SOUNDROID

Application mobile réalisée avec Android Studio

RESUME

Projet Android Studio réalisé dans le cadre du cours d'Interfaces Graphiques du cursus d'Ingénieur Informatique à l'ESIPE, Champs sur Marne.

Nathan Lemoine, Chahinaz Dindane et Emilie Marti

Interfaces Graphiques – Michel Chilowicz

Table des matières

1. Introduction

1.1. Description du projet

Dans le cadre du cours d'Interfaces Graphiques de la deuxième année du cursus d'Ingénieur Informatique à l'ESIPE, Marne la Vallée, on a réalisé en trinôme un projet de développement d'une application mobile avec Android Studio. Cette application sera déployée sur des appareil Android, un système d'exploitation open source basé sur le noyau de Linux, destiné principalement aux appareils connectés tels que le smartphone ou les tablettes.

Cette application est un lecteur musical adorné de fonctionnalités supplémentaires.

Ce lecteur est déployé sur un appareil Android (téléphone ou tablette). Il est capable de récupérer les musiques du stockage de l'appareil et permettre à l'utilisateur de les rechercher par un ou plusieurs critères, les écouter et les étiqueter depuis l'application. L'utilisateur peut également créer des listes de reproduction.

Une pause automatique sur critère de batterie est proposée : en effet, si la batterie de l'appareil est basse, la musique qui était en train de se jouer se met en pause pour préserver la batterie.

L'export de la base est possible depuis la fenêtre des paramètres.

2. Environnement de travail

2.1. Android Studio

L'outils utilisé pour réaliser les applications Android est Android Studio. Il s'agit d'un IDE semblable à IntelliJ et PyCharm mais permet la création d'interfaces graphiques facilement par des drag-and-drop de composants qui se traduisent automatiquement en fichiers xml.

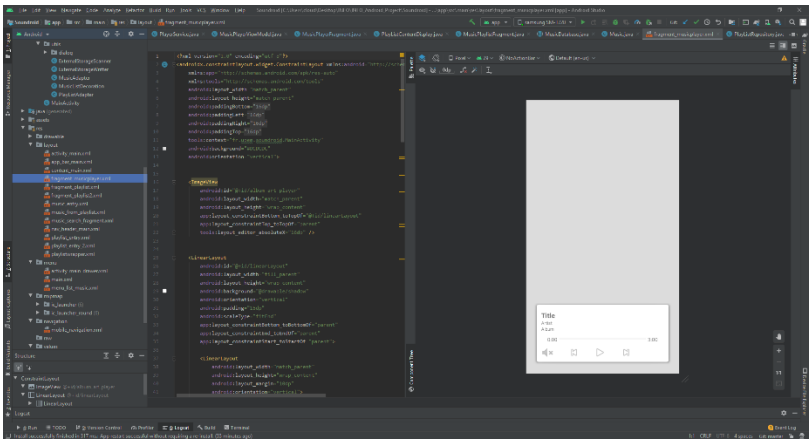


Figure 2.1. Android Studio: Edition d'un fragment

Android Studio permet également le déploiement automatique de l'application sur un appareil Android connecté (ou sur une machine virtuelle android avec le AVD manager) et la récupération des logs de l'application pour le test et le débug.

2.2. Langages de programmation

2.2.1. Java

Le langage de programmation choisi pour ce projet est le Java. Il s'agit du langage le plus répandu pour le développement d'applications Android (même si Kotlin devient de plus en plus populaire) et il s'agit d'un langage déjà connu et étudié dans le cadre de notre cursus. Deux personnes sur trois étant néophytes sur le développement Android, il paraissait plus raisonnable de développer avec un langage qu'on maîtrisait déjà.

2.2.2. XML

XML est le langage qui sert à décrire les interfaces graphiques et la position de chaque élément. Néanmoins, Android Studio génère automatiquement du code XML lorsqu'on crée une interface en utilisant la fonction de drag-and-drop de l'IDE.

3. Architecture et conception

3.1. Architecture générale

Le projet est divisé en trois grandes parties :

3.1.1. Base de données

package fr.upem.soundroid.dataBaseComponents

Les métadonnées des musiques récupérées sur l'appareil sont sauvegardées dans une base de données propre à l'application Soundroid. Les métadonnées des musiques sont récupérées à partir de *MetaDataRetriever*.

Pour créer et gérer la base de données, on utilise la librairie *ROOM* qui accepte les requêtes SQLite. Sa création se fait dans la classe *model.MusicDatabase*.

Chaque type de structure sauvegardée en base est représentée par la classe de son nom dans le package model, et ce sont les objets qui seront instanciés et utilisés dans l'application. Les classes qui terminent par DAO (*MusicDAO*, *PlayListDAO* et *TagDAO*) sont les classes qui permettent de faire des requêtes au niveau de la base de données.

Dans le package *providers*, on a les classes qui exposent les méthodes de gestion de la base de données au reste du projet, notamment pour les classes qui gèrent l'interface graphique. Plus précisément, les classes de type *ViewModel* wrappent les fonctions implémentées dans les classes *Repository*.

3.1.2. Service

package fr.upem.soundroid.service

Le service de lecture des musiques est implémenté dans la classe *PlayerService* de ce package. Un service est un composant de l'application qui peut réaliser des opérations même lorsque l'application est en arrière-plan ; dans ce cas, ce qu'il nous intéresse c'est de pouvoir continuer à écouter la musique même si on utilise l'appareil pour faire autre chose.

3.1.3. Interface utilisateur

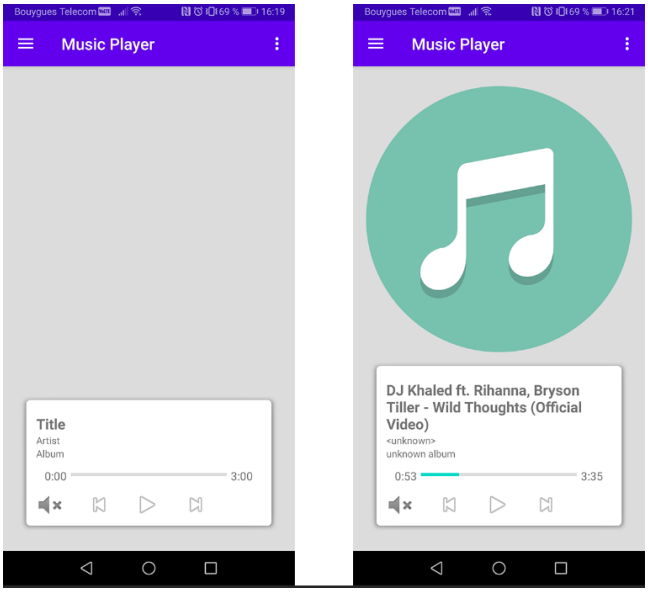
package fr.upem.soundroid.ui

L'interface est gérée par les classes de ce package et celles du package *utils*. Chaque fenêtre de l'application est gérée dans un fragment, qui est un composant qui peut être combiné avec d'autres fragments dans la même activité. Chaque fragment a son propre cycle de vie et gère ses propres inputs.

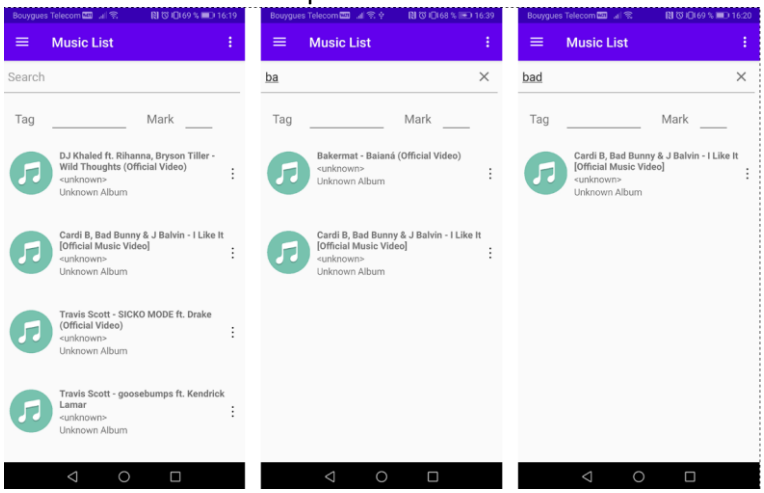
L'organisation visuelle de l'interface graphique est gérée par des fichiers de type xml. On les retrouve dans le dossier *res/layout*. Le premier fragment affiché lorsque l'application est lancée est le lecteur de musique, géré par la classe *MusicPlayerFragment*. Un menu de type slider horizontal permet de naviguer à travers les différents fragments librement.

3.2. Parties de l'application

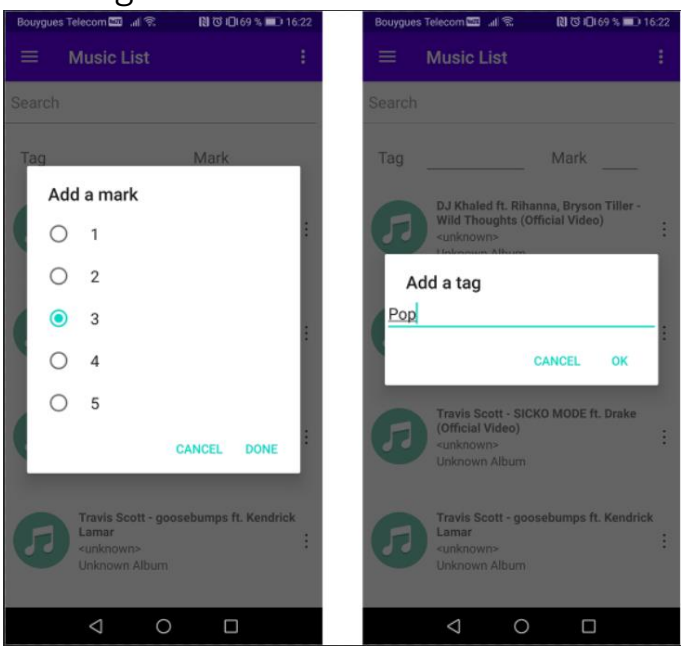
3.2.1. Lecteur musicale



3.2.2. Recherche de musique

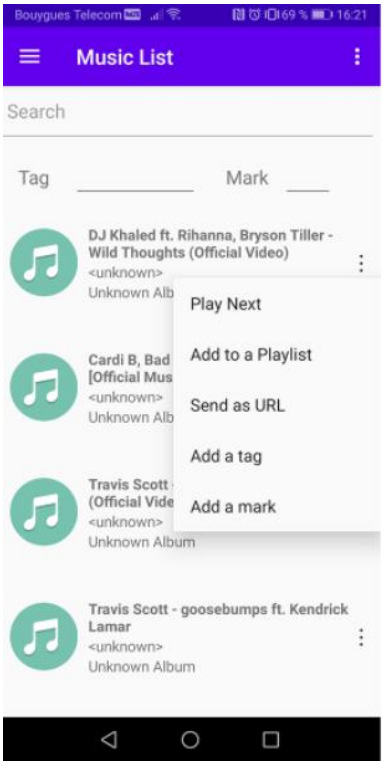


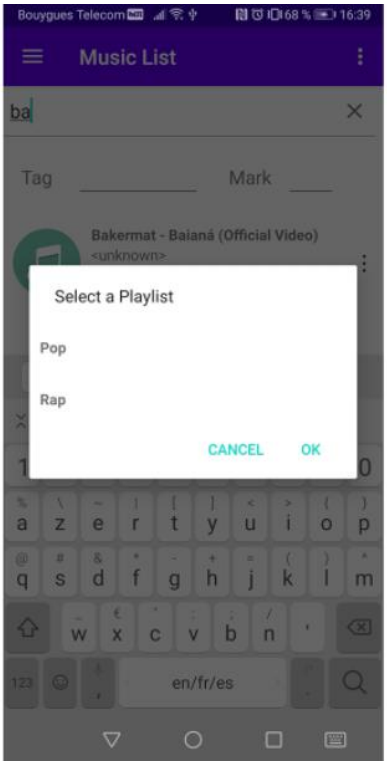
3.2.3. Ajout de tag

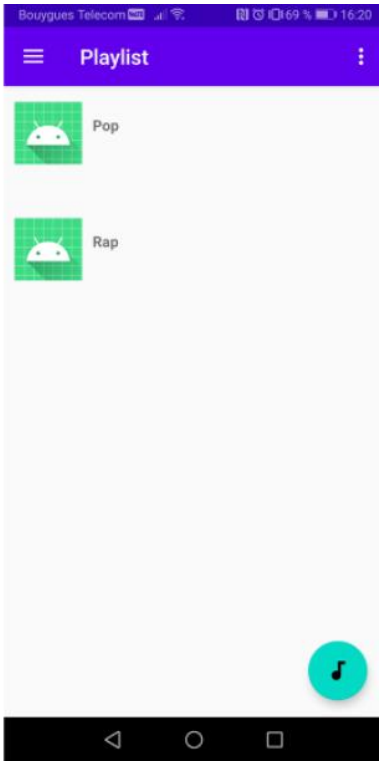


3.2.4. Listes de reproductions

La création de listes de reproductions est possible depuis un menu à trois points qu'on peut dérouler depuis la liste des musiques.







4. Organisation du projet

4.1. Gestionnaire de versions

Le projet a été géré sous le gestionnaire de versions Git, plus précisément GitHub. Git permet aux développeurs de tenir un suivi sur un projet. Un commit correspond à une « snapshot » du projet, c'est-à-dire, une version à laquelle le développeur peut revenir si besoin. Il permet le travail collaboratif.

La version du dépôt distant peut être récupérée par tous les collaborateurs du projet. Pour assurer une bonne intégration continue, tout a été fait sur la branche master et les commit / push ont été faits régulièrement à chaque étape du projet.

4.2. Difficultés rencontrées

A cause de la crise du covid-19, tout le monde a été confiné chez soi pendant toute la durée de ce semestre. De ce fait, la communication était exclusivement à distance et le travail en trinôme n'a pas été aussi évident que si on avait pu se réunir au sein de l'école. La solution était de faire des comptes rendus du travail personnel de chacun mis en ligne et surtout d'assurer l'intégration continue, afin d'éviter de partir sur plusieurs architectures différentes.

Aussi, ayant eu des problèmes techniques sur l'ordinateur d'un des membres de l'équipe, une grosse partie du développement a été réalisé sur le même PC. La gestion sous git a été donc faite au nom de la même personne.

Conclusion

Ce projet a été développé sur l'espace du dernier mois de la séquence académique. Cependant, le mois n'aura pu être consacré entièrement à ce projet et encore moins à la fin, ayant d'autres projets à réaliser pour les autres matières.

La plus grande difficulté résidait sur le fait que pas tous les composants de l'équipe avaient atteint le niveau nécessaire pour faire un projet de telle envergure et les fonctionnalités annexes rajoutaient ont beaucoup complexifié le développement (par rapport à une version « de base » d'un lecteur musicale).

On y est néanmoins arrivé, en développant le premier niveau ainsi que des fonctionnalités diverses entre le deuxième et le troisième niveau.