Alban RAHIER

Compte-rendu application Android ToDoListe

Introduction :

L'application ToDoListe permet à plusieurs utilisateurs de stocker des listes personnelles de tâches à effectuer. Elle doit notamment utiliser des données persistantes pour pouvoir garder en mémoire les Listes ainsi que le dernier pseudo utilisé lors de l'arrêt de l'activité. Mon choix s'est porté sur les préférences de l'activité pour stocker le pseudo de l'utilisateur et sur des fichiers .json pour stocker les objets ProfilToDo. Il existe un fichier par utilisateur.

Analyse :

Chaque sous-partie est à lire en complément avec le code commenté.

Classes Objet :

J'ai réutilisé le diagramme de classe de l'énoncé. J'ai juste ajouté à la correction proposée par Mme Le Glaz la méthode rechercherListe() dans la classe ProfilListeToDo et la méthode uncheckItem() (inverse de validerItem) dans la classe ListeToDo.

```
public int rechercherListe(String s)
{
    int retour = -1; //la méthode renvoie -1 dans le cas où la Liste n'a pas été trouvée
    for (int i=0; i < this.mesListeToDo.size() ;i++)
    {
        Log.i( tag: "PMR", msg: "on compare " + this.mesListeToDo.get(i).getTitreListeToDo() + "et " + s);
        if (this.mesListeToDo.get(i).getTitreListeToDo().equals(s)){
            Log.i( tag: "PMR", msg: "on a trouvé");
            retour=i;
            i=this.mesListeToDo.size();
        }
    }
    return retour;
}
```

MainActivity :

L'activité de démarrage propose de choisisr quel profil va-t-on utiliser. Un profil est affiché lors de l'ouverture de l'activité ; c'est le dernier utilisé. Si l'utilisateur rentre un profil qui n'est pas en mémoire, un profil vide est créé.

J'ai quasiment tout gardé de la correction lors du TP. J'ai ajouté la méthode qui permet de sauvegarder le profil sous la forme d'un pseudo.

PreferenceActivity :

J'ai tout gardé de la correction lors du TP. On se contente d'afficher le profil en cours d'utilisation. L'utilisateur peut changer ce profil. J'ai essayé d'afficher la liste des profils existants mais je n'ai pas réussi à implémenter le recyclerview au sein de l'activité de préférences.

Alban RAHIER

ChoixListActivity et RecyclerViewAdapter1 :

Pour créer une nouvelle liste, plutôt que d'utiliser le modèle proposé par l'énoncé, j'ai préféré utiliser un FloatingActionButton. L'avantage du bouton flottant est qu'il suit le scrolling de l'utilisateur. De plus, c'est la méthode la plus parlante visuellement et la plus utilisée sous développement Android. Il faut la coupler avec un AlertDialog qui demande le nom de la nouvelle liste. Lors de la création d'une nouvelle liste, on sauvegarde le fichier json modifié.

Pour afficher les changements, il est nécessaire de rafraîchir l'activité. Le rafraîchissement de l'activité se fait en la relançant. Je trouve que cette méthode n'est ni fluide ni adaptée, mais je n'ai pas trouvé d'autre manière de faire.

Pour implémenter le RecyclerView, je crois avoir utilisé la même méthode que M. BOUKADIR. On déclare une class Adapter qui va créer les views à afficher en liste (elle fait appel au fichier R.layout.layout_listitem). Lors de la création de l'adapter, on crée également un viewHolder qui va ordonner et relier à l'adapter les données de l'activité. De plus, le viewHolder déclare une interface onListListener appelée lors d'un clique sur un élément du recyclerView. On implémente l'interface OnListListener dans ChoixListActivity. Cette méthode permet de séparer le traitement qui se fait dans l'activité et la gestion des ViewHolder.

ShowListActivity et RecyclerViewAdapter2 :

L'activité est quasiment identique à ChoixListActivity. Et le RecyclerViewAdapter2 est quasiment identique à RecyclerViewAdapter1. Cela indique que l'usage de deux fragments est ici approprié.

La différence réside dans les checkBox à cocher ou décocher si l'Item est fait. Je n'ai pas réussi à relier la valeur prise par la checkBox du RecyclerView et la valeur de l'objet dans le profil de l'activité. J'ai donc choisi de faire le traitement dans le OnBindViewHolder ce qui ne me semble pas adapté.

La méthode d'ajout d'Item est la même que la méthode d'ajout de Liste dans ChoixListActivity.

Conclusion :

Ma connaissance parcellaire en programmation orientée objet m'a beaucoup ralenti. J'ai toujours eu l'impression de mal coder car je réutilise souvent les mêmes méthodes dans des activités différentes. Cependant, ce premier TP m'a permis de progresser énormément dans ce domaine.

Enfin, je pense avoir mal utilisé les outils de debuggage d'Android Studio. Chaque bug fut long à comprendre une fois détécté. Afficher des Log doit devenir pour moi un réflexe lors de l'ajout d'une méthode. (Il existe même des raccourcis pour cela !)

Perspectives :

Utilisation des fragments.

Envoyer l'objet en lui-même plutôt que son nom et de devoir reconstruire l'objet à chaque nouvelle activité. Utilisation de Parcelable.

Alban RAHIER

Changer le mode de raffraîchissement lors d’un ajout d’élément.

Fusionner les 3 RecyclerViewAdapter en un seul.

Optimiser l’activité ShowListActivity.

Ajouter une option de suppression des listes lors d’un appui long.

Bibliographie :

“RecyclerView OnClickListener (Best practice way)” : https://youtu.be/69C1ljfDvl0

“Dialogs” : https://developer.android.com/guide/topics/ui/dialogs

“Add a floating Button” : https://developer.android.com/guide/topics/ui/floating-action-button