Alexandre Pizigo

Rapport PMR

Le but de l'application est d'avoir des todolists modifiables, propres à chaque utilisateur. Pour cela :

- une page de login où l'on rentre son pseudo

- une page qui affiche les Todo Lists propres aux pseudo

- une page par Todo List qui permet de la modifier

- une page de Préférences où l'on peut voir les pseudos et leur todolists associés

La grande difficulté que je n'ai toujours pas résolu pour moi est l'utilisation de la libraire gson pour sauvegarder un JsonArray des utilisateurs dans SharedPreferences. J'aurais préféré partir sur l'écriture d'un fichier par utilisateur (nommé alors après l'utilisateur ou un identifiant) mais j'ai choisi de suivre le fil du TD.

La structure envisagée était la suivante :

[

{'pseudo' :'…',

'listestodo' : ['liste1' : ['item11', 'item22'],

'liste2' :['item21', 'item22'] },

{'pseudo2' :'…',

'listestodo' : ['liste1' : ['item11', 'item22'],

'liste2' :['item21', 'item22'] },

…]

La difficulté majeure pour moi était la persistance des données. Par exemple, mettre un pseudo déjà existant dans l'écran d'accueil ne devait pas créer un Objet Json vide avec des listesTodo vides qui allaient remplacer l'objet asocié au pseudo déjà existant. Pour cela j'envisagais une vérification simple du style « prefs.toString().contains('\'pseudo:\''+pseudo+'\'') où prefs est le résultat de getDefaultSharedPreferences converti en JsonArray grâce à Gson. Cependant je me

retrouvais bloqué par de nombreuses exceptions, plus particulièrement la JsonSyntaxException qui annonçait s'attendre à un JsonArray et recevoir un JsonNull.

D'autres difficultés similaires sur la gestion des exception null, où je devais faire attention à ne pas rajouter des items à des ListesToDo null (càd non instanciées) font que la persistance des données n'est toujours pas fonctionnelle.

Pour ce qui est de l'utilisation de RecyclerView pour faire apparaître les listes, la documentation android a été d'une très grande utilité. J'ai codé en un premier temps un dataset en dur pour fournir des données à mon adapteur, puis ensuite passait les données que je récupérais de SharedPreferences pour mon adapteur. J'ai aussi rajouté des OnClickListener à chaque TextView de mes listes en passant par l'adapteur et son holder (ça avait l'air plus simple que « gesture touch »).

Je remarque une redondance de code assez forte entre ChoixListActivity et ShowListActivity : même layout, et mêmes méthodes pour sauvegarder les données. Il y a donc peut-être intérêt à créer une classe plus générale que l'on pourrait étendre pour ChoixListActivity et ShowListActivity.

Je pense aussi qu'il est plus simple (plus clair) de sauvegarder des données utilisateur dans des fichiers séparés plutôt que dans un tableau, en tout cas il serait préférable de procéder de cette manière ou même d'utiliser SQLite si le nombre d'utilisateurs est grand.

Bibliographie : ressources utilisées

https://stackoverflow.com/questions/1598119/is-there-an-easy-way-to-add-a-border-to-the-top-and-bottom-of-an-android-view

https://www.bignerdranch.com/blog/understanding-androids-layoutinflater-inflate/

https://developer.android.com/guide/

https://sites.google.com/site/gson/gson-user-guide#TOC-Primitives-Examples