Compte Rendu - TEA Séquence 3

José Vitor MARTINS MAKIYAMA
Danilo ORTIZ RODRIGUES

Introduction

Dans ce projet séquence, nous allons améliorer notre application Android, développée dans les dernières séquences, en ajoutant un cache avec une base SQLite avec l'ORM Room, nous allons aussi gérer le mode offline de notre application et comment il mètrera à jour les données lorsque la connexion est rétablie.

Implémentation

Base SQLite (Room)

Pour l'implémentation de la base de données, nous avons utilisé des classes spécifiées dans la première séquence pour définir les tables: Le Pseudo, le Liste et l'Item de chaque liste. Pour accéder la base et modifier les données, Room a besoin d'une interface DAO (*Data Access Object*) et des annotations pour générer automatiquement le "*boilerplate code*" pour chaque classe et les méthodes de chaque SQLite *query* que nous avons besoin. Pour chaque classe, nous avons créé des méthodes pour l'ajouter, la supprimer et la récupérer de la base, en ce qui concerne les Item, une fonction permet de les cocher.

Après, pour rendre notre base Room accessible à l'application, nous avons créé une classe DataProvider qui est responsable de tous les accès aux données, que ce soit à partir de la base de données ou par l'intermédiaire de l'API, il y a des méthodes que nos Activities utilisent pour avoir accès à ces données. Toutes ces fonctions ont le *keyword* "suspend", ça indique au Kotlin, et à la bibliothèque Room, que nous voulons utiliser ces fonctions avec les Coroutines, sans bloquer la Main Thread, donnant le boulot à un thread au background.

Pour utiliser les Coroutines, nous devons créer une portée de coroutine, en donnant comme paramètre le Dispatcher MAIN, vers lequel le résultat de la coroutine sera renvoyé, et les fonctions DataProvider utilisent un contexte Dispatchers IO ou Default, indiquant que le travail doit être effectué sans bloquer le thread de l'UI.

Retrofit pour accéder aux API REST

La bibliothèque Retrofit nous permet d'implémenter les requêtes Rest d'une manière simple : en utilisant des annotations avec les informations de la requête, déclarées dans une interface Apiservice qui est implémentée par un objet Retrofit. Cet objet doit être créé avec un interpréteur qui peut lire la réponse, dans notre cas en JSON, du serveur.

Mode Offline

Dans le cas du mode hors ligne, certains conflits peuvent survenir entre les données de l'API et celles de la base de données locale (cache). Comme l'utilisateur a la possibilité d'ajouter des Lists et des Items, en plus de les marquer ou non, nous avons décidé de donner la priorité à la base de données au moment de la synchronisation : de cette façon,

les données locales sont toujours considérées comme étant plus à jour. Dans chaque Activity, nous avons une boucle qui surveille la connexion Internet, lorsqu'il n'y a pas de connexion, elle modifie une variable indiquant à l'Activity d'utiliser les données mises en cache, lorsque la connexion est rétablie, elle appelle la fonction de synchronisation des différences du DataProvider.

Perspectives

Nous avons encore de points à améliorer : il y a encore quelques bugs dans notre application ou dans quelques cas très spécifiques (elle peut crasher et fermer abruptement ou ne pas faire ce que nous voulons), alors nous devrons chercher leur source et les corriger. D'autres points que nous pouvons améliorer :

- Développer l'application en plusieurs langues
- Gérer des IHM différentes suivant les modes portrait et paysage
- Des propositions de complétion pour le pseudo s'affichent lors de la saisie, à partir de l'historique des pseudos saisis auparavant.

Conclusion

Enfin, nous comprenons les difficultés liées à la création d'une application Android, ainsi que l'apprentissage des meilleures pratiques de manière concrète. Nous avons appris à gérer la persistance des données à l'aide d'une bibliothèque ORM et Room et à accéder à une API Rest avec Retrofit. Ainsi que les meilleures pratiques d'unification et d'abstraction de l'accès aux données à l'aide d'une classe DataProvider.

Bibliographie

- Google's Android Developer
  https://developer.android.com/