## Tri des fichiers

Bouton de lancement : **android:id="@+id/btnDossier".**

Méthode associée au clic du bouton : **public void *openDossierMusic*(View v)** de la classe **AAP.** Cette méthode se contente de lancer l’activity **MusicListActivity** du package fr.unice.aap.musics.

MusicListActivity initialise ensuite le menu des optons de tri des fichiers audio et récupère les fichiers audio supportés présents dans les répertoires de stockage configurés.

L’activity MusicListActivity  comporte également toutes les méthodes communes au tri des fichiers audio de manière à centraliser le rafraîchissement de la liste des musiques à afficher selon l’option de tri sélectionnée.

Une fois que l’utilisateur sélectionne une option de tri (« Toutes les chansons », « Artiste », « Album », « Genre »), MusicListActivity va lancer l’activity correspondante à l’option de tri sélectionnée (respectivement **AllSongsListActivity**, **ArtistsListActivity**, **AlbumsListActivity**, **GenresListActivity**).

Si l’utilisateur clique sur le bouton « retour » de son téléphone, le résultat de l’’activity en cours sera 0 :

@Override

**public** **void** onBackPressed() {

**this**.setResult(0);

**this**.finish();

}

En revanche, s’il sélectionne une musique, il faut retourner au lecteur et donc arrêter les activity en cours (en dehors de l’activity du lecteur, le résultat de l’acvtivty sera donc 1 :

@Override

**protected** **void** onActivityResult(**int** requestCode, **int** resultCode, Intent data) {

**this**.setResult(resultCode);

**if**(resultCode == 1) {

**this**.finish();

}

}

Pour la récupération des métadonnées à afficher (cf Rapport), on utilise un objet de type **android.media.MediaMetadataRetriever**. Cet classe permet de récupérer aisément les métadonnées présentes au sein d’un fichier audio (cf activity **AllSongsListActivity**).

## Configuration des répertoires de stockage

Bouton de lancement : **android:id="@+id/config"**.

Méthode associée au clic du bouton : **public void config(View v)** de la classe **AAP**. Cette méthode affiche une fenêtre de type **AlertDialog** pour proposer une interface affichant la liste des répertoires de stockage sélectionnés et la possibilité d’en ajouter de nouveau.

Si l’utilisateur clique sur le bouton **android:id="@+id/addMusicDirectory"**, la méthode **ajouterDossierMusique(View v)** de la classe **AAP** sera exécutée. Cette dernière se contentera de lancer l’activity **FileExplorerActivity**.

L’activity **FileExplorerActivity**, correspond à une sorte d’explorateur de fichiers présents au sein du téléphone. L’utilistaeur partira alors de la racine, et parcourra les dossiers du téléphone, jusqu’à sélectionner le dossier courant en cliquant sur le bouton **android:id="@+id/btnSelectRepCourant"**.

Si l’utilisateur veut parcourir un dossier mais que ce dernier est inaccessible à la lecture, une fenêtre d’erreur sera affichée. Le code correspondant à cet vérification se trouve dans la méthode **protected void onListItemClick(ListView l, View v, int position, long id)** de l’activity **FileExplorerActivity**.

Lorsque l’utilisateur sélectionne le répertoire courant, le chemin de ce dernier est récupéré et sauvegardé dans les paramètres d’application sous la clé « **music\_files\_paths** ». La récupération et la sauvegarde de paramètres d’application se réalise à l’aide des interfaces **android.content.SharedPreferences** et **android.content.SharedPreferences.Editor** (**Editor** étant une classe interne de la classe **SharedPreferences**).