Aula 27
Arquitetura prática

DigitalHouse >
Coding School

MVVM

Model

Sempre que você pensar em manipulação de dados, pense em model. Ele é responsável pela leitura e escrita de dados, e também de suas validações.

View

A View liga-se a variáveis Observable e ações expostas pelo ViewModel de forma flexível.

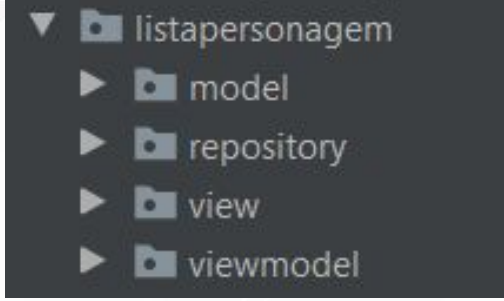
ViewModel

A ViewModel é responsável por apresentar funções, métodos e comandos para manter o estado da View, operar a Model e ativar os eventos na View.

Estrutura de pacotes

Feature by package - é criado um pacote para cada funcionalidade da aplicação

Estrutura de pacotes



Model - Nossos modelos de dados. Ex.: Data Class

View - Tudo referente a Interface. Ex.: Activity, Adapter, Fragment, etc

ViewModel - Comunicação com o Repository, regras de negócio, etc

Repository - Chamadas de API ou banco de dados, etc

Dependências

```
implementation 'androidx.lifecycle:lifecycle-extensions:2.1.0'
implementation 'androidx.lifecycle:lifecycle-viewmodel-ktx:2.1.0'
```

Criar uma ViewModel

Extende de ViewModel() e pode receber um repository no construtor

```
class MeuViewModel(
    private val repository: MeuRepository
): ViewModel()
```

Utilizar o ViewModel na View

Obter o ViewModel no onCreate da Activity

```
viewModel = ViewModelProvider(
    this,
    MeuViewModel.MeuViewModelFactory(MeuRepository(this))
).get(MeuViewModel::class.java)
```

Sobre o MutableLiveData

Permite que observemos alterações de valores

```
MutableLiveData<TIPO>()
```

Utilizar o ViewModel na View

Usar o viewModel.minhaVariavel.observe para monitorar possível atualização.

```
viewModel.minhaVariavel.observe(this, Observer {
    // Código aqui
})
```

Utilizar o ViewModel na View

Implementar a lógica na ViewModel e chamá-la depois na View

```
viewModel.obterFilmes()
```

Exercício