Say Kotlin one more time

...

by Hendrik Kokocinski
Senior Android Engineer @ WeltN24 GmbH


SAY KOTLIN
ONE MORE TIME
memegenerator.net


SAY KOTLIN
ONE MORE TIME!
memegenerator.net

Why you should not use Kotlin

- You have to learn a new language (as well as new team members)
- Work out and get used to best practices for usage with Android
- At the mercy of the android toolchain to build your code
- Dependant on Jetbrains in terms of
  - further development
  - future compatibility with Android
- Increase compilation time
- Clean often to resolve annotation processing issues
- No static code analysis tools
- sometimes unspecific dagger errors that are hard to figure out

Why you should use Kotlin

- Easily accessable for java developers (learn it in one day)
- IDE Support
- Compatible with java
  - choose how much kotlin you want in your project file by file
  - low risk giving it a try
  - you can use any Java library
- Strong commercial support by Jetbrains
- small footprint (kotlin-stdlib 835 KB / 5508 methods)
- Open source
- developed with Android in mind

Why you should use Kotlin

- explicit Android support by JetBrains
- open source

Basics

```
class Cat( var name: String, private val age: Int ) : Animal() {

    fun eat( mouse: Mouse ): Poo {

        val poo = Poo( mouse )

        return poo

    }

}
```

Null Safety

```
var string: String = "String"   // variables are null safe by default

var nullableString: String? = null  // use nullable types to allow null values

var length = nullableString?.length ?: 0   // operators to deal with nullable types

string = nullableString // does not compile: different types
```

Type inferance

```
var string: String = ""

var anotherString = "String"


val list: List<String> = ArrayList<String>()

val list2 = ArrayList<String>()

val list3: List<String> = emptyList()

val list4 = emptyList<String>()
```

Sample App

https://github.com/blob0815/kotlin-android-sample

- Kotlin
- Dagger2
- Retrofit / okhttp
- Databinding
- Anko
- (Konvenant)

Let’s see it

Extension functions

- add function to any class
- get rid of static utility classes
- enhance classes from third party libraries
- enhance generated classes
- improve readability
- use them where you tend to forget chained api calls (e.g. Snackbar.show(), SQLiteDatabase.setTransactionSuccessful() etc.)
- use them in the right context (don’t do: Long.millisToMinutes())

Mocking

- Classes and function final by default
- Use PowerMock to make final classes mockable
- @PrepareForTest(YourClass::class)

Kotlin libraries

- **Awesome-kotlin**: (https://github.com/KotlinBy/awesome-kotlin#libraries-frameworks)
- **Anko** (https://github.com/Kotlin/anko)
- **Kovenant** (https://github.com/mplatvoet/kovenant)
- **Kotterknife** (https://github.com/JakeWharton/kotterknife)
- **Hamkrest** (https://github.com/npryce/hamkrest)
- **Mockito-Kotlin** (https://github.com/nhaarman/mockito-kotlin)

Getting started

- Read the documentation (https://kotlinlang.org/docs/reference/)
- Kotlin Koans https://github.com/Kotlin/kotlin-koans
- “Edu Kotlin” - plugin for IntelliJ IDEA
- Try it out: http://try.kotlinlang.org/

questions?.answer ?: finish()

Thank you!