MIT
Global Startup Labs
México 2013

http://gsl.mit.edu
Coming Soon!

Lección 03 – Event Handlers and Intents

MISTI

Google

Agenda

- Multiple Activites
- Switching Activities
- Passing data between Activities
- Intents

Multiple Activities

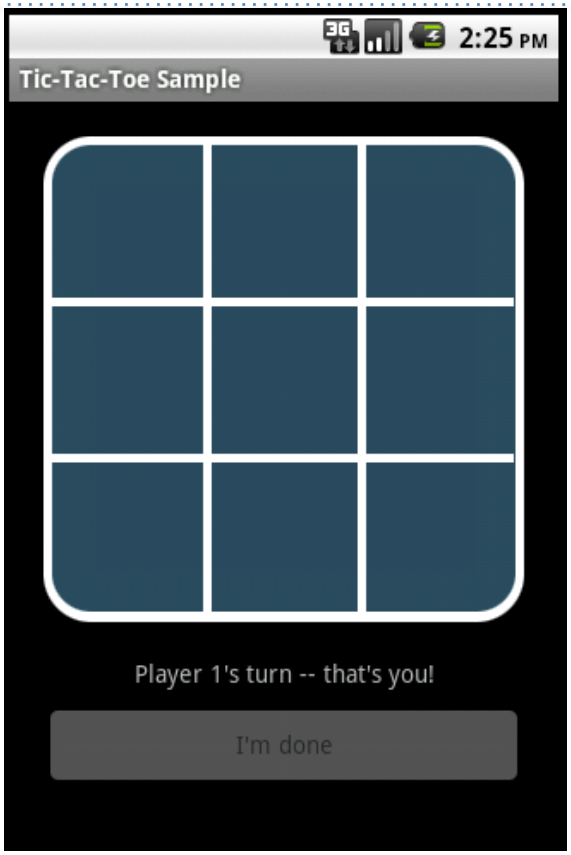
- An android application consists of multiple Activity objects
- Each Activity is like one “page” of the app
- Only one activity can be the *main* activity

```
<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)

Second Activity (clicking on a button on the Main Activity brings user to this one)

Switching between Activities

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

Main Activity

Second Activity

```
<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">
    </activity>

</application>
```

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

```
Intent intent = new Intent(this, AnotherActivity.class);
startActivity(intent);
```

Passing data between Activities

in your current activity, create an intent

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

then in the other activity, retrieve those values.

```
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…

Then, you fetch that data in the second activity using the .getExtras().getString(key) approach.

Other ways to exchange data between Activities

- Intent approach is best for *primitive* data types that don't need to last forever (i.e. they are *not persistent*)
- For *primitive* types that need to last forever (i.e. *persistent* objects), use Preferences
- For *non-primitive* types that are *not persistent*:
  - Public Static Fields
  - Maintain global application state in the Application class (all Activity objects have access to this).
- For non-primitive types that are *persistent*:
  - Use ContentProvider, SQL Database on the phone, Files, etc.

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

```
/** 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

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

Implicit Intent (cont)

- Example: View a Map

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

- Example: Initiate a Phone Call

```
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:

```
PackageManager packageManager = getPackageManager();
List<ResolveInfo> activities = packageManager.queryIntentActivities(intent, 0)
boolean isIntentSafe = activities.size() > 0;
```

Start the Activity with an Intent

- Example:

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

// Verify it resolves
PackageManager packageManager = getPackageManager();
List<ResolveInfo> activities = packageManager.queryIntent
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

```
static final int PICK_CONTACT_REQUEST = 1;  // The request code
...
private void pickContact() {
    Intent pickContactIntent = new Intent(Intent.ACTION_PICK, new Uri("content://co
    pickContactIntent.setType(Phone.CONTENT_TYPE); // Show user only contacts w/ ph
    startActivityForResult(pickContactIntent, PICK_CONTACT_REQUEST);
}
```

Receive the Result

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

```
@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)
        }
    }
}
```

Allow other Apps start your Activity

- Add an intent filter in AndroidManifest.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

```
@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:

```
// 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
  - http://developer.android.com/training/basics/intents/index.html
- Intents Filters in your Activity
  - http://developer.android.com/training/basics/intents/filters.html