Android - Kotlin, Architecture components

Lukas Prokop

24th November 2017

Ketchup?


Kotlin
KETCHUP PIKANTNY
HOT KETCHUP
TERAZ JESZCZE WIĘCEJ
DOJRZEWAJĄCYCH W SŁOŃCU POMIDORÓW
KETCHUP
POD KONTROLĄ

Island?

Kotlin

http://kotlinlang.org/

Kotlin

- Open source language
- Support for JVM, Android, JavaScript and Native
- Statically-typed
- Kotlin-java interop

Kotlin - history

- Developed by JetBrains, no other language except Scala did not have features, they needed
- July 2011 unveiled Project Kotlin
- February 2012 open sourced
- February 2016 Kotlin v1.0
- Google IO 2017 first-class support on Android
- November 2017 Kotlin v1.2

Kotlin features

- Null safety
- Extension function
- High order function
- Interop with java
- Smart cast
- Default and named arguments
- Data class
- Multi-value return from function

Kotlin - syntax

```
package com.avast.android.kotlinlecture

import android.support.v7.app.AppCompatActivity
import android.os.Bundle

class MainActivity : AppCompatActivity() {

    lateinit var variable: String

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }
}
```

Kotlin - null safety

```kotlin
val stringVal = "Test value"
stringVal = null  // Compile error
```

```kotlin
var stringVar :String? = "Test value"
stringVar.length // Compile error
```

```kotlin
var stringVar :String? = "Test value"
stringVar?.length
```

```kotlin
var stringVar :String = "Test value"
stringVar = null // Compile error
```

Kotlin - extension functions

- Possibility to add methods to class which you don’t own
- i.e. pretty print

```
fun <T> ArrayList<T>.toJsonString(): String {
    val sb = StringBuilder()
    sb.append('[')
    forEachIndexed { index, value ->
        sb.append("\"$value\"")
        if (index != lastIndex)
            sb.append(", ")
    }
    sb.append(']')
    return sb.toString()
}


val list = arrayListOf("val1", "val2", "val3")
Log.d("TAG", list.toJsonString())
```

Kotlin - high order function

- function as argument
- function as return value

```
fun logValue(value: String, format:(String) -> String) {
    Log.d("TAG", format(value))
}


logValue("value", { it.toUpperCase()})
logValue("value", { it.substring(0,3)})
```

Kotlin - java interop

- Possible to call java from kotlin and kotlin from java

Kotlin - smart casts

- If you ensured the type once, you don’t need to cast it

```kotlin
fun smartCast(value: Any) {

    when(value) {
        is String -> value.toUpperCase()
        is Int -> value.absoluteValue
    }
}
```

```java
public void cast(Object value) {
    if (value instanceof String) {
        ((String) value).toUpperCase();
    } else if (value instanceof Integer) {
        int i = (Integer) value + 1;
    }
}
```

Kotlin - default and named arguments

- Readable code especially with high count parameters of the same type

```
fun power(number: Double, exponent: Double = 2.0): Double {
    return Math.pow(number, exponent)
}

Log.d("TAG", "result: ${power(3.0)}")
Log.d("TAG", "result: ${power(3.0, 3.0)}")
Log.d("TAG", "result: ${power(exponent = 4.0, number = 3.0)}")
```

Kotlin - data classes

- Simplify pojo classes
- derives equals/hashCode, toString, copy

```
data class Person(
        val firstName: String,
        val secondName: String,
        val age: Long)


val me = Person("Lukas", "Prokop", 27)
```

Kotlin - multi value return

```
val me = Person("Lukas", "Prokop", 27)
val (firstName, secondName) = me
Log.d("TAG", "first name: $firstName")
Log.d("TAG", "second name: $secondName")
```

Architecture components

- Lifecycle
- Room
- Paging