

Conclusão

Transcrição

[00:00] Chegamos ao final de curso de Android com Kotlin Parte 3. E se você chegou até aqui você está de parabéns, porque aqui a gente aprendeu bastante coisa, bastante conteúdo e passamos por diversos desafios relacionados ao Kotlin com o Android. E agora para gente tentar retomar tudo o que a gente viu aqui dentro do curso, vamos lembrar como tava desde o começo e como ele chegou aqui até o final.

[00:19] Então vamos lá. A princípio, quando a gente começou o curso a gente pegou um projeto com a continuidade do Android com Kotlin Parte 2. E o que esse projeto dava para gente? Basicamente, a gente conseguia apenas fazer com que o nosso usuário ele criasse essas transações aqui, seja de receita ou seja despesa, e apenas isso. O nosso usuário conseguia criar essas transações.

[00:37] Só que a gente percebeu que esse tipo de comportamento ele não é o ideal porque a gente também precisa fazer com que ele altere ou remova essas transações, pra ele conseguir ter uma experiência melhor. E a gente conseguiu dar continuidade nesses processos, então a gente fez com que no momento que ele tocasse aqui nessa transação ele fosse aqui e colocasse aqui um valor para ela, alterasse, colocasse uma data diferente. A gente conseguiu permitir esse tipo de comportamento para o nosso usuário.

[01:01] Agora se ele quiser remover ele também consegue, a gente conseguiu implementar isso. Mas a gente viu que teve diversos desafios para gente conseguir implementar isso utilizando boas práticas do Kotlin, utilizando recursos novos do Kotlin. Então vamos agora retomar esses recursos que a gente viu.

[01:15] A princípio a gente viu que quando a gente tava aqui no nosso projeto, deixa eu entrar aqui dentro do Java "br.com.alura.financask" e aqui no "ui" "dialog" a gente começou com esse "AdicionaTransacaoDialog", a gente começou com ele. E a gente viu que ele tava com um código bem grande e para a gente criar, por exemplo, a função para alterar as transações, a gente teve que fazer um novo dialog, que é esse "AdicionaTransacaoDialog".

[01:38] E a princípio a gente começou a fazer uma cópia entre eles e reparamos que tinha muitos comportamentos que eram comuns, tinha muitos comportamentos que dava pra ser reutilizado. E a gente aprendeu que uma das técnicas pra gente fazer isso foi justamente fazendo a herança, criando essa "FormularioTransacaoDialog". E a gente começou a aprender como que a gente faz herança aqui no Kotlin.

[01:55] Como que a gente por exemplo consegue criar uma classe abstrata, como a gente consegue criar properties abstrata como também funções abstratas e tudo que a gente consegue reutilizar a gente também aprendeu como a gente consegue fazer aqui no Kotlin. A gente aprendeu esse recurso que é bem avançado até, seja em qualquer linguagem de programação, que é justamente aprender a usar a herança.

[02:13] A gente aprendeu a fazer isso aqui no curso. Além disso, quando a gente estava nesse processo de herança a gente também começou a fazer uma refatoração aqui na nossa Activity. Lembra que a gente tava usando aqui a nossa Activity, e aqui a gente estava precisando criar essa View da Activity como uma property, que ela pegava esse membro aqui que é o "window.decorView".

[02:30] E a gente viu que ele não pode ser inicializado logo no momento que ela é criada, porque ela só vem com a view de fato na Activity, quando a gente chama o "setContentview". E a gente percebeu que pra lidar com esse tipo de situação, a gente teve diversas peculiaridades. Como é o caso de lidar com diversas variáveis que são nulas, aí a gente viu que o Kotlin tem aquela questão do null safety que é para nos proteger de receber um Null Pointer Exception.

[02:50] A gente aprendeu a lidar com esses valores nulos, vimos que é uma coisa bem comum que é justamente quando a gente está lidando com esses valores que podem receber valor nulo com esse ponto de interrogação, a gente aprendeu diversas técnicas para lidar com isso.

[03:00] E pra lidar com essa parte aqui da "viewDaActivity" a gente viu duas estratégias diferentes além dessa parte do null. A gente viu o "LateInit" que é aquela inicialização atrasada, que permite que a gente coloque aqui a nossa property e inicialize ela em outro momento, mas a gente viu que tem certas peculiaridades.

[03:16] Como é o caso de sempre usar o "var" ao invés do "val", e também a gente aprendeu a delegação de properties que foi por meio dessa estratégia do "lazy" que é a estratégia preguiçosa. A gente aprendeu que essa técnica ela nos permite fazer com que essa inicialização ocorra sem dar aquele bug, porque só vai acontecer apenas quando alguém usar essa variável pela primeira vez.

[03:33] A gente aprendeu essa técnica aqui de inicialização. Além disso, a gente aprendeu o seguinte, a gente viu que quando a gente estava implementando, por exemplo, as nossas dialogs, a gente estava implementando eles por meio daquele recurso de delegate daquela Design Pattern que eu tinha comentado com vocês. A gente tinha aquela implementação do Object Expression.

[03:53] E a gente tinha uma interface que tinha uma single abstract method, que é o método abstrato único, que permite com que a gente converta para expressões lambdas as interfaces do Java. E a gente viu que a gente não consegue fazer isso no Kotlin. E como que a gente fez também pra permitir essa expressão lambda? A gente utilizou as High Order Functions.

[04:10] A gente aprendeu que as High Order Functions é uma técnica na qual a gente consegue mandar funções como parâmetro de outras funções, e a gente consegue executá-las em algum momento. Da mesma maneira como a gente fez no delegate. Só que a gente não precisa mais utilizar as interfaces pra isso, e a gente consegue fazer utilizando esse recurso que o Kotlin nos disponibiliza. A gente aprendeu também essa técnica.

[04:28] Além disso, quando a gente estava utilizando aqui a nossa app, a gente percebeu que quando o nosso usuário, ele adicionava uma transação, por exemplo, de R\$ 100, na hora que ele rotacionada, essa transação sumia, a gente perdia essa referência. Por que a gente perdia? Porque a gente utilizava um objeto que estava atrelado ao objeto da Activity. A nossa lista de transações, ela ficava atrelada ao objeto da Activity. Então, por isso morria.

[04:51] E pra isso, a gente aprendeu uma técnica, na qual a gente conseguiu manter essa lista de transações de maneira estática, na memória. E assim dessa forma, agora como vocês podem ver, as nossas transações elas são mantidas na nossa aplicação mesmo que a Activity morra, ou aconteça alguma coisa inesperada. A gente viu isso também.

[05:05] Foi basicamente isso que a gente viu aqui no curso galera, eu sei que foi bastante conteúdo, realmente é uma coisa bem poderosa o que a gente aprendeu aqui. E eu espero que vocês tenham gostado bastante do conteúdo. Eu espero também um feedback de vocês pra falar o que vocês acharam.

[05:21] O que vocês gostariam de ver depois, pra gente de repente fazer melhorias aqui no curso. E é isso que eu queria passar pra vocês. Obrigado pela atenção, obrigado por ter feito esse curso. Um abraço, até mais!