03

Aplicando a herança na classe de alteração

Transcrição

[00:00] Agora que a gente conseguiu colocar a herança aqui na nossa "AdicionaTransacaoDialog", vamos começar com processo para, também, utilizar a herança aqui na "AlteraTransacaoDialog". Vamos lá, como que a gente pode começar a fazer isso?

[00:12] Vou começar aqui do topo da classe e vou descendo aos poucos, a princípio, eu vou modificar aqui essa parte dos parâmetros, para poder ficar mais visível, Ctrl+Alt+L aqui para poder ajustar. E agora eu vou fazer a herança, portanto, os dois pontos aqui, e agora a gente vai herdar de quem, de "FormularioTransacaoDialog". Reparem que a gente tem que colocar o Construtor que a gente vai utilizar, no momento que a gente faz a herança, qual é o construtor?

[00:35] É o único construtor disponível que a gente tem, que é enviando o context, enviando a ViewGroup". Então esse é o princípio que a gente tem para poder usar a herança dessa classe. Agora, como a gente viu, a ideia quando a gente está utilizando essa parte da herança, é reutilizar membros que a gente já tem dentro dela, que a gente não precisa declarar de novo, como é o caso dessas properties aqui, a "viewCriada", o "campoValor", o "campoCategoria", entre outros membros que a gente tem aqui dentro dessa "AlteraTransacaoDialog".

[01:01] Agora, o que a gente vai fazer? A gente vai remover esses membros que a já tem aqui, para poder reutilizá-los, a gente vai fazer isso agora. A princípio, a gente já tem todas essas properties lá nossa SuperClass, vamos apagar. Agora vamos ver, aqui, essa função chama, a gente já tem essa função lá dentro do nosso SuperClass, se a gente olhar, a gente já tem essa função, aqui para gente, só que na "Altera", ela tem um pouquinho de peculiaridade, porque reparem que aqui, a gente tá mandando uma transação.

[01:29] Sendo que, a chamada dela dentro da SuperClass, ela recebe o tipo, então já começa com um pouquinho de peculiaridade, como também, aqui dentro, a gente faz aquele processo de inicialização. Então considerando esse aspecto que tem uma certa diferença, entre as duas, a gente vai manter essa função chama e não vamos removê-la, depois a gente vai ver como a gente pode tentar reutilizar uma parte do código aqui, a princípio, não vamos remove-la.

[01:53] Vamos descer um pouquinho para outra função, que é essa "configuraFormulario", reparem que a gente tem exatamente o mesmo comportamento, portanto, a gente pode chegar aqui e removê-la sem nenhum problema. A gente vem aqui, seleciona ela e apaga. Agora a gente tem a "tituloPor", também, a gente tem exatamente o mesmo comportamento, a única coisa que muda é que lá vai ser "Altera" e, aqui, vai ser "Adiciona", a gente vai ver como resolver isso aos poucos.

[02:13] Não tem muita peculiaridade como a gente tem aqui na chama, como vimos, é a mesma chamada, são os mesmos parâmetros. Aqui no "converteCampoValor", também, aqui no "configuraCampoCategoria" também, a mesma coisa, o mesmo tipo de chamada, "categoriasPor" também, o mesmo tipo de chamada, "configuraCampoData" também, o mesmo tipo de chamada e, aqui, no "criaLayout" também, inclusive, "criaLayout" pode só ser responsabilidade da nossa transação, "formularioTransacaoDialog".

[02:39] Porque ele que vai criar o Layout, e vai o mesmo para todos, a gente não precisa se preocupar com ele, veja que a gente conseguiu fazer uma limpeza bem bacana e, agora que a gente fez essa limpeza, vamos retomar novamente o que ficou aqui e vamos ver o que pode estar sendo modificado. Voltando aqui nos parâmetros, que são as nossas properties, reparem que essa property aqui não está sendo mais utilizada, porque a gente não precisa mais dela para poder criar o layout, quem tá criando o layout e o "formularioTransacaoDialog".

[03:02] Inclusive, a gente manda para lá, portanto, pode ser um parâmetro normal. O context ainda está sendo utilizado aqui dentro, então, por enquanto, a gente não pode estar de desfazendo dele, porque ele ainda está sendo utilizado nessa parte que a gente está retornando essas categorias. Essa parte aqui dos properties, a gente conseguiu resolver, agora vamos entrar nessa função chama.

[03:23] Reparem que nela, a gente está recebendo a transação e recebendo o delegate, e aqui dentro a gente pega esse tipo, a gente configura o campo data, configura o campo categoria e configura também o formulário, e reparem que até nem compila, porque não compila? Porque o acesso dessas funções, está privado, eles estão sendo acessíveis apenas para a "formularioTransacaoDialog",

[03:42] Se a gente reparar aqui também novamente, nessa "TransacaoDialog", reparem que nessa função chama, que é onde chama o "CampoData", o "configuraCampoCategoria" e a "configuraFormulario", exatamente a mesma chamada está sendo feita aqui, ela pode ser reutilizada também. Portanto, ao invés de a gente ter que lidar com esse tipo de chamada, dentro do "AlteraTransacaoDialog", a gente pode reutilizar aquela chamada do "FormularioTransacaoDialog".

[04:04] A gente pode apagar todas essa chamada e chamar o chama, só que agora, ao invés de chamar esse cara que vem com uma transação e o transação delegate, a gente chama o que vem com o tipo e com o transação delegate. Então a gente chama dessa maneira, enviando agora tipo e o transação delegate, só um detalhe que é bem bacana da gente estar observando, nesse tipo de abordagem, é justamente o seguinte.

[04:25] Reparem que a gente está fazendo essa chamada do chama, duas vezes, e parece um pouquinho ambíguo o que tá acontecendo aqui, mas não dá nem para entender, logo de cara, que isso daqui trata-se de uma chamada lá da classe mãe, da superclass. Portanto, para deixar bem claro que, essa chamada aqui, está sendo feita pela nossa superclass, a gente vai colocar o "super" aqui, justamente porque a gente tem a chama aqui também e, as vezes, pode dar uma confusão, só para deixar bem claro, que isso daqui, está relacionado com a superclass.

[04:53] A gente conseguiu resolver essa parte da configuração e, agora, reparem que a gente está com um problema de compilação aqui nos campos, por que que está acontecendo isso? Novamente, é aquele caso de estar privada. Só que nesse momento, a gente precisa desses caras, porque a gente precisa deles? Porque o processo de inicialização da "AlteraTransacaoDialog ", exige que a gente tenha esses campos para poder colocar os valores nele, portanto, deixá-los privados, não é um aspecto que a gente precisa nesse momento.

[05:19] Não é um aspecto que a gente necessita, a gente precisa que ele seja acessível pelas suas classes filhas, mas agora entra aquele caso, que se a gente chegar aqui no nosso "FormularioTransacaoDialog", e por exemplo, pegar o campo valor, e deixa-lo sem o private, dessa forma, ele vai lá e compila, ele compila na "AlteraTransacaoDialog". Só que quando a gente deixa dessa maneira, como eu comentei com vocês, é a mesma coisa de a gente chegar, e deixar como público.

[05:43] Portanto, mesmo o "FormularioTransacaoDialog", mesmo suas filhas, como qualquer classe do projeto, vai ter a capacidade de pegar esse campo aqui, e manipular da maneira que quiser. Em outras palavras, o que eu quero falar é o seguinte, por mais a gente tenha que ter acesso, dentro da "AlteraTransacaoDialog".

[06:00] Já que a gente tá lidando com essa parte herança, a gente tem um modificador de acesso específico, para poder dar o nível de acesso é razoável, tanto para a classe que está implementando, que é a "FormularioTransacaoActivity", quanto para as suas filhas, e fazer com que as classes de fora não tenham acesso. E qual é esse modificador de acesso? É o modificador de acesso protegido, ou seja, o protected, da mesma maneira que a gente conhece no Java.

[06:20] A gente pode chegar aqui e colocar como "protected", agora, as nossas filhas têm acesso a esse membro, só que apenas elas e a classe que implementou que tem acesso, as classes de fora não tem, quem tiver de fora não vai ter acesso. Claro, quem estiver no mesmo pacote ainda vai ter acesso, só que é uma restrição bem maior, porque quem tá vindo aqui, dentro do mesmo pacote, são só os nossos Dialogs, a gente não tá colocando outras classes, que são perigosas de ter esse tipo de acesso.

[06:45] É o mesmo caso que a gente tem no Java, agora que a gente conseguiu colocar esse modificador de acesso, vamos colocar também para os outros, tanto para o campo data, "campoData" aqui, pelo "campoCategoria", "protected". Agora, todos esses campos, são acessíveis pelas filhas, a gente consegue manipular, dentro das nossas filhas aqui. Reparem que agora a gente conseguiu compilar nessas partes dos campos, só que agora vem essa parte da "categoriasPor", porque também está privada.

[07:13] A gente está precisando reutiliza-la, se a gente perceber, a gente pode deixar ela, também, acessível para suas filhas, porque ela não modifica o comportamento de acordo com a ação que a gente tem no nosso Dialog, seja alteração, ou seja adição. Logo, a gente pode chegar aqui, no nosso "FormularioTransacaoActivity " e lá no nosso "categoriasPor", vamos pegar "categoriasPor", a gente pode deixar como "protected", e o "AlteraTransacaoDialog", já tem a capacidade usar de membro da superclasse sem nenhum problema.

[07:45] Agora que a gente tá conseguindo compilar, vamos fazer um teste para ver qual é o comportamento que tá acontecendo, quando a gente conseguiu colocar essa herança? Vamos lá, Alt+Shift+F10, veja que o Android Studio conseguiu executar. Vamos ver como ficou nossa App, a princípio está executando normalmente. Deixa eu adicionar uma transação, uma transação de R$ 100, ele adicionou sem nenhum problema, agora vamos tentar alterar.

[08:06] Quando a gente tenta alterar, ele abre o dialog novamente, só que agora tem algumas peculiaridades, reparem que o título ainda está como Adiciona receita, só que o campo está funcionando, campo funcionando, que é o de valor, o campo de data também, a princípio tá funcionando, e a categoria está vindo as da receita sem nenhum problema. Aqui em baixo o botão está adicionar, por mais que a gente abriu o diálogo aqui, para poder alterar, algumas informações de adicionar estão sendo mantidas, a gente vai resolver isso logo.

[08:33] Só vamos ver se o comportamento de alteração ainda está sendo mantido, vamos clicar em adicionar, considerando que agora a gente quer alterar, e vamos ver o que acontece. Ele tá conseguindo alterar, a única peculiaridade que a gente tem que resolver agora, é justamente essa informação de manter essa parte de adicionar, aliás, não manter essa possibilidade de só adicionar e, também, ser flexível o suficiente para, no momento que for alterar, colocar o título de alterar, seja no título dialog, seja no botão positivo.

[09:00] Quando for adicionar, deixar adicionar, sem problemas. A gente precisa, agora, só apenas colocar essa implementação, e veremos isso a seguir. Até já!