Kotlin language specification

Version 1.9-rfc+0.1

Marat Akhin Mikhail Belyaev

Introduction

Kotlin took inspiration from many programming languages, including (but not limited to) Java, Scala, C# and Groovy. One of the main ideas behind Kotlin is being *pragmatic*, i.e., being a programming language useful for day-to-day development, which helps the users get the job done via its features and its tools. Thus, a lot of design decisions were and still are influenced by how beneficial these decisions are for Kotlin users.

Kotlin is a multiplatform, statically typed, general-purpose programming language. Currently, as of version 1.9, it supports compilation to the following platforms.

- JVM (Java Virtual Machine)
- JS (JavaScript)
- Native (native binaries for various architectures)

Furthermore, it supports transparent interoperability between different platforms via its Kotlin Multiplatform Project (Kotlin MPP) feature.

The type system of Kotlin distinguishes at compile time between nullable and non-nullable types, achieving null-safety, i.e., guaranteeing the absence of runtime errors caused by the absence of value (i.e., `null` value). Kotlin also extends its static type system with elements of gradual and flow typing, for better interoperability with other languages and ease of development.

Kotlin is an object-oriented language which also has a lot of functional programming elements. From the object-oriented side, it supports nominal subtyping with bounded parametric polymorphism (akin to generics) and mixed-site variance. From the functional programming side, it has first-class support for higher-order functions and lambda literals.

This specification covers Kotlin/Core, i.e., fundamental parts of Kotlin which should function *mostly* the same way irregardless of the underlying platform. These parts include such important things as language expressions, declarations, type system and overload resolution.

> Important: due to the complexities of platform-specific implementations, platforms may extend, reduce or change the way some as-

> pects of Kotlin/Core function. We mark these platform-dependent Kotlin/Core fragments in the specification to the best of our abilities.

Platform-specific parts of Kotlin and its multiplatform capabilities will be covered in their respective sub-specifications, i.e., Kotlin/JVM, Kotlin/JS and Kotlin/Native.

Compatibility

Kotlin Language Specification is still in progress and has **experimental** stability level, meaning no compatibility should be expected between even incremental releases of the specification, any parts can be added, removed or changed without warning.

> Important: while the specification has experimental stability level, the Kotlin language itself and its compiler have different stability levels for different components, which are described in more detail here.

Experimental features

In several cases this specification discusses *experimental* Kotlin features, i.e., features which are still in active development and which may be changed in the future. When so, the specification talks about the *current* state of said features, with no guarantees of their future stability (or even existence in the language).

The experimental features are marked as such in the specification to the best of our abilities.

Acknowledgments

We would like to thank the following people for their invaluable help and feedback during the writing of this specification.

> Note: the format is “First name Last name”, ordered by last name

- Zalim Bashorov
- Andrey Breslav
- Roman Elizarov
- Stanislav Erokhin
- Neal Gafter
- Dmitrii Petrov
- Victor Petukhov
- Vladimir Reshetnikov
- Dmitry Savvinov
- Anastasiia Spaseeva

- Mikhail Zarechenskii
- Denis Zharkov

We would also like to thank Pandoc, its authors and community, as this specification would be much harder to implement without Pandoc's versatility and support.

Feedback

If you have any feedback for this document, feel free to create an issue at our GitHub. In case you prefer to use email, you can use marat.akhin@jetbrains.com and mikhail.belyaev@jetbrains.com.

Reference

If one needs to reference this specification, they may use the following:

> Marat Akhin, Mikhail Belyaev et al. "Kotlin language specification: Kotlin/Core", JetBrains / JetBrains Research, 2020