Documentação de Projeto

Desafio Mobile Android

Lucas Menezes Rech (https://github.com/lmrech/desafio-mobile-android)

1. Especificações Técnicas

SDK (Versão Mínima)	21
SDK (Versão Alvo)	33
Package	com.desafio.android
Design de Arquitetura	MVVM
Bibliotecas	- Room - Dagger Hilt - Retrofit - Accompanist - Glide - Compose - Mockito

2. Descrição

Este projeto foi estruturado com o padrão de arquitetura MVVM, portanto temos a separação entre classes das camadas de **Interface de Usuário**, **ViewModel**, **Use Cases** e **Data**.

Interface de Usuário: Neste projeto, está sendo feito o uso da tecnologia Jetpack Compose, portanto os arquivos de interface estão em métodos com anotação @Composable, como por exemplo HomeScreen.kt e HomeScreenContent.kt.

ViewModel: Para cada tela, existe uma ViewModel única. Como neste aplicativo há somente uma tela, a ViewModel é HomeViewModel, e possui a lógica de tratamento dos dados. A ViewModel é a única classe responsável pela alteração dos dados da interface do usuário, e as informações ficam salvas em MutableState, na classe HomeScreenState.

Use Cases: Operações que envolvam carregamento de dados, seja qualquer a origem, são feitas através de use cases. Neste projeto existe apenas um: GetMarvelCharactersCase. Os Use Cases possuem sua resposta sempre atrelada a uma Sealed Class, que irá representar se o carregamento foi concluído com sucesso ou fracasso, além de sinalizar quando um carregamento está em andamento.

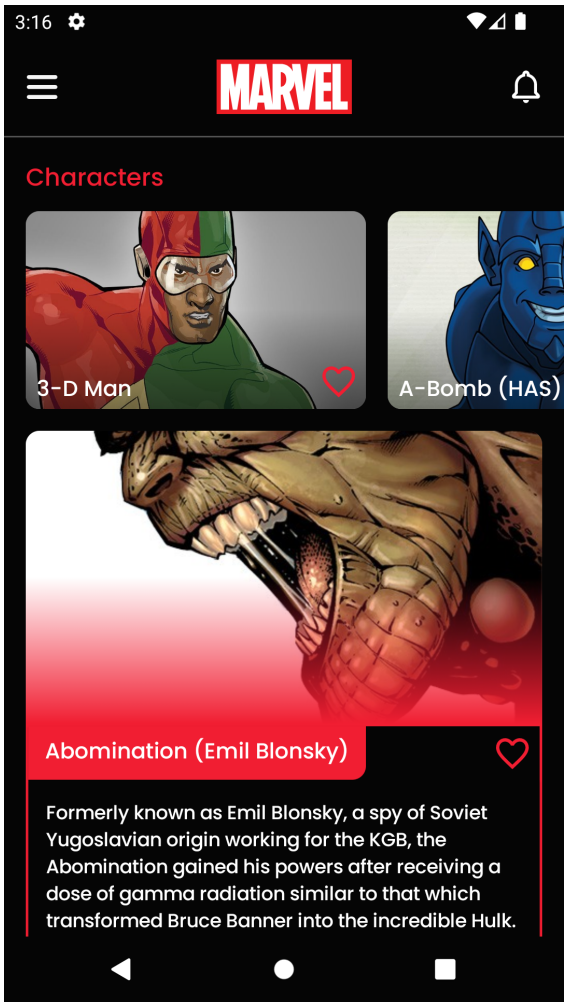
Data: O projeto possui apenas um repositório, que realiza o acesso aos dados. Os dados podem ser carregados através da API disponibilizada, ou pelo banco de dados interno do aplicativo. Ao carregar dados pela API, os mesmos serão copiados para o banco de dados interno, assim na próxima vez que forem solicitados, não será necessário realizar a chamada externa.

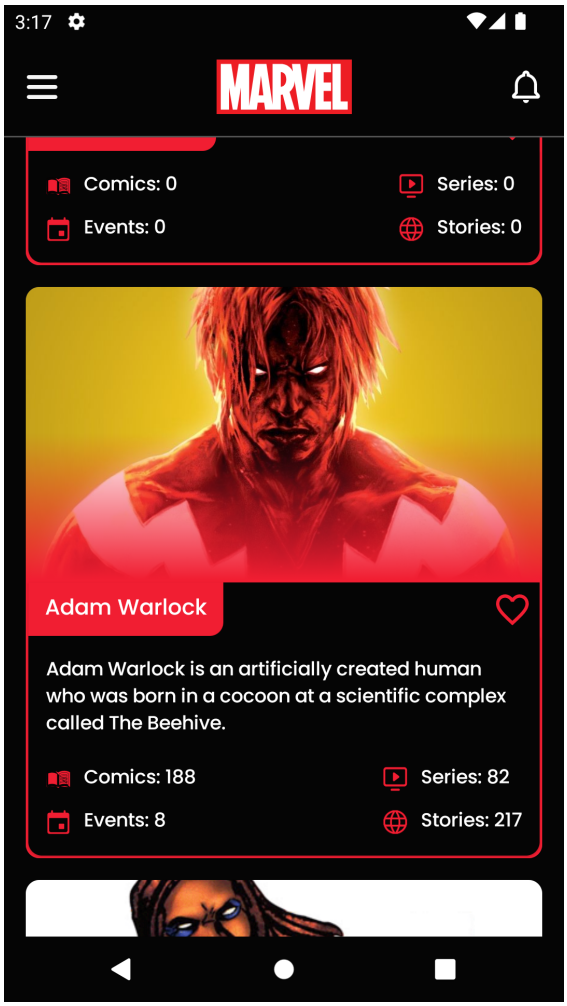
3. Testes Unitários

O aplicativo possui uma classe de teste chamada HomeViewModelTest. Como são poucas funcionalidades implementadas, a classe possui apenas 4 testes unitários integrados, que são:

- Sucesso ao carregar a lista de personagens.
- Falha ao carregar a lista de personagens.
- Reiniciar a lista de personagens carregados.
- Visualizar o último item da lista de personagens.

4. Demonstração





5. Observações

    - Existem botões na interface que estão sem funcionalidade, apenas existem para fins demonstrativos (Ex.: botão para marcar como favorito, menu lateral, etc.)
    - Novos elementos serão carregados quando o usuário atingir o fim da lista.
    - O projeto demorou cerca de 5 horas no total para ser desenvolvido.

6. Melhorias

    - Poderiam haver mais testes, e mais funcionalidades.
    - **A chave privada da API está no Gradle, pois ela é necessária para gerar a hash na chamada do método para obter personagens. Essa chave não poderia estar no aplicativo caso ele fosse público, e sim deveria haver um servidor back-end para apenas informar a hash ao aplicativo na hora da chamada, ou o próprio back-end fazê-la e retornar a resposta.**