Développement Android
Laboratoire n°2

Les briques logicielles de base
Les *Activités* et les *Fragments*

03.10.2024

Introduction

Ce laboratoire est constitué de plusieurs manipulations destinées à vous familiariser avec le fonctionnement des briques logicielles de base du SDK *Android*.

Manipulations

1. Les Activités

Dans cette première manipulation nous allons étudier le fonctionnement des *Activités* et plus particulièrement la navigation et la communication entre plusieurs *Activités*.

Nous vous demandons de mettre en œuvre deux *Activités*, la première servira à souhaiter la bienvenue à l'utilisateur, et l'invitera à saisir son prénom pour personnaliser le message de bienvenue. La saisie ou la modification du prénom seront réalisées par une seconde *Activité* (cf. Fig. 1), il convient donc de mettre en place le lancement de la seconde *Activité* en passant le prénom en paramètre, ainsi que la réponse, contenant la modification du prénom, lorsque l'on revient sur la première *Activité*.

Pour pouvoir suivre le cycle de vie de chacune des *Activités*, nous vous demandons d'ajouter des sorties dans les logs de l'application permettant de reporter les appels aux méthodes appelées lors des changements d'état (*onCreate*, *onStart*, *onResume*, *onPause*, *onStop* et *onDestroy*).

Veuillez faire attention à respecter les bonnes pratiques en matière de gestion des ressources textuelles.

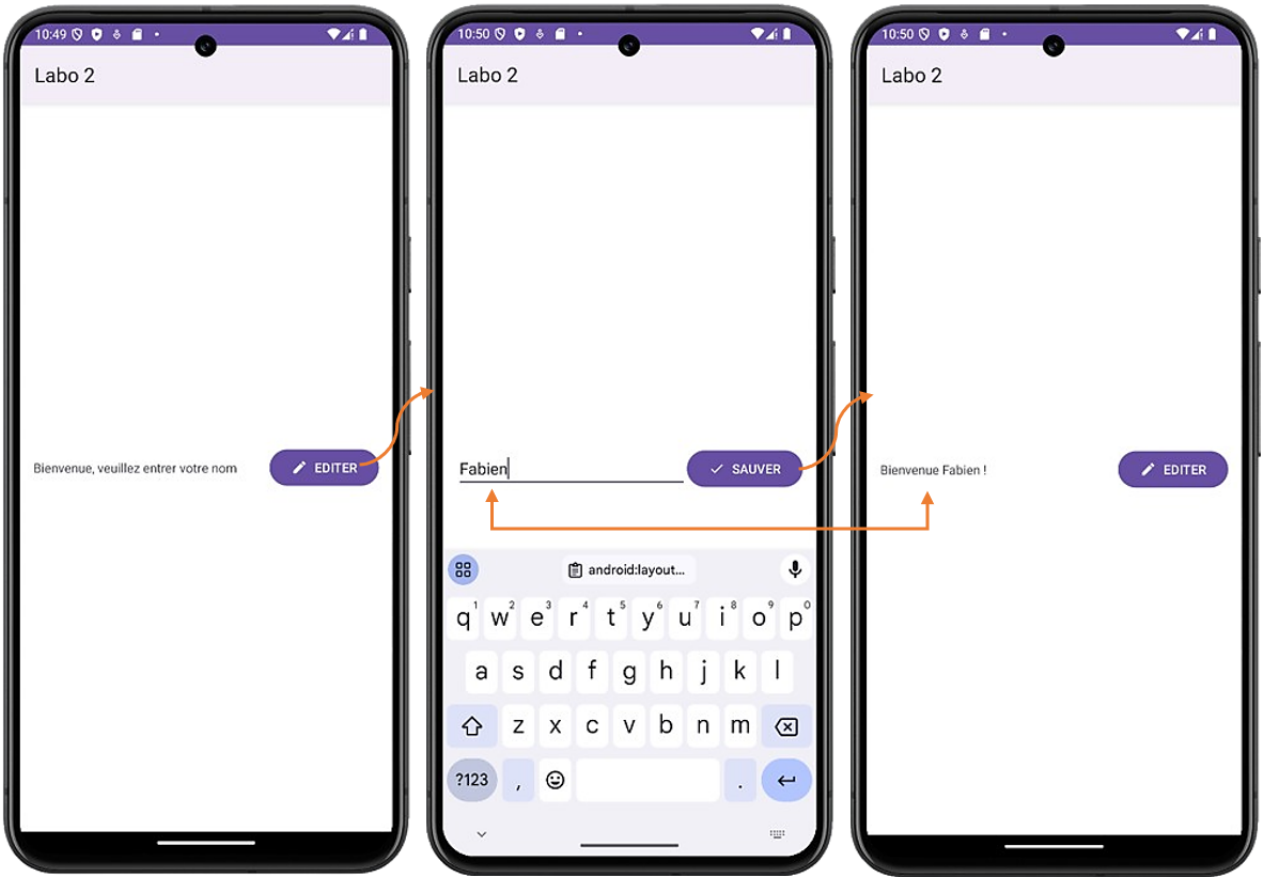

Figure 1 - Echange de données entre les deux Activités

Veuillez discuter de la mise en œuvre de votre solution, en incluant les points suivants :

- Que se passe-t-il si l'utilisateur appuie sur « back » lorsqu'il se trouve sur la seconde *Activité* ?

- Veuillez réaliser un diagramme des changements d'état des deux *Activités* pour les utilisations suivantes, vous mettrez en évidence les différentes *instances* de chaque *Activité* :
  - L'utilisateur ouvre l'application, clique sur le bouton *éditer*, renseigne son prénom et sauve.
  - L'utilisateur ouvre l'application en mode portrait, clique sur le bouton *éditer*, bascule en mode paysage, renseigne son prénom et sauve.

- Que faut-il mettre en place pour que vos *Activités* supportent la rotation de l'écran ? Est-ce nécessaire de le réaliser pour les deux *Activités*, quelle est la différence ?

  Hint : Que se passe-t-il si on bascule la première Activité après avoir saisi son prénom ? Comment peut-on éviter ce comportement indésirable ? Quelle est la différence avec la seconde Activité ?

2. Les Fragments, premiers pas

Nous souhaitons créer une *Activité* regroupant deux *Fragments*, telle que présentée dans la Fig. 2. Nous vous fournissons le code et les Layouts pour ces deux *Fragments*, le premier implémente un simple compteur et le second permet de sélectionner une couleur.

Dans un premier temps, veuillez créer une *Activité*, par exemple nommée *MainActivityFragment1.kt* accompagnés de son propre *Layout*. Vous pouvez ensuite y inclure les deux fragments fournis, *ColorFragment* et *CounterFragment*.

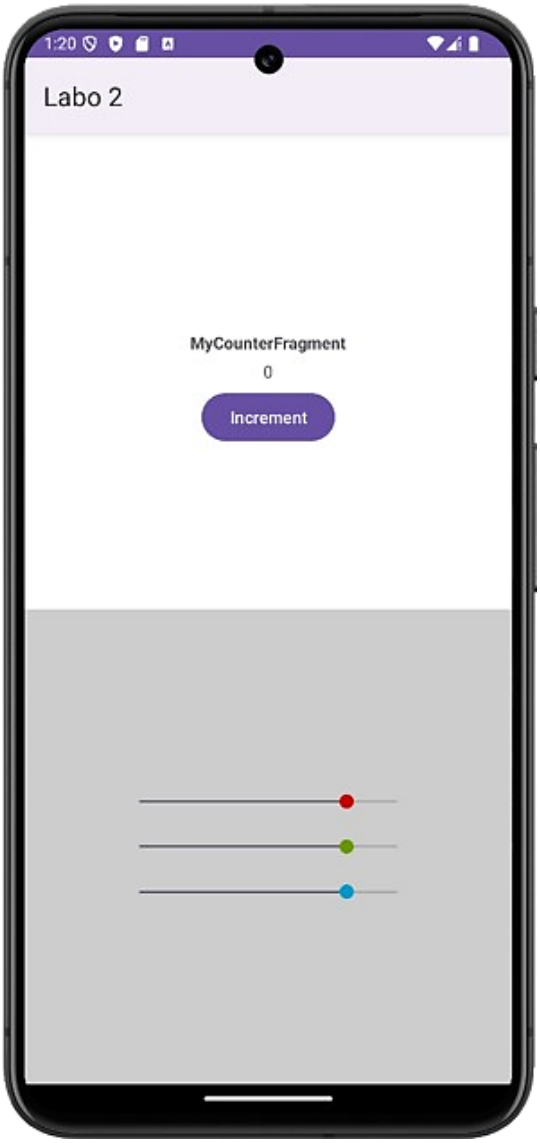

Figure 2 - Activité composée de deux fragments

Hint : Pour les différentes manipulations, vous pouvez définir l'Activité lancée par défaut dans le fichier Manifest.xml

Veuillez discuter de la mise en œuvre de votre solution, en incluant les points suivants :

- Les deux *Fragments* fournis implémentent la restauration de leur état. Si on enlève la sauvegarde de l'état sur le *ColorFragment* sa couleur sera tout de même restaurée, comment pouvons-nous expliquer cela ?
- Si nous plaçons deux fois le *CounterFragment* dans l'*Activité*, nous aurons deux instances indépendantes de celui-ci. Comment est-ce que la restauration de l'état se passe en cas de rotation de l'écran ?

3. Le FragmentManager

Nous souhaitons mettre en œuvre le squelette d'une *Activité* permettant de réaliser une configuration en plusieurs étapes. Dans une unique *Activité* hôte, plusieurs *Fragments* se succéderont, l'utilisateur peut ensuite à l'aide de trois boutons appartenant à l'*Activité* (précédent, fermer et suivant), passer d'un fragment à l'autre. La Fig. 3 montre une possible réalisation de cette *Activité*.

La première étape de cette manipulation est de définir le *Layout* de l'*Activité*, celui-ci doit contenir un *FragmentContainerView* qui accueillera les différents *Fragments*, ainsi que 3 *Boutons*. Il faut ensuite créer la classe implémentant l'*Activité*, par exemple nommée *MainActivityFragment2.kt*.

Vous pourrez ensuite créer les *Fragments* pour les différentes étapes, pour ceux-ci il vous faudra créer le *Layout* ainsi que l'implémentation. Pour cette manipulation, il n'est pas forcément nécessaire de créer plusieurs classes de *Fragments* différentes, vous pouvez en créer une seule que vous pourrez initialiserez à chaque fois avec une valeur numérique représentant une des étapes (cf. Fig. 3).

Vous implémenterez ensuite les actions suivantes pour les trois *Buttons* de l'activité :

- **Précédent**
  On revient au *Fragment* précédent dans la pile, s'il n'y en a pas/plus, on termine l'*Activité*
- **Fermer**
  On termine l'*Activité*
- **Suivant**
  On passe à l'étape suivante, on empile et on affiche le prochain *Fragment*.

Hint : Nous pouvons utiliser le nombre de Fragments présents dans la pile pour déterminer le numéro de l'étape actuelle.

Veuillez discuter de la mise en œuvre de votre solution, en incluant les points suivants :

- A l'initialisation de l'*Activité*, comment peut-on faire en sorte que la première étape s'affiche automatiquement ?
- Comment pouvez-vous faire en sorte que votre implémentation supporte la rotation de l'écran ? Nous nous intéressons en particulier au maintien de l'état de la pile de *Fragments* et de l'étape en cours lors de la rotation.
- Dans une transaction sur le *Fragment*, quelle est la différence entre les méthodes *add* et *replace* ?

Figure 3 - Exemple d'une Activité permettant de naviguer entre plusieurs Fragments

Durée / Evaluation

- 4 périodes
- A rendre le lundi **14.10.2024** à **23h55** au plus tard.
- Pour rendre votre code, nous vous demandons de bien vouloir zipper votre projet Android Studio en veillant à bien **supprimer les dossiers build** (à la racine et dans app/) pour limiter la taille du rendu.
- Vous remettrez également un rapport au format **pdf** en vous fiant au contenu demandé dans les **règles de laboratoire** à savoir : les réponses aux questions posées, une description de l'implémentation de chaque partie en justifiant d'éventuels choix ainsi que la liste des manipulations effectuées avec les résultats associés.
- Merci de rendre votre travail sur *CyberLearn* dans un **zip unique**. N'oubliez pas d'indiquer vos noms dans le code, sur vos réponses et de commenter vos solutions.