Android
Intents

Victor Matos
Cleveland State University

Notes are based on:
Android Developers
Intents

Android Activities

An Android application could include any number of activities.

- An *activity* uses the `setContentView(...)` method to expose (usually) 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*.

- Activities interact with each other in an *asynchronous* mode.
Intents

Android Activities

Android Application

Main Activity

Sub-Activity-1

Sub-Activity-n

results

intents

extras

12. Android – Intents
## Intents

Intents are invoked using the following options

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>startActivity(intent)</code></td>
<td>launches an Activity</td>
</tr>
<tr>
<td><code>sendBroadcast(intent)</code></td>
<td>sends an intent to any interested BroadcastReceiver components</td>
</tr>
<tr>
<td><code>startService(intent)</code> or <code>bindService(intent, ...)</code></td>
<td>communicate with a background Service.</td>
</tr>
</tbody>
</table>
Intents

The main arguments of an Intent are:

1. **Action**  
The built-in action to be performed, such as `ACTION_VIEW`, `ACTION_EDIT`, `ACTION_MAIN`, ... or *user-created-activity*

2. **Data**  
The primary data to operate on, such as a phone number to be called (expressed as a **Uri**).
Intents

Typically an intent is called as follows:

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

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

Built-in or user-created activity
Examples of *action/data* pairs are:

**ACTION_DIAL**  \textit{tel:123}
Display the phone dialer with the given number filled in.

**ACTION_VIEW**  \textit{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**  \textit{content://contacts/people/2}
Edit information about the person whose identifier is "2".

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

**ACTION_VIEW**  \textit{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.
## Built-in Standard Actions

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

<table>
<thead>
<tr>
<th>Left Column</th>
<th>Right Column</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTION_MAIN</td>
<td>ACTION_ANSWER</td>
</tr>
<tr>
<td>ACTION_VIEW</td>
<td>ACTION_INSERT</td>
</tr>
<tr>
<td>ACTION_ATTACH_DATA</td>
<td>ACTION_DELETE</td>
</tr>
<tr>
<td>ACTION_EDIT</td>
<td>ACTION_RUN</td>
</tr>
<tr>
<td>ACTION_PICK</td>
<td>ACTION_SYNC</td>
</tr>
<tr>
<td>ACTION_CHOOSER</td>
<td>ACTION_PICK_ACTIVITY</td>
</tr>
<tr>
<td>ACTION_GET_CONTENT</td>
<td>ACTION_SEARCH</td>
</tr>
<tr>
<td>ACTION_DIAL</td>
<td>ACTION_WEB_SEARCH</td>
</tr>
<tr>
<td>ACTION_CALL</td>
<td>ACTION_FACTORY_TEST</td>
</tr>
<tr>
<td>ACTION_SEND</td>
<td></td>
</tr>
<tr>
<td>ACTION_SENDTO</td>
<td></td>
</tr>
</tbody>
</table>

**Intents**

---

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

<table>
<thead>
<tr>
<th>Action Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTION_MAIN</td>
<td>Main action</td>
</tr>
<tr>
<td>ACTION_VIEW</td>
<td>View action</td>
</tr>
<tr>
<td>ACTION_ATTACH_DATA</td>
<td>Attach data action</td>
</tr>
<tr>
<td>ACTION_EDIT</td>
<td>Edit action</td>
</tr>
<tr>
<td>ACTION_PICK</td>
<td>Pick action</td>
</tr>
<tr>
<td>ACTION_CHOOSER</td>
<td>Chooser action</td>
</tr>
<tr>
<td>ACTION_GET_CONTENT</td>
<td>Get content action</td>
</tr>
<tr>
<td>ACTION_DIAL</td>
<td>Dial action</td>
</tr>
<tr>
<td>ACTION_CALL</td>
<td>Call action</td>
</tr>
<tr>
<td>ACTION_SEND</td>
<td>Send action</td>
</tr>
<tr>
<td>ACTION_SENDTO</td>
<td>Sendto action</td>
</tr>
<tr>
<td>ACTION_ANSWER</td>
<td>Answer action</td>
</tr>
<tr>
<td>ACTION_INSERT</td>
<td>Insert action</td>
</tr>
<tr>
<td>ACTION_DELETE</td>
<td>Delete action</td>
</tr>
<tr>
<td>ACTION_RUN</td>
<td>Run action</td>
</tr>
<tr>
<td>ACTION_SYNC</td>
<td>Sync action</td>
</tr>
<tr>
<td>ACTION_PICK_ACTIVITY</td>
<td>Pick activity</td>
</tr>
<tr>
<td>ACTION_SEARCH</td>
<td>Search action</td>
</tr>
<tr>
<td>ACTION_WEB_SEARCH</td>
<td>Web search action</td>
</tr>
<tr>
<td>ACTION_FACTORY_TEST</td>
<td>Factory test action</td>
</tr>
</tbody>
</table>
Example

Display the phone dialer with the given number filled in.

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

1. Category
2. Components
3. Type
4. Extras

Example: Doing a Google search looking for golf clubs

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

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

startActivity(intent);
```

Apparently the Google answer is ‘none’
Intents - Secondary Attributes

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

```java
Intent intent = new Intent(Intent.ACTION_SENDTO, Uri.parse("sms://"));
intent.putExtra("address", "555-1234");
intent.putExtra("sms_body", "remember to buy bread and milk");
startActivity(intent);
```

“address”, “sms_body”  are keywords
Intents

Intents - Secondary Attributes

Example: Showing Pictures (using extra attributes)

```java
Intent myIntent = new Intent();
myIntent.setType("image/pictures/*");
myIntent.setAction(Intent.ACTION_GET_CONTENT);
startActivity(myIntent);
```
1. A Complete Example: Activity1 displays an interface to accept a phone number and requests (built-in) Activity2 to make the call.

```xml
<?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="tel:555-1234"
        android:textSize="18sp" />
    <Button
        android:id="@+id/btnCallActivity2"
        android:layout_width="149px"
        android:layout_height="wrap_content"
        android:text="Make Phone Call"
        android:textStyle="bold" />
</LinearLayout>
```
12. Android – Intents

**Intents**

**1. A Complete Example:** Activity1 displays an interface to accept a phone number and requests (built-in) Activity2 to make the call.
1. A Complete Example: Activity1 displays an interface to accept a phone number and requests (built-in) Activity2 to make the call.

```java
//IntentDemo1_Intent: making a phone call
package cis493.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 IntentDemo1 extends Activity {
    TextView label1;
    EditText text1;
    Button btnCallActivity2;
```
1. A Complete Example: Activity1 displays an interface to accept a phone number and requests (built-in) Activity2 to make the call.

```java
@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.btnCallActivity2);
        btnCallActivity2.setOnClickListener(new ClickHandler());
    }
    catch (Exception e) {
        Toast.makeText(getBaseContext(), e.getMessage(), Toast.LENGTH_LONG).show();
    }
} //onCreate
```
1. A Complete Example: Activity1 displays an interface to accept a phone number and requests (built-in) Activity2 to make the call.

```java
private class ClickHandler implements OnClickListener {
    @Override
    public void onClick(View v) {
        try {
            // myActivity2 places a phone call
            // for ACTION_CALL or ACTION_DIAL
            // use 'tel:' formatted data: "tel:555-1234"
            // for ACTION_VIEW use data: "http://www.youtube.com"
            // (you also need INTERNET permission - see Manifest)

            String myData = text1.getText().toString();
            Intent myActivity2 = new Intent(Intent.ACTION_DIAL, Uri.parse(myData));
            startActivity(myActivity2);
        } catch (Exception e) {
            Toast.makeText(getBaseContext(), e.getMessage(), Toast.LENGTH_LONG).show();
        }
    }
}
```

//IntentDemo1
1. A Complete Example: Activity1 displays an interface to accept a phone number and requests (built-in) Activity2 to make the call.

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="cis493.intents"
    android:versionCode="1"
    android:versionName="1.0">
    <application android:icon="@drawable/icon"
        android:label="@string/app_name">
        <activity android:name=".IntentDemo1"
            android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
    <uses-sdk android:minSdkVersion="3" />
</manifest>
```
## Intents

### Built-in Standard Broadcast Actions

List of standard actions that Intents can use for receiving broadcasts (usually through `registerReceiver(BroadcastReceiver, IntentFilter)` or a `<receiver>` tag in a manifest).

<table>
<thead>
<tr>
<th>Action Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTION_TIME_TICK</td>
</tr>
<tr>
<td>ACTION_TIME_CHANGED</td>
</tr>
<tr>
<td>ACTION_TIMEZONE_CHANGED</td>
</tr>
<tr>
<td>ACTION_BOOT_COMPLETED</td>
</tr>
<tr>
<td>ACTION_PACKAGE_ADDED</td>
</tr>
<tr>
<td>ACTION_PACKAGE_CHANGED</td>
</tr>
<tr>
<td>ACTION_PACKAGE_REMOVED</td>
</tr>
<tr>
<td>ACTION_UID_REMOVED</td>
</tr>
<tr>
<td>ACTION_BATTERY_CHANGED</td>
</tr>
</tbody>
</table>
More Examples: Using Standard Actions

Call Immediately
Modify the complete example 1 replacing the method ‘ClickHandler’ with the following code

```java
String myData = "tel:555-1234";
Intent myActivity2 = new Intent(Intent.ACTION_CALL,
                                 Uri.parse(myData));
startActivity(myActivity2);
```

Needs Permission:
```xml
<uses-permission android:name="android.permission.CALL_PHONE" />
```
More Examples: Using Standard Actions

Show all your Contacts
Modify the *complete* example1 replacing the method ‘ClickHandler’ with the following code

```java
String myData = "content://contacts/people/";

Intent myActivity2 = new Intent(Intent.ACTION_VIEW,
                                Uri.parse(myData));

startActivity(myActivity2);
```
Intents

More Examples: Using Standard Actions

Show a Particular Contact (ID = 2)
Modify the complete example1 replacing the method ‘ClickHandler’ with the following code

```java
String myData = "content://contacts/people/2";
Intent myActivity2 = new Intent(Intent.ACTION_VIEW,
    Uri.parse(myData));
startActivity(myActivity2);
```
Intents

More Examples: Using Standard Actions

Edit a Particular Contact (ID = 2)
Modify the complete example1 replacing the method ‘ClickHandler’ with the following code

```java
String myData = "content://contacts/people/2";
Intent myActivity2 = new Intent(Intent.ACTION_EDIT,
                              Uri.parse(myData));
startActivity(myActivity2);
```
More Examples: Using Standard Actions

View a Webpage
Modify the complete example1 replacing the method ‘ClickHandler’ with the following code

```java
String myData = "http://www.youtube.com";

Intent myActivity2 = new Intent(Intent.ACTION_VIEW, Uri.parse(myData));

startActivity(myActivity2);
```

Caution. Add to the Manifest a request to use the Internet:

```
<uses-permission android:name="android.permission.INTERNET"/>
```
More Examples: 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=1860+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.permissionINTERNET" />
More Examples: Using Standard Actions

Geo Mapping Coordinates (latitude, longitude)
Provide a geoCode holding latitude and longitude (also an additional zoom ‘?z=xx’ with xx in range 1..23)

```java
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:

```xml
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.INTERNET" />
```
More Examples: Using Standard Actions

Geo Mapping - Google StreetView

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

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.permissionINTERNET" />
More Examples: Using Standard Actions

Launching the Music Player


```java
//launch music player

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

startActivity(myActivity2);
```
More Examples: Using Standard Actions

Playing a song stored in the SD card


```
// 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);
```
More Examples: Using Standard Actions

Sending MMS
Add picture #1 from SD to MMS


```java
//send mms attach picture #1 to it

Uri uri = Uri.parse("content://media/external/images/media/1");

myActivity2 = new Intent(Intent.ACTION_SEND);

myActivity2.putExtra("address", "555-1234");
myActivity2.putExtra("sms_body", "some text message goes here");
myActivity2.putExtra(Intent.EXTRA_STREAM, uri);
myActivity2.setType("image/png");

startActivity(myActivity2);
```
More Examples: Using Standard Actions

Sending Email


```java
// send email
Uri uri = Uri.parse("mailto:v.matos@csuohio.edu");
Intent myActivity2 = new Intent(Intent.ACTION_SENDTO, uri);

// you may skip the next two pieces [subject/text]
myActivity2.putExtra(Intent.EXTRA_SUBJECT, "subject goes here");
myActivity2.putExtra(Intent.EXTRA_TEXT, "The email's body goes here");

startActivity(myActivity2);
```
12. Android – Intents

**Intents**

**More Examples: Using Standard Actions**

**Setting System**


```java
Intent intent = new Intent(
    android.provider.Settings.ACTION_SETTINGS);
startActivity(intent);
```
More Examples: Using Standard Actions

Setting System Locale: Language & Keyboard


```java
Intent intent = new Intent(
    android.provider.Settings.ACTION_LOCALE_SETTINGS);
startActivity(intent);
```
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.
Starting Activities and Getting Results

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

```java
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

```java
onActivityResult ( requestCodeID, resultCode, Intent )
```
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

Activity-1

startActivityForResult
...
...

onActivityResult()
...
...

Intent: {action + data + requestCodeID }

Activity-2

onResult()
...
...

requestCodeID
resultCode
optional data
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).
Example 2. Let’s play golf - *Call for a tee-time.*
Example2 (cont.) Let’s play golf - *Call for a tee-time*

- Place the call
- Terminate the call
- Selected contact’s URI
Example 2. *Calling a sub-activity, receiving results.*

```java
//IntentDemo2_Intent: making a phone call
//receiving results from a sub-activity
package cis493.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.view.View.OnClickListener;
import android.widget.*;
public class IntentDemo2 extends Activity {
    TextView label1;
    EditText text1;
    Button btnCallActivity2;
```
Example 2. *Calling a sub-activity, receiving results.*

```java
@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
```
Example 2. **Calling a sub-activity, receiving results.**

```java
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
```
Example 2. *Calling a sub-activity, receiving results.*

```java
@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);
                }
            }
        }
    }
}
```
Example 2. *Calling a sub-activity, receiving results.*

```java
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
```
Example 2. *Calling a sub-activity, receiving results.*

```xml
<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

Example 3. Showing Pictures and Video - Calling a sub-activity, receiving results.

```java
private void showSoundTracks() {
    Intent myIntent = new Intent();
    myIntent.setType("video/*, images/*");
    myIntent.setAction(Intent.ACTION_GET_CONTENT);
    startActivityForResult(myIntent, 0);
}

@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
Example3. Showing Pictures and Video - Calling a sub-activity, receiving results.
Example 4. Showing/Playing Sound Tracks - Calling a sub-activity, receiving results.

```java
private void showSoundTracks() {
    Intent myIntent = new Intent();
    myIntent.setType("audio/mp3");
    myIntent.setAction(Intent.ACTION_GET_CONTENT);
    startActivityForResult(myIntent, 0);
} //showSoundTracks
```

The returned string value is similar to the following “content://media/external/audio/media/14” ACTION_VIEW on that Uri would produce a result similar to the image on the right.
Intents

Questions ?
## Intents

### Built-in Standard Broadcast Actions

List of standard actions that Intents can use for receiving broadcasts (usually through `registerReceiver(BroadcastReceiver, IntentFilter)` or a `<receiver>` tag in a manifest).

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTION_TIME_TICK</td>
<td></td>
</tr>
<tr>
<td>ACTION_TIME_CHANGED</td>
<td></td>
</tr>
<tr>
<td>ACTION_TIMEZONE_CHANGED</td>
<td></td>
</tr>
<tr>
<td>ACTION_BOOT_COMPLETED</td>
<td></td>
</tr>
<tr>
<td>ACTION_PACKAGE_ADDED</td>
<td></td>
</tr>
<tr>
<td>ACTION_PACKAGE_CHANGED</td>
<td></td>
</tr>
<tr>
<td>ACTION_PACKAGE_REMOVED</td>
<td></td>
</tr>
<tr>
<td>ACTION_UID_REMOVED</td>
<td></td>
</tr>
<tr>
<td>ACTION_BATTERY_CHANGED</td>
<td></td>
</tr>
</tbody>
</table>
Appendix: Getting Permissions

Becomes:

```xml
<uses-permission android:name="android.permission.CALL_PHONE"/>
```