Rapport TEA1

Programmation Mobile et Réalité Augmentée

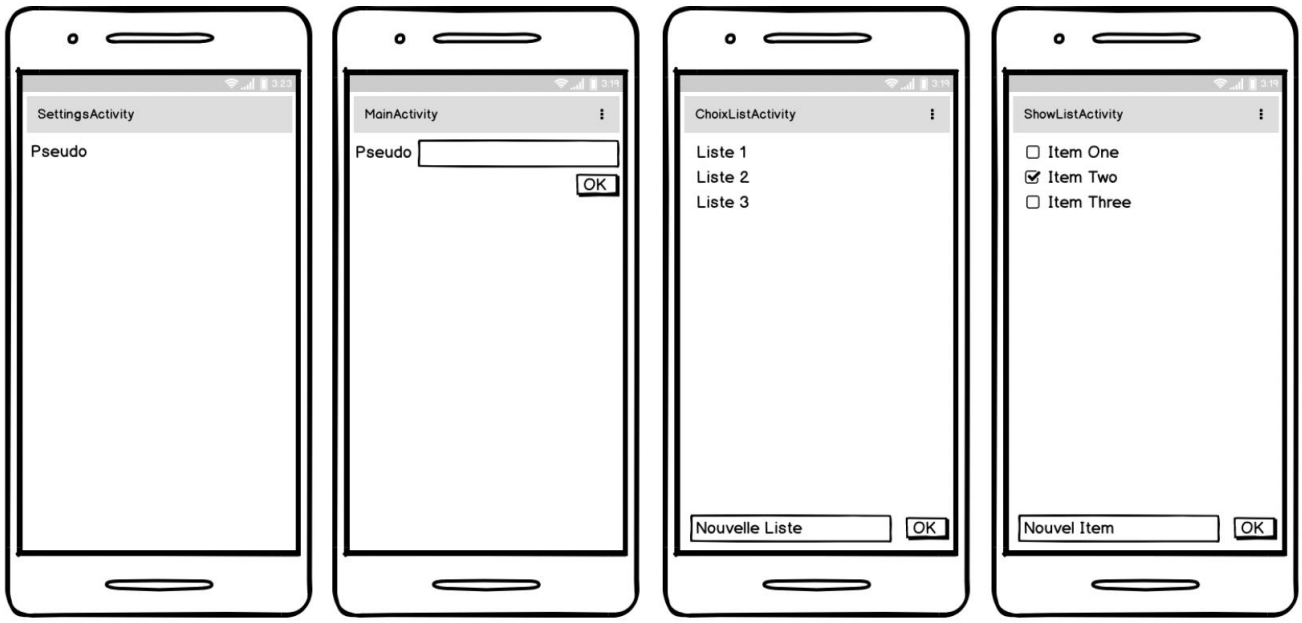
ROSSI Marien - INGELAERE Clément

5 juin 2023

Rappel des objectifs

- Mise en place de layouts évolués utilisant des ListView et des Adapters
- Sérialisation/Déserialisation en JSON avec la librairie GSON
- Persistance de données en préférences / Fichiers

Storyboard



MainActivity :

- Permet de saisir son pseudo
- Le dernier pseudo saisi est automatiquement renseigné dans le champ de saisie
  - [Facultatif] Des propositions de complétion pour les pseudo s'affichent lors de la saisie, à partir de l'historique des pseudos saisis auparavant
- Un menu **ActionBar** est proposé en haut à droite.
  - Lors du clic sur le bouton "Préférences" de ce menu, une activité **SettingsActivity** s'affiche
- Lors du clic sur le bouton OK, le pseudo est sauvegardé dans les préférences partagées de l'application et l'activité **ChoixListActivity** s'affiche
  - On lui passera la valeur du pseudo saisi

SettingsActivity :

- Hérite de PreferenceActivity, permet de manipuler les préférences de l'application
- Champ pseudo affiché par défaut
  - [Facultatif] On peut vider l'historique des pseudos saisis auparavant
- D'autres champs utiles peuvent être ajoutés, pour débogage par exemple

ChoixListActivity :

- Affiche la liste des listes de l'utilisateur dont le pseudo a été saisi
- Lors du clic sur une liste, une activité **ShowListActivity** s'affiche
  - On lui passera l'indice de la liste sélectionnée
- Permet d'ajouter une nouvelle liste pour cet utilisateur

ShowListActivity :

- Affiche les items des listes de l'utilisateur dont le pseudo a été saisi
- Permet de cocher ou décocher un item
- Permet d'ajouter un nouvel item dans cette liste

La liste ci-dessus énumère les différentes activités à implémenter afin de réaliser une application mobile permettant la gestion de listes (visualisation des listes, ajout de nouveaux items …).

Chaque fichier lié à une activité est accompagné d'un fichier .xml qui permet d'implémenter les graphismes de l'activité.

Ce projet a été réalisé en Kotlin qui présente plusieurs avantages par rapport à java tels que la concision du code, la gestion de la nullabilité des objets ainsi que la prise en main d'un écosystème intuitif avec AndroidStudio.

Activité MainActivity

L'activité "MainActivity" est conçue pour permettre à l'utilisateur de saisir son pseudo et d'accéder à différentes fonctionnalités de l'application. Voici une description détaillée du fonctionnement de cette activité :

Saisie du pseudo

Lorsque l'utilisateur ouvre l'application, il est accueilli par l'activité "MainActivity". Cette activité affiche un champ de saisie où l'utilisateur peut entrer son pseudo. Il peut saisir son pseudo à l'aide du clavier virtuel ou du clavier physique de l'appareil.

Renseignement automatique du dernier pseudo

Lorsque l'utilisateur a déjà saisi un pseudo précédemment, celui-ci est automatiquement renseigné dans le champ de saisie lors de l'ouverture de l'activité "MainActivity". Cela permet à l'utilisateur de visualiser et de modifier facilement son pseudo précédent s'il le souhaite. Le dernier pseudo est stocké avec la clé 'pseudo' dans les préférences et mis a jour lors du clic sur OK.

Menu ActionBar

En haut à droite de l'activité "MainActivity", un menu ActionBar est affiché. Ce menu propose différentes options à l'utilisateur. Parmi ces options, se trouve le bouton « Préférences". On met à jour les actions a effectuer lors du clic sur les items du menu dans la fonction 'onOptionsItemSelected'

Clic sur le bouton "Préférences"

Lorsque l'utilisateur clique sur le bouton "Préférences" dans le menu ActionBar, une nouvelle activité appelée "SettingsActivity" s'affiche. Cette activité permet à l'utilisateur de configurer diverses préférences de l'application.

Sauvegarde du pseudo dans les préférences partagées

Après avoir saisi son pseudo dans l'activité "MainActivity" et cliqué sur le bouton "OK" dans l'activité "SettingsActivity", le pseudo saisi est sauvegardé dans les préférences partagées de l'application. On stocke également la liste des pseudos enregistrés dans la clé 'pseudos' afin de pouvoir adapter le code selon si on a un nouvel utilisateur ou non.

Si on a un nouvel utilisateur, on crée une clé à son nom et on stocke une liste basique avec des items basiques en utilisant la librairie GSON() :

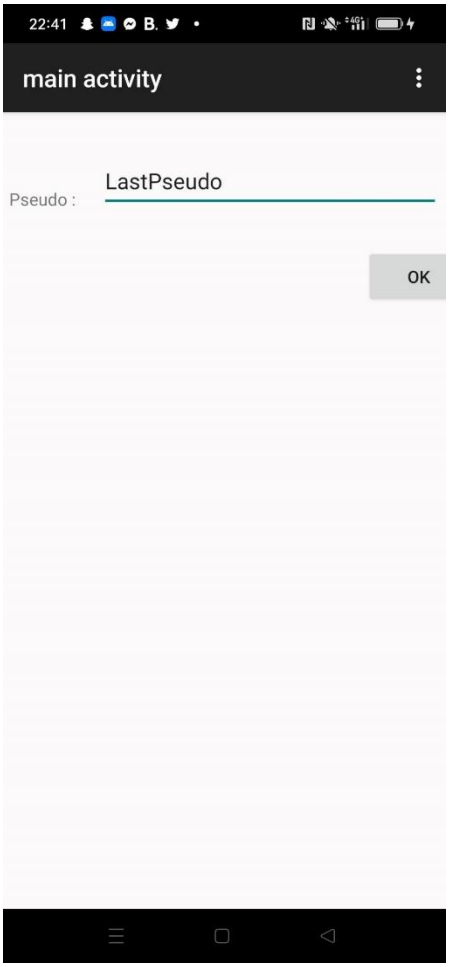
```
val gson = Gson()
val defaultObject = ChoixListActivity.DataType(
    listOf("Liste 1"),
    listOf(listOf(Item("Item1", false)))
)
val jsonString = gson.toJson(defaultObject)
editor.putString("$pseudoInput", jsonString)
```

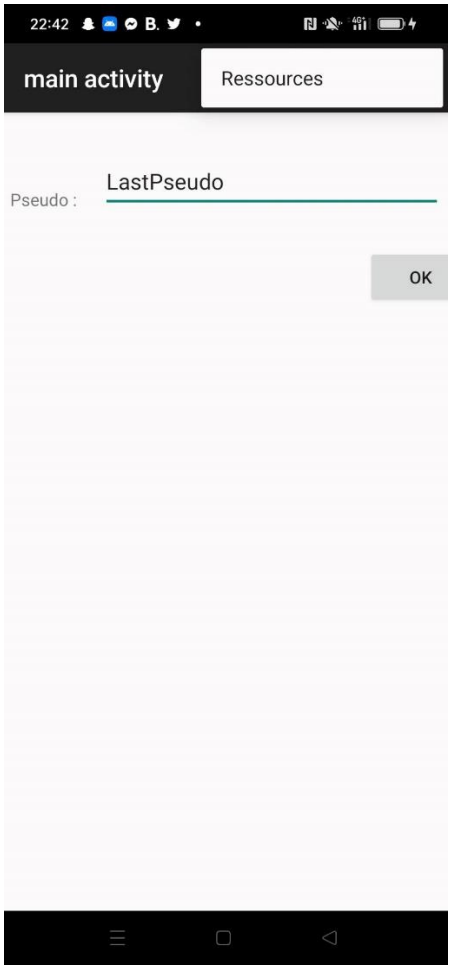
Ici, dataType est une classe créée dans le but de stocker les données d'un utilisateur et contient une liste de titres, ainsi qu'une liste de liste d'items.

Les préférences partagées permettent de stocker des données de configuration de manière persistante.

Affichage de l'activité "ChoixListActivity"

Une fois le pseudo sauvegardé, l'activité "ChoixListActivity" s'affiche. Cette activité est responsable de la gestion des listes et nécessite le pseudo de l'utilisateur pour fonctionner correctement. Le pseudo saisi est passé à cette activité en tant que valeur, permettant à l'activité "ChoixListActivity" d'utiliser cette information pour effectuer des opérations spécifiques.


22:41
main activity
Pseudo :
LastPseudo
OK


22:42
main activity
Ressources
Pseudo :
LastPseudo
OK

Activité SettingsActivity

L'activité "SettingsActivity" est conçue pour permettre à l'utilisateur de manipuler les préférences de l'application. Voici une description détaillée du fonctionnement de cette activité :

Héritage de PreferenceActivity

L'activité "SettingsActivity" hérite de la classe "PreferenceActivity", qui est une classe fournie par Android pour faciliter la manipulation des préférences. Cela permet à l'activité d'avoir des fonctionnalités intégrées pour afficher et gérer les différentes préférences de l'application.

Affichage du champ pseudo par défaut

Lorsque l'utilisateur ouvre l'activité "SettingsActivity", un champ pour le pseudo est affiché par défaut. Ce champ peut être utilisé pour permettre à l'utilisateur de saisir ou de modifier son pseudo dans les préférences de l'application. Il affiche le champ pseudo stocké dans les préférences.

Sauvegarde des préférences

Lorsque l'utilisateur effectue des modifications dans les préférences de l'application, les nouvelles valeurs des préférences sont sauvegardées. Cela permet de conserver les préférences choisies par l'utilisateur même après la fermeture de l'application.

On a surtout utilisé cette page pour afficher nos préférences et débuguer nos erreurs.

Activité ChoixListActivity

L'activité "ChoixListActivity" est conçue pour afficher la liste des listes de l'utilisateur, en fonction du pseudo saisi.

On a ici une exception NullPointer que nous n'arrivons pas à résoudre lors de l'appel à Intent.getStringExtra().

Nous n'avons pas réussi à débuguer cette page, mais nous avons essayé de mettre en place une RecyclerView tout de même, ainsi que la récupération des listes et items de l'utilisateur en fonction de son login, en utilisant la librairie GSON() et les préférences partagées.

Voici une description détaillée du fonctionnement de cette activité :

Affichage de la liste des listes de l'utilisateur

Lorsque l'utilisateur accède à l'activité "ChoixListActivity", la liste des listes spécifiques à cet utilisateur est affichée. Cette liste est récupérée grâce à la clé fournie dans l'intent, qui correspond au pseudo de l'utilisateur.

Clic sur une liste

Lorsque l'utilisateur clique sur l'une des listes affichées dans l'activité "ChoixListActivity", une nouvelle activité appelée "ShowListActivity" est lancée. L'indice de la liste sélectionnée est passé en tant que paramètre à cette activité, afin de pouvoir afficher et manipuler la liste sélectionnée spécifiquement

Passage de l'indice de la liste sélectionnée

Pour transmettre l'indice de la liste sélectionnée à l'activité "ShowListActivity", on peut utiliser l'Intent pour transmettre à ShowList le numéro de l'activité ainsi que le pseudo. A partir de ces deux données, on peut accéder à une liste d'Items.

Ajout d'une nouvelle liste

L'activité "ChoixListActivity" offre également la possibilité d'ajouter une nouvelle liste pour cet utilisateur. Cela peut être réalisé en cliquant sur un bouton d'ajout ou en utilisant une autre interaction utilisateur appropriée. Lorsque l'utilisateur ajoute une nouvelle liste, les informations de cette liste peuvent être collectées à partir de l'interface utilisateur et enregistrées dans la source de données appropriée pour cet utilisateur.

Activité ShowListActivity

L'activité "ShowListActivity" est conçue pour afficher les éléments des listes de l'utilisateur, en fonction du pseudo saisi. Voici une description détaillée du fonctionnement de cette activité :

Affichage des éléments de la liste

Lorsque l'utilisateur accède à l'activité "ShowListActivity", les éléments de la liste spécifique à cet utilisateur sont affichés. Ces éléments peuvent être récupérés à partir d'une source de données, telle qu'une base de données locale ou un service Web, en utilisant le pseudo de l'utilisateur comme identifiant pour récupérer les éléments de la liste associée.

Cocher ou décocher un élément

L'activité "ShowListActivity" permet à l'utilisateur de cocher ou décocher un élément de la liste. Cette interaction permet de marquer ou démarquer un élément comme étant complété ou non. Lorsqu'un élément est coché ou décoché, son état est généralement mis à jour dans la source de données pour refléter cette modification.

Ajouter un nouvel élément

L'activité "ShowListActivity" offre également la possibilité d'ajouter un nouvel élément à cette liste. L'utilisateur peut saisir les informations nécessaires pour l'élément, telles que le titre, la description, etc., à partir de l'interface utilisateur. Une fois que l'utilisateur a ajouté un nouvel élément, celui-ci est enregistré dans la source de données appropriée pour cette liste.

Mise à jour de l'interface utilisateur

Lorsque l'utilisateur coche ou décoche un élément ou ajoute un nouvel élément, l'interface utilisateur de l'activité "ShowListActivity" est mise à jour pour refléter ces modifications. Cela peut inclure le rafraîchissement de l'affichage des éléments de la liste pour refléter l'état mis à jour des éléments cochés ou décochés, ainsi que l'affichage du nouvel élément ajouté.

Conclusion

Le projet étudié, réalisé sur Android Studio en utilisant le langage de programmation Kotlin, vise à créer une application mobile de gestion de listes. À travers les différentes activités telles que MainActivity, SettingsActivity, ChoixListActivity et ShowListActivity, l'application permet une gestion locale, qu'on pourra améliorer par la suite en faisant des appels à une API par exemple.

Malheureusement, par manque d'expérience dans les projets Android Studio, nous avons perdu beaucoup de temps sur des problèmes relativement simples (affichage du menu, debuguage des différents crashs) et nous n'avons pas réussi à coder une application fonctionnelle.