



**Dipl.-Inform. Michael Inden** ist Oracle-zertifizierter Java-Entwickler für JDK 6. Nach seinem Studium in Oldenburg war er lange Zeit als Softwareentwickler und -architekt bei verschiedenen internationalen Firmen tätig.

Dabei hat er fast 15 Jahre Erfahrung beim Entwurf objektorientierter Softwaresysteme gesammelt, an diversen Fortbildungen und an mehreren Java-One-Konferenzen in San Francisco teilgenommen. Sein besonderes Interesse gilt dem Design qualitativ hochwertiger Applikationen mit ergonomischen, grafischen Oberflächen sowie dem Coaching von Kollegen.

**Michael Inden**

# **Java 8 – Die Neuerungen**

**Lambdas, Streams, Date and Time API und JavaFX 8  
im Überblick**

2., aktualisierte und erweiterte Auflage



**dpunkt.verlag**

Michael Inden  
michael\_inden@hotmail.com

Lektorat: Dr. Michael Barabas  
Technischer Review: Torsten Horn, Aachen  
Copy-Editing: Ursula Zimpfer, Herrenberg  
Satz: Michael Inden  
Herstellung: Susanne Bröckelmann  
Umschlaggestaltung: Helmut Kraus, [www.exclam.de](http://www.exclam.de)  
Druck und Bindung: M.P. Media-Print Informationstechnologie GmbH, 33100 Paderborn

Bibliografische Information der Deutschen Nationalbibliothek  
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie;  
detaillierte bibliografische Daten sind im Internet über <http://dnb.d-nb.de> abrufbar.

ISBN

Buch: 978-3-86490-290-1  
PDF: 978-3-86491-749-3  
epub: 978-3-86491-750-9  
mobi: 978-3-86491-751-6

2., aktualisierte und erweiterte Auflage 2015  
Copyright © 2015 dpunkt.verlag GmbH  
Wieblinger Weg 17  
69123 Heidelberg

Die vorliegende Publikation ist urheberrechtlich geschützt. Alle Rechte vorbehalten. Die Verwendung der Texte und Abbildungen, auch auszugsweise, ist ohne die schriftliche Zustimmung des Verlags urheberrechtswidrig und daher strafbar. Dies gilt insbesondere für die Vervielfältigung, Übersetzung oder die Verwendung in elektronischen Systemen.  
Es wird darauf hingewiesen, dass die im Buch verwendeten Soft- und Hardware-Bezeichnungen sowie Markennamen und Produktbezeichnungen der jeweiligen Firmen im Allgemeinen warenzeichen-, marken- oder patentrechtlichem Schutz unterliegen.  
Alle Angaben und Programme in diesem Buch wurden mit größter Sorgfalt kontrolliert. Weder Autor noch Verlag können jedoch für Schäden haftbar gemacht werden, die in Zusammenhang mit der Verwendung dieses Buches stehen.

5 4 3 2 1 0

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b> .....	<b>1</b>
<b>2</b>	<b>Lambda-Ausdrücke</b> .....	<b>3</b>
2.1	Einstieg in Lambdas .....	4
2.1.1	Lambdas am Beispiel .....	4
2.1.2	Functional Interfaces und SAM-Typen .....	5
2.1.3	Type Inference und Kurzformen der Syntax .....	8
2.1.4	Lambdas als Parameter und als Rückgabewerte .....	9
2.1.5	Unterschiede: Lambdas vs. anonyme innere Klassen .....	9
2.2	Defaultmethoden .....	11
2.2.1	Interface-Erweiterungen .....	12
2.2.2	Vorgabe von Standardverhalten .....	14
2.2.3	Erweiterte Möglichkeiten durch Defaultmethoden .....	15
2.2.4	Spezialfall: Was passiert bei Konflikten? .....	16
2.2.5	Vorteile und Gefahren von Defaultmethoden .....	17
2.3	Statische Methoden in Interfaces .....	18
2.4	Methodenreferenzen .....	20
2.5	Exceptions in Lambdas .....	22
2.6	Fazit .....	26
2.7	Übungen zu Lambdas, Methodenreferenzen und Defaultmethoden ..	27
<b>3</b>	<b>Bulk Operations on Collections</b> .....	<b>31</b>
3.1	Externe vs. interne Iteration .....	31
3.1.1	Externe Iteration .....	32
3.1.2	Interne Iteration .....	32
3.1.3	Externe vs. interne Iteration an einem Beispiel .....	33
3.2	Collections-Erweiterungen .....	35
3.2.1	Das Interface <code>Predicate&lt;T&gt;</code> .....	35
3.2.2	Die Methode <code>Collection.removeIf()</code> .....	39
3.2.3	Das Interface <code>UnaryOperator&lt;T&gt;</code> .....	40
3.2.4	Die Methode <code>List.replaceAll()</code> .....	41
3.3	Streams .....	42
3.3.1	Streams erzeugen — Create Operations .....	43

---

3.3.2	Intermediate und Terminal Operations im Überblick .....	47
3.3.3	Zustandslose Intermediate Operations .....	49
3.3.4	Zustandsbehaftete Intermediate Operations .....	56
3.3.5	Terminal Operations .....	57
3.3.6	Wissenswertes zur Parallelverarbeitung .....	65
3.4	Filter-Map-Reduce .....	71
3.4.1	Herkömmliche Realisierung .....	71
3.4.2	Filter-Map-Reduce mit JDK 8 .....	73
3.5	Datenaufbereitung mit Kollektoren .....	75
3.5.1	Das Interface <code>Collector</code> und die Methode <code>collect()</code> ....	76
3.5.2	Vertiefung bereits vorgestellter Kollektoren .....	77
3.5.3	Aufbereitung statistischer Daten .....	79
3.5.4	Aufbereitung als Map – einfache Gruppierungen .....	80
3.5.5	Aufbereitung mit Gruppierung und Partitionierung .....	84
3.6	Fallstricke bei Lambdas und funktionaler Programmierung .....	89
3.6.1	Java-Fehlermeldungen werden zu komplex .....	89
3.6.2	Fallstrick: Imperative Lösung 1:1 funktional umsetzen .....	91
3.6.3	Fallstrick: Zustandsbehaftete Lambdas .....	92
3.7	Fazit .....	98
3.8	Übungen zu Collections und Bulk Operations .....	99
3.9	Übungen zu Streams und Filter-Map-Reduce .....	103
<b>4</b>	<b>JSR-310: Date and Time API .....</b>	<b>107</b>
4.1	Datumsverarbeitung vor JSR-310 .....	107
4.2	Überblick über die neu eingeführten Typen .....	110
4.2.1	Neue Aufzählungen, Klassen und Interfaces .....	110
4.2.2	Die Aufzählungen <code>DayOfWeek</code> und <code>Month</code> .....	112
4.2.3	Die Klassen <code>MonthDay</code> , <code>YearMonth</code> und <code>Year</code> .....	113
4.2.4	Die Klasse <code>Instant</code> .....	114
4.2.5	Die Klasse <code>Duration</code> .....	115
4.2.6	Die Aufzählung <code>ChronoUnit</code> .....	118
4.2.7	Die Klassen <code>LocalDate</code> , <code>LocalTime</code> und <code>LocalDateTime</code> .....	119
4.2.8	Die Klasse <code>Period</code> .....	120
4.2.9	Die Klasse <code>ZonedDateTime</code> .....	122
4.2.10	Zeitzone und die Klassen <code>ZoneId</code> und <code>ZoneOffset</code> .....	123
4.2.11	Die Klasse <code>Clock</code> .....	125
4.2.12	Formatierung und Parsing – die Klasse <code>DateTimeFormatter</code> .....	127
4.3	Datumsarithmetik .....	129
4.4	Das neue Date and Time API im Einsatz .....	132
4.4.1	Beispiel: Berechnung einer Zeitdifferenz .....	133
4.4.2	Simulationen und die Klasse <code>SpecialClock</code> .....	134
4.4.3	Selbst definierte <code>TemporalAdjusters</code> .....	135
4.4.4	Interoperabilität mit Legacy-Code .....	140

4.5	Fazit .....	142
4.6	Übungen zum Date and Time API .....	143
<b>5</b>	<b>Einstieg JavaFX 8 .....</b>	<b>147</b>
5.1	Einführung – JavaFX im Überblick .....	147
5.1.1	Motivation für JavaFX und Historisches .....	148
5.1.2	Grundsätzliche Konzepte .....	148
5.1.3	Layoutmanagement .....	152
5.2	Deklarativer Aufbau des GUIs .....	162
5.2.1	Deklarative Beschreibung von GUIs .....	162
5.2.2	Hello-World-Beispiel mit FXML .....	162
5.2.3	Diskussion: Design und Funktionalität strikt trennen .....	165
5.3	Rich-Client Experience .....	167
5.3.1	Gestaltung mit CSS .....	167
5.3.2	Effekte .....	173
5.3.3	Animationen .....	175
5.4	Properties, Data Binding und Observable Collections .....	177
5.4.1	Properties .....	177
5.4.2	Bindings .....	179
5.4.3	Observable Collections .....	183
5.4.4	Dynamisches Filtern von <code>ObservableList</code> .....	187
5.5	Neuerungen in JavaFX 8 .....	188
5.5.1	Unterstützung von Lambdas als <code>EventHandler</code> .....	189
5.5.2	Neue Controls .....	189
5.5.3	Filter- und sortierbare Listen .....	196
5.5.4	Texteffekte .....	199
5.5.5	JavaFX 3D .....	200
5.6	Neuerungen in JavaFX 8 Update 40 .....	203
5.6.1	Dialoge .....	203
5.6.2	Neuerungen in Bedienelementen .....	204
5.7	Fazit .....	208
5.8	Übungen zu JavaFX 8 .....	209
<b>6</b>	<b>Weitere Änderungen in JDK 8 .....</b>	<b>213</b>
6.1	Erweiterungen im Interface <code>Comparator&lt;T&gt;</code> .....	213
6.2	Die Klasse <code>Optional&lt;T&gt;</code> .....	219
6.2.1	Grundlagen zur Klasse <code>Optional&lt;T&gt;</code> .....	219
6.2.2	Weiterführendes Beispiel und Diskussion .....	222
6.3	Parallele Operationen auf Arrays .....	224
6.4	Erweiterungen im Interface <code>Map&lt;K, V&gt;</code> .....	228
6.5	Erweiterungen im Bereich Concurrency .....	233
6.6	»Nashorn« – die neue JavaScript-Engine .....	239
6.7	Keine Permanent Generation mehr .....	242

6.8	Base64-Codierungen .....	242
6.9	Erweiterungen im Bereich Reflection .....	243
6.10	Erweiterungen im NIO und der Klasse <code>Files</code> .....	245
6.11	Änderungen bei Annotations .....	247
6.12	Berechnungen mit Überlaufprüfung .....	249
6.13	Übungen zu Diverses .....	250
<b>7</b>	<b>Java 8 im Praxiseinsatz .....</b>	<b>255</b>
7.1	Erste Schritte zur Informationsaufbereitung .....	256
7.2	Grafische Darstellung .....	260
7.3	Fazit .....	266
<b>8</b>	<b>Tipps zur Migration von Java 7 auf Java 8 .....</b>	<b>267</b>
8.1	Stolpersteine in den Bibliotheken .....	267
8.1.1	Verarbeitung mit <code>HashMaps</code> .....	268
8.1.2	Unterschiede beim Einsatz von <code>SimpleDateFormat</code> .....	269
8.1.3	Rundung beim <code>NumberFormat</code> .....	269
8.1.4	Änderungen in der JavaScript-Engine .....	270
8.2	Externe in interne Iterationen überführen .....	271
8.3	Von Swing zu JavaFX .....	273
8.3.1	JavaFX in Swing einbinden .....	273
8.3.2	Swing in JavaFX einbinden .....	276
<b>9</b>	<b>Zusammenfassung und Ausblick .....</b>	<b>277</b>
9.1	Zusammenfassung und Fazit .....	277
9.2	Ausblick auf JDK 9: Mit JDK 8 nicht umgesetzte Features .....	280
9.2.1	Integration von Collection-Zugriffen .....	280
9.2.2	Vergleiche von Enums mit Operatoren .....	282
9.3	Weiterführende Literatur .....	283
<b>A</b>	<b>Java und funktionale Programmierung .....</b>	<b>287</b>
A.1	Programmierparadigmen im Überblick .....	287
A.2	Funktionale Programmierung an Beispielen .....	288
	<b>Literaturverzeichnis .....</b>	<b>291</b>