Brian Jones

Robert Jong

Alexey Kourganov

Bryant Barron

Group 3, Lab 5

2/13/2014

Comp 380

This program is intended to show how Activities and Intents work in the Android Operating System. A bulleted list of explanations will go over the basics of what this program does, and how it accomplishes those tasks.

1. First, we need to create the Edit Box that appears on the first Activity. We do this via the Activity’s Layout XML file (within an <EditText> caption).
2. Once the Edit Box control has been created, we assign it an ID. This allows us to create reference Handles to that specific object, giving us full-control over the Edit Box for later operations.
3. We also need to add a Send button to the first Activity, and we will do so by inserting a <Button> caption into the first Activity’s Layout XML file and giving the button an ID and on “android:onClick=”event\_name”” message handler (with a unique “event name”).
4. Next, we create a new Activity. This new Activity will contain a Static Text control object, which we will be using to display whatever was entered into the first Activity’s Edit Box control.
5. Create the Static Text control object for the second Activity by inserting a <TextView> caption into the second Activity’s Layout XML file.
6. Give the Static Text control object an ID, so that we can create reference Handles to that specific object, giving us full-control over the Static Text control for later operations.
7. Now, we need to override the event-handler for clicking on the Send button. We do this by creating a method with the same name as the event\_name from step #3.
8. We need to create an Intent so that we can link the Send button to the second Activity. We do this by creating an object of type Intent in the first method body from step #7, and instantiating the Intent with “this” for the first activity and the second Activity’s class-name).
9. We also need to create a Handle to the EditText control by instantiating an object of type EditText using findViewById(). This allows us to link the EditText control object with the EditText variable in the Java code, giving us an easy route to retrieve the string that was entered into it.
10. We can retrieve the EditText input string by creating a String and populating it using the EditText object’s .getText().toString() method.
11. Now, we need to add a tag to the input string and attach both to the Intent. We do this by using Intent.putExtra(“<unique name>”, <input string>).
12. Now, we need to fire off the Intent (since we are coding within the button’s onClick method, which means the user has clicked the Send button and we need to display the second Activity) and start the second Activity. This is accomplished by calling startActivity(Intent).
13. Now, in the second Activity, we need to override the onCreate() method, so that we can add our own code to be executed as soon as the second Activity is created.
14. Within the onCreate() method, we need to create an Intent and link it to the Intent from the first Activity. We do this by calling getIntent().
15. Now, we need to instantiate a new String with the input string retrieved from the Intent, and we do this by calling Intent.getStringExtra(“<unique name>”), and passing the return pointer to the new String.
16. After designating the Content View to the second Activity with setContentView(), we get a Handle to the TextView object by creating a TextView object and pointing it to the TextView object returned by Activity.findViewById(<Static Text Control ID>).
17. And finally, we simply call that TextView Handle’s .setText(“<input string>”) method to manually output text through the TextView control object.