# Modern Related Technology on Mobile Devices

Lec4: Intent

Ekarat Rattagan, PhD

# Intents

An Android application could include any number of activities.

- An *activity* uses the *setContentView(...)* method to expose a single UI from which a number of actions could be performed.
- Activities are independent of each other; however they usually cooperate exchanging data and actions.
- Typically, one of the activities is designated as the first one *(main)* that should be presented to the user when the application is launched.
- Moving from one activity to another is accomplished by asking the current activity to execute an ***intent***.

# Intents

- An asynchronous message.
- Bind individual components to each other at runtime.
- Think of them as the messengers that request an action from other components, whether the component belongs to your app or another.

# Intents

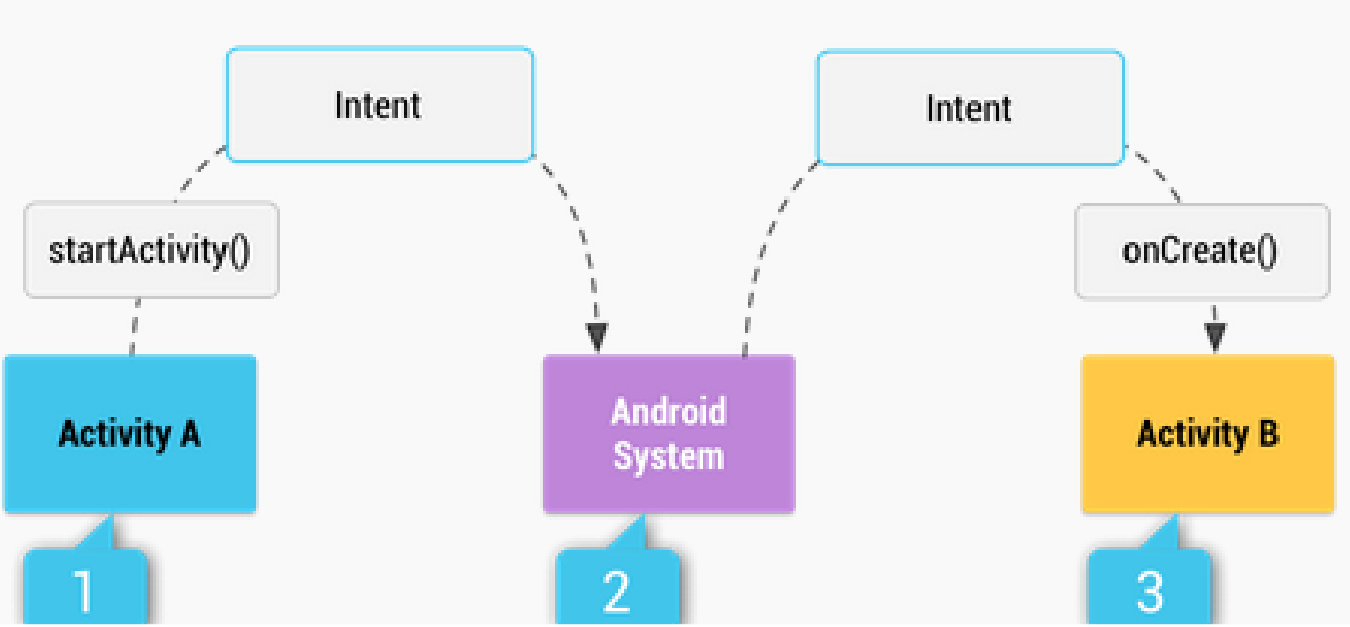



**Figure 1.** Illustration of how an implicit intent is delivered through the system to start another activity:

**[1]** *Activity A* creates an Intent with an action description and passes it to startActivity().
**[2]** The Android System searches all apps for an intent filter that matches the intent. When a match is found,
**[3]** the system starts the matching activity (*Activity B*) by invoking itsonCreate() method and passing it the Intent.

# Intents

| Public constructors |
| --- |
| `Intent()`<br>Create an empty intent. |
| `Intent(Intent o)`<br>Copy constructor. |
| `Intent(String action)`<br>Create an intent with a given action. |
| `Intent(String action, Uri uri)`<br>Create an intent with a given action and for a given data url. |
| `Intent(Context packageContext, Class<?> cls)`<br>Create an intent for a specific component. |
| `Intent(String action, Uri uri, Context packageContext, Class<?> cls)`<br>Create an intent for a specific component with a specified action and data. |

# Type of Intents

1. **Implicit intents**

   Do not directly specify the Android components which should be called,

   e.g., Intent int1 = new Intent(Intent.ACTION_VIEW,
   Uri.parse("http://www.google.com"));
   startActivity(int1);

2. **Explicit intents**

   Specify the Android component to be called, e.g., Activity B and A,

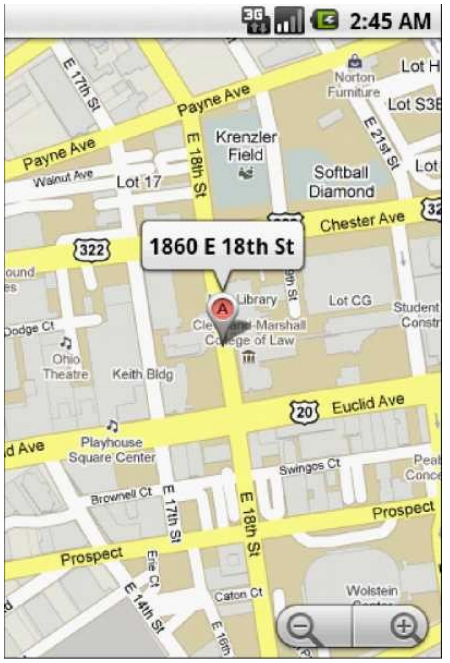
   e.g., Intent intent = new Intent(this,ActivityB.class);
   startActivity(intent);

# Action/Data

The main arguments of an Intent are:

1. **Action** The built-in action to be performed, such as ACTION_VIEW, ACTION_EDIT, ACTION_MAIN, etc.
2. **Data** The primary data to operate on, such as a phone number to be called (expressed as a **Uri**).

# Intents




## Example 1 : **Using Standard Actions**

### Geo Mapping an Address

Provide a geoCode expression holding a street address (or place, such as 'golden gate ca' )

Replace spaces with '+'.

```
String geoCode = "geo:0,0?q=18 60+east+18th+street+cleveland+oh";
Intent intent = new Intent(Intent.ACTION VIEW, Uri. parse(geoCode));
startActivity(intent);
```

Modify the Manifest adding the following requests:

```
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.INTERNET" />
```

# Intents


## Example 2 : Using Standard Actions

**Geo Mapping Coordinates (latitude, longitude)**

Provide a geoCode holding latitude and longitude (also an addittional zoom '**?z=xx**' with xx in range 1..23)

```
String geoCode = "geo:41.5020952,-81.6789717";

Intent intent = new Intent(Intent.ACTION_VIEW,
                           Uri.parse(geoCode));
startActivity(intent);
```

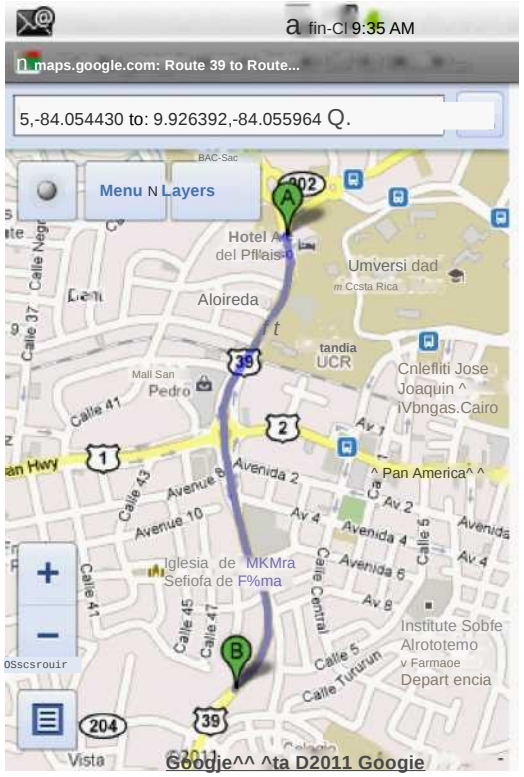
Modify the Manifest adding the following requests:

```
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.INTERNET" />
```

# Intents

**Examples 3**

```
// Getting driving directions: how to go from loaction A to location B?

Intent intent = new Intent(android.content.Intent.ACTION_VIEW,

Uri.parse("http://maps.google.com/maps?saddr=9.938083,-84.054430&daddr=9.926392,-84.055964"));

startActivity(intent);
```



# Intents


## Example 4 : Using Standard Actions

### Geo Mapping - Google StreetView

geoCode Uri structure:

google.streetview:cbll=lat,lng&cbp=1,*yaw,,pitch,zoom&mz=mapZoom*

*Reference:* http://developer.android.com/guide/appendix/g-app-intents.html

```
String geoCode =
google.streetview:cbll=41.5020952,-81.6789717&cbp=1,270,,45,1&mz=1";
```

```
Intent intent = new Intent(Intent.ACTION_VIEW, Uri.parse(geoCode));
startActivity(intent);
```

Modify the Manifest adding the following requests:

```
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.INTERNET" />
```

# Action/Data

```
Intent  myActivity = new Intent (action, data);

startActivity (myActivity);
```

Built-in or user-created activity

Primary data (as an URI)
tel://
http://
sendto://

# Action/Data

**Examples** of **action/data** pairs are:

**ACTION_DIAL** ***tel:123***
Display the phone dialer with the given number filled in.

**ACTION_VIEW** ***http://www.google.com***
Show Google page in a browser view. Note how the VIEW action does what is considered the most reasonable thing for a particular URI.

**ACTION_EDIT** ***content://contacts/people/2***
Edit information about the person whose identifier is ”2”.

**ACTION_VIEW** ***content://contacts/people/2***
Used to start an activity to display *2-nd* person.

**ACTION_VIEW** ***content://contacts/* people/**
Display a list of people, which the user can browse through. Selecting a particular person to view would result in a new intent

# Action/Data

## Built-in Standard Actions

List of standard actions that Intents can use for launching activities (usually through *startActivity(Intent).*

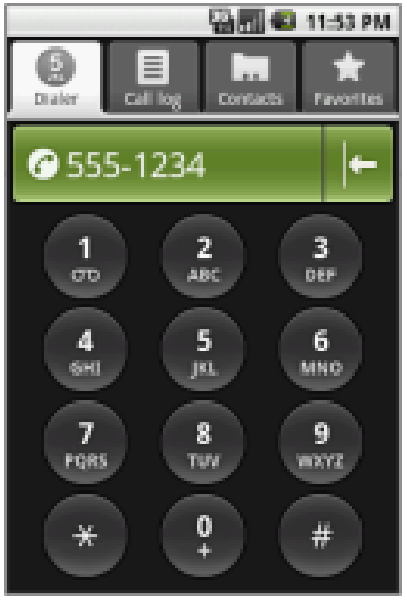
| | |
|---|---|
| **ACTION_MAIN** | ACTION_ANSWER |
| ACTION_VIEW | ACTION_INSERT |
| ACTION_ATTACH_DATA | ACTION_DELETE |
| **ACTION_EDIT** | ACTION_RUN |
| ACTION_PICK | ACTION_SYNC |
| ACTION_CHOOSER | ACTION_PICK_ACTIVITY |
| ACTION_GET_CONTENT | **ACTION_SEARCH** |
| ACTION_DIAL | **ACTION_WEB_SEARCH** |
| **ACTION_CALL** | ACTION_FACTORY_TEST |
| ACTION_SEND | |
| **ACTION_SENDTO** | |

# Intents

**Example 5**

Display the phone dialer with the given number filled in.

```
Intent myActivity2 = new Intent (Intent.ACTION_DIAL,
                                 Uri.parse( "tel:555-1234"));

startActivity(myActivity2);
```



# Send values between Activities

Intents - Secondary Attributes

In addition to the primary *action/data* attributes, there are a number of *secondary attributes* that you can also include with an intent, such as:
putExtra: Send data between activities

**Activity A**

```
Intent intent = new Intent(AdultTeeth.this, MainScreen.class);
intent.putExtra("int_value", int_variable);
startActivity(intent);
```

```
Intent intent = getIntent();
int temp = intent.getIntExtra("int_value", 0);
```

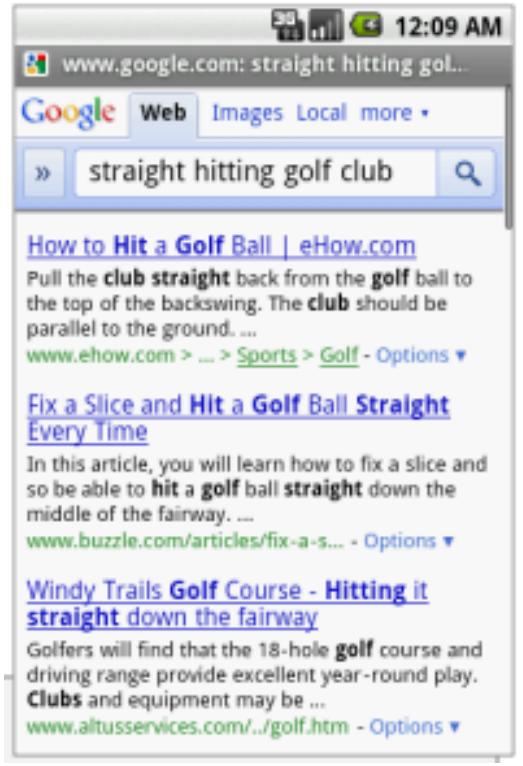
# Send values between Activities

**Data type of extra**

- boolean
- boolean[]
- byte
- byte[]
- char
- char[]
- CharSequence
- CharSequence[]
- double
- double[]
- float
- float[]
- int
- int[]
- long
- long[]
- short
- short[]
- String
- String[]
- ArrayList<CharSequence>
- ArrayList<String>
- ArrayList<Integer>
- Parcelable
- Serializable

**Exercise 1** Implement Intent sends values app

# Intents - Secondary Attributes




Not only defined variables, but also built-in variables

**Example 6 : Doing a Google search looking for golf clubs**

```
Intent intent = new Intent (Intent.ACTION_WEB_SEARCH );

intent.putExtra(SearchManager.QUERY,
                "straight hitting golf clubs");

startActivity(intent);
```

Secondary data

# Intents


Not only defined variables, but also built-in variables

**Example 7 :** Sending a text message (using extra attributes)

```
Intent intent = new Intent(Intent.ACTION_SENDTO,
Uri.parse("sms:5551234"));

intent.putExtra("sms_body", "are we playing golf next Saturday?”);


startActivity(intent);
```

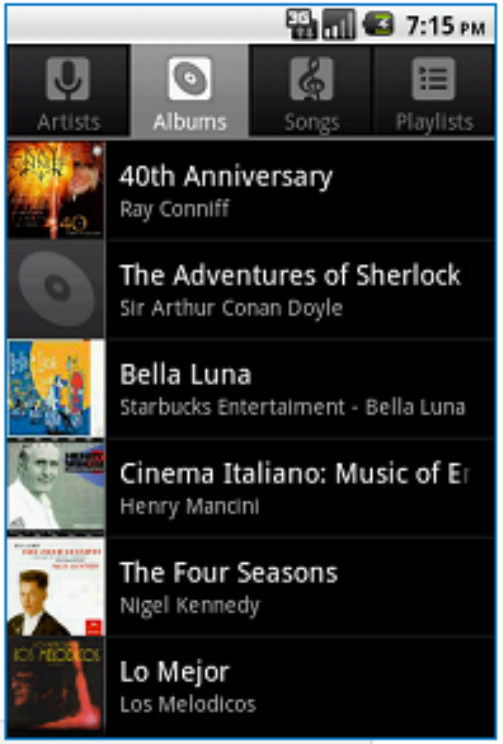
# Intents


Not only defined variables, but also built-in variables
- Set type

**Example 8 :** Showing Pictures (using extra attributes)

```
Intent myIntent = new Intent();

myIntent.setType("image/pictures/*");
myIntent.setAction(Intent.ACTION_GET_CONTENT);

startActivity(myIntent);
```

# Intents



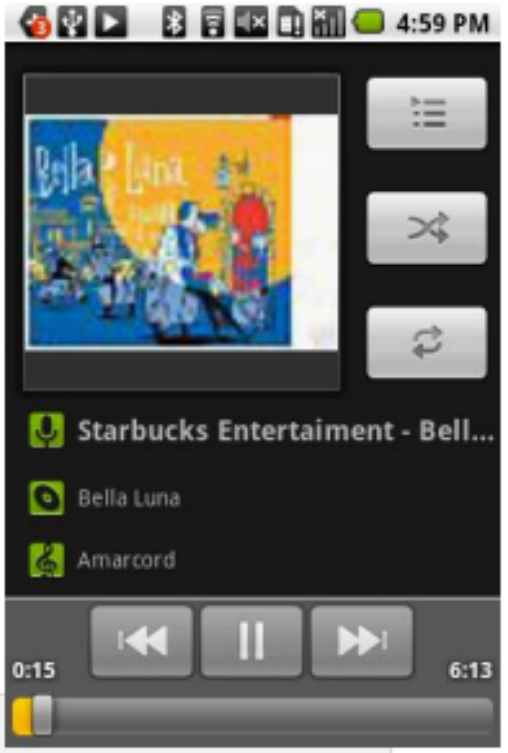

## More Examples: Using Standard Actions

## Launching the Music Player

*Reference:* http://developer.android.com/guide/appendix/g-app-intents.html

```
//launch music player

Intent myActivity2 =
       new Intent("android.intent.action.MUSIC_PLAYER");

startActivity(myActivity2);
```

# Intents




## More Examples: Using Standard Actions

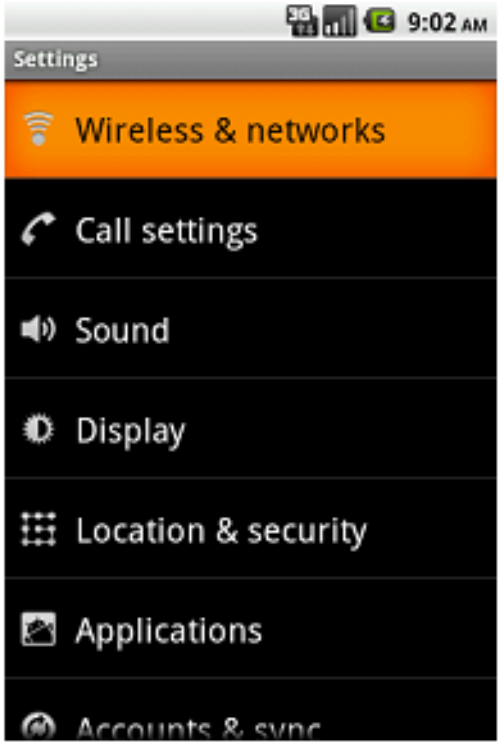
## Playing a song stored in the SD card

*Reference:* http://developer.android.com/guide/appendix/g-app-intents.html

```
// play song "amarcord.mp3" saved in the SD
Intent myActivity2 =
        new Intent(android.content.Intent.ACTION_VIEW);

Uri data = Uri.parse("file:///sdcard/amarcord.mp3");
String type = "audio/mp3";

myActivity2.setDataAndType(data, type);

startActivity(myActivity2);
```



# Intents




## More Examples: Using Standard Actions

## Setting System

*Reference:* http://developer.android.com/reference/android/provider/Settings.html

```
Intent intent = new Intent(
            android.provider.Settings.ACTION_SETTINGS);

startActivity(intent);
```

# Intents with getting results

## Starting Activities and Getting Results

The **startActivity(Intent)** method is used to start a new activity, which will be placed at the top of the activity stack.

Sometimes you want to get a result back from the called sub-activity when it ends.

For example, you may start an activity that let the user pick a person from a list of contacts; when it ends, it returns the person that was selected.

# Intents

## Starting Activities and Getting Results

In order to get results back from the called activity we use the method

**startActivityForResult ( Intent, requestCodeID )**

Where the second (*requestCodeID*) parameter identifies the call.

The result sent by the sub-activity could be picked up through the asynchronous method

**onActivityResult ( requestCodeID, resultCode, Intent )**

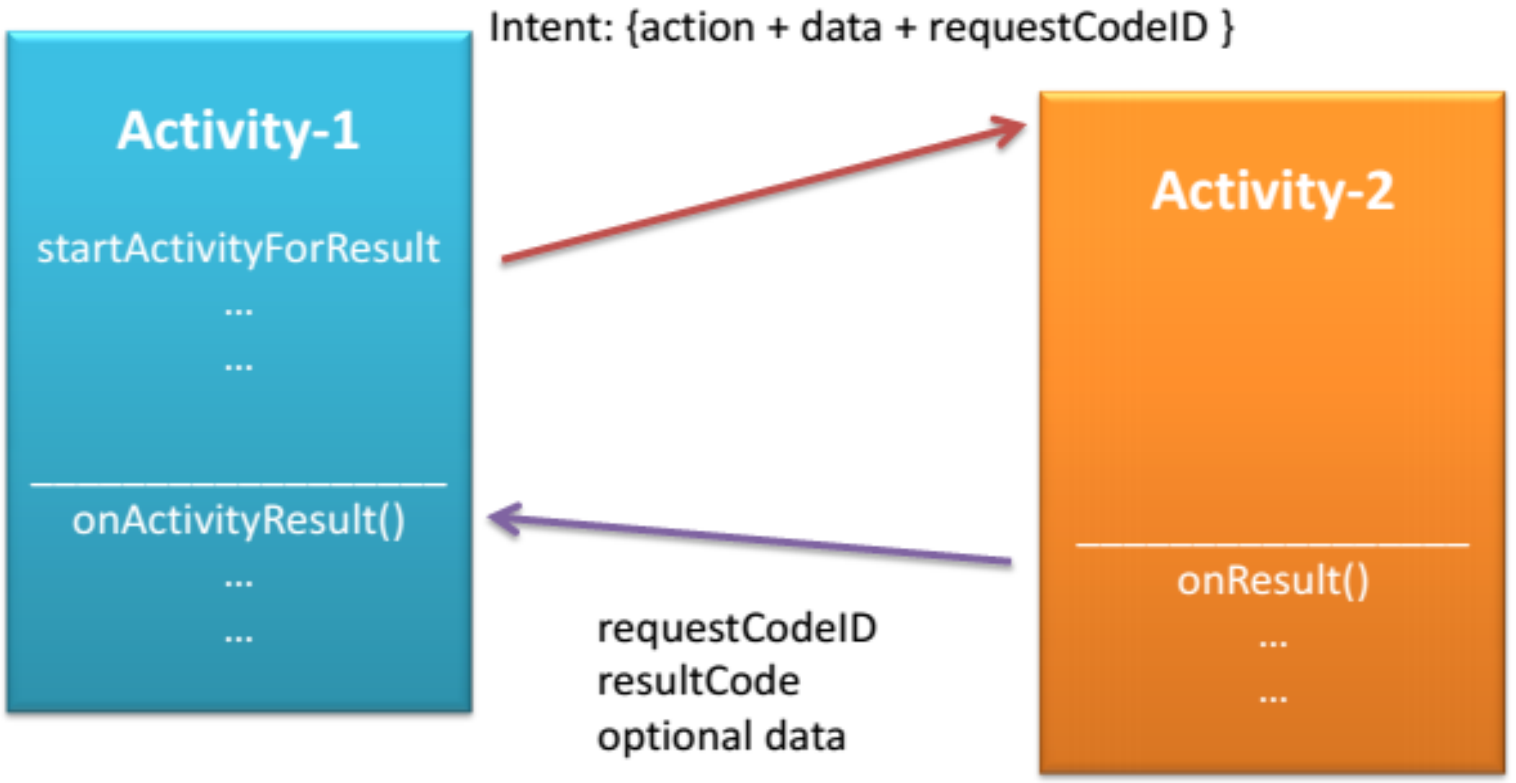
# Intents

## Starting Activities and Getting Results

- Before an activity exits, it can call **setResult (resultCode)** to return a termination signal back to its parent.
- Always supply a result code, which can be the standard results **Activity.RESULT_CANCELED, Activity.RESULT_OK**, or any custom values.
- All of this information can be capture back on the parent's `onActivityResult (int requestCodeID, int resultCode, Intent data)` along with the integer identifier it originally supplied.
- If a child activity fails for any reason (such as crashing), the parent activity will receive a result with the code **RESULT_CANCELED**.

# Intents

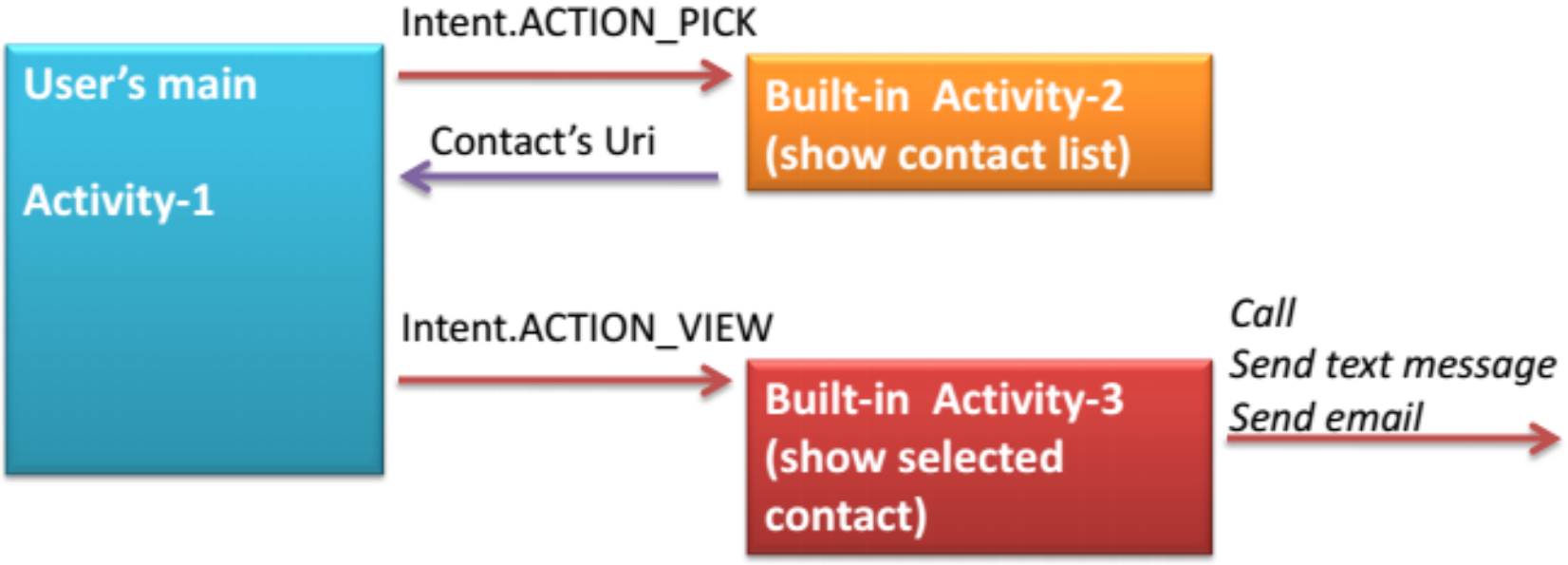
## Starting Activities and Getting Results



# Intents

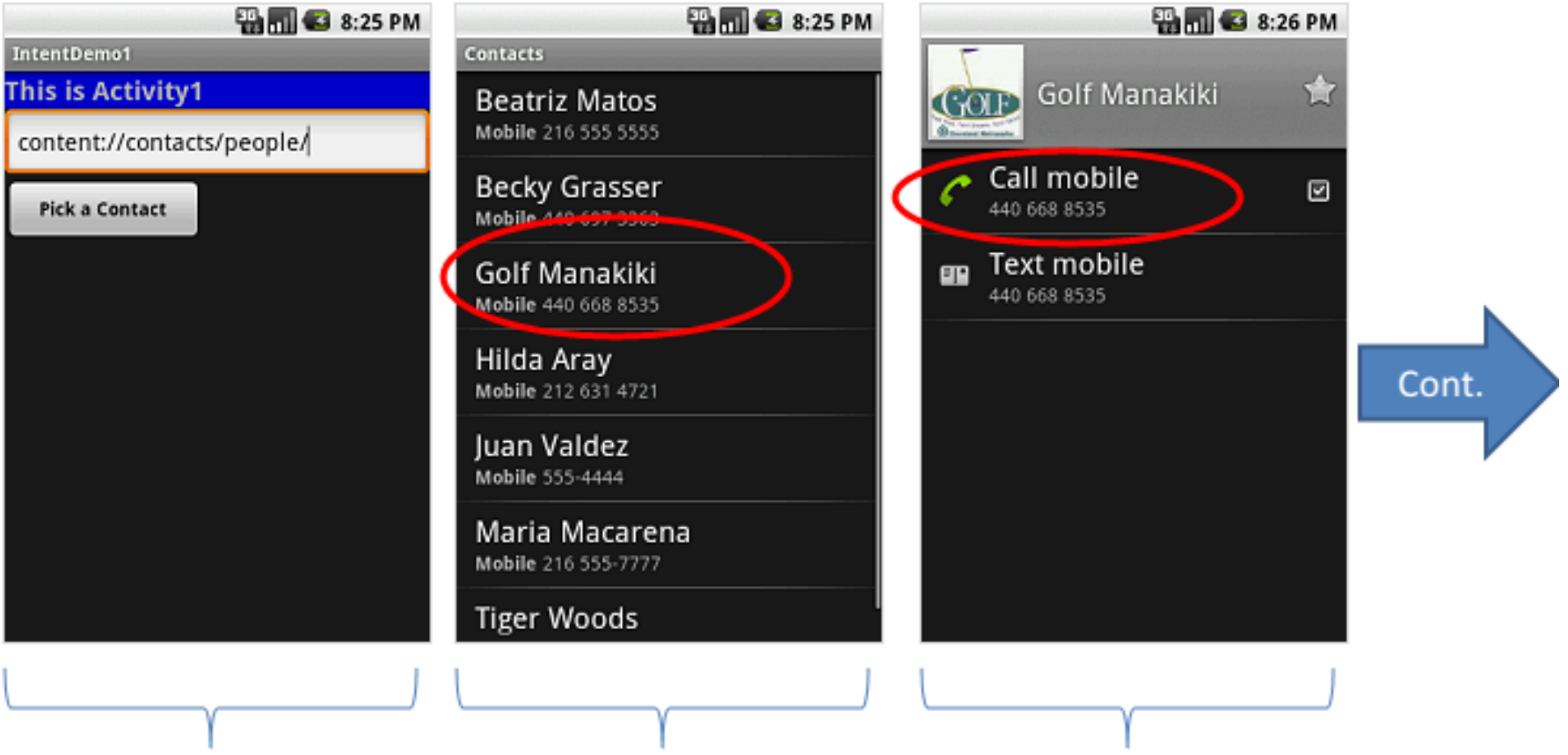
**Example2**. Let's play golf - Call for a tee-time.

1. Show all contacts and pick a particular one (*Intent.ACTION_PICK*).
2. For a successful interaction the main-activity accepts the returned URI identifying the person we want to call (*content://contacts/people/n*).
3. 'Nicely' show the selected contact's entry allowing calling, texting, emailing actions (Intent.ACTION_VIEW).



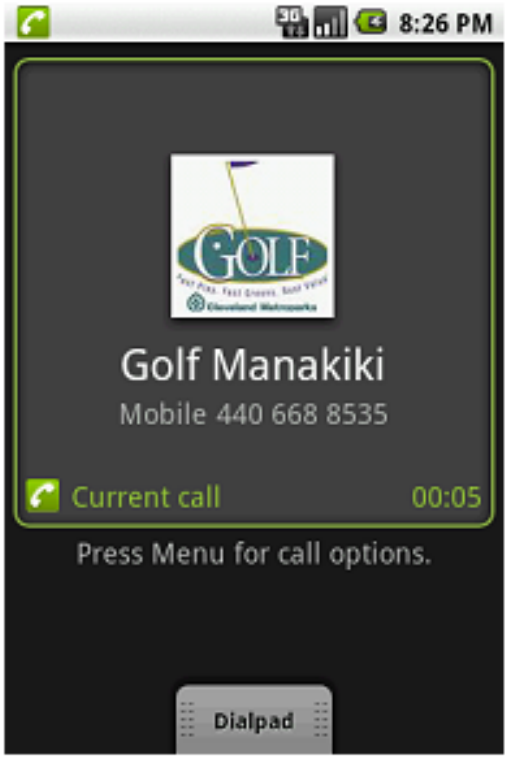
# Intents

**Example2**. Let's play golf - *Call for a tee-time*.

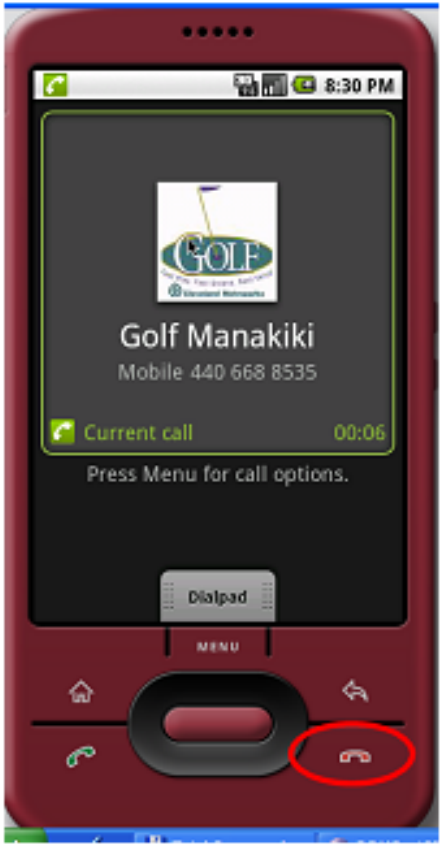


# Intents

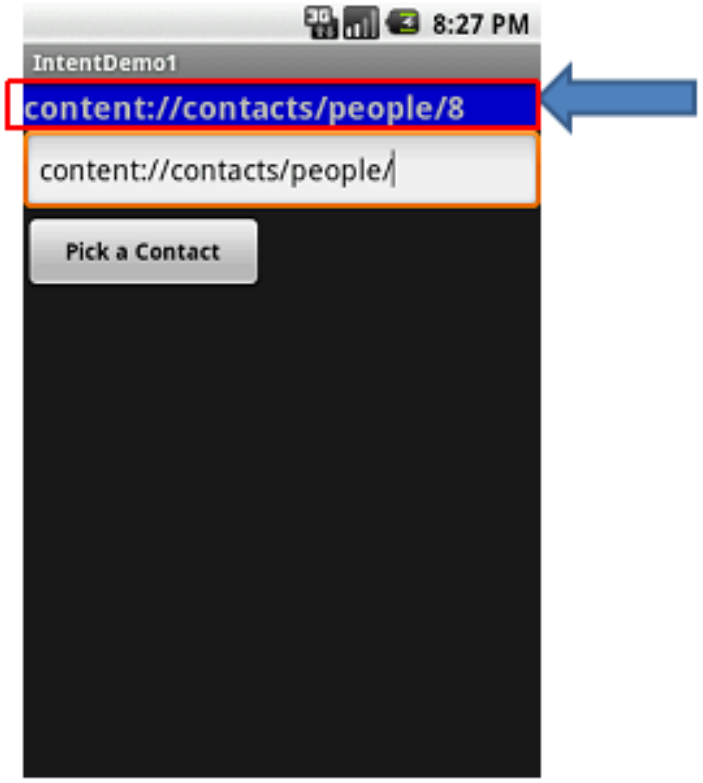
**Example2 (cont.)** Let's play golf - *Call for a tee-time*




Place the call




Terminate the call




Selected contact's URI



# Intents

**Example2.** *Calling a sub-activity, receiving results.*

```
//IntentDemo2_Intent: making a phone call
//receiving results from a sub-activity
package bim211.intents;
import android.app.Activity;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.*;

public class IntentDemo2 extends Activity {
    TextView label1;
    EditText text1;
    Button   btnCallActivity2;
```

# Intents

**Example2.** *Calling a sub-activity, receiving results.*

```
@Override
   public void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   try {
            setContentView(R.layout.main);
            label1 = (TextView)findViewById(R.id.label1);
            text1 = (EditText)findViewById(R.id.text1);

            btnCallActivity2 = (Button)findViewById(R.id.btnPickContact);
            btnCallActivity2.setOnClickListener(new ClickHandler());
   }
   catch (Exception e) {
            Toast.makeText(getBaseContext(),
            e.getMessage(), Toast.LENGTH_LONG).show();
   }
 }//onCreate
```

# Intents

**Example2.** *Calling a sub-activity, receiving results.*

```
private class ClickHandler implements OnClickListener {
    @Override
    public void onClick(View v) {
        try {
             // myData refer to: content://contacts/people/
             String myData = text1.getText().toString();

             //you may also try ACTION_VIEW instead
             Intent myActivity2 = new Intent(Intent.ACTION_PICK,
                                      Uri.parse(myData));
            // start myActivity2.
            // Tell it that our requestCodeID (or nickname) is 222
            startActivityForResult(myActivity2, 222);

            // Toast.makeText(getApplicationContext(),
            //                "I can't wait for you", 1).show();
        }
        catch (Exception e) {
            label1.setText(e.getMessage());
    }
    }//onClick
  }//ClickHandler
```

# Intents

**Example2.** *Calling a sub-activity, receiving results.*

```
@Override
protected void onActivityResult(int requestCode,
                                int resultCode,
                                Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    try {
        // use requestCode to find out who is talking back to us
        switch (requestCode){
            case (222): {
                // 222 is our friendly contact-picker activity
                if (resultCode == Activity.RESULT_OK) {
                    String selectedContact = data.getDataString();
                    // it will return an URI that looks like:
                    // content://contacts/people/n
                    // where n is the selected contacts' ID
                    label1.setText(selectedContact.toString());

                    //show a 'nice' screen with the selected contact
                    Intent myAct3 = new Intent (Intent.ACTION_VIEW,
                                                Uri.parse(selectedContact));
                    startActivity(myAct3);
                }
```

Listener



# Intents

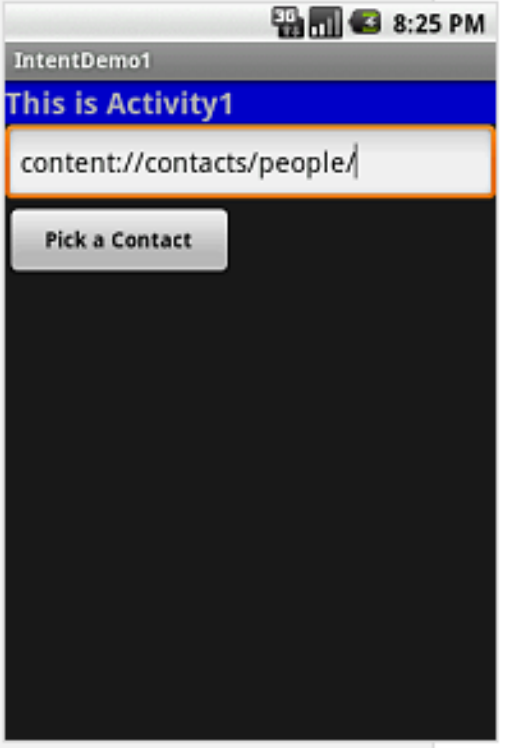
**Example2.** *Calling a sub-activity, receiving results.*

```
                    else {
                         //user pressed the BACK button
                         label1.setText("Selection CANCELLED "
                                        + requestCode + " " + resultCode);
                         }
                         break;
                    }
               }//switch
          }
          catch (Exception e) {
               Toast.makeText(getBaseContext(), e.getMessage(),
                              Toast.LENGTH_LONG).show();
          }
     }// onActivityResult

}//IntentDemo2
```

# Intents

**Example2.** *Calling a sub-activity, receiving results.*

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >
     <TextView
          android:id="@+id/label1"
          android:layout_width="fill_parent"
          android:layout_height="wrap_content"
          android:background="#ff0000cc"
          android:text="This is Activity1"
          android:textStyle="bold"
          android:textSize="20sp"/>
     <EditText
          android:id="@+id/text1"
          android:layout_width="fill_parent"
          android:layout_height="54px"
          android:text="content://contacts/people/"
          android:textSize="18sp"  />
     <Button
          android:id="@+id/btnPickContact"
          android:layout_width="149px"
          android:layout_height="wrap_content"
          android:text="Pick a Contact"
          android:textStyle="bold" />
</LinearLayout>
```



# Intents

**Example3.** Showing Pictures and Video - Calling a sub-activity, receiving results.

# Intents

**Example3.** Showing Pictures and Video - Calling a sub-activity, receiving results.

```
private void showSoundTracks() {

  Intent myIntent = new Intent();
  myIntent.setType("video/*, images/*");
  myIntent.setAction(Intent.ACTION_GET_CONTENT);
  startActivityForResult(myIntent, 0);

}//showSoundTracks


@Override
protected void onActivityResult(int requestCode, int resultCode, Intent intent) {
  super.onActivityResult(requestCode, resultCode, intent);

  if ((requestCode == 0) && (resultCode == Activity.RESULT_OK)) {

      String selectedImage = intent.getDataString();

      Toast.makeText(this, selectedImage, 1).show();

      // show a 'nice' screen with the selected image
      Intent myAct3 = new Intent(Intent.ACTION_VIEW, Uri.parse(selectedImage));
      startActivity(myAct3);
  }
}//onActivityResult
```

*All videos and all still images*