06

Refatorando o código

Por mais que tenhamos implementado o aspecto visual do resumo da mesma maneira como está na App base, ainda existem alguns pontos do código que pode melhorar.

Pensando justamente neste detalhe, vamos começar com o processo de refatoração para melhorar o código.

Extraindo properties para representar as cores

Uma das técnicas que podemos aplicar é a extração das properties para representar a cor tanto de receita como de despesa.

Para isso utilize o recurso de extração do Android Studio com o atalho **Ctrl + Alt + F** ou no Mac **CMD + Alt + F**.

> Note que nessa parte da refatoração o AS substitui todo o código para que agora utilize a property

Removendo a duplicidade de chamada do objeto

Dentro das funções do `ResumoView` repare que a referência dos componentes estão sendo chamadas mais de uma vez, sendo assim, utilize a feature `with()` para chamar os membros do objeto (properties ou funções) sem a necessidade de chamar o objeto todas as vezes.

Extraindo a função para pegar a cor pelo valor

O próximo passo da refatoração é extrair a função que vai devolver a cor de acordo com a variável `total` dentro da função `adicionaTotal()`.

Para isso, faça com que a inserção da cor no componente `resumo_card_total` fique fora do escopo do `if`, então, faça com que o `if` se torne um **if expression** que vai devolver a variável `cor`.

Em seguida, extraia a função `corPor()` que vai receber via parâmetro a variável `total` e vai devolver a cor.

Convertendo o if expression

Depois da extração, transforme o **if expression** em um early return para deixar o código mais simples.

Modificando a chamada do `compareTo()`

Atualmente utilizamos a comparação das variáveis do tipo `BigDecimal` por meio da função `compareTo()`, porém, vimos que somos capazes de utilizar os operadores lógicos que o Kotlin já converte em funções de comparação de objetos. Portanto, utilize o operador `>=` no lugar do `compareTo()`.

Por fim, execute a App e veja se é mantido o mesmo aspecto visual.