Zaawansowane biblioteki Android

DataStore, LiveData.
Coroutines i Flow. Retrofit.

mgr inż. Stanisław Lota

Jetpack DataStore

Jetpack DataStore to rozwiązanie do przechowywania danych, które umożliwia przechowywanie par klucz-wartość lub wpisanych obiektów za pomocą buforów protokołu .

DataStore według oficjalnego bloga Android ma na celu zastąpienie SharedPreferences poznanych na poprzednim wykładzie, które ma kilka głównych wad.

Jetpack DataStore

Programiści, którzy używają SharedPreferences do przechowywania danych,powinni rozważyć migrację do DataStore.

DataStore wykorzystuje technologie **Coroutines i Flow.** Przechowuje dane w sposób asynchroniczny, spójnie i transakcyjnie.

Flow czyli asynchroniczna obsługa strumienia danych przypomina RXJavę i została stworzona przez Jetbrains.

W czasie rzeczywistym dane mogą ulec zmianie w ciągu kilku sekund i zostać zaktualizowane w aplikacji na każdym smartfonie.

Jetpack DataStore

DataStore udostępnia dwie różne implementacje: Preferences DataStore i Proto DataStore.

Preferences DataStore przechowuje i uzyskuje dostęp do danych za pomocą kluczy. Ta implementacja nie wymaga wstępnie zdefiniowanego schematu i nie zapewnia bezpieczeństwa typu.

Proto DataStore przechowuje dane jako instancje niestandardowego typu danych. Ta implementacja wymaga zdefiniowania schematu przy użyciu buforów protokołu , ale zapewnia bezpieczeństwo typów.

Dwie implementacje DataStore

Jetpack udostępnia dwie implementacje DataStore:

Preferencje DataStore — przechowuje dane w parach klucz-wartość, takich jak, SharedPreferencesale nie zapewnia żadnego bezpieczeństwa typu.

Proto DataStore — przechowuje dane jako obiekty niestandardowe. Zapewnia bezpieczeństwo typów po wyjęciu z pudełka, ale wymaga określenia schematu za pomocą buforów protokołów.

Jetpack DataStore

DataStore udostępnia asynchroniczne API do przechowywania i odczytywania danych, natomiast **SharedPreferences** zapewnia asynchroniczne API **tylko podczas odczytywania zmienionych wartości za pomocą detektorów.**

DataStore można bezpiecznie wywołać w wątku interfejsu użytkownika.

Proto DataStore zapewnia lepsze bezpieczeństwo.

LiveData

LiveData jest częścią architektury Androida i tzw. obserwowalnym posiadaczem danych. Służy do obserwacji zmian w danych i aktualizowania widoku.

Użytkownicy aplikacji oczekują, że składniki interfejsu użytkownika będą reagować interakcyjnie na zmiany i aktualizacje powiązanych danych, aby zapewnić bardziej interaktywne i dynamiczne środowisko użytkowania jak np. przy popularnych społecznościowych aplikacjach mobilnych.

LiveData

LiveData jako komponent architektury Androida jest trochę podobny do RxJava, z wyjątkiem tego, że LiveData jest świadomy istnienia cyklu życia aktywności. LiveData jest zgodny ze wzorcem Obserwator. Bibliotekę tą najlepiej łączyć z ViewModel.

RxJava była jedną z przełomowych bibliotek w Javie. Spopularyzowana mocno przez rozwiązania Netflix, urzekła wiele osób swoim podejściem do reaktywnego programowania.

LiveData

Programowanie reaktywne to asynchroniczny paradygmat programowania polegający na przetwarzaniu strumieni danych i propagowaniu ich zmian. Podejście to nawiązuje do wzorca projektowego Observer i przyjmuje się koncepcja programowania reaktywnego wywodzi się właśnie z niego.

Coroutines i Flow

Coroutines (korutyny, współprogramy) znacznie ułatwiają realizację zadań współbieżnych poprzez zmianę stylu pisania kodu asynchronicznego w sposób sekwencyjny.

Dzięki takiemu podejściu kod jest bardziej czytelny i zrozumiały, a zarządzanie zadaniami staję się łatwiejsze. Ponadto jeden wątek potrafi obsługiwać jednocześnie wiele coroutines co przekłada się na znaczny wzrost wydajności.

Coroutines

Tradycyjny model tworzenia kodu asynchronicznego opartego o metody zwrotne (Callback) narażony jest na występowanie różnych trudności. Wywołanie zwrotne jest funkcją przekazywaną do innej funkcji jako argument, który jest następnie wywoływany wewnątrz funkcji zewnętrznej w celu wykonania pewnego rodzaju procedury lub akcji.

Współprogram cechuje się posiadaniem ciągu instrukcji do wykonania i ponadto możliwością zawieszania wykonywania jednego współprogramu A i przenoszenia wykonywania do innego współprogramu B.

Flow

Jetbrains zbudował Kotlin Flow w oparciu o Coroutines. Coroutines świetnie nadają się do jednorazowego pobierania lub innych operacji CRUD. Flow wykorzystuje się do obsługi strumieni i można przekształcać dane w złożony wielowątkowy sposób.

Flow umożliwia:

- Zbieranie danych i strumienie.
- Synchroniczne i asynchroniczne wywołania interfejsu API.
- Gorące i zimne strumienie danych.
- Obsługa wyjątków podczas przetwarzania przepływu.

Retrofit

Retrofit to biblioteka do obsługi zapytań HTTP w środowisku aplikacji Java oraz Kotlin.

Umożliwia w łatwy sposób pobieranie i przesyłanie danych w formie obiektów lub schemacie JSON za pośrednictwem usługi internetowej opartej na REST. Ponadto pozwala na wykonywanie zapytań w sposób synchroniczny i asynchroniczny z uwzględnieniem uwierzytelniania i rejestrowania stanu operacji. Retrofit umożliwia współpracę z RxJava.

Zadania do wykonania

https://developer.android.com/codelabs/kotlin-android-training-room-database#0

https://developer.android.com/codelabs/kotlin-android-training-coroutines-and-room#0

https://developer.android.com/codelabs/kotlin-android-training-quality-and-states#0

https://developer.android.com/codelabs/kotlin-android-training-internet-data#0

https://developer.android.com/codelabs/kotlin-android-training-internet-images#0

https://developer.android.com/codelabs/kotlin-android-training-internet-filtering#0