

**UVS**  
UNIVERSITE VIRTUELLE DU SENEGAL

android 

**DEVELOPPEMENT MOBILE ANDROID**

**Elément d'Interaction et Groupes**

INSA BADJI Doctorant à l'Université de Thiès / Tuteur à l'UVS



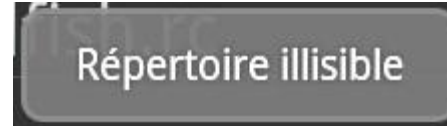
# Séquence 5: les notifications et intents

- Objectif
- Plan:
  - Les Toasts
  - Les Boîtes de dialogues
  - Navigation dans les pages
  - Les intents
  - Les intents prédéfinis



# Notifications: La classe Toast

- Texte qui apparaît en premier plan puis disparaît au bout d'un temps donné



- **Création d'un Toast**

- `Toast.makeText(Context, String, int)` renvoie l'objet de classe Toast créé.
- Le premier paramètre est l'activité
- Le deuxième paramètre est le message à afficher
- Le dernier paramètre indique la durée d'affichage les seules valeurs possibles sont :  
Toast.LENGTH\_SHORT (2 secondes) ou Toast.LENGTH\_LONG (5 secondes).

- **Positionnement d'un Toast**

- `setGravity(int, int, int)` appelée avant l'affichage par `show` pour indiquer où s'affichera le message.
- Le premier paramètre sert à placer le message par rapport à l'écran. Il peut prendre l'une des valeurs définies dans la classe Gravity soit : Gravity. (TOP, BOTTOM, LEFT, RIGHT, CENTER\_VERTICAL, FILL\_VERTICAL, CENTER\_HORIZONTAL, FILL\_HORIZONTAL, CENTER, FILL).
- Les deux paramètres suivants indiquent le décalage (en pixels).

- **Affichage d'un Toast**

- `show()` affiche le message pour la durée définie lors de sa création.

# Notifications: La classe AlertDialog

- Style plus classique des boîtes de dialogue. Un AlertDialog s'ouvre, prend le focus et reste affiché tant que l'utilisateur ne le ferme pas
- **Pour créer un AlertDialog:**
  - Utiliser la classe Builder offrant un ensemble de méthodes permettant de configurer un AlertDialog. Méthodes renvoie le Builder afin de faciliter le chaînage des appels.
  - À la fin, il suffit d'appeler la méthode show() de l'objet Builder pour afficher la boîte de dialogue.
- **Méthodes de configuration de Builder :**
  - setMessage() permet de définir le "corps" de la boîte de dialogue
  - setTitle() et setIcon() permettent de configurer le texte et/ou l'icône
  - setPositiveButton(), setNegativeButton() et setNeutralButton() permettent d'indiquer les boutons qui apparaîtront en bas de la boîte de dialogue, leur emplacement latéral (respectivement, à gauche, au centre ou à droite), leur texte et le code qui sera appelé lorsqu'on clique sur un bouton (en plus de refermer la boîte de dialogue).

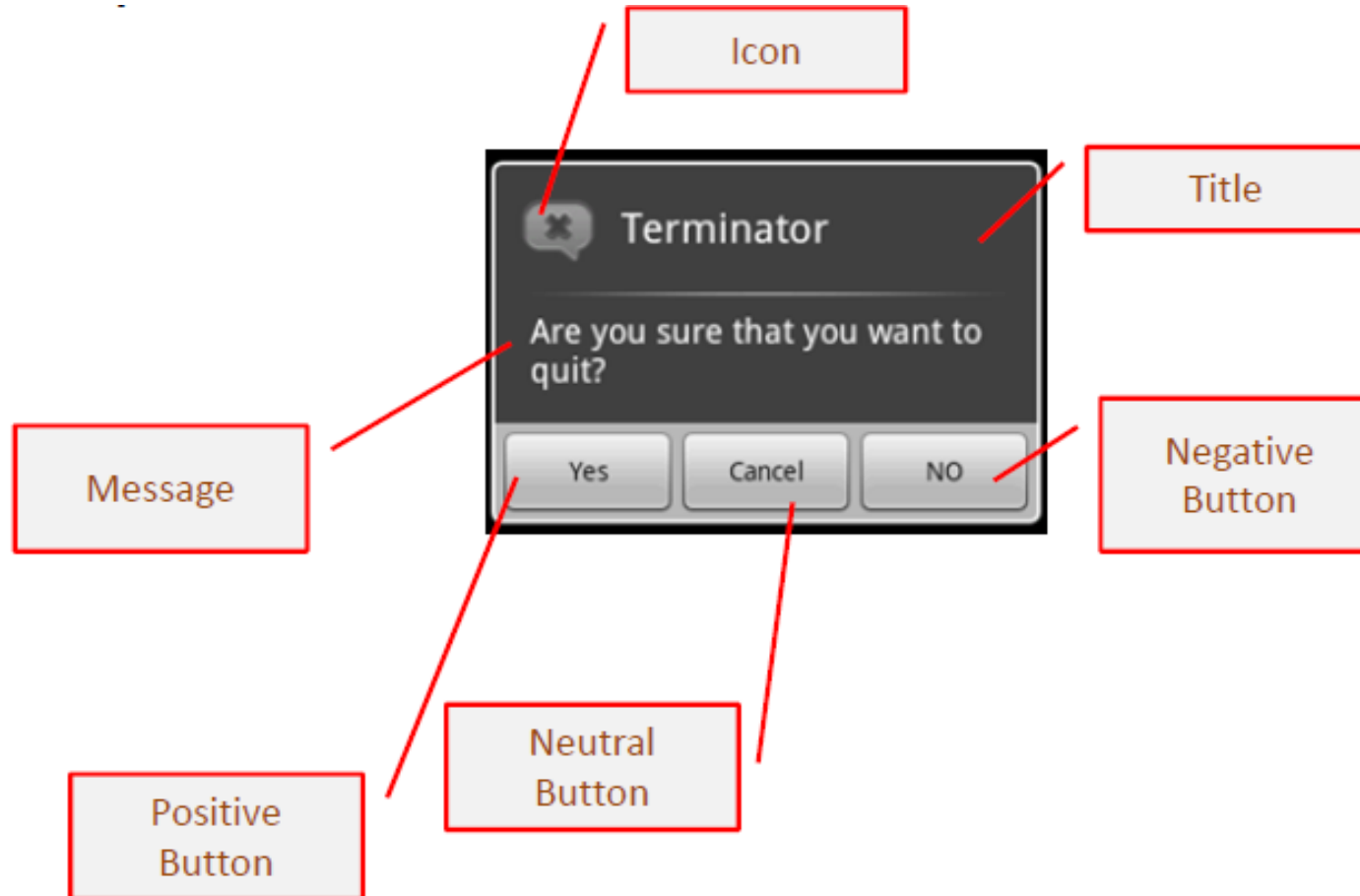
# Notifications: : Exemple de Toast

- Un **Toast** est un message qui apparaît et disparaît
- Il est impossible de savoir si l'utilisateur l'a vu ou pas
- Configuration:
  - Un **String** contenant le message
  - Une durée
    - **Toast.LENGTH\_LONG** ou **Toast.LENGTH\_SHORT**
- Il est aussi possible de donner une **View** de votre choix en paramètre

# Toast : exemple

```
Toast.makeText(MainActivity.this,  
    "C'est vous qui voyez!!!",  
    Toast.LENGTH_SHORT).show();
```

# Boites de dialogue



# Exemple: AlertDialog

```
<?xml version="1.0" encoding="utf-8" ?>
<RelativeLayout 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"
    tools:context=".MainActivity">

    <Button
        android:id="@+id/btnCliquer"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:layout_marginTop="16dp"
        android:text="Cliquer" />

    <EditText
        android:id="@+id/txtmsg"
        android:layout_width="232dp"
        android:layout_height="wrap_content"
        android:layout_alignBaseline="@+id/btnCliquer"
        android:layout_alignBottom="@id/btnCliquer"
        android:layout_alignParentRight="true"
        android:layout_marginRight="32dp"
        android:layout_marginBottom="2dp"
        android:ems="10"
        android:hint="Cliquer sur le Bouton" />

</RelativeLayout>
```



# Exemple: AlertDialog

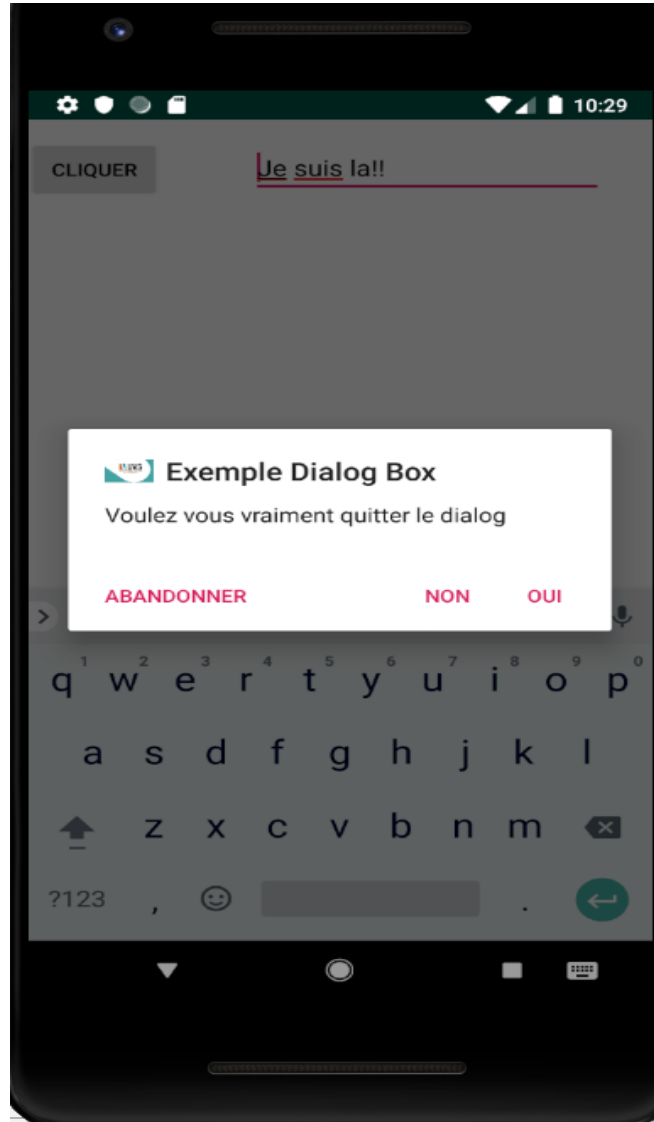
```
package com.example.myappuvs;

import android.app.Activity;
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TabHost;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends Activity {
    Button cliquerBouton;
    EditText txtMsg;
    String msg;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        txtMsg = (EditText) findViewById(R.id.txtmsg);
        cliquerBouton =
            (Button) findViewById(R.id.btnCliquer);
        cliquerBouton.setOnClickListener(new
            View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    AlertDialog diaBox = createDialogBox();
                    diaBox.show();
                    txtMsg.setText("Je suis la!!!");
                }
            });
    }
}
```

# Exemple: AlertDialog



```
protected AlertDialog createDialogBox() {
    AlertDialog myDialogBox = new AlertDialog.Builder(this)
        .setTitle("Exemple Dialog Box")
        .setMessage("Voulez vous vraiment quitter le dialog")
        .setIcon(R.drawable.logo_uvs)
        .setPositiveButton("Oui", new
DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                msg = "oui" +Integer.toString(which);
                txtMsg.setText(msg);
            }
        })
        .setNegativeButton("Abandonner", new
DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                msg = "Abandonner" +Integer.toString(which);
                txtMsg.setText(msg);
            }
        })
        .setNeutralButton("Non", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int
which) {
                msg = "Non" +Integer.toString(which);
                txtMsg.setText(msg);
            }
        })
        .create();
    return myDialogBox;
}
```

# Principe des intents

- Les *Intents* permettent de gérer l'envoi et la réception de messages afin de faire coopérer les applications. Le but des *Intents* est de déléguer une action à un autre composant, une autre application ou une autre activité de l'application courante. Un objet **Intent** contient les informations suivantes:
  - le nom du composant ciblé (facultatif)
  - l'action à réaliser, sous forme de chaîne de caractères
  - les données: contenu MIME et URI
  - des données supplémentaires sous forme de paires de clé/valeur
  - une catégorie pour cibler un type d'application
  - des drapeaux (informations supplémentaires)
- On peut envoyer des *Intents* informatifs pour faire passer des messages. Mais on peut aussi envoyer des *Intents* servant à lancer une nouvelle activité

# Intents pour une nouvelle activité

- Plusieurs façons de créer l'objet de type *Intent* qui permettra de lancer une nouvelle activité. Si l'on passe la main à une activité interne à l'application, on peut créer l'Intent et passer la classe de l'activité ciblée par l'Intent:

```
Intent login = new Intent(this, SeLoguer.class);  
startActivity(login);
```

# Intents pour une nouvelle activité

S'il s'agit de passer la main à une autre application, on donne au constructeur de l'Intent les données et l'URI cible: l'OS est chargé de trouver une application pouvant répondre à l'Intent.

```
Button b = (Button) findViewById(R.id.Button01);
b.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {
        Uri telnumber = Uri.parse("tel:770000000");
        Intent call = new Intent(Intent.ACTION_CALL, telnumber);
        startActivity(call);
    }
});
```

**Sans oublier la permission dans le Manifest**

# Intent avec résultat attendu en retour

- Lorsque le bouton *retour* est pressé, l'activité courante prend fin et revient à l'activité précédente. Cela permet par exemple de terminer son appel téléphonique et de revenir à l'interface ayant initié l'appel.
- Au sein d'une application, une activité peut vouloir récupérer un code de retour de l'activité "enfant". On utilise pour cela la méthode **startActivityResult** qui envoie un code de retour à l'activité enfant.  

```
public void startActivityResult (Intent intent, int requestCode)
```
- Lorsque l'activité parent reprend la main, il devient possible de filtrer le code de retour dans la méthode **onActivityResult** pour savoir si l'on revient ou pas de l'activité enfant.
- ```
protected void onActivityResult(int requestCode, int resultCode, Intent data)
```

# Retour d'une activité

- L'appel d'un *Intent* devient donc dans l'activité parent:

```
public void onCreate(Bundle savedInstanceState) {  
    Intent login = new Intent(getApplicationContext(),  
        SaisirNumeroTelephone.class);  
    startActivityForResult(login,48);  
    ...  
}
```

- Le filtrage dans la classe parent permet de savoir qui avait appelé cette activité enfant:

```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
    if (requestCode == 48)  
        Toast.makeText(this,  
            "Code de requête récupéré (je sais d'ou je viens)",  
            Toast.LENGTH_LONG).show();  
}  
}
```

# Résultat d'une activité

- Il est aussi possible de définir un résultat d'activité, avant d'appeler explicitement la fin d'une activité avec la méthode **finish()**. Dans ce cas, la méthode **setResult** permet d'enregistrer un code de retour qu'il sera aussi possible de filtrer dans l'activité parente.

- Dans l'activité enfant, on met donc:

```
Button terminer = (Button)findViewById(R.id.terminer);
terminer.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {
        setResult(RESULT_OK, getIntent());
        finish();
    }
});
```



# Résultat d'une activité

- Et la classe parente peut filtrer ainsi:

```
protected void onActivityResult(int requestCode, int resultCode, Intent data){  
    if (requestCode == 48)  
        Toast.makeText(this, "Code de requête récupéré (je sais d'ou je viens)",  
            Toast.LENGTH_LONG).show();  
    if (requestCode == 50)  
        Toast.makeText(this, "Code de retour ok (on m'a renvoyé le bon code)",  
            Toast.LENGTH_LONG).show();  
}
```

# Ajout d'informations

- Les *Intent* permettent de transporter des informations à destination de l'activité cible. On appelle ces informations des *Extra*: les méthodes permettant de les manipuler sont **getExtra** et **putExtra**.
- Lorsqu'on prépare un Intent et que l'on souhaite ajouter une information de type "clef -> valeur" on procède ainsi:

# Ajout d'informations

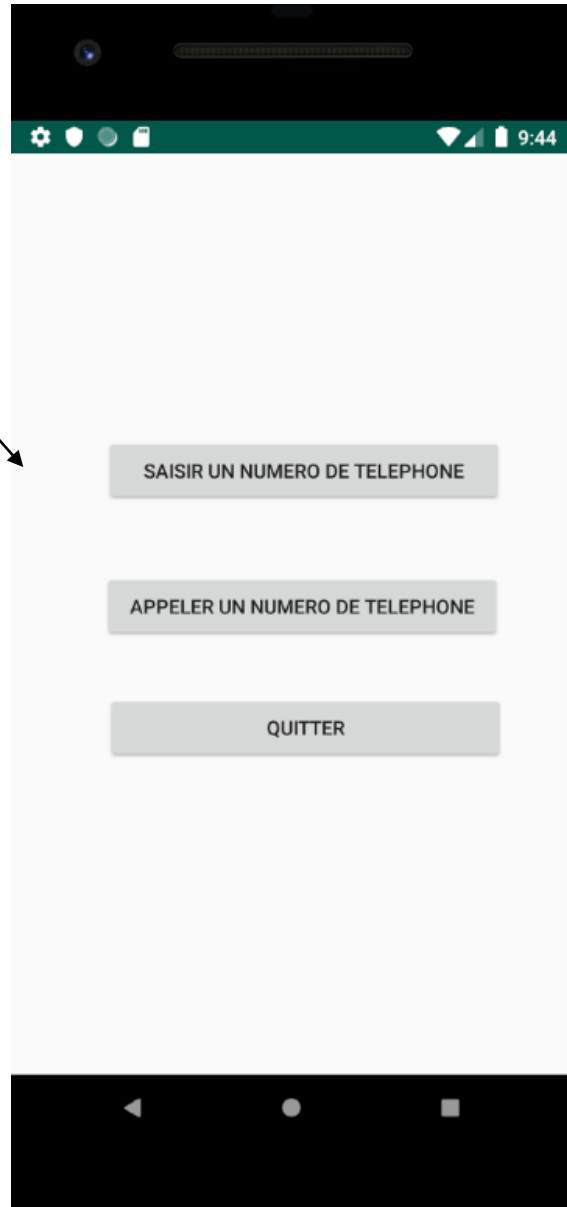
```
Intent callactivity2 = new Intent(getApplicationContext(),  
                                     Activity2.class);  
callactivity2.putExtra("login", "Lamane");  
startActivity(callactivity2);
```

- Du côté de l'activité recevant l'Intent, on récupère l'information de la manière suivante:

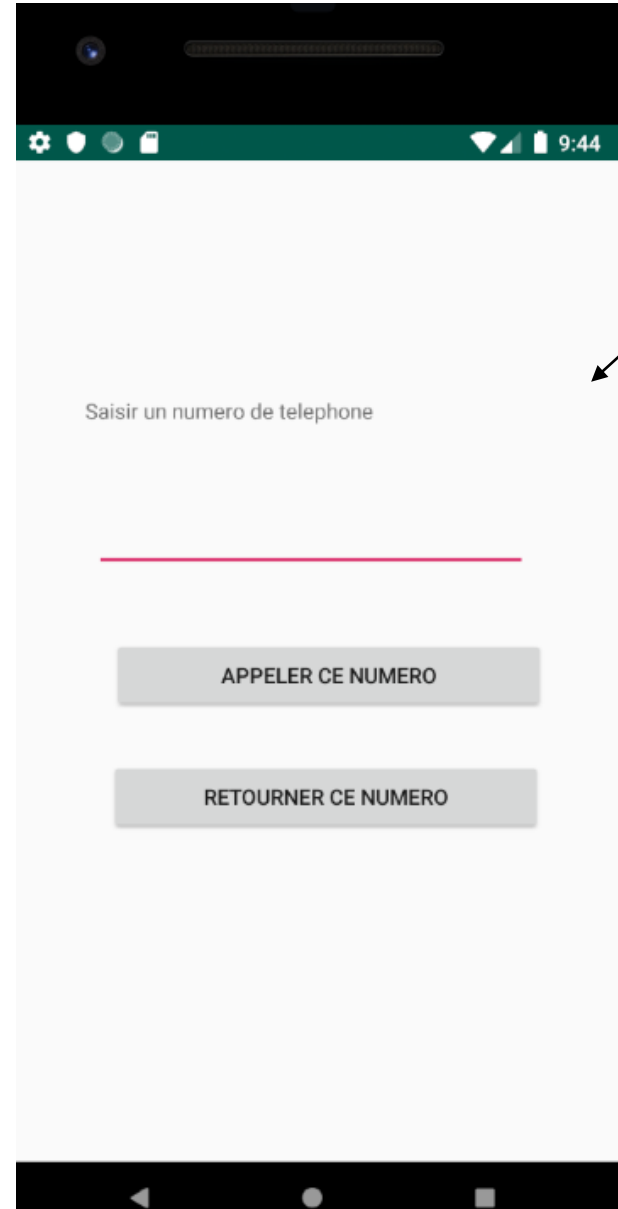
```
Bundle extras = getIntent().getExtras();  
String s = new String(extras.getString("login"));
```

# Exemple

Activite  
Pricipale



Activite  
Appeler



# Pour l'activité principale

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
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"
tools:context=".MainActivity">

<Button
    android:id="@+id/btnNumeTel"
    android:layout_width="285dp"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true"
    android:layout_marginLeft="72dp"
    android:layout_marginTop="203dp"
    android:text="Saisir un numero de telephone" />
```

```
<Button
    android:id="@+id/btnAppeler"
    android:layout_width="285dp"
    android:layout_height="wrap_content"

    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true"
    android:layout_marginLeft="71dp"
    android:layout_marginTop="300dp"
    android:text="Appeler un numero de
telephone" />
<Button
    android:id="@+id/btnQuitter"
    android:layout_width="285dp"
    android:layout_height="wrap_content"

    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true"
    android:layout_marginLeft="73dp"
    android:layout_marginTop="387dp"
    android:text="Quitter" />
</RelativeLayout>
```

# Pour l'activité Appeler

```
<EditText
```

```
    android:id="@+id/editText"  
    android:layout_width="285dp"  
    android:layout_height="wrap_content"  
    android:layout_alignParentLeft="true"  
    android:layout_alignParentTop="true"  
    android:layout_marginLeft="62dp"  
    android:layout_marginTop="226dp"  
    android:ems="10"  
    android:inputType="textPersonName" />
```

```
<TextView
```

```
    android:id="@+id/textView"  
    android:layout_width="285dp"  
    android:layout_height="wrap_content"  
    android:layout_alignParentLeft="true"  
    android:layout_alignParentTop="true"  
    android:layout_marginLeft="56dp"  
    android:layout_marginTop="155dp"  
    android:text="Saisir un numero de  
telephone" />
```

```
<Button
```

```
    android:id="@+id/btnAppeler"  
    android:layout_width="285dp"  
    android:layout_height="wrap_content"  
    android:layout_alignParentLeft="true"  
    android:layout_alignParentTop="true"  
    android:layout_marginLeft="74dp"  
    android:layout_marginTop="315dp"  
    android:text="Appeler ce numero" />
```

```
<Button
```

```
    android:id="@+id/btnQuitter"  
    android:layout_width="285dp"  
    android:layout_height="wrap_content"  
    android:layout_alignParentLeft="true"  
    android:layout_alignParentTop="true"  
    android:layout_marginLeft="72dp"  
    android:layout_marginTop="395dp"  
    android:text="Retourner ce numero" />
```

# Pour l'activité principale

```
package com.example.myappuvs;
import android.app.Activity;
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TabHost;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends Activity {
    Button applerBouton, numeroBouton, quitterBouton;
    EditText txtMsg;
    String msg;
    @Override
    protected void onCreate( Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        applerBouton = (Button)findViewById(R.id.btnAppeler);
        applerBouton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent appelerIntent = new Intent(getApplicationContext(),
AppelActivity.class);
                startActivity(appelerIntent);
            }
        });
    }
};
```

```

numeroBouton = (Button)findViewById(R.id.btnNumeTel);
numeroBouton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent saisirIntent = new Intent(getApplicationContext(), AppelActivity.class);
        startActivityForResult(saisirIntent, 1);
    }
});
quitterBouton = (Button)findViewById(R.id.btnQuitter);
quitterBouton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        finish();
    }
});
/* txtMsg = (EditText)findViewById(R.id.txtmsg);
cliquerBouton = (Button)findViewById(R.id.btnCliquer);
cliquerBouton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        AlertDialog diaBox = createDialogBox();
        diaBox.show();
        txtMsg.setText("Je suis la!!");
    }
});*/
}

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data){
    if(requestCode == 1){
        if(resultCode == RESULT_OK){
            String filiere = data.getStringExtra("NumTel");
            Toast.makeText(getApplicationContext(), "VOtre numero est "+filiere, Toast.LENGTH_LONG).show();

        } else if (resultCode == RESULT_CANCELED){
            Toast.makeText(getApplicationContext(), "Operation Annule ", Toast.LENGTH_LONG).show();
        }
    }
}
}

```



# Pour Android Manifest

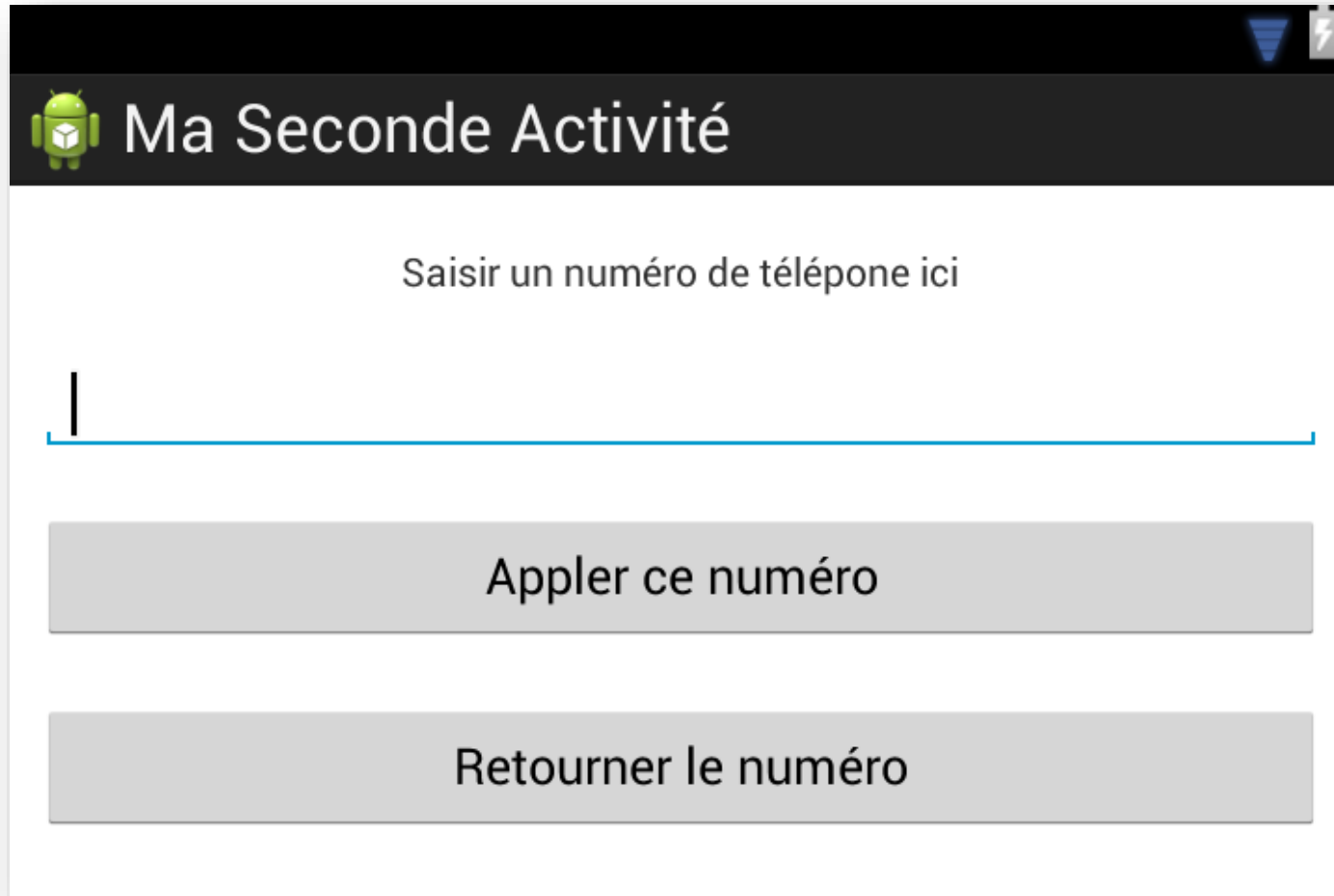
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  package="com.example.myappuvs">
  <uses-permission
android:name="android.permission.CALL_PHONE"/>
  <application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/AppTheme">

    <activity android:name=".MainActivity">
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />

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

</manifest>
```

# Pour la seconde activité



The screenshot shows an Android application window with a black title bar. On the left of the title bar is the Android logo, and on the right are system icons for signal strength and battery. The main content area has a white background. At the top, the text 'Ma Seconde Activité' is displayed in white. Below this, the instruction 'Saisir un numéro de téléphone ici' is centered. A text input field with a blue underline contains a single vertical bar cursor. Below the input field are two grey buttons with black text: 'Appeler ce numéro' and 'Retourner le numéro'.

Ma Seconde Activité

Saisir un numéro de téléphone ici

|

Appeler ce numéro

Retourner le numéro

# Pour la seconde Activité Appeler

```
package com.example.myappuvs;

import android.app.Activity;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

public class AppelActivity extends Activity {
    EditText numTelEditText;
    Button retourBouton, appelBouton;
    @Override
    protected void onCreate( Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.appel_activity);

        numTelEditText = (EditText)findViewById(R.id.editText);
        appelBouton = (Button)findViewById(R.id.btnAppeler);
        appelBouton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Uri numTel = Uri.parse("tel:" +numTelEditText.getText());
                Intent appelerIntent = new Intent(Intent.ACTION_CALL, numTel);
                startActivity(appelerIntent);
            }
        });
    }
}
```

```
retourBouton = (Button)findViewById(R.id.btnQuitter);
retourBouton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if(numTelEditText.getText().toString().trim().length()>0){
            getIntent().putExtra( name: "NumTel", numTelEditText.getText().toString());
            setResult(RESULT_OK, getIntent());
        } else {
            setResult(RESULT_CANCELED, getIntent());
        }
        finish();
    }
});
```

# Intentions prédéfinies

- **ACTION\_MAIN**: action principale
- **ACTION\_VIEW**: visualiser une donnée
- **ACTION\_ATTACH\_DATA**: attachement de donnée
- **ACTION\_EDIT**: Edition de donnée
- **ACTION\_PICK**: Choisir un répertoire de donnée
- **ACTION\_CHOOSER**: menu de choix pour l'utilisateur – EXTRA\_INTENT contient l'Intent original, EXTRA\_TITLE le titre du menu
- **ACTION\_GET\_CONTENT**: obtenir un contenu suivant un type MIME
- **ACTION\_SEND**: envoyé un message (EXTRA\_TEXT|EXTRA\_STREAM) à un destinataire non spécifié

# Intentions prédéfinies

- **ACTION\_SEND\_TO**: on spécifie le destinataire dans l'URI
- **ACTION\_INSERT**: on ajoute un élément vierge dans le répertoire spécifié par l'URI
- **ACTION\_DELETE**: on supprime l'élément désigné par l'URI
- **ACTION\_PICK\_ACTIVITY**: menu de sélection selon l'EXTRA\_INTENT mais ne lance pas l'activité
- **ACTION\_SEARCH**: effectue une recherche etc...

**Fin de la  
séquence 3**