

Kotlin

Anthony Chen, Edward Gervis, Austin Cho, Michael Rooplall,
Joe Goggin, Rex Al Saud

Brief History of Kotlin

- ❖ Kotlin was developed by JetBrains, the Czech software company behind the widely used IDE IntelliJ.
- ❖ It was initially designed to improve the programming experience for Java developers.
- ❖ Kotlin compiles to the same bytecode as Java and so it runs seamlessly on the JVM.
- ❖ It quickly gained Google's support for Android development.

Evolution of Kotlin

- ❖ Kotlin development initially started in 2010 and was first unveiled in 2011.
- ❖ Kotlin version 1.0 was released in February 2016 and by mid 2017 was officially supported by Google for Android development.
- ❖ Kotlin version 1.2 came out in November 2017 and added compatibility with Javascript platforms in addition to the already existing Java ones.
- ❖ Kotlin version 1.3 dropped in October 2018 and brought significant improvements for asynchronous programming.
- ❖ During May of 2019 Google announced that while Java is still supported, Kotlin was now its preferred language for Android development

Kotlin on the Tiobe Index and Google Trends

- ❖ As of October 2019, Kotlin is ranked #35 with a rating of 0.319% out of 50 programming languages on the TIOBE Index. According to the TIOBE Index October headline, there are 6 new languages that are fighting to enter the top 20. These languages include Kotlin, Dart, Scala, Lua, Rust, and TypeScript.
- ❖ According to Google Trends, following the official release of Kotlin during February 2016, the popularity of Kotlin has been slowly rising until it spiked during May 2017 with a Google Trend score of 69. Interest in the language dipped until October 2017 where it steadily rose up until present day with a score of 100 as of October 2019.
- ❖ The top five states that shown interest for Kotlin are Washington, Utah, California, Montana, and Colorado ranging from greatest to least.

Sources: <https://www.tiobe.com/tiobe-index/> , <https://trends.google.com/trends/?geo=U>

Kotlin in the Industry

- ❖ As mentioned in the previous slides that stated that Google supported Kotlin in its development with Android. Various other companies soon followed suit in taking part with Kotlin.
- ❖ Those Companies are:
 - Uber: using Kotlin for building internal tools
 - Gradle: for writing build scripts
 - Atlassian: powers their Trello on Android (Trello is a web-based Kanban-style list making app)
 - Pinterest: incorporated Kotlin in their application with an average use of 150M users a month.

Kotlin in the Industry Cont.

- ❖ Here are some apps you may recognize that were actually written in kotlin:
 - Postmates, Kickstarter, Basecamp 3, Twidere, and Square.
- ❖ The use of Kotlin in the industry is do to various reasons:
 - First, it's concise, meaning the code made via Kotlin reduced the total number of boilerplate codes. Boilerplate codes or boilerplate, are sections of code that have to be included in many places with little or no alterations. This causes repetitive code to appear again and again; causing debugging to be difficult.
 - Second, Kotlin has shown to have less instances of app crashes and the tools to handle bugs for better stability in app builds.
 - Third, Kotlin is also designed to Fail-Fast; which is a system that will immediately report any issue that could lead to failure when coding.

Sources: <https://kotlinlang.org/docs/kotlin-docs.pdf>, https://site.ieee.org/r1/r1_event/kotlin-and-java-better-together-nejug-meeting/, <https://appinventiv.com/blog/apps-migrated-from-java-to-kotlin/>, <https://ieeexplore.ieee.org/document/8479507>

Main Features of Kotlin

Problems Intended to Solve

Kotlin fixes a series of issues that Java suffers from:

- One of the most common pitfalls in many programming languages, including Java, is that accessing a member of a null reference will result in a null reference exception. In Java this would be the equivalent of a `NullPointerException`.

```
val nullable: String? = item // allowed, always works
```

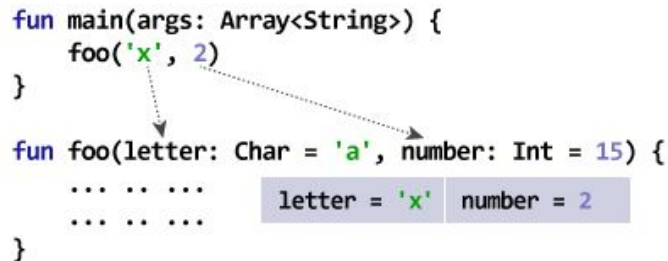
```
val notNull: String = item // allowed, may fail at runtime
```

- Fixes other issues, such as
 - ◆ No raw types (Eg. Declaring a list, but not the type of the list)
 - ◆ Arrays in Kotlin are invariant (Cannot assign an `Array<String>` to an `Array<Any>`)
 - ◆ Kotlin as proper function types (As opposed to Java's SAM-conversions)
 - ◆ Kotlin does not have checked exceptions (A type of exception that must be either caught or declared in the method in which it is thrown)

Main Features of Kotlin

- ❖ Designed with **Java Interoperability** in mind. Existing Java code can be called from Kotlin in a natural way, and Kotlin code can be used from Java rather smoothly as well.
- ❖ **Open Source.**
- ❖ Can be **transpiled to JavaScript** (ECMAScript 5.1 latest)
- ❖ Kotlin can have **defaulted parameters**

```
fun main(args: Array<String>) {  
    foo('x', 2)  
}  
  
fun foo(letter: Char = 'a', number: Int = 15) {  
    ... ..  
    ... ..  
}
```



The diagram illustrates the concept of defaulted parameters in Kotlin. It shows a function call `foo('x', 2)` within the `main` function. Dotted arrows indicate the argument passing: one arrow points from the string literal `'x'` to the `letter` parameter in the `foo` function definition, and another arrow points from the integer `2` to the `number` parameter. The `foo` function definition shows default values: `letter: Char = 'a'` and `number: Int = 15`. Below the definition, a table-like structure shows the actual values passed: `letter = 'x'` and `number = 2`.

- ❖ **Null Safety**

Main Features of Kotlin

- ❖ No Run-Time Overhead.

Many of its functions are inline-only that just become inline code. Kotlin has many optimizations which, specifically, help Android development.

- ❖ **Write Less Code** - Kotlin understands and **infers variable types**.

Unlike Java, which needs you to write everything, Kotlin compiler can understand from the code and write the remaining code, for example, it can infer types in variable declarations. This increases productivity and saves time.

- ❖ Lambda expressions + Inline functions
- ❖ Smart Casts
- ❖ Operator Overloading
- ❖ Coroutines

And Much, Much More . . .

Code Example (1)

```
package com.joegoggin.kotlincalculator

import android.support.v7.app.AppCompatActivity
import android.os.Bundle
import kotlinx.android.synthetic.main.activity_main.*

enum class Operation {
    PLUS, MINUS, MULTIPLY, DIVIDE
}

class MainActivity : AppCompatActivity() {

    private var numberText = ""
    private var num1: Double = 0.0
    private var num2: Double = 0.0
    private var active = true
    private var operation: Operation? = null
```

Code Example (2)

```
private fun calculate(): Double {  
    return when (operation) {  
        Operation.PLUS -> Math.plus(num1, num2)  
        Operation.MINUS -> Math.minus(num1, num2)  
        Operation.DIVIDE -> Math.divide(num1, num2)  
        Operation.MULTIPLY -> Math.mutiply(num1, num2)  
        else -> 0.0  
    }  
}
```

Sources

<https://www.intertech.com/Blog/kotlin-explained-the-java-alternative-why-android-is-supporting-it/>

<https://www.coursera.org/lecture/kotlin-for-java-developers/developing-kotlin-GB6SU>

<https://medium.com/@deeptypednekar17/kotlin-evolution-f93276a4e9b8>

<https://fosbytes.com/kotlin-features-android-programming/>

<https://kotlinlang.org/docs/reference/comparison-to-java.html>

https://medium.com/@hinchman_amanda/kotlin-the-language-the-industry-switches-to-in-5-years-86719aa9c172#targetText=It%20is%20a%20statically%20typed,the%20official%20language%20for%20Android.

<https://appinventiv.com/blog/apps-migrated-from-java-to-kotlin/>

<https://ieeexplore.ieee.org/document/8479507>

https://site.ieee.org/r1/r1_event/kotlin-and-java-better-together-nejug-meeting/

IDE for Android:

<https://developer.android.com/studio>

Other IDE:

<https://www.jetbrains.com/idea/>

Code Example:

<https://github.com/joegoggin/KotlinCalculator>