Accelerating Information Technology Innovation

http://aiti.mit.edu

Cali, Colombia
Summer 2013
Lesson 6 – Intents
Agenda

• Introduction to Intents
• Explicit Intents: switching activities
• Implicit Intents: switching apps
• Get Results from Intents
• Make your app callable
Introduction Intent

• An object that provides runtime binding between separate components (such as two activities).
• The **Intent** represents an app’s "intent to do something."
• You can use an **Intent** for a wide variety of tasks, but most often they’re used to start another activity.
• Intents can be **implicit** or **explicit**
Explicit Intent

Switch Activities
Multiple Activities

- An android application consists of multiple Activity objects
- Each Activity is like one “page” of the app
- Only one activity can be the main activity

```xml
<application android:label="Snake on a Phone">
  <activity android:name="Snake"
    android:theme="@android:style/Theme.NoTitleBar"
    android:screenOrientation="portrait"
    android:configChanges="keyboardHidden|orientation">
    <intent-filter>
      <action android:name="android.intent.action.MAIN"/>
      <category android:name="android.intent.category.LAUNCHER"/>
    </intent-filter>
  </activity>
  define other activities here...
</application>
```
Multiple Activities, example:

- **Main Activity (first thing you see when App starts):**
  - Welcome to the Tic-Tac-Toe Sample!
  - This activity is defined in one project. The second activity, launched by one of the buttons below, is located in another project which is a "library" to the main one and merged in the same APK.
  - This sample code demonstrates how to split an application in multiple projects by using the 'project library' available in the Froyo SDK Tools.

- **Second Activity (clicking on a button on the Main Activity brings user to this one):**
  - Player 1's turn -- that's you!
  - I'm done

- **Main Activity Stopped!!**
Switching between Activities

Step 1: Define all Activities in your App in the AndroidManifest.xml file

```xml
<application android:name=".MyApplication" android:icon="@drawable/icon" android:label="@string/app_name">
  <activity android:name=".OneActivity"
            android:label="@string/app_name">
    <intent-filter>
      <action android:name="android.intent.action.MAIN"/>
      <category android:name="android.intent.category.LAUNCHER"/>
    </intent-filter>
  </activity>
  <activity android:name=".AnotherActivity" android:label="picture capture"/>
</application>
```

Step 2: Switch from Main Activity to the activity defined in `AnotherActivity.class`, using Intent objects.

```java
Intent intent = new Intent(this, AnotherActivity.class);
startActivity(intent);
```
Passing data between Activities

in your current activity, create an intent

```java
Intent i = new Intent(getApplicationContext(), ActivityB.class);
i.putExtra(key, value);
startActivity(i);
```

then in the other activity, retrieve those values.

```java
Bundle extras = getIntent().getExtras();
if(extras !=null) {
    String value = extras.getString(key);
}
```

Note: you can use the putExtra method to add data in key value pairs to the Intent. The **key must be a String** object but the value can be any of the following: integer, integer[], float, float[], double, double[], String, String[], etc… (Strings, primitive types, arrays of the last ones).

Then, you fetch that data in the second activity using the `.getExtras().getString(key)` approach.
## Exchange data between activities

<table>
<thead>
<tr>
<th></th>
<th>Persistent</th>
<th>Non-persistent</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Primitive</strong></td>
<td>SharedPreferences</td>
<td>Intent</td>
</tr>
<tr>
<td><strong>Objects</strong></td>
<td>Database (SQL), Files, ...</td>
<td>Application Class, Public Static Instance Variables</td>
</tr>
</tbody>
</table>
Implicit Intent

Switch Apps
Implicit Intent

- Send a request to open an activity based on an "action" it would like to perform.
- Specify the **action to perform**, not the activity to invoke.
- Uri – Uniform Resource Identifier
- Example: View a web page

```java
Uri webpage = Uri.parse("http://www.android.com");
Intent webIntent = new Intent(Intent.ACTION_VIEW, webpage);
```
Examples Implicit Intent

- View a Map

```java
// Map point based on address
Uri location = Uri.parse("geo:0,0?q=1600+Amphitheatre+Parkway,+Mountain+View,+California");  
// Or map point based on latitude/longitude
Uri location = Uri.parse("geo:37.422219,-122.08364?z=14");  
Intent mapIntent = new Intent(Intent.ACTION_VIEW, location);
```

- Initiate a Phone Call

```java
Uri number = Uri.parse("tel:5551234");
Intent callIntent = new Intent(Intent.ACTION_DIAL, number);
```
Examples Implicit Intent

• Share content

```java
Intent intent = new Intent(Intent.ACTION_SEND);
intent.setType("text/plain");
intent.putExtra(android.content.Intent.EXTRA_TEXT, "News for you!");
startActivity(intent);
```

• Other examples:
  – Send sms?
Verify an App is available to Receive the Intent

- Your app will crash if there is no app to receive your (implicit) intent.
- Example how to check:

```java
PackageManager packageManager = getPackageManager();
List<ResolveInfo> activities = packageManager.queryIntentActivities(intent, 0);
boolean isIntentSafe = activities.size() > 0;
```
Start the Activity with an Intent

• Example:

```java
// Build the intent
Uri location = Uri.parse("geo:0,0?q=1600+Amphitheatre+Park");
Intent mapIntent = new Intent(Intent.ACTION_VIEW, location);

// Verify it resolves
PackageManager packageManager = getPackageManager();
List<ResolveInfo> activities = packageManager.queryIntentActivities(mapIntent, 0);
boolean isIntentSafe = activities.size() > 0;

// Start an activity if it's safe
if (isIntentSafe) {
    startActivity(mapIntent);
}
```
Get results from intent
Get a Result from an Intent

• Call an Activity and get a result back
• Examples: Call Camera app and get picture taken, call Contacts app and get a certain contact
• Use `startActivityForResult()` instead of `startActivity()`
Get a Result from an Intent (cont)

- Example of `startActivityForResult()` usage

```java
static final int PICK_CONTACT_REQUEST = 1; // The request code

private void pickContact() {
    Intent pickContactIntent = new Intent(Intent.ACTION_PICK, Uri.parse("content://contacts"));
    pickContactIntent.setType(Phone.CONTENT_TYPE); // Show user only contacts w/ phone numbers
    startActivityForResult(pickContactIntent, PICK_CONTACT_REQUEST);
}
```
Receive the Result

- **Use of `onActivityResult()`**
- `requestCode =` same as from start activity
- `resultCode = RESULT_OK, RESULT_CANCELED`

```java
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    // Check which request we're responding to
    if (requestCode == PICK_CONTACT_REQUEST) {
        // Make sure the request was successful
        if (resultCode == RESULT_OK) {
            // The user picked a contact.
            // The Intent's data Uri identifies which contact was selected.

            // Do something with the contact here (bigger example below)
        }
    }
}
```
Make your app callable
Allow other Apps start your Activity

- Add an intent filter in AndroidManifest.xml

```xml
<activity android:name="ShareActivity">
  <!-- filter for sending text; accepts SENDTO action with sms URI schemes -->
  <intent-filter>
    <action android:name="android.intent.action.SENDTO"/>
    <category android:name="android.intent.category.DEFAULT"/>
    <data android:scheme="sms"/>
    <data android:scheme="smsto"/>
  </intent-filter>
  <!-- filter for sending text or images; accepts SEND action and text or image data -->
  <intent-filter>
    <action android:name="android.intent.action.SEND"/>
    <category android:name="android.intent.category.DEFAULT"/>
    <data android:mimeType="image/*"/>
    <data android:mimeType="text/plain"/>
  </intent-filter>
</activity>
```
Handle the Intent in your Activity

```java
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    setContentView(R.layout.main);

    // Get the intent that started this activity
    Intent intent = getIntent();
    Uri data = intent.getData();

    // Figure out what to do based on the intent type
    if (intent.getType().indexOf("image/") != -1) {
        // Handle intents with image data ...
    } else if (intent.getType().equals("text/plain")) {
        // Handle intents with text ...
    }
}
```
Return a Result

```java
@Override
public void finish() {
    // Prepare data intent
    Intent data = new Intent();
    data.putExtra("returnKey1", "Swinging on a star. ");
    data.putExtra("returnKey2", "You could be better then you are. ");
    // Activity finished ok, return the data
    setResult(RESULT_OK, data);
    super.finish();
}
```
Resources

• Intents

• Intents Filters in your Activity

• Vogella tutorial (next slide)
Lab

1. Vogella Lab http://www.vogella.com/articles/AndroidIntent/
   1. Pass data between activities
   2. Implicit intents showcase TRY IT!
   3. Implement a browser
   4. Take a picture!! TRY IT!

2. Project