O'REILLY®

Kotlin

von Kopf bis Fuß

Haben Sie Spaß mit der Standardbibliothek von Kotlin

Entdecken Sie die Feinheiten der generischen Programmierung

Finden Sie heraus, wie Elvis Ihr Leben verändern kann

Eine Einführung in die Kotlin-Programmierung

Vermeiden Sie peinliche Lambda-Fehler

Schreiben Sie außerirdisch gute Funktionen höherer Ordnung

Sehen Sie sich Collections unter dem Mikroskop an

Dawn Griffiths & David Griffiths

Deutsche Übersetzung von Jørgen W. Lang

Kotlin
von Kopf bis Fuß

Dawn Griffiths
David Griffiths

Deutsche Übersetzung
von Jørgen W. Lang

O'REILLY®

Dawn Griffiths und David Griffiths

Lektorat: Alexandra Follenius
Übersetzung: Jørgen W. Lang
Korrektorat: Sibylle Feldmann, *www.richtiger-text.de*
Satz: Ulrich Borstelmann, *www.borstelmann.de*
Herstellung: Stefanie Weidner
Umschlaggestaltung: Randy Comer, Michael Oréal, *www.oreal.de*
Druck und Bindung: mediaprint solutions 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:
Print 978-3-96009-112-7
PDF 978-3-96010-309-7
ePub 978-3-96010-310-3
mobi 978-3-96010-311-0

Dieses Buch erscheint in Kooperation mit O'Reilly Media, Inc. unter dem Imprint »O'REILLY«.
O'REILLY ist ein Markenzeichen und eine eingetragene Marke von O'Reilly Media, Inc. und wird mit Einwilligung des Eigentümers verwendet.

1. Auflage 2019

Wieblinger Weg 17
69123 Heidelberg

Hinweis:
Dieses Buch wurde auf PEFC-zertifiziertem Papier aus nachhaltiger Waldwirtschaft gedruckt. Der Umwelt zuliebe verzichten wir zusätzlich auf die Einschweißfolie.

Schreiben Sie uns:
Falls Sie Anregungen, Wünsche und Kommentare haben, lassen Sie es uns wissen: kommentar@oreilly.de.

5 4 3 2 1 0

Gewidmet den klugen Köpfen hinter
Kotlin für das Erschaffen einer großartigen
Programmiersprache.

Die Autoren von Kotlin von Kopf bis Fuß

Dawn Griffiths besitzt über 20 Jahre Erfahrung in der IT-Industrie und arbeitet als Senior-Entwicklerin und Senior-Softwarearchitektin. Sie hat bereits eine Reihe von Büchern aus der *Head-First*-Reihe (deutsch *Von Kopf bis Fuß)* geschrieben, darunter *Head First Android Development*. Zusammen mit ihrem Mann David hat sie außerdem die Lehrvideoreihe *The Agile Sketchpad* entwickelt, um Schlüsselkonzepte und -techniken auf eine Art zu vermitteln, die das Hirn aktiv und auf Trab hält.

Wenn Dawn keine Bücher schreibt oder Videos macht, arbeitet sie an ihren Tai-Chi-Fähigkeiten, liest, läuft, klöppelt oder kocht. Besonders mag sie es, Zeit mit ihrem Mann David zu verbringen.

David Griffiths hat als Agile Coach, Entwickler und als Werkstattmeister gearbeitet, aber nicht in dieser Reihenfolge. Er begann im Alter von 12 Jahren mit dem Programmieren, nachdem er eine Dokumentation über die Arbeit von Seymour Papert gesehen hatte. Als er 15 war, schrieb er eine Implementierung von Paperts Computersprache LOGO. Vor *Kotlin von Kopf bis Fuß* hat David bereits verschiedene andere Bücher aus der *Von Kopf bis Fuß*-Reihe verfasst, darunter *Head First Android Development*, und zusammen mit Dawn die Lehrvideoreihe *The Agile Sketchpad* entwickelt.

Wenn David nicht schreibt, programmiert oder als Coach arbeitet, verbringt er seine Freizeit damit, mit seiner wunderbaren Frau und Mitautorin Dawn zu verreisen.

Sie finden Dawn und David auf Twitter unter: *https://twitter.com/HeadFirstKotlin*.

Der Inhalt (im Überblick)

Inhalt (jetzt ausführlich)

Einführung

Ihr Gehirn und Kotlin. *Sie* versuchen, etwas zu *lernen*, und Ihr *Hirn* tut sein Bestes, damit das Gelernte nicht *hängen bleibt*. Es denkt nämlich: »Wir sollten lieber ordentlich Platz für wichtigere Dinge lassen, z. B. für das Wissen darüber, welche Tiere einem gefährlich werden könnten, oder dass es eine ganz schlechte Idee ist, nackt Snowboard zu fahren.« Tja, wie schaffen wir es nun, Ihr Gehirn davon zu überzeugen, dass Ihr Leben davon abhängt, wie man in Kotlin programmiert?

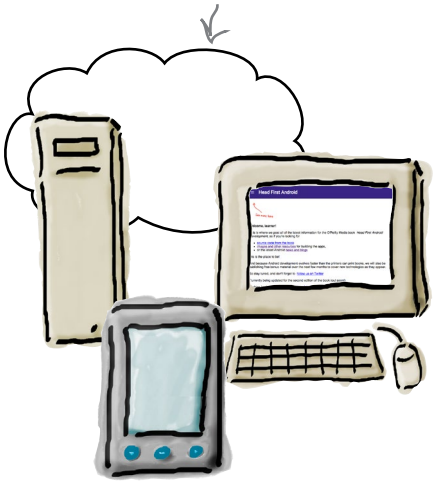
Erste Schritte

1 Ein Sprung ins kalte Wasser

Kotlin schlägt Wellen.

Seit seiner ersten Veröffentlichung hat Kotlin Programmierer mit seiner ***freundlichen Syntax, Knappheit, Flexibilität und Leistungsfähigkeit*** beeindruckt. In diesem Buch zeigen wir Ihnen, wie Sie **Ihre eigenen Kotlin-Applikationen erstellen** können. Wir beginnen, indem wir ein einfaches Programm schreiben und laufen lassen. Unterwegs stellen wir Ihnen wesentliche Teile der Kotlin-Syntax vor, z. B. *Anweisungen*, *Schleifen* und *bedingungsbasierte Verzweigungen*. Ihre Reise hat gerade erst begonnen …

Die Möglichkeit, auszuwählen, gegen welche Plattform Ihr Code kompiliert wird, bedeutet, dass Kotlin auf Servern, in der Cloud, in Browsern, auf Mobilgeräten und mehr funktioniert.

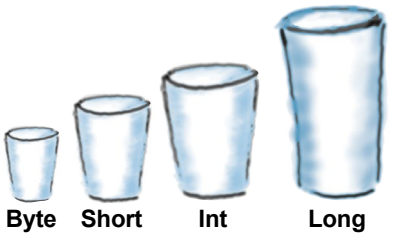


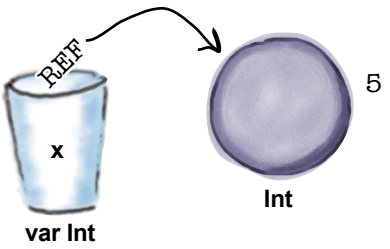
Basistypen und Variablen

2 Eine Variable sein

Es gibt eine Sache, von der jeder Code abhängt: Variablen.

In diesem Kapitel werfen wir einen Blick unter die Motorhaube und zeigen Ihnen, wie ***Kotlin-Variablen tatsächlich funktionieren***. Sie werden Kotlins **Basisdatentypen** wie *Integer*, *Floats* und *boolesche Werte* kennenlernen. Sie werden sehen, wie Kotlins Compiler den **Typ einer Variablen anhand des übergebenen Werts feststellen** kann. Außerdem lernen Sie den Einsatz von **String-Templates** für die Erstellung komplexer Strings mit wenig Code sowie das Anlegen von **Arrays**, um mehrere Werte zu speichern. Abschließend kümmern wir uns noch um die Frage: »*Warum sind Objekte für das Leben in Kotlinville so wichtig?*«



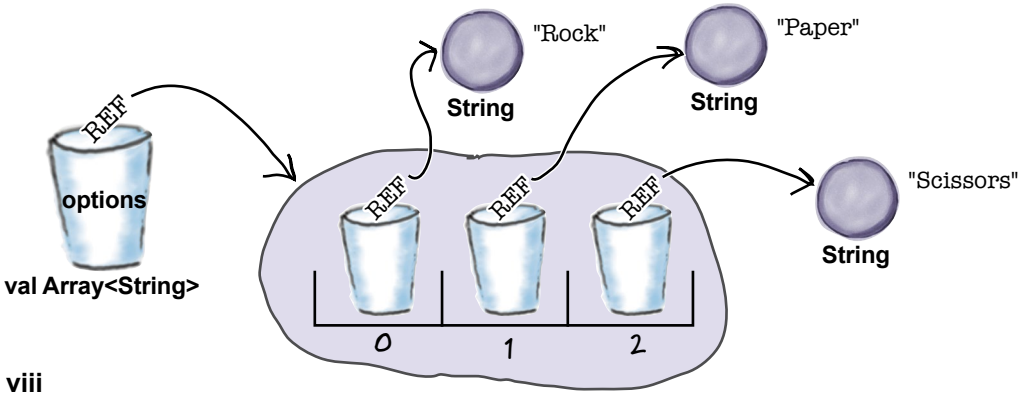


Funktionen

3 Raus aus main

Es ist Zeit für den nächsten Schritt: Funktionen.

Bisher befand sich der Code ausschließlich in der *main*-Funktion Ihrer Applikation. Wenn Sie Ihren Code **besser organisieren** und **leichter pflegen** wollen, müssen Sie wissen, ***wie Sie den Code in separate Funktionen aufteilen können***. In diesem Kapitel lernen Sie, wie man ***Funktionen schreibt*** und damit ***interagiert***, indem Sie ein Spiel programmieren. Wir zeigen Ihnen, wie man kompakte ***Einzelausdrucksfunktionen*** schreibt. Und unterwegs finden Sie auch noch heraus, wie man ***über Bereiche (Ranges) und Sammlungen (Collections) iteriert*** und wie die mächtige *for*-Schleife funktioniert.



Klassen und Objekte

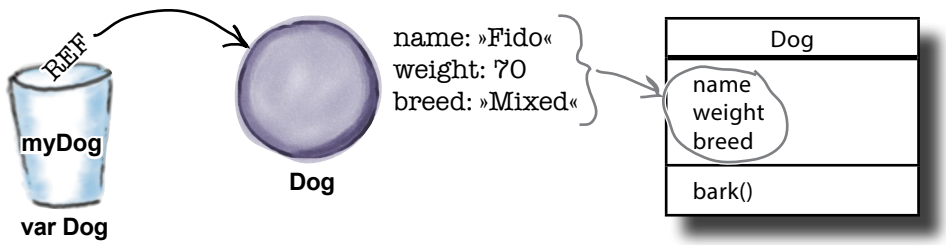
4 Etwas mehr Klasse

Jetzt ist es Zeit, über Kotlins Basistypen hinaus weiterzublicken.

Früher oder später sind Kotlins Basisdatentypen nicht mehr genug. Sie wollen *mehr*. Und da kommen *Klassen* ins Spiel. Klassen sind *Vorlagen*, mit denen Sie **Ihre eigenen Objekttypen erstellen** und deren Eigenschaften und Funktionen Sie selbst definieren können. In diesem Kapitel lernen Sie, wie Sie **eigene Klassen entwickeln und definieren** und wie diese verwendet werden, um **neue Arten von Objekten zu erstellen**. Sie werden ***Konstruktoren*** und ***Initialisierungsblocks***, ***Getter und Setter*** kennenlernen und herausfinden, wie sie benutzt werden können, um Ihre Eigenschaften zu schützen. Schließlich werden Sie lernen, wie Verkapselung (»Data Hiding«) **in *sämtlichem* Kotlin-Code** bereits eingebaut ist, wodurch Sie Zeit, Aufwand und eine Menge Tipparbeit sparen können.

Eine Klasse

Viele Objekte

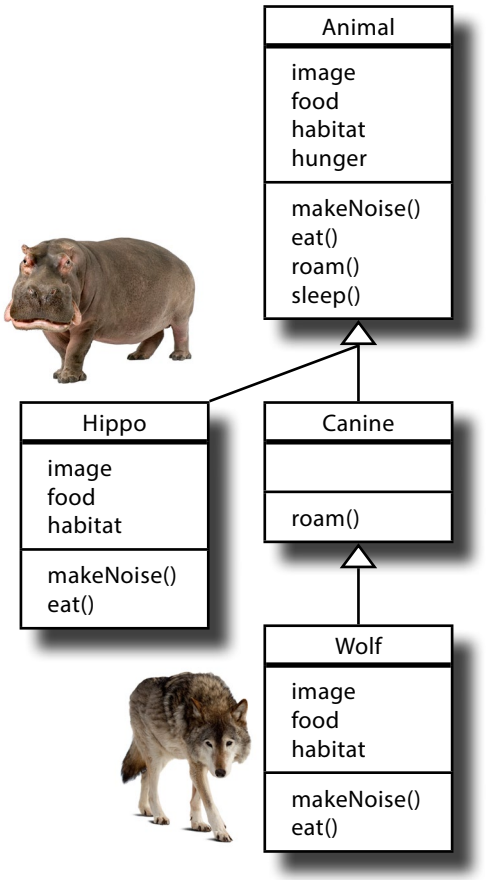


Subklassen und Superklassen

5 Vererbung

Manchmal begegnet man Objekttypen, die ideal wären, wenn man nur ein paar Kleinigkeiten ändern könnte!

Genau das ist einer der Vorteile von **Vererbung**. In diesem Kapitel lernen Sie das Erstellen von **Subklassen** und erfahren, wie Sie Eigenschaften und Funktionen einer **Superklasse** erben können. Sie lernen, wie man ***Funktionen und Eigenschaften überschreibt***, damit sich Klassen so verhalten, wie ***Sie*** es wollen. Außerdem erfahren Sie, wann Vererbung sinnvoll ist (und wann nicht). Schließlich zeigen wir Ihnen, wie Vererbung dabei hilft, **doppelten Code** zu vermeiden, und wie Sie Ihre Flexibilität mithilfe von **Polymorphismus** steigern können.

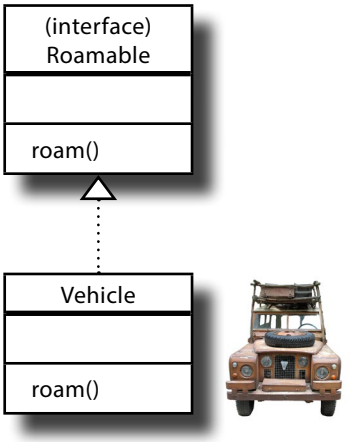


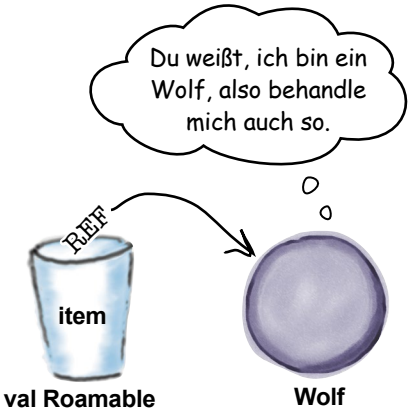
Abstrakte Klassen und Interfaces

6 Ernsthafter Polymorphismus

Eine Superklassen-Vererbungshierarchie ist nur der Anfang.

Wenn Sie alle Möglichkeiten des ***Polymorphismus nutzen wollen***, müssen Sie beim Design **abstrakte Klassen und Interfaces (Schnittstellen) einsetzen**. In diesem Kapitel lernen Sie, wie man abstrakte Klassen verwendet, um zu kontrollieren, welche Klassen Ihrer Hierarchie ***instanziiert werden können und welche nicht***. Sie werden sehen, wie man konkrete Subklassen dazu zwingt, ***ihre eigenen Implementierungen zu verwenden***. Und Sie werden erfahren, wie man Interfaces benutzt, um ***Verhalten zwischen unabhängigen Klassen*** zu teilen. Zwischendurch zeigen wir Ihnen noch, was es mit ***is***, ***as*** und ***when*** auf sich hat.



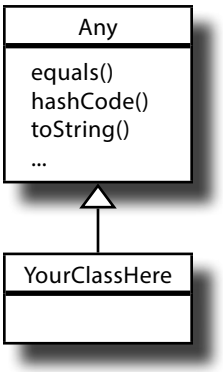


Datenklassen

7 Mit Daten umgehen

Niemand will sein Leben lang das Rad neu erfinden.

Die meisten Applikationen enthalten Klassen, die ausschließlich für die *Datenspeicherung* zuständig sind. Um Ihnen das Programmieren zu erleichtern, haben die Kotlin-Entwickler sich das Konzept der **Datenklassen** ausgedacht. In diesem Kapitel zeigen wir Ihnen, wie Sie Datenklassen verwenden, um Code zu schreiben, der ***sauberer und knapper*** ist, als Sie es je für möglich gehalten haben. Sie werden die Datenklassen-***Hilfsfunktionen*** kennenlernen und entdecken, wie man ein ***Datenobjekt in seine Bestandteile destrukturieren*** kann. Unterwegs zeigen wir Ihnen außerdem, wie ***Standardparameter*** Ihren Code flexibler machen können. Außerdem stellen wir Ihnen **Any** vor, die *Mutter aller Superklassen*.



Datenobjekte gelten als gleich, wenn sie die gleichen Eigenschaftswerte enthalten.

Nullwerte und Ausnahmen

8 Gesund und munter

Alle wollen sicheren Code schreiben.

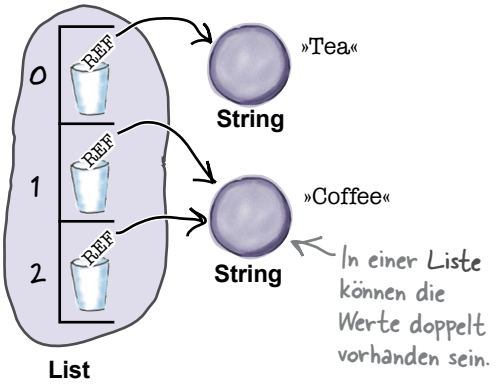
Die gute Nachricht ist: Kotlin wurde *mit dem Ziel der Codesicherheit* entwickelt. Wir beginnen, indem wir Ihnen Kotlins **nullwertfähige Datentypen** zeigen und warum dadurch *in Kotlinville so gut wie keine Fehler vom Typ NullPointerException auftreten*. Sie werden erfahren, wie man *sichere Aufrufe* durchführt und wie Kotlins **Elvis**-Operator verhindert, dass Sie vollkommen durcheinanderkommen (»*All shook up«)*. Wenn wir damit fertig sind, zeigen wir Ihnen noch, wie Sie **Ausnahmen auslösen und abfangen können wie ein Profi**.

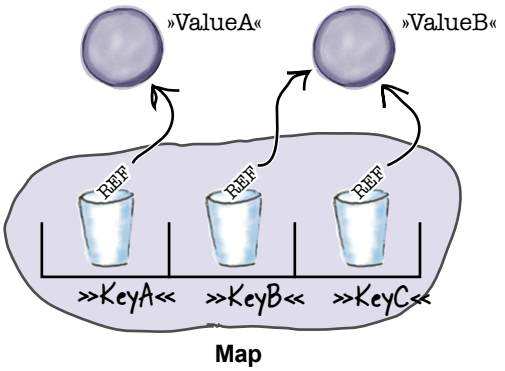
Collections

9 Dinge organisieren

Haben Sie jemals etwas Flexibleres als ein Array gebraucht?

Kotlin besitzt eine Reihe nützlicher **Collections** (Sammlungen), die mehr Flexibilität und eine größere Kontrolle über ***die Speicherung und Verwaltung von Objektgruppen*** bieten. Brauchen Sie eine ***erweiterbare Liste***? Wollen Sie den Inhalt ***mischen oder umkehren***? Wollen Sie ***etwas anhand seines Namens finden***? Oder ***wollen Sie Duplikate entfernen***, ohne auch nur einen Finger rühren zu müssen? Wenn Sie auch nur eines dieser Merkmale brauchen, lesen Sie weiter. In diesem Kapitel finden Sie, was Sie suchen …





In einer Map dürfen die Werte doppelt vorhanden sein. Die Schlüssel müssen dagegen einmalig sein.

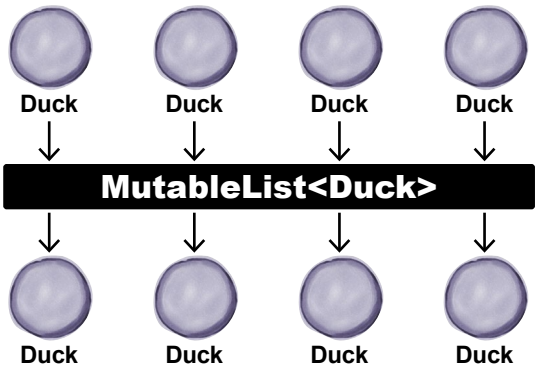
Generische Programmierung

10 Innen und außen unterscheiden

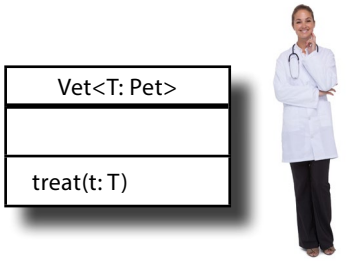
Alle mögen konsistenten Code.

Generics helfen dabei, konsistenteren Code zu schreiben, der weniger Probleme verursacht. In diesem Kapitel zeigen wir Ihnen, wie **Kotlins Collection-Klassen Generics** einsetzen, um zu verhindern, dass Sie versehentlich einen Salat in einer List<Seemöwe> speichern. Sie werden lernen, wann und wie Sie ***Ihre eigenen generischen Klassen, Interfaces und Funktionen*** erstellen und wie Sie einen **generischen Typ** auf einen bestimmten Supertyp beschränken können. Schließlich lernen Sie, wie Sie mit ***Kovarianz und Kontravarianz*** das Verhalten generischer Typen SELBST bestimmen können.

Mit Generics werden ausschließlich Referenzen auf Duck-Objekte IN der MutableList gespeichert ...



... und kommen als Referenzen auf den Objekttyp Duck auch wieder HERAUS.

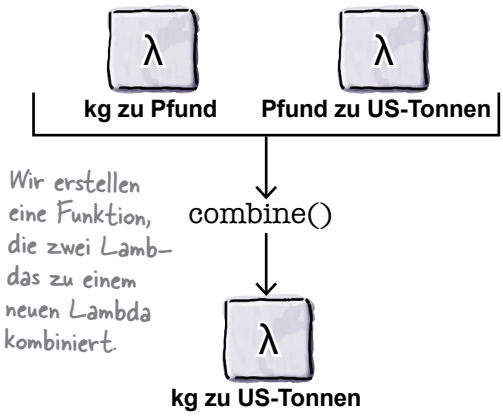


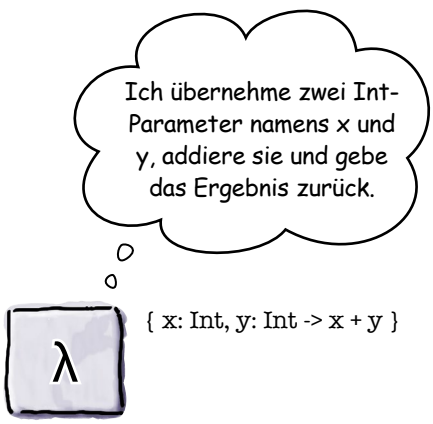
Lambdas und Funktionen höherer Ordnung

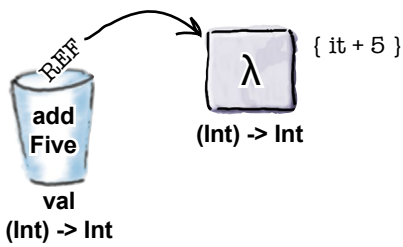
11 Code wie Daten behandeln

Wollen Sie Code schreiben, der noch flexibler und mächtiger ist?

Falls ja, brauchen Sie **Lambdas**. Ein *Lambda* – oder *Lambda-Ausdruck* (offiziell auch Lambda-Funktion oder anonyme Funktion) – ist ein Codeblock, den Sie wie ein Objekt herumreichen können. In diesem Kapitel erfahren Sie, wie man ein ***Lambda definiert, es einer Variablen zuweist*** und ***seinen Code ausführt***. Sie lernen verschiedene **Funktionstypen** kennen und wie diese Ihnen beim Schreiben von **Funktionen höherer Ordnung** helfen können, die Lambdas für ihre Parameter- und Rückgabewerte zu benutzen. Nebenbei zeigen wir Ihnen noch, wie ein wenig ***syntaktischer Zucker das Programmiererleben etwas versüßen kann***.





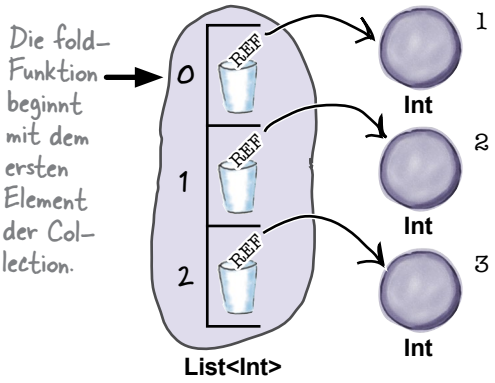


Eingebaute Funktionen höherer Ordnung

12 Dem Code Beine machen

Kotlin besitzt eine Vielzahl eigener Funktionen höherer Ordnung.

In diesem Kapitel werden wir Ihnen einige davon vorstellen. Sie werden die flexible ***filter-Familie*** kennenlernen und erfahren, wie Sie Ihre Collections damit auf die richtige Größe zurechtstutzen können. Außerdem zeigen wir Ihnen, wie Sie eine ***Collection mit map umwandeln, per forEach über ihre Elemente iterieren*** und die enthaltenen ***Elemente per groupBy ordnen können***. Darüber hinaus erläutern wir Ihnen, wie Sie ***fold*** benutzen können, um komplexe Berechnungen *mit nur einer Codezeile* durchzuführen. Am Ende dieses Kapitels werden Sie **mächtigeren Code** schreiben können, **als Sie je gedacht haben**.



Koroutinen

Code parallel ausführen

Manche Aufgaben laufen am besten im Hintergrund.

Sollen Daten von einem langsamen externen Server gelesen werden, wollen Sie vermutlich nicht bis zum Ende danebensitzen und Däumchen drehen. In solchen Fällen sind **Koroutinen Ihre neuen besten Freunde**. Mit Koroutinen können Sie Code ***asynchron ausführen***, das heißt *weniger Däumchen drehen* und eine *bessere Benutzbarkeit*. Außerdem können Ihre Applikationen durch Koroutinen *skalierbarer* werden. Wenn Sie weiterlesen, werden Sie das Geheimnis lüften, wie Sie gleichzeitig mit Bob reden und Suzy zuhören können.

Bam! Bam! Bam! Bam! Bam! Bam!
Tish! Tish!

Jetzt werden Toms und Becken parallel gespielt.

Testen

Ziehen Sie Ihren Code zur Rechenschaft

Jeder weiß, dass guter Code funktionieren muss.

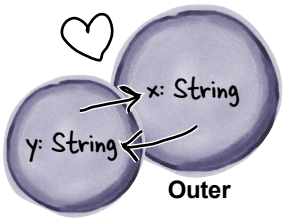
Aber jede Codeänderung birgt das Risiko neuer Bugs, die verhindern, dass Ihr Code wie gewünscht funktioniert. Darum ist *sorgfältiges Testen* so wichtig: Sie erfahren von möglichen Problemen im Code, *bevor er in einer Produktionsumgebung eingesetzt wird*. In diesem Anhang besprechen wir ***JUnit*** und ***KotlinTest***, zwei Bibliotheken für die Durchführung von **Unit-Tests**. Dadurch haben Sie *grundsätzlich ein Sicherheitsnetz zur Verfügung*.

Was übrig bleibt

Die Top Ten der Themen, die wir nicht behandelt haben

Nach allem, was wir behandelt haben, gibt es immer noch ein paar weitere wichtige Dinge.

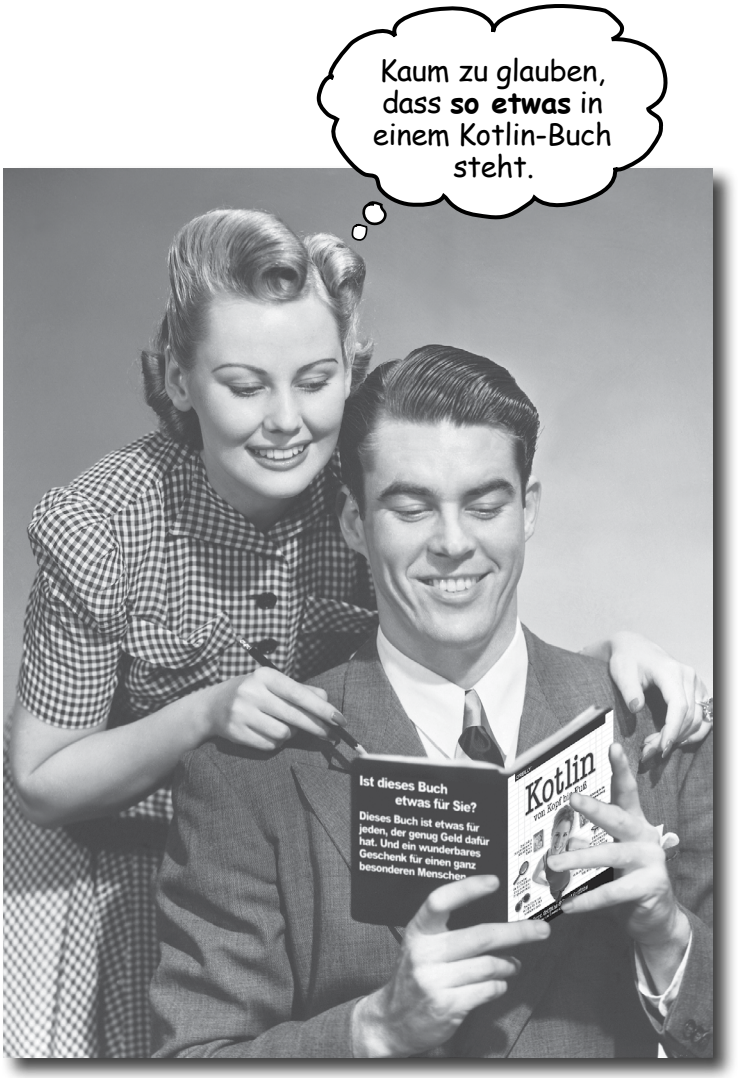
Ein paar Themen haben wir noch für Sie. Wir wollten sie nicht ignorieren, aber es war uns wichtig, dass man unser Buch noch hochheben kann, ohne vorher ein Fitnessstudio besuchen zu müssen. Bevor Sie das Buch zur Seite legen, sollten Sie sich **diese Leckerbissen** nicht entgehen lassen.



Die Inner- und Outer-Objekte haben eine besondere Beziehung zueinander. Inner kann auf die Variablen von Outer zugreifen und umgekehrt.

Wie man dieses Buch verwendet

Einführung



In diesem Abschnitt beantworten wir die brennende Frage:
»Warum steht SO ETWAS in einem Buch über Kotlin?«

Für wen ist dieses Buch?

Wenn Sie alle diese Fragen mit »Ja« beantworten können:

1. Haben Sie schon mal programmiert?
2. Wollen Sie Kotlin lernen?
3. Liegt es Ihnen mehr, Dinge selbst auszuprobieren und das Gelernte anzuwenden, als stundenlang dem öden Sermon eines Dozenten zuzuhören?

Dies ist KEIN Referenzbuch. Kotlin von Kopf bis Fuß ist ein Lehrbuch, aber kein Lexikon der Kotlin-Fakten.

dann ist dieses Buch etwas für Sie.

Wer sollte von diesem Buch besser die Finger lassen?

Wenn Sie mindestens eine der folgenden Fragen mit »Ja« beantworten:

1. Ist Ihr Programmierhintergrund auf HTML ohne irgendwelche Skriptsprachen beschränkt?
   (Wenn Sie schon mal mit Schleifen oder if/then-Logik gearbeitet haben, sollten Sie mit diesem Buch klarkommen, HTML-Tags allein reichen aber nicht.)
2. Sind Sie ein ausgefuchster Kotlin-Programmierer auf der Suche nach einem *Referenz*-Buch?
3. Würden Sie sich lieber die Zehennägel von fünfzehn kreischenden Affen herausreißen lassen, als etwas Neues zu lernen? Glauben Sie, ein Kotlin-Buch sollte *alles* behandeln, besonders das ganze seltsame Zeug, das Sie sowieso nie benutzen und das den Leser vor Langeweile zum Heulen bringt? Und davon möglichst viel?

dann ist dieses Buch ***nichts*** für Sie.

[Hinweis aus der Marketingabteilung: Dieses Buch eignet sich für alle mit einer Kreditkarte oder einem PayPal-Konto.]

Wir wissen, was Sie gerade denken.

»Wie kann *das* ein ernsthaftes Kotlin-Buch sein?«

»Was sollen all die Abbildungen?«

»Kann ich auf diese Weise wirklich *lernen*?«

Und wir wissen, was Ihr *Gehirn* gerade denkt.

Ihr Gehirn lechzt nach Neuem. Es ist ständig dabei, Ihre Umgebung abzusuchen, und es *wartet* auf etwas Ungewöhnliches. So ist es nun einmal gebaut, und es hilft Ihnen zu überleben.

Also, was macht Ihr Gehirn mit all den gewöhnlichen, normalen Routinesachen, denen Sie begegnen? Es tut alles in seiner Macht Stehende, damit es dadurch nicht bei seiner *eigentlichen* Arbeit gestört wird: Dinge zu erfassen, die *wirklich* wichtig sind. Es gibt sich nicht damit ab, die langweiligen Sachen zu speichern, sondern lässt sie gar nicht erst durch den »Dies-ist-offensichtlich-nicht-wichtig«-Filter.

Woher *weiß* Ihr Gehirn denn, was wichtig ist? Nehmen Sie an, Sie machen einen Tagesausflug und ein Tiger springt vor Ihnen aus dem Gebüsch – was passiert dabei in Ihrem Kopf und Ihrem Körper?

Neuronen feuern. Gefühle werden angekurbelt. *Chemische Substanzen durchfluten Sie*.

Und so weiß Ihr Gehirn:

Dies muss wichtig sein! Vergiss es nicht!

Aber nun stellen Sie sich vor, Sie sind zu Hause oder in einer Bibliothek. In einer sicheren, warmen, tigerfreien Zone. Sie lernen. Bereiten sich auf eine Prüfung vor. Oder Sie versuchen, irgendein schwieriges Thema zu lernen, von dem Ihr Chef glaubt, Sie bräuchten dafür eine Woche oder höchstens zehn Tage.

Da ist nur ein Problem: Ihr Gehirn möchte Ihnen einen großen Gefallen tun. Es versucht, dafür zu sorgen, dass diese *offensichtlich* unwichtigen Inhalte nicht knappe Ressourcen verstopfen. Ressourcen, die besser dafür verwendet würden, die wirklich *wichtigen* Dinge zu speichern. Wie Tiger. Wie die Gefahren des Feuers. Oder dass Sie nie wieder in kurzen Hosen Snowboard fahren sollten.

Und es gibt keine einfache Möglichkeit, Ihrem Gehirn zu sagen: »Hey, Gehirn, vielen Dank, aber egal, wie langweilig dieses Buch auch ist und wie klein der Ausschlag auf meiner emotionalen Richterskala gerade ist, ich *will* wirklich, dass du diesen Kram behältst.«

Wir stellen uns unseren Leser als einen aktiv Lernenden vor.

Also, was ist nötig, damit Sie etwas *lernen*? Erst einmal müssen Sie es *aufnehmen* und dann dafür sorgen, dass Sie es nicht wieder *vergessen*. Es geht nicht darum, Fakten in Ihren Kopf zu schieben. Nach den neuesten Forschungsergebnissen der Kognitionswissenschaft, der Neurobiologie und der Lernpsychologie gehört zum *Lernen* viel mehr als nur Text auf einer Seite. Wir wissen, was Ihr Gehirn anmacht.

Einige der Lernprinzipien dieser Buchreihe:

Wir setzen Bilder ein. An Bilder kann man sich viel besser erinnern als an Worte allein und lernt so viel effektiver (bis zu 89 % Verbesserung bei Abrufbarkeits- und Lerntransferstudien). Außerdem werden die Dinge dadurch verständlicher. **Wir setzen Text in oder neben die Grafiken,** auf die sie sich beziehen, anstatt darunter oder auf eine andere Seite. Die Leser werden auf den Bildinhalt bezogene Probleme dann mit *doppelt* so hoher Wahrscheinlichkeit lösen können.

Wir verwenden einen gesprächsorientierten Stil mit persönlicher Ansprache. Nach neueren Untersuchungen haben Studenten nach dem Lernen bei Tests bis zu 40 % besser abgeschnitten, wenn der Inhalt den Leser direkt in der ersten Person und im lockeren Stil angesprochen hat statt in einem formalen Ton. Halten Sie keinen Vortrag, sondern erzählen Sie Geschichten. Benutzen Sie eine zwanglose Sprache. Nehmen Sie sich selbst nicht zu ernst. Würden *Sie* einer anregenden Unterhaltung beim Abendessen mehr Aufmerksamkeit schenken oder einem Vortrag?

Wir bringen den Lernenden dazu, intensiver nachzudenken. Mit anderen Worten: Falls Sie nicht aktiv Ihre Neuronen strapazieren, passiert in Ihrem Gehirn nicht viel. Ein Leser muss motiviert, begeistert und neugierig sein und angeregt werden, Probleme zu lösen, Schlüsse zu ziehen und sich neues Wissen anzueignen. Und dafür brauchen Sie Herausforderungen, Übungen, zum Nachdenken anregende Fragen und Tätigkeiten, die beide Seiten des Gehirns und mehrere Sinne einbeziehen.

Wir ziehen die Aufmerksamkeit des Lesers auf den Lernstoff – nachhaltig. Wir alle haben schon Erfahrungen dieser Art gemacht: »Ich will das wirklich lernen, aber ich kann einfach nicht über Seite 1 hinaus wach bleiben.« Ihr Gehirn passt auf, wenn Dinge ungewöhnlich, interessant, merkwürdig, auffällig, unerwartet sind. Ein neues, schwieriges, technisches Thema zu lernen, muss nicht langweilig sein. Wenn es das nicht ist, lernt Ihr Gehirn viel schneller.

Wir sprechen Gefühle an. Wir wissen, dass Ihre Fähigkeit, sich an etwas zu erinnern, wesentlich von dessen emotionalem Gehalt abhängt. Sie erinnern sich an das, was Sie bewegt. Sie erinnern sich, wenn Sie etwas *fühlen*. Nein, wir erzählen keine herzzerreißenden Geschichten über einen Jungen und seinen Hund. Was wir erzählen, ruft Überraschungs-, Neugier-, Spaß- und Was-soll-das?-Emotionen hervor und dieses Hochgefühl, das Sie beim Lösen eines Puzzles empfinden oder wenn Sie etwas lernen, das alle anderen schwierig finden. Oder wenn Sie merken, dass Sie etwas können, was dieser »Ich-bin-ein-besserer-Techniker-als-du«-Typ aus der Technikabteilung *nicht kann*.

Metakognition: Nachdenken übers Denken

Wenn Sie wirklich lernen möchten, und zwar schneller und nachhaltiger, dann schenken Sie Ihrer Aufmerksamkeit Aufmerksamkeit. Denken Sie darüber nach, wie Sie denken. Lernen Sie, wie Sie lernen.

Die meisten von uns haben in ihrer Jugend keine Kurse in Metakognition oder Lerntheorie gehabt. Es wurde von uns *erwartet,* dass wir lernen, aber nur selten wurde uns auch *beigebracht,* wie man lernt.

Wir nehmen aber an, dass Sie wirklich etwas über Kotlin lernen möchten, wenn Sie dieses Buch in den Händen halten. Und wahrscheinlich möchten Sie nicht viel Zeit aufwenden. Und Sie wollen sich an das *erinnern,* was Sie lesen, und es anwenden können. Und deshalb müssen Sie es *verstehen.* Wenn Sie so viel wie möglich von diesem Buch profitieren wollen oder von irgendeinem anderen Buch oder einer anderen Lernerfahrung, übernehmen Sie Verantwortung für Ihr Gehirn. Ihr Gehirn im Zusammenhang mit diesem Lernstoff.

Der Trick besteht darin, Ihr Gehirn dazu zu bringen, neuen Lernstoff als etwas wirklich Wichtiges anzusehen. Als entscheidend für Ihr Wohlbefinden. So wichtig wie einen Tiger. Andernfalls stecken Sie in einem dauernden Kampf, in dem Ihr Gehirn sein Bestes gibt, um die neuen Inhalte davon abzuhalten, hängen zu bleiben.

Also WIE bringen Sie Ihr Hirn dazu, Kotlin so zu behandeln, als sei es ein hungriger Tiger?

Da gibt es den langsamen, ermüdenden Weg oder den schnelleren, effektiveren Weg. Der langsame Weg geht über bloße Wiederholung. Natürlich ist Ihnen klar, dass Sie lernen und sich sogar an die langweiligsten Themen erinnern *können,* wenn Sie sich die gleiche Sache immer wieder einhämmern. Wenn Sie nur oft genug wiederholen, sagt Ihr Gehirn: »Er hat zwar nicht das *Gefühl,* dass das wichtig ist, aber er sieht sich dieselbe Sache *immer und immer wieder* an – dann muss sie wohl wichtig sein.«

Der schnellere Weg besteht darin, **alles zu tun, was die Gehirnaktivität erhöht**, vor allem verschiedene Arten von Gehirnaktivität. Eine wichtige Rolle dabei spielen die auf der vorhergehenden Seite erwähnten Dinge – alles Dinge, die nachweislich dabei helfen, dass Ihr Gehirn *für* Sie arbeitet. So hat sich z. B. in Untersuchungen gezeigt: Wenn Wörter *in* den Abbildungen stehen, die sie beschreiben (und nicht irgendwo anders auf der Seite, z. B. in einer Bildunterschrift oder im Text), versucht Ihr Gehirn, herauszufinden, wie die Wörter und das Bild zusammenhängen, und dadurch feuern mehr Neuronen. Und je mehr Neuronen feuern, umso größer ist die Chance, dass Ihr Gehirn mitbekommt: Bei dieser Sache lohnt es sich, aufzupassen, und vielleicht auch, sich daran zu erinnern.

Ein lockerer Sprachstil hilft, denn Menschen tendieren zu höherer Aufmerksamkeit, wenn ihnen bewusst ist, dass sie ein Gespräch führen – man erwartet dann ja von ihnen, dass sie dem Gespräch folgen und sich beteiligen. Das Erstaunliche daran ist: Es ist Ihrem Gehirn ziemlich egal, dass die »Unterhaltung« zwischen Ihnen und einem Buch stattfindet! Wenn der Schreibstil dagegen formal und trocken ist, hat Ihr Gehirn den gleichen Eindruck wie bei einem Vortrag, bei dem in einem Raum passive Zuhörer sitzen. Nicht nötig, wach zu bleiben.

Aber Abbildungen und ein lockerer Sprachstil sind erst der Anfang.

Das haben WIR getan:

Wir haben ***Bilder*** verwendet, weil Ihr Gehirn auf visuelle Eindrücke eingestellt ist, nicht auf Text. Soweit es Ihr Gehirn betrifft, sagt ein Bild *wirklich* mehr als 1.024 Worte. Und dort, wo Text und Abbildungen zusammenwirken, haben wir den Text *in* die Bilder eingebettet, denn Ihr Gehirn arbeitet besser, wenn der Text *innerhalb* der Sache steht, auf die er sich bezieht, und nicht in einer Bildunterschrift oder irgendwo vergraben im Text.

Wir haben ***Redundanz*** eingesetzt, d. h. dasselbe auf *unterschiedliche* Art und mit verschiedenen Medientypen ausgedrückt, damit Sie es über *mehrere Sinne* aufnehmen. Das erhöht die Chance, dass die Inhalte an mehr als nur einer Stelle in Ihrem Gehirn verankert werden.

Wir haben Konzepte und Bilder in ***unerwarteter*** Weise eingesetzt, weil Ihr Gehirn auf Neuigkeiten programmiert ist. Und wir haben Bilder und Ideen mit zumindest *etwas* ***emotionalem*** *Charakter* verwendet, weil Ihr Gehirn darauf eingestellt ist, auf die Biochemie von Gefühlen zu achten. An alles, was ein *Gefühl* in Ihnen auslöst, können Sie sich mit höherer Wahrscheinlichkeit erinnern, selbst wenn dieses Gefühl nicht mehr ist als ein bisschen ***Belustigung, Überraschung oder Interesse.***

Wir haben einen ***umgangssprachlichen Stil*** mit direkter Anrede benutzt, denn Ihr Gehirn ist von Natur aus aufmerksamer, wenn es Sie in einer Unterhaltung wähnt als wenn es davon ausgeht, dass Sie passiv einer Präsentation zuhören – sogar dann, wenn Sie *lesen.*

Wir haben ***Aktivitäten*** für Sie vorgesehen, denn Ihr Gehirn lernt und behält von Natur aus besser, wenn Sie Dinge ***tun,*** als wenn Sie nur darüber *lesen.* Und wir haben die Übungen zwar anspruchsvoll, aber doch lösbar gemacht, denn so ist es den meisten Lesern am liebsten.

Wir haben ***mehrere unterschiedliche Lernstile*** eingesetzt, denn vielleicht bevorzugen *Sie* ein Schritt-für-Schritt-Vorgehen, während ein anderer erst einmal den groben Zusammenhang verstehen und ein Dritter einfach nur ein Codebeispiel sehen möchte. Aber ganz abgesehen von den jeweiligen Lernvorlieben profitiert *jeder* davon, wenn er die gleichen Inhalte in unterschiedlicher Form präsentiert bekommt.

Wir liefern Inhalte für ***beide Seiten Ihres Gehirns,*** denn je mehr Sie von Ihrem Gehirn einsetzen, umso wahrscheinlicher werden Sie lernen und behalten, und umso länger bleiben Sie konzentriert. Wenn Sie mit einer Seite des Gehirns arbeiten, bedeutet das häufig, dass sich die andere Seite des Gehirns ausruhen kann; so können Sie über einen längeren Zeitraum produktiver lernen.

Und wir haben ***Geschichten*** und Übungen aufgenommen, die ***mehr als einen Blickwinkel repräsentieren,*** denn Ihr Gehirn lernt von Natur aus intensiver, wenn es gezwungen ist, selbst zu analysieren und zu beurteilen.

Wir haben ***Herausforderungen*** eingefügt: in Form von Übungen und indem wir ***Fragen*** stellen, auf die es nicht immer eine eindeutige Antwort gibt, denn Ihr Gehirn ist darauf eingestellt, zu lernen und sich zu erinnern, wenn es an etwas *arbeiten* muss. Überlegen Sie: Ihren *Körper* bekommen Sie ja auch nicht in Form, wenn Sie nur die Leute auf dem Sportplatz *beobachten.* Aber wir haben unser Bestes getan, um dafür zu sorgen, dass Sie – wenn Sie schon hart arbeiten – an den *richtigen* Dingen arbeiten. Dass Sie ***nicht einen einzigen Dendriten darauf verschwenden,*** ein schwer verständliches Beispiel zu verarbeiten oder einen schwierigen, mit Fachbegriffen gespickten oder übermäßig gedrängten Text zu analysieren.

Wir haben ***Menschen*** eingesetzt. In Geschichten, Beispielen, Bildern usw. – denn *Sie sind* ein Mensch. Und Ihr Gehirn schenkt *Menschen* mehr Aufmerksamkeit als *Dingen.*

Schneiden Sie dies aus und heften Sie es an Ihren Kühlschrank.

Und das können SIE tun, um sich Ihr Gehirn untertan zu machen

So, wir haben unseren Teil der Arbeit geleistet. Der Rest liegt bei Ihnen. Diese Tipps sind ein Anfang; hören Sie auf Ihr Gehirn und finden Sie heraus, was bei Ihnen funktioniert und was nicht. Probieren Sie neue Wege aus.

1 Immer langsam. Je mehr Sie verstehen, umso weniger müssen Sie auswendig lernen.

Lesen Sie nicht nur. Halten Sie inne und denken Sie nach. Wenn das Buch Sie etwas fragt, springen Sie nicht einfach zur Antwort. Stellen Sie sich vor, dass Sie das wirklich jemand *fragt*. Je gründlicher Sie Ihr Gehirn zum Nachdenken zwingen, umso größer ist die Chance, dass Sie lernen und behalten.

2 Bearbeiten Sie die Übungen. Machen Sie selbst Notizen.

Wir haben sie entworfen, aber wenn wir sie auch für Sie lösen würden, wäre das, als würde jemand anderes Ihr Training für Sie absolvieren. Und *sehen* Sie sich die Übungen *nicht einfach nur an*. **Benutzen Sie einen Bleistift.** Es deutet vieles darauf hin, dass körperliche Aktivität beim Lernen den Lernerfolg erhöhen kann.

3 Lesen Sie die Abschnitte »Es gibt keine Dummen Fragen«.

Und zwar alle. Das sind keine Zusatzanmerkungen – ***sie gehören zum Kerninhalt!*** Überspringen Sie sie nicht.

4 Lesen Sie dies als Letztes vor dem Schlafengehen. Oder lesen Sie danach zumindest nichts *Anspruchsvolles* mehr.

Ein Teil des Lernprozesses (vor allem die Übertragung in das Langzeitgedächtnis) findet erst statt, *nachdem* Sie das Buch zur Seite gelegt haben. Ihr Gehirn braucht Zeit für sich, um weitere Verarbeitung zu leisten. Wenn Sie in dieser Zeit etwas Neues aufnehmen, geht ein Teil dessen, was Sie gerade gelernt haben, verloren.

5 Trinken Sie Wasser. Viel.

Ihr Gehirn arbeitet am besten in einem schönen Flüssigkeitsbad. Austrocknung (zu der es schon kommen kann, bevor Sie überhaupt Durst verspüren) beeinträchtigt die kognitive Funktion.

6 Reden Sie darüber. Laut.

Sprechen aktiviert einen anderen Teil des Gehirns. Wenn Sie etwas verstehen oder Ihre Chancen verbessern wollen, sich später daran zu erinnern, sagen Sie es laut. Noch besser: Versuchen Sie, es jemandem laut zu erklären. Sie lernen dann schneller und haben vielleicht Ideen, auf die Sie beim bloßen Lesen nie gekommen wären.

7 Hören Sie auf Ihr Gehirn.

Achten Sie darauf, Ihr Gehirn nicht zu überladen. Wenn Sie merken, dass Sie etwas nur noch überfliegen oder dass Sie das gerade erst Gelesene vergessen haben, ist es Zeit für eine Pause. Ab einem bestimmten Punkt lernen Sie nicht mehr schneller, indem Sie mehr hineinzustopfen versuchen; das kann sogar den Lernprozess stören.

8 Aber bitte mit *Gefühl!*

Ihr Gehirn muss wissen, dass es *um etwas Wichtiges geht*. Lassen Sie sich in die Geschichten hineinziehen. Erfinden Sie eigene Bildunterschriften für die Fotos. Über einen schlechten Scherz zu stöhnen, ist *immer noch* besser, als gar nichts zu fühlen.

9 Erschaffen Sie etwas!

Wenden Sie dies auf Ihre tägliche Arbeit an; setzen Sie das, was Sie gerade lernen, ein, um Entscheidungen in Ihren Projekten zu fällen. Tun Sie irgendetwas, um Erfahrungen zu sammeln, die über die Übungen und Aktivitäten in diesem Buch hinausgehen. Sie brauchen nur einen Bleistift und ein Problem, das es zu lösen gilt … ein Problem, das vielleicht vom Einsatz der Tools und Techniken profitiert, von denen Sie in diesem Buch erfahren.

Lies mich

Das hier ist ein Lehrbuch, kein Referenzwerk. Wir haben bewusst alles herausgestrichen, was an irgendeiner Stelle des Buchs hinderlich für den Lernprozess sein könnte. Beim ersten Lesen sollten Sie unbedingt auch am Anfang des Buchs beginnen. Das Buch geht zu jedem Zeitpunkt davon aus, dass Sie bestimmte Dinge bereits gesehen und gelernt haben.

Wir gehen davon aus, dass Kotlin neu für Sie ist, aber nicht das Programmieren selbst.

Wir erwarten, dass Sie schon mal programmiert haben. Das muss nicht viel sein. Aber Dinge wie Schleifen und Variablen sollten Sie in anderen Sprachen bereits gesehen haben. Übrigens: Im Gegensatz zu vielen anderen Kotlin-Büchern erwarten wir nicht, dass Sie Java bereits kennen.

Zu Beginn zeigen wir Ihnen ein paar Kotlin-Grundkonzepte. Danach sorgen wir dafür, dass Sie Kotlin so schnell wie möglich benutzen können.

In Kapitel 1 geht es um die Grundlagen. Dadurch werden Sie schon in Kapitel 2 Programme schreiben, die tatsächlich etwas tun. Im restlichen Buch bauen wir auf diese Kenntnisse auf. In kurzer Zeit verwandeln Sie von einem *Kotlin-Grünschnabel* in einen *Kotlin-Ninja-Meister*.

Die Redundanz ist wichtig und beabsichtigt.

Ein wesentliches Anliegen eines *Von Kopf bis Fuß*-Buchs ist, dass wir wollen, dass Sie es *wirklich* kapieren. Und wir wollen, dass Sie sich am Ende des Buchs an das Gelernte erinnern. Beständigkeit und Erinnern ist bei den meisten Referenzbüchern nicht das Ziel, aber in diesem Buch geht es ums *Lernen*. Deshalb werden Sie einige der hier gezeigten Konzepte mehr als einmal sehen.

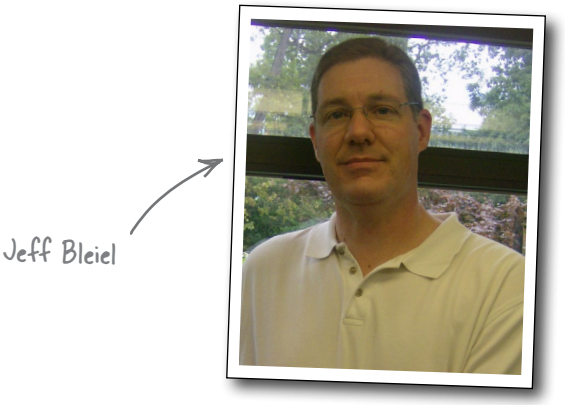
Die Beispiele sind so kurz wie möglich.

Unsere Leser sagen uns, dass sie es frustrierend finden, sich durch 200 Zeilen Code graben zu müssen, um die beiden Zeilen zu finden, die sie wirklich verstehen müssen. Die meisten Beispiele in diesem Buch werden mit so wenig Kontext wie möglich gezeigt, damit der Teil, den Sie lernen sollen, klar und einfach ist. Sie dürfen nicht erwarten, dass der Code robust oder gar vollständig ist. Die Beispiele in diesem Buch wurden speziell für Lehrzwecke geschrieben und sind nicht immer rundum funktionsfähig (obwohl wir versucht haben, das so weit wie möglich sicherzustellen).

Die Aktivitäten sind NICHT optional – Sie müssen die Arbeit selbst machen.

Die Übungen und Aktivitäten sind kein Beiwerk, sondern wesentliche Bestandteile des Buchs. Einige sollen Ihnen beim Erinnern helfen, andere beim Verstehen und wieder andere beim Anwenden. ***Überspringen Sie nichts.***

Danksagungen



Unser Lektor:

Ein herzliches Dankeschön geht an unseren großartigen Lektor **Jeff Bleiel** für all seine Arbeit und Hilfe. Sein Vertrauen, seine Unterstützung und sein Zuspruch haben uns sehr geholfen. Seine vielen Hinweise auf Unklarheiten und Dinge, die wir neu überdenken mussten, haben uns dabei geholfen, ein viel besseres Buch zu schreiben.

Das O'Reilly-Team:

Ein großes Dankeschön geht darüber hinaus an **Brian Foster** für seine Hilfe, *Kotlin von Kopf bis Fuß* auf den Weg zu bringen, **Susan Conant**, **Rachel Roumeliotis** und an **Nancy Davis** für ihre Hilfe, die Ecken und Kanten zu glätten, an **Randy Comer** für das Coverdesign, an das **Early-Release-Team** für die Bereitstellung früher Versionen dieses Buchs zum Download und an **Kristen Brown, Jasmine Kwityn, Lucie Haskins und den Rest des Produktionsteams** dafür, dass sie dieses Buch souverän durch den Produktionsprozess gebracht haben, und für ihre harte Arbeit hinter den Kulissen.

Freunde, Familie und Kollegen:

Das Schreiben eines *Von Kopf bis Fuß*-Buchs ist immer eine Achterbahnfahrt. Das Wohlwollen und die Unterstützung unserer Familie sowie unserer Freunde und Kollegen hat uns dabei sehr geholfen. Ein besonderes Dankeschön geht an **Jacqui**, **Ian**, **Vanessa**, **Dawn**, **Matt**, **Andy**, **Simon**, **Mum**, **Dad**, **Rob** und **Lorraine**.

Die Liste der Menschen, ohne die nichts geht:

Unser großartiges Team technischer Sachverständiger hat hart gearbeitet, um uns ihre Gedanken zu unserem Buch mitzuteilen. Für ihre Anmerkungen sind wir unglaublich dankbar. Sie haben dafür gesorgt, dass die Themen akkurat behandelt wurden und wir dabei noch gut unterhalten wurden. Unserer Meinung nach ist das Buch durch ihre Rückmeldungen noch viel besser geworden.

Unser abschließender Dank gilt **Kathy Sierra** und **Bert Bates** für die Erfindung dieser außergewöhnlichen Buchreihe und dafür, dass sie uns an ihrer Denkweise teilhaben lassen.