08

Para saber mais: constantes globais

Vimos que properties globais têm a característica de manter um valor desde o começo do programa até o final sem vínculo a objetos.

No caso de **mutáveis** (`var`), devem ser evitadas, pois podem apresentar inconsistência por permitir a edição em vários pontos do código, e vimos esse problema acontecendo no contador. Porém, não temos esse problema com properties globais **imutáveis** (`val`).

Portanto, se criar uma property `val` que:

- não é declarada via construtor primário;
- não tem o objetivo de manter exclusividade a um objeto (como é o caso de um endereço que pode ser `val` mas cada um precisa ter o seu);
- e é um valor fixo que não muda, como é o caso de um primitivo ou String;

Considere o uso da property global. Um exemplo seria o seguinte:

```
class Produto(
    private val nome: String,
    private val valor: Double
) {

    val taxaAdicional = 0.1

    val preco: Double get() = valor + (valor * taxaAdicional)

}
```

Note que o produto nesse exemplo vai ter a mesma taxa adicional, e ela não muda! Portanto, não precisamos fazer um código que crie uma taxa para cada objeto, ela pode ser uma property global ou até mesmo um object declarations (companion):

```
val taxaAdicional = 0.1

class Produto(
    private val nome: String,
    private val valor: Double
) {

//    companion object {
//        val taxaAdicional = 0.1
//    }

    val preco: Double get() = valor + (valor * taxaAdicional)

}
```

Em casos como esses, a própria documentação sugere o uso de constantes em tempo de compilação (https://kotlinlang.org/docs/reference/properties.html#compile-time-constants) (Compile-Time Constants), que são properties imutáveis que não mudam o seu estado, como é o caso de tipos primitivos ou `String` s, e que não tenham um getter personalizado:

```
const val taxaAdicional = 0.1

class Produto(
    private val nome: String,
    private val valor: Double
) {

    val preco: Double get() = valor + (valor * taxaAdicional)

//    companion object {
//        const val taxaAdicional = 0.1
//    }

}
```