

Criando a classe DAO

Transcrição

[00:00] Por mais que a gente tenha permitido o nosso usuário de adicionar, alterar e remover transações, ainda existe um caso peculiar que a gente não tomou tanta atenção e que é importante a gente estar sempre reparando. Vamos dar uma olhada nele.

[00:12] Então como que seria esse caso peculiar. Perceba que atualmente a gente só usou a nossa aplicação nesse modo retrato, o portrait. Só que o que acontece, por exemplo, se com essas transações que a gente adicionou aqui, no caso só adicionou uma mesmo, vou até adicionar mais uma. O que acontece, por exemplo, se eu tenho uma ou mais transações, e eu rotaciono a minha app pra que ela fique no modo paisagem, modo portrait. O que acontece?

[00:33] Eu vou rotacionar, e as nossas transações somem. Por que será que isso acontece? Inclusive se a gente usar no modo portrait, se a gente for adicionar uma transação, reparem que funciona, ela é adicionada, e se a gente volta no modo portrait, a gente tava no landscape, a gente voltar pro portrait, ela some de novo. Por que isso acontece?

[00:53] Pra gente conseguir entender isso, vamos entrar na nossa Activity e ver como está funcionando, por exemplo, a técnica pra manter essas transações. Vamos lá, entrando aqui na "ListaTransacoesActivity". Reparem que as nossas transações atualmente estão todas concentradas dentro dessa property que a gente criou, que é essa "transacoes" aqui, que é a MutableList, a gente já viu como ela funciona.

[01:13] Só que o ciclo de vida dela, ela fica viva em qual momento? Justamente no momento em que a instância da nossa Activity ainda está de pé. E o que acontece com essa instância da Activity quando a gente rotaciona? Basicamente aquela instância que a gente tem, ela morre, ela acaba se perdendo porque ela é destruída. Isso é um comportamento comum do android mesmo.

[01:36] A nossa Activity, ela tende a morrer, ela tende a ser destruída a qualquer momento. Portanto, manter essas informações aqui apenas como responsabilidade da Activity não é uma prática na qual vai garantir que nossas informações sejam mantidas. Então considerando justamente esse aspecto, a gente precisa aplicar uma técnica na qual as transações que a gente vai adicionar na nossa ListaTransacoes, elas não podem ficar aqui diretamente na Activity.

[01:58] Elas precisam ficar no lugar que vai ficar responsável e manter esses dados, num lugar que vai ficar persistente essas informações. E como a gente já viu no primeiro curso lá de Android, é justamente aquelas classes reconhecidas como DAO (Data Access Object). Então a gente vai criar agora um DAO pra manter as nossas transações e ele vai ficar responsável em manter as nossas transações.

[02:19] Se a Activity, ela for destruída, ou então se de repente a Activity girar, ou qualquer coisa que aconteça, que tenha um risco da Activity morrer as informações vão estar no DAO pra gente. É justamente isso que a gente vai fazer. Qual o nosso primeiro passo, por exemplo, pra criar um DAO? A gente pode vir aqui no nosso projeto, aqui em java, aqui nos pacotes. E agora a gente vai criar um pacote chamado de DAO.

[02:39] Então, pacote, DAO. Opa, coloquei errado, vou dar um Shift+F6, DAO. Agora sim. Agora que a gente colocou esse pacote, a gente pode criar a nossa classe, então Alt+Insert "Kotlin File/Class", e aqui vai ser a "TransacaoDAO". Vai ser uma classe aqui.

[02:59] Repara que colocou esse comentário, vou apagar ele aqui. Então agora que a gente tem a nossa "TransacaoDAO", como a gente viu, a gente tem aqueles comportamentos de adicionar, remover e alterar uma transação. Como também a gente precisa listar essas transações pra quem vai utilizar elas. A princípio como a gente pode estar fazendo isso?

[03:14] A gente pode começar com uma função de adiciona. Então "adiciona" e o que a gente vai adicionar? A gente sempre vai adicionar uma transação, da mesma maneira que a gente fez lá na nossa Activity, só que agora a responsabilidade vai ser pra cá. Então, adiciona, a gente vai receber uma transação. E aqui a gente vai abrir o corpo dela.

[03:33] Só que repara que pra adicionar uma transação, deixa eu até escrever direito que eu escrevi errado, agora foi certinho. Pra gente adicionar a transação, a gente precisa de um meio persistente, a gente precisa de uma estrutura na qual vai manter essas transações. A princípio vamos colocar numa lista interna aqui da transação DAO.

[03:49] Vamos fazer com que essa lista interna ela mantenha essas informações pra gente. Da mesma maneira de como a gente fez lá na Activity. A gente vai resolver isso, vai deixar de uma maneira mais coerente. Mas a princípio vamos fazer dessa maneira só pra gente implementar toda a transação DAO e depois vamos ajustando aos poucos.

[04:02] Vamos criar essa property, então vai ser "transações", da mesma maneira quando a gente viu lá na Activity que ela vai ser aqui uma MutableList de transação. E aqui, a princípio, quando essa instância foi criada, ela vai ser uma lista vazia, pra poder ter uma instância dela. Então é uma MutableListOf, e aqui a gente está criando uma lista de transações vazia.

[04:35] Então quando alguém chamar esse adiciona, o que a gente vai fazer é a mesma coisa que a gente fazia na Activity. Só que agora a responsabilidade vai ser da "TransacaoDAO", de poder persistir essas informações. Agora a gente está colocando aqui essa informação. Agora a gente vai dar um "add" e vai adicionar essa transação. A gente fez o primeiro comportamento.

[04:52] Agora o que a gente tem também? A gente tem o comportamento de alterar. Então da mesma maneira, a gente vai vir aqui e vai colocar o "altera", e pra alterar, como a gente viu, primeiro precisamos do que a gente quer alterar e a posição dele. A gente vai querer uma transação também, então "transação: Transacao", ele até já completa pra gente. E agora uma posição, então "posição: Int".

[05:13] Então agora a gente tem essa cara aqui, o que a gente pode fazer? A gente pode chamar a lista de transações, a gente pode fazer aquele método indexável que é pegando a posição aqui como se fosse um array e atribuindo pra essa "posicao" a transação que a gente está querendo aqui. Dessa maneira a gente vai lá e altera.

[05:29] Por fim, o que a gente precisa agora é remover. Então "remove", e aqui a gente só precisa do que? Da posição, então "posicao: Int", e a gente vai lá, "transacoes.removeAt(posicao)". A gente já conseguiu implementar um comportamento como que a gente espera desse nosso DAO, que é adicionar, alterar e remover.

[05:51] E aí também ele já remove essa lista aqui pra quem for utilizar. Então o que a gente precisa fazer? A gente precisa fazer com que a nossa Activity use esse DAO que a gente criou. Então vou dar um Ctrl+Shift+F12 pra aumentar o código, e vamos fazer esse processo agora. Vamos lá.

[06:04] Qual o nosso primeiro passo? Repara que aqui, a gente vai precisar de um DAO pra poder usar em diversos pontos, como no caso esse de adiciona, como no caso do de remove, como também no de altera. Então a gente precisa de uma property. Essa property, ela vai vir aqui "private val dao = TransacaoDAO". Então a gente fez aqui a instância, aliás fiz a instância agora. Então a gente tem aqui a nossa property.

[06:30] Então a gente pode vir aqui no adiciona por exemplo, então, adiciona, a gente vai fazer o "dao.adiciona" então já está delegando pro DAO essa responsabilidade de adicionar a transação. Da mesma maneira, a gente vai vir aqui, e vai pegar esse "remove" aqui. Então agora a gente vai usar o "dao.remove", então a gente está removendo.

[06:52] Da mesma maneira o altera, então agora vai ser "dao.altera(transacao, posicao)". Então agora a gente conseguiu delegar esses comportamentos principais pro nosso DAO. Aí agora o que a gente precisa fazer? Repara que aqui a

princípio, a gente está tendo aqui essas "transacoes" que a gente está criando na nossa Activity. A Activity está se responsabilizando por isso, ou seja, agora essa responsabilidade vai ser do nosso próprio DAO.

[07:18] Portanto galera, a gente vai fazer com que a inicialização dela seja feita pelo nosso próprio DAO, então "dao.transacoes", essas transações vão vir do nosso DAO. Repara que ele está reclamando aqui porque precisa ser inicializado a princípio. Por que isso acontece? Porque o nosso DAO está sendo inicializado logo depois dessa "transacoes" então é a mesma peculiaridade que a gente tem na programação nas outras linguagens.

[07:41] A gente precisa agora esse DAO aqui em cima, aí agora sim a gente vai ter acesso a ele pra poder inicializar a nossa lista de transações. Vamos testar pra ver o que acontece, Alt+Shift+F10, veja que o Android Studio conseguiu executar. Vamos testar aqui.

[07:55] Se a gente adicionar uma transação, vamos ver o que acontece, a gente adiciona, e o comportamento que a gente espera agora é no momento que a gente rotaciona, pra ver se ela mantém. Vamos testar pra ver se isso acontece. Então a gente testa, e olha só, ela ainda está sumindo. Porque isso acontece?

[08:08] Porque acontece o seguinte, por mais que a gente esteja delegando pro DAO, por mais que é ele que está sendo responsável por isso, até então essa nossa lista de transações aqui ela também está sendo referida a instância do nosso DAO, ou seja, a instância do nosso DAO também está morrendo junto com a Activity.

[08:25] Em outras palavras, esse nosso recurso no qual a gente está salvando, ele está sendo salvo dentro da instância. E o que a gente precisa fazer? A gente precisa fazer com que ele seja salvo dentro da memória, de uma forma estática, da mesma maneira que a gente via lá no Java quando a gente usava o static, por exemplo.

[08:38] É justamente isso que precisamos fazer. E a gente vai ver como a gente pode estar fazendo essas variáveis, essas properties estáticas no próximo vídeo. Até já!