10

Cálculo: delegando responsabilidade

Agora que somamos todas as transações de receita e despesa, precisamos fazer a diferença para apresentar o total.

Entretanto, antes de começarmos com essa implementação, repara que a classe `ResumoView`, além de colocar as informações nos componentes, está também calculando.

Em outras palavras, vamos refatorar o código para que essa responsabilidade de calcular tanto a receita como a despesa seja delegada para uma classe específica.

Delegando o código para a classe Resumo

Para isso, crie primeiro a classe `Resumo` no pacote **model**.

Dentro da classe `Resumo`, crie a função `receita()` e envie todo o código que calcula a receita dentro desta função.

Da mesma maneira que foi feito na função `receita()`, crie também a função `despesa()` e adicione todo o código que calcula a despesa dentro dela.

Ajustando a referência da lista de transações

Repara que apenas copiando e colando o código que calcula as transações não é o suficiente, pois a referência `transacoes` não é acessível na classe `Resumo`.

Sendo assim, receba a lista de transações via construtor primário na classe `Resumo` e a transforme numa property para que seja acessível por todos os membros.

Modificando o código da ResumoView

Após a implementação da classe `Resumo`, no lugar dos códigos que calculavam tanto a receita, como também a despesa, modifique para que agora chame a função `receita()` e a função `despesa()`.

> **Dica:** Lembre-se de transformar o objeto da classe `Resumo` em uma property para que todos os membros acessem o mesmo objeto.

Por fim, execute a App e veja se o cálculo das transações está funcionando.