Accelerating Information Technology Innovation

http://aiti.mit.edu

Cali, Colombia
Summer 2012
Lesson 6 – Intents
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

- Specifies the exact recipient activity
- Add additional information to the intent
- **Use** `startActivity()` **to send the intent**

```java
/** Called when the user selects the Send button */
public void sendMessage(View view) {
    Intent intent = new Intent(this, DisplayMessageActivity.class);
    EditText editText = (EditText) findViewById(R.id.edit_message);
    String message = editText.getText().toString();
    intent.putExtra(EXTRA_MESSAGE, message);
    startActivity(intent);
}
```
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

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

• Example: 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"); // z param is zoom level
Intent mapIntent = new Intent(Intent.ACTION_VIEW, location);
```

• Example: Initiate a Phone Call

```java
Uri number = Uri.parse("tel:5551234");
Intent callIntent = new Intent(Intent.ACTION_DIAL, number);
```
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 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, new Uri("content://contacts/contacts"));
    pickContactIntent.setType(Phone.CONTENT_TYPE); // Show user only contacts w/ phone number
    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) {
    if (requestCode == PICK_CONTACT_REQUEST) {
        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)
        }
    }
}
```
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>
```
@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

If you want to return a result to the activity that invoked yours, simply call `setResult()` to specify the result code and result `Intent`. When your operation is done and the user should return to the original activity, call `finish()` to close (and destroy) your activity. For example:

```java
// Create intent to deliver some kind of result data
Intent result = new Intent("com.example.RESULT_ACTION", Uri.parse("content://result_uri
setResult(Activity.RESULT_OK, result);
finish();
```
Resources

• Intents

• Intents Filters in your Activity