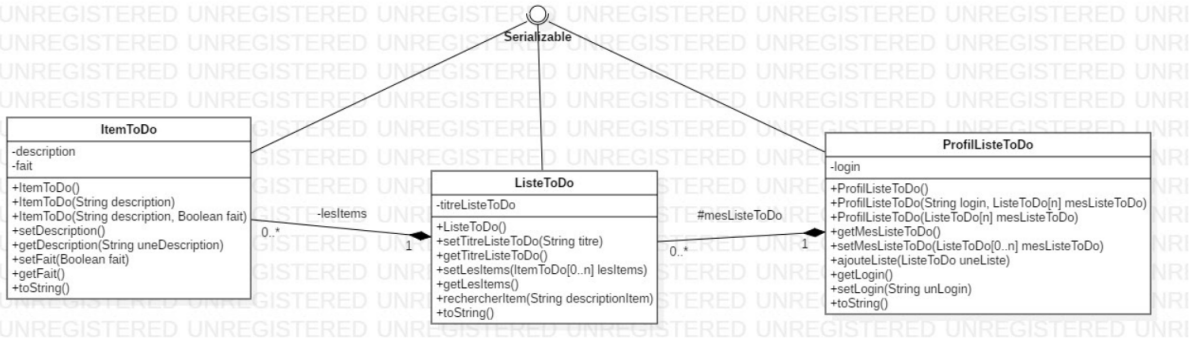
Électif PMR - Séquence 1

Introduction:

Ceci est le rapport de la séquence 1 de l'électif PMR sur la création d'une application de ToDoList en kotlin. L'application a été réalisé sous Android Studio et reprend le storyboard et le diagramme de classes suivant:



De plus, certaines classes ont été implantées dans l'application pour gérer les données de l'utilisateur et les adapter pour les recyclerview des différents objets de l'application.

Présentation des classes:

1) Les classes du diagrammes de classes

Les classes du diagramme de classes ont été implémentées avec les fonctions présentes mais certaines ont été ajoutées pour permettre différentes choses.
Pour la classe ProfilListeToDo, la fonction pasDejaDedansListe a été ajoutée pour savoir si une chaine de caractère n'est pas le nom d'une ListToDo de mesListesToDo. Il y a aussi la fonction RemoveListe qui permet de supprimer une liste de mesListesToDo.
Pour la classe ListeToDo, la fonction pasDejaDedansItem a été ajoutée pour savoir si une chaine de caractère n'est pas le nom d'un itemToDo dans mesItemsToDo. Il y a aussi la fonction AddItem pour ajouter un nouvel item dans mesItemsToDo et RemoveItem pour supprimer un item dans mesItemsToDo.
Pour la classe ItemToDo, la fonction SwitchState a été ajoutée pour changer l'état de l'item (fait ou non) sans connaître son état avant.

2) La classe pour l'enregistrement des données

Dans l'application, l'enregistrement des données se fait en enregistrant dans un fichier une chaîne de caractère représentant un objet JSON avec les différents profils, avec leurs listes à faire et les items de chaque liste. Pour gérer cela, c'est la classe LocalData.
Une instance de la classe est créée au lancement de l'application (pendant la création de la main activity) qui pourra être utilisée dans toutes les activités. Elle est référencée mesDonnees dans ces activités. A sa création, elle crée un fichier s' il n'existe pas déjà dans le répertoire. Le répertoire utilisé est dans les fichiers du téléphone dans l'endroit où les applications peuvent stocker des données en interne. Si le fichier existe, la classe récupère les données pour les stocker dans un objet JSON, sinon cet objet JSON sera initialisé lors de la première création d'un login par l'utilisateur.
Pour gérer les données, il ne peut y avoir deux profil avec le même nom, ni deux listes avec le même nom dans un même profil ni deux items avec le même nom dans une même liste. Si l'utilisateur essaye de ne pas faire comme ca, une erreur s'affiche dans toutes les activités sous forme de toast.
Cette classe est dotée des fonctions suivantes :
MAJInfos()
Cette fonction sert pour mettre à jour les données stockées dans l'objet JSON et réécrire cet objet comme une chaîne de caractères dans le fichier de sauvegarde.

addProfil(S :String)
Cette fonction sert à ajouter un profil dans la liste des profils.

addListe(newListe : String, ProfilToAdd : String) : Boolean
Cette fonction permet d'ajouter une liste dans un profil de la liste de profil si c'est possible. Elle renvoie un boolean, true pour savoir si on a pu ajouter la liste ou false sinon.

addItem(newListe : String, ListeToAdd :String, ProfilToAdd : String) Boolean
Cette fonction permet d'ajouter un item dans une liste dans un profil de la liste de profil si c'est possible. Elle renvoie un boolean, true pour savoir si on a pu ajouter la liste ou false sinon.

Sur les mêmes principes, il y a les mêmes fonctions pour enlever une liste ou un item.

getProfil(ProfilName : String)
Cette fonction renvoie le ProfilListeToDo qui a le nom ProfilName.

getListe(ProfilName : String, ListeName : String)
Cette fonction renvoie la ListeToDo qui a le nom ListeName dans le profilToDo ProfilName.

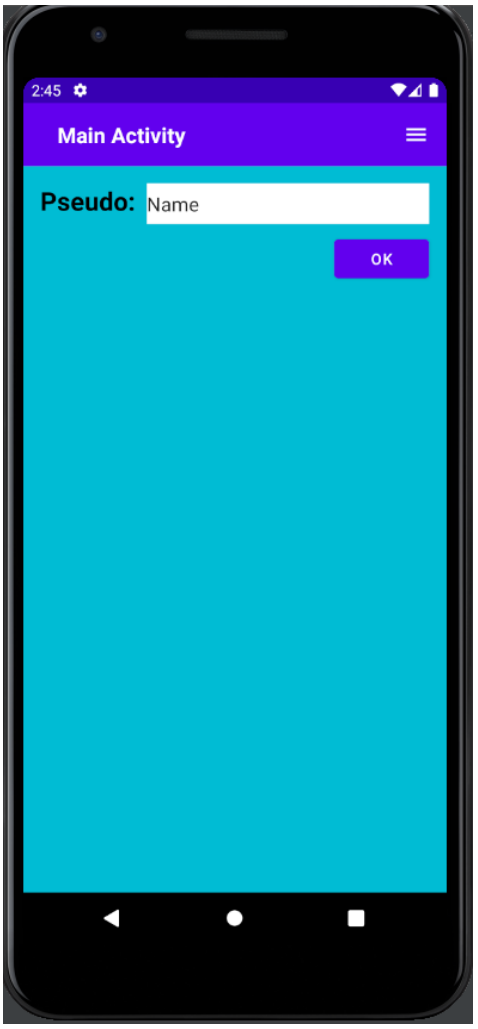
pasdejadedansLog(Log : String) : boolean
Cette fonction renvoie true si le Log n'est pas déjà dans la liste des profils et false sinon.

3) Autres classes

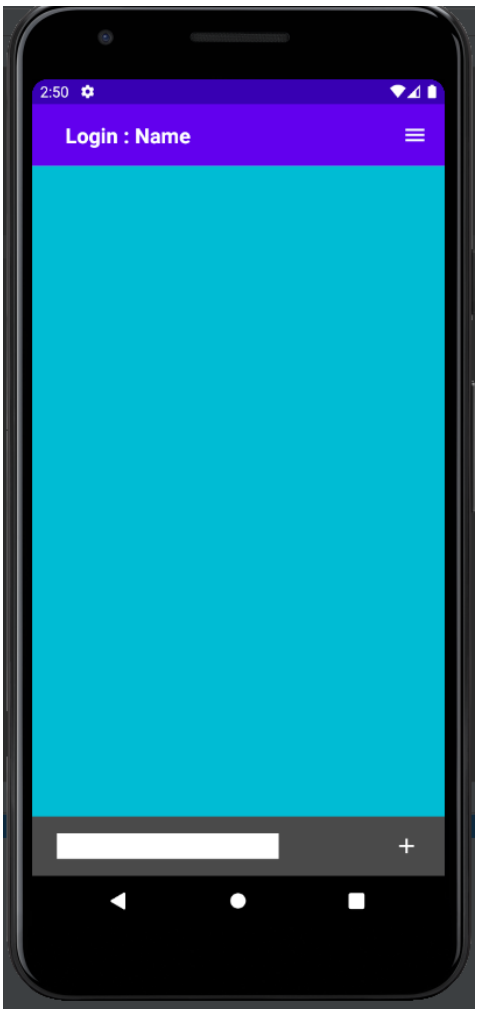
Les autres classes sont les classes pour chaque activité, donc pour la main activity la ListeActivity, la ItemListeActivity et les préférences, et pour les adapter pour les recyclerview.
Toutes les activités, sauf les préférences, sont munis d'un OnClick pour gérer les clicks sur l'activité.

Présentation de l’application

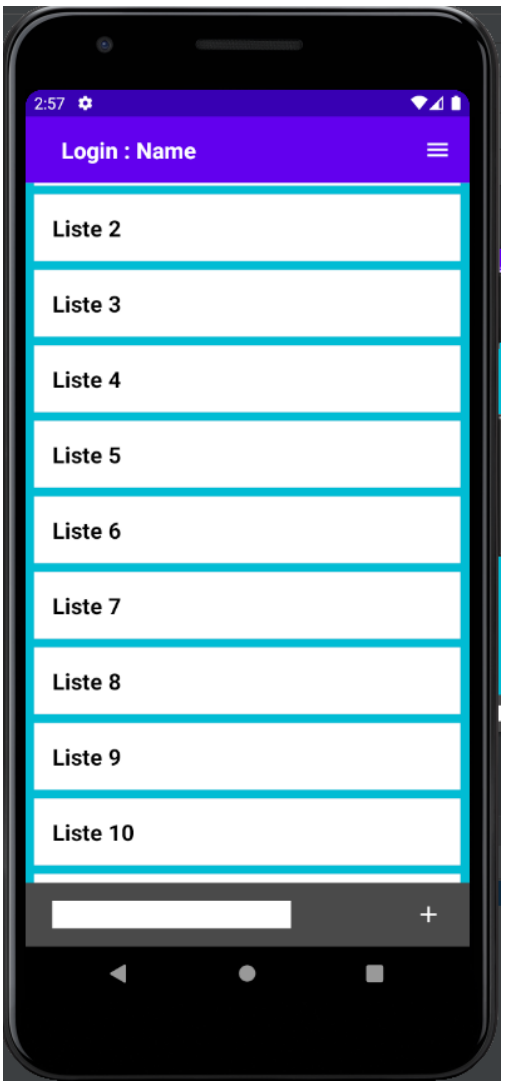
Au lancement de l’application, on a la mainActivity qui s’affiche:



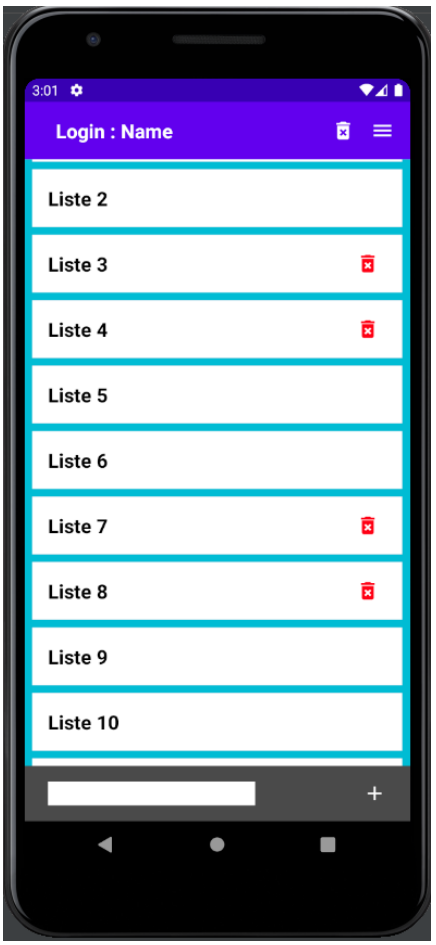
On peut y voir une toolbar avec un menu (cliquable pour ouvrir les préférences ensuite), un TextView, un EditText et un bouton “OK”. On rentre le nom du profil que l’on souhaite et en cliquant sur “OK“, on enregistre ce profil dans les préférences et on passe à la listeactivity avec un bundle de donnée pour le nom du profil courant. On crée alors un nouveau profil si le pseudo entré par l’utilisateur n'existe pas dans le fichier et sinon on récupère ses listes dans les données avec la variable mesDonnées.
On entre alors dans la ListeActivity:



La toolbar est encore présente avec le pseudo qui est entré par l'utilisateur, ici “Name”. Il y a encore le menu pour accéder aux préférences. Ici il n’y a pas de liste pour ce profil nous pouvons donc en créer plusieurs avec la zone du bas de l’écran. C’est un RelaTiveLayout avec une zone de text et un bouton + . On entre le nom de la liste à ajouter et on clique sur +. Si il y a déjà une liste avec le nom que nous avons entré, un toast s’affiche en disant qu’il y a déjà cette liste. On obtient, après la création des listes:

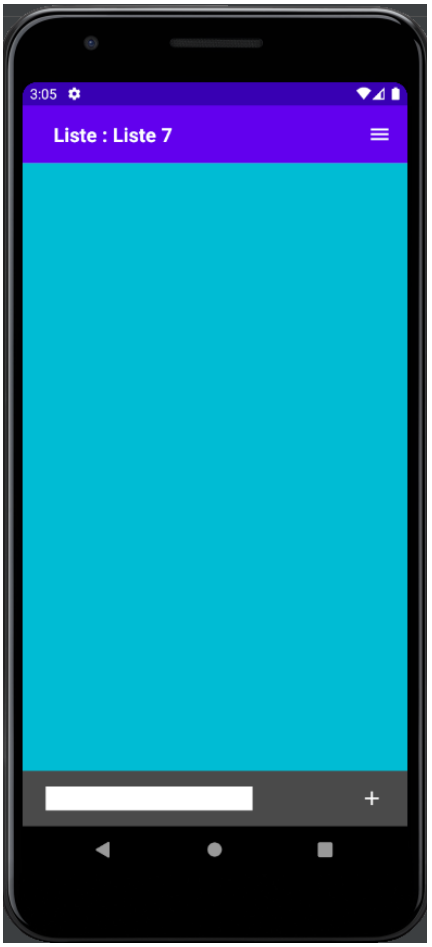


On voit ici les différentes listes possibles avec un recyclerView. Si on clique sur une liste on passe à la ItemActivity. Si on clique longtemps, une poubelle rouge sur la liste cliqué est ajoutée et une poubelle blanche dans la toolbar. La poubelle blanche est un bouton qui permet de supprimer les listes qui ont une poubelle rouge:

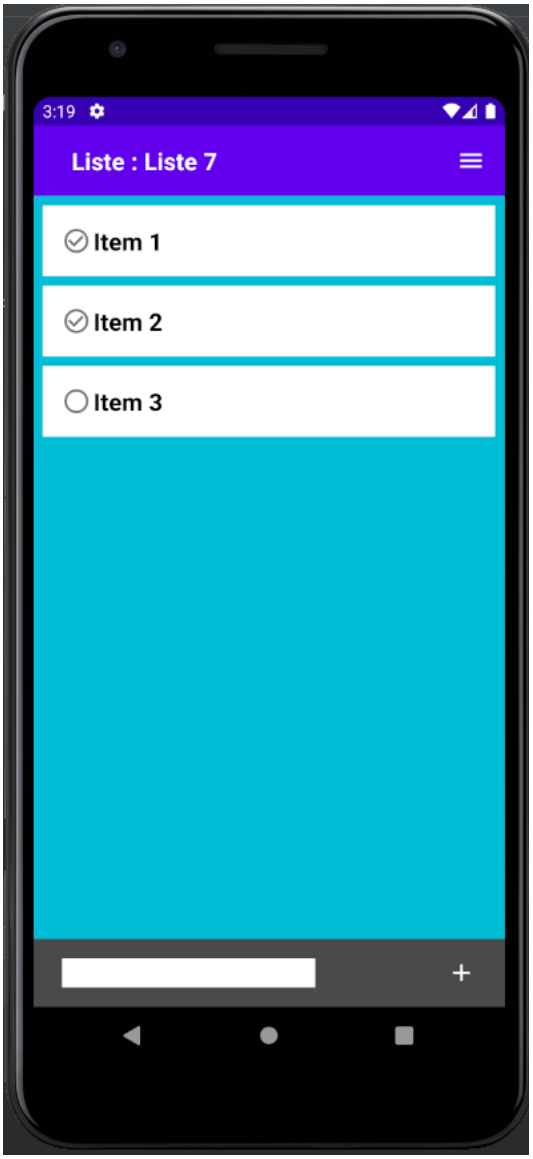

3:01
Login : Name
Liste 2
Liste 3
Liste 4
Liste 5
Liste 6
Liste 7
Liste 8
Liste 9
Liste 10


3:03
Login : Name
Liste 1
Liste 2
Liste 4
Liste 6
Liste 7
Liste 8
Liste 10
Liste 11

Imaginons que nous cliquons sur la liste 7, on va dans une activité ItemActivity avec un bundle pour le profil courant et la liste courante, ici “Name” et “Liste 7”. On obtient:



On arrive donc dans cette interface qui est quasiment la même que pour les Listes avec la toolbar et son menu, et le relativeLayout pour ajouter un item dans la liste 7. Après avoir créé des Items, on a:



On voit que les items 1 et 2 sont validés, ils sont réalisés. Pour changer l'état d'un item, il faut cliquer dessus.
Il y a aussi le même principe que pour les Liste pour supprimer les items de la liste.

Conclusion:

Pour conclure, on voit que l'application répond au cahier des charges de la ToDoListe. L'application est créée avec la classe LocalData pour gérer l'enregistrement des données mais peut être remplacée par une autre classe qui gère les données via une API ou autre sans changer les autres activités. De plus, un mode paysage de l'application pourrait être ajouté ainsi que le développement dans d'autres langues de l'application pour pouvoir être utilisé par tout le monde. L'ergonomie même de l'application peut être sans doute améliorée pour l'utilisation quotidienne de l'application avec un rangement par date ou par nom des listes et items pour un profil. Il y a donc encore des choses à ajouté pour l'application mais la base est là de façon fonctionelle.