## CET3013: - Using Activities and Intents

**Week 3 Exercises: Using Activities and Intents**

###### Objectives – What this lesson is trying to achieve.

You are trying to learn

1. How to use Intents and Multiple Activities
2. The Android Manifest

This week we will be building a simple app with multiple layouts. This will allow us to practice using multiple intents, combined with trying out simple different Views.

IMPORTANT: Please try to complete Week 2 exercises first.

### Set the Baseline

This week’s class builds upon the work done so far. That is, you need to have a working version of week two’s exercises.

1. Download the sample solution for last week, if you did not complete the exercises to your satisfaction. It is better to use yours if you have one.
2. If you are going to use your solution, back up the working version by zipping the previous work. You can then go back to this version if need be.
3. If you are going to work with the sample version, unzip this to your project folder. When you start Studio, select ‘Import Project’ to load the file into the current system.
4. Start an Emulator, and delete any other student applications. You can do this on the device by selecting Settings, Apps. Student applications will usually be identifiable as the use the standard Android green man icon.
5. Run the app and check it works by seeing Log cat methods in the Android Monitor in Studio.

### Create the second activity and layout

Android Studio has a wizard that lets you add extra activities and layouts to your apps. It’s like a cut-down version of the wizard you use to create an app, and you use it whenever you want to create a new activity.

1. Create the second activity using file, new, Activity. Call the activity **quizActivity**, and the layout something similar (e.g. **activity\_quiz**).
2. In the second layout we need a textview, and a button labelled ‘Finish’.
3. As last week, add a general **onClick** method to the View class, by extending the **quizActivity** class.
4. Create a suitable Log to show this has been clicked  
   **Log.i(TAG, "Click Finish ")**

### Add the Second intent to the Android Manifest

Every Android app must include a file called AndroidManifest.xml. You can find it in the app/src/main folder of your project. The AndroidManifest.xml file contains essential information about your app.

The wizard should add the basics of the second activity to manifest.

1. Find the manifest.xml and check it includes the new activity

### Edit the second Layout

We will develop the layout for the second screen later after we have mastered activities. For now, we need a textView, and Button.

1. Use the layout designer to edit the **activity\_quiz** layout
2. Add a TextView called **question**
3. Add a button called **finish\_button**

### Test Launching the Intent

It is important to develop in small pieces. If you add large quantities of code to your project without testing it is hard to find bugs or logic flaws. This is especially true in Android development where the application is running at arm’s length.

In this task, we just want to check that we can launch the second intent.

1. In the **onClick** method for the **Accept\_Button** in the **MainActivity** find the second intent using **val intent = Intent(this, QuizActivity::class.java)**
2. Then call **startActivity(intent)**
3. Launch your application and test that the second Layout appears when the user has added their username and password, and clicks the **Accept\_Button**

### Returning from the Intent

Now that we have launched the second intent, we need to be able to return from it to the parent activity. This can be done by calling the **finish()**method.

1. In the layout for the second activity edit the **onClick** for the finish button you added
2. If the **View** clicked is the **finish\_button**, then call **finish();**
3. This should finish that intent and return to the first layout
4. Launch your application and test that the second is launched and returns

### Passing data to the Second Activity

It is often useful to pass data to other activities. Remember these are in separate areas of memory so you cannot simply access the object. It is possible though to pass strings, and integers.

1. In the **MainActivity**, before you start the 2nd activity (i.e. before the call to **startActivity(intent)**, call **intent.putExtra("user", uname)** where “user” is a label for the data you are passing, and **uname** would be the users name from the login form.
2. In the onCreate of the 2nd Activity we now need to get the string passed:  
    **val bundle = intent.extras?:return**

**Val username = bundle.getStringExtra("user")**

1. After you have the string value, check that it is not null, and set this in the question TextView, by saying **"Hello Again: " + user**
2. Test this in the Emulator.

### Returning results from the Second Activity

1. In the **MainActivity,** register the activity with the callback handler.

val startForResult = registerForActivityResult(  
 ActivityResultContracts.StartActivityForResult()) {  
result: ActivityResult ->  
 if (result.resultCode == Activity.RESULT\_OK) {

//TODO CODE:

}  
 }  
 }  
}

1. In the **MainActivity,** replace the call to **startActivity** with  
   **startForResult.launch(intent)**
2. In 2nd Activity if you want to send back data (say an int, with particular value of 180) to the **MainActivity** you need to place the following in the **finish\_button**

**val returnIntent = Intent()**

**returnIntent.putExtra("result",result)**

**setResult(Activity.RESULT\_OK,returnIntent)**

**super.finish()**

1. Now you have sent data back to the **MainActivity** this needs to be decoded.
2. Override the **TODO CODE:** in part 1. If so you can retrieve the “result” from the Intent bundle using **val data = result.data**

**data?.let {**

**if (it.hasExtra("user")) {**

**val returnString = it.extras?.getString("user")**

**textView1.text = returnString**

**}**

**}**

1. Finally, show that you have returned data by showing the result in the EditText welcome text.
2. Test this in the Emulator, adding **Log.i** calls to help find any issues.

### Programming Challenge

So far, you have return fixed data to the 1st activity. More challenging would be to allow the user to choose a value and return this. You could use an Edittext as we have done already, or experiment with another View, such as a spinner, or Radiobuttons (which belong in a radio group).