CR TP 1 : To Do List

- **Introduction :**

L'objectif de l'application à développer est de permettre à l'utilisateur de gérer des To Do List. Plusieurs utilisateurs doivent pouvoir être ajouter. Pour chacun de ces utilisateurs, plusieurs listes seront crées contenant chacune plusieurs tâches.

Les enjeux ici étaient d'appréhender Android Studio pour le développement d'une première application simple, de découvrir les layouts de type RecyclerView et la librairie Gson. L'application devait permettre une persistance des données si bien qu'entre deux ouvertures de l'application, les utilisateurs et leurs listes sont toujours disponibles.

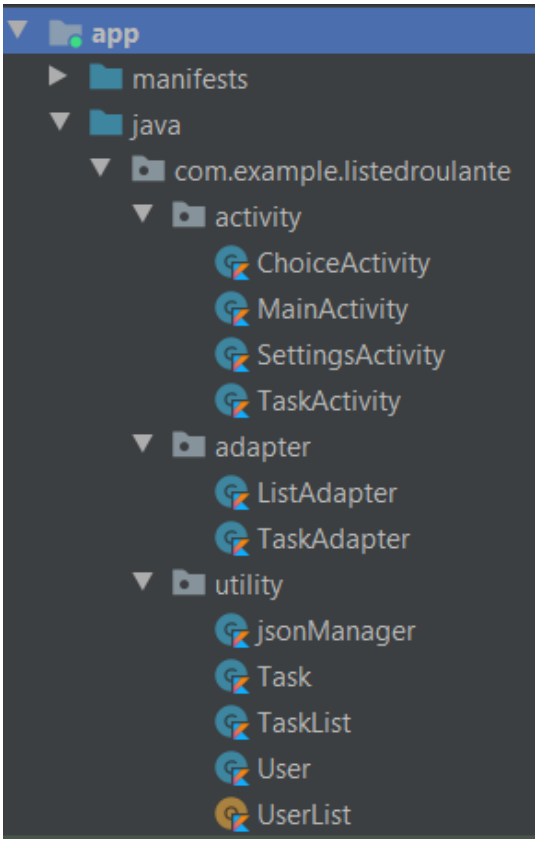
Le code proposé ici ne propose pas l'application finale. Celui-ci ne propose pas la persistance des données. Le codeur n'a en effet pas réussis à lire et écrire un fichier texte dans lequel écrire les données sous format Json.

- **Analyse :**

-Architecture

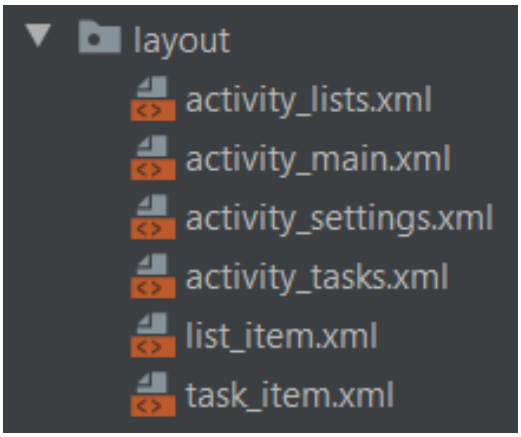
L'architecture proposée est la suivante. Un singleton UserList contient une MutableList<User>. Les User sont définis par leur nom (une String) et une MutableList<TaskList>. Chaque TaskList a pour attributs un nom et une description (String) ainsi qu'une MutableList<Task>. Les Task enfin, ont chacune un nom et une description de type String.

Le passage des données d'une activité à l'autre se fait donc grâce à l'utilisation du singleton UserList contenant toutes les informations nécessaires pour le développement de l'application. Ci-dessous est détaillée l'architecture de l'application.



-Fig 1 : architecture de l'application-

Les différentes ressources utilisées sont principalement les différents Layouts développés. Ceux-ci sont détaillés ci-dessous (Fig 2).



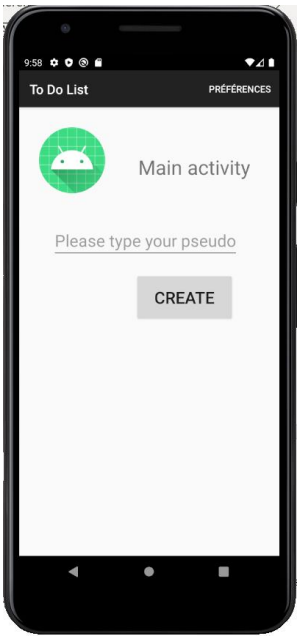
-Fig 2 : détail des layouts utilisés-

-Résultats

Ici est décrit l'ensemble des résultats obtenus.

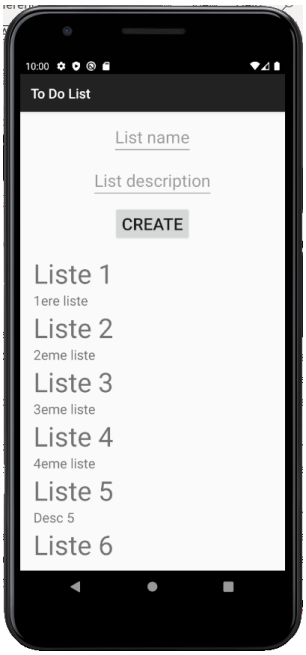
La première activité (Fig 3) présente la possibilité de créer un utilisateur à l'aide d'une zone de saisie. En appuyant sur le bouton CREATE, l'utilisateur est ajouté à la UserList et l'activité

suivante est affichée.



-Fig 3 : MainActivity-

Celle-ci (Fig 4) permet de faire défiler une liste de TaskList qui peut être modifiée en ajoutant des listes via deux zones de saisie et un bouton CREATE.



-Fig 4 : ChoiceActivity-

Au clique sur ce bouton CREATE, une nouvelle liste est ajoutée à la liste de TaskList de l'utilisateur en question. La prochaine activité est lancée.

Cette dernière (Fig 5) affiche les différentes tâches de la liste sélectionnée et permet d'ajouter des tâches via une zone de saisie et un bouton CREATE.

-Fig 5 : TaskActivity-

-Difficultés

La difficulté majeure rencontrée dans ce TP a été l'utilisation de méthodes et d'outils bien plus complexes que ceux approchés en POO. Malgré un suivi assidu de cet électif, ne codant pas à mes heures perdues, je ne disposais de l'aisance nécessaire pour avancer à un rythme convenable. En plus de cela, l'utilisation du langage Kotlin qui à terme s'avère plus aisé que le langage Java a tout de même demandé une phase d'adaptation non négligeable.

La deuxième difficulté à été de comprendre l'utilisation des Layouts. Ceux-ci ont un fonctionnement nouveau qui permettent une approche pratique des différents affichages de l'application.

Le troisième obstacle qui a demandé une petite semaine de travail a été l'implémentation des RecyclerView et l'utilisation des classes Adapter. Celles-ci ont pour rôle de gérer l'affichage d'une liste scrollable des différents éléments (les différentes listes ou encore les différentes tâches). Celui-ci contient une fonction getItemCount qui permet à l'application de savoir combien d'éléments elle doit afficher, une fonction onCreateViewHolder qui permet de charger le Layout contenant un item donné de la liste à afficher et enfin une fonction onBindViewHolder qui permet de charger le contenu d'un item dans le layout d'item. Le lien entre onCreatViewHolder et onBindViewHolder se fait à l'aide d'une classe ViewHolder contenant une fonction bind permettant de fournir les id des layouts à remplir. C'est aussi via cette fonction qu'on gère le click sur un des items de la liste avec un setOnClickListener.

La troisième difficulté qui n'a pas été résolue est l'implémentation de la persistance des

données. Une classe JsonManager a été développée pour gérer la transformation des classes en format Gson et inversement ainsi que l'écriture ou la lecture de ces données depuis un fichiers local. Malgré tous les essais effectués, cette étape n'a pas été fructueuse. Elle aurait pourtant permis d'avoir une application presque terminée, à l'exception des préférences qui ont ici été laissées de côté : l'activité existe mais ne permet aucune manipulation.

- **Conclusion :**

  L'application n'est pas terminée et n'est pas utilisable dans son état actuel. Etant donné l'avancement, des dizaines d'heures supplémentaires serait nécessaire pour que je puisse développer le reste des fonctionnalités de cette application.

- **Perspectives :**

  Beaucoup d'améliorations et de compléments sont à ajouter : gérer la persistance des données, récupérer la position de l'éléments cliqué pour afficher effectivement cet élément et implémenter la partie préférences de l'application. Il semble compliqué de passer du temps sur ce développement étant donnés les travaux suivants à réaliser et l'application de réalité augmentée à programmer. Je prie donc l'évaluateur de croire en ma bonne volonté vis-à-vis des tentatives nombreuses et chronophages qui ont été effectuées.

- **Bibliographie :**

Menu :

ThomasBourdeaudhuy_menu_android on Vimeo,

ThomasBourdeaudhuy_ecouteurs_android on Vimeo,

ThomasBourdeaudhuy_toast_android on Vimeo, How to implement switch-case statement in Kotlin - Stack Overflow

RecyclerView :

How To: RecyclerView with a Kotlin-Style Click Listener in Android – andreasjakl.com,

android - onclicklistener sur le point spécifique de la recyclerview dans android,

Recyclerview | Android Developers Objects and companion objects - Kotlin Programming Language,

android - Intents in Kotlin - Stack Overflow, RecyclerView OnClickListener (Best practice way) - YouTube,

RecyclerView Kotlin Tutorial - AndroidWave, Android RecyclerView Tutorial with Kotlin | raywenderlich.com,

Android working with RecyclerView Using the RecyclerView | CodePath Android Cliffnotes,

Getting Started With RecyclerView in Android — Kotlin Start another activity | Android Developers

How to change activity in android? - Stack Overflow,

How to create a singleton class in Kotlin?,

Android getText from EditText field - Stack Overflow

Persistance des données :

Android Persistence with preferences and files - Tutorial

How to create a file in Android? - Stack Overflow ,

How to Read and Write JSON data in Kotlin with GSON ,

Kotlin Android - Read JSON file from assets using Gson - BezKoder, Gson - How to convert Java object to / from JSON - Mkyong.com ,

java - Read/Write String from/to a File in Android - Stack Overflow,

How to convert Java object to JSON String - Gson Java/JSON Serialization Example | Java67,

How to Convert a Java Object into a JSON String | Codota Blog Android Read Write Internal Storage File Example,

How to Read JSON from a File using Gson (Example)

How to read a JSON file on Android - Quora,

How to create a file in Android? - Stack Overflow,

How to Read and Write JSON data in Kotlin with GSON

Kotlin Android - Read JSON file from assets using Gson - BezKoder,

Gson - How to convert Java object to / from JSON - Mkyong.com,

java - Read/Write String from/to a File in Android - Stack Overflow,

How to convert Java object to JSON String - Gson Java/JSON Serialization Example | Java67,

How to Convert a Java Object into a JSON String | Codota Blog

Android Read Write Internal Storage File Example,

How to Read JSON from a File using Gson (Example),

How to read a JSON file on Android - Quora