09

Para saber mais - Evoluindo o Resource

Durante a aula criamos a classe resource para que fosse possível devolver um objeto via LiveData contendo o dado esperado e um possível erro.

A nossa abordagem inicial é bastante simples, ou seja, é muito comum o uso de variações mais completas que contenham, por exemplo, o estado do recurso.

Considerando um exemplo de código, poderíamos criar um recurso com a seguinte estrutura:

```
open class Resource<T>(
    val dado: T?,
    val erro: String? = null
)

class SucessoResource<T>(dado: T) : Resource<T>(dado)

class FalhaResourc<T>(erro: String) : Resource<T>(dado = null, erro = erro)
```

Então, nas situações de sucesso criamos o `SucessoResource` e na falha o `FalhaResource`, então, na atualização do LiveData, somos capazes de identificar explicitamente as notificações de sucesso ou falha, usando o *when expression*:

```
viewModel.buscaTodos().observe(this, Observer { resource ->
    resource.dado?.let { adapter.atualiza(it) }
    when(resource) {
        is SucessoResource -> {
            //executa procedimento de sucesso
        }
        is FalhaResource -> {
            //executa procedimento de falha
            mostraErro(MENSAGEM_FALHA_CARREGAR_NOTICIAS)
        }
    }
})
```

É válido ressaltar que essa é uma possível versão para o Resource, ou seja, nada impede de estender mais e colocar mais status ou comportamentos esperados para essa abordagem.