CS 4985 – Fall 2013
Lab 2 Part 1
Due: TBA when Part 2 of this lab is posted.

Objectives

- Be able to create an options/activity bar menu.
- Be able to create a "subactivity" and return data from the subactivity to the invoking activity.

Specifications

This is the first part of a two-part lab exercise. The lab exercise will be submitted at the end of Part 2.

A. Getting Started

1. Create a new Android project in Eclipse.

2. Name the project *FirstnameLastName*Lab2.

    a. Fill in or verify the following properties:

        i. **Application Name** = Lab2: Options Menu by *Firstname Lastname*

        ii. **Project Name** = Lab2*FirstnameLastname*

        iii. **Package Name** = edu.uwg.*firstinitialLastname*.lab2

        iv. **Min SDK Version** = 10

    b. Have it create a Blank Activity:

        i. **ActivityName** = StudentListActivity

B. Display a list of students in the StudentListActivity class

1. Open the StudentListActivity class and have the class extend the ListActivity class.

2. Create a string ArrayList that holds the following values:

   John Doe, Fred Flintstone, Jane Doe, Bugs Bunny, Daffy Duck, Road Runner, Sallie Mae, Freddie Mac, Wilma Flintstone, Barney Rubble, Foghorn Leghorn, Tweety Bird, Porky Pig, Pepe Le Pew, Wil E. Coyote

   This data set will be used to hold a list of "students" that will be displayed as a scrollable list.

3. Create a new layout called students.xml.

   a. The root element for the layout should be a TextView.

      i. This will be the View that will be displayed in our ListActivity for each item that is displayed in the list. In this case, each View will just be a TextView that will contain a student's name.

   b. Set the id attribute to studentListView.

   c. Set the padding attribute to 10dp.

   d. Set the textSize attribute to 16sp.

4. In the onCreate method of the StudentListActivity class do the following:

   a. Delete the call to setContentView.

   b. Create a new ArrayAdapter that binds the R.layout.students to the students' data set.

   c. Add a call to setListAdapter passing it the array adapter that you created.

      The ArrayAdapter *binds each element of the data set to a single View object specified within the layout resource. The call to* setListAdapter *sets the adapter for the* ListActivity.

5. Compile and execute your code and the program should display a scrollable list of students from the student data set.

C. Modify the options menu that will allow a student to be added to the ListView

1. Delete the settings menu item and add a menu items that allows you to:

    a. Add a student
    b. Delete all students

2. Add code that will respond to the *Options* menu item being selected by adding the following to the StudentListActivity class:

    a. Override the onOptionsItemSelected method with the following code:

    Note: You will need to replace the id's below with the ids you defined for the menu items:

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {

    switch (item.getItemId()) {
    case R.id.addStudent: // Use the ID you defined
        Toast.makeText(getApplicationContext(), "Adding student",
              Toast.LENGTH_SHORT).show();
        return true;
    case R.id.deleteAllStudents: // Use the ID you defined
        Toast.makeText(getApplicationContext(), "Deleting all students",
              Toast.LENGTH_SHORT).show();
        return true;
    default:
        return super.onOptionsItemSelected(item);
    }
}
```

    For now, we are just displaying a toast of what item was invoked. In this situation it is OK to use a string literal for the toast, as the toasts are the equivalent of diagnostic print statements.

3. Compile and verify that the correct toast is displayed when each *Options* menu item is invoked.

D. Create a "sub" activity to prompt the user for a student's name

1. Create an activity that will prompt the user for the student's name. This activity will be similar to the GetUserNameActivity from the first lab exercise, except it will only need to display a label, an EditView to allow the user to enter the student's name, and an Add button. Complete the layout of this activity and the onCreate method on your own.

2. Add on onClickListener for the Add button. For this listener, you will **not** have it start a new activity. Instead, it needs to return to the activity that invoked it, in this instance, StudentListActivity. If you started a new activity, you would just add it to the activity stack. However, to act as a subactivity, we want it to return back to the invoking activity. In order, for the GetStudentNameActivity to return to the activity that invoked it we need to implement different behavior in the onClick method hander. In this method do the following:

    a. Declare and allocate an Intent object with the **default constructor**.

    b. Add a <*key*>, <*value*> pair to the Intent object that will store the student name entered.

        i. You will need to add code to get the student name that was entered.

    c. Add the following line of code which will set the return value and the data to return where *data* is your Intent object.

```
setResult(RESULT_OK, data);
finish();
```

3. The next step is to get the StudentListActivity class to invoke the GetStudentNameActivity as a subactivity by doing the following in the StudentListActivity class:

    a. Add a constant integer called ADD_STUDENT assigning it the value of 1.

    b. In the onOptionsItemSelected for the Add Student options menu item "handler" comment out the toast and add the following code:

```
Intent intent = new
   Intent(getApplicationContext(), GetStudentNameActivity.class);
   startActivityForResult(intent, ADD_STUDENT);
```

    When we startActivity *is used to start a new activity, it will be placed at the top of the activity stack. In this instance, we want* GetStudentNameActivity *to be a "subactivity" that will return a result when it ends. To do this, you call the* *startActivityForResult(Intent, int)* *version with a second integer parameter identifying the call. The result will come back through your* *onActivityResult(int, int, Intent)* *method.*

4. Add the onActivityResult callback method to the StudentListActivity class by adding the following code:

```
protected void onActivityResult(int requestCode, int resultCode,
                                Intent data){
    // See which subactivity is calling back.
    switch (requestCode) {
        case ADD_STUDENT:
            // Check what result was sent back from the activity
            if (resultCode == RESULT_OK) {
              String name = data.getStringExtra("name");
              Toast.makeText(getApplicationContext(),
                    "Adding student:" + name, Toast.LENGTH_LONG).show();
            }
        default:
            break;
    }
}
```

5. For now, we will just display a toast to verify the data from the subactivity was sent back to the invoking activity. Test the code to verify that it the GetStudentNameActivity returns the student name to the StudentListActivity.

E. Complete the options menu/active bar functionality

1. Modify the code so that the student that was entered is added to the list and then list view is updated.

   Note: After modifying the associated data set , you will need to figure out how to inform the adapter that the data set has changed.

2. Modify the code so that if Delete All was selected that the data set and list view have all the students deleted.

Submission

Name your zip file *FirstnameLastName*Lab3.zip, e.g, JohnDoeLab3.zip and store the file in Moodle. You will build upon this lab when the Part 2 specifications are posted. The due date for the both parts of the lab exercise will be posted with Part 2.