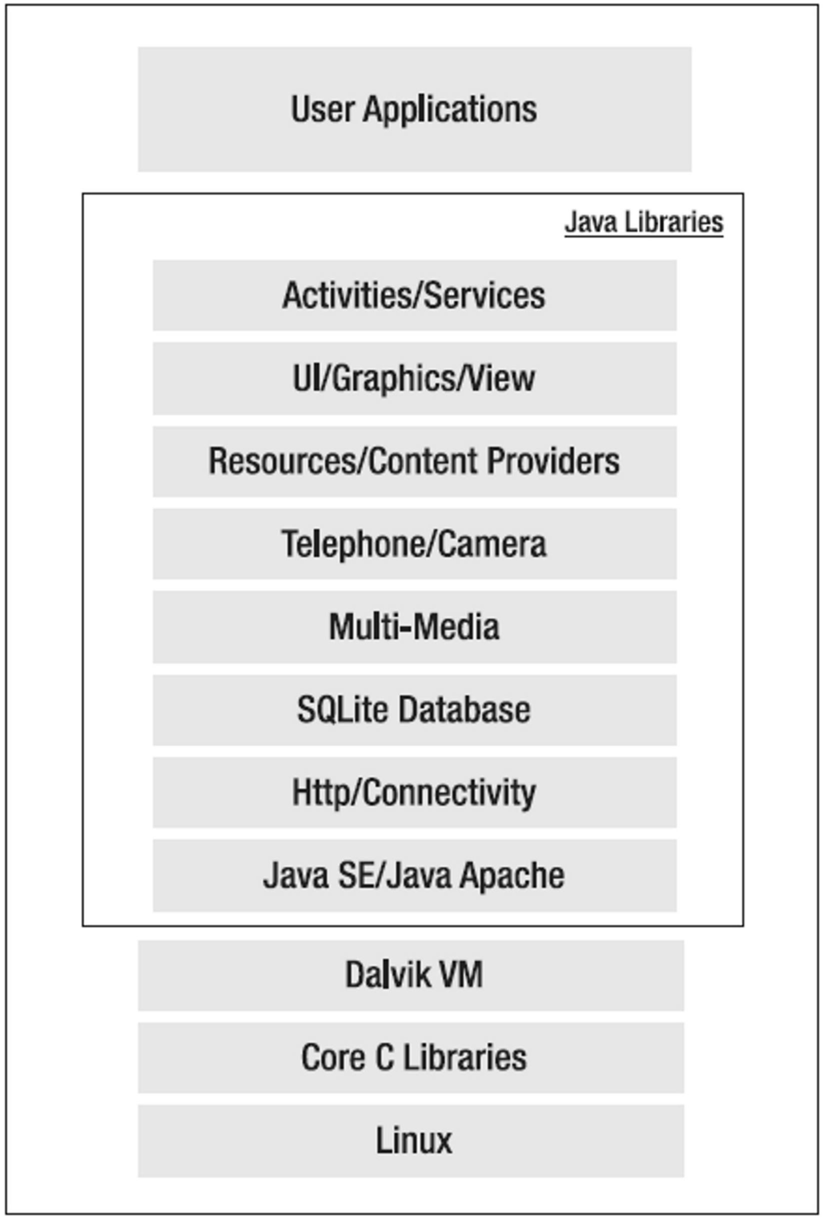
Mobile and Web Computing

Introduction to Android Development

Android Software Stack



Android Emulator

- Android SDK includes an Eclipse plug-in called Android Development Tools (ADT).
  - Developing
  - Debugging
  - Testing
- Android SDK can be used without ADT
  - command-line tools are available
- Android Emulator
  - Works with both ADT and Command line tool
  - Emulates 90% of features in real devices
- Emulator limitations:
  - USB connections
  - camera and video capture
  - Headphones
  - battery simulation
  - Bluetooth, WiFi and NFC
  - OpenGL ES 2.0.

Android User Interface

- Essentially a fourth-generation UI framework
  - **First generation:** traditional C-based Microsoft Windows API
  - **Second generation:** C++-based MFC (Microsoft Foundation Classes).
  - **Third generation:** Java-based Swing UI framework
    - introducing design flexibility far beyond that offered by MFC.
  - **Fourth-generation:** The Android UI, JavaFX, Microsoft Silverlight, and Mozilla XML User Interface Language (XUL)
    - The UI is declarative and independently themed.

Fundamental Components

- **View**
  - The basic building blocks of a user interface.
  - Ex: button, label, text field etc.
  - Views are also used as containers for views→ hierarchy of views
  - Everything you see is a view
  - Subclasses:-
    - AnalogClock, ImageView, KeyboardView, MediaRouteButton, ProgressBar, Space, SurfaceView, TextView, TextureView, ViewGroup, ViewStub

Fundamental Components

- **Activity**
  - A user interface concept.
  - Usually represents a single screen in application.
  - May contains one or more views
  - Gives ability to viewing data, creating data, or editing data.
  - Most Android applications have several activities.

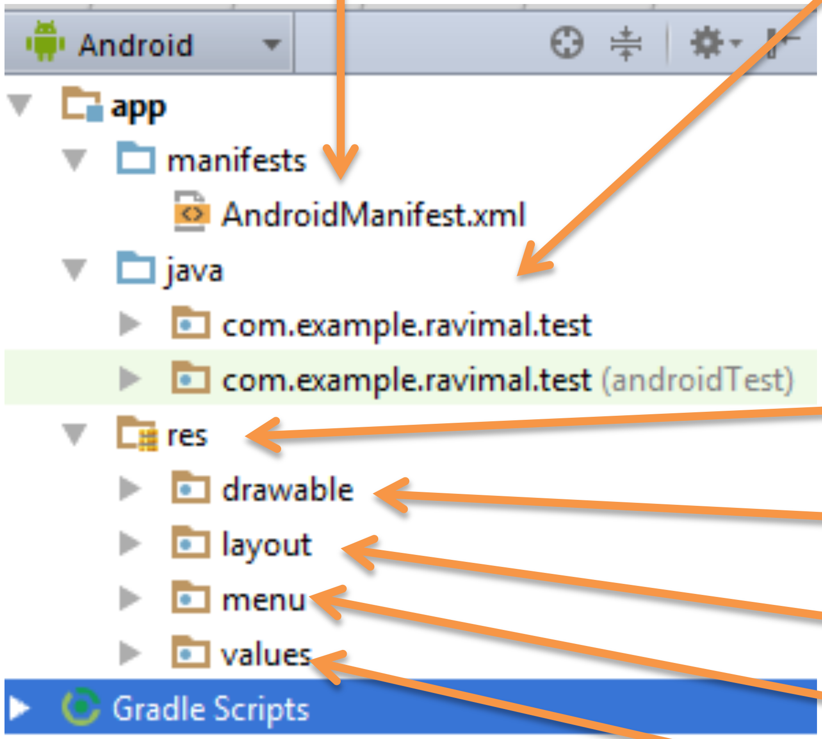
Fundamental Components

- **Intent**
  - Generically defines an “intention” to do some work.
  - encapsulate several concepts
  - examples
    - Broadcast a message.
    - Start a service.
    - Launch an activity.
    - Display a web page or a list of contacts.
    - Dial a phone number or answer a phone call.
  - Not always initiated by third party application
    - also used by the system to notify your application of specific events (such as the arrival of a text message).

Structure of the Project Directory

information the system must have before it can run any of the app's code

The application code that includes the logic



```
app
  manifests
    AndroidManifest.xml
  java
    com.example.ravimal.test
    com.example.ravimal.test (androidTest)
  res
    drawable
    layout
    menu
    values
Gradle Scripts
```

Resources

Graphical objects and related information

Layout definition files

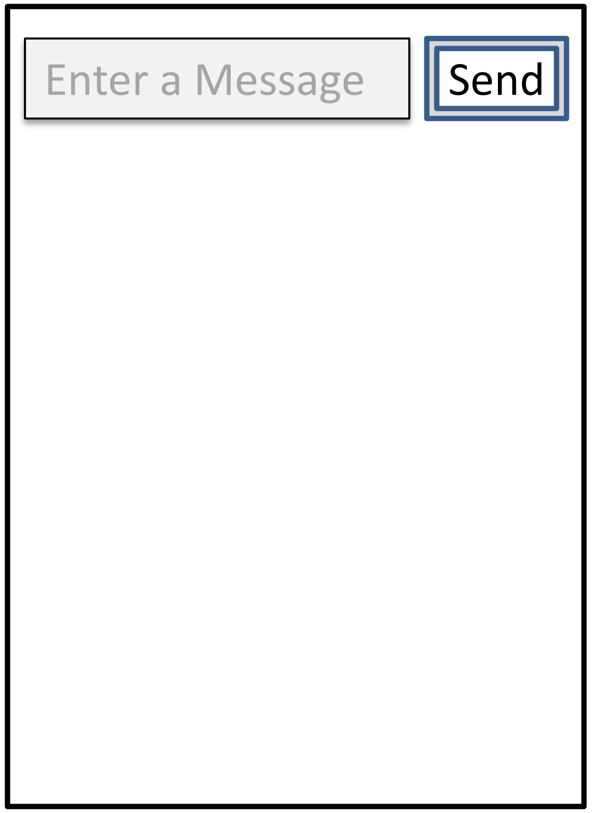
Menu definition files

User-defined values

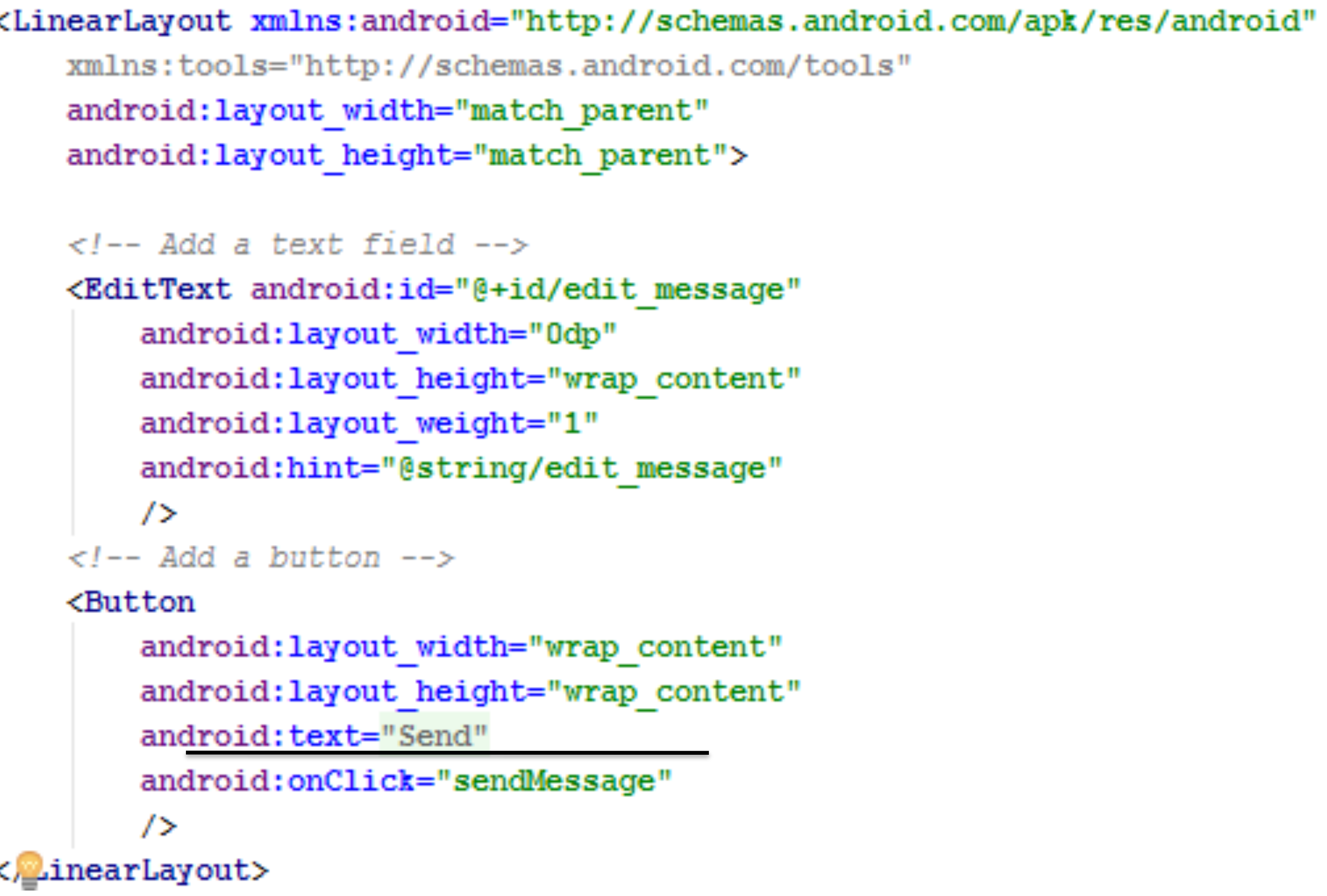
Includes build instruction

Hands on the First Application

- Lets Create a Simple UI.
  - Create a blank activity
  - Edit the layout file to create the following layout

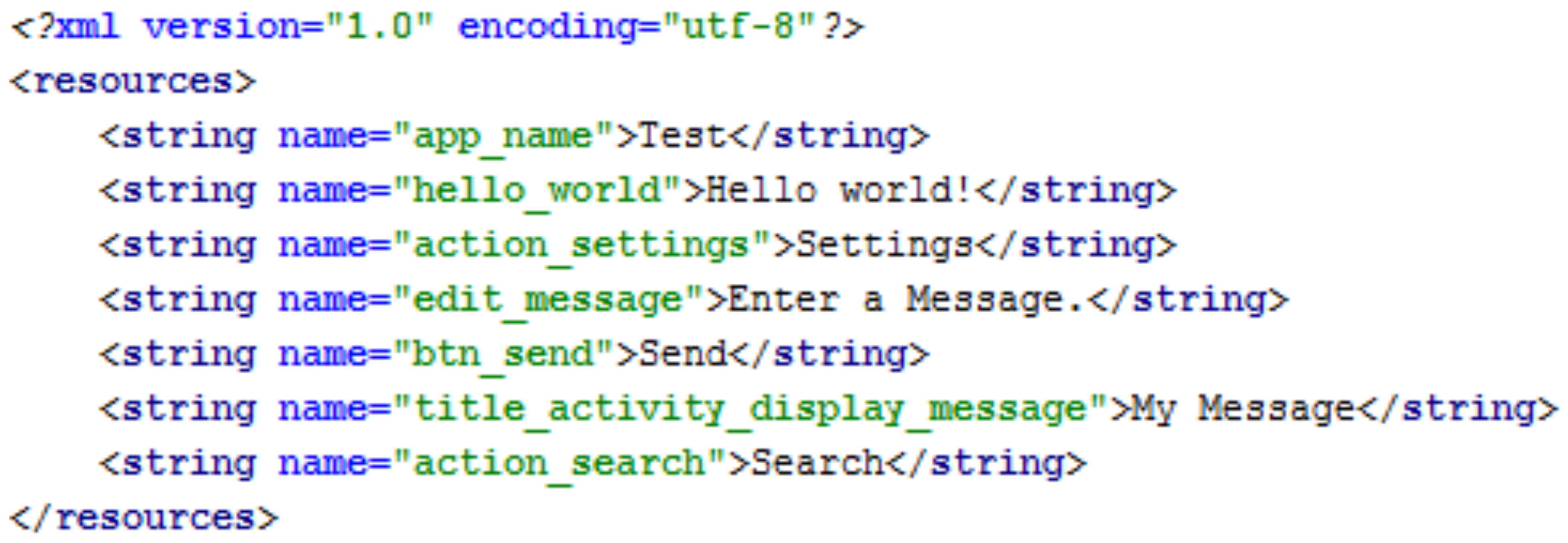


Layout XML



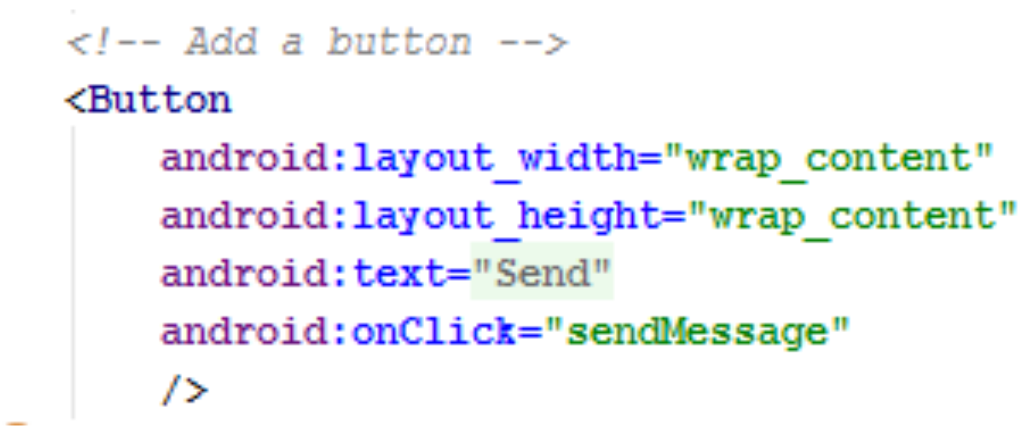
```xml
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <!-- Add a text field -->
    <EditText android:id="@+id/edit_message"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:hint="@string/edit_message"
        />
    <!-- Add a button -->
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Send"
        android:onClick="sendMessage"
        />
</LinearLayout>
```

String Resources



```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">Test</string>
    <string name="hello_world">Hello world!</string>
    <string name="action_settings">Settings</string>
    <string name="edit_message">Enter a Message.</string>
    <string name="btn_send">Send</string>
    <string name="title_activity_display_message">My Message</string>
    <string name="action_search">Search</string>
</resources>
```

Handling Events



```xml
<!-- Add a button -->
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Send"
    android:onClick="sendMessage"
    />
```

```java
public void sendMessage(View view){
                // Do Something
}
```

Open an Activity

```
public void sendMessage(View view){
        Intent intent = new Intent(this,DisplayMessageActivity.class);
        EditText editTet = (EditText)findViewById(R.id.edit_message);
        String message = editTet.getText().toString();
        intent.putExtra(EXTRA_MESSAGE,message);
        startActivity(intent);
    }
```

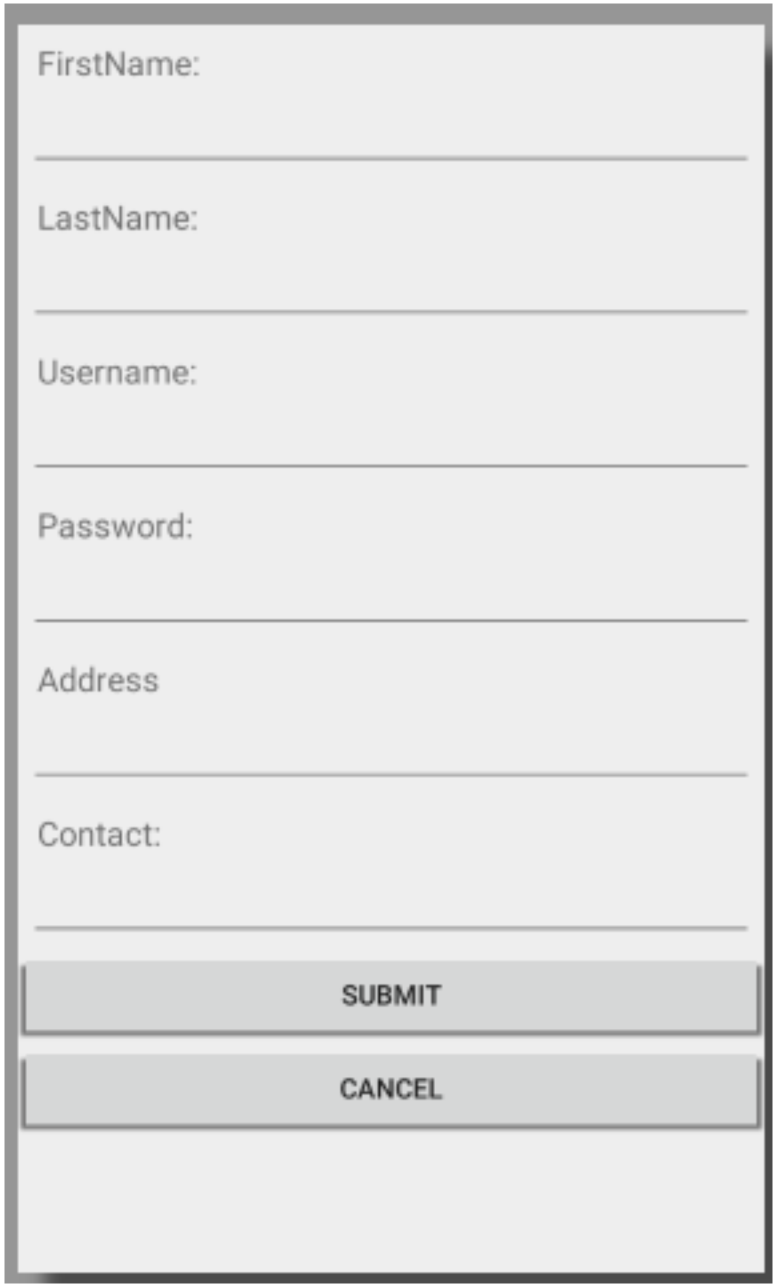
Send Message to an Activity

```
public void sendMessage(View view){
        Intent intent = new Intent(this,DisplayMessageActivity.class);

        EditText editTet = (EditText)findViewById(R.id.edit_message);
        String message = editTet.getText().toString();
        intent.putExtra(EXTRA_MESSAGE,message);

        startActivity(intent);
    }
```

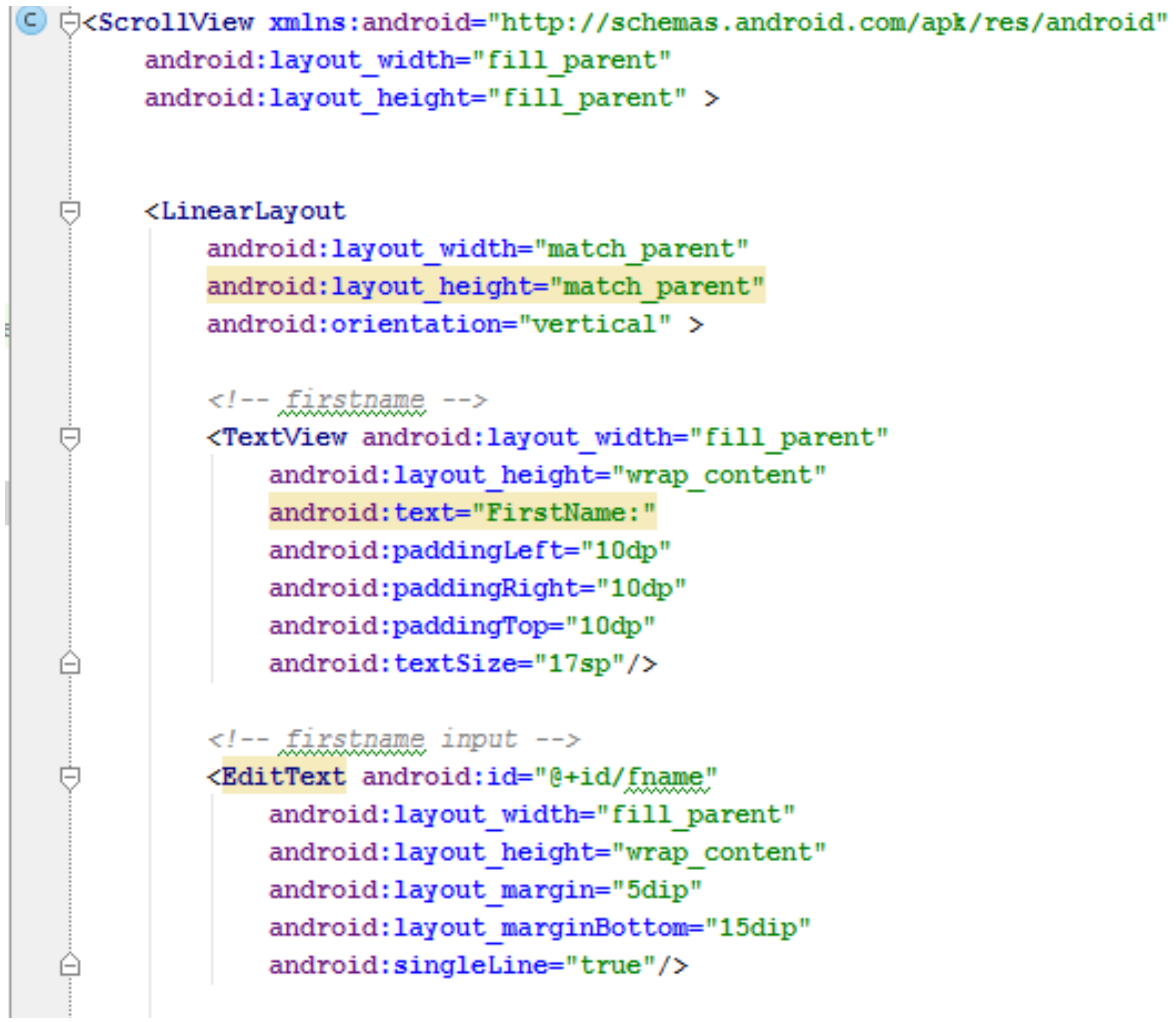
Extract Messages from an Intent

```
Intent intent = getIntent();
        String message = intent.getStringExtra(MainActivity.EXTRA_MESSAGE);
        TextView tw = (TextView)findViewById(R.id.txt);
        tw.setText(message);
```

Vertical LinearLayout



Vertical LinearLayout



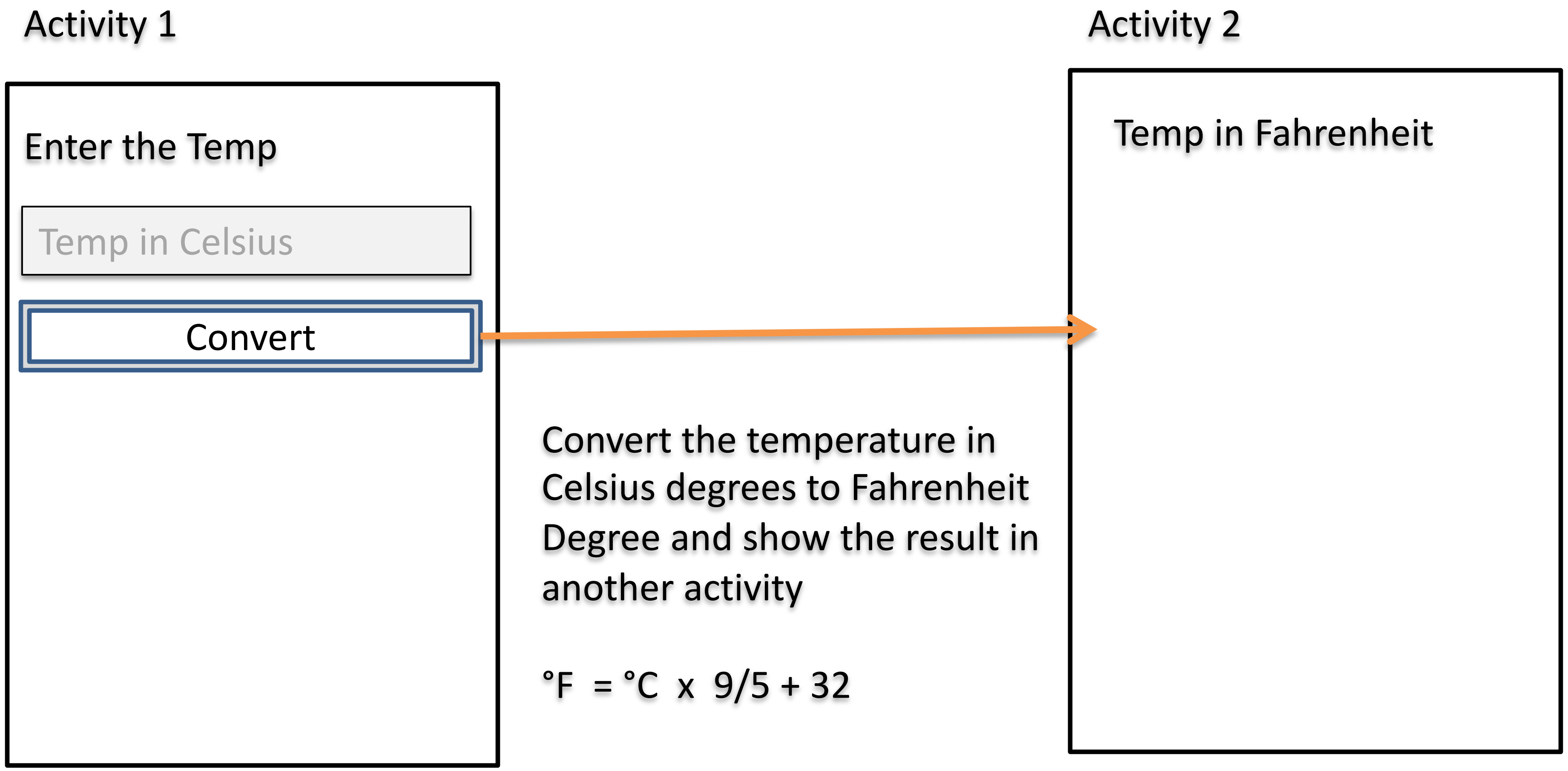
```
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >


    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical" >

        <!-- firstname -->
        <TextView android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="FirstName:"
            android:paddingLeft="10dp"
            android:paddingRight="10dp"
            android:paddingTop="10dp"
            android:textSize="17sp"/>

        <!-- firstname input -->
        <EditText android:id="@+id/fname"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_margin="5dip"
            android:layout_marginBottom="15dip"
            android:singleLine="true"/>
```

Dimension Values in Android

- dp
  - Density-independent Pixels - Based on the physical density of the screen. Relative to a 160 dpi (dots per inch) screen, on which 1dp is roughly equal to 1px. When running on a higher density screen, the number of pixels used to draw 1dp is scaled up by a factor appropriate for the screen's dpi.
- sp
  - Scale-independent Pixels - This is like the dp unit, but it is also scaled by the user's font size preference. It is recommend you use this unit when specifying font sizes, so they will be adjusted for both the screen density and the user's preference.
- pt
  - Points - 1/72 of an inch based on the physical size of the screen.
- px
  - Pixels - Corresponds to actual pixels on the screen. This unit of measure is not recommended because the actual representation can vary across devices;
- mm
  - Millimeters - Based on the physical size of the screen.
- In
  - Inches - Based on the physical size of the screen.

Exercise 01

- Create the following layout and implement the relevant functionality of the button “Convert”



Convert the temperature in Celsius degrees to Fahrenheit Degree and show the result in another activity

°F = °C x 9/5 + 32

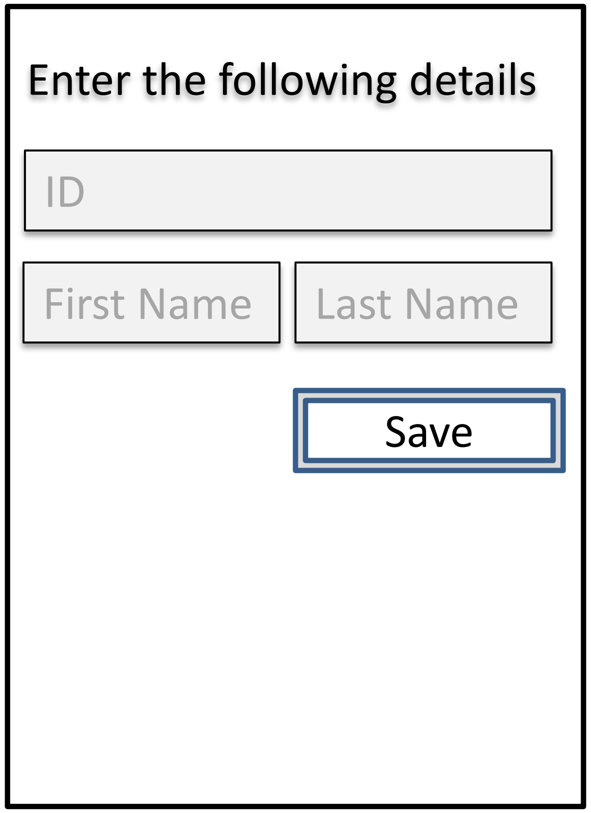
RelativeLayout

- RelativeLayout lets child views specify their position relative to the parent view or to each other (specified by ID).
- By default, all child views are drawn at the top-left of the layout.
  - The position of each view must be defined using the various layout properties available from RelativeLayout.LayoutParams

- android:layout_alignParentTop
  - If "true", makes the top edge of this view match the top edge of the parent.
- android:layout_centerVertical
  - If "true", centers this child vertically within its parent.
- android:layout_below
  - Positions the top edge of this view below the view specified with a resource ID.
- android:layout_toRightOf
  - Positions the left edge of this view to the right of the view specified with a resource ID.

Exercise 02

- Create the following layout

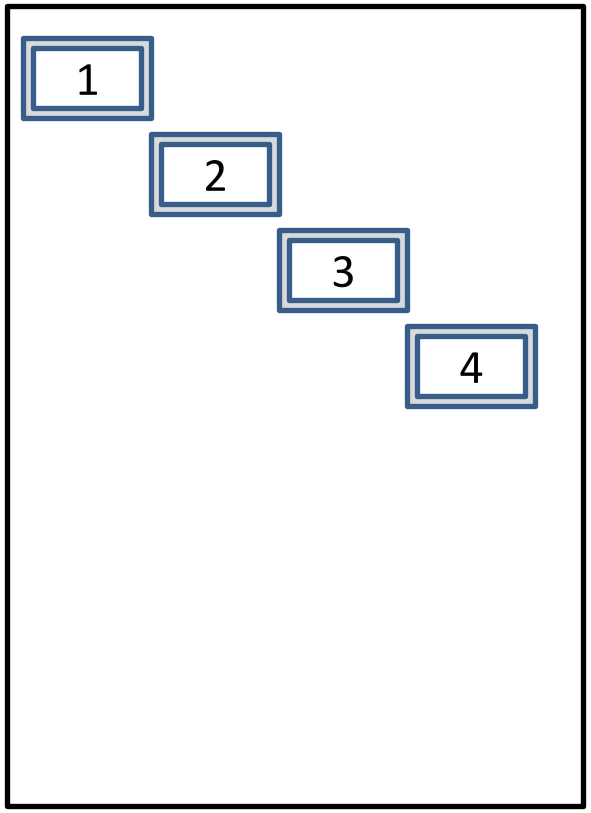


GridLayout

- A layout that places its children in a rectangular *grid*.
- Specifies the number of cells by the attributes
  - android:columnCount="5"
  - android:rowCount="5"
- Position the children in a particular cell by using,
  - android:layout_row="0"
  - android:layout_column="0"
- Children may span over several cells
  - android:layout_columnSpan="2"
  - android:layout_rowSpan="2"

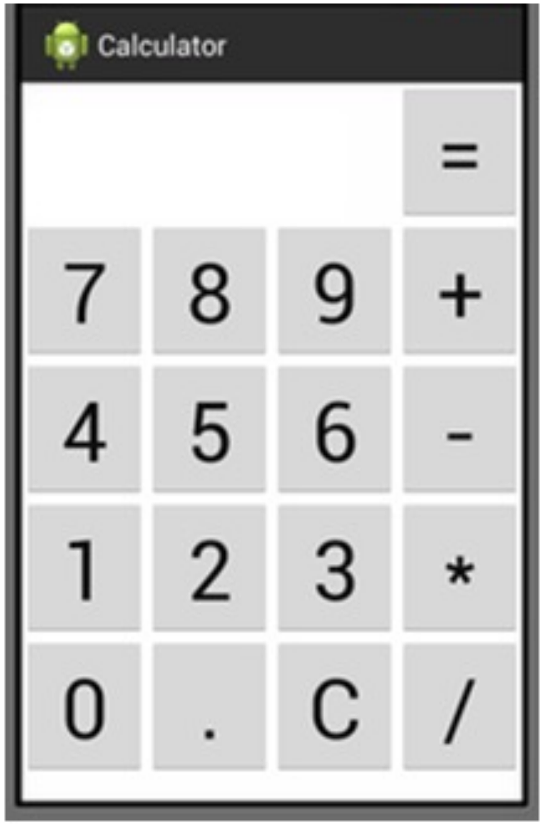
Exercise 03

- Create the following layout



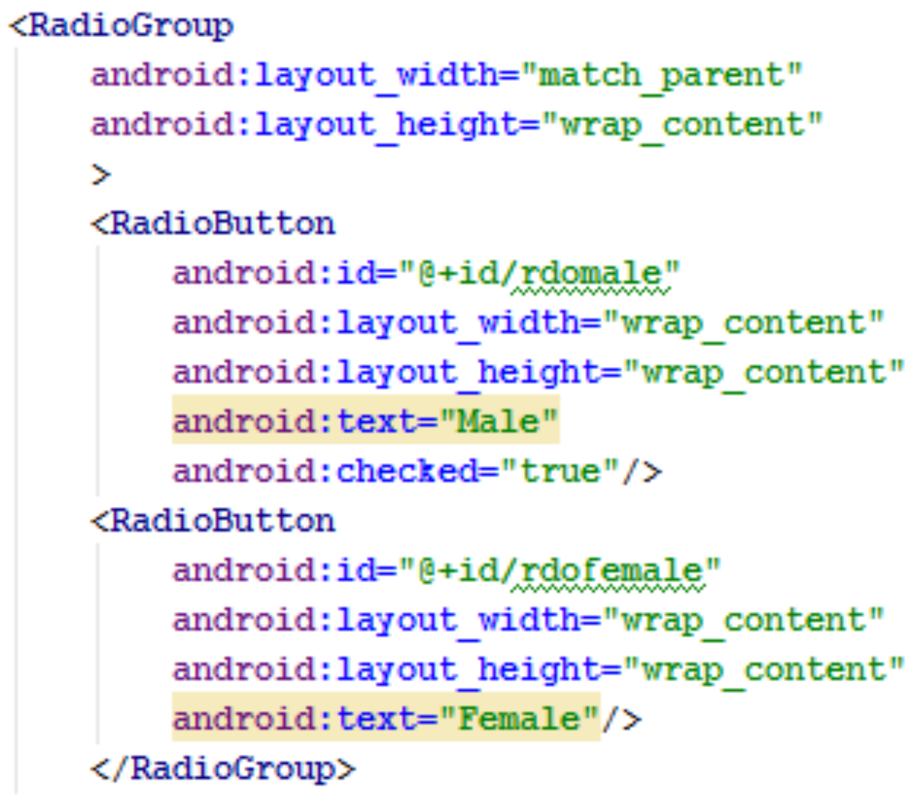
Exercise 04

- Create the following UI using suitable layouts.



RadioButtons

Layout XML File

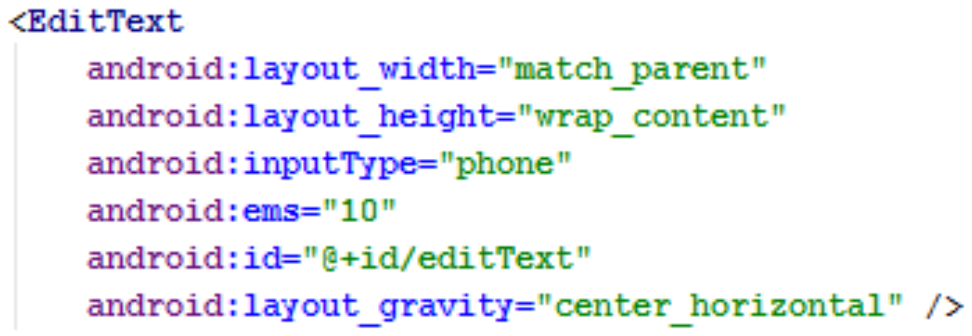


```
<RadioGroup
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    >
    <RadioButton
        android:id="@+id/rdomale"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Male"
        android:checked="true"/>
    <RadioButton
        android:id="@+id/rdofemale"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Female"/>
    </RadioGroup>
```

Java Code

```
//Check the state of the radio buttons
RadioButton rdoMale = (RadioButton)findViewById(R.id.rdomale);
 if(rdoMale.isChecked()){
     //Do Something
 }
```

Input Types of Text Fields



```
<EditText
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:inputType="phone"
    android:ems="10"
    android:id="@+id/editText"
    android:layout_gravity="center_horizontal" />
```

- Other Input Types:
  - Name
  - textPassword
  - textEmailAddress
  - textPostalAddress
  - textMultiLine
  - time
  - date
  - number

Exercise 05

- Implement a fully functional calculator with the layout you have created in the exercise 04.
    - Assign different background colors to the buttons (i.e. Three separate colors for numbers, operators and function keys)