267
A.1.2	Functional Interfaces und SAM-Typen	268
A.1.3	Type Inference und Kurzformen der Syntax	271
A.1.4	Methodenreferenzen	272
A.2	Streams im Überblick	273
A.2.1	Streams erzeugen – Create Operations	274
A.2.2	Intermediate und Terminal Operations im Überblick	275
A.2.3	Zustandslose Intermediate Operations	277
A.2.4	Zustandsbehaftete Intermediate Operations	279
A.2.5	Terminal Operations	280
A.3	Neuerungen in der Datumsverarbeitung	283
A.3.1	Die Klasse <code>Instant</code>	284
A.3.2	Die Klassen <code>LocalDate</code> , <code>LocalTime</code> und <code>LocalDateTime</code>	284
A.3.3	Die Klasse <code>Duration</code>	286
A.3.4	Die Klasse <code>Period</code>	287
A.3.5	Datumsarithmetik mit <code>TemporalAdjusters</code>	288
A.4	Diverse Erweiterungen	290
A.4.1	Erweiterungen im Interface <code>Comparator<T></code>	290
A.4.2	Erweiterungen in der Klasse <code>Optional<T></code>	292
A.4.3	Erweiterungen in der Klasse <code>CompletableFuture<T></code>	294
B	Einführung Gradle	299
B.1	Projektstruktur für Maven und Gradle	299
B.2	Builds mit Gradle	301
C	Einführung Maven	311
C.1	Maven im Überblick	311
C.2	Maven am Beispiel	314
	Literaturverzeichnis	317
	Index	319