MarioKartIRL : Instructions pour la création de l'application en version sans RenderScript

Mathis RIGAUD
Département Sciences du Numérique - 2SN-T

Année 2025
Stage de 2A

Table des matières

1 Introduction

Ce rapport est destiné à donner des instructions et des indications pour recréer l'application MarioKartIRL sur les téléphones Xiaomi Redmi Note 9. L'application à laquelle ce rapport s'intéresse est l'application originale donnée par Katia Jaffres-Runser au début du stage, amputée de la partie RenderScript (détails en section 3). Cette application tourne avec :

— *compileSdk* = 34
— *minSdkVersion* = 19, 20 ou 21
— *targetSdkVersion* = 34
— *ndkVersion* = 19, 20 ou 21

compileSdk et *targetSdkVersion* peuvent être augmentés jusqu'à 36 (dernière version du SDK - Android 16 ; testé uniquement avec *ndkVersion* 19 et *minSdkVersion* 19 ou 21).

2 Instructions pour la création de l'application

2.1 Compilation

Il est en premier lieu nécessaire d'installer le NDK correspondant à la version utilisée pour le SDK Parrot qui vient d'être compilé. Cela se fait en allant dans `Tools > SDK Manager > SDK Tools`, puis en cochant la case `Show Package Details` et en sélectionnant dans `NDK (Side by side)` la version du NDK voulue.

L'application utilise le SDK Parrot pour la connexion au drone. Le SDK utilisé est une version modifiée de celui de Parrot, qui est fournie par Katia Jaffres-Runser. Il existe plusieurs versions (en fonction de la version du NDK et de la version du SDK choisies). Pour construire le SDK, il suffit de se placer dans le dossier du SDK choisi, puis de lancer la commande :

```
./build.sh -p arsdk-android -t build-jni -v
```

Après compilation, il est possiblle d'utiliser le SDK compilé dans l'application qui en a besoin.

2.2 Intégration dans l'application

L'application fournie est déjà configurée pour l'intégration des bibliothèques dans le projet.

2.3 Modifications dans l'application

Une fois le SDK intégré dans l'application, quelques modifications dans les fichiers du projet doivent être effectuées pour que l'application fonctionne correctement.

2.3.1 build.gradle(project)

Dans le `build.gradle` global du projet, il faut modifier la version utilisée du SDK Parrot en modifiant la ligne du chemin du SDK `buildDir`. Cette modification permet à Gradle à la fois de localiser les bibliothèques mais également d'utiliser la bonne version de ces dernières. Ensuite, il faut modifier le bloc `ext` afin de bien utiliser les bonnes versions du SDK :

- `compileSdkVersion = 34`
- `minSdkVersion = 21`
- `targetSdkVersion = 34`

2.3.2 manifest.xml

Ensuite, il convient de noter que la connexion du téléphone au drone en passant par l'application dépend du service `ARDiscoveryService`. Ce service ne peut pas être appelé par l'application à moins d'être renseigné dans son `manifest`. Ainsi, il faut ajouter ces lignes au `manifest` de l'application afin de pouvoir utiliser `ARDisoveryService` :

```
<service android:name="com.parrot.arsdk.ardiscovery
.ARDiscoveryService"
    tools:ignore="MissingClass"> </service>
```

2.3.3 ARSALPrint.java

In fine, il est possible, si besoin en est, de modifier le fichier `ARSALPrint.java` de la librairie `libARSAL` (localisation : `libARSAL.java.com.parrot.arsdk.arsal.ARSALPrint`), afin d'afficher les logs donnés par ARSAL dans les logs Android. En effet, les logs ARSAL ne s'affichent pas systématiquement, et il est alors nécessaire d'ajouter dans les méthodes de `ARSAL` une ligne pour que ces méthodes appelles les méthodes `Log` fournies par Android pour afficher les logs dans Logcat.

3 Instructions pour retirer RenderScript

RenderScript (RS) est déprécié et ne sera bientôt plus utilisable sur Android avec l'arrivée des futures versions. L'application utilise RS pour convertir le flux vidéo arrivant du drone Parrot de l'encodage ARGB vers l'encodage NV21 afin d'utiliser la librairie ARToolkit, qui ne supporte que l'encodage NV21. Cela permet de gagner en performances pour la conversion car RS permet de gérer les accès au GPU du téléphone pour augmenter la rapidité des calculs.
Les téléphones actuels (Xiaomi Redmi Note 9) sont cependant suffisamment puissants pour ne pas avoir besoin de cette accélération ; il est donc possible de supprimer RS de l'application sans impact majeur sur les performances de cette dernière.

Pour supprimer RS, il suffit de supprimer toutes les références à RS dans le `build.gradle(:app)`, le `GUIGame.java` (importation des bilbiothèques RS et liaison avec `DetectionTask.java`) et le `DetectionTask.java` (importation des bibliothèques RS, création des variables RS, et préparation et lancement du script RS).
Si encore présent, il est possible de supprimer également le script RS présent dans le dossier dédié à l'API nommé `rs`.

4 Conclusion

Une fois ces étapes réalisées, l'application devrait fonctionner normalement sur les téléphones Xiaomi Redmi Note 9.