

CS 646 Android Mobile Application Development
Spring Semester, 2015
Doc 5 Intents, Keyboard
Feb 3, 2015

Copyright ©, All rights reserved. 2015 SDSU & Roger Whitney, 5500 Campanile Drive, San Diego, CA 92182-7700 USA. OpenContent (<http://www.opencontent.org/openpub/>) license defines the copyright on this document.

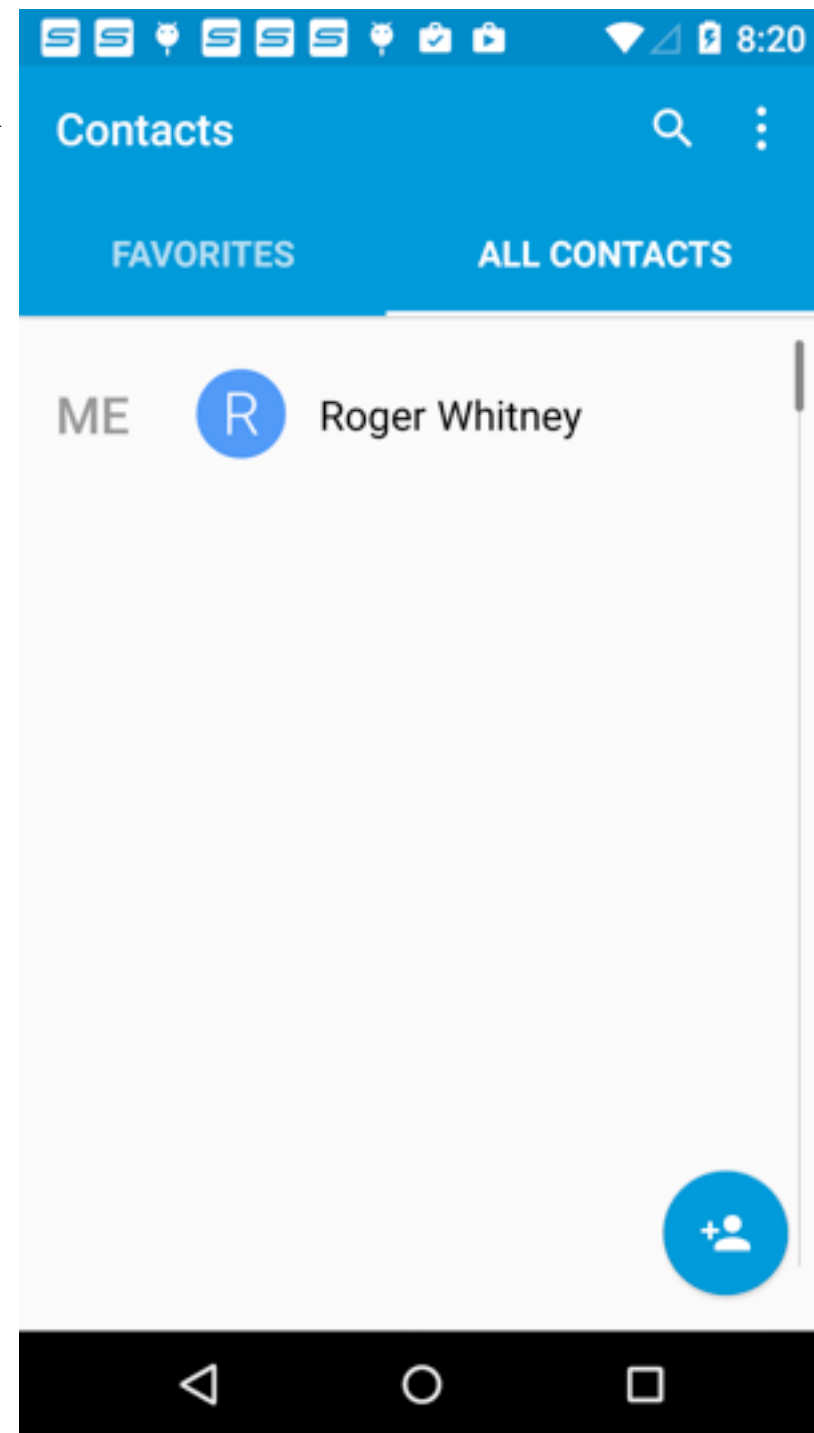
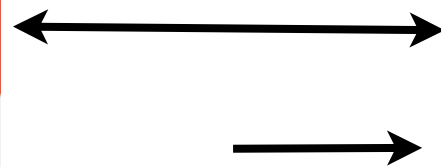
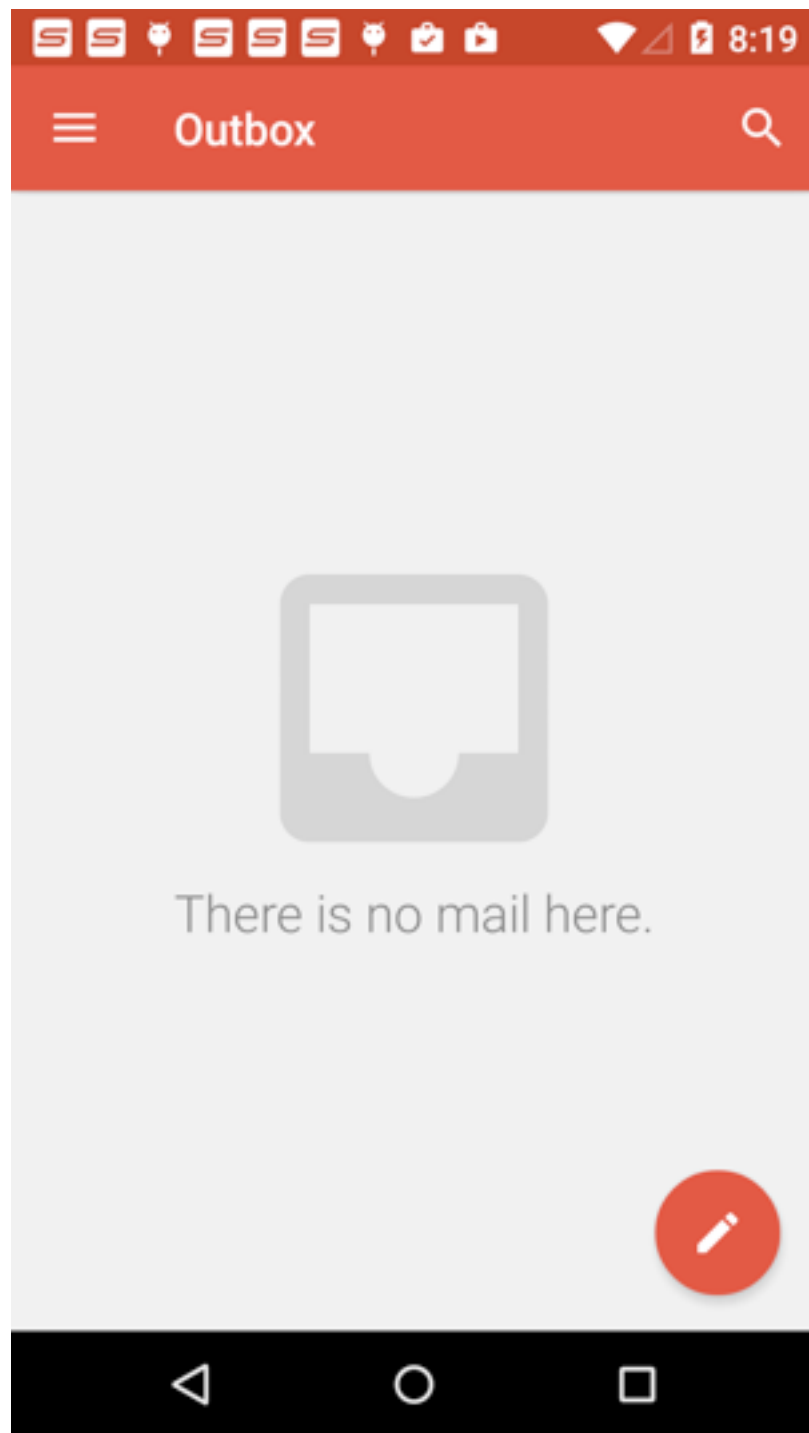
Assignment 1 Issues

Life cycle & Event methods

53 onXXX methods in Activity class

Action Bar

Added Android 3.0 (API level 11)



Activity verses ActionBarActivity

ActionBarActivity

Supports Action bar in Android 2.1 and up

Intents

Starting Activities

Calling your activities

Calling activities from other apps

Passing data to new activity

Returning data from an activity

Going back to previous activity

Intents - Calling Activities

Android application consists of multiple activities

Activity represents one screen or view

Going from one screen to another

Requires calling activity

Can't call new activity directly

Use intent to indicate activity to start

Intents

Starts another activity

Explicit Intents

Used to start activities in your app

Implicit Intents

Used to start activities from other apps

A bit more complex than explicit intents

Actions

Data URI

Category

type

filters

Explicit Intents

Explicit Intents

Specify the component (class) an intent is to run

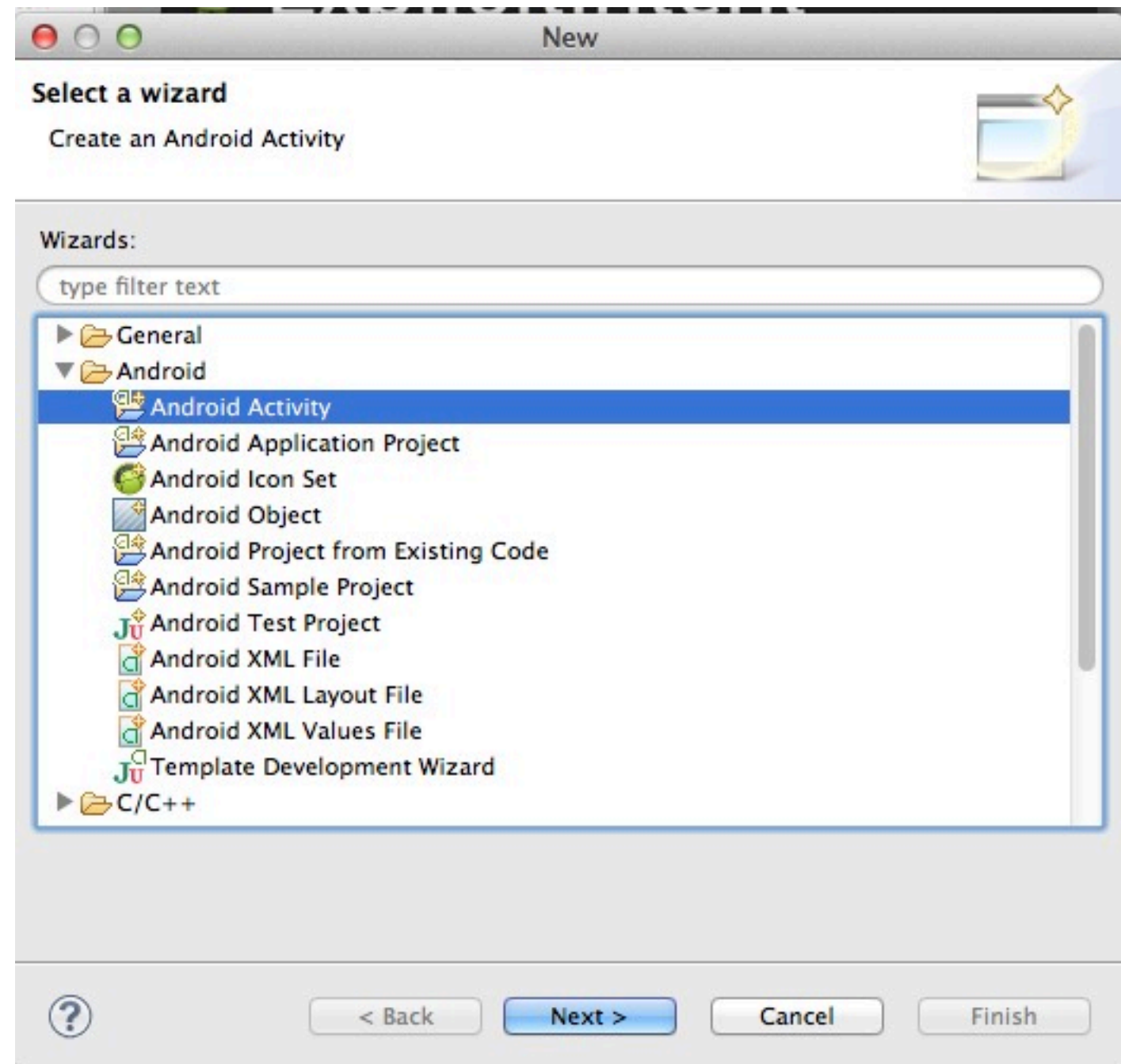
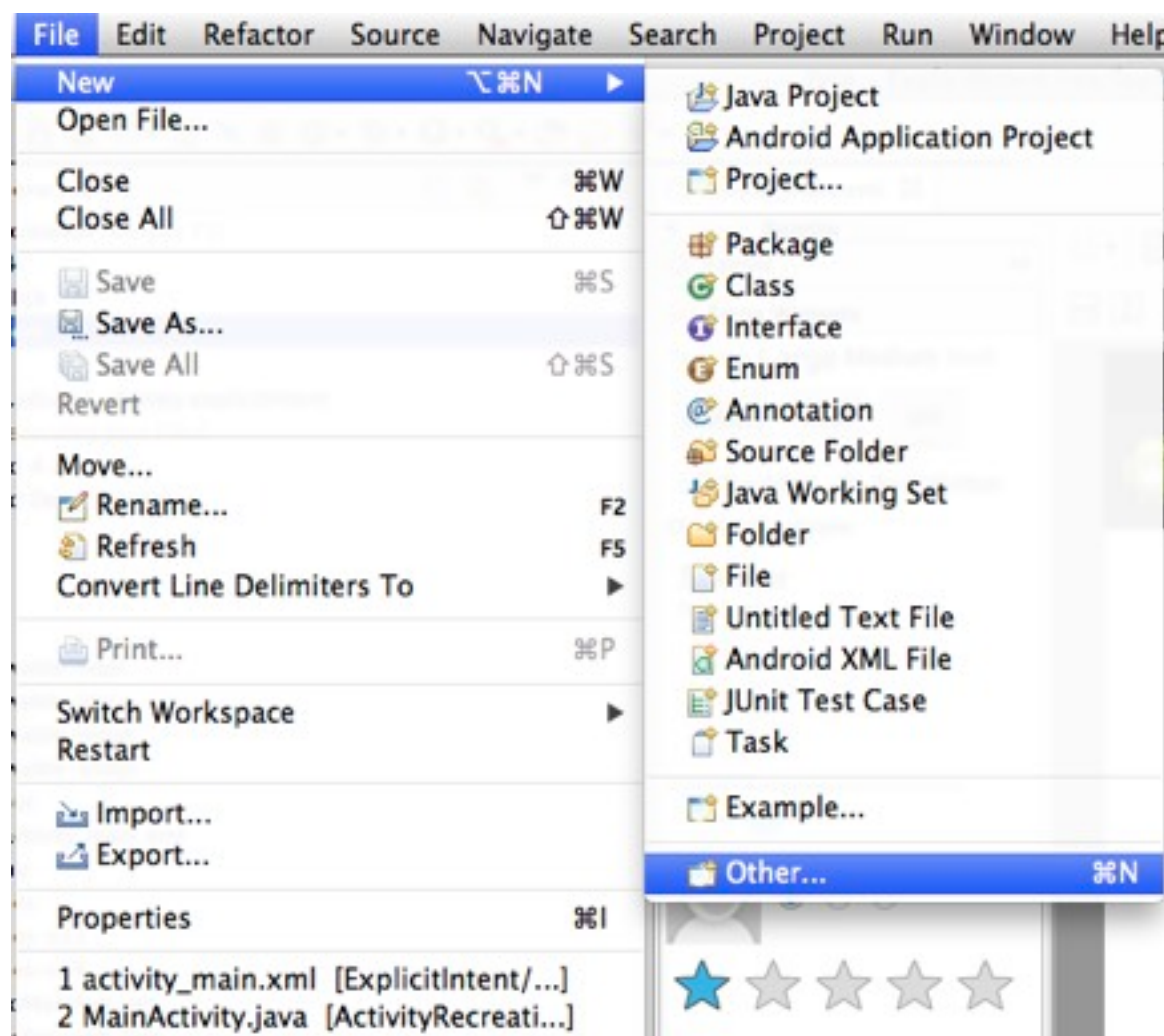
Common way to call your own code

```
Intent go = new Intent();  
go.setClassName("edu.sdsu.cs.whitney.explicitintent",  
"edu.sdsu.cs.whitney.explicitintent.SecondActivity");  
startActivity(go);
```

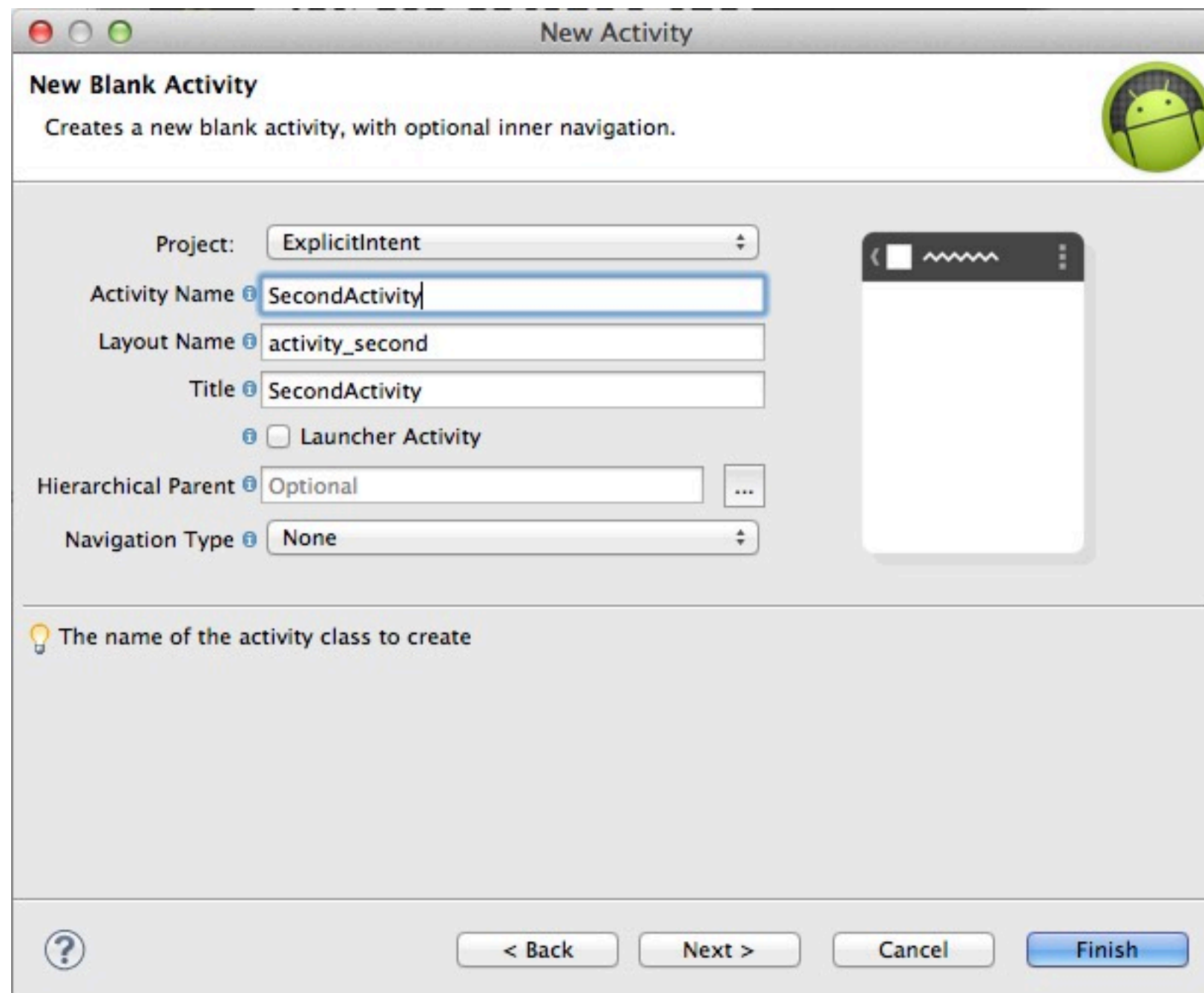
A Simpler way to create the Intent

```
Intent go = new Intent(this,SecondActivity.class);  
startActivity(go);
```

Adding a Second Activity



Adding a Second Activity



New Activity

New Blank Activity
Creates a new blank activity, with optional inner navigation.

Project: ExplicitIntent

Activity Name: SecondActivity


Layout Name: activity_second


Title: SecondActivity


☐ Launcher Activity


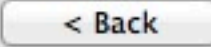


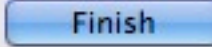
Hierarchical Parent: Optional

Navigation Type: None

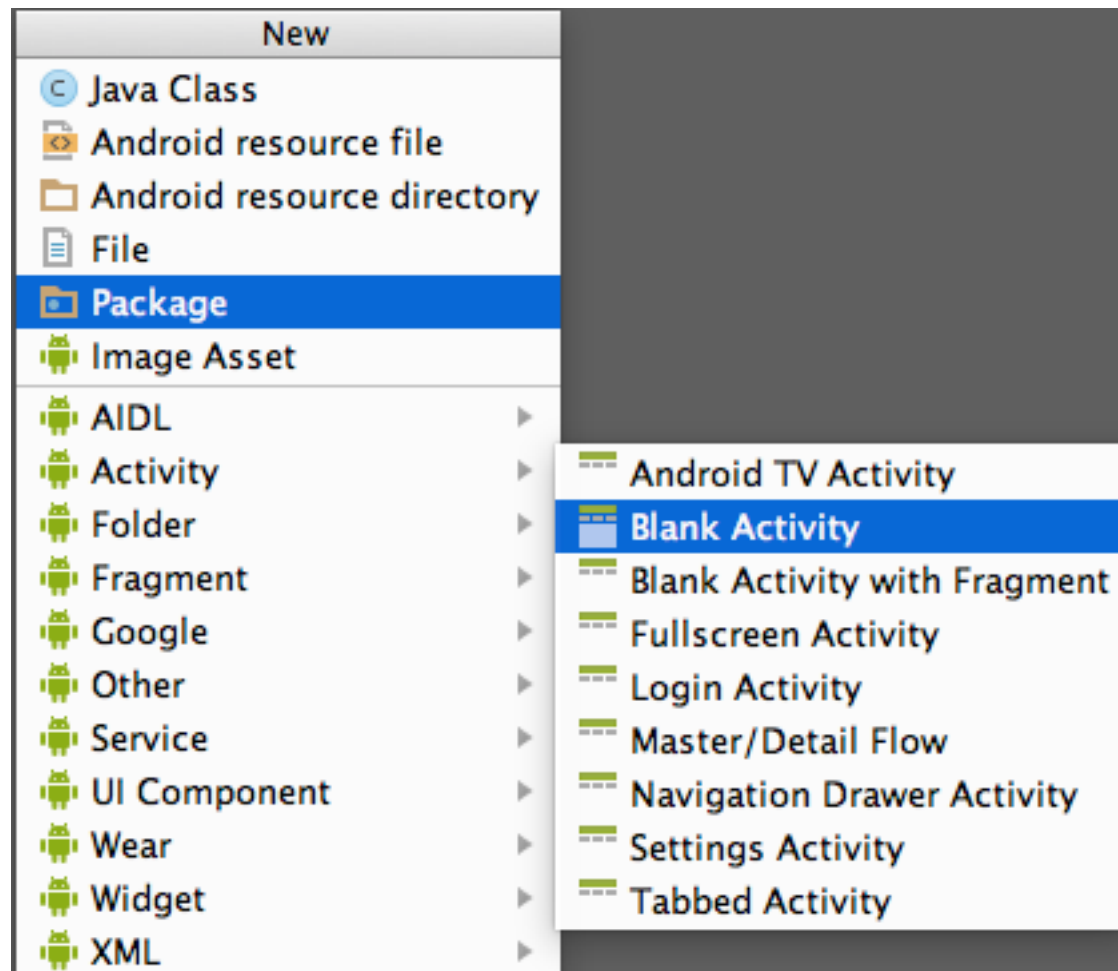




 The name of the activity class to create

Android Studio



Launcher Activity

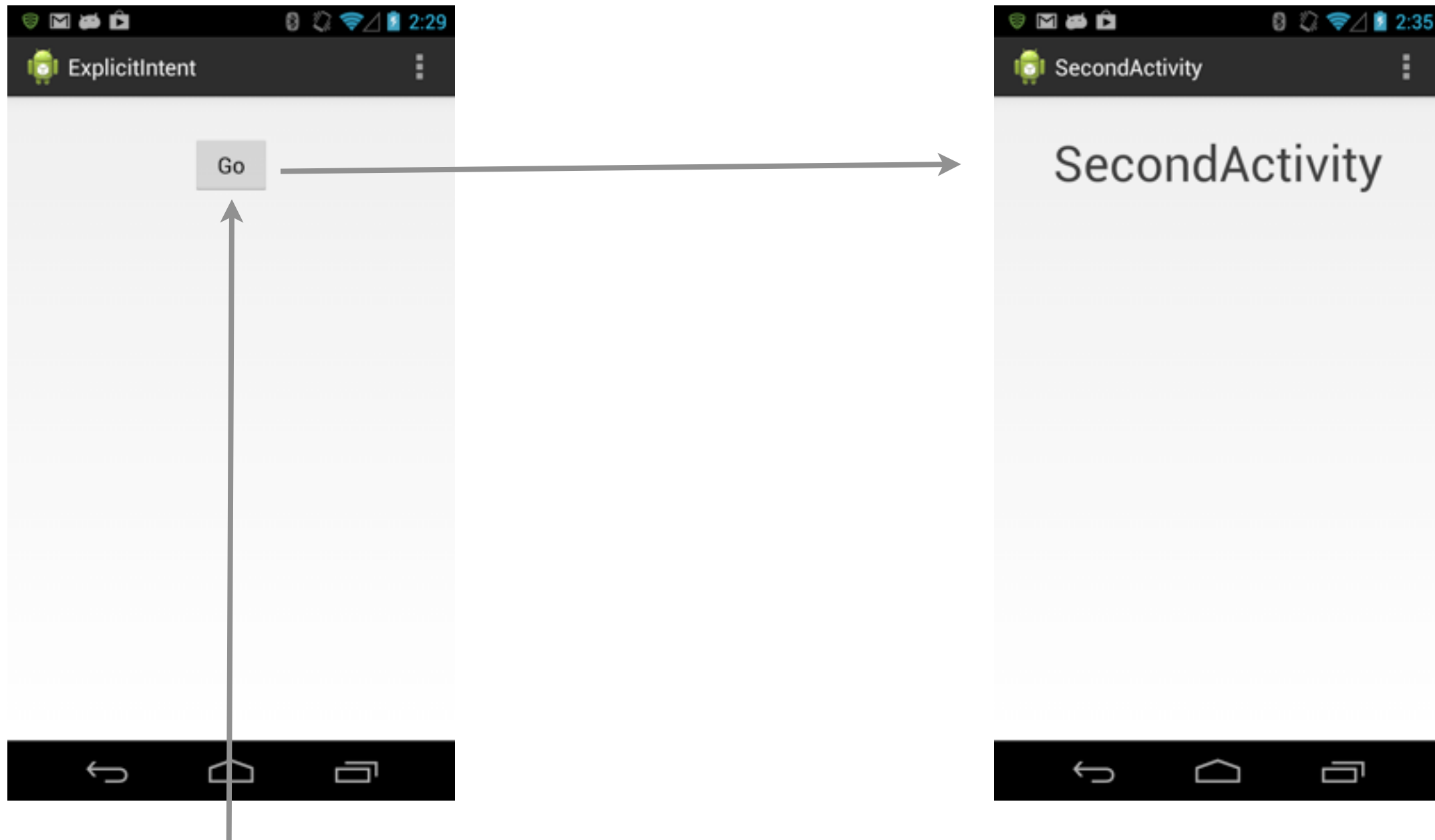
Display the activity in the list of app on device

Normally an app only has one launcher Activity

AndroidManifest.xml

```
<application
  android:allowBackup="true"
  android:icon="@drawable/ic_launcher"
  android:label="@string/app_name"
  android:theme="@style/AppTheme" >
  <activity
    android:name="edu.sdsu.cs.whitney.explicitintent.MainActivity"
    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="edu.sdsu.cs.whitney.explicitintent.SecondActivity"
    android:label="@string/title_activity_second" >
  </activity>
</application>
```

Simple Example



```
public void go(View button) {  
    Intent go = new Intent(this,SecondActivity.class);  
    startActivity(go);  
}
```

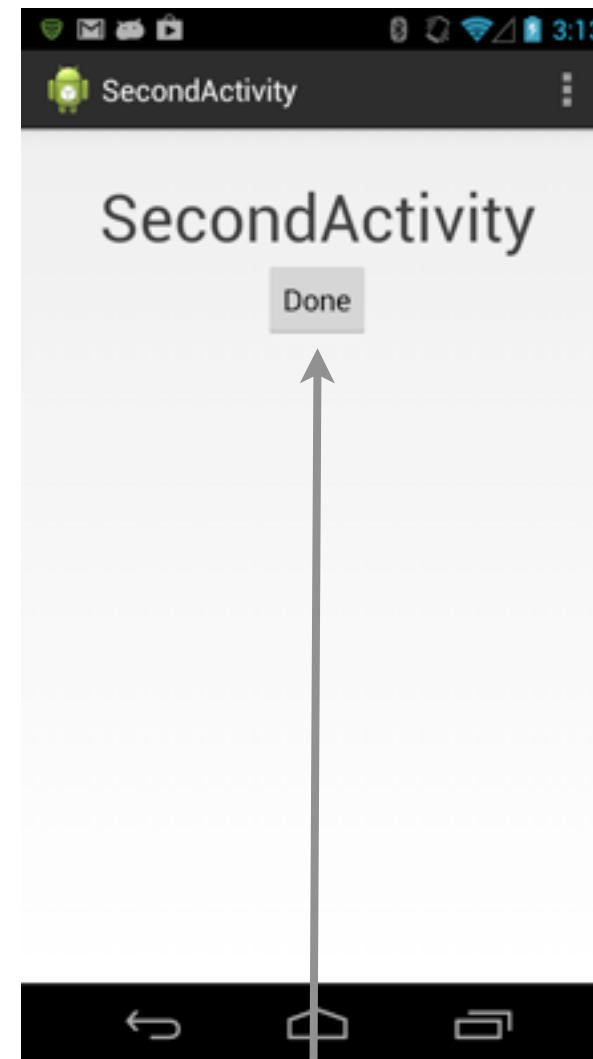
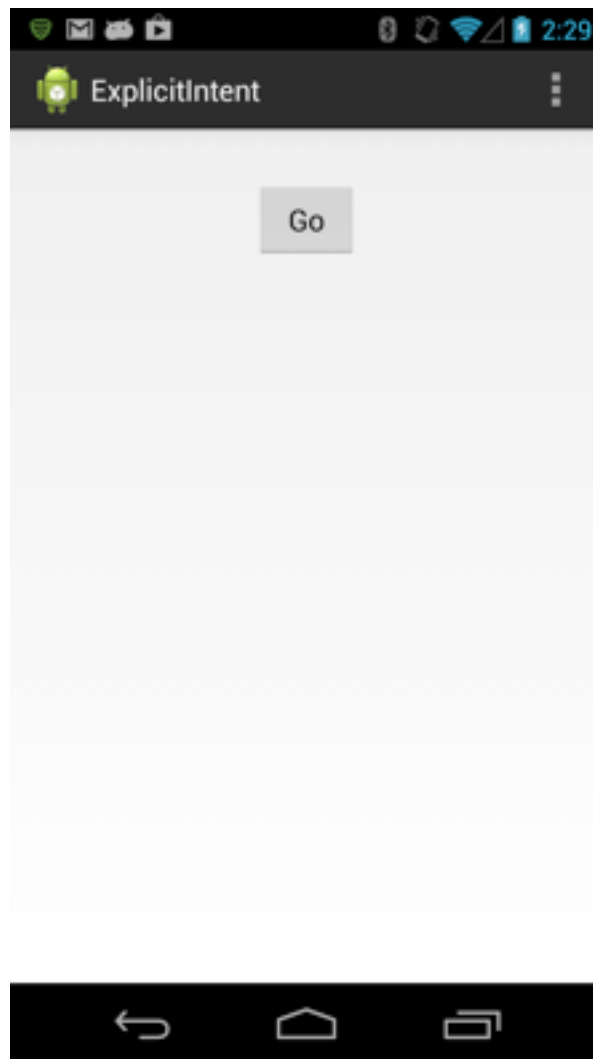
Main Activity

```
public class MainActivity extends Activity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
    }  
  
    public void go(View button) {  
        Intent go = new Intent(this, SecondActivity.class);  
        startActivity(go);  
    }  
}
```

Second Activity

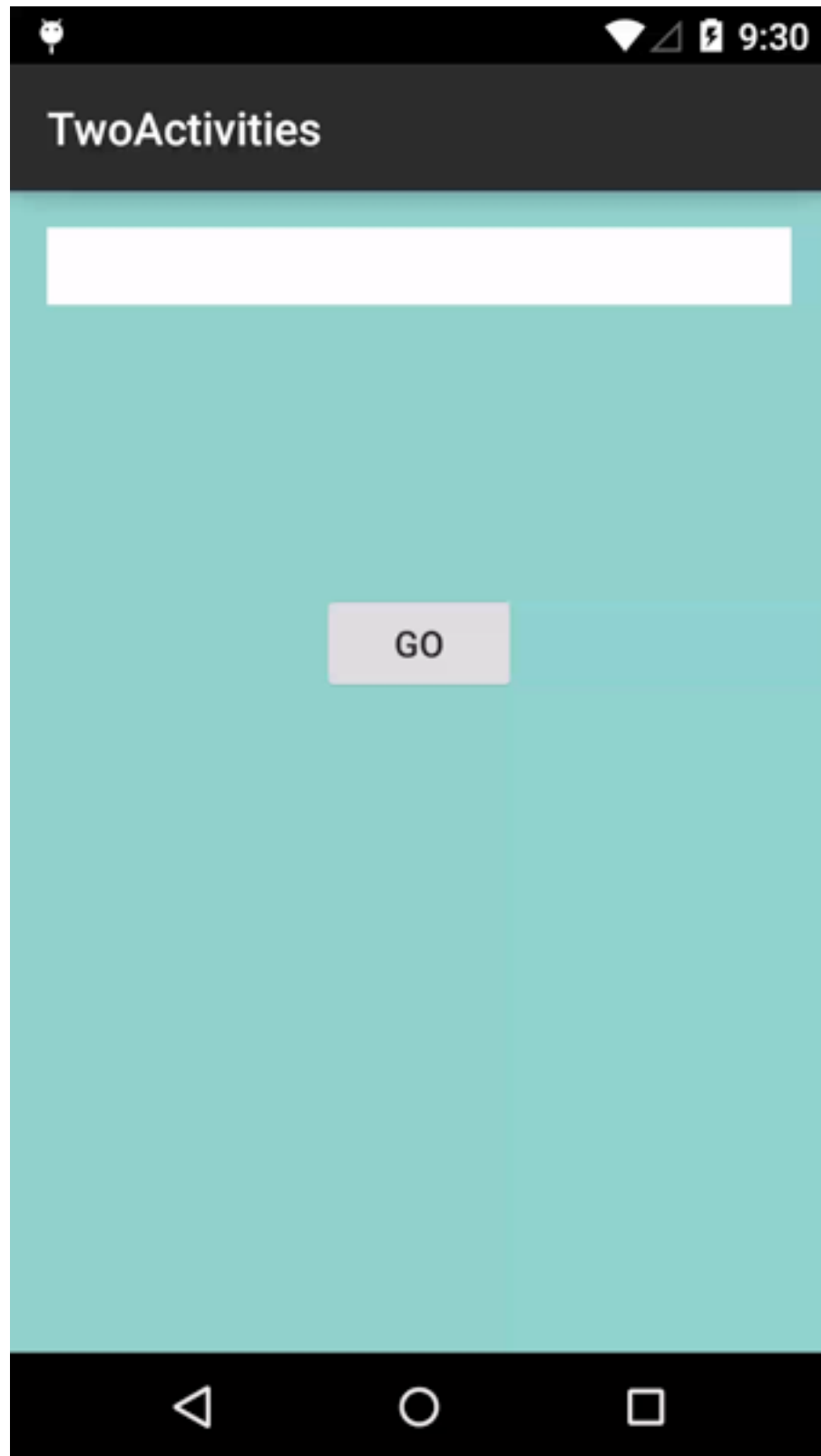
```
public class SecondActivity extends Activity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_second);  
    }  
}
```

Ending the Second Activity



```
public void back(View button) {  
    finish();  
}
```

Sample Run



finish

```
public class SecondActivity extends Activity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_second);  
    }  
  
    public void back(View button) {  
        finish();  
    }  
}
```

finish()

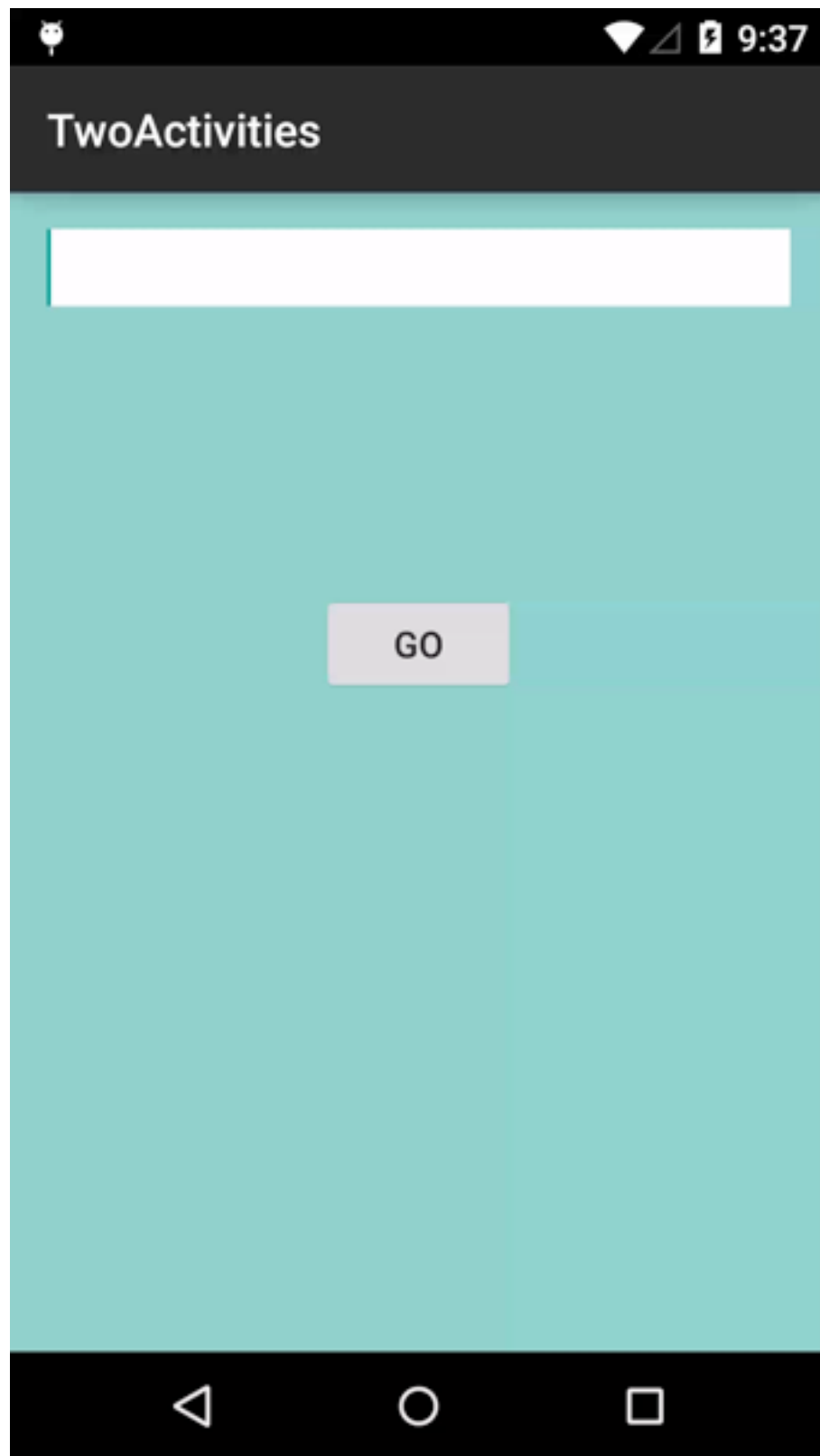
Current activity is destroyed
onStop and onDestroy are called on current activity

Go back to activity on the back stack

Intents do not go Back

```
public class SecondActivity extends ActionBarActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_second);  
    }  
  
    public void back(View button) {  
        Intent go = new Intent(this,MainActivity.class);  
        startActivity(go);  
    }  
}
```

Intents do not go Back



Passing Data to New Activity Object

In MainActivity

```
public void go(View button) {  
    Intent go = new Intent(this, SecondActivity.class);  
    go.putExtra("age", 21);  
    go.putExtra("name", "Roger");  
    startActivity(go);  
}
```

In Second Activity

```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_second);  
    Bundle personData = getIntent().getExtras();  
    int age = personData.getInt("age");  
    String name = personData.getString("name");  
}
```

Bundle can hold

Base types

boolean, byte, char, CharSequence, double, float, long, short, String

Arrays of base types

Parcelable & Serializable objects

Implement one of these interfaces in your class so bundles can hold instances

Passing Data Back

Step 1 - Calling the new Activity

In Main Activity

```
private static final int INTENT_EXAMPLE_REQUEST = 123;

public void go(View button) {
    Intent go = new Intent(this, SecondActivity.class);
    startActivityForResult(go, INTENT_EXAMPLE_REQUEST);
}
```

Passing Data Back

Step 2 - Passing back data

In Second Activity

```
public void back(View button) {  
    Log.i("rew", "Back");  
    Intent toPassBack = getIntent();  
    toPassBack.putExtra("age", 10);  
    setResult(RESULT_OK, toPassBack);  
    finish();  
}
```

Passing Data Back

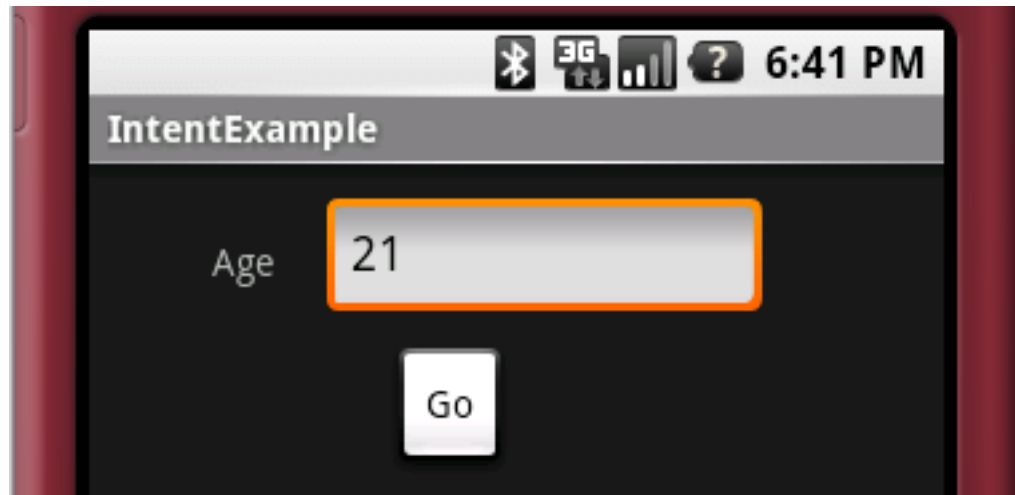
Step 3 - Reading the data

In Main Activity

```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
    if (requestCode != INTENT_EXAMPLE_REQUEST) {  
        return;  
    }  
    switch (resultCode) {  
        case RESULT_OK:  
            int editedAge = data.getIntExtra("age",-1);  
            break;  
        case RESULT_CANCELED:  
            break;  
    }  
}
```

Passing Data Example

IntentExample



Displays/Edits age

Go button

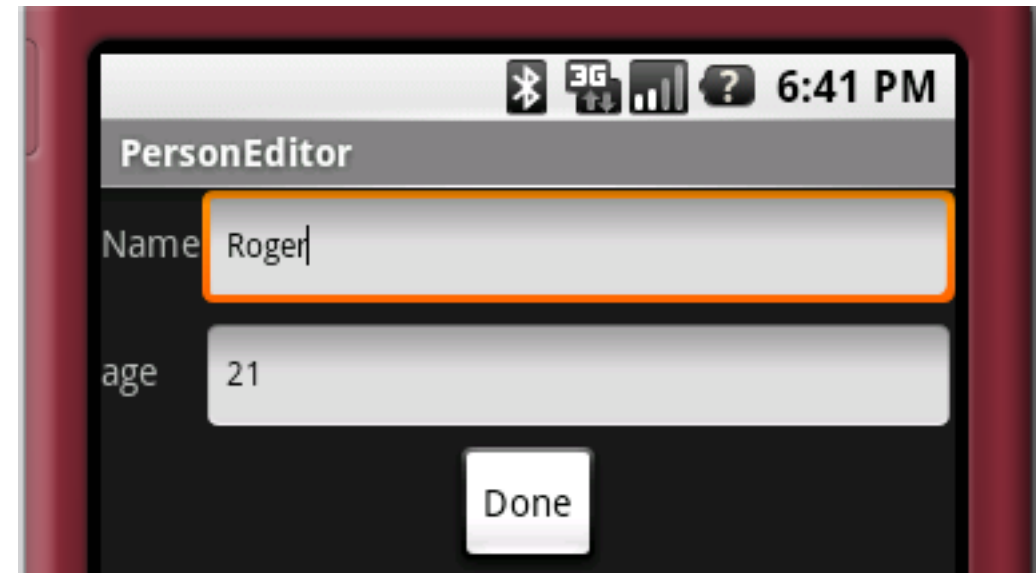
- Calls PersonEditor

- Passes data

 - Name

 - Age

PersonEditor



Displays/Edits Name and age

Done button

- Returns edited data back

- Age = 0 cancels edit

IntentExample.java

```
public class IntentExample extends Activity implements View.OnClickListener {  
    private EditText numberText;  
    private static final int INTENT_EXAMPLE_REQUEST = 123;  
  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.intent);  
        Button ok = (Button) findViewById(R.id.go);  
        ok.setOnClickListener(this);  
        numberText = (EditText) this.findViewById(R.id.number);  
        numberText.setText("21");  
    }  
}
```

IntentExample.Java continued

Sending the data to PersonEditor

```
public void onClick(View v) {  
    Intent go;  
    go = new Intent();  
    go.setAction("android.intent.action.EDIT");  
    go.addCategory("person_editor");  
    String newAge = numberText.getText().toString();  
    go.putExtra("age", newAge);  
    go.putExtra("name", "Roger");  
    startActivityForResult(go, INTENT_EXAMPLE_REQUEST);  
}
```

IntentExample.Java continued

Getting the Results back

```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
    if (requestCode != INTENT_EXAMPLE_REQUEST) {  
        numberText.setText("Not from me");  
        return;  
    }  
    switch (resultCode) {  
    case RESULT_OK:  
        String editedAge = data.getStringExtra("age");  
        numberText.setText(editedAge);  
        break;  
    case RESULT_CANCELED:  
        numberText.setText("Cancelled");  
        break;  
    }  
}  
}
```

PersonEditor.java

```
public class PersonEditor extends Activity implements View.OnClickListener {  
    private EditText ageText;  
    private EditText nameText;  
  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.person_editor);  
        Button done = (Button) findViewById(R.id.edit_done);  
        done.setOnClickListener(this);  
        ageText = (EditText) this.findViewById(R.id.edit_age);  
        nameText = (EditText) this.findViewById(R.id.edit_name);  
        Bundle personData = getIntent().getExtras();  
        String age = personData.getString("age");  
        String name = personData.getString("name");  
        if ((age != null) && (name != null)) {  
            ageText.setText(age);  
            nameText.setText(name);  
        }  
    }  
}
```

PersonEditor.java

Returning the data

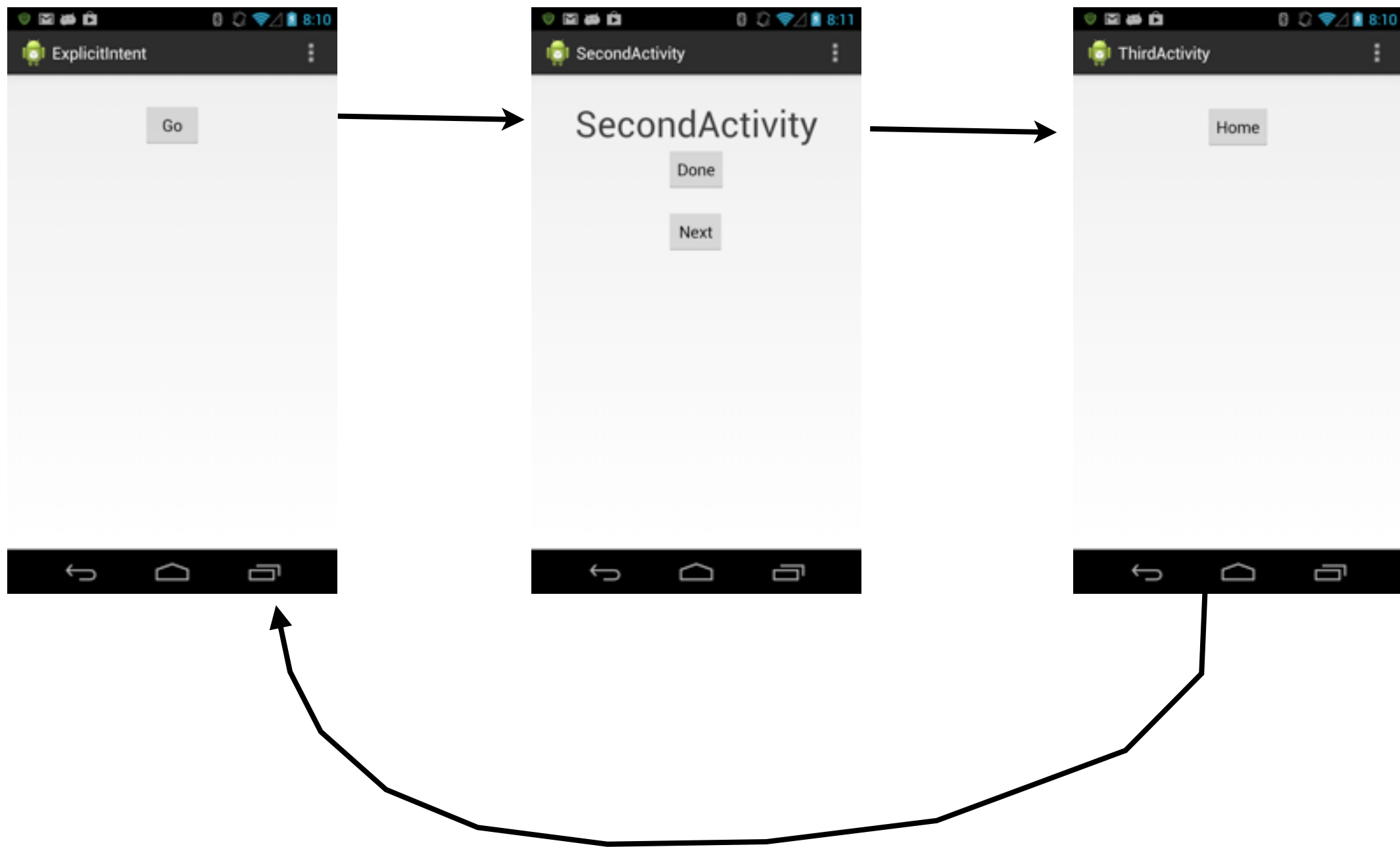
```
public void onClick(View v) {  
    String newAge = ageText.getText().toString();  
    Intent result = getIntent();  
    result.putExtra("age", newAge);  
    if (newAge.equals("0"))  
        setResult(RESULT_CANCELED, result);  
    else  
        setResult(RESULT_OK, result);  
    finish();  
}
```

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="edu.sdsu.cs683.example" android:versionCode="1"
    android:versionName="1.0.0">
    <application android:icon="@drawable/icon" android:label="@string/app_name">
        <activity android:name=".IntentExample" 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:label="PersonEditor" android:name="PersonEditor">
            <intent-filter>
                <action android:name="android.intent.action.EDIT"></action>
                <category android:name="person_editor"></category>
                <category android:name="android.intent.category.DEFAULT">
                    </category>
                </intent-filter>
            </activity>
        </application>
    </manifest>
```

Going Back More than One Activity

SDK 16 and greater



navigateUpTo(intent)

SDK 16 and greater

Goes back to

Activity indicated by the intent

To first activity if can not find indicated activity

```
public void goHome(View button) {  
    Intent goHome = new Intent(this,MainActivity.class);  
    boolean foundHome = navigateUpTo(goHome);  
    Log.i("rew", "Found home - " + foundHome);  
}
```


Implicit Intents

Implicit Intents

Specify which activity to call indirectly

Action

Each activity specifies actions it supports
Specified in manifest file

Category

Used to narrow down which activity you want

If more than one activity matches your request user selects the one to use

Intent Filters

Each activity declares in manifest what it can handles

```
<intent-filter>  
    <action android:name="android.intent.action.MAIN" />  
    <category android:name="android.intent.category.LAUNCHER" />  
</intent-filter>
```

Intents Handled By Google Android Apps

Scheme	Action	Description
http://web_address https://web_address	VIEW	Open a browser window to the URL specified.
"" (empty string) http://web_address https://web_address	WEB_SEARCH	Opens the file at the location on the device in the browser.
tel: phone_number	CALL	Calls the entered phone number.
tel:phone_number voicemail:	DIAL	Dials but does not actually initiate the call the number given
geo:latitude,longitude geo:latitude,longitude?z=zoom geo:0,0?q=my+street+address geo:0,0?q=business+near+city	VIEW	Opens the Maps application to the given location

First Intent Example - Dial Phone

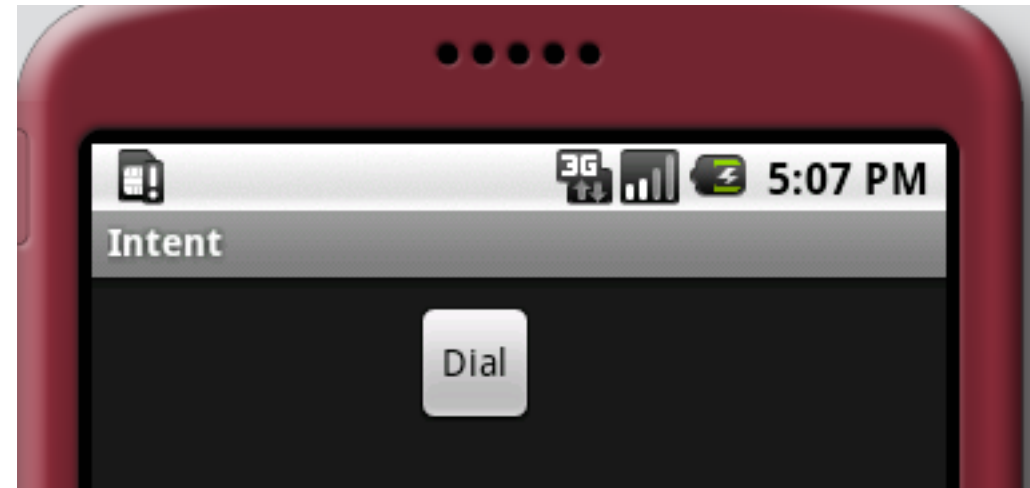
Activity with button

When button is pressed

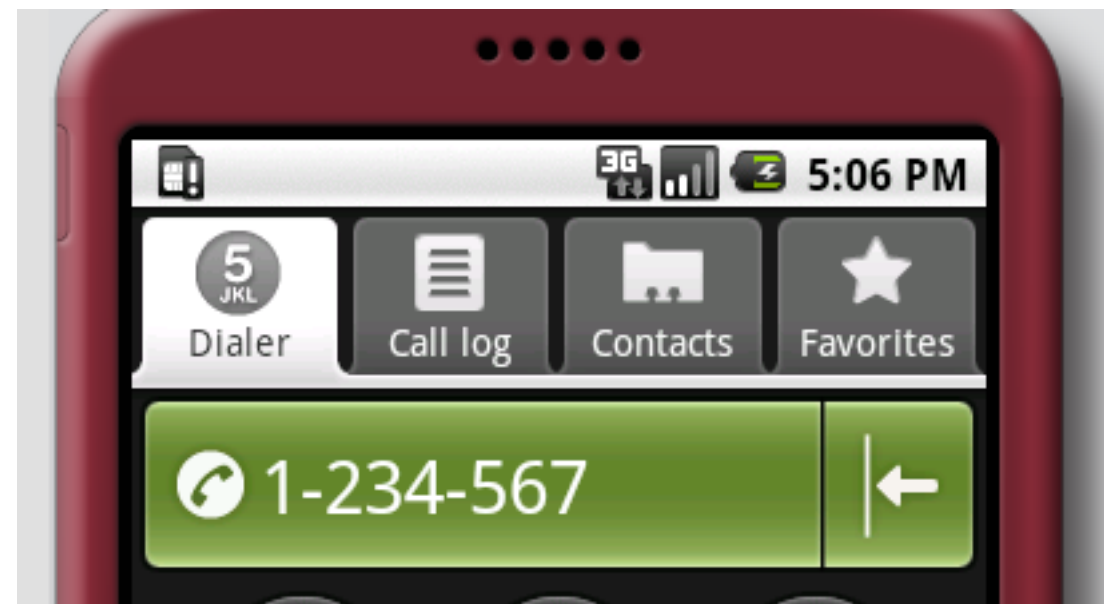
- Phone activity is run

- Phone number is entered

- Phone number is hard coded



Implicit Intent to another application



IntentExample.java

```
public class IntentExample extends Activity implements View.OnClickListener {

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.intent);
        Button ok = (Button) findViewById(R.id.go);
        ok.setOnClickListener(this);
    }

    public void onClick(View v) {
        Intent dial = new Intent();
        dial.setAction(android.content.Intent.ACTION_DIAL);
        dial.setData(Uri.parse("tel:1234567"));
        startActivity(dial);
    }
}
```

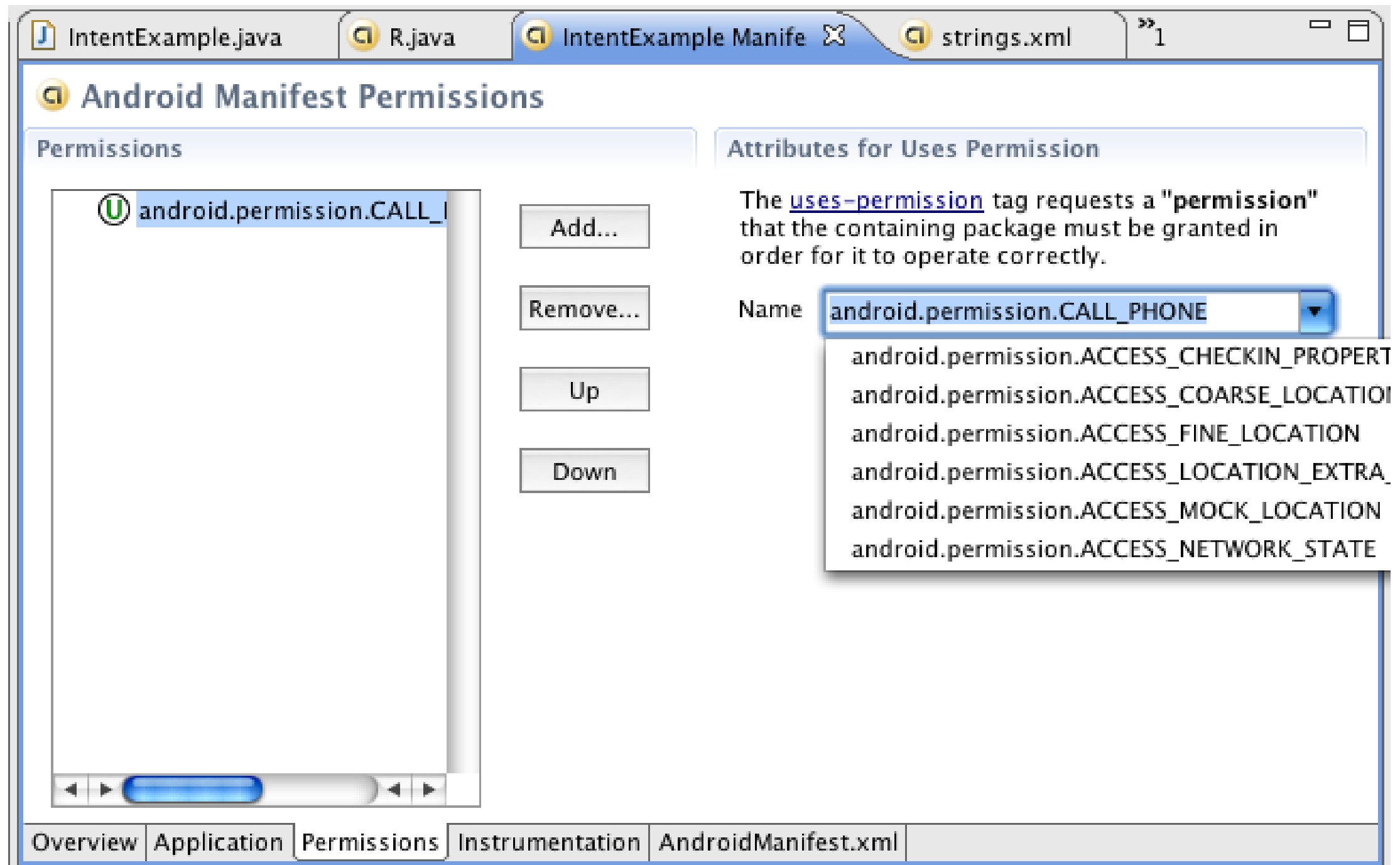
Other Ways to create the Intent

```
public void onClick(View v) {  
    Intent dial = new Intent(Intent.ACTION_DIAL, Uri.parse("tel:1234567"));  
    startActivity(dial);  
}
```

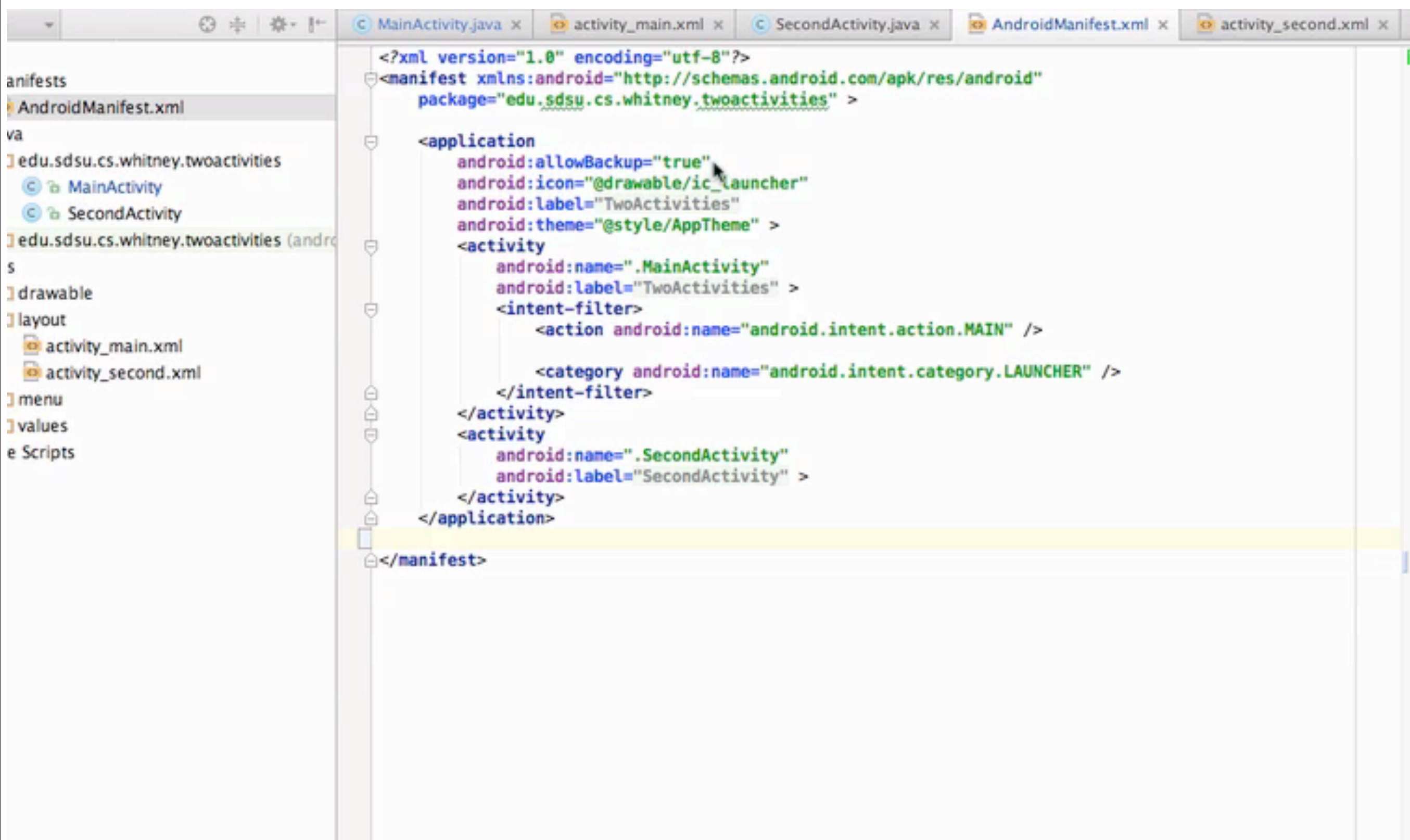
IntentExamples Manifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="edu.sdsu.cs696"
    android:versionCode="1"
    android:versionName="1.0">
    <application android:icon="@drawable/icon" android:label="@string/app_name">
        <activity android:name=".IntentExample"
            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="2" />
    <uses-permission android:name="android.permission.CALL_PHONE"></uses-
permission>
</manifest>
```


Adding the Permission - Eclipse



Adding the Permission - Android Studio

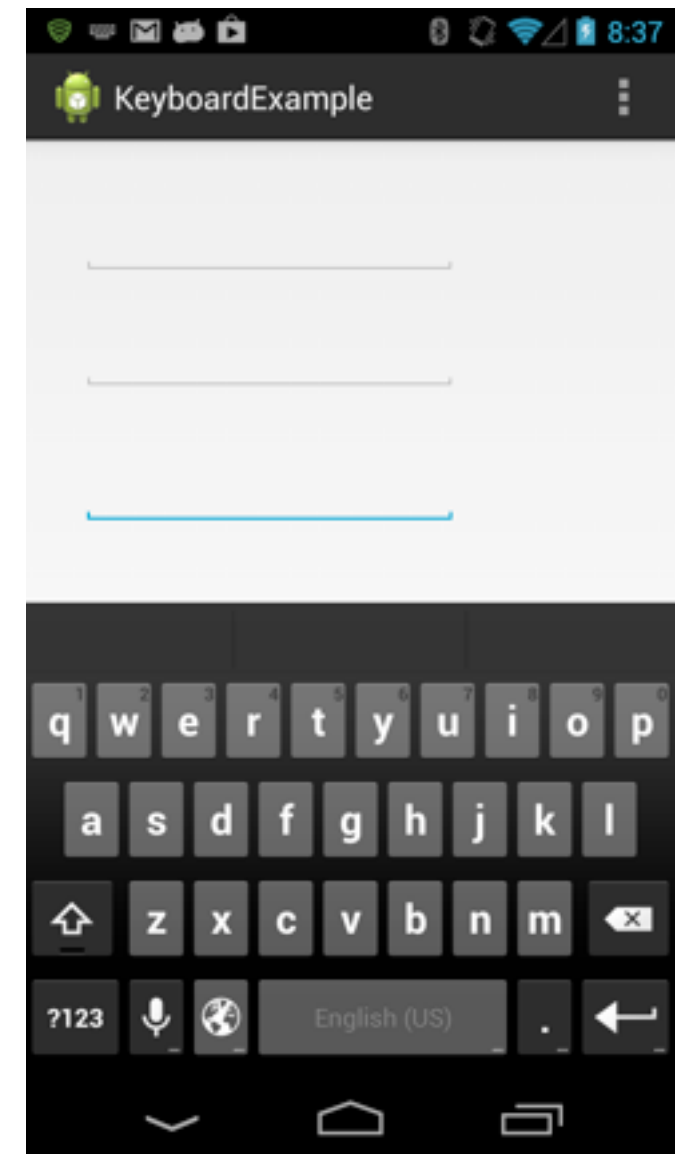
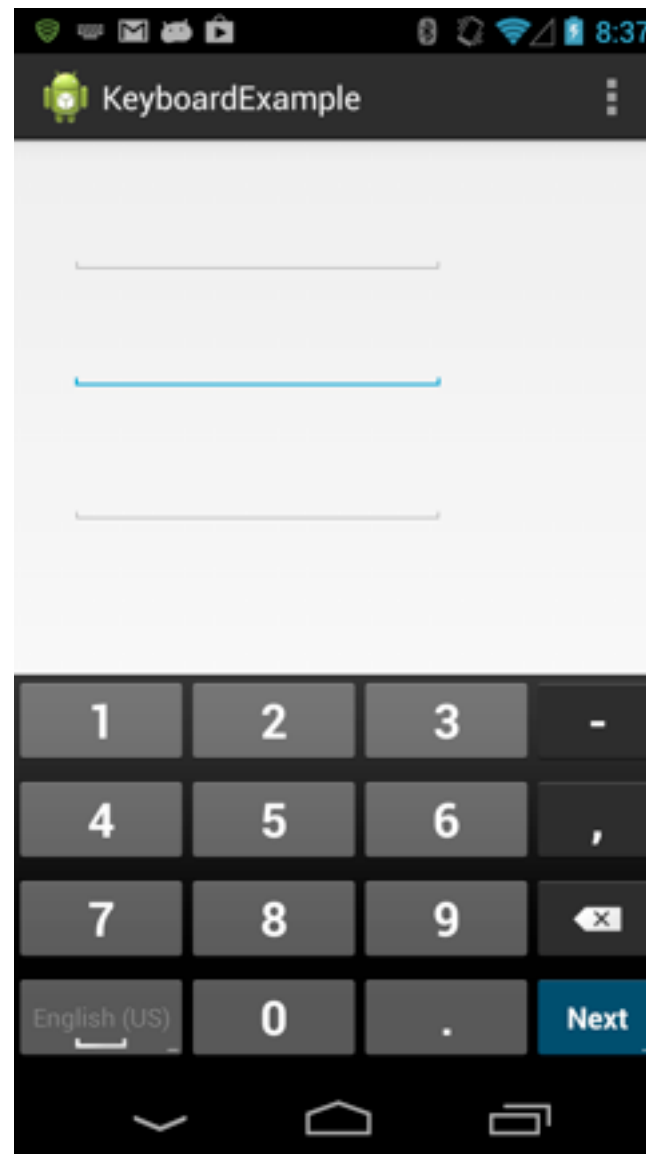
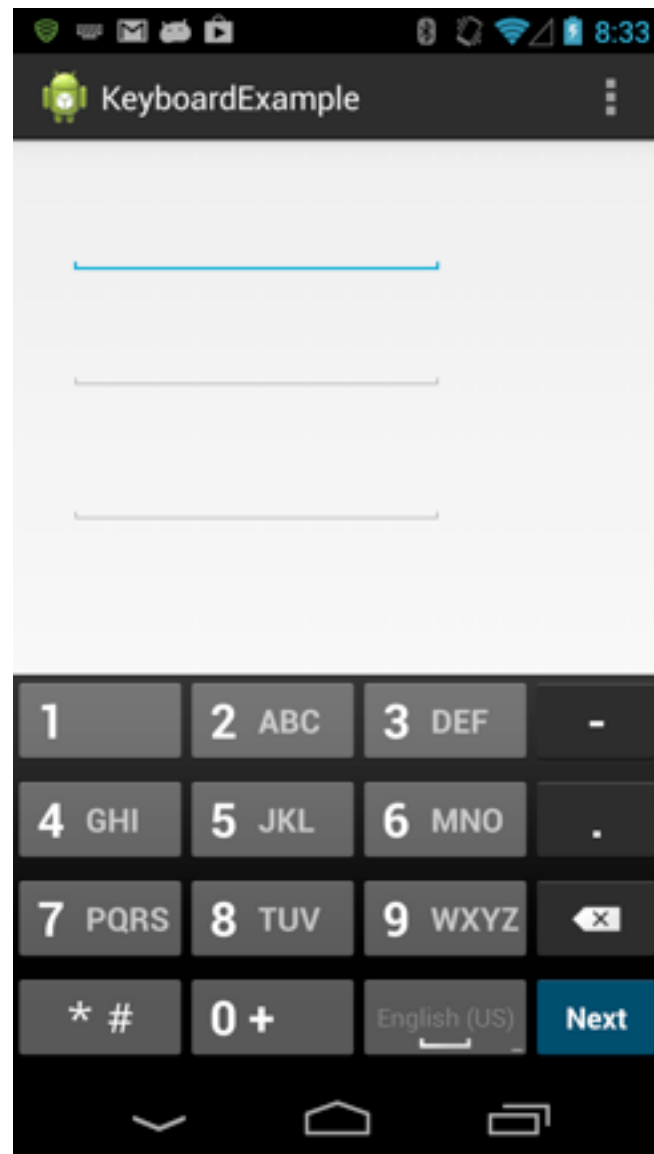


intent.xml

```
<?xml version="1.0" encoding="utf-8"?>
<AbsoluteLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:id="@+id/layout"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
<Button
    android:id="@+id/go"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/go_button"
    android:gravity="center"
    android:layout_x="120px"
    android:layout_y="10px"
    >
</Button>
</AbsoluteLayout>
```

Keyboard

Provide the Correct Keyboard



Input Type

```
<EditText  
    android:id="@+id/editText1"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_alignParentLeft="true"  
    android:layout_alignParentTop="true"  
    android:layout_marginLeft="30dp"  
    android:layout_marginTop="36dp"  
    android:ems="10"  
    android:inputType="phone" >
```

Values for Input Type

Some Common

"text"

"textEmailAddress"

"textUri"

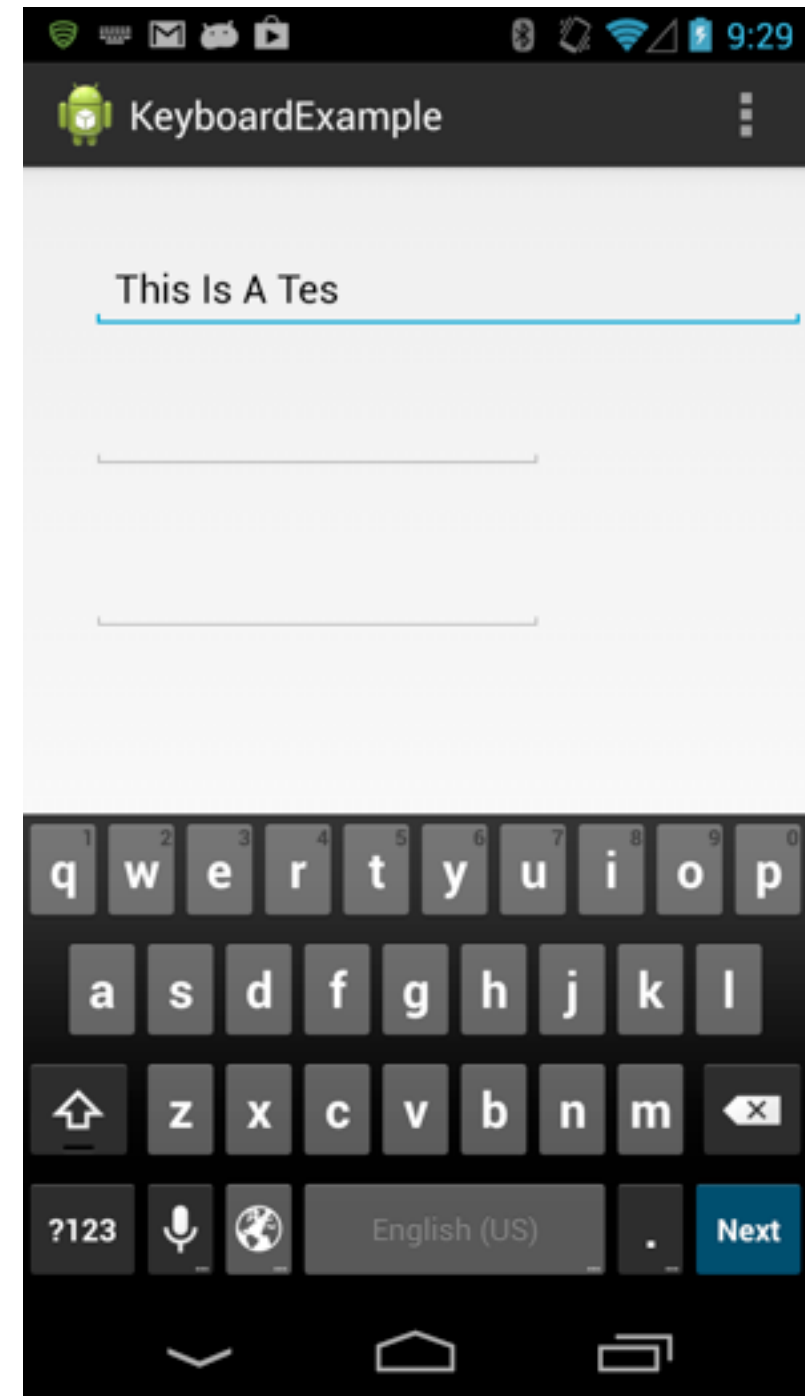
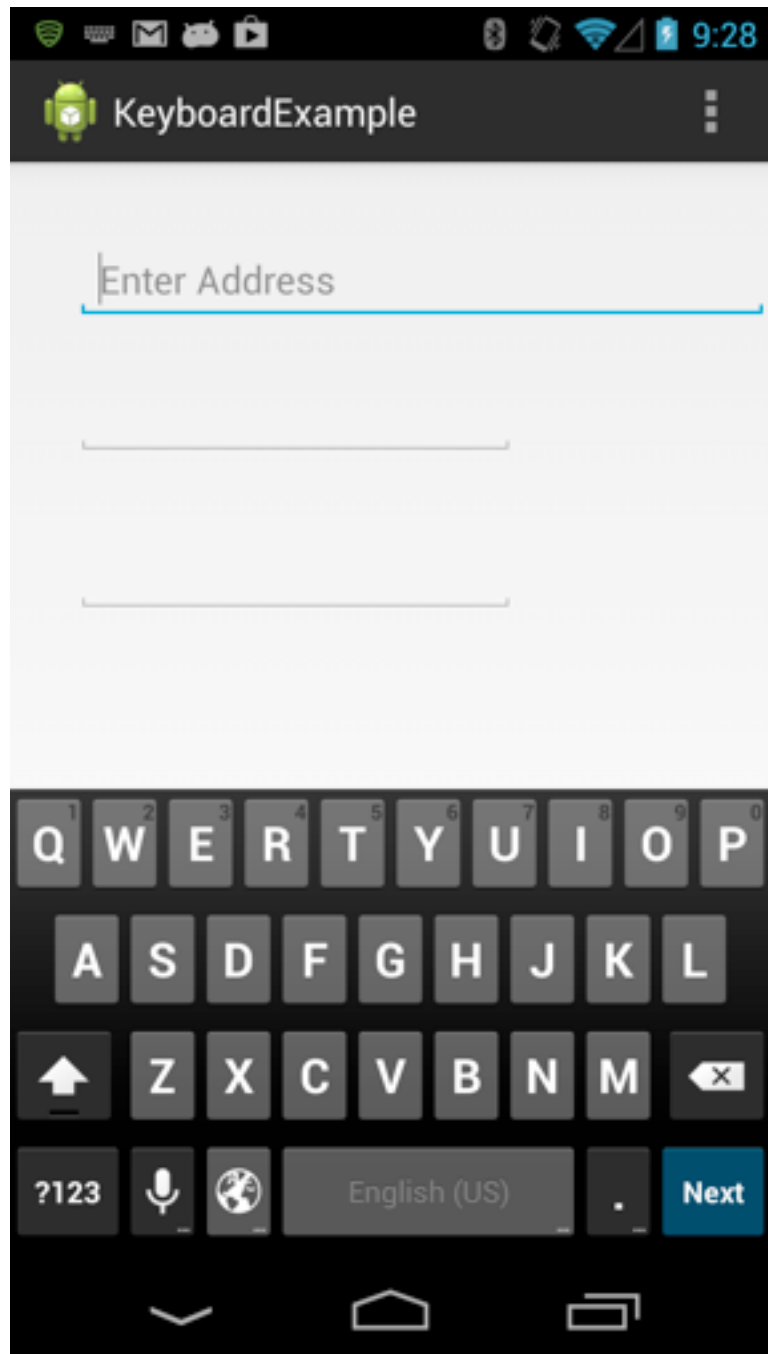
"number"

"phone"

Full List

http://developer.android.com/reference/android/widget/TextView.html#attr_android:inputType

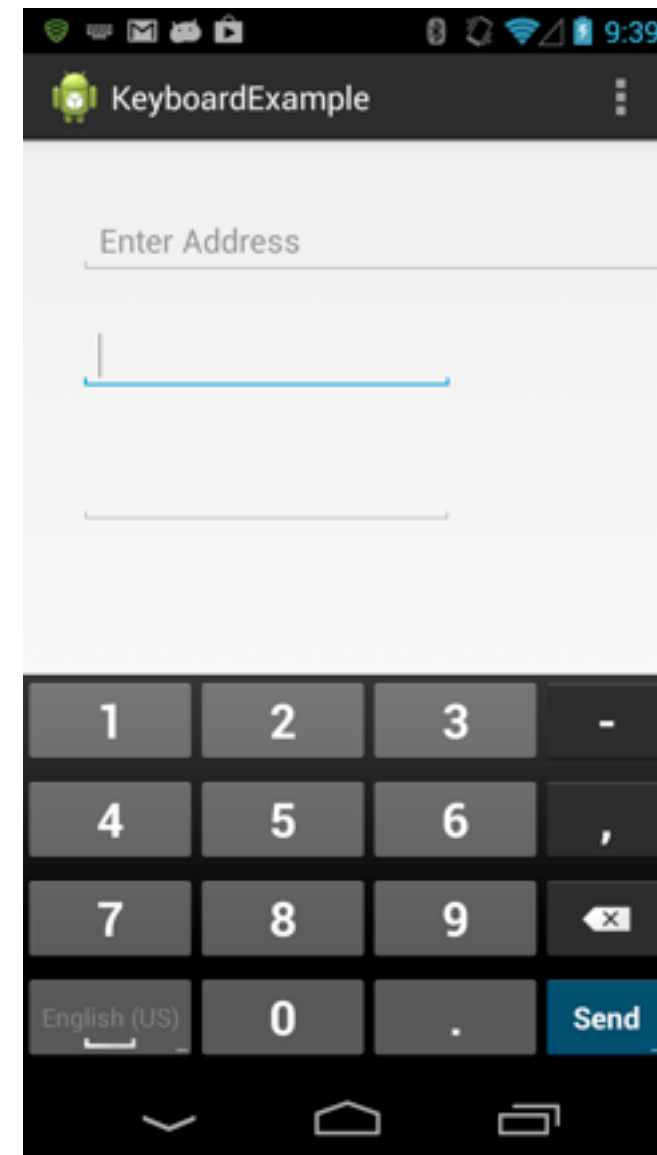
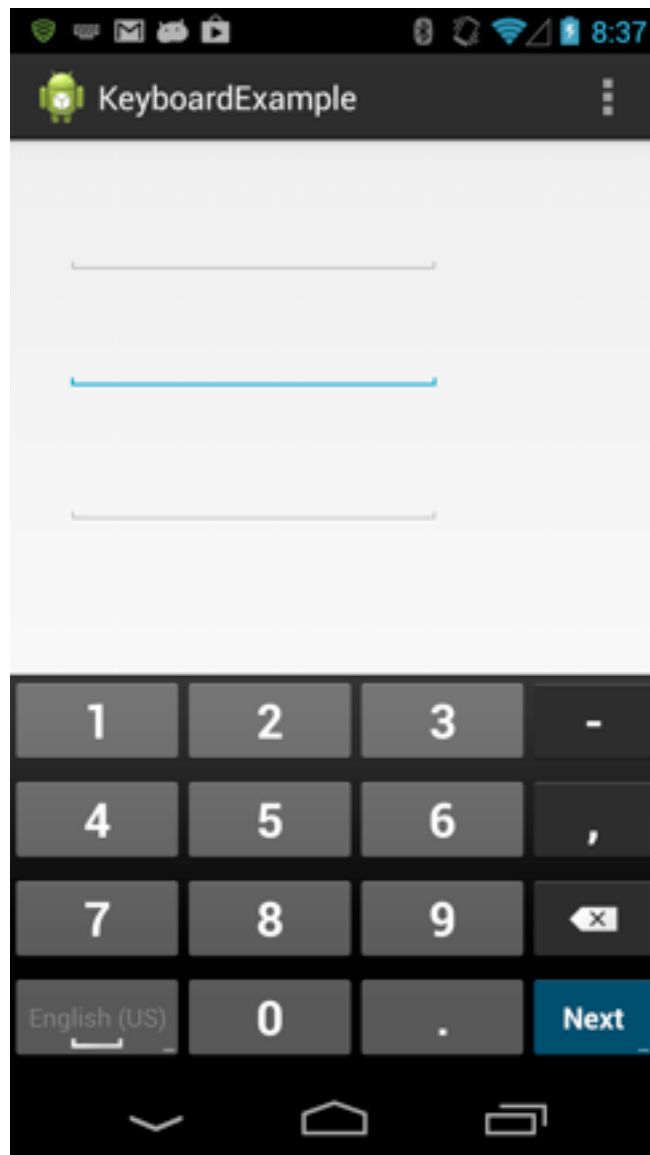
Combining Text Features



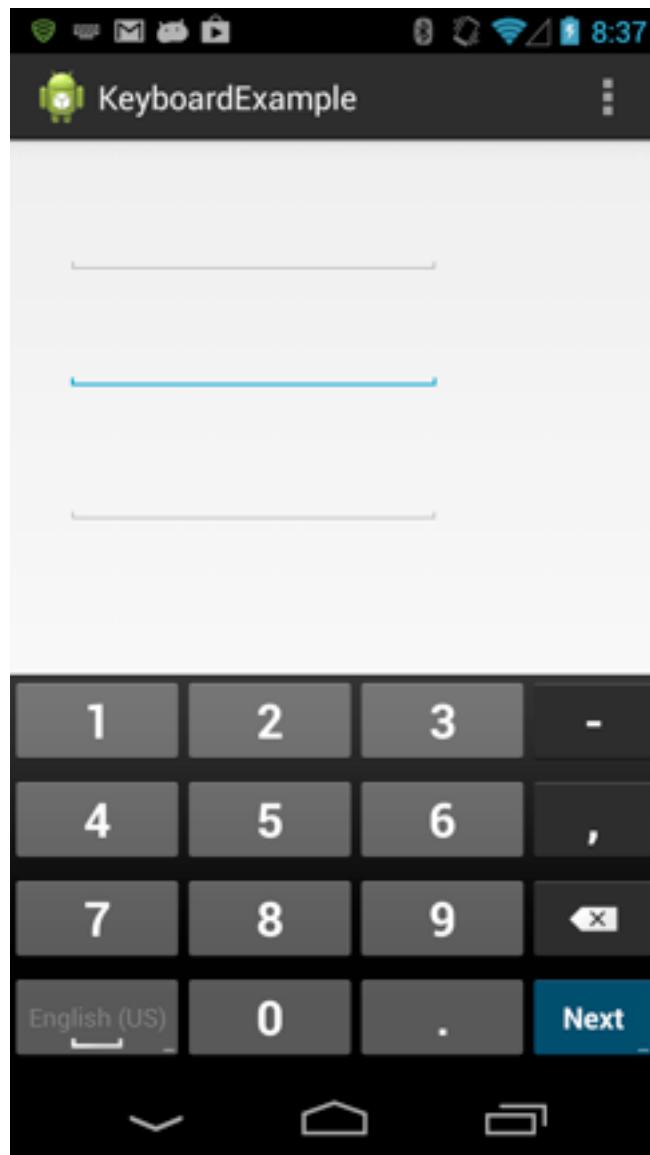
Combining Text Features

```
<EditText
    android:id="@+id/editText1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:ems="20"
    android:hint="@string/postal_address_hint"
    android:inputType="textPostalAddress|textCapWords|textNoSuggestions" >
    <requestFocus />
</EditText>
```

Keyboard Actions



Keyboard Actions



Default action

Next

If other edit fields come after

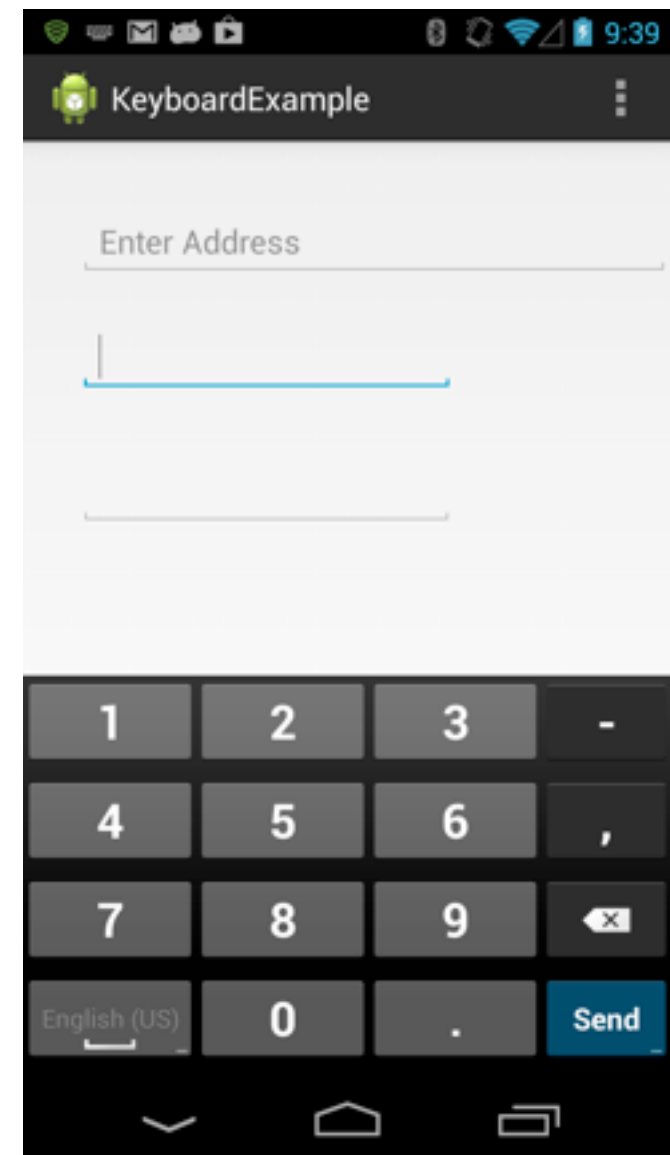
Done

If last edit field

Specifying Keyboard Action

```
<EditText  
    android:id="@+id/editText2"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:ems="10"  
    android:inputType="number"  
    android:imeOptions="actionSend" />
```

actionNone
actionGo
actionSearch
actionSend
actionNext
actionDone
actionPrevious



Responding to Action Button Events

```
public class MainActivity extends Activity implements OnEditorActionListener{

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        EditText editText = (EditText) findViewById(R.id.sendField);
        editText.setOnEditorActionListener(this);
    }
```

Responding to Action Button Events

```
public boolean onEditorAction(TextView arg0, int actionId, KeyEvent event) {  
    boolean handled = false;  
    if (actionId == EditorInfo.IME_ACTION_SEND) {  
        Log.i("rew", "Got the message");  
        //Respond here  
        handled = true;  
    }  
    return handled;  
}
```

Hiding the Keyboard

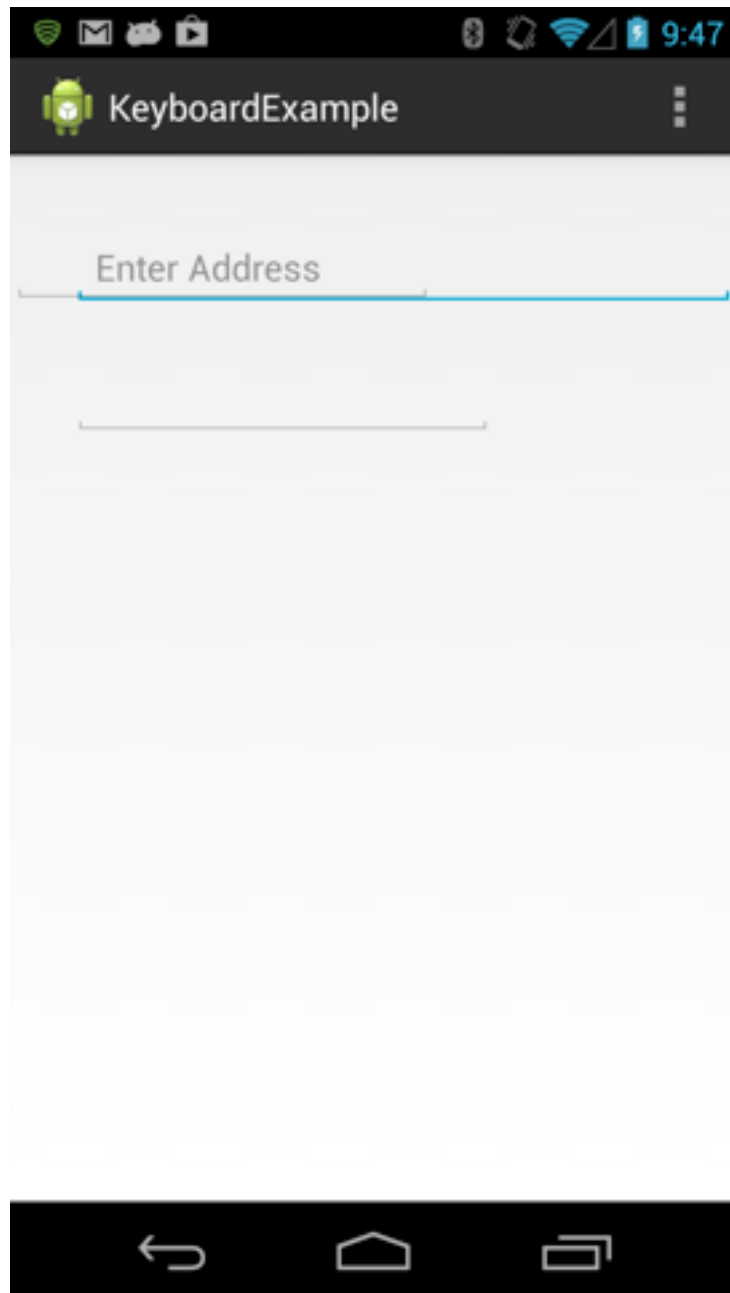
The user can hide the keyboard using the back button

But you may wish to do it programmatically

Input manager needs reference to field that has focus

```
sendField = (EditText) findViewById(R.id.sendField);  
InputMethodManager manager;  
manager =(InputMethodManager) getSystemService(INPUT_METHOD_SERVICE);  
manager.hideSoftInputFromWindow(sendField.getWindowToken(), 0);
```

Showing the Keyboard



When activity starts first text field is given focus

But keyboard is not shown

User has to tap the field

Showing the Keyboard

Set android:windowSoftInputMode in manifest file

To have keyboard show up when when activity starts and have field with focus

```
<activity
  android:name="edu.sdsu.cs.whitney.keyboardexample.MainActivity"
  android:label="@string/app_name"
  android:windowSoftInputMode="stateVisible">
  <intent-filter>
    <action android:name="android.intent.action.MAIN" />
    <category android:name="android.intent.category.LAUNCHER" />
  </intent-filter>
</activity>
```

Showing the Keyboard

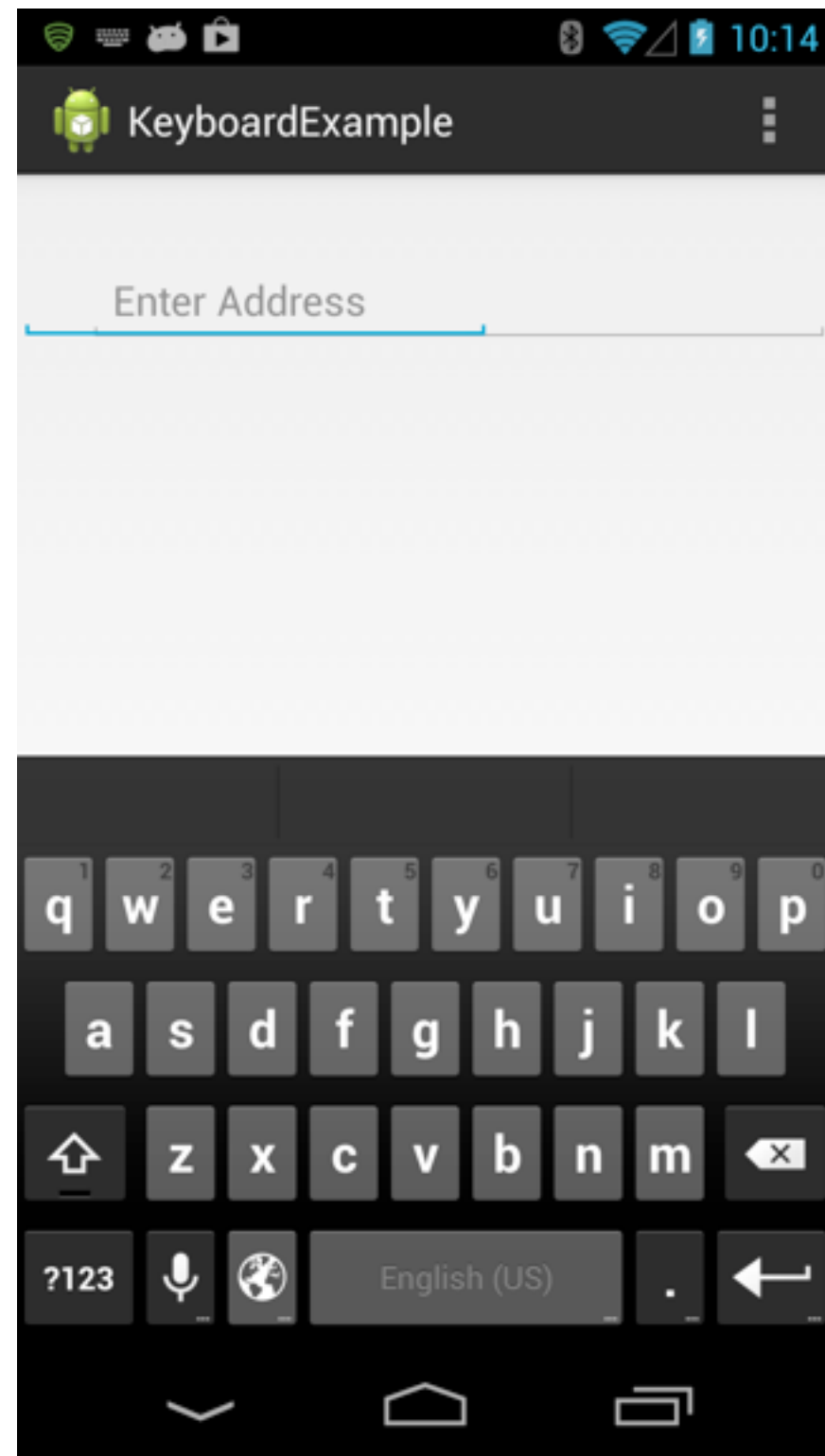
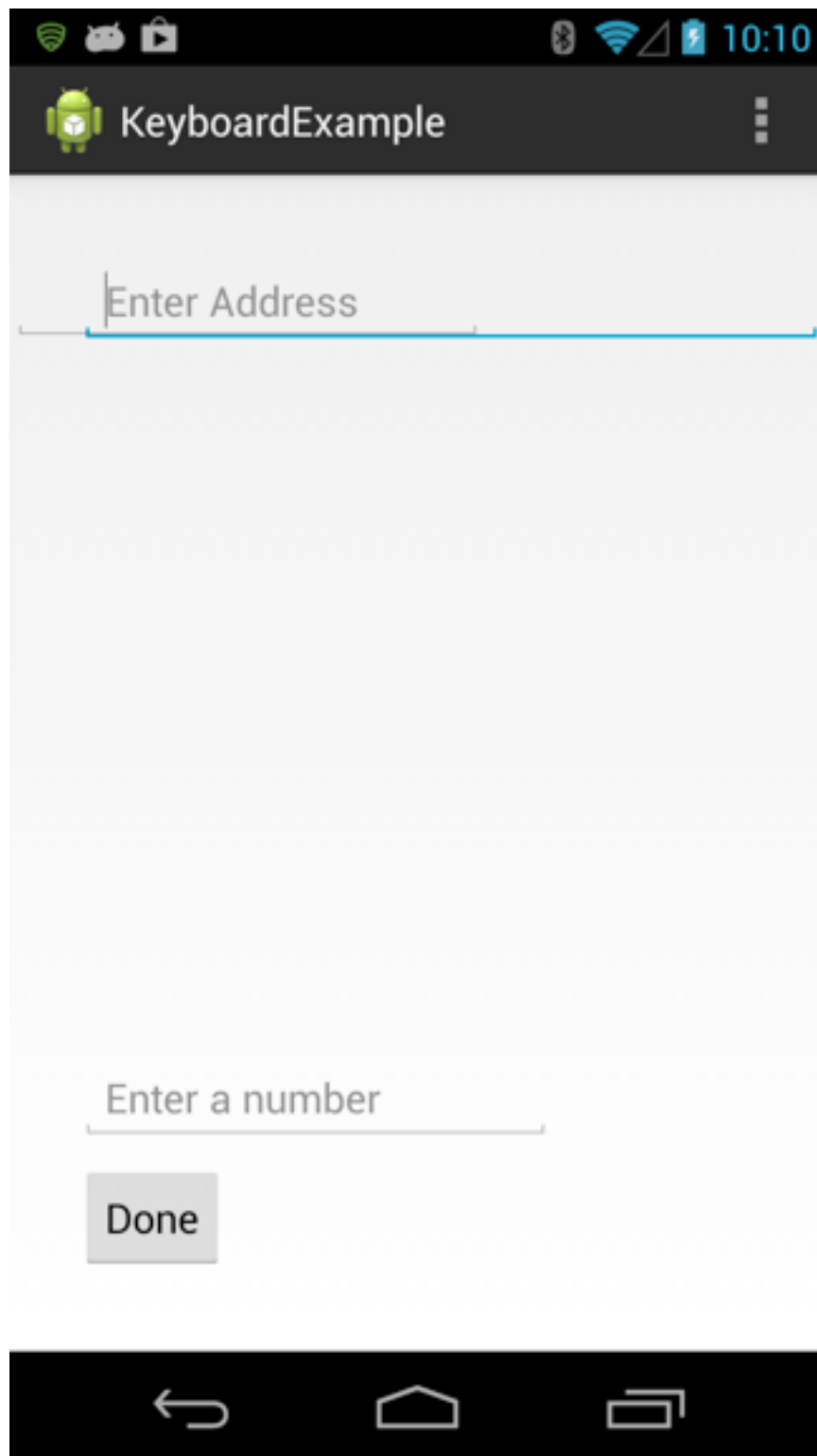
Your code can

- Give view focus

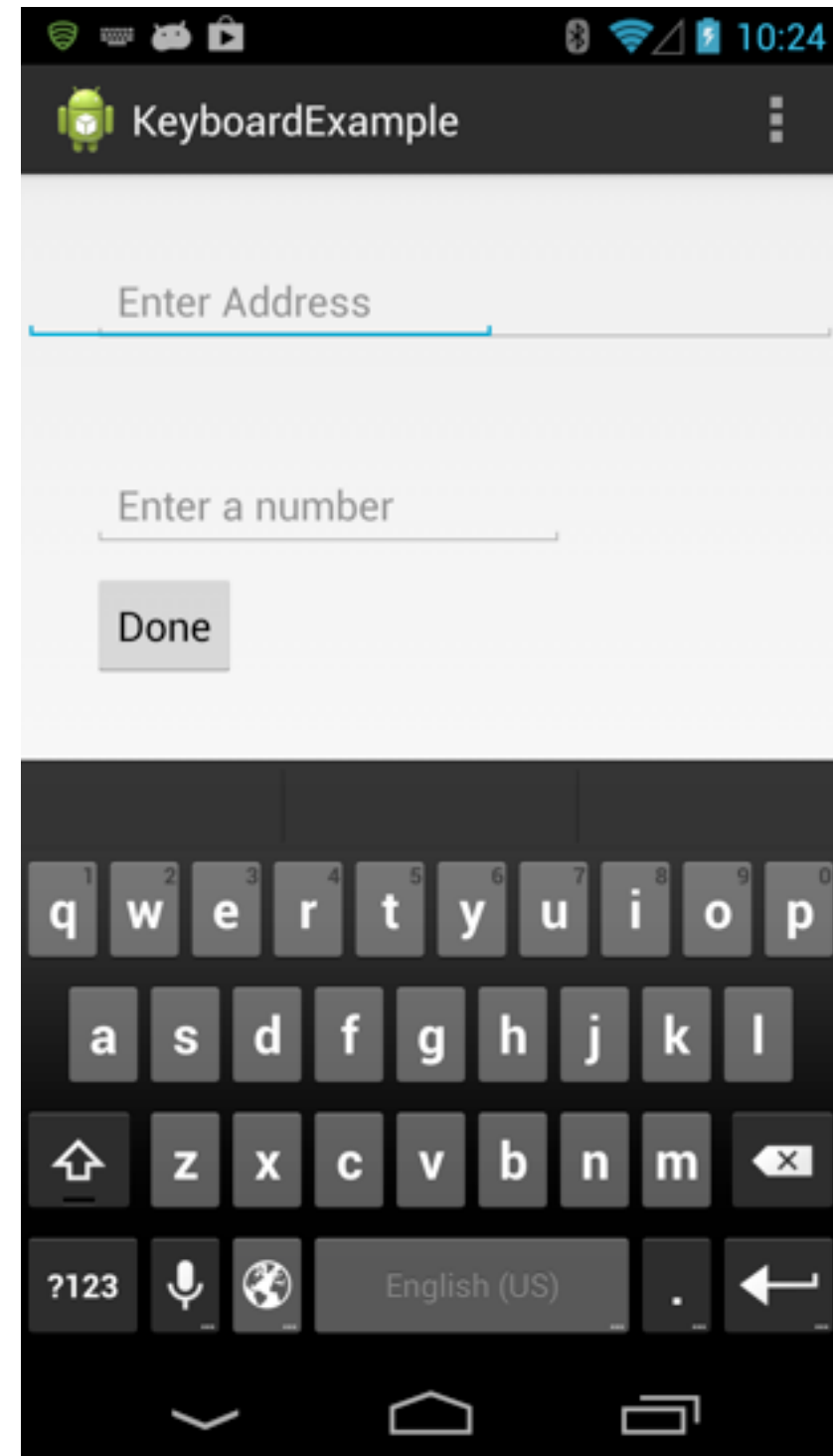
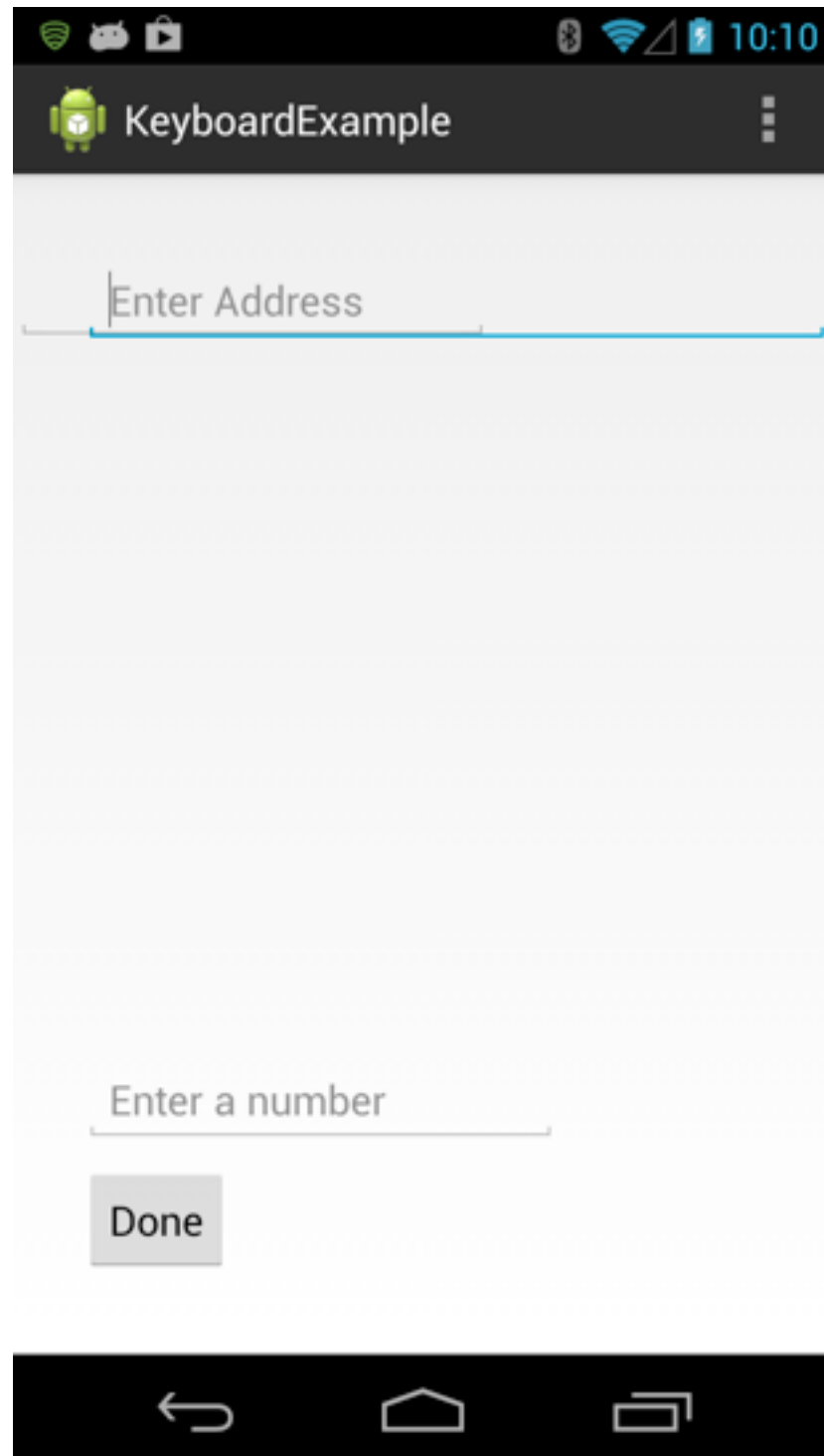
- Show keyboard

```
public void showSoftKeyboard(View view) {  
    if (view.requestFocus()) {  
        InputMethodManager imm = (InputMethodManager)  
            getSystemService(Context.INPUT_METHOD_SERVICE);  
        imm.showSoftInput(view, InputMethodManager.SHOW_IMPLICIT);  
    }  
}
```

Keyboard hides Controls



Re-adjusting View Size



Re-adjusting View Size

```
<activity
    android:name="edu.sdsu.cs.whitney.keyboardexample.MainActivity"
    android:label="@string/app_name"
    android:windowSoftInputMode="adjustResize">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
```

Panning the View

```
<activity
    android:name="edu.sdsu.cs.whitney.keyboardexample.MainActivity"
    android:label="@string/app_name"
    android:windowSoftInputMode="adjustPan|stateVisible">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
```