REPUBLIQUE DU CAMEROUN
Paix - Travail - Patrie
UNIVERSITE DE NGAOUNDERE

REPUBLIC OF CAMEROON
Peace - Work – Fatherland
THE UNIVERSITY OFNGAOUNDERE

FACULTE DES SCIENCES
BP. 454 NGAOUNDERE

FACULTY OF SCIENCE
P.O. Box 454 NGAOUNDERE

DEPARTEMENT DE MATHEMATIQUES ET INFORMATIQUE
DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE

UE : Système d'Exploitation Mobile

Travaux Pratique

NOMS DES MEMBRES DU GROUPE

YIMBOU WAKAM Fabiola.....16B143fs
VARMANTCHAONALA MOUDINA Charles 16B015fs
NGUELEODAI REVED MAINA.....16A020fs
NSANGOU PARE ABDERRAHIM 16A028fs
FEUNKEU KUATE FORSTER DE LAFARGES 16B375fs
KOUANOU NATHANAEL.....16A009fs
SAVIMBI ALEMEGUE Jonas 16A030fs
MAHAMAT SAID ABDOU.....16B147fs
DJENDANG DJENONKAR Fabien 15B282fs
ABDEL-HADI YOUSSOUF ABDOULAYE 16A022fs
NGUEMA Ferdinand 18A002fs

Enseignant : Dr. Ing. Franklin Tchakounté

Année Académique :2018-2019

Travaux Pratiques INF345
Fiche Proposée par Dr Franklin Tchakounté
Octobre 16, 2018

1 Sujet 1

- Télécharger une application de votre choix
- La décompiler
- Disséquer le contenu du fichier manifeste et y ressortir toutes les composantes telles que vues en cours

2 Sujet 2

- Télécharger une application Android de votre choix
- La décompiler
- Retrouver le code source et les classes de cette application
- La recompiler pour avoir l'application initiale

3 Sujet 3

- Simuler l'intercommunication (implicite et explicite) entre deux applications Android via les intents.

Détails :

- Délai de dépôt sur GitHub : 25 Novembre 2018
- Travail en groupe de 5 étudiants avec pour responsable une fille.
- Chaque TP doit être accompagné d'un rapport contenant trois parties : le titre, les objectifs, la méthode à utiliser, les résultats obtenus ainsi que la bibliographie.

SOLUTION TP 1

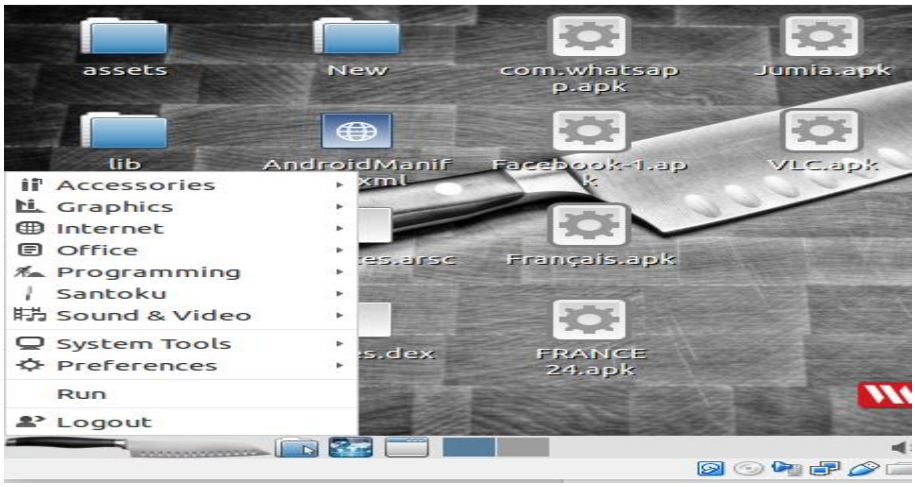
Titre : Décompilation et description d'une application androïde

Objectif :

L'objectif de cette exercice est d'amener l'étudiant à retrouver le code source de l'application afin de comprendre quel sont les différentes parties d'une application.

LA METHODE:

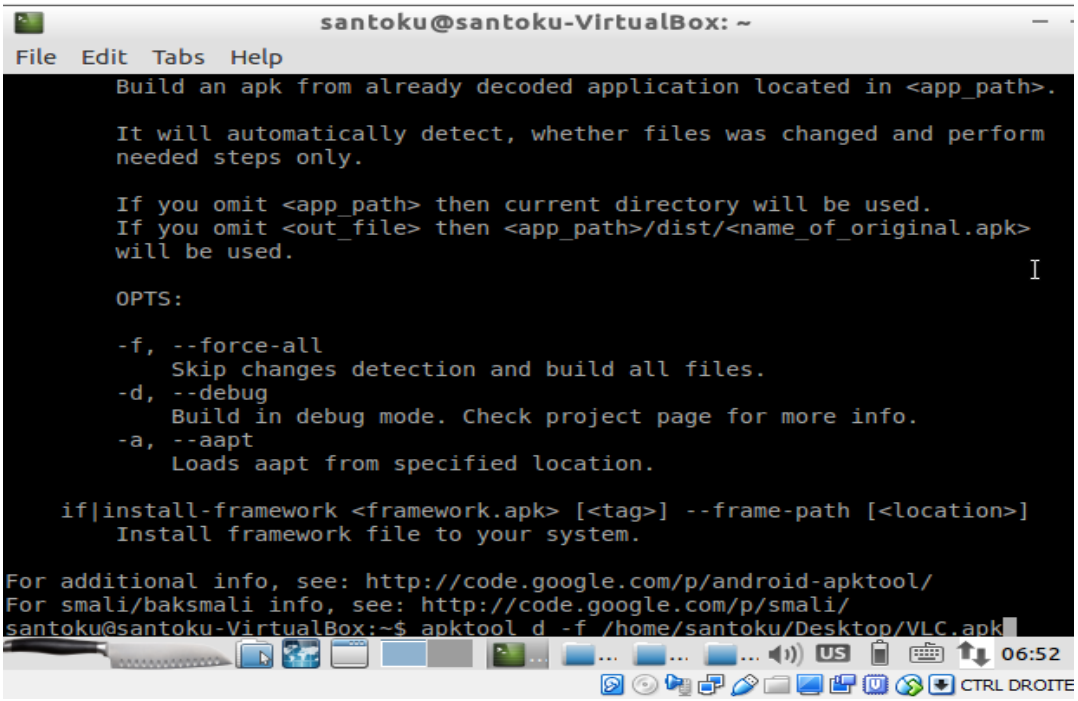
1- Premièrement, nous devons télécharger une application androïde (un apk bien sûr) sur le Play store.
2- Nous devons installer un environnement de développement Android par exemple, ici précisément santoku en virtuel, qui intègre l'environnement de développement Android (Android studio et d'autre).
3- En suite sous santoku on suit la procédure suivante :
   - On part au menu santoku



   - On choisit **reverse engineering**
   - En suit on choisit l'outil APKTOOL

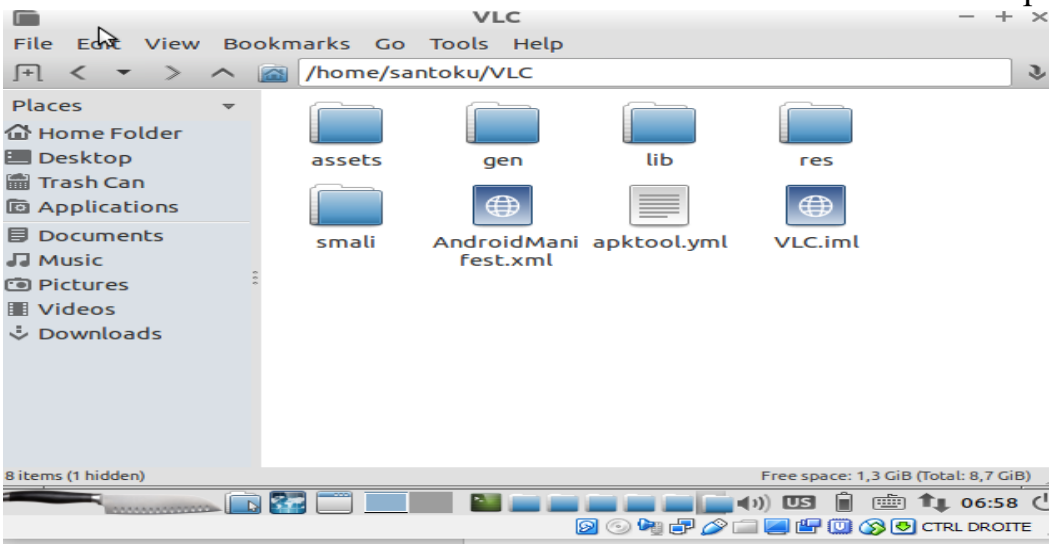
-Là s'ouvre un terminal

-On tape sur ce terminal la commande :apktool d -f /chemin_d'accès/fichier.apk



- On valide la commande.

- En fin l'outil **apktool** décompile l'application et nous génère les codes sources de l'apk.

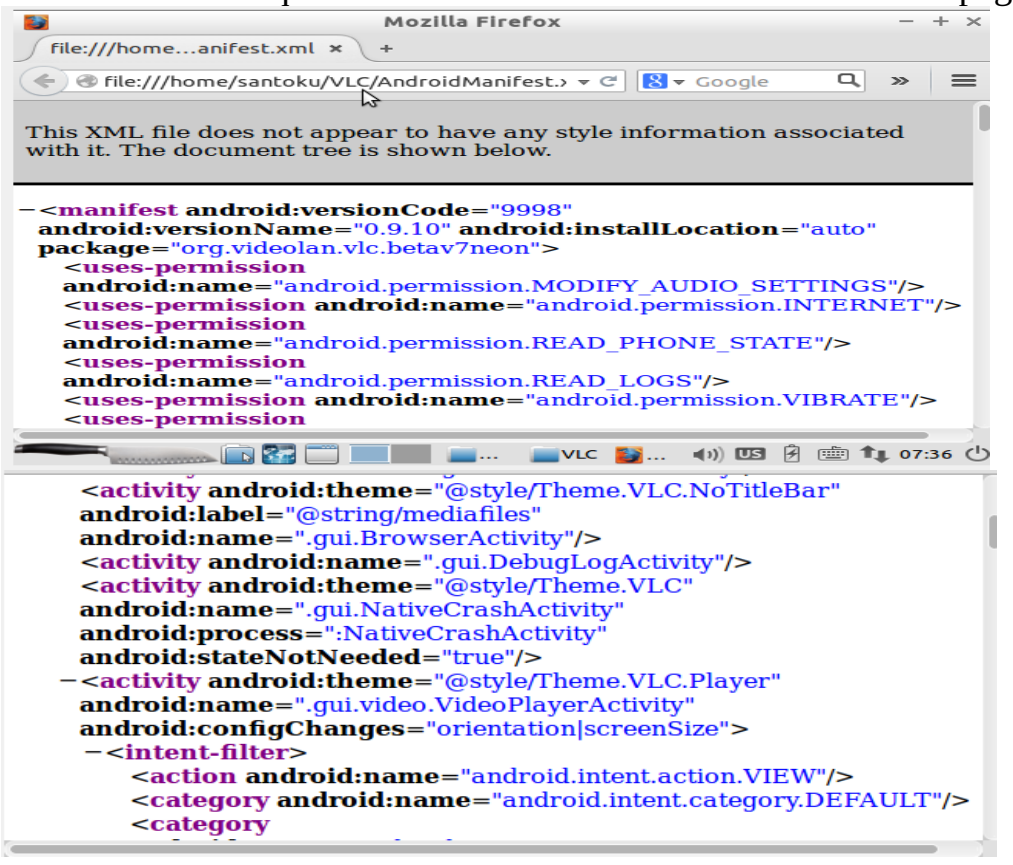


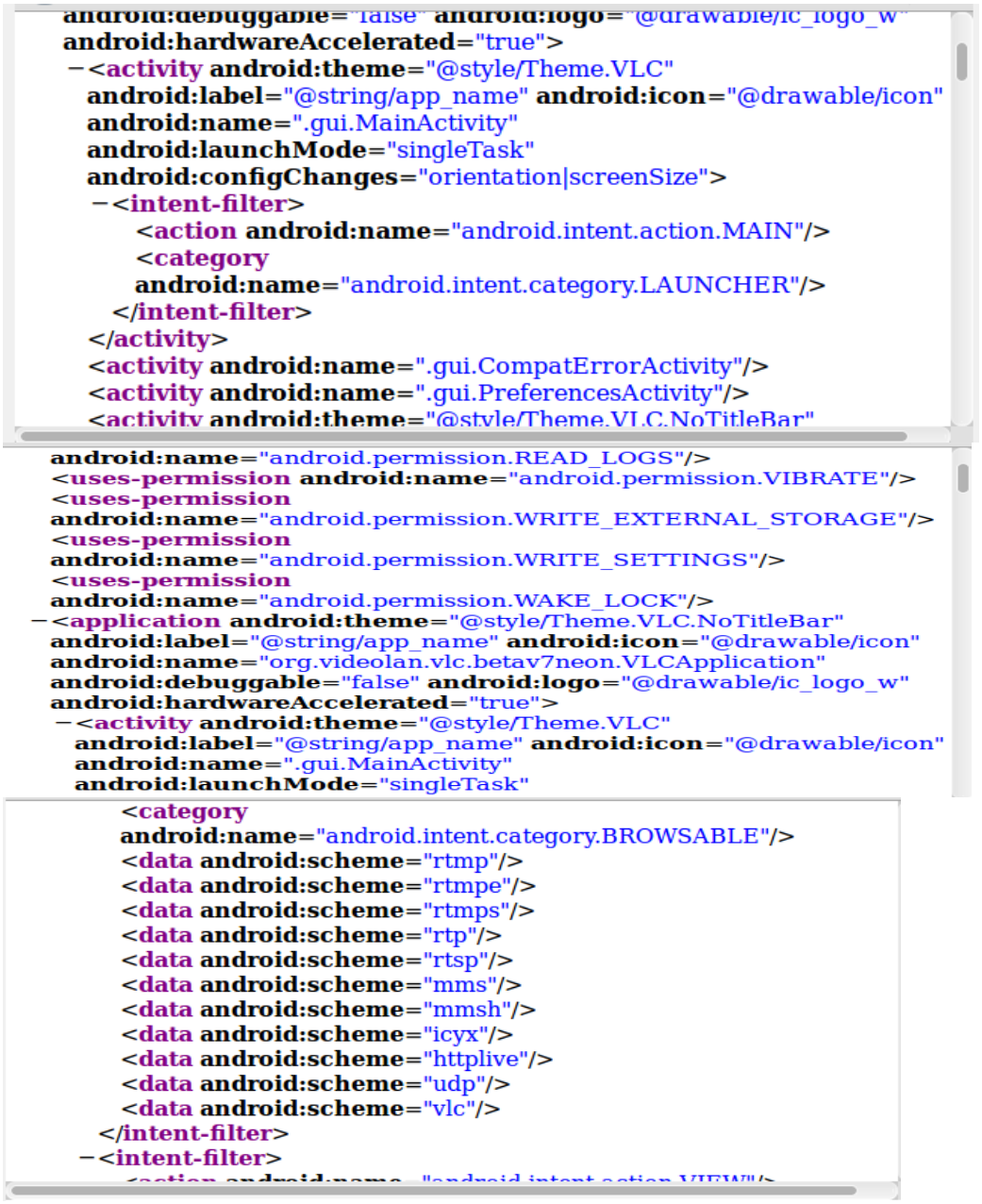
Dans ce dossier VLC, correspondant au nom VLC.apk de l'application même, nous retrouvons visiblement le fichier AndroidMainesfest.xml qu'on recherche pour sortir les différentes composantes de l'application

Description du fichier AndroidManifest.xml

Nous allons commencer par ouvrir le contenu du fichier AndroidManifest.xml

En doubler cliquant sur le fichier on obtient la page suivante :





Nous retrouvons dans ce contenu des codes sources encadrés par des balises :

Les différentes balises et le rôle de son contenu :

1- **<manifest android :versionCode= ‘’9998 ‘’**
… >

</manifest>

Cette composante indique dans quel package se trouve notre application – dans notre cas il indique : **package= ‘’org.videolan.vlc.betav7neon’’,** nous permettant d’utiliser directement les classes de ce package sans repréciser encore où se situe ces classes.

Elle indique également quelle est la version actuelle de l'application. **Android :versionCode=''9998'' android :versionName= ''0.9.10''** ici le premier attribut, son contenu sera considéré seulement par le Play store et le deuxième attribut est la version de l'utilisateur car il peut voir la version indiquée. Donc en bref nous pouvons dire que cette balise définit les propriétés de l'application.

2- **<uses-permission**

android :name=''android.permission.Type_De_Permission''/>

Cette balise permet de définir les permissions c'est-à-dire elle permet de limiter ou laisser l'accès des applications aux composantes des unes aux autres. Ainsi, si notre application a la permission d'accéder à l'internet nous pouvons indiquer dans notre application la permission suivante : **<uses-permission android :name=''android.permission.INTERNET''/>**

3- **<application android:theme="@style/Theme.VLC.NoTitleBar" android:label="@string/app_name" android:icon="@drawable/icon" android:name="org.videolan.vlc.betav7neon.VLCApplication" android:debuggable="false" android:logo="@drawable/ic_logo_w" android:hardwareAccelerated="true">…</application>**

Cette composante permet définir les caractéristiques notre application à savoir le thème, l'icône, le logo, nom de l'application.

4- **<activity android:theme="@style/Theme.VLC" android: label="@string/app_name" android:icon="@drawable/icon" android:name=".gui.MainActivity" android:launchMode="singleTask" android:configChanges="orientation|screenSize">**

Cette composante permet définir l'activité principale ou les autres activités de notre application. Une activité a plusieurs propriétés comme le nom, le thème, l'icône, … définies à l'aide des attribues de cette balise **activity.**

5- **<intent-filter>**
<action android:name="android.intent.action.MAIN" />
<category
android:name="android.intent.category.LAUNCHER" />
</intent-filter>

Etant donné que les applications androïdes communiquent entre eux et avec les multitudes de messages qui circulent entre les applications, chaque application doit filtrer les messages afin de ne recevoir que les messages qui la concerne exclusivement : c'est le rôle de Intent-filters. Ces composantes permettent d'indiquer, seulement, les messages qui sont perceptibles par une application.

6- **<data android:…="Données" />**
Cette composante permet d'indiquer les données de la basse de données utilisée au sein de l'application.

7- **<service android:name="org.videolan.vlc.betav7neon.audio.AudioService" />**
Cette composante permet d'exécuter des taches d'une application en arrière-plan d'une manière invisible.

8- **<receiver android:name=".PhoneStateReceiver">**
</receiver>
Les receveurs sont les composantes qui permettent aux applications de recevoir les messages qui circulent entre les applications afin de les passer ensuite au filtre afin de les trier. A l'intérieur de ces récepteurs se trouve les filtres.

SOLUTION TP 2

TITRE : LA DECOMPILATION ET LA RECOMPILATION D'UNE APPLICATION ANDROID.

OBJECTIF : L'objectif de ce TP est d'amener l'étudiant à être capable de retrouver le code source d'une application Android afin de le connaitre, le maitriser et même le modifier et en suite être à mesure de reconstruire l'application pour qu'elle soit de nouveau fonctionnelle comme au départ.

METHODE :

NB : Nous dans l'environnement Santoku !

Ici nous allons vous décrire la procédure à suivre pour décompiler, modifier et recompiler un apk.

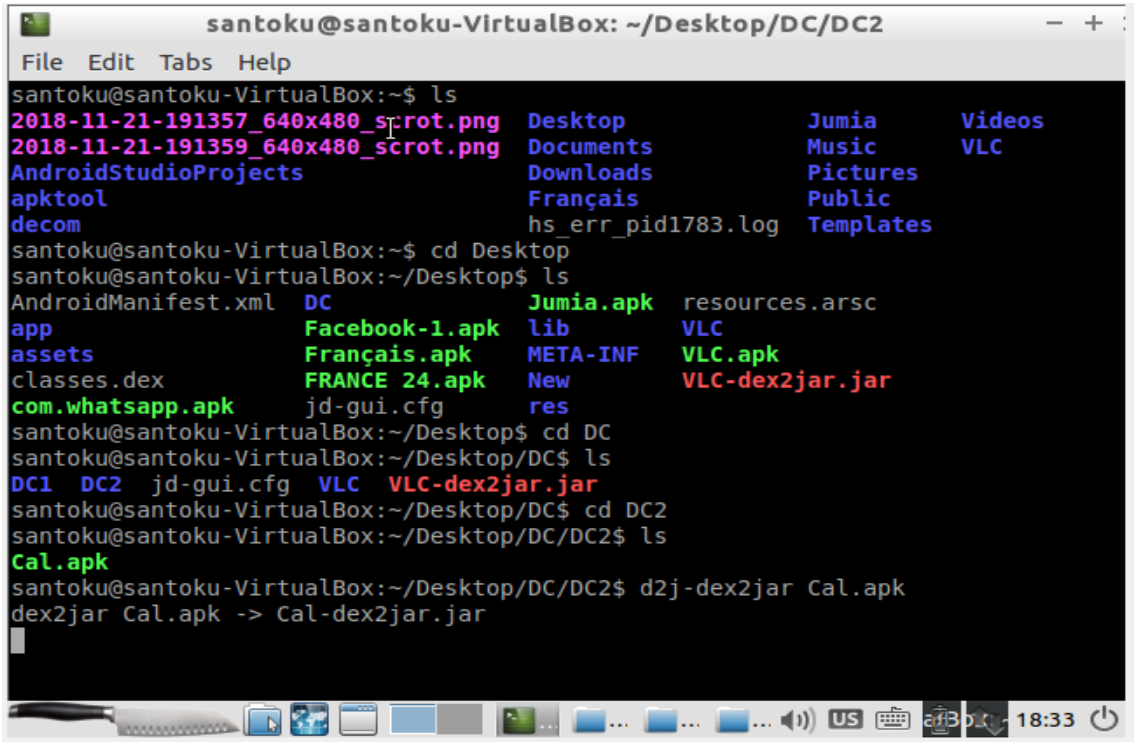
Dans ce TP nous aurons deux parties : la première partie consistera à décompiler l'application afin d'accéder à toutes les classes qui le constituent. Et

la deuxième partie consistera à accéder aux code sources afin de les modifier et enfin les recompiler.

PREMIERE PARTIE : DECOMPILATION ET L'ACCES AUX DIFFERENTES CLASSES.

NB : Dans cette partie nous utiliserons l'outil **: d2j-dex2jar et l'apk a décompiler sera Cal.apk**

1- Nous ouvrons le terminal et nous tapons la commande suivante : (l'avant dernière ligne)

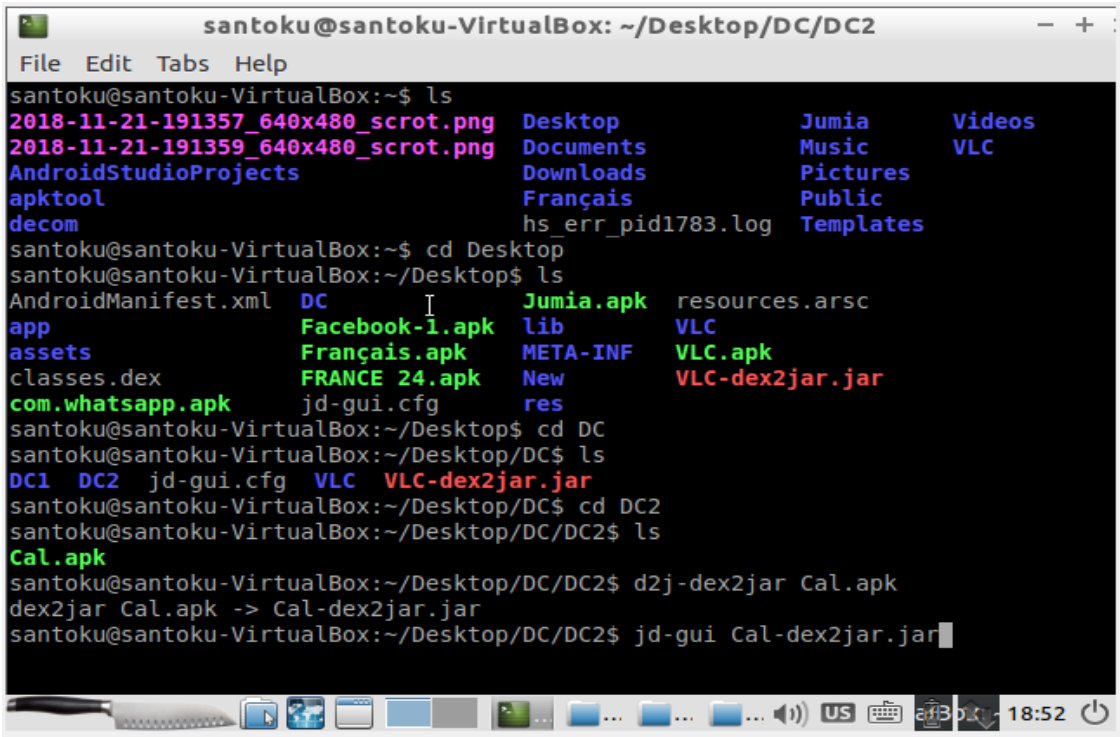


Et nous obtenons dans la console la ligne suivante : dex2jar Cal.apk -> Cal-dex2jar.jar

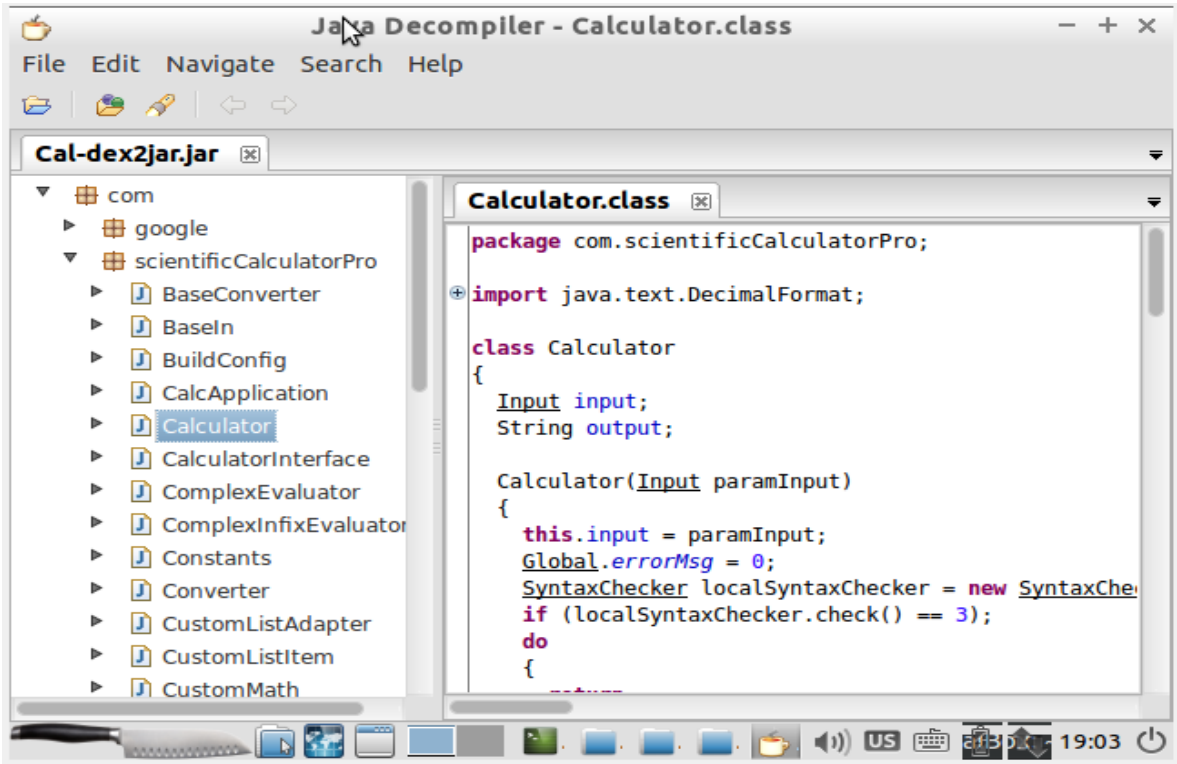
Cette ligne signifie que le fichier d'extension .apk est transformé en un fichier d'extension .jar.

2- Ce dernier fichier d'extension .jar nous allons le décortiquer afin d'accéder aux différentes classes en le passant au Décompileur Java (jd en anglais).

Pour cela nous allons taper la commande suivante :(dernière ligne)



Et nous obtenons, à l'aide du jd (Java Decompiler), la fenêtre suivante à l'intérieur de laquelle nous avons toutes les classes de l'application ouvertes :

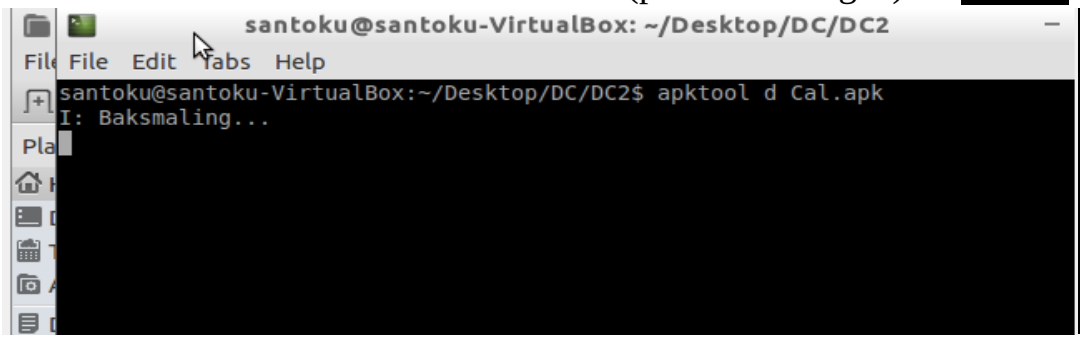


Dans cette fenêtre nous avons toutes les classes et nous pouvons défiler pour retrouver une classe et l'ouvrir pour voir son contenu. Ici nous avons ouvert seulement la classe Calculator.class et nous pouvons ouvrir toutes les autres.

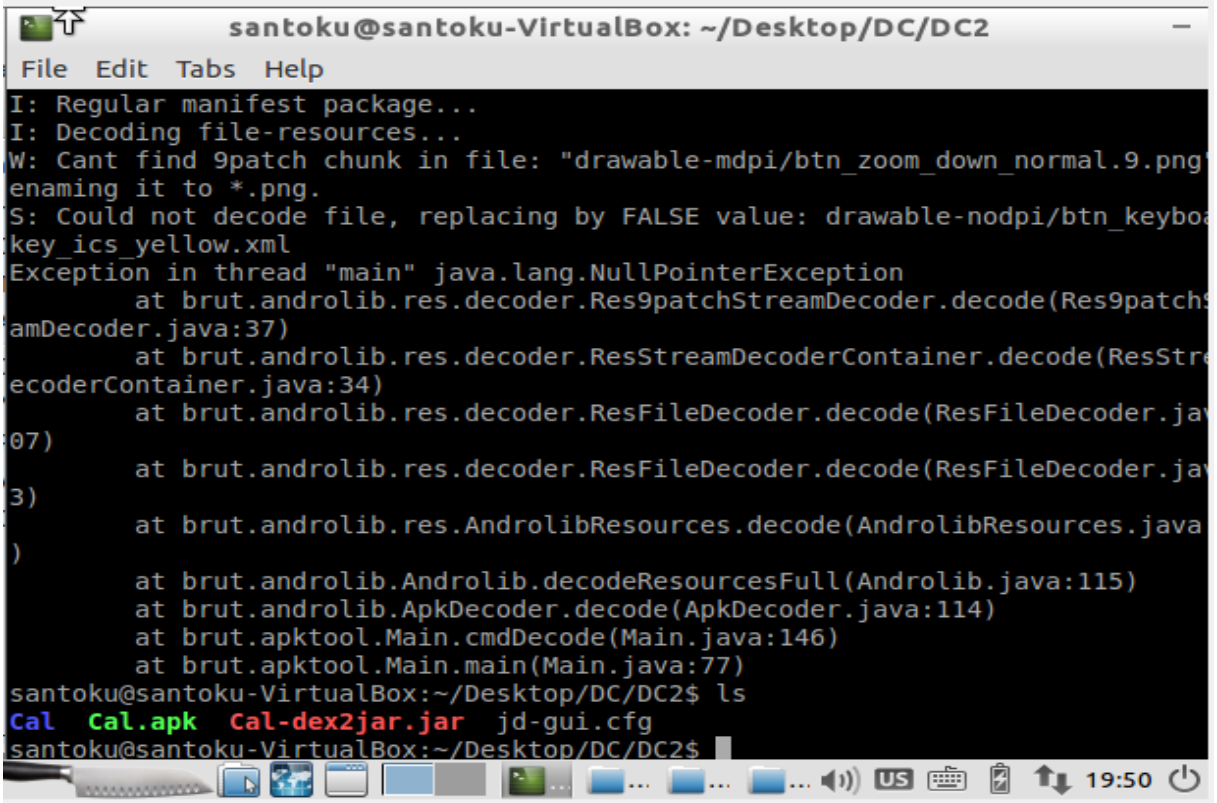
DEUXIEME PARTIE : MODIFICATION ET RECOMPILATION DE L'APPLICATION

Ici nous allons utiliser l'outil **apktool** sous **santoku**.

1- Nous fermons la fenêtre précédente et nous retournons et tapons sur le terminal la commande suivante (première ligne) : [redacted]

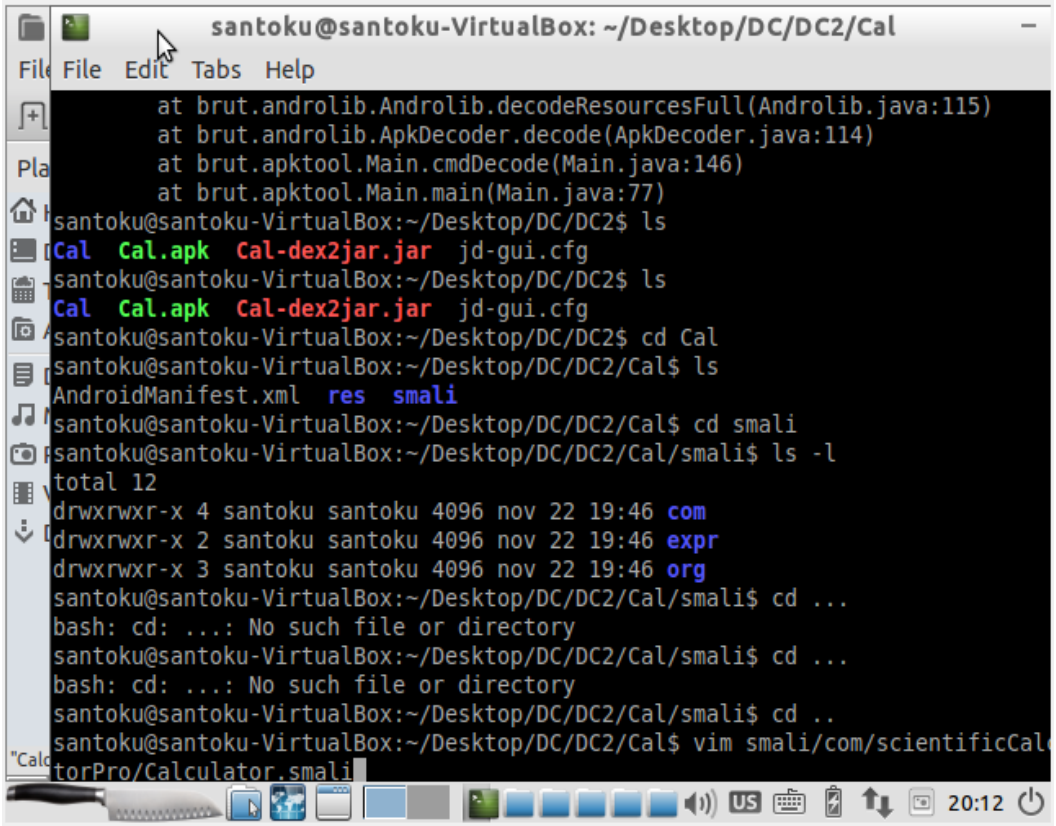


Et nous obtenons juste à coté de l'apk un dossier correspondant au nom de l'apk. la fenêtre suivante nous montre tous les dossiers se trouvant dans le dossier de l'apk:

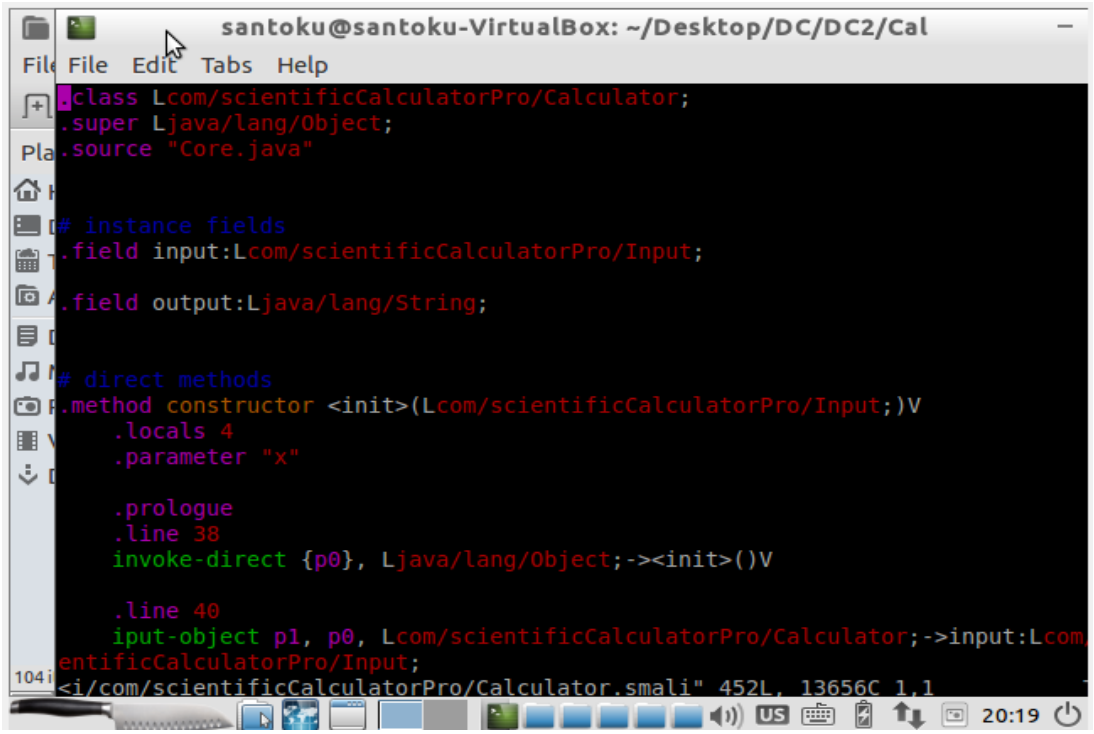


L’avant dernière ligne nous montre qu’évidemment nous avons un dossier nommé Cal correspondant au nom Cal.apk.

2- Ce dossier correspondant au nom de l’application contient toutes les classes de l’applications lesquelles nous pouvons les ouvrir. Ces classes sont précisément dans le dossier **smali** du dossier précèdent. Pour les ouvrir nous allons taper la commande dans le terminal en choisissant la classe à ouvrir et en indiquant le chemin d’accès, à la classe, à l’éditeur qui nous permet de modifier cette classe.
3- Exemple : ici nous avons décider d’ouvrir seulement la classe **Calculator** du dossier **com** se trouvant dans le dossier **smali**. Nous tapons la commande suivante (dernière ligne) :



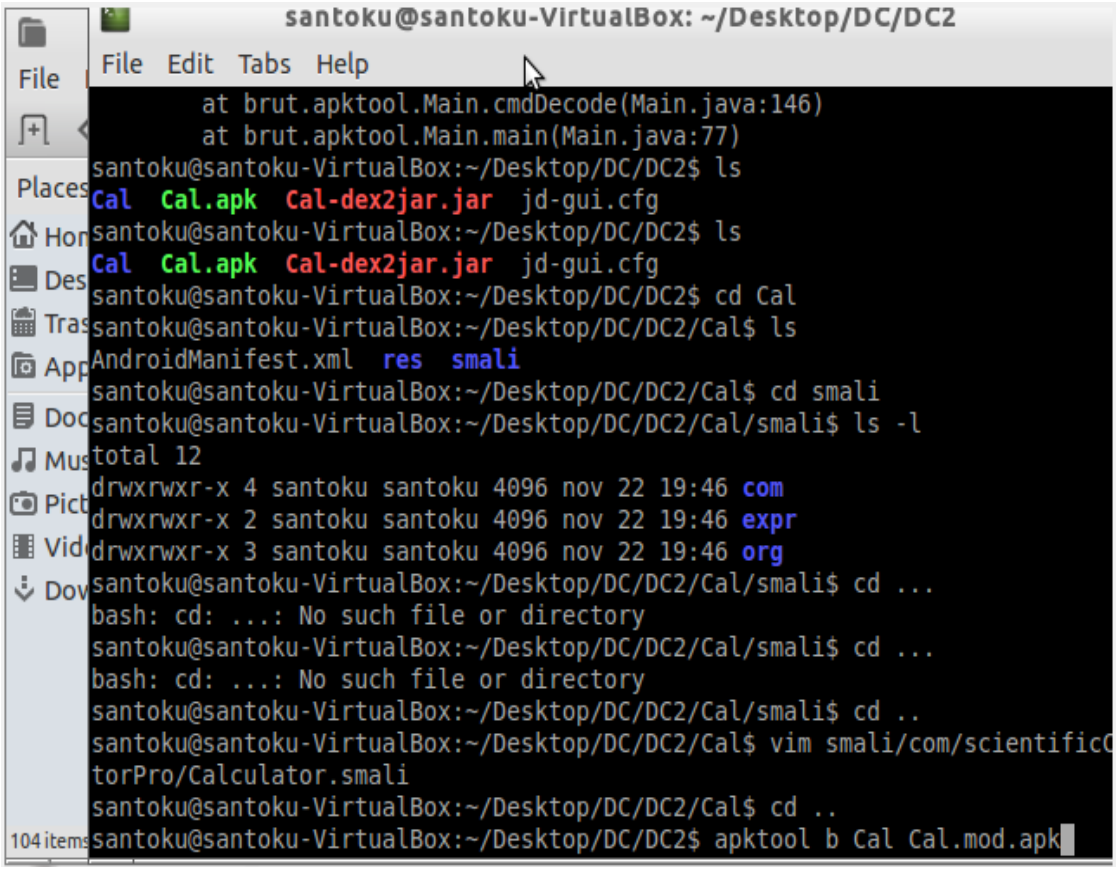
Nous obtenons la figure suivante :



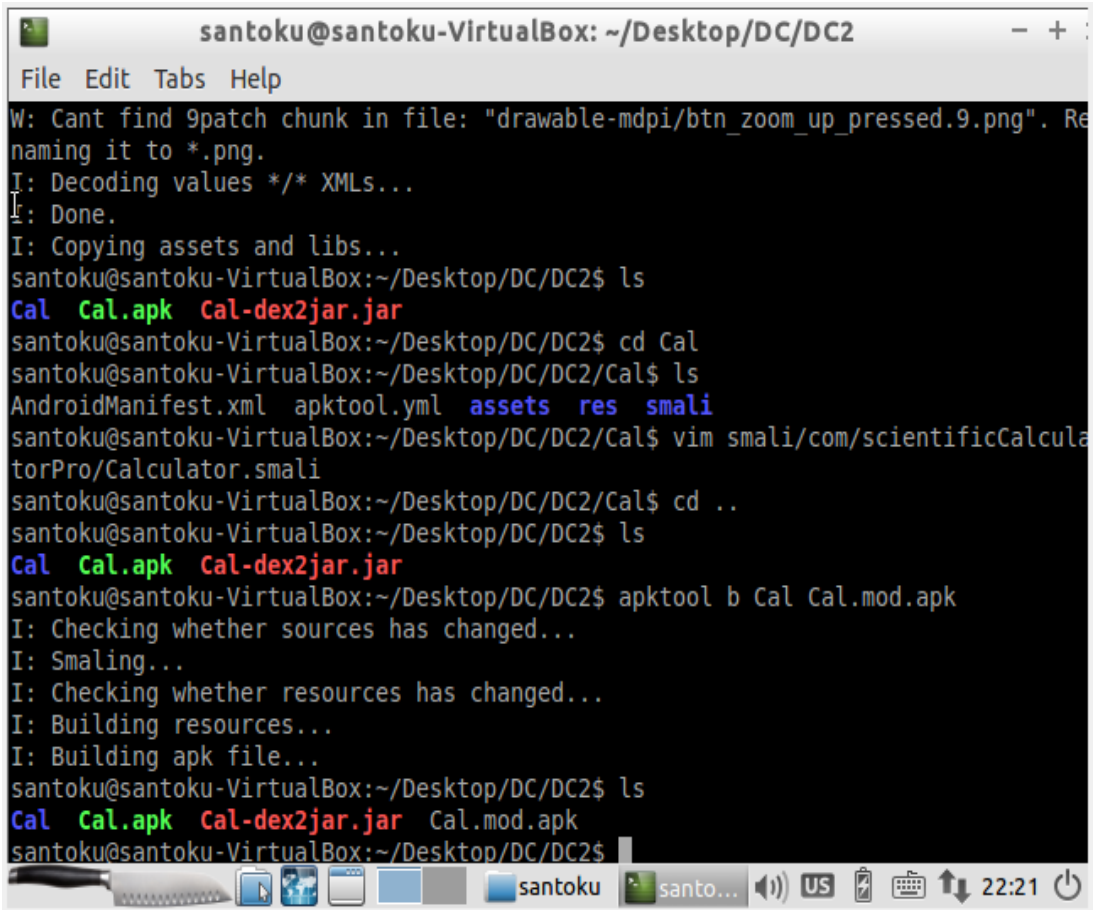
Dans cette fenêtre nous avons la classes Calculator qui est éditée et nous avons la possibilité de la modifier pour modifier ainsi notre application.

4- Apres modification nous pouvons maintenant recompiler notre application pour retrouver le fichier .apk de la nouvelle application. Pour cela nous allons taper la commande suivante (après avoir modifier les

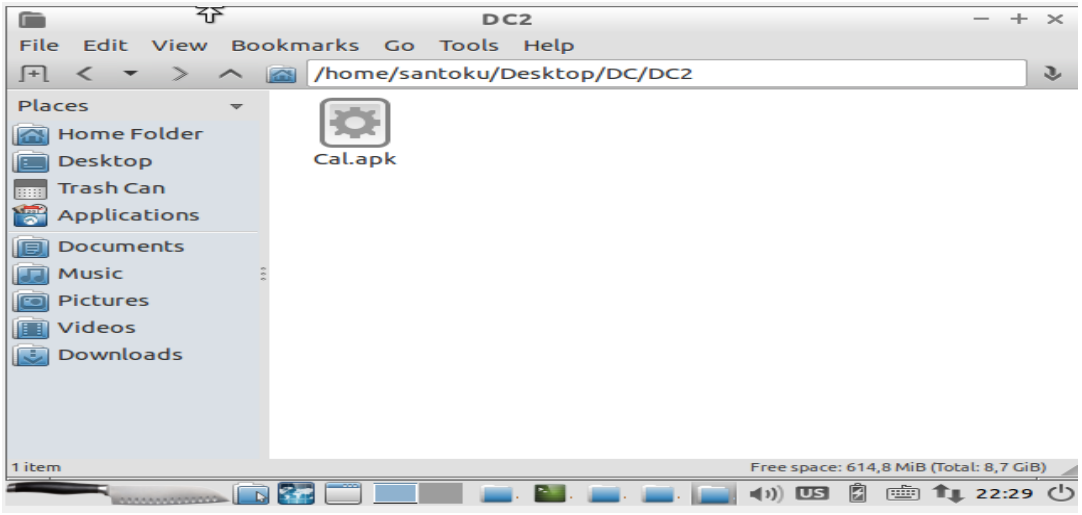
classes et quitter l'éditeur), la dernière ligne :



Après cette commande nous retrouvons dans le dossier de l'apk initial un nouveau apk nommé Cal.mod.apk, correspondant à l'apk initial modifié, comme nous le montre la figure suivante (avant dernière ligne)

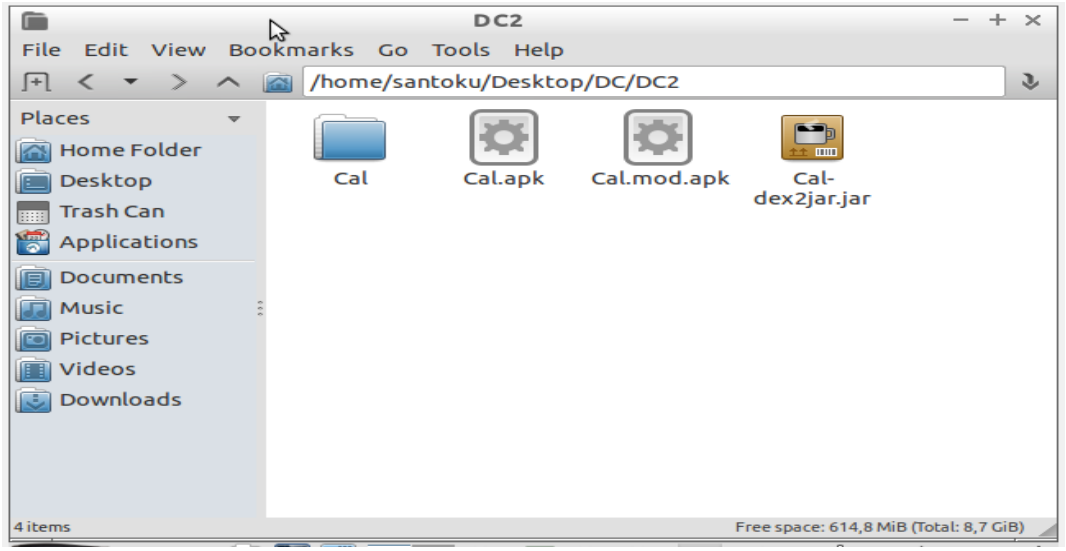


5- Notons que au début du TP le dossier qui contient l'apk se présente comme suit :



6- A la fin du TP le dossier qui contenait seulement l'apk contient maintenant un .jar qui contient le classes de l'application, un dossier correspondant au dossier du projet de l'application et en fin un nouveau

apk recompilé comme le montre la figure suivante :



SOLUTION TP 3

TITRE : L'INTER-COMMUNICATION ENTRE DEUX APPLICATIONS

OBJECTIF : Ce TP a pour objectif d'amener l'étudiant à maitriser et à établir l'intercommunication entre deux applications.

METHODE : Nous allons utiliser, dans le cadre de ce TP, l'environnement de développement **Android Studio.**

1- Nous allons dans un premier temps créer deux applications représentées par deux activités. Lorsqu'on se retrouvera sur l'une des deux activités ouvertes nous marquerons application 1 ou application 2.

Après avoir créé les deux activités représentant nos deux applications nous allons définir dans les deux activités des liens qui leur permettent de passer l'une à l'autre application grâce aux intents.

2- Les intents définis dans les deux activités pour passer d'une application à une autre grâce à un évènement qui s'est déroulé dans l'une ou l'autre application sont définis dans les figures suivantes capturées sous Android :

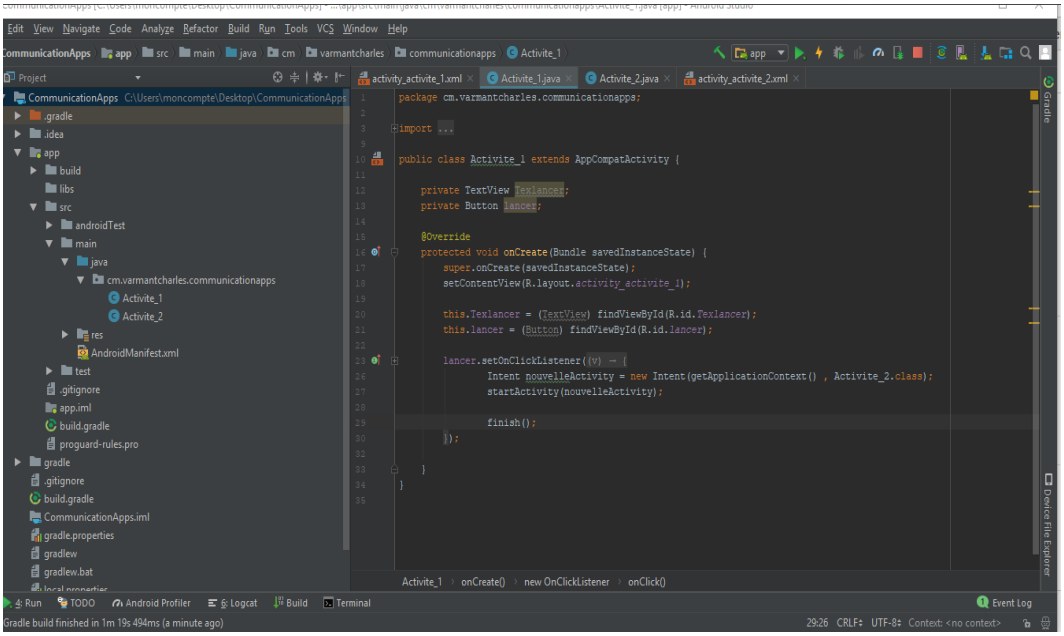


Figure1 : Les intents dans la première application

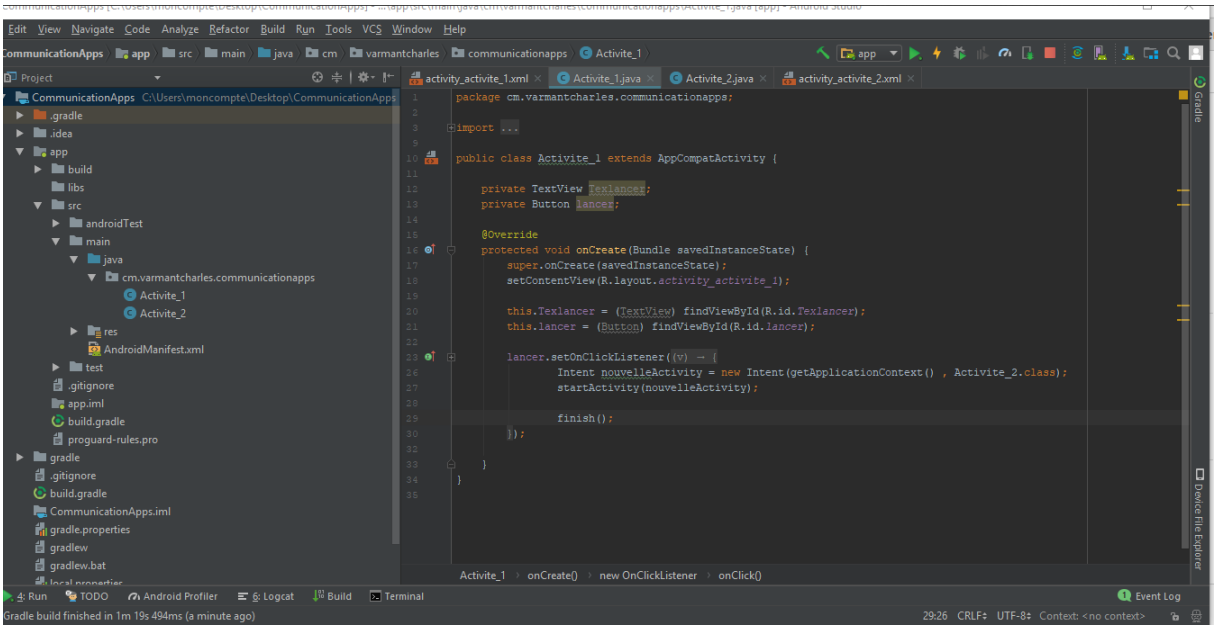


Figure2 : Les intents dans la deuxième application

3- Enfin nous avons les deux figures suivantes qui représentent les deux applications lorsqu'on clique sur le bouton ‘’**CLIQUER**’’ de l'une ou l'autre pour passer à l'autre application.

Figure3 : De l'application 1

figure 4 :De l'application 2.

4- Les deux images sont obtenues pendant la simulation. On a parfaitement la possibilité de simuler l'intercommunication entre deux applications.

Bibliographie

- Cours d'introduction aux systemes d'exploitation mobile par Dr.-Ing. Franklin Tchakounté
- Développement Android de Jean-Francois Lalande – Novembre 2016 - Version 2.5
- Tutoriel%20_%20Créez%20des%20applications%20pour%20Android.html