Linguagem de Programação Kotlin

A linguagem de programação Kotlin recentemente foi oficializada como a “outra” linguagem oficial para desenvolvimento Android.

Assim como o Java, o Kotlin é executado na Java Virtual Machine. Também é totalmente compatível com o Java e não causa nenhum obstáculo nem aumenta o tamanho dos arquivos.

A principal diferença é que o Kotlin requer menos código, o que significa que é uma linguagem de programação mais simples e fácil de ler. Ele também trata alguns erros de forma automática e não exige que você utilizar um ponto-e-vírgula no final das linhas de código.

Essa é uma excelente linguagem de programação, mas se você estiver aprendendo a desenvolver aplicativos Android pela primeira vez, ainda recomendo o Java.

O Kotlin é definitivamente uma linguagem para aprender em um futuro próximo para iniciantes, e o fato de que você ainda pode usar o Android Studio é uma grande vantagem.

O **Kotlin** agora é uma linguagem de programação oficial para o desenvolvimento de aplicativos Android.

**E agora? Largo o Java e foco no Kotlin? Aprendo as duas linguagens?**

Certamente, neste momento você deve estar com muitas dúvidas em sua cabeça de qual caminho seguir e como começar a estudar a mais nova linguagem oficial do Android.

De fato, o **Kotlin** veio para revolucionar a maneira de criar aplicativos Android, mas devemos tomar muito cuidado na hora de escolher qual caminho seguir.

# O que é o Kotlin

O Kotlin é uma linguagem estaticamente tipada, desenvolvida pela JetBrains, cuja sintaxe é mais expressiva e concisa do que a do Java. Com recursos como expressões lambda, sobrecarga de operadores, templates de strings e muito mais.

Como o Java e o Kotlin são altamente interoperáveis, elas podem ser usadas juntas no mesmo projeto. Inclusive, o Kotlin tem como base, e é executado na Máquina Virtual do próprio Java.

O Kotlin é muito intuitivo e fácil de aprender para desenvolvedores Java. A maioria das partes da linguagem são muito semelhantes ao que já sabemos, e as diferenças nos conceitos básicos podem ser aprendidas rapidamente.

Se você é um programador Java experiente, você será capaz de aprender o Kotlin de forma muito fácil, caso contrário, será como aprender qualquer outra linguagem.

## Algumas características interessantes da linguagem.

### Expressiva

Esta é uma das suas qualidades mais importantes, você pode escrever mais com muito menos código.

Segura

O Kotlin tem seguro contra valores nulos, o que significa pe validado situações de valores nulos em tempo de compilação, para evitar exceções em tempo de execução.

### Funcional

O Kotlin é uma linguagem orientada a objetos, não uma linguagem funcional pura.

No entanto, como muitas outras linguagens modernas, ela usa muitos conceitos funcionais, como expressões lambda, para resolver alguns problemas de uma maneira muito mais fácil.

## Outra característica interessante é a maneira como lida com coleções.

### Funções de Extensão

Isso significa que podemos estender qualquer classe e adicionar novos recursos, mesmo se não tivermos acesso ao código-fonte.

### Interoperável

Você pode continuar usando a maioria das bibliotecas e códigos escritos em Java, porque a interoperabilidade entre elas é excelente. É até possível criar projetos mistos, com os arquivos Kotlin e Java coexistindo.

## Kotlin vs Java

Pode ser empolgante, mas também um pouco assustador para qualquer programador trocar de uma linguagem que você já tem uma certa experiência, por algo totalmente novo.

Nas comunidades de desenvolvimento Android, o Kotlin está sendo comparado ao Java da mesma maneira que o Swift foi comparado ao Objective-C.

### Curva de Aprendizado

A curva de aprendizado para usar do Kotlin é bastante baixa: qualquer desenvolvedor Java experiente pode aprender em poucas horas. A documentação oficial da JetBrains é completa e muito bem feita.

Desenvolvedores C# também vão se sentir em casa trabalhando com a linguagem, uma vez que as duas línguas compartilham algumas características.

### Suporte

Para desenvolvimento Android utilizando o Android Studio, existe um suporte muito bom, já que o Android Studio é baseado no IntelliJ, que também foi construído pela JetBrains.

Como já falado, ele também interage perfeitamente com o Java podendo utilizar arquivos das duas linguagens no mesmo projeto.

O uso de bibliotecas Java em projetos Kotlin também é simples: ele simplesmente funciona.

O inverso também é verdade: você pode facilmente incluir qualquer biblioteca escrita em Kotlin dentro do código Java.

### Principais Vantagens

Talvez a vantagem mais importante do Kotlin sobre o Java seja o conjunto de recursos. O Kotlin adiciona novas capacidades importantes nos projetos Android que ainda não é possível encontrar em Java.

#### Algumas capacidades incluem.

##### Proteção Contra Nulo (Null Safety)

Essa proteção elimina a maioria dos problemas referências nulas, tornando todos os tipos não nulos por padrão – o que significa que o compilador não permitirá que você use uma variável não inicializada, ou com possibilidade de nulo.

Se você precisar de uma variável nula, você deve declarar o tipo como “anulável”, adicionando um ponto de interrogação após o tipo.

var nonNullable: String = "My string"

var nullable: String?

##### Funções Estendidas

Você pode adicionar comportamentos a uma classe sem estendê-la diretamente. Através de uma função de extensão, você pode chamar uma função de um objeto como se fosse parte de sua classe.

Poderíamos criar uma função de extensão para String chamada novaFuncao e chamá-la de minhaString.novaFuncao(). Como sabemos, novaFuncao não é uma função de String, mas sintaticamente ela é.

Então, como isso funciona? Você só precisa adicionar a função a classe String conforme exemplo abaixo:

StringExtensions.kt

fun String.novaFuncao(): String {

// implementacao

}

##### Lambdas

Se você estiver familiarizado com JavaScript (ou C# e muitos outros), você provavelmente já conhece sobre as funções lambdas.

Uma função lambda recebe funções como parâmetros ou retorna uma função. As funções podem ser armazenadas em variáveis para uso posterior, passadas como parâmetro ou criadas dentro de outra função.

// Criando a função

fun networkCall(onSuccess: (ResultType) -> Unit,

onError: (Throwable) -> Unit) {

try {

onSuccess(myResult)

} catch(e: Throwable) {

onError(e)

}

}

// Chamando a função

networkCall(result -> {

// use successful result

}, error -> {

// handle error

});

##### Classes de Dados (Data Classes)

Este recurso traz uma grande economia de tempo. Pois a maioria das nossas aplicações são orientadas por dados, muitas vezes precisamos criar classes com apenas propriedades e campos para armazenar dados.

Em Java, isso é muito custoso, exigindo os métodos get/set para cada campo. Com o Kotlin, podemos declarar a classe e todas as suas propriedades em uma única linha. O compilador irá gerar todos os getters e setters.

data class User(var name: String, var age: Int)

##### Imutabilidade

Uma grande preocupação que os desenvolvedores precisam ter ao desenvolver aplicativos multithread é a administração do estado. Se uma variável é mutável, ela pode ser alterada por qualquer thread que possa acessá-la.

Isso significa que, se os dados puderem ser alterados por várias fontes, você precisará implementar manualmente a sincronização, o que evita a perda de dados, mas aumenta a complexidade do código e o tempo de execução. Se os dados nunca podem ser alterados, eles podem ser acessados por várias threads sem problema, uma vez que são imutáveis.

No Kotlin, você pode declarar variáveis com as palavras-chave var e val. O primeiro declara uma variável que pode ser alterada; O último uma variável que uma vez atribuído um valor nunca pode mudar.

Isso dá ao desenvolvedor e ao compilador confiança de que a variável não pode ser alterada. Em Java temos funcionalidade semelhante com palavra-chave final.

A biblioteca padrão do Kotlin inclui uma série de interfaces de coleções imutáveis e funções para ajudá-lo a escrever um código imutável. Por exemplo, a função listOf() cria uma lista imutável.

##### Co-rotinas

Normalmente, quando os desenvolvedores precisam executar uma tarefa de longa execução, como uma chamada externa de rede ou carregar um arquivo do disco, a thread de chamada fica bloqueada aguardando a conclusão da operação.

As co-rotinas nos permitem executar esses tipos de operações sem bloquear seu código. Em vez disso, é utilizada uma operação mais leve chamada co-rotina.

Podemos escrever um código aparentemente síncrono que é, na verdade, assíncrono e durante a compilação, o código será transformado para assíncrono. Isso significa que essa técnica não depende da máquina virtual ou do sistema operacional.

#### Desvantagens

Embora desenvolver aplicativos Android utilizando o Kotlin, em muitos casos é melhor do que com Java, temos alguns desafios.

Por exemplo, um aplicativo construído com Kotlin provavelmente resultará em um tamanho de pacote de arquivos maior do que um construído puramente em Java. Isso ocorre porque o Kotlin tem sua própria biblioteca que é adicionada no seu aplicativo.

Além disso, o tempo de compilação de códigos Kotlin é um pouco mais lento usando o Gradle, que pode ser frustrante. Embora o Gradle também seja lento usando apenas Java, não é tão lento quanto com Kotlin.

### Presente ou Futuro?

Como vimos, o Kotlin roda em cima da Máquina Virtual do Java e facilita muitas coisas para o desenvolvedor Android, então basicamente, ele ajuda a escrever códigos mais limpos e de forma mais fácil.

Depois disso, uma vez que o código está sendo executado, ele cria um arquivo de classe semelhante ao que é obtido quando o arquivo .java é executado.

Então, no final das contas, o Kotlin é um recurso complementar e um bônus para o desenvolvedor Android poder codificar de forma eficiente, com muito mais desempenho e de forma amigável.

Em um futuro a médio prazo, é provável que a maioria dos aplicativos Android sejam desenvolvidos utilizando apenas o Kotlin, mas a chance disso acontecer em um futuro próximo é muito pequena.

Então, não precisamos sair de forma desesperada tentando aprender a nova linguagem e esquecer o Java completamente, permitindo os desenvolvedores fazerem essa transição de forma tranquila.

#### Para Quem Já Desenvolve Aplicativos

Espero que esse post tenha te ajudado a sanar várias dúvidas sobre o Kotlin.

Para você, que já desenvolve seus aplicativos Android utilizando Java, essa é uma ótima oportunidade de dar um upgrade nos seus conhecimentos, habilidades e diferenciar-se dos outros desenvolvedores.

Se você quiser aprender ainda mais sobre o Koltin e mergulhar nessa nova linguagem, dá uma olhada nesse Treinamento Completo que foi preparado para revelar absolutamente todos conceitos essenciais do Kotlin para desenvolvimento Android.

Depois é só juntar suas habilidades em desenvolvimento Android e começar a criar seus projetos com o Kotlin agora mesmo.

#### Para Quem Está Começando a Desenvolver Aplicativos

Minha recomendação, comece ou continue estudando o Java para desenvolver seus aplicativos Android e paralelo a isso, ao poucos, comece a estudar o Kotlin e fazer pequenas coisas com a linguagem.

Imagino que você já esteja estudando o Java (caso não, comece por aqui), e em paralelo, comece a estudar a linguagem Kotlin. Na hora de desenvolver seus projetos de estudo, fala duas versões, uma em Java e outra em Kotlin.