

Inhaltsverzeichnis

	Einleitung	13
1	Eigenschaften von Kotlin	21
1.1	Eine neue Sprache	21
1.2	Kotlin und Android	23
1.3	Vorteile von Kotlin	24
1.4	Ziele der Entwicklung von Kotlin	25
1.4.1	Übersichtlichkeit, präziser Ausdruck	25
1.4.2	Sicherheit	28
1.4.3	Interoperabilität	29
2	Arbeiten mit Kotlin	33
2.1	Entwicklungsumgebung einrichten	33
2.1.1	IntelliJ	33
2.1.2	Android Studio	37
2.2	Tools	39
2.2.1	Kotlin-Compiler	40
2.2.2	Kotlin-Compiler mit Gradle	40
2.2.3	Integrierte Entwicklungsumgebungen/IDE	41
2.2.4	Konvertierung von existierendem Code	41
3	Kotlin by example	43
3.1	Hello world!	44
3.2	Hello null!	46
3.2.1	Ein erster Test	47
3.2.2	Eine erste Exception	48
3.2.3	Eine erste Variable – Null-Sicherheit	49
3.2.4	Let und Lambdas	50
3.2.5	Eine erste Schleife	53
3.3	Hello Android!	55
3.4	Hello Kotlin!	59
4	Grundlagen und Daten	63
4.1	Expressions vs. Statements	63

4.2	Strukturierung von Code durch Dateien, Klassen, Funktionen und Blöcke	65
4.3	Funktionen	69
	4.3.1 Direkte Funktionsaufrufe	70
	4.3.2 Benutzung von Lambdas	71
4.4	Variablen	72
	4.4.1 Deklaration: Immutable vs. Mutable, Zuweisung	74
	4.4.2 Typen und Typsicherheit	77
	4.4.3 Null-Sicherheit	77
	4.4.4 Typinferenz	84
	4.4.5 Generics benutzen	85
	4.4.6 Boxing	85
4.5	Besonderheiten der Datentypen	88
	4.5.1 Zahlen	88
	4.5.2 Boolean	90
	4.5.3 Chars und Strings – Zeichen und Zeichenketten	90
	4.5.4 Objekte und Any	93
	4.5.5 Arrays und Listen	94
	4.5.6 Maps	98
	4.5.7 Ranges	99
5	Grundstrukturen	101
5.1	Kontrollstrukturen	101
5.2	Verzweigungen mit if	101
5.3	Verzweigung mit when	103
5.4	Schleifen mit for und while	105
	5.4.1 for-Schleifen	106
	5.4.2 while-Schleifen	109
	5.4.3 repeat	110
	5.4.4 break/named break und continue	110
	5.4.5 Funktionsbasierte Alternativen	112
5.5	Exception Handling mit try/catch	114
5.6	Annotations	116
5.7	Datentypen	118
	5.7.1 Type-Casting	119
	5.7.2 Type-Casting und null	120
	5.7.3 Smart Casts	121
	5.7.4 Type Aliases	122

6	Funktionen und Lambdas	125
6.1	Funktionale Entwicklung	125
6.2	Syntax von Funktionen	127
6.2.1	Kurzformen von Funktionsdefinitionen	129
6.2.2	Parameter	129
6.2.3	Variable Anzahl von Parametern	131
6.2.4	Rückgabe	132
6.2.5	Destrukturierende Deklarationen	133
6.2.6	Kombination mehrerer Funktionsaufrufe	135
6.3	Höhere Funktion und Lambdas	136
6.3.1	Funktionen als Parameter	136
6.3.2	Funktionstyp für Variablen	137
6.4	Chaining von Lambdas	138
6.5	Rekursive Funktionen	138
6.6	Spezielle Funktionen	139
6.6.1	Lokale Funktionen	139
6.6.2	Inline-Funktionen	141
6.6.3	Infix-Funktionen	143
6.6.4	Actual-Funktionen	143
6.6.5	Überladene Operatoren	144
7	Packages, Klassen und Objekte	145
7.1	Architektur und Entwurf	145
7.2	Packages, Klassen und Interfaces	145
7.2.1	Klassen und Interfaces organisieren das Modell	146
7.2.2	Klassen und Interfaces, Vererbung	146
7.2.3	Packages organisieren das Projekt	149
7.2.4	Sichtbarkeitsmodifier	151
7.2.5	Import	152
7.3	Von der Klasse zum Objekt – Meet the members	154
7.3.1	Konstruktoren und Initializer	154
7.3.2	Properties	158
7.3.3	Custom Getter/Setter	159
7.3.4	Methoden	161
7.3.5	Innere Klassen	163
7.3.6	Companion Objects	165
7.4	Objekt-Anweisungen	166
7.4.1	Singletons via Objekt-Deklaration	166

7.4.2	Anonyme Klassen via Objekt-Ausdruck	168
7.4.3	Object-Statements bei Rückgaben	169
7.5	Datenklassen	170
7.6	Generics	171
7.6.1	Generische Klassen	171
7.6.2	Beschränkung der Typen: Invariant, Kovariant, Kontravariant.	173
7.6.3	Generische Methoden	177
7.6.4	Reified parameters	178
7.7	Enum classes	179
7.8	Sealed classes	181
7.9	Extensions	182
7.10	Destrukturierende Deklarationen	184
8	Testing	187
8.1	Einführung	187
8.1.1	Unit-Tests	188
8.1.2	Integrationstests	188
8.1.3	Testsuite	189
8.1.4	Mocking und Dependency Injection	189
8.1.5	Konkret: Wie werden Tests geschrieben?	190
8.2	Erster JUnit-Test	191
8.3	Einrichten eines Testprojekts	193
8.4	Praxisbeispiel: Verkleinern von Funktionen	194
8.5	Das Testresultat lesen	200
8.6	Der Test von Hello world!	202
8.7	Test einer Klasse	204
8.8	Parametrisierte Tests	207
8.9	Testing Kotlin-Style	208
8.9.1	Testkomponenten	208
8.9.2	Spek	210
8.9.3	Kotlintest.io	210
8.9.4	kotlin.test: Unit-Tests in Multiplattform-Projekten	211
9	Kotlin für Fortgeschrittene	215
9.1	Alles zusammenbringen	215
9.2	Zielplattformen	216
9.3	Kotlin-Standard	217
9.3.1	run, let, apply, with, also	218

9.3.2	Takeif/Takeunless	223
9.3.3	Collections, Arrays und Listen.....	224
9.3.4	Operationen auf allen iterierbaren Elementen.....	226
9.3.5	Texttools	233
9.3.6	Reflection	234
9.4	Delegation	236
9.4.1	Property Delegation	237
9.4.2	Class Delegation.....	241
9.5	Service Location/Dependency Injection	246
9.5.1	Warum eigentlich?	247
9.5.2	Service Locator	248
9.5.3	Dependency Injection	251
9.6	Coroutines und Nebenläufigkeit	253
9.6.1	Launch und Scopes	253
9.6.2	Async, Await und Deferred	257
9.6.3	Producer, Consumer und Actor	258
9.7	Contracts	258
9.8	Idiomatischer Code	261
9.8.1	Unveränderliche/immutable Variablen	261
9.8.2	Kurzformen für null.....	262
9.8.3	Kurzschreibweisen für Funktionen	262
9.8.4	Nutzung von Class Properties.....	262
9.8.5	Chaining von nicht chainbaren Aufrufen.....	262
9.8.6	Extensions nutzen	263
9.8.7	DSL mit Lambdas.....	263
9.8.8	Infix-Notation für klare Sätze	263
10	Kotlin für Android	265
10.1	Android <3 Kotlin	265
10.2	Android-Komponenten	266
10.2.1	Package-Level und Überblick.....	266
10.2.2	Klassenlevel.....	267
10.3	Bestandteile des Android-Frameworks	270
10.3.1	Activities und Fragmente.....	271
10.3.2	Eine minimale App	272
10.3.3	Views, Rendering und Layouting	275
10.3.4	Der Lifecycle	282
10.3.5	Android Architecture Components.....	283

10.4	Architektur für größere App-Projekte	285
10.5	Das Architekturmodell MVP(I)	286
10.5.1	Der Vertrag	287
10.5.2	Initialisierung	288
10.5.3	Daten laden und initiale UI erstellen	289
10.5.4	Mehrere Presenter/Komponenten	292
10.5.5	Updates der UI und Reagieren auf Dateneingaben	293
10.6	Das Architekturmodell MVVM	296
10.6.1	MVVM-Komponenten anlegen	297
10.6.2	Initialisierung	297
10.6.3	Daten laden und initiale UI erstellen	298
10.6.4	Updates der UI und Reagieren auf Dateneingaben	301
10.7	MVP oder MVVM? It depends	302
10.8	Erweiterung von Android-Klassen	302
10.9	Android und Coroutines	303
10.10	Delegation für Android	304
10.10.1	Property Delegation	304
10.10.2	Klassendelegation	306
11	Zielpattformen für Kotlin	311
11.1	Multiplattform-Logik	311
11.2	Java	312
11.2.1	Java-Libraries	312
11.2.2	Reflection	312
11.2.3	Streams (Java 8)	314
11.2.4	Java Collections	314
11.2.5	Java Annotations	314
11.2.6	Annotation Use-Site Target	317
11.3	JavaScript in Kotlin	318
11.3.1	Einrichtung oder: Wie geht denn das?	318
11.3.2	Grundsätzliches zu Kotlin und JS	318
11.3.3	Standalone JS-App	320
11.4	JS-Standard-Bibliothek	320
11.4.1	Dynamic	321
11.4.2	Promises	322
11.4.3	Browser	322
11.4.4	JSON-Parsing	324
11.4.5	Nutzung von JS-Bibliotheken/-Aufrufen	326

11.5	Konvertierung von existierendem Java-Code zu JS.....	327
11.6	Konvertierung und Bearbeitung von existierendem JS-Code.....	328
11.7	Multiplattform-Projekte	328
11.7.1	Ein Multiplattform-Projekt beginnen	330
11.7.2	Expected/Actual	333
11.8	Native	338
12	Rück- und Ausblick	339
A	Referenztabellen	341
A.1	Factories	341
A.1.1	Factories für Arrays	341
A.1.2	Factories für Listen und Maps	342
A.2	Konvertierung von Listen/Arrays	342
A.3	Operationen auf allen iterierbaren Elementen	343
A.3.1	Zugriff auf einzelne Elemente	343
A.3.2	Zugriff auf Maps	344
A.3.3	Iteration über alle oder viele Elemente	344
A.3.4	Manipulation von Listen	345
A.3.5	Manipulation von Maps	346
	Stichwortverzeichnis	347