Bestseller Book Watch Application Overview

Aaron Quaday | AAQ-AND-DEV April 10. 2021

Android UI/UX

Navigable interface: I have 4 screens (watched category chooser, BestsellersList display, Book details, and an About page) These are all accessed via NavigationController. I use argument field in nav_graph to pass the book to the details screen

Display: Most layouts that are not super simple are constraintLayouts, with ids and appropriate constraints. Resources are stored in the res directory. My data collections (categories and book lists) are displayed using ViewHolder Pattern and RecyclerViews

Animation: My Book Details page is animated upon creation via MotionLayout. These MotionLayout behaviors are defined in a *MotionScene* using one or more *Transition* nodes and *ConstraintSet* blocks.

Local and Network Data

API: My application connects to the New York Times Books Api to provide data, accessed via Retrofit. This is converted to models via MoshiConverterFactory. This work is done on background threads

Loading Network Resources: Picasso is used to load images of book covers, with appropriate placeholder and error images. Picasso handles the background threading.

Local Storage: I use Room to store all Categories and Books. I use SharedPreferences to track the selectedCategories that are tracked by the user (due to api network limits, this is limited to 7 categories). Data is stored in the db for at most 20 hrs (it is refreshed if more than 20 hrs has passed since access to not violate Terms of Service, which preclude storing data for longer than 1 day.) These storage operations are all performed on Dispatcher.IO threads. Data is structured with appropriate data types and scope as required.

Android system and hardware integration

MVVM: My app follows the MVVM pattern, with one SharedViewModel (since Categories are shared between two screens). Observers in views access viewModel properties. Repository is the source of data.

Hardware/System events: The application handles system events. Notification properly utilizes Intent to open app when triggered.

System Hardware Utilization: A FirebaseMessagingService is implemented to notify users when the Bestseller Lists have been Updated on the NYT Books API server. This is scheduled weekly, but if you email aaqjavadevmentor@gmail.com, I can generate a test notification for you when evaluating the app.