DAA – Lab 3

Auteurs :
Thibault Seem
Pascal Perrenoud

Date : 05.12.2022

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD

www.heig-vd.ch

1. Sauvegarde de l'option de tri

Question : Quelle est la meilleure approche pour sauver le choix de l'option de tri de la liste de notes ?

Il n'y a pas vraiment de meilleur choix, mais dans notre cas, nous avons optés pour une LiveData d'Enumération stockée dans la ViewModel qui indiquera quel tri nous souhaitons. Lors d'un choix de tri, nous allons changer cette variable pour, qui est observée par le Fragment contenant la RecyclerView, afin d'indiquer quel tri effectuer.

```
enum class EnumSort{
    SCHEDULE_SORT,
    CREATION_SORT,
    DEFAULT
}

val sortEnum = MutableLiveData<MyViewModel.EnumSort>(EnumSort.DEFAULT)
```

2. Limites de LiveData

Question : L'accès à la liste des notes issues de la base de données Room se fait avec une LiveData. Est-ce que cette solution présente des limites ?

Nous observons cette LiveData dans les fragments. En cas de mise à jour, la liste entière est mise à jour et doit être retriée pour réaffichée, ce qui peut poser des problèmes de performances quand la LiveData notifie les fragments. Pour mitiger ceci, il est possible d'appliquer une transformation « distinctUntilChanged » sur la LiveData afin de n'être notifié que des modifications.

Les LiveData sont également stockés en mémoire. Si notre base de donnée stocke beaucoup d'éléments, cela fera une grande utilisation de la mémoire, ce qui est ni utile ni désirable. Il faudra dans ce cas utiliser un Flow.

3. RecycleView sélectionnable et cliquable

Question : Les notes affichées dans la RecyclerView ne sont pas sélectionnable ni cliquables. Comment procéder pour proposer une interface permettant de sélectionner une note pour l'éditer ?

Il est possible de d'ajouter une détection de click sur un item d'un RecyclerView. Pour ce faire, il faut surcharger la méthode setOnClickListener de la view passée en paramètre de l'inner class ViewHolder, et ceci dans son init. Dans cette surcharge, la définition de la méthode que l'on souhaite utiliser.

Ensuite, il faut, depuis l'activité/fragment gérant la RecyclerView, surcharger la méthode définie dans la surcharge de setOnClickListener afin de faire ce que l'on souhait lors d'un click. Nous nous somme basé sur cette vidéo pour la réponse:
https://www.youtube.com/watch?v=dB9JOsVx-yY