

Sobrecarga com Named Parameter

Vimos que uma proposta para evitar o problema de compatibilidade durante as diferentes formas de construção de objetos, é por meio de sobrecarga de construtores.

No Kotlin podemos utilizar o construtor secundário pra isso, ou seja, podemos manter o construtor primário da mesma maneira como ele sempre foi e aplicar o construtor secundário:

```
class Transacao(val valor: BigDecimal,
                val categoria: String = "Indefinida",
                val tipo: Tipo,
                val data: Calendar = Calendar.getInstance()) {
    constructor(valor: BigDecimal, tipo: Tipo) : this(valor, "Indefinida", tipo)
}
```

Entretanto, observe que quando fazemos esse tipo de chamada, ainda sim precisamos chamar o construtor primário enviando um valor padrão da property `categoria` ...

Isso significa que, se criarmos uma outra variação de sobrecarga com o construtor secundário, teremos que fazer novamente a mesma chamada para o construtor primário, deixando cada vez o nosso código difícil ler e manter.

Utilizando Named Parameter

Pensando justamente em evitar esse tipo de situação, o Kotlin nos fornece um recurso conhecido como Named Parameter que vem com a proposta de permitir uma sobrecarga de construtores e funções de tal forma que não seja necessário manter a mesma ordem das sobrecargas.

Para isso, basta apenas chamar a label do parâmetro, adicionar um operador de igual e enviar o valor, por exemplo, para enviar o valor da property `valor` em uma instância da classe `Transacao` utilizando o Named Parameter fica da seguinte maneira:

```
Transacao(valor = BigDecimal(100.0))
```

Com essa abordagem, podemos realizar a sobrecarga de construtores sem precisar declarar os construtores secundários. Em outras palavras, caso tenha implementado os construtores secundários remova-os e faça a instância das transações utilizando apenas o recurso do Named Parameter.

Lembrando que o Named Parameter não exige ordem, porém, é necessário enviar todos os parâmetros que não possuam valores padrões, como é caso das properties `valor` e `tipo`.

Depois de finalizar a implementação do Named Parameter, rode a App e veja se funciona como o esperado.