

# 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	Default-Methoden	11
2.2.1	Interface-Erweiterungen	12
2.2.2	Vorgabe von Standardverhalten	14
2.2.3	Erweiterte Möglichkeiten durch Default-Methoden	15
2.2.4	Spezialfall: Was passiert bei Konflikten?	16
2.2.5	Vorteile und Gefahren von Default-Methoden	17
2.2.6	Statische Methoden in Interfaces	18
2.3	Methodenreferenzen	20
2.4	Fazit	21
<b>3</b>	<b>Bulk Operations on Collections</b>	<b>23</b>
3.1	Externe vs. interne Iteration	23
3.1.1	Externe Iteration	24
3.1.2	Interne Iteration	24
3.1.3	Externe vs. interne Iteration an einem Beispiel	25
3.2	Collections-Erweiterungen	27
3.2.1	Das Interface <code>Predicate&lt;T&gt;</code>	27
3.2.2	Die Methode <code>Collection.removeIf()</code>	29
3.2.3	Das Interface <code>UnaryOperator&lt;T&gt;</code>	30
3.2.4	Die Methode <code>List.replaceAll()</code>	32
3.3	Streams	32
3.3.1	Streams erzeugen — Create Operations	33
3.3.2	Intermediate und Terminal Operations im Überblick	37
3.3.3	Zustandslose Intermediate Operations	39

3.3.4	Zustandsbehaftete Intermediate Operations .....	47
3.3.5	Terminal Operations .....	48
3.3.6	Wissenswertes zur Parallelverarbeitung .....	56
3.4	Filter-Map-Reduce .....	57
3.4.1	Herkömmliche Realisierung .....	57
3.4.2	Filter-Map-Reduce mit JDK 8 .....	59
3.5	Fallstricke bei Lambdas und funktionaler Programmierung .....	62
3.5.1	Java-Fehlermeldungen werden zu komplex .....	62
3.5.2	Fallstrick: Imperative Lösung 1:1 funktional umsetzen .....	64
3.6	Fazit .....	65
<b>4</b>	<b>JSR-310: Date And Time API .....</b>	<b>67</b>
4.1	Datumsverarbeitung vor JSR-310 .....	67
4.2	Überblick über die neu eingeführten Klassen .....	70
4.2.1	Die Klasse Instant .....	70
4.2.2	Die Aufzählung ChronoUnit .....	71
4.2.3	Die Klasse Duration .....	72
4.2.4	Die Klassen LocalDate, LocalTime und LocalDateTime .....	74
4.2.5	Die Aufzählungen DayOfWeek und Month .....	75
4.2.6	Die Klassen YearMonth, MonthDay und Year .....	76
4.2.7	Die Klasse Period .....	77
4.2.8	Die Klasse Clock .....	79
4.2.9	Die Klasse ZonedDateTime .....	79
4.2.10	Beispiel: Berechnung einer Zeitdifferenz .....	80
4.2.11	Interoperabilität mit Legacy-Code .....	81
4.3	Fazit .....	82
<b>5</b>	<b>Einstieg JavaFX 8 .....</b>	<b>83</b>
5.1	Einführung – JavaFX im Überblick .....	83
5.1.1	Motivation für JavaFX und Historisches .....	83
5.1.2	Grundsätzliche Konzepte .....	84
5.1.3	Layoutmanagement .....	88
5.2	Deklarativer Aufbau des GUIs .....	98
5.2.1	Deklarative Beschreibung von GUIs .....	98
5.2.2	Hello-World-Beispiel mit FXML .....	98
5.2.3	Diskussion: Design und Funktionalität strikt trennen .....	101
5.3	Rich-Client Experience .....	103
5.3.1	Gestaltung mit CSS .....	103
5.3.2	Effekte .....	109
5.3.3	Animationen .....	111
5.4	Neuerungen in JavaFX 8 .....	113
5.4.1	Unterstützung von Lambdas als EventHandler .....	113
5.4.2	Texteffekte .....	114

---

5.4.3	Neue Controls .....	115
5.4.4	JavaFX 3D .....	122
5.5	Fazit .....	124
<b>6</b>	<b>Weitere Änderungen in JDK 8 .....</b>	<b>127</b>
6.1	Erweiterungen im Interface Comparator<T> .....	127
6.2	Die Klasse Optional<T> .....	133
6.2.1	Grundlagen zur Klasse Optional<T> .....	133
6.2.2	Weiterführendes Beispiel und Diskussion .....	136
6.3	Parallele Operationen auf Arrays .....	138
6.4	Erweiterungen im Interface Map<K, V> .....	142
6.5	Erweiterungen im NIO und der Klasse Files .....	146
6.6	Erweiterungen im Bereich Concurrency .....	148
6.7	»Nashorn« – die neue JavaScript-Engine .....	152
6.8	Keine Permanent Generation mehr .....	155
6.9	Erweiterungen im Bereich Reflection .....	156
6.10	Base64-Codierungen .....	158
6.11	Änderungen bei Annotations .....	159
<b>7</b>	<b>Zusammenfassung und Ausblick .....</b>	<b>161</b>
7.1	Zusammenfassung und Fazit .....	161
7.2	Ausblick auf JDK 9: Mit JDK 8 nicht umgesetzte Features .....	164
7.2.1	Integration von Collection-Zugriffen .....	164
7.2.2	Vergleiche von Enums mit Operatoren .....	166
7.3	Weiterführende Literatur .....	167
<b>A</b>	<b>Java und funktionale Programmierung .....</b>	<b>173</b>
A.1	Programmierparadigmen im Überblick .....	173
A.2	Funktionale Programmierung an Beispielen .....	174
<b>Literaturverzeichnis .....</b>		<b>177</b>
<b>Index .....</b>		<b>179</b>