Kotlin Presentation Abstract

The presentation's chosen language is Kotlin, a language created by JetBrains in 2010 which aims to offer an alternative to Java with functional and imperative features. Kotlin is fully backwards compatible with Java, open source, and adds a variety of safety and quality of life features for developers[1]. A few of the Kotlin specific features are covered more in depth to express their importance to their goal of creating a better user experience than found in Java. One of the most common errors experienced by Java programmers is the Null Pointer Exception or NPE, which is caused by accessing a null member. Kotlin avoids the billion dollar mistake by adding features into its type system to prevent NPE by allowing variables to be nullable or non-nullable. Through the use of nullable receivers, safe calls, and the elvis and !! operators, Kotlin makes it incredibly difficult for a developer to accidentally incur a NPE[2]. As touched on previously, Kotlin also integrates several common functional programming features including Lambda expressions, higher-order functions, and extension functions[3]. The incorporation of "when" expressions also allow for a version of pattern matching to exist within Kotlin[4]. While not being a purely functional language, developers familiar with functional programming will be able to take advantage of these features to get functional performance. Kotlin's choice to use a more concise syntax also offers many benefits to programmers[5]. This choice allows for developers to more easily read and understand code at a glance and also has the bonus of cutting down the total lines of code needed for an equivalent Java program. Kotlin's unique type inference system is also touched on. The implementation of smart casting prevents the need for gratuitous explicit type casting[6]. Coroutines in Kotlin also offer something unique to the language, a concept

similar to threads, coroutines allow a developer to let coroutines swap what execution context they're in[7]. Overloading operators also exists to give the developer freedom to overload predefined operators and create code that's easy to understand for future user's reading it[8]. The presentation also covers noteworthy projects that use kotlin in their codebases. It covers pinterest, Uber, and Duolingo, as well as some of the decision making behind their decisions to move their codebases from Java to Kotlin[9]. Here, some research is shown that was created by these companies to show explicit performance differences between Java and Kotlin as well as research of how the use of either language can impact a developers daily workflow. Google's choice to make Kotlin the preferred language of android development is also featured in this section. This could be the reason that upwards of 60% of android developers use Kotlin in their projects as well as the rapid popularity spike that Kotlin has received and maintained within the past decade[10]. The presentation also features a short tutorial on how a developer curious about Kotlin could get started with their own project. It goes over how to take advantage of their possible student status to get access to free resources and suggestions on frameworks to use like Ktor and Spring[11]. After this, to show the familiarity with the language, a member of the presentation performs a demonstration of a short project made with Kotlin. The project was created in android studio and the aim of the program is to ping an IP address that a user would provide it with. The topic of composable functions and how they are used in the code of the demonstration is then discussed[12]. The presentation ends with an overview of the content covered as well as that state of Kotlin today and if it will ever completely overtake Java's role in the programming ecosystem.

By: Jason Dunn, Jason Scott, Matthew Kubisa, & Azan Nazar

References:

1. https://developer.android.com/kotlin/overview#:~:text=Kotlin%20is%20an%20open%2Dsource,object%2Doriented%20and%20functional%20programming.
2. https://kotlinlang.org/docs/null-safety.html#nullable-types-and-non-nullable-types
3. https://kotlinlang.org/docs/extensions.html#extension-functions
4. https://kotlinlang.org/docs/control-flow.html#when-expression
5. https://www.freecodecamp.org/news/kotlin-vs-java-whats-the-difference/
6. https://kotlinlang.org/spec/type-inference.html
7. https://kotlinlang.org/spec/asynchronous-programming-with-coroutines.html#asynchronous-programming-with-coroutines
8. https://kotlinlang.org/docs/operator-overloading.html
9. https://infostride.com/apps-built-with-kotlin/
10. https://codeop.tech/what-exactly-is-kotlin/
11. https://www.jetbrains.com/community/education/#students
12. https://developer.android.com/develop/ui/compose/mental-model