Eine Vergleichsstudie: Java vs. Kotlin in Bezug auf Einsteiger in der Android-App-Entwicklung

Franz Lea
FH JOANNEUM Kapfenberg
Graz, Austria
lea.franz@edu.fh-joanneum.at

Trummer Julia
FH JOANNEUM Kapfenberg
Graz, Austria
julia.trummer@edu.fh-joanneum.at

Wieser Stefanie
FH JOANNEUM Kapfenberg
Graz, Austria
stefanie.wieser2@edu.fh-joanneum.at

FH Joanneum, April 19, 2020, Kapfenberg, AUSTRIA
ACM 978-1-4503-XXXX3/20/04.
https://doi.org/10.1145/3334480.XXXXXXX

Abstract
UPDATED—June 28, 2020. This paper presents the results of a comparative study on Kotlin and Java as programming languages for realizing Android Applications. In particular we cover different fields like readability and coding costs. Furthermore, we undertook a closer look at the new features of Kotlin. Conclusively we will go over which coding language fits beginners best. In sum, the Kotlin Project has a third of the Java Project Code and considering all the metrics (readability, languagefeatures and Lines of Code), we would recommend Kotlin to amateur Android developers.

Author Keywords
Java; Kotlin; Application Development; Android; Beginner;

Einleitung
Derzeit werden immer mehr Android-Apps mit Kotlin entwickelt. Diese Programmiersprache ist im Gegensatz zu Java relativ neu und wurde erstmals 2016 veröffentlicht. Zugleich ist Kotlin seit dem ersten Release als Open-Source-Software erhältlich und zielt im speziellen auf JVM (Java Virtual Machine) und Android ab. [5] In diesem Paper vergleichen wir Java mit Kotlin in den Aspekten Aufwand und Lesbarkeit. Darüber hinaus gehen wir auch auf einige Sprachfeatures die Kotlin bietet ein. Zur Unterstützung unserer Vergleichsstudie haben wir uns einen Working Prototype

einer Slideshow App ausgedacht und in beiden Programmiersprachen (Java und Kotlin) umgesetzt. Ziel dieser Studie ist es herauszufinden, ob die Android Entwicklung in Java oder Kotlin einfacher für Einsteiger ist. Die Kriterien für die Vergleichsstudie sind unter anderem Lesbarkeit, Sprachfeatures und Lines of Code.

Kotlin

Kotlin ist, gleich wie Java, eine objektorientierte Programmiersprache. Ein in dieser Sprache geschriebener Code kann in Bytecode für die JVM (Java Virtual Machine) kompiliert werden. Mit Kotlin können große Teile der Java-Boilerplate-Codes[1] verhindert werden.[6]

Java

Java ist eine objektorientierte Programmiersprache und besteht aus den JDK (Java-Entwicklungstools) und der JRE (Java-Laufzeitumgebung). Grundsätzlich wird diese Programmiersprache für die Android Entwicklung verwendet, weil sie sehr bekannt ist. [1]

Lesbarkeit

Die Syntax von Kotlin und Java sind sehr ähnlich und daher ist es schwierig hier einen Vergleich bezüglich der Lesbarkeit zu ziehen. Nur ein Beispiel zu dieser Aussage: Wenn man versucht einen fremdsprachigen Text zu lesen, ist es schwierig einen Satz zu verstehen ohne, dass man die Bedeutung der einzelnen Wörter kennt. Versteht man aber mehrere Wörter und den Kontext, ist es einfacher den Text zu lesen. Deshalb hat die Wahl der Sprache keinen Einfluss auf die Lesbarkeit, solange der Kontext verständlich ist.

[1] Boilerplate-Code: Sind häufig verwendete und meist unveränderte Code-Snippets mit gleicher Funktion.

Link zu den Apps: https://github.com/LeaFranz/KotlinJavaSlideshow

Sprachfeatures

Kotlin bietet viele neue sprachenspezifische Features, die es von Java unterscheiden und Beginnern der Android-App-Entwicklung den Einstieg erleichtern. Im Folgenden wird auf vier solcher Features des Prototypen eingegangen.

Null Safety

Die größten Fehlerquellen in Java-Systemen sind fehlende Null-Checks und die daraus entstehenden Null-Pointer Exceptions zur Laufzeit. In Kotlin können diese Fehler bereits zur Kompilierzeit aufgezeigt werden durch die klare Trennung von Referenzen, die Null sein können und Referenzen, die nicht null sein können. Variablen müssen explizit als Nullable mit dem Anhängen eines Fragezeichens an den Datentyp (string vs. string?) gekennzeichnet werden. Der Safe-Call-Operator (?.) kann dafür verwendet werden, dass immer entweder ein Ergebnis oder Null (ohne Null-Pointer Exception) zurückgegeben wird. [3]

```
gpsText?.text = slide.GPS
```

Datenklassen

Um Klassen als Datencontainer verwenden zu können, braucht es in Java min. einen Konstruktor und Getter/Setter. In Kotlin kann mit dem Keyword "data", eine Datenklasse erstellt werden, die ohne diesen Boilerplate-Code zurechtkommt.

```
data class Slide(val id:Int, var title:String
= "untitled", ...)
```

Der Compiler generiert hier Getter und Setter sowie die Methoden copy(), toString(), hashCode() und equals() automatisch im Hintergrund. [4]

Objekt Deklaration und Singletons

In Kotlin gibt es das Sprachkonstrukt der Objekt Deklaration, mit dem das Singleton Pattern einfacher und kürzer als in Java implementiert. Dafür wird das Keyword "class"

		Kotlin
Dateiname	*LoC*	*Methoden*
Slide.kt	15	1
Slideshow.kt	31	6
MainActivity.kt	119	5
SharedPreference.kt	16	2

Figure 1: In dieser Tabelle sind alle wichtigen Files der in Kotlin implementierten Applikation samt Zeilenanzahl und Methodenanzahl enthalten.

		Java
Dateiname	*LoC*	*Methoden*
Slide.java	57	14
ObservedObject.java	16	2
Slideshow.java	56	7
MainActivity.java	133	7
SharedPref.java	17	2

Figure 2: In dieser Tabelle sind alle wichtigen Files der in Java implementierten Applikation samt Zeilenanzahl und Methodenanzahl enthalten.

mit dem Keyword "object" ausgetauscht und somit kann nur mehr eine Instanz einer Klasse erstellt werden.

```
object Slideshow {....}
```

Einer Objekt Deklaration können Methoden und Properties genau wie einer Klasse hinzugefügt werden. [3]

Delegated Properties und Observer
Auch mit diesem Sprachfeature gibt es eine Erleichterung bei der Verwendung eines Design Patterns. Anstatt einer eigener Implementierung des Observer Patterns kann das in Kotlin integrierte Observable Delegate verwendet werden. Dabei ruft die Delegated Property eine Callback Funktion mit drei Parametern bei einer Änderung auf.

```
private var currentSlideId:  Int by
Delegates.observable(0) { prop, oldSlideId,
newSlideId -> ....  }
```

Eine Delegated Property funktioniert wie eine reguläre Property, aber delegiert das Schreiben und Lesen von Werten an andere Funktionen. [2]

Lines of Code (LoC)

Mittels dieser Metrik sollen die Umfänge der beiden Prototypen miteinander verglichen werden. Das Ziel ist es, herauszufinden, welche der beiden Apps eine geringere Anzahl an Programmzeilen und Methoden hat.

Vergleich
Wie man deutlich an der Anzahl an Codezeilen in den beiden Abbildungen Figure 1 und 2 sehen kann, hat der Java-Prototype um etwa ein Drittel mehr Zeilen als die in Kotlin implementierte Applikation. Des Weiteren ist deutlich zu erkennen, dass die Anzahl an Methoden in der in Java implementierten Applikation doppelt so hoch ist, wie in der Kotlin-App. Dieser eindeutige Unterschied entsteht unter anderem durch die in Kotlin integrierten Sprachfeatures. Einen wesentlichen Beitrag dazu leisten die schnell zu implementierenden Patterns(Singleton, Observer) und die Verwendung einer Data Class, durch die auf Boilerplate-Code verzichtet werden kann. Ein weiterer Unterschied liegt beim "Shufflen" einer ArrayList. Anstatt eine fertige Methode verwenden zu können, braucht man in Java dazu Collections und einen Comparator, wodurch der Code für die gleiche Funktionalität maßgeblich länger wird.

Fazit

Zusammenfassend kann man sagen, dass die in Kotlin implementierte Applikation nicht nur um ein Drittel des Codes kürzer, sondern auch durch die einzelnen Sprachfeatures deutlich einfacher zu implementieren ist. In Bezug auf die Lesbarkeit kann man aufgrund der syntaktische Ähnlichkeit keine Empfehlung abgeben. Durch die Analyse der Prototypen anhand der verwendeten Metriken (Lesbarkeit, Sprachfeatures und Lines of Code) sind wir zu dem Schluss gekommen, dass wir Einsteigern in die Android-App-Entwicklung Kotlin raten würden.

REFERENCES

[1] Madhurima Banerjee, Subham Bose, Aditi Kundu, and Madhuleena Mukherjee. 2018. A comparative study: Java vs kotlin programming in android application development. *International Journal of Advanced Research in Computer Science* 9, 3 (2018), 41.

[2] Husayn Hakeem. 2018. The magic in Kotlin: Delegates. (05 February 2018). `https://proandroiddev.com/the-magic-in-kotlin-delegates-377d27a7b531` Accessed: 18. May 2020.

[3] Marcin Moskala and Igor Wojda. 2017. *Android Development with Kotlin*. Packt Publishing Ltd.

[4] Stephen Samuel and Stefan Bocutiu. 2017. *Programming kotlin*. Packt Publishing Ltd.

[5] Vasiliy. 2017. Kotlin vs Java The Whole Story. (07 September 2017). https://www.techyourchance.com/kotlin-vs-java-whole-story/ Accessed: 18. May 2020.

[6] Vasiliy. 2020. Was versteht man unter einer Boilerplate? (2020). https://kulturbanause.de/faq/boilerplate/ Accessed: 18. May 2020.