Université De Montpellier

TP3

Programmation Mobile

Ingo DIAB
Chamy KACI

Mars 2023

Table des matières

1 Fragment

Pour cet exercice, nous avons une MainActivity ainsi que deux fragments (Saisie et Affichage). La view de MainActivity contient un FragmentContainerView qui est par défaut sur le fragment Saisie. Lorsque nous cliquons sur le bouton "Soumettre" du fragment Saisie, nous effectuons une transaction vers le fragment Affichage en lui passant un bundle contenant toutes les informations nécessaires. Dans le fragment Affichage, nous affichons les informations du bundle.

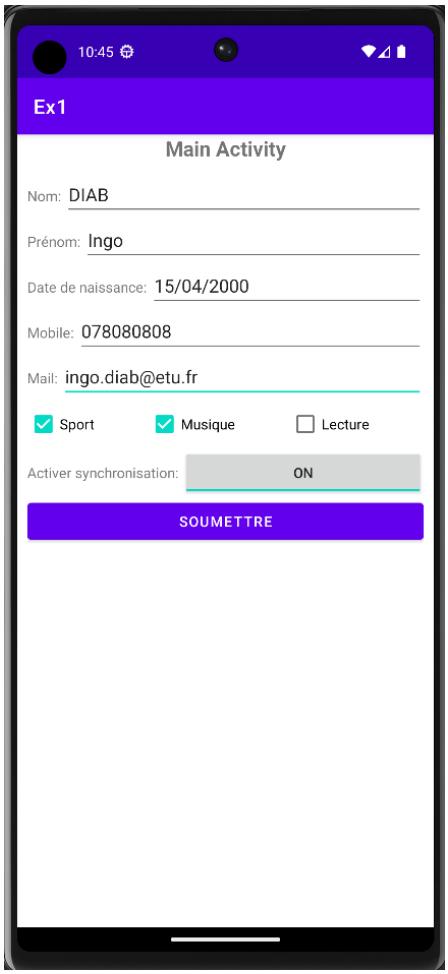

FIGURE 1 – Fragment Saisie

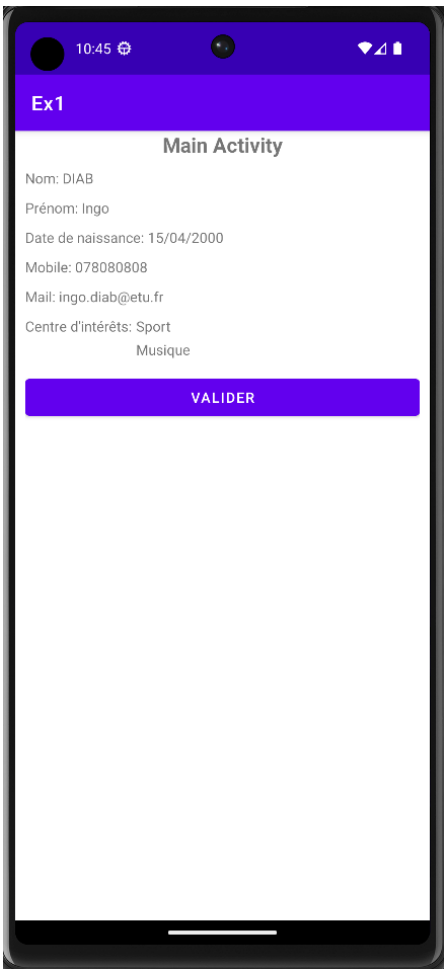

FIGURE 2 – Fragment Affichage

2 Fichier et persistance de l'état d'une activité

Pour cet exercice, nous reprenons les deux fragments du dernier exercice. Nous allons ajouter un évènement OnClick sur le bouton "Valider" du fragment Affichage. Lorsque nous cliquons dessus, l'application va créer un JSONObject. Ce JSONObject est remplit avec les informations de l'utilisateur. Une fois le JSONObject remplit, nous avons une méthode static SaveJson dans la classe FileManager qui nous permet de sauvegarder le JSONObject.

Nous ajoutons aussi un bouton "Retour" au fragment Affichage. Lors de la transaction de Saisie à Affichage, nous ajoutons cette transition à la BackStack. Quand nous cliquons sur le bouton "Retour" du fragment Affichage, nous n'avons plus qu'à faire un "PopBackStack()" pour revenir sur le fragment Saisie. Après le PopBackStack(), le OnResume() du fragment Saisie sera déclenché et, dans cette méthode, nous vidons certains EditText (nous avons décider de vider la date de naissance, le numéro de mobile ainsi que le mail).

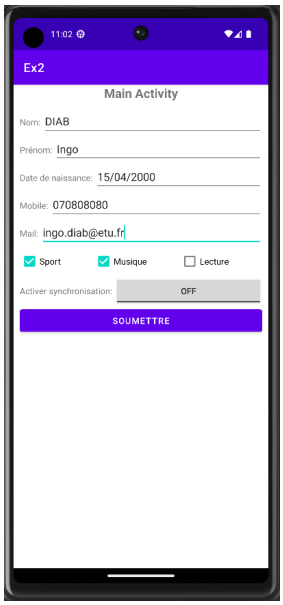

FIGURE 3 – Fragment Saisie

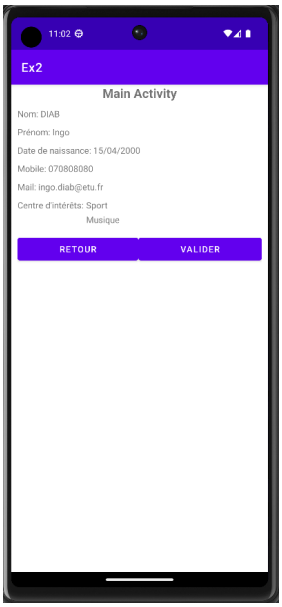

FIGURE 4 – Fragment Affichage

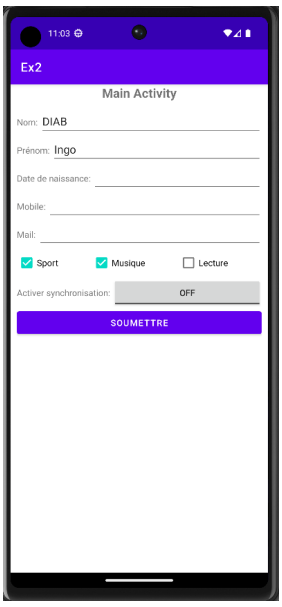

FIGURE 5 – Fragment Saisie après PopBack()

3 Réseau et utilisation de services web

Pour cet exercice, nous reprenons les deux fragments du dernier exercice. Nous allons ajouter un bouton "Télécharger" au fragment Saisie. Nous allons utiliser l'url https ://cat-fact.herokuapp.com/facts afin de récupérer ce fichier Json. Nous allons donc, quand nous cliquons sur le bouton "Télécharger", établir une connexion avec l'url demandée et récupérer le fichier sous forme de stream. Nous allons ensuite copier un certain nombre de char dans un buffer que nous transformerons en String. Nous afficherons ensuite le contenu du Json récupéré dans le fragment Saisie (feedback).

Pour parser le fichier Json qui contient les données de l'utilisateur, nous avons ajouté au FileManager une méthode LoadJson. Cette méthode va chercher un fichier Json à un chemin donné. Il va créer un buffer de la taille de l'objet et le remplir avec le contenu du fichier. Il va ensuite créer un JSONObject à partir du buffer obtenu. Pour s'assurer que notre parser marche, nous avons utiliser cette méthode quand nous cliquons sur le bouton "Valider" du fragment Affichage. En plus de sauvegarder les données dans un Json, le bouton va charger le Json, le parser et afficher le nom contenu dans le fichier Json avec Toast().

FIGURE 6 – Fragment Saisie

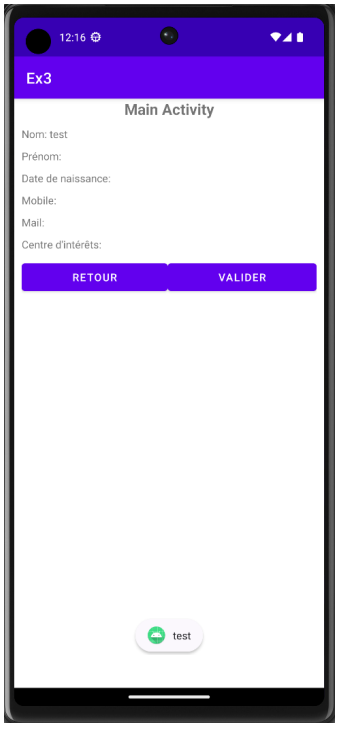

FIGURE 7 – Fragment Affichage

4 Services

Pour cet exercice, nous reprenons les deux fragments du dernier exercice. Nous avons ajouter un IntentService qui sera appelé (startService) lorsque l'utilisateur aura cliqué sur le bouton "Télécharger". Le service va chercher le fichier, s'il existe, il enverra le fichier parsé dans un JSONObject au Receiver créé par le fragment. Le fragment pourra donc accéder au JSONObject contenu dans le Receiver et l'afficher.

Si le fichier n'est pas présent, il sera téléchargé depuis le web (sur un nouveau thread puisque c'est un IntentService). Le fichier sera ensuite parsé dans un JSONObject avant d'être enregistré et envoyé au Receiver. Le fragment pourra donc accéder au JSONObject contenu dans le Receiver et l'afficher.

Un service n'ayant pas d'interface, nous n'avons pas de vue .xml pour notre service.