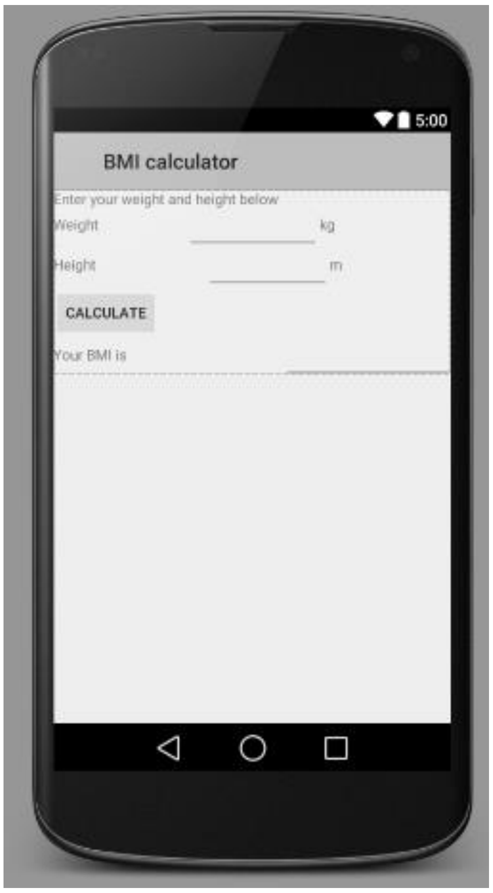
Lab 3: Building Interfaces & Adding Functionality to the App

BMI Calculator App

The interface from last week, repeated on the right, was created with xml file below. Start a new project and cut and paste the code below into the layout file over the default script.



```
<LinearLayout
  android:orientation="vertical"
  android:layout_width="fill_parent"
  android:layout_height="wrap_content"
xmlns:android="http://schemas.android.com/apk/res/android">

  <TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Enter your weight and height below"

    android:id="@+id/textView" />

  <LinearLayout
    android:orientation="horizontal"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content">

    <TextView
      android:layout_width="0dp"
      android:layout_height="wrap_content"
      android:layout_weight="30"
      android:text="Weight"
      android:id="@+id/textView2" />

    <EditText
      android:layout_width="0dp"
      android:layout_height="wrap_content"
      android:layout_weight="20"
      android:id="@+id/inWeight" />

    <TextView
      android:layout_width="0dp"
      android:layout_height="wrap_content"
      android:layout_weight="50"
      android:text="kg"
      android:id="@+id/textView3" />
  </LinearLayout>

  <LinearLayout
    android:orientation="horizontal"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
```

```
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="30"
            android:text="Height"
            android:id="@+id/textView4" />

        <EditText
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="20"
            android:id="@+id/inHeight" />

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="50"
            android:text="m"
            android:id="@+id/textView5" />
    </LinearLayout>

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Calculate"
        android:id="@+id/btnCalculate" />

    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent">

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Your BMI is"
            android:id="@+id/textView6" />

        <EditText
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:id="@+id/bmiResult" />
    </LinearLayout>

</LinearLayout>
```

Look at the Component Tree window to understand the structure. An overall vertical linear layout “contains” a textView, three horizontal linear layouts and a button within it. There are 4 components that we need to use to make the app work, the two EditText inputs for weight and height, the button to make the app work and the output EditText for the BMI result.

You should be able to identify the identifiers of these 4 components either from the Component tree or the XML code. Now let’s wire up the app so that it works. The first thing to do is get the button to work. This is the same procedure as last week – the code is given below so you should slot it into the right places.

```
implements View.OnClickListener
Button myButton;
myButton = (Button)findViewById(R.id.btnCalculate);
myButton.setOnClickListener(this);

public void onClick(View view) {
    Toast.makeText(this, "Button has been clicked.", Toast.LENGTH_LONG).show();
}
```

Import the various libraries required and make sure this works. This should give us confidence that the onClick() method is working and that we can now get it to do something useful. Now we want to handle the input from the EditText boxes. You should see in the emulator that you can enter values – and we want to deal with this when the button is pressed so any action must go in the onClick() method.

First we need to connect a Java declared EditText to the xml id of the appropriate component and we might as well make use of the toast to check we are doing this:

```
        EditText weightText = (EditText)findViewById(R.id.inWeight);
        Toast.makeText(this, weightText.getText(), Toast.LENGTH_LONG).show();
```

Insert these lines into the onClick() method, run the app, enter a number into the field and check that the button causes this to be displayed in the toast. weightText.getText() is the text property of the EditText. The EditText has a text value – this needs to be converted to a number to do the BMI calculation. To do this we declare a float value myWeight and use the following: `float myWeight = Float.parseFloat(weightText.getText().toString());`

Do the same to get a float value myHeight from the other EditText input. The formula to calculate BMI is weight(kg)/height(m)^2: `float bmi = myWeight/(myHeight*myHeight);`

The final step is to convert this float value to a string and set it as the text property of the output EditText bmiResult.

```
    EditText bmiResult = (EditText)findViewById(R.id.bmiResult);
    bmiResult.setText(String.valueOf(bmi));
```

Check that this all works. As a check for an 80Kg person 1.8 m tall the BMI is 24.7.

NOTE: If you do not have an entry in either of your editText fields the app will stop working. Can you think of a way of preventing this happening if there is no entry?

Appendix – List and Menus

The examples below show some other ways of getting input. To get an example running start simply cut and paste the code and it should work.

List

Create a new project with the following layout. This is an example where the MainActivity will extend ListActivity rather than Activity.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:orientation="vertical" >

  <TextView
    android:id="@+id/selection"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content" />

  <ListView
    android:id="@android:id/list"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:drawSelectorOnTop="false" />

</LinearLayout>
```

Notice that this does not render in in the preview – this is because the list needs to pick up some of its content from the MainActivity.java.

```
public class MainActivity extends ListActivity {
	private TextView selection;
	private static final String[] items = { "FROOME Christopher", "VAN GARDEREN Tejay", "URAN URAN Rigoberto", "ESP - RODRIGUEZ Joaquim", "GESINK Robert", "POZZOVIVO Domenico", "HENAO MONTOYA Sergio", "KREUZIGER Roman", "TSCHOPP Johann", "MORENO FERNANDEZ Daniel", "JEANNESSON Arnold", "NIBALI Vincenzo", "BARDET Romain", "IMPEY Daryl", "LÖVKVIST Thomas", "VAN DEN BROECK Jurgen", "KONIG Leopold", "SANTAROMITA Ivan", "SCHLECK Frank", "NIEVE ITURRALDE Mikel"};

@Override
public void onCreate(Bundle icicle) {
	super.onCreate(icicle);
	setContentView(R.layout.activity_main);
	setListAdapter(new ArrayAdapter<String>(this, android.R.layout.simple_list_item_1, items));
		selection = (TextView) findViewById(R.id.selection);
	}

	public void onListItemClick(ListView parent, View v, int position, long id) {
		Toast.makeText(this,items[position],	Toast.LENGTH_LONG).show();
		selection.setText(items[position]);
	}
}
```

This gives you the capacity to make selections from a list. Note the use of toast just to make sure it is working. The main activity in this app is a bit different from the previous projects in that the activity extended in this case is a ListActivity, which is a subclass of Activity. The ListActivity creates a list which each item can be made active – through the onListItemClick method.

Menu

Create a new project. For the menu to work you need the header or action bar provided by your MainActivity extending AppCompatActivity (not simple Activity). The default project has the onCreate() method to initialise the layout. Two other methods need to be added, one to populate the menu and the other to give it functionality.

The menu is in the action bar at the top of the app. The onCreateOptionsMenu() method populates the menu. The method onOptionsItemSelected() is then used to take appropriate action when a menu item is selected. There are other ways but this is simple as the example shows.

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
        menu.add(0, 1, 0, “cereal);         // Group, Item ID, Order, Title
        menu.add(0, 2, 1, "muesli");
        menu.add(1, 3, 2, "bacon");
        menu.add(1, 4, 1, "eggs");
        menu.add(1, 5, 2, "toast");
        menu.add(2, 6, 3, "porridge");
        menu.add(2, 7, 4, "kipper");
        return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
                case (1): {
                        Toast.makeText(this, item.getTitle(), Toast.LENGTH_LONG).show();
                        return true;
                }
                case (2): {
                        Toast.makeText(this, item.getTitle(), Toast.LENGTH_LONG).show();
                        return true;
                }

                // and so on
        }
        return super.onOptionsItemSelected(item);
}
```