TP : Appel d’autres applications

Introduction

Sous Android, lorsqu’une application souhaite effectuer une tâche qui est du domaine d'une autre application, il est d’usage de déléguer cette tâche à cette autre application (Appeler la synthèse vocale, Google Map ou encore un lecteur de code barre. Nous allons aujourd’hui étudier comment appeler une application tierce depuis notre propre application Android.

Objectif du TP

Dans ce TP, nous allons simplement créer une application qui nous permettra d'accéder à la galerie de notre téléphone et de choisir une image afin de l’afficher dans notre application.

A vous de jouer

Tout d’abord, nous allons créer un projet Android Studio. (Tp réalisé avec minimum SDK : API 19)

Création du layout

Nous allons créer un layout contenant au minimum un “`Button`” et un “`ImageView`”. Ensuite, nous allons définir une méthode “`onClick`” au bouton (ici `onClick=”parcourir”`) et un id à l’`imageView` (ici `id=”@+id/imageView`)

activity_main.xml × MainActivity.java ×

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="16dp"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp"
    android:orientation="vertical"
    tools:context="com.appelappli.android.appelappli.MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Choisissez une image : " />

    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Parcourir"
        android:onClick="parcourir"
        />

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
</LinearLayout>
```

Design Text

Main

Dans le `MainActivity.java`, nous allons écrire la ligne suivante :

```
private static final int GALLERY_PICTURE = 42;
```

Cela nous permettra d'identifier notre appel d'application par la suite.

Ensuite, nous allons créer la méthode "`parcourir`" qui se déclenchera lorsque l'on cliquera sur notre bouton. Cette méthode crée un `Intent` définissant une action `ACTION_PICK` qui permet de récupérer une donnée de type image (grâce au `setType(String type)` type étant un MIME), puis démarre l'application qui correspond à la demande.

```
public void parcourir(View view){
    Intent photoPickerIntent = new Intent(Intent.ACTION_PICK);
    photoPickerIntent.setType("image/*");
    startActivityForResult(photoPickerIntent, GALLERY_PICTURE);

}
```

Il nous faut maintenant créer la méthode traitant les données récupérées. Nous allons pour cela surcharger la méthode “`onActivityResult`”.
Tout d’abord, il faut vérifier que le résultat que l’on reçoit est bien lié à l’appel de notre méthode “`parcourir`” (`requestCode == GALLERY_PICTURE`) et que les données ne sont pas nulles.
Il faut savoir que notre `Intent.ACTION_PICK` nous renvoie une Uri (une chaîne de caractères permettant d’identifier la ressource choisie)
Nous allons donc récupérer l’uri, la convertir en `Bitmap` afin de l’insérer dans notre `imageView`.

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (requestCode == GALLERY_PICTURE && resultCode == RESULT_OK && data != null) {
        Uri uri = data.getData();

        try{
            Bitmap bitmap = MediaStore.Images.Media.getBitmap(getContentResolver(), uri);

            ImageView imageView = (ImageView) findViewById(R.id.imageView);
            imageView.setImageBitmap(bitmap);
        }catch(IOException e){
            e.printStackTrace();
        }

    }
}
```

Une fois ceci recopié, nous pouvons tester l’application.

Complément d’informations

Tutoriel scan de code barre

http://tutorielsandroid.com/lire-un-qr-code-ou-un-code-barre-dans-son-application-android/

Tutoriel appel synthèse vocale

http://www.androidhive.info/2014/07/android-speech-to-text-tutorial/

Les différentes actions des intents

(voir Standard Activity Actions) :
https://developer.android.com/reference/android/content/Intent.html

Exemples de MIME :

image/jpeg
audio/mpeg4-generic
text/html

audio/mpeg
audio/aac
audio/wav
audio/ogg
audio/midi
audio/x-ms-wma
video/mp4
video/x-msvideo
video/x-ms-wmv
image/png
image/jpeg
image/gif
.xml ->text/xml
.txt -> text/plain
.cfg -> text/plain
.csv -> text/plain
.conf -> text/plain
.rc -> text/plain
.htm -> text/html
.html -> text/html
.pdf -> application/pdf
.apk -> application/vnd.android.package-archive