

Travaux Dirigés et Pratiques de Programmation Android n° 6

Préférences d'une application

Les objectifs de ce TD sont :

- La mise en place d'un fragment de préférences, affiché lorsque l'utilisateur clique sur le bouton correspondant du menu et enlevé lorsqu'il clique sur le bouton de retour ;
 - La réaction de l'application aux choix de préférences de l'utilisateur ;
 - Le chargement des préférences au démarrage de l'application.
-

Pour cela, nous allons ajouter un menu de préférences au TP5, qui déterminera le style du texte.

Mise en place

Reprendre le TP5, et créer au niveau d'AndroidStudio un nouveau fragment, appelé par exemple Reg lages.

Ouverture et fermeture du menu de préférences

- Faire en sorte qu'un fragment de type Reg lages soit attaché (dans le même espace réservé que celui où on met le fragment de « Click the button! ») lorsque l'utilisateur clique sur le bouton Set - tings du menu.

Une fois que l'utilisateur aura choisi ses préférences, il voudra quitter le menu de préférences, c'est-à-dire dans notre cas détacher le fragment de type Reg lages. Cela se fait généralement en appuyant sur le bouton de retour d'Android.

- Tester : ouvrir les préférences, puis cliquer sur le bouton de retour. Qu'observe-t-on ?

Android offre la possibilité de modifier l'action sur le bouton de retour. Lorsque l'utilisateur appuie sur ce bouton, la méthode [void onBackPressed\(\)](#) de la classe [Activity](#) (dont notre activité TP5Activity hérite) est appelée ; il suffit donc de redéfinir cette méthode pour modifier le comportement du bouton de retour.

- Redéfinir cette méthode pour que le fragment de préférences soit détaché lorsque l'utilisateur clique sur le bouton. Tester.
- Faire attention à ce qui se passe lorsque l'utilisateur clique sur le bouton de retour alors que les préférences ne sont pas ouvertes : on peut soit ne rien faire, soit retrouver le comportement précédent, mais il est inutile de chercher à détacher le fragment de préférences.

Plutôt qu'utiliser une transaction pour détacher le fragment de préférences, le gestionnaire de fragments mémorise les transactions dans une structure de pile, ce qui permet de revenir facilement à l'état précédent.

- On peut empiler une transaction grâce à la méthode [FragmentTransaction addToBackStack \(String name\)](#) de la classe [FragmentTransaction](#). Utiliser cette méthode pour empiler la transaction qui vous sert à attacher le fragment de préférences.
- On peut dépiler la dernière transaction grâce à la méthode [void popBackStack \(\)](#) de la classe [FragmentManager](#). Modifier le code de votre méthode `onBackPressed` afin de dépiler la dernière transaction plutôt que d'en créer une nouvelle. Tester.

Menu de préférences

Nous allons maintenant faire de notre fragment un fragment de préférences.

- Faire que la classe Reglages hérite de PreferenceFragment (au lieu de simplement Fragment).

Interface de préférences

- Dans l'arborescence du projet, faire un clic droit sur app puis aller dans New/Android Resource File. Choisir un nom (par exemple, preferences, qui est le nom donné usuellement) et prendre XML comme Resource type.
- Associer cette interface au fragment :
 - o Supprimer la méthode onCreateView de la classe Reglages.
 - o Redéfinir la méthode onCreate de la classe Reglages, de manière à ce qu'elle lance l'affichage du fichier d'interface XML :

```
@Override
public void onCreate(Bundle savedInstanceState) {
    // Appel de la méthode de la super-classe
    super.onCreate(savedInstanceState);
    // Chargement de l'interface XML
    addPreferencesFromResource(R.xml.preferences);
}
```

- Ajouter à l'interface de préférences une case à cocher, en insérant entre les deux balises d'ouverture et de fermeture PreferenceScreen :

```
<CheckBoxPreference
    android:key="text_size"
    android:title="Texte en grand"
    android:defaultValue="false"/>
```

- o android:key contient une chaîne de caractères qui est un identifiant unique de la préférence, servant à la sauvegarder et à la distinguer des autres ;
- o android:title est le sujet de la préférence, qui sera affiché à l'utilisateur ;
- o android:defaultValue est la valeur par défaut de la préférence, la première fois que l'application est lancée (ensuite, on utilise la valeur sauvegardée).
- Pour que le fichier de sauvegarde soit généré lors du premier lancement de l'application, ajouter à la fin de la méthode onCreate de la classe TP5Activity :
PreferenceManager.setDefaultValues(this, R.xml.preferences, false);
- Tester. Cocher la case, puis fermer l'application. Relancer l'application et vérifier que la case est bien restée cochée.

Réaction au changement d'une préférence

Lorsque l'utilisateur modifie une préférence, Android appelle la méthode void onSharedPreferenceChanged(SharedPreferences sharedPreferences, String key) de l'interface SharedPreferences.OnSharedPreferenceChangeListener. Il suffit d'implanter cette interface pour réaliser les actions que l'on souhaite en fonction des préférences.

- Spécifier que la classe Reglages implante l'interface SharedPreferences.OnSharedPreferenceChangeListener.
- Définir la méthode :

```
public void onSharedPreferenceChanged(SharedPreferences sharedPreferences,
String key) {
    if (key.equals("text_size")) {
        // Récupération de la préférence associée à la clé
        CheckBoxPreference pref = (CheckBoxPreference) findPreference(key);
        // Action à effectuer
    }
}
```

```
}  
}
```

Compléter la méthode pour quelle change la taille du texte affichant le niveau en fonction de la préférence. On pourra utiliser la méthode [boolean isChecked\(\)](#) de la classe [CheckBoxPreference](#).

- Lancer l'écouteur (`getPreferencesScreen().getSharedPreferences().registerOnSharedPreferenceChangeListener(this);`) et l'arrêter (`getPreferencesScreen().getSharedPreferences().unregisterOnSharedPreferenceChangeListener(this);`) dans les méthodes `onResume` et `onPause`, réciproquement, de la classe `Reglages` (ne pas oublier d'appeler la méthode de la super-classe). Tester.
- Cocher la case, puis quitter l'application et la relancer. Qu'observe-t-on ?

Chargement des préférences au démarrage d'une application

Ce qu'on vient de faire ne déclenche une action que lorsque l'utilisateur modifie les préférences. Au lancement de l'application, il faut lire les préférences (qui sont déjà sauvegardées) afin de la lancer selon le souhait de l'utilisateur.

La méthode [SharedPreferences getDefaultSharedPreferences \(Context context\)](#) de la classe [PreferenceManager](#) permet de récupérer l'ensemble des préférences, que l'on peut ensuite inspecter pour récupérer des valeurs. Par exemple, dans notre cas, on peut savoir si la case est cochée ainsi :

```
SharedPreferences sharedPref = PreferenceManager.getDefaultSharedPreferences(this);  
boolean b = sharedPref.getBoolean("text_size", false);
```

- Faire en sorte que les préférences soient appliquées au démarrage de l'application. Tester.

Persistance longue

Il serait peut-être temps qu'on puisse conserver nos scores d'un lancement de l'application à l'autre...!

On veut conserver le score et le niveau, c'est-à-dire une association clé/valeur de petite taille, ce pour quoi les préférences partagées sont adaptées.

Sauvegarde lors d'une modification

Lorsque le score et le niveau sont mis à jour :

- [Ouvrir](#) les préférences partagées. (On pourra utiliser comme entier [MODE_PRIVATE](#), car on ne veut pas que d'autres applications puissent lire notre score.)
- [Démarrer un éditeur](#) pour ces préférences.
- [Associer](#) à la clé de votre choix (qui doit être une variable contenant une chaîne de caractères non mutable) le score, puis faire de même avec le niveau.
- [Enregistrer](#) ces modifications.

Chargement du score et du niveau au lancement de l'application

- Charger le score et le niveau au lancement de l'application, en [lisant](#) ces valeurs dans les préférences partagées.
- Tester.

À vous de jouer

1. Ajouter dans les préférences un bouton curseur de type [SwitchPreference](#).
2. Faire en sorte que ce bouton contrôle la couleur du texte donnant le niveau (avec les couleurs de votre choix). On pourra utiliser pour cela la méthode de la classe , prenant en argument un entier étant la représentation « ARGB » (« Alpha » (transparence), « Red », « Green », « Blue ») de la couleur (par exemple, la couleur noire est représentée par 0xFF000000, et la couleur rouge par 0xFFFF0000.).
3. Ajouter dans le menu de l'activité un bouton qui restaure les paramètres par défaut.
4. Faire en sorte que les préférences agissent sur le jeu « Click the button! ». On aura pour cela besoin de faire communiquer deux fragments d'une même activité entre eux : on pourra s'inspirer de ce [tutoriel](#).