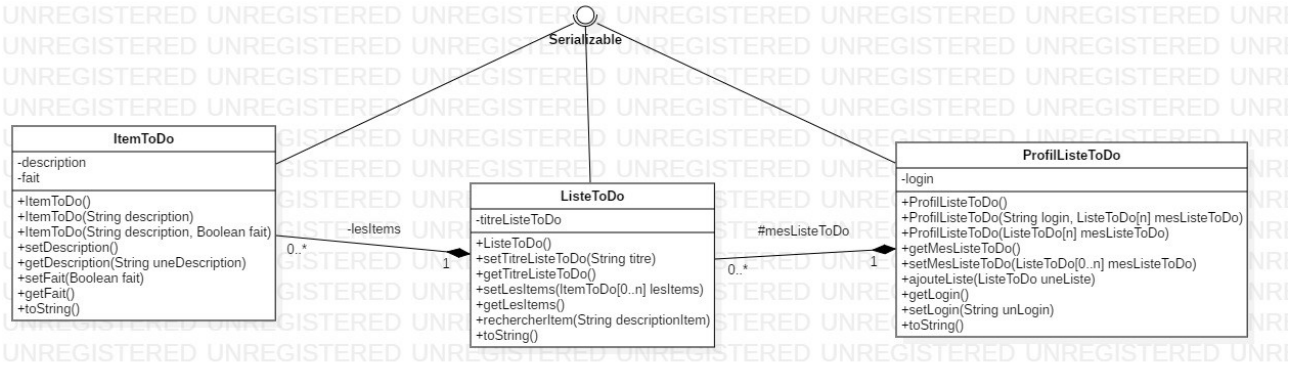
Le Guillou Eve
Thomas Iona

Compte-rendu Application TodoList

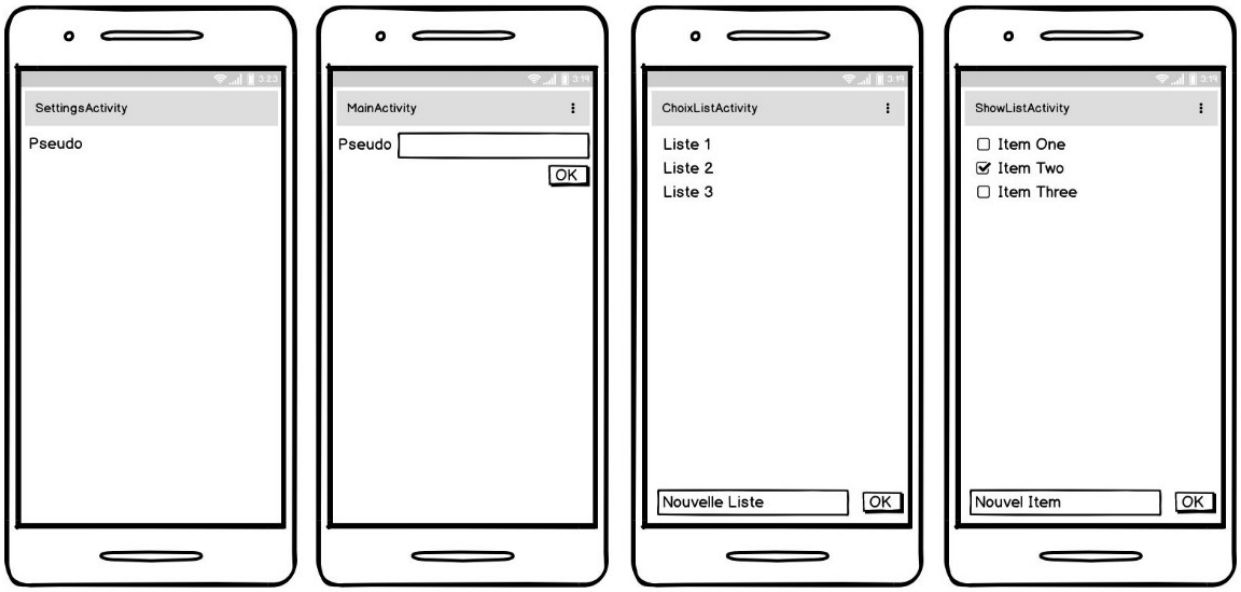
Introduction

Le mini-projet de cette séquence était le développement d'une application android en Java permettant de gérer des todo listes associées à des profils d'utilisateur.

Un diagramme UML de classe était fourni et servait de base à l'implémentation des classes nécessaires :



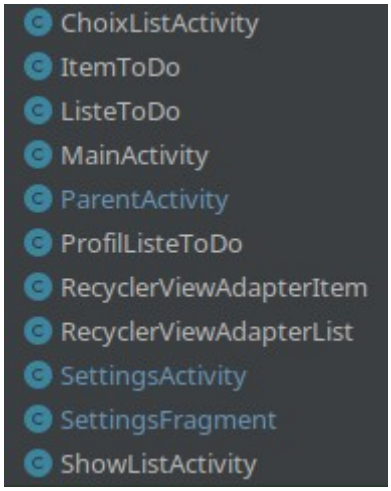
Et les mockups et le storyboard suivant était également fournis :



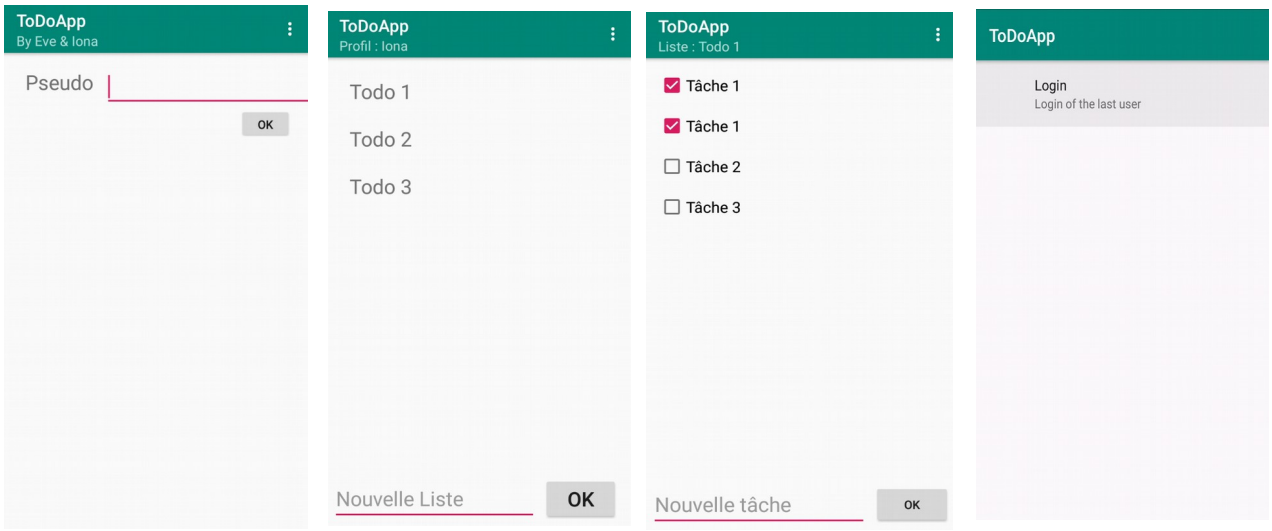
En résumé on dispose d'une activité MainActivity sur lequel l'utilisateur rentre son pseudo. On passe ce pseudo à l'activité suivante « ChoixListActivity ». Si le profil existe déjà on charge le fichier correspondant pour récupérer ses informations, sinon on en crée un nouveau. Cette activité affiche ensuite les listes correspondant au profil. On peut en

ajouter ou aller sur une liste déjà existante. Dans l'activité « ShowListActivity » on affiche les tâches à faire incluses dans la liste et leur état (faites ou non). Enfin une « SettingsActivity » est accessible à tout instant en utilisant le menu de l'actionBar.

Dans nos sources on dispose donc des classes suivantes qu'on détaillera dans l'analyse :



Les différentes activités ont pour apparence :



MainActivity ChoixListActivity ShowListActivty SettingsActivity

Afin de remplacer le logo par défaut et de rendre notre application plus identifiable nous avons mis en place ce logo :

1. Analyse

i. ParentActivity

Dans un effort de factorisation du code, on crée une nouvelle activité dont vont hériter toutes les autres activités sauf SettingsActivity. On y regroupe ainsi toutes les opérations et déclarations de méthodes effectuées à répétition, notamment :
- Les méthodes et attributs permettant de gérer le menu dans l'ActionBar
- La méthode alerter permettant de créer à la fois un log et un Toast pour un message
- Les méthodes importProfil, newProfil et sauveProfilToJson qui permettent respectivement d'importer le profil depuis un fichier où les données sont écrites en Json et d'initialiser un objet ProfilToDoList avec les valeurs obtenues du fichier, de créer un nouveau fichier correspondant au profil, et de sauvegarder la valeur d'un objet ProfilToDoList dans un fichier en Json.

ii. SettingsActivity

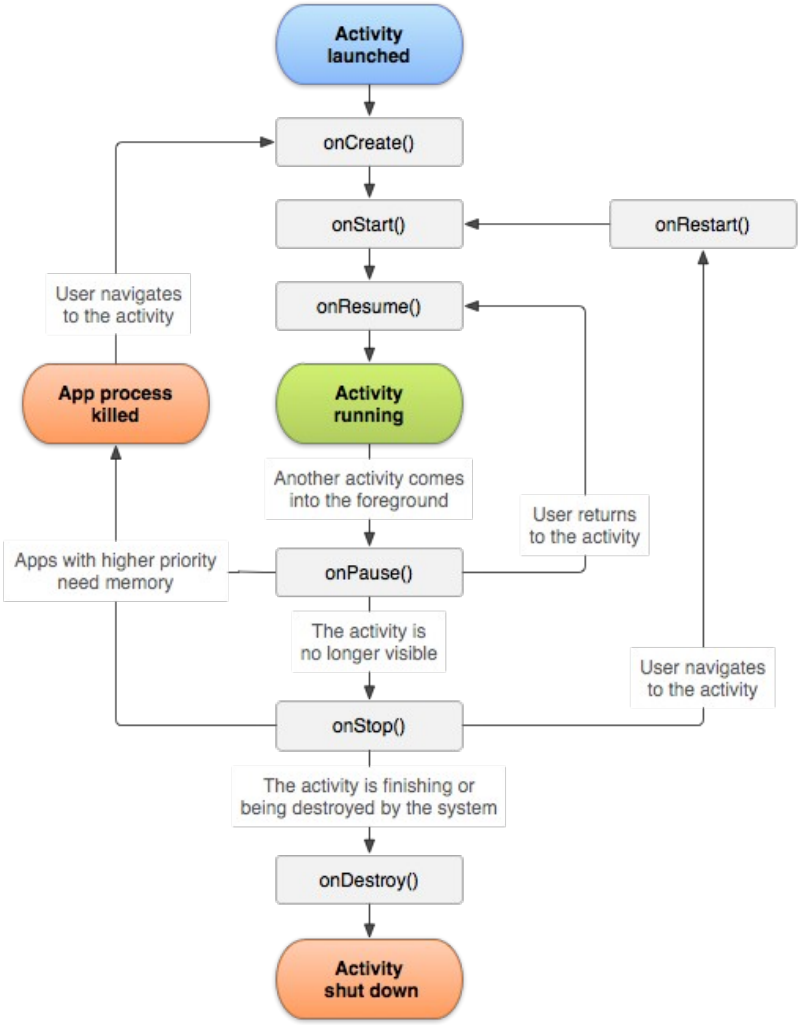
Depuis cette activité, on peut accéder au menu Préférences, qui lance SettingsActivity, et on peut entrer son pseudo. Le pseudo est alors mémorisé pour la prochaine connexion sur la « MainActivity ».

iii. ChoixListActivity

Depuis cette activité, on peut accéder aux Listes Todo déjà créées et en ajouter de nouvelles.

Cette activité peut être chargée lorsqu'on arrive de MainActivity, auquel cas elle récupère les informations du profil à l'aide de la chaîne de caractère pseudo placée dans le bundle. Elle peut également être « chargée » lorsqu'on revient de « ShowListActivity ». Dans ce second cas il est nécessaire de réimporter le profil sauvegardé sinon les modifications effectuées dans ShowListActivity seront perdues dès qu'on quittera cette activité puisqu'on écrasera le fichier du profil avec des informations périmées.

Donc, lors de onCreate, on récupère le bundle créé par MainActivity, mais c'est dans onStart que l'on fait la majorité de nos actions, car onCreate n'est appelée qu'à la création de l'activité alors que onStart sera appelée à chaque fois que l'activité est chargée. Donc quand on reviendra de ShowListActivities elle s'exécutera ce qui permet notamment de charger le profil à chaque fois pour ne pas perdre d'informations. On y a également placé le setOnClickListener et l'initialisation du RecyclerView puisque ce dernier a besoin des informations du profil pour être initialisé.



Les listes Todo sont affichées grâce à un RecyclerView dont l'adaptateur est défini dans RecyclerViewAdapterList. Pour adapter le layout en fonction de l'orientation du téléphone, on choisie un LinearLayoutManager si on est en orientation portrait et un GridLayoutManager avec 2 colonnes si on est en paysage. Puisque l'activité est détruite et reconstruite lorsqu'on change d'orientation, le layout manager est bien changé lorsqu'on change l'orientation du téléphone.

Pour générer des changements plus importants dans les IHM lors du changement d'orientation on pourrait utiliser des dossiers de ressources layout-port, layout-land pour spécifier des layouts qui devraient changer selon l'orientation du téléphone.

Lorsqu'on crée une nouvelle liste, on ajoute celle-ci aux listes du profils, on vide le champ texte pour une meilleure expérience utilisateur, et on notifie l'adaptateur qu'il y a eu des changements ce qui lui permet de recalculer la nouvelle vue.

En cliquant sur une des listes, on ouvre une activité ShowListActivity en lui passant le pseudo dans un Bundle.

iv. ShowListActivity

Depuis cette activité, on peut accéder aux tâches déjà créées et en ajouter de nouvelles. Les tâches sont affichées grâce à un RecyclerView dont l'adaptateur est défini

dans RecyclerViewAdapterItem. Lors de onCreate, on importe le profil, pour ainsi accéder aux tâches de la ToDo dans laquelle on est. Comme expliqué précédemment dans ChoixListActivity on initialise le recyclerView et selon l'orientation le layout manager. Chaque tâche est associée à une checkbox, laquelle peut être cochée ou décochée en cliquant dessus. Plutôt qu'utiliser les méthodes suggérées pour gérer l'état d'une checkbox, on utilise une nouvelle méthode : toggleFait, qui permet de simplifier le code, en limitant le nombre d'appel à des méthodes et la transmission d'information entre classes.

v. SettingsActivity

Cette activité permet d'afficher les préférences de l'utilisateur, ici limitées à son pseudonyme, et de le changer en cliquant dessus. On peut accéder à cette activité depuis toutes les autres activité de l'application.

Elle utilise un fragment pour les préférences qui hérite de PreferenceFragment-Compat. Cette dernière classe est spécialisée pour la gestion des préférences à l'aide de fragments. Le fragment est ajouté statiquement dans le layout activity_settings.xml.

vi. RecyclerViewAdapter[Item/List]

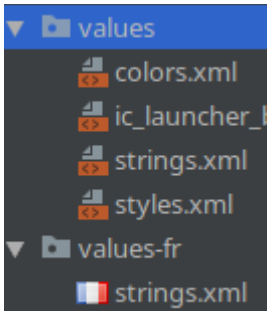
Ces classes définissent les adaptateurs pour les RecyclerView respectivement des tâches et des todo listes. Elles permettent de faire le lien entre les données qu'on veut afficher et le Layout du RecyclerView.

Les données des tâches et des listes étant différentes, on a mis en place deux layout différents pour les afficher. Les tâches sont affichées dans des textes View et les tâches dans des checkbox associées a leur texte. En raison de ces différences on implémente deux adaptateurs avec deux ViewHolder différents.

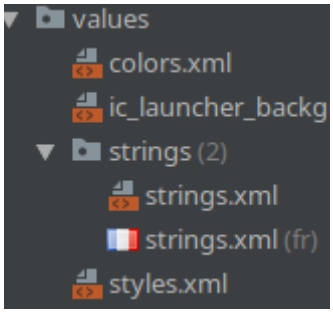
vii. Internationalisation

Pour internationaliser l'application, on crée de nouveaux dossiers et fichiers : un dossier par langue/localisation qu'on souhaite ajouter portant le même nom que le dossier ressources qu'on souhaite modifier suivi de -[Code de la langue]. Ici on l'applique sur le dossier values, on crée dedans un fichier string.xml et on associe à chaque chaîne de caractères la valeur dans la langue choisie. Le système Android choisira le bon fichier en fonction des paramètres de l'appareil.

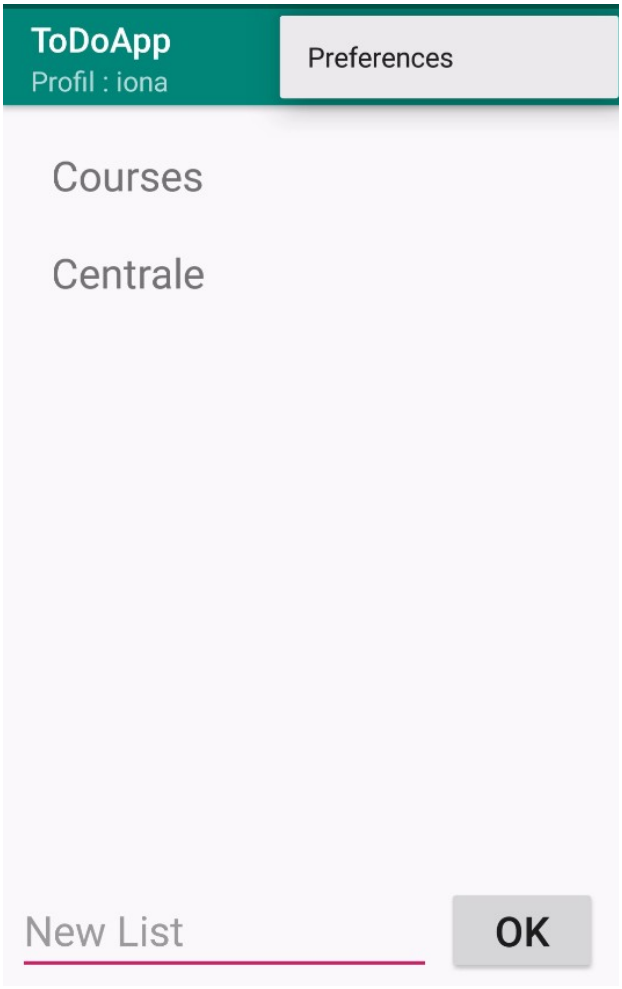
Vue Project :

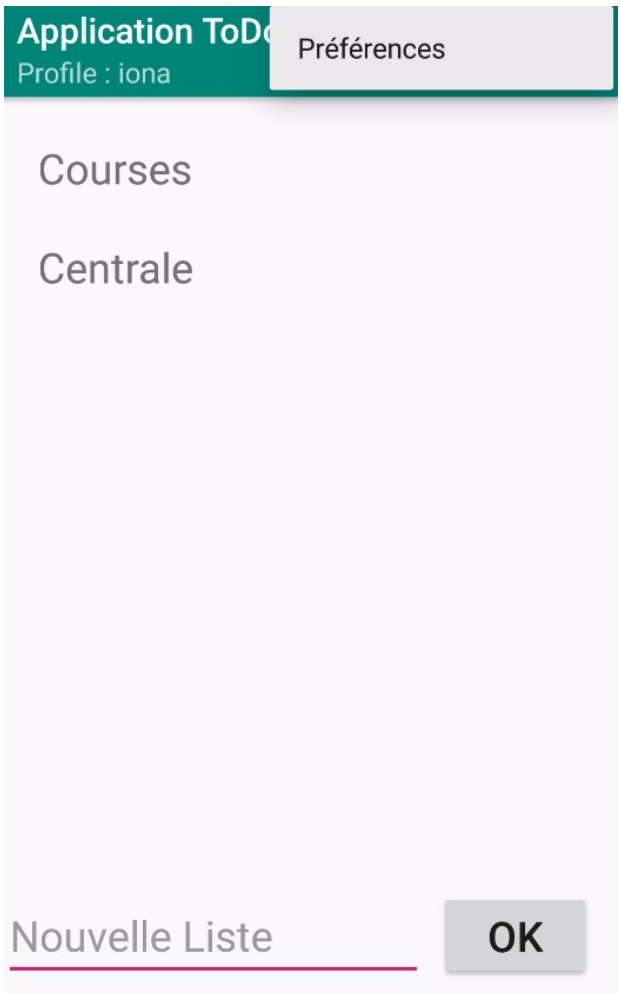


Vue Android :



Résultat :





Bien entendu, les listes étant des données entrées par l'utilisateur elles ne sont pas traduites.

Conclusion

Conduit de bout en bout, cet exercice permet de manipuler et mettre en œuvre de nombreux concepts vus en cours et d'explorer beaucoup de pistes différentes.

L'exercice pousse à se poser des questions sur certains éléments de design, voire à les remettre en question : en effet, afficher le nom de l'activité dans l'actionBar n'est pas très parlant ni intéressant. Nous avons donc choisi de garder le titre de l'application mais en ajoutant des sous-titres apportant des informations complémentaires à l'utilisateur : profil en cours d'édition, liste en cours d'édition … Nous avons également ajouter une méthode toggle à la classe itemToDo afin de faciliter l'interaction correspondant aux checkbox.

Nous avons également mis en place des garde-fous concernant les entrées vides, que ce soit des pseudo, des listes ou des tâches.

Enfin nous avons également pu mettre en place un fragment pour les préférences ce qui nous a permi d'avoir un premier aperçu simplifié des fragments. En effet Preference-FragmentCompat simplifie grandement la mise en place.

Perspectives

Il subsiste encore de nombreuses perspectives d'amélioration. On peut ajouter une classe intermédiaire pour factoriser les méthodes onCreate des activités ShowListActivty et ChoixListActivity, y placer les méthodes concernant l'import, la création et l'export des profils, et ainsi réduire encore le code, sans surcharger MainActivity. On peut aussi penser à ajouter un mot de passe, afin de sécuriser l'accès aux listes Todo, de pouvoir masquer et de supprimer les tâches déjà faites. L'historique des pseudo n'a pas encore été mis en place.

Enfin on peut envisager une utilisation plus efficace de nos préférences. En effet si on change les préférences depuis ShowListActivity ou ChoixListActivity, le profil précédent est gardé en mémoire. Ce n'est que lorsqu'on repasse par MainActivity que le pseudo entré dans les préférences est suggéré et peut être sélectionné. C'est dû à la méthode d'import du pseudo qui est passée depuis MainActivity et dont les modifications effectuées sont gardées en mémoire après par ChoixListActivity, même quand on modifie le pseudo dans les préférences. Pour contrer ce problème une solution serait charger le profil depuis les préférences et non pas depuis le bundle où l'historique dans le onStart() de ChoixListActivity. Comme cela, à chaque fois que l'activité choixListActivity apparaît à l'écran, le profil correspondra à celui qui se trouve dans les préférences.

Bibliographie

Construire un RecyclerView :
https://www.youtube.com/watch?v=Vyqz_-sJGFk

Faire un OnItemClick dans un RecyclerView :
https://www.youtube.com/watch?v=69C1IjfDvl0

Personnalisation de l'AppBar :
https://developer.android.com/training/appbar/setting-up

Internationalisation :
https://developer.android.com/guide/topics/resources/localization.html

Fragement pour les préférences :
https://developer.android.com/guide/topics/ui/settings.html

Cycle de vie d'une application android :
https://developer.android.com/guide/components/activities/activity-lifecycle