Afpa

2021

Beata Krzyzosiak

FIL ROUGE :

. PET ID / KITCHEN CONVERTER

Dossier Technique

TABLE DES MATIÈRES

LA PROPOSITION 1

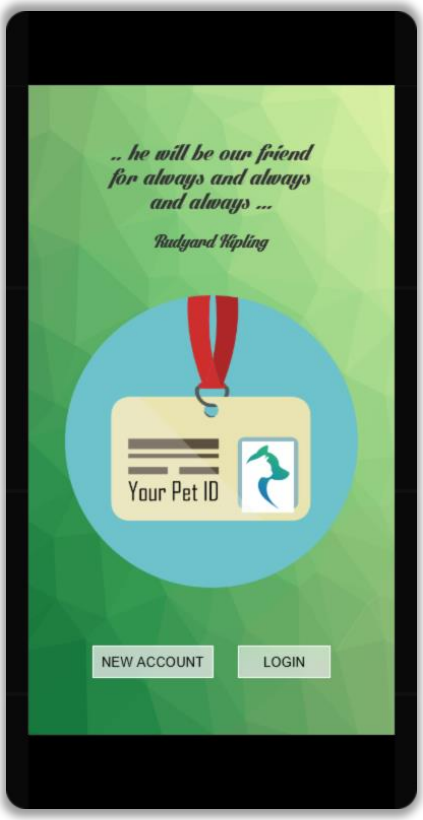


Figure 1 : écran de démarrage (maquette)

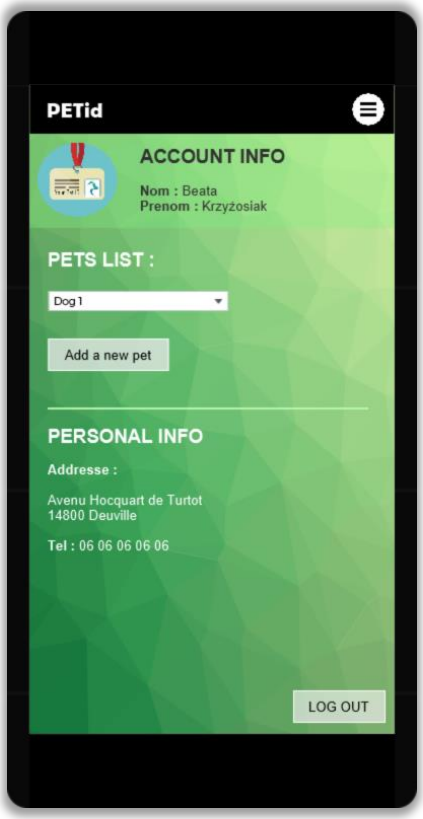


Figure 2 : écran de profil utilisateur (maquette)

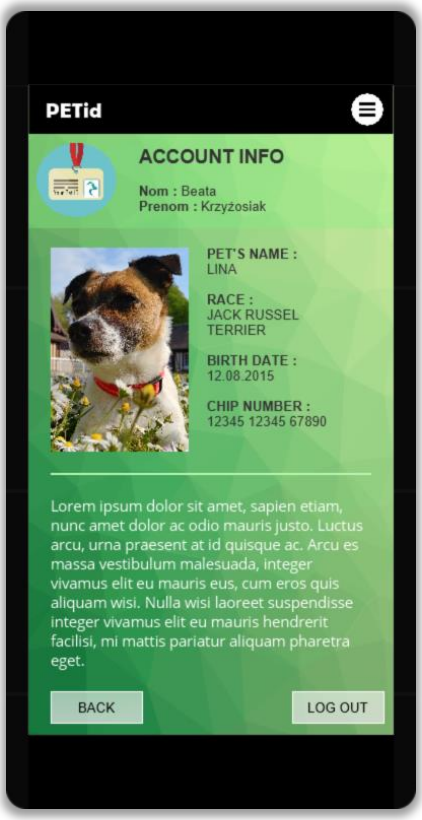


Figure 3 : Écran de profil d'animal de compagnie (maquette)

PET ID : brève description de l'application

Une application conçue pour les amoureux des animaux.

Logo et Icône / Background / Menu :

Figure 4 : Logo

Figure 5 : background de l'application

Figure 6 : menu

Fonctions de base :

- Profil en ligne d'un propriétaire et de son animal
- Stockage de tous les informations de l'animal en un seul endroit

Opportunité de développement - fonctionnalités à ajouter à l'avenir :

- Protection de l'animal lorsqu'il est perdu (Localisation GPS de l'animal)
- Enregistrement dans la base de données animale internationale
- Stockage de photos de l'animal en un seul endroit
- Ajout de rendez-vous vétérinaires au calendrier
- Ajout des notifications, pour exemple : rappels des vaccinations à venir

Exemples d'applications similaires existantes :

- https://play.google.com/store/apps/details?id=com.animalid&hl=en_US&gl=US
- https://apkpure.com/pet-identity-card/com.nudecreatives.petid
- https://animal-id.net/en/app/
- https://pet.digitail.io
- https://www.puptodate.com

LA PROPOSITION 2

Figure 1 : Ecran de démarrage (maquette)

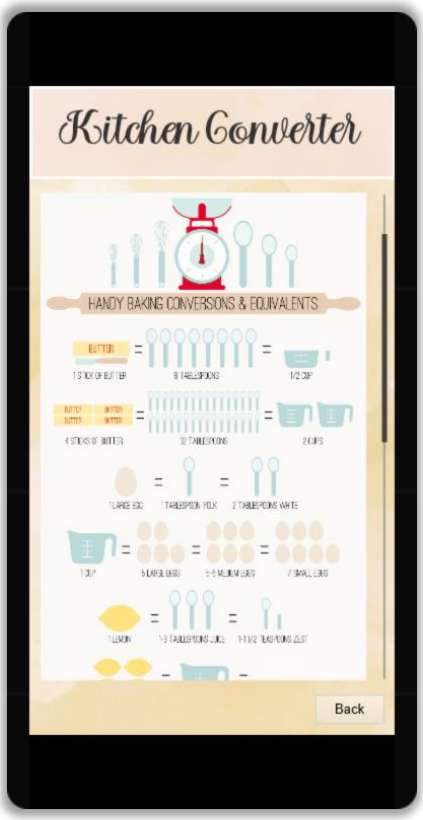


Figure 2 : Table (maquette)

Figure 3 : Convertisseur (maquette)

KITCHEN CONVERTER : brève description de l'application

Le convertisseur de cuisine convertit le volume de l'unité afin que vous puissiez facilement échanger par exemple des litres en tasses, etc.

Logo et Icône / Background / Menu :

Figure 4 : Logo

Figure 5 : background de l'application

Fonctions de base :

- **Tableau de mesure**
- **Convertisseur de cuisine :** volume : Cuillères à café, cuillères à soupe, tasses, litres, centimètres, millilitres, pincées ; poids : livres, onces, grammes, kilogrammes ; température : Fahrenheit à Celsius

Opportunité de développement - fonctionnalités à ajouter à l'avenir :

- Calculatrice de cuisine : vous pouvez additionner, soustraire, multiplier et diviser et afficher les résultats arrondis aux fractions de cuisson standard.
- Fractions de cuisson : saisissez les quantités et affichez les résultats en fractions de cuisson standard (1/2, 1/3, 1/4, etc.)

Exemples d'applications similaires existantes :

- https://play.google.com/store/apps/details?id=kalkulator.kuchenny&hl=pl&gl=US
- https://play.google.com/store/apps/details?id=com.mykitchencalculator.kitchencalculator&hl=en_US&gl=US
- https://play.google.com/store/apps/details?id=com.mg.kalkulatorkuchenny&hl=en_US&gl=US

Introduction

Sélection de proposition

Après avoir pris en compte le temps disponible et le niveau initial de préparation des deux projets, l'auteur a décidé de choisir la deuxième option *: Kitchen Converter*.

Une brève note sur l'application sélectionnée

L'idée de cette application a été créée pour des raisons strictement pratiques. De nombreuses recettes de cuisine donnent différentes mesures. Parfois, il est difficile de s'y retrouver et de se souvenir d'eux tous. Cette application est conçue pour organiser ces informations et faciliter la conversion des quantités de base utilisées en cuisine.

Réalisation graphique

Concept

Le leitmotiv ici sont des couleurs chaudes et pastel, rappelant de savoureuses pâtisseries agrémentées de fruits et de chantilly. L'auteur fait référence au style clipart Powerpoint, mais ajoute en même temps une touche d'élégance (par exemple sous la forme d'une police utilisée pour le titre). Tout cela est conservé dans une structure propre et harmonieuse.

Maquettes et la version finale

À gauche : maquette, à droite : conception finale (écran de démarrage)

Hormis des modifications mineures dues à des différences logicielles :

(Maquettes : Justinmind 9.4.1
⬇
application : Android Studio 11.0.10+),

ainsi que pour optimiser l'espace et profiter d'un accueil plus agréable, le concept global a cependant été retenu.

Détails techniques de la mise en œuvre graphique

- **Le programme utilisé pour les maquettes :** Justinmind 9.4.1
- **Background :** https://pngtree.com/freebackground/wild-yellow-gradient-watercolor-poster-background-material_978320.html
- **Autres graphiques :** https://www.pngegg.com, https://www.clipartmax.com/ , Android Studio
- **Polices :** *Baby sweet* (pour le texte : Kitchen Converter), *Bauhaus 93* (pour le boutons)
- **Programmes graphiques utilisés pour le traitement :** PhotoScape X 4.1.1
- **Couleurs :**

```
<color name="black">#FF000000</color> // buttons, app headband
<color name="white">#FFFFFFFF</color> // text
<color name="silver">#C0C0C0</color> // hint text in EditView
```

- **Positions d'écran actives :** verticale (la position horizontale est bloqué)

Plan de navigation

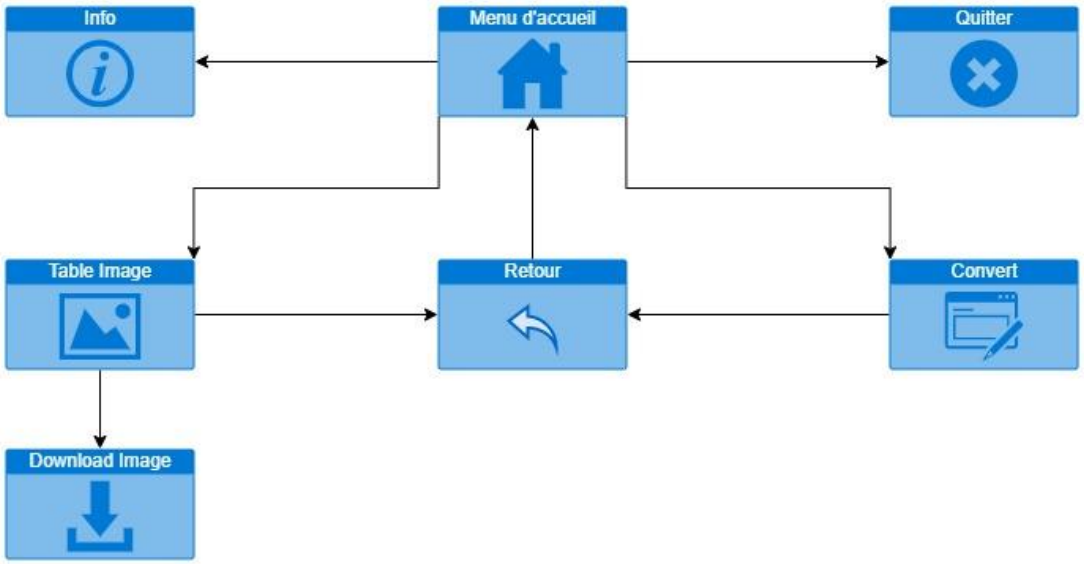
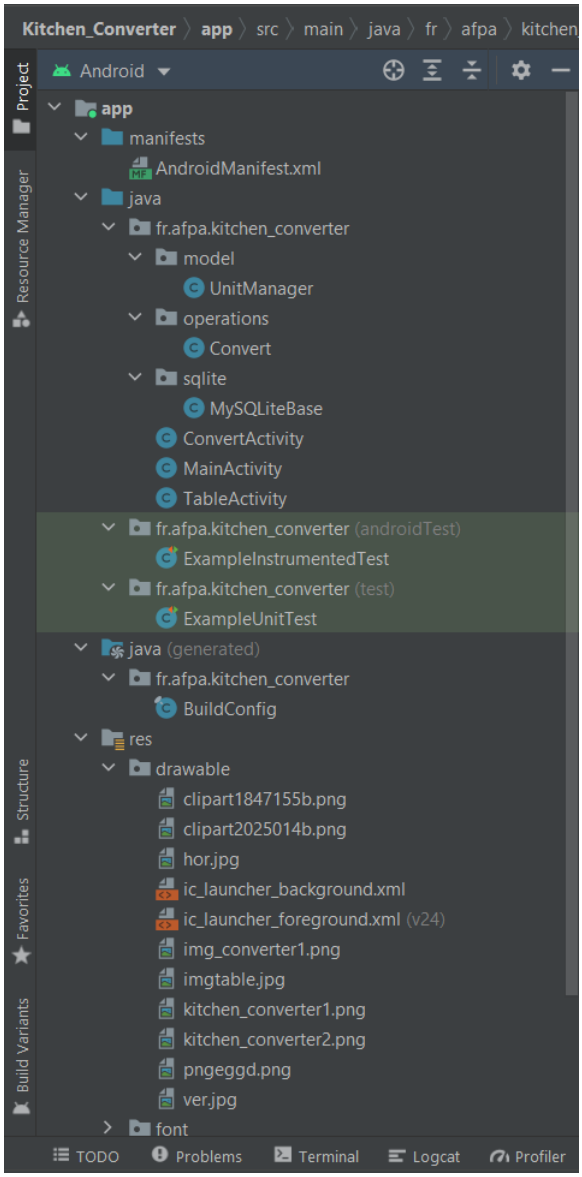
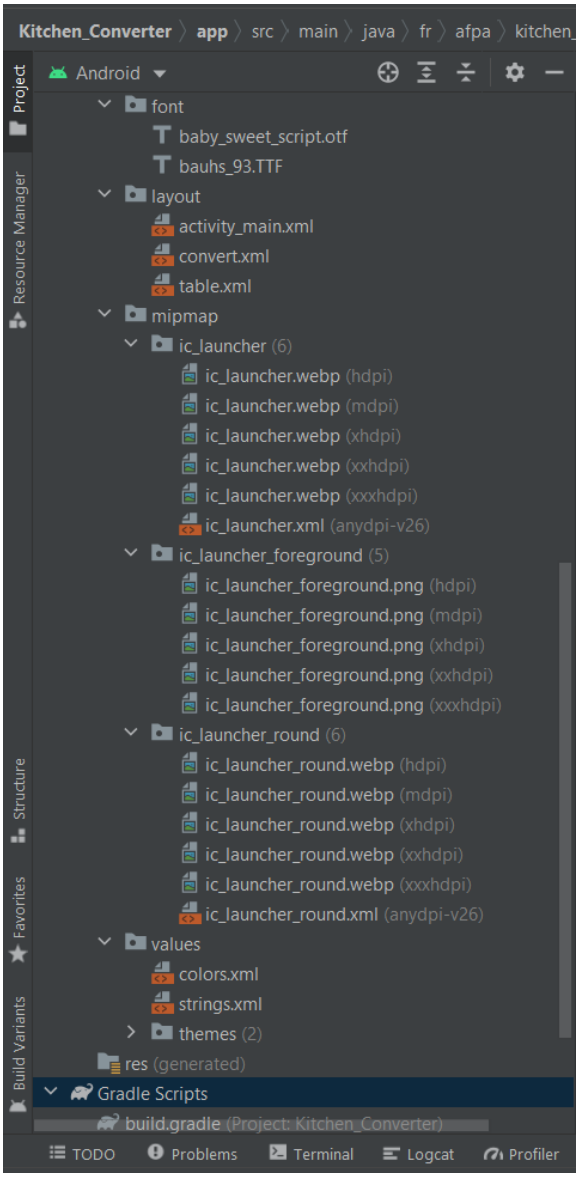


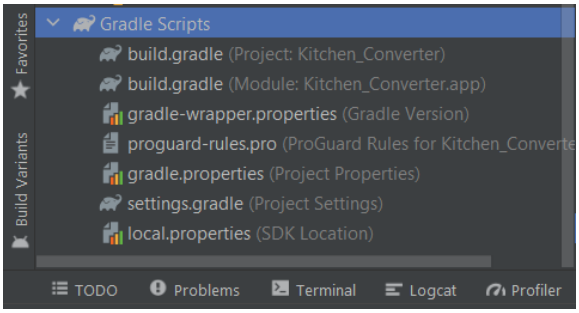
Figure : Plan de navigation

DEVELOPPEMENT

Liste des fichiers







Liste des fonctions

Fichier MainActivity.java

```
protected void onCreate(Bundle savedInstanceState) {
```

- état de l'instance enregistré
- définition de la vue appropriée

```
public void table(View view) {
```

- envoie vers la vue d'écran de la nouvelle activité *Table*

```
public void convert(View view) {
```

- envoie vers la vue d'écran de la nouvelle activité *Convert*

```
(1) public void info(View view) {
    (2) public void onClick(DialogInterface dialogInterface, int i) {
```

L'alerte se déclenche lorsque vous tapé sur l'image : 

```
android:onClick="info"
```

- (1) alert dialog builder - construction de la fenêtre d'alerte qui apparaît à l'écran
- (2) Action onClick « Close » - cliquer sur le bouton *Close* ferme la fenêtre d'alerte

```
(1) public void quit(View v) {
    (2) public void onClick(DialogInterface dialogInterface, int i) {
```

L'alerte se déclenche lorsque vous tapé sur l'image : 

```
android:onClick="quit"
```

- (1) alert dialog builder - construction de la fenêtre d'alerte qui apparaît à l'écran
- (2) Action onClick - deux options : NON ou OUI.

NO - provoque la sortie de l'alerte, YES - provoque la sortie de l'application

Fichier TableActivity.java

```
protected void onCreate(Bundle savedInstanceState) {
```

- état de l'instance enregistré
- définition de la vue appropriée

```
private void initializeDownloadManager() {
```

- initialiser le gestionnaire de téléchargement

```
private void downloadFile(){
```

- définissez le titre et la description de ce téléchargement, à afficher dans les notifications

```
public void clickView(View view){
```

- gérer les clics sur le bouton : bouton TÉLÉCHARGER

```
public void back(View view) {
```

- appuyer sur le bouton RETOUR met fin à l'action (et à la vue) rappelée et revient à l'action précédente (et à la vue correspondante qui l'accompagne)

Fichier ConvertActivity.java

```
protected void onCreate(Bundle savedInstanceState) {
```

- état de l'instance enregistré
- définition de la vue appropriée

```
private void chargeSpinner(int idSpinner) {
```

- charge le spinner avec la liste de propositions

```
private void chargeUnit() {
```

- charge les propositions de la liste avec le values

```
public void do_convert(View v) {
```

- vérifier les champs, convertir la mesure de départ et afficher le montant de la mesure de fin
- met le contrôle sur chaque champ (*spinners*, *EditView*) du convertisseur
- bouton « CONVERT » - tapé produit l'affichage du résultat dans le *ViewText (android:id="@+id/result")*, sauf s'il existe une ou des conditions (*case*) qui rendent impossible l'affichage du résultat, alors le contrôle établi

provoque l'affichage du message sous forme de *toast* (voir : le chapitre : Fonctions et Jeux d'essai (répartition des fonctions par écran)).

```
public void back(View view) {
```

- appuyer sur le bouton RETOUR met fin à l'action (et à la vue) rappelée et revient à l'action précédente (et à la vue correspondante qui l'accompagne)

Package : model : UnitManager.java

```
public static Map<String, Double> getAll (Context context) {
```

- obtient de données a partit d'une table de base de données

Package : operations : Convert.java

```
public static double convertir(String source, String target, double amount)
```

- renvoie un double pour le << montant >> unité << source >> converti en unité << cible >>

```
public static Map<String, Double> getConversionTable(Context context)
```

- assessor of the association table of units

Package : sqlite : MySQLiteBase.java

```
public class MySQLiteBase extends SQLiteOpenHelper {
```

- classe de construction de base de données SQLite

```
public MySQLiteBase(@Nullable Context context, @Nullable String name, @Nullable
SQLiteDatabase.CursorFactory factory, int version) {
```

- établit les propriétés de la base de données

```
public void onCreate(SQLiteDatabase db) {
```

- définit la table de la base de données

```
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
```

- spécifie une modification de base de données

Fonctions et Jeux d'essai (répartition des fonctions par écran)

ÉCRAN DE DÉMARRAGE (activity_main.xml)

Screenshot : ***écran de démarrage***

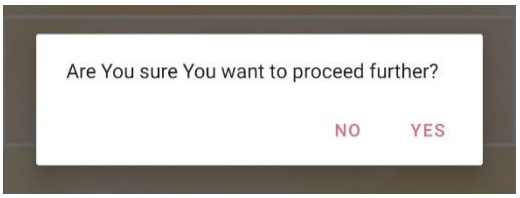
Description :

(1) **Mobile Headband** avec le nom d'application

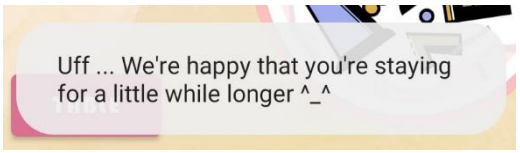
(2) **Quitter** image-button :

```
/**
 * QUIT button
 * @param v
 */
public void quit(View v) {
```

Après avoir appuyé sur l'image, une fenêtre d'alerte apparaît, demandant si l'utilisateur souhaite poursuivre l'action. L'utilisateur peut choisir entre une réponse affirmative, qui ferme l'application, ou une réponse négative, qui ferme la fenêtre d'alerte, et un message *toast* s'affiche en bas de l'écran.



CASE « NO » provoque le message *toast* suivant :



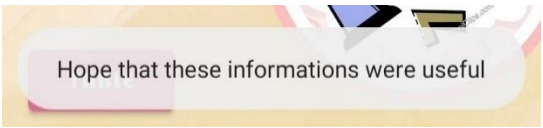
(3) Image **logo** d'application

(4) **Info** image-button :

```
/**
 * INFO button
 * @param view
 */
public void info(View view) {
```

Après avoir appuyé sur l'image, une fenêtre d'alerte apparaît. Il contient des informations sur l'application : version, auteur, client, chef de projet, et un contact e-mail qui peut être utilisé, par exemple, pour signaler des bugs. Vous pouvez quitter la fenêtre en appuyant sur le bouton de sortie.

CASE « CLOSE » provoque le message *toast* suivant :



(5) **Bannière** avec le nom de l'application

(6,8) **CONVERT** button et image-button

```
/**
 * SHOW << a CONVERT View >>
 * (Activity : Convert)
 * @param view
 */
public void convert(View view) {
```

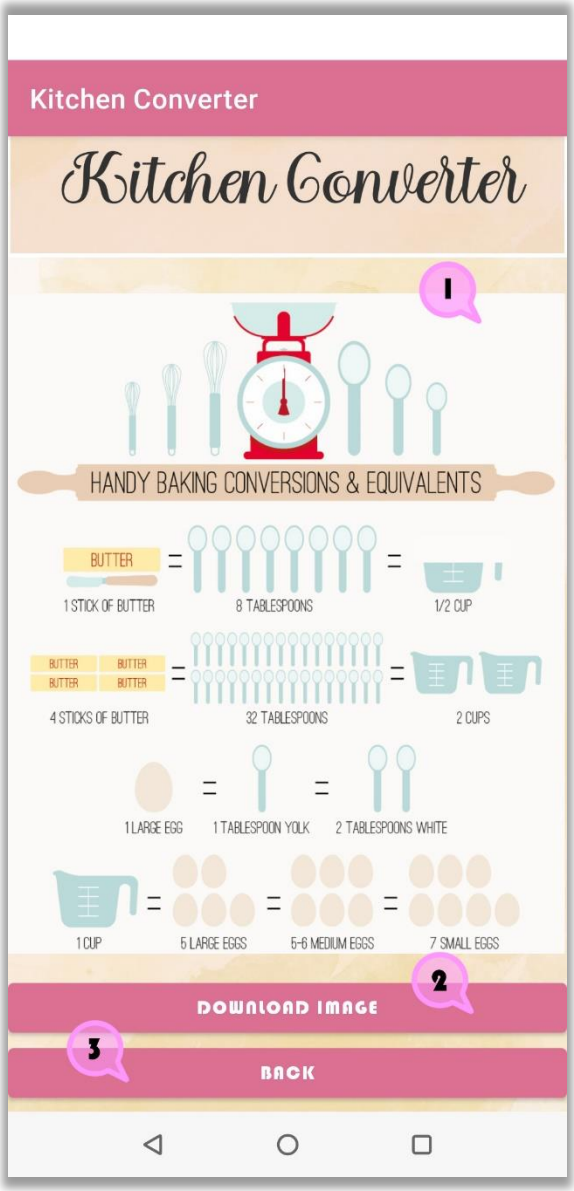
En appuyant sur l'image ou sur le bouton, l'utilisateur bascule vers une autre vue d'écran (dans ce cas : vers le convertisseur).

(7,9) **TABLE** button et image-button

```
/**
 * SHOW << a CONVERT TABLE View >>
 * (Activity : Table)
 * @param view
 */
public void table(View view) {
```

En appuyant sur l'image ou sur le bouton, l'utilisateur bascule vers une autre vue d'écran (dans ce cas : vers la table).

ÉCRAN AVEC LA TABLE (table.xml)



Screenshot : ***écran avec table***
Description :
(1) **Table image** dans le ScrollView
(2) **Download image** button :

```
/**
 * Handle button clicks : DOWNLOAD button
 * @param view
 */
public void clickView(View view){
```

Après avoir appuyé sur le bouton, message toast s'affiche en bas de l'écran. En arrière-plan, la notification indique que le téléchargement est terminé.

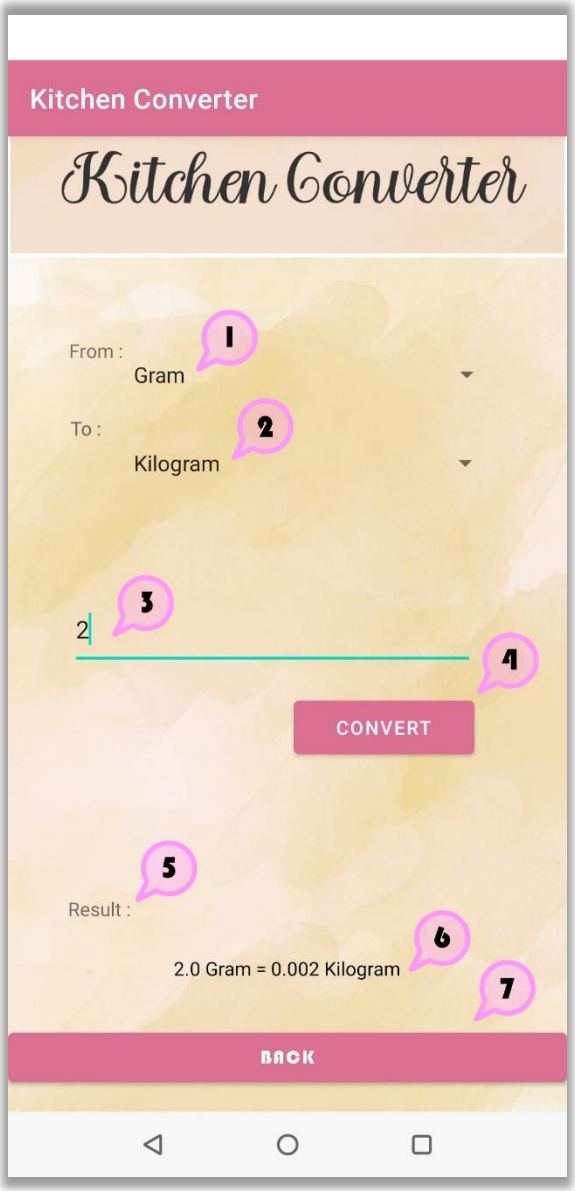
Après avoir tapé sur la notification, l'image téléchargée s'ouvre. L'image peut être réduite ou agrandie à volonté.

(3) **« BACK »** button **:**

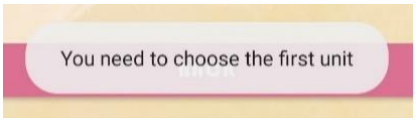
```
/**
 * BACK button
 * @param view
 */
public void back(View view) {
```

Termine l'action ouverte (vue ouverte) et revient à l'action précédente.

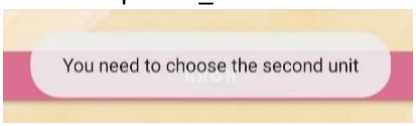
ÉCRAN DU CONVERTISSEUR (convert.xml)



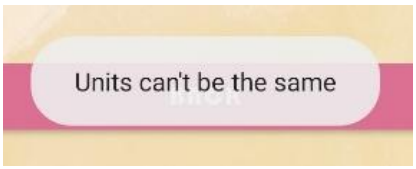
Screenshot : ***écran du convertisseur***
Description :
(1) **Spinner** *spinner_from* : l'utilisateur peut ouvrir la liste des unités disponibles et sélectionner celle dont il a besoin.
(2) **Spinner** *spinner_to* : l'utilisateur peut ouvrir la liste des unités disponibles et sélectionner celle dont il a besoin.
(3) **EditView** android:id="@+id/enter_an_amount" : dans ce champ l'utilisateur entre la quantité à convertir. S'il oublie de le faire, un message *toast* apparaît lui rappelant d'entrer un nombre à convertir.
* La même chose se produit avec les *spinners* - lorsqu'une valeur n'a pas été choisie ou que les deux sont identiques un message *toast* apparaît.
CASE A : spinner_from est vide:



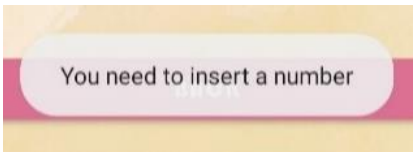
CASE B : spinner_to est vide:



CASE C : spinner_from et spinner_to sont pareil :



CASE D : EditView est vide :



(4) **« CONVERT » button :** après que l'utilisateur a tapé le button , le résultat s'affiche (voir : 6) dans le *TextView* à moins que l'un des cas mentionnés ci-dessus ne se produise.

```
/**
 * Check fields, convert the starting unit and display the amount of the ending unit
 * @param v
 */
public void do_convert(View v) {
```

(5) **TextView : result_txt**

```
android:id="@+id/result_txt"
```

(6) **TextView : result**

```
android:id="@+id/result"
```

(7) **« BACK » button :**

```
/**
 * BACK button
 * @param view
 */
public void back(View view) {
```

Termine l'action ouverte (vue ouverte) et revient à l'action précédente.

Liste des fonctionnalités non présentes dans le cours

Fonction de téléchargement (DownloadManager)

Cette fonctionnalité sera discutée en détail dans le tutoriel créé pour ce projet (cliquez pour aller).

```
/**
 * Initialize download manager
 */
private void initializeDownloadManager() {


/**
 * Set the title and desc of this download, to be displayed in notifications.
 */
private void downloadFile(){


/**
 * Handle button clicks : DOWNLOAD button
 * @param view
 */
public void clickView(View view){
```

AndroidManifest.xml (code)

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="fr.afpa.kitchen_converter">

    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Kitchen_Converter">
        <activity
            android:name=".MainActivity"
            android:screenOrientation="portrait"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <activity
            android:name=".TableActivity"
            android:screenOrientation="portrait"
            android:requestLegacyExternalStorage="true"
            android:usesCleartextTraffic="true"></activity>

        <activity
            android:name=".ConvertActivity"
            android:screenOrientation="portrait"></activity>

    </application>

</manifest>
```

Installation et donnes techniques

Installation

Installer l'application Kitchen_Converter.apk

Jeux d'essai

La plupart des cas possibles sont traités dans le chapitre :
Fonctions et Jeux d'essai (répartition des fonctions par écran).

Description générale jeux d'essai possible :

- l'utilisateur décide de vérifier les informations sur l'application et appuie sur l'image "i" (voir : ici) > une fenêtre de type *alerte* avec des informations apparaît > l'utilisateur quitte la fenêtre en cliquant sur la fenêtre adjacente ou sur le bouton CLOSE > un message *toast* apparaît brièvement.

```
// ALERT DIALOG BUILDER
AlertDialog.Builder alertDialogBuilder = new AlertDialog.Builder(this);
alertDialogBuilder.setMessage("Version: 1.0_04112021" +
                              "\nAuthor: Beata Krzyżosiak (Rimwor)" +
                              "\nFor: AFPA -  Caen - France" +
                              "\nProject supervisor: Damien Bin" +
                              "\nContact: kitchen_converter@gmail.com");

// << QUIT >> BUTTON
alertDialogBuilder.setNegativeButton("Close", new
DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialogInterface, int i) {
        Toast.makeText(MainActivity.this, "Hope that these informations were
useful", Toast.LENGTH_LONG).show();
    }
});
```

- l'utilisateur décide de quitter l'application et appuie sur l'image correspondant à cette tâche (voir ici). > une fenêtre de type *alerte* apparaît avec une question et deux réponses > l'utilisateur peut approuver ce qui quitte l'application ou démissionner, ce qui fermera la fenêtre de type d'alerte et affichera un message *toast*.

```
// ALERT DIALOG BUILDER
AlertDialog.Builder alertDialogBuilder = new AlertDialog.Builder(this);
alertDialogBuilder.setMessage("Are You sure You want to proceed further?");

// << YES >> BUTTON
alertDialogBuilder.setPositiveButton("Yes", new
DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialogInterface, int i) {
        MainActivity.this.finish();
        System.exit(0);
    }
});
// << NO >> BUTTON
alertDialogBuilder.setNegativeButton("No", new
DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialogInterface, int i) {
        Toast.makeText(MainActivity.this, "Uff ... We're happy that you're
staying \nfor a little while longer ^_^", Toast.LENGTH_LONG).show();
    }
});
```

- l'utilisateur décide d'appuyer sur l'image ou sur le bouton TABLE (voir : ici) > cela l'amène à une nouvelle vue (voir : ici), où l'utilisateur peut voir l'image placée dans le *ScrollView* et, s'il le souhaite, la capturer en appuyant sur le bouton TÉLÉCHARGER L'IMAGE > après avoir appuyé sur le bouton TÉLÉCHARGER L'IMAGE, le téléchargement commencera et un message *toast* s'affichera > l'utilisateur peut revenir à l'action/écran précédent en appuyant sur le bouton RETOUR. Il peut également consulter ses notifications pour voir l'image ou la trouver directement dans le dossier TÉLÉCHARGEMENT de l'application.

```
/**
 * Handle button clicks : DOWNLOAD button
 * @param view
 */
public void clickView(View view){
    switch (view.getId()){
        case R.id.download:
            downloadFile();
            Toast.makeText(getBaseContext(), R.string.msg_downloading,
Toast.LENGTH_SHORT).show();
            break;
    }
}
```

- l'utilisateur décide d'aller sur CONVERT en appuyant sur l'image ou le bouton (voir : ici) > cela l'amène à une nouvelle vue (voir : ici) > l'utilisateur peut convertir - si l'une des conditions n'est pas remplie, des avertissements apparaîtront sous la forme d'un message *toast*.

```
// TEST if the initial unit is EMPTY
if (strInitUnit.equals("")) {
    Toast.makeText(getBaseContext(),R.string.err_init_unit,Toast.LENGTH_SHORT).show();

// TEST if final unit is EMPTY
} else if (strFinalUnit.equals("")) {

Toast.makeText(getBaseContext(),R.string.err_final_unit,Toast.LENGTH_SHORT).show();

// TEST if units are different
} else if (strInitUnit.equals(strFinalUnit)) {
    Toast.makeText(getBaseContext(),R.string.err_double,Toast.LENGTH_SHORT).show();

// TEST if there are only numbers (and no letters)
} else if (strAmount.equals("") || strAmount.matches("[^0-9]")) {
    Toast.makeText(getBaseContext(),R.string.err_amount,Toast.LENGTH_SHORT).show();

} else {

    // ALL OK
...
} catch (NumberFormatException e) {

    // The Amount error
    Toast.makeText(getBaseContext(),R.string.err_amount,Toast.LENGTH_SHORT).show();
}
```

Spécificités techniques

```
com.android.tools.build:gradle:7.0.3
applicationId "fr.afpa.kitchen_converter"
minSdk 21
targetSdk 30
versionCode 1
versionName "1.0"
```

Optimisé pour : téléphone mobile notamment pour le Asus Zenfone 5z.

Versions linguistiques

- English
- Polski
- Français

Mesure de conversion

```
('Kilogram',1000);";
('Gram',1);";
('Centigram',0.01);";

('Decagram',10);";
('Hectogram',100);";

('Litre',1000);";
('Millilitre',1);"
('Centilitre',0.01);";

('1 Glass',250);";
 ('3/4 Glass',180);";
 ('2/3 Glass',160);";
 ('1/2 Glass',125);";
 ('1/3 Glass',80);";
 ('1/4 Glass',60);";
 ('1/6 Glass',40);";
 ('1/8 Glass',30);";
 ('1/16 Glass',15);";

 ('1 Tea Spoon',5);";
 ('1 Table Spoon',15);";
```

Outils

Logiciel :

- Justinmind 9.4.1
- Android Studio 11.0.10+
- PhotoScape X 4.1.1

Application testée sur :

- (ASUS_Z01RD) Asus Zenfone 5z avec Android 10

Sources (tutorials et autres)

- https://www.geeksforgeeks.org/locking-screen-orientation-in-android/ : bloquer la position d'écran
- https://stackoverflow.com/questions/22521958/android-layouts-imageview-inside-scrollview-space-before-image/37070308 : scrollview tips
- https://stackoverflow.com/questions/1839273/how-to-apply-click-event-listener-to-image-in-android : click-event-listener pour l'image
- https://github.com/vad-zuev/ImageDownloader/blob/master/app/src/main/java/com/so/example/tools/BasicImageDownloader.java : TAG – version améliore
- https://camposha.info/android-examples/android-downloadmanager/#gsc.tab=0 : download manager

DOWNLOAD (from URL) | JAVA | ANDROID STUDIO

AFPA (v.1 : 2021)

DEMO TUTORIAL

Beata Krzyżosiak

Table des matières

Remarques

L'emplacement où le fichier a été enregistré

Dans mon cas, le fichier est téléchargé dans le dossier de téléchargement de l'application. Il est possible de modifier ce dossier, mais je n'ai pas abordé ce sujet dans ce tutoriel.

Sources

Le tutoriel est basé sur l'application Kitchen_Converter. La fonction de téléchargement est basée sur les liens disponibles ci-dessous.

Autre

Cet exemple particulier fonctionne mieux pour télécharger une seule image/fichier.

Liens utile :

https://camposha.info/android-examples/android-downloadmanager/#gsc.tab=0

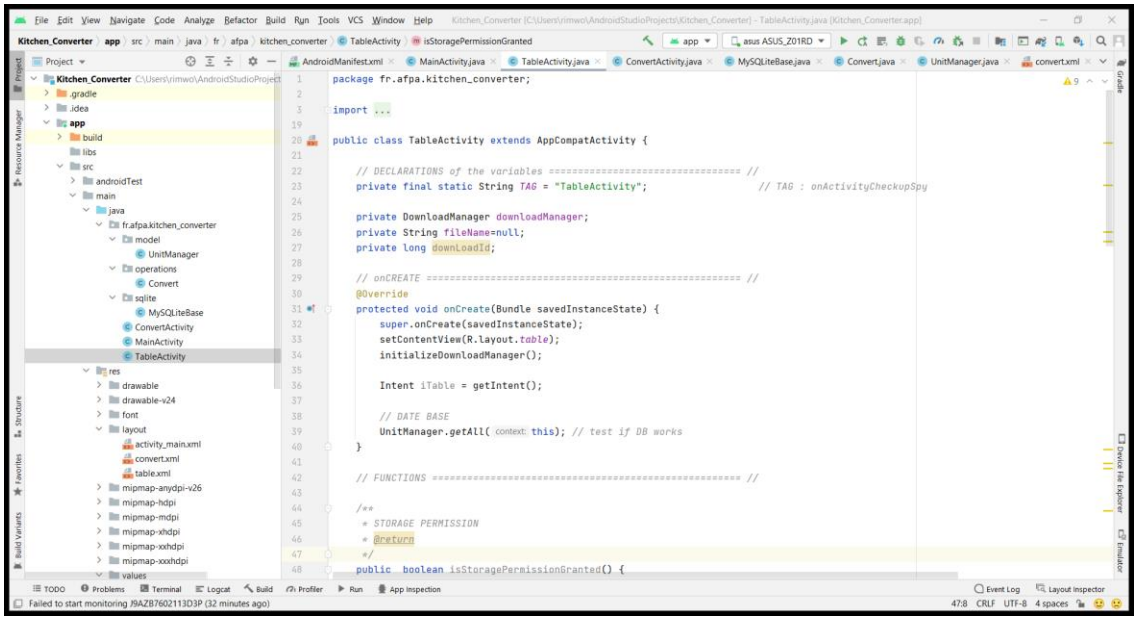
https://developer.android.com/reference/android/app/DownloadManager

https://developer.android.com/reference/android/Manifest.permission#INTERNET

https://developer.android.com/training/notify-user/build-notification

CODE Activity.java

On met le code dans le code de l'action/vue dans laquelle s'effectue le téléchargement.
Dans mon cas, il s'agit de *TableActivity.java*



Déclaration des variables

```
// DECLARATIONS of the variables ================================== //
private final static String TAG = "TableActivity";          // TAG : onActivityCheckupSpy

private DownloadManager downloadManager;
private String fileName=null;
private long downLoadId;
```

Demande d'autorisation d'utiliser le disque de l'appareil

```
/**
 * STORAGE PERMISSION
 * @return
 */
public  boolean isStoragePermissionGranted() {
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
        if (checkSelfPermission(android.Manifest.permission.WRITE_EXTERNAL_STORAGE)
                == PackageManager.PERMISSION_GRANTED) {
            Log.v(TAG,"Permission is granted");
            return true;
        } else {

            Log.v(TAG,"Permission is revoked");
            ActivityCompat.requestPermissions(this, new
String[]{Manifest.permission.WRITE_EXTERNAL_STORAGE}, 1);
            return false;
        }
    }
    else { // permission is automatically granted on sdk<23 upon installation*
        Log.v(TAG,"Permission is granted");
        return true;
    }
}
```

* l'autorisation est automatiquement accordée sur sdk<23 lors de l'installation

Initialiser DownloadManager

Informations sur Download Manager
https://developer.android.com/reference/android/app/DownloadManager

```
/**
 * Initialize download manager
 */
private void initializeDownloadManager() {
    downloadManager= (DownloadManager) getSystemService(DOWNLOAD_SERVICE);
    fileName="Kitchen_Converter_Table.jpg";
}
```

Lors de la saisie du nom du fichier, n'oubliez pas son extension (par exemple JPG), sinon vous ne pourrez pas ouvrir le fichier enregistré !!

Paramètres - nom de fichier et les informations - affichées dans la notification de l'appareil

L'utilisateur peut voir la progression du téléchargement du fichier dans les notifications et l'ouvrir une fois le processus terminé (En savoir plus sur les notifications : https://developer.android.com/training/notify-user/build-notification).

```
/**
 * Set the title and desc of this download, to be displayed in notifications.
 */
private void downloadFile(){
    DownloadManager.Request request=new
DownloadManager.Request(Uri.parse("https://i.pinimg.com/originals/b0/25/be/b025be365c934c
8ae4eda1fee4f82153.jpg"));
    request.setTitle("Kitchen_Converter_Table")
            .setDescription("File is downloading...")
            .setDestinationInExternalFilesDir(this,
                    Environment.DIRECTORY_DOWNLOADS,fileName)

.setNotificationVisibility(DownloadManager.Request.VISIBILITY_VISIBLE_NOTIFY_COMPLETED);
    // Enqueue the download.The download will start automatically once the download
manager is ready
    // to execute it and connectivity is available.
    downLoadId=downloadManager.enqueue(request);
}
```

Lien URL

C'est ici que vous définissez le lien à partir duquel l'image / le fichier est téléchargé :

```
DownloadManager.Request(Uri.parse("https://i.pinimg.com/originals/b0/25/be/b025be36
5c934c8ae4eda1fee4f82153.jpg"));
```

Étape finale : un bouton ou un autre support pour invoquer l'action de téléchargement.

```
/**
 * Handle button clicks : DOWNLOAD button
 * @param view
 */
public void clickView(View view){
    switch (view.getId()){
        case R.id.download:
            downloadFile();
            Toast.makeText(getBaseContext(), R.string.msg_downloading,
Toast.LENGTH_SHORT).show();
            break;
    }
}
```

Selon vos préférences, vous pouvez définir *onClick* ou *focus* sur un bouton, une image ou autre.

CODE Activity.java (complet)

```
package fr.afpa.kitchen_converter;

import android.Manifest;
import android.app.DownloadManager;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.net.Uri;
import android.os.Build;
import android.os.Bundle;
import android.os.Environment;
import android.util.Log;
import android.view.View;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import fr.afpa.kitchen_converter.model.UnitManager;

public class TableActivity extends AppCompatActivity {

    // DECLARATIONS of the variables ================================== //
    private final static String TAG = "TableActivity";              // TAG : onActivityCheckupSpy
    private DownloadManager downloadManager;
    private String fileName=null;
    private long downLoadId;

    // onCREATE ======================================================= //
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.table);
        initializeDownloadManager();

        Intent iTable = getIntent();

        // DATE BASE
        UnitManager.getAll(this); // test if DB works
    }

    // FUNCTIONS ====================================================== //
    /**
     * STORAGE PERMISSION
     * @return
     */
    public  boolean isStoragePermissionGranted() {
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
            if (checkSelfPermission(android.Manifest.permission.WRITE_EXTERNAL_STORAGE)
                    == PackageManager.PERMISSION_GRANTED) {
                Log.v(TAG,"Permission is granted");
                return true;
            } else {
                Log.v(TAG,"Permission is revoked");
                ActivityCompat.requestPermissions(this, new
String[]{Manifest.permission.WRITE_EXTERNAL_STORAGE}, 1);
                return false;
            }
        }
        else { //permission is automatically granted on sdk<23 upon installation
            Log.v(TAG,"Permission is granted");
            return true;
        }
    }

    /**
     * Initialize download manager
     */
    private void initializeDownloadManager() {
        downloadManager= (DownloadManager) getSystemService(DOWNLOAD_SERVICE);
        fileName="Kitchen_Converter_Table.jpg";
    }
```

```
    /**
     * Set the title and desc of this download, to be displayed in notifications.
     */
    private void downloadFile(){
        DownloadManager.Request request=new
DownloadManager.Request(Uri.parse("https://i.pinimg.com/originals/b0/25/be/b025be365c934c8ae4eda1fe
e4f82153.jpg"));
        request.setTitle("Kitchen_Converter_Table")
                .setDescription("File is downloading...")
                .setDestinationInExternalFilesDir(this,
                        Environment.DIRECTORY_DOWNLOADS,fileName)

.setNotificationVisibility(DownloadManager.Request.VISIBILITY_VISIBLE_NOTIFY_COMPLETED);
        // Enqueue the download.The download will start automatically once the download manager is
ready
        // to execute it and connectivity is available.
        downLoadId=downloadManager.enqueue(request);
    }

    /**
     * Handle button clicks : DOWNLOAD button
     * @param view
     */
    public void clickView(View view){
        switch (view.getId()){
            case R.id.download:
                downloadFile();
                Toast.makeText(getBaseContext(), R.string.msg_downloading,
Toast.LENGTH_SHORT).show();
                break;
        }
    }

    /**
     * BACK button
     * @param view
     */
    public void back(View view) {
        Log.i(TAG, "back");
        TableActivity.this.finish();
    }

}
```

CODE Activity.xml (complet)

Dans mon cas, le nom de ce fichier est : table.xml (voir capture d'écran sur la premier page, dossier res/*layout*).

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/ver">
    <ImageView
        android:id="@+id/headerimg"
        android:contentDescription="@string/header_img"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="521dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="@+id/scrollviewtab"
        app:srcCompat="@drawable/kitchen_converter2" />
    <ScrollView
        android:id="@+id/scrollviewtab"
        android:layout_width="410dp"
        android:layout_height="486dp"
        android:layout_centerVertical="true"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.498"
        tools:ignore="SpeakableTextPresentCheck">
        <ImageView
            android:id="@id/imgtable"
            android:contentDescription="@string/string_table_img"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:adjustViewBounds="true"
            android:orientation="vertical"
            app:srcCompat="@drawable/imgtable"
            tools:ignore="ImageContrastCheck" />
    </ScrollView>
    <Button
        android:id="@+id/download"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="16dp"
        android:fontFamily="@font/bauhs_93"
        android:onClick="clickView"
        android:clickable="true"
        android:focusable="true"
        android:text="@string/download"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/scrollviewtab"
        tools:ignore="TextContrastCheck,UsingOnClickInXml" />
    <Button
        android:id="@+id/back"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="64dp"
        android:fontFamily="@font/bauhs_93"
        android:onClick="back"
        android:clickable="true"
        android:focusable="true"
        android:text="@string/back"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/scrollviewtab"
        tools:ignore="TextContrastCheck,UsingOnClickInXml" />
</androidx.constraintlayout.widget.ConstraintLayout>
```