Mobile Application Development

Unit- 4

Designing User interface with View

Mrs. Chavan P.P.

ANDROID UI CONTROLS

There are number of UI controls provided by Android that allow you to build the graphical user interface for your app.

Following are the commonly used UI or input controls in android applications.

TextView	RadioGroup
EditText	ProgressBar
AutoCompleteTextView	TimePicker
Button	DatePicker
ImageButton	SeekBar
ToggleButton	AlertDialog
CheckBox	Switch
RadioButton	RatingBar

ANDROID UI CONTROLS

The “views” are the building blocks of a U.I design and composes of almost every basic U.I element like TextViews, EditTexts, ImageViews etc. This ‘***view***’ however comes along with a few properties bundled to them. Some of the important and are often used to build up a complete meaningful screen design.

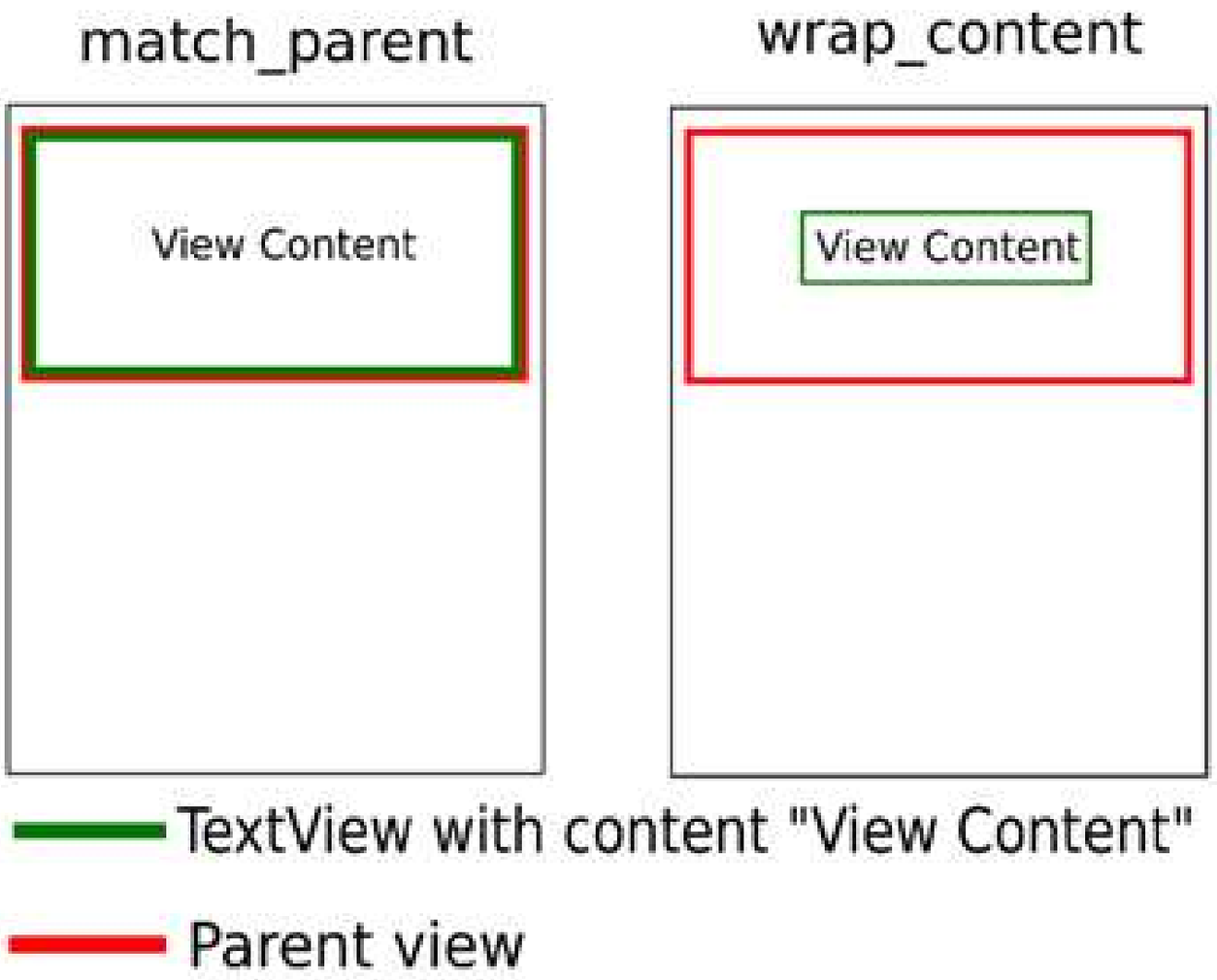
“id”

This is basically the name of the particular view and will be used to refer that particular view through out the project. It has to be unique(*multiple views referencing to same id will confuse the compiler*).

“width” and “height”

As the name of these properties suggest, these are the dimensions of that particular view. Now these dimensions can be set using hard-coded values and it will adopt to them in most layouts, but its not a very good design as the content inside them might get cropped or will have unwanted spaces. Android provides two pre-defined options to set these dimensions — ***“match_parent”*** and ***“wrap_content”.***

ANDROID UI CONTROLS

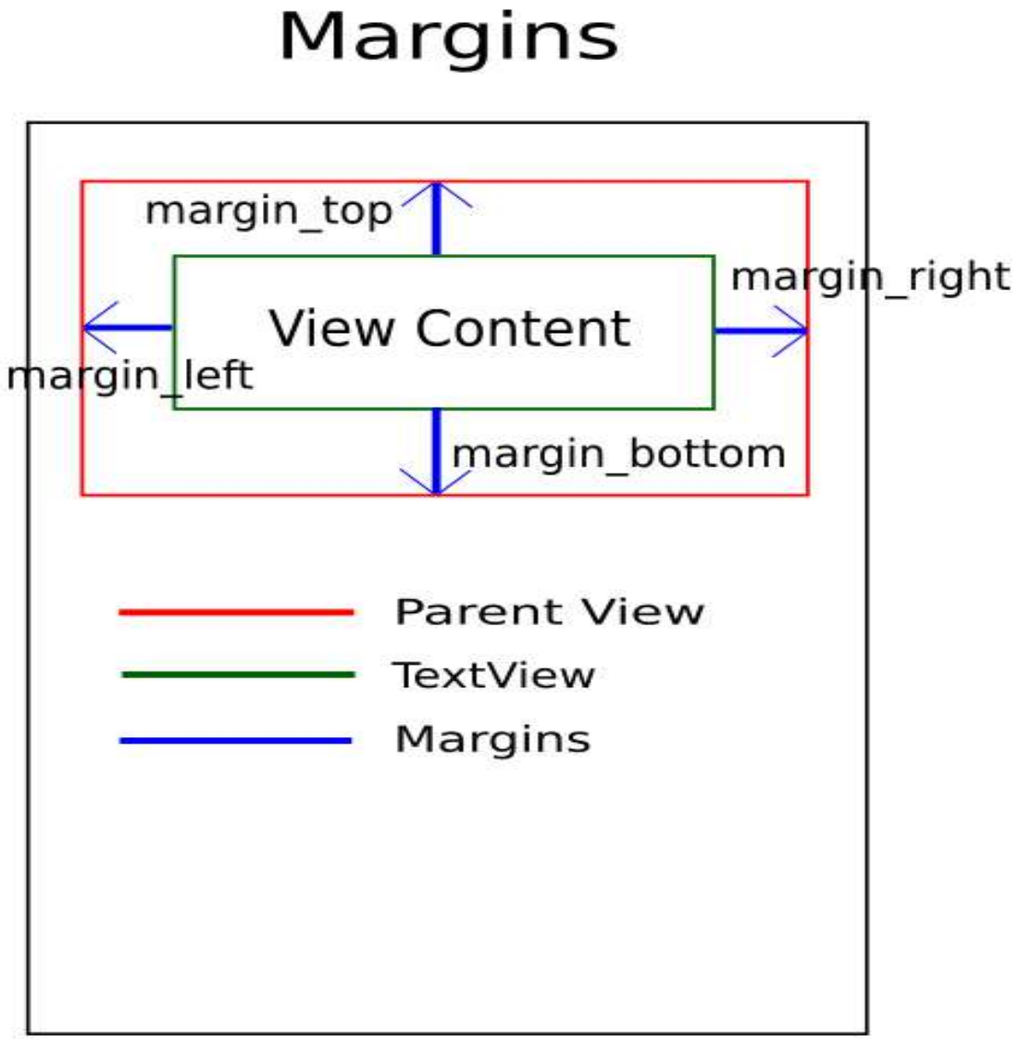


Setting the dimensions to “match_parent” will make them equal to those of its parent’s dimensions. If there is no parent to that particular view, it merely sets to the screen’s dimensions (parent of a view is the U.I element in which it is housed in). And setting it to “wrap_content” will force the view to adopt its dimensions according to its content.

ANDROID UI CONTROLS

"margin"

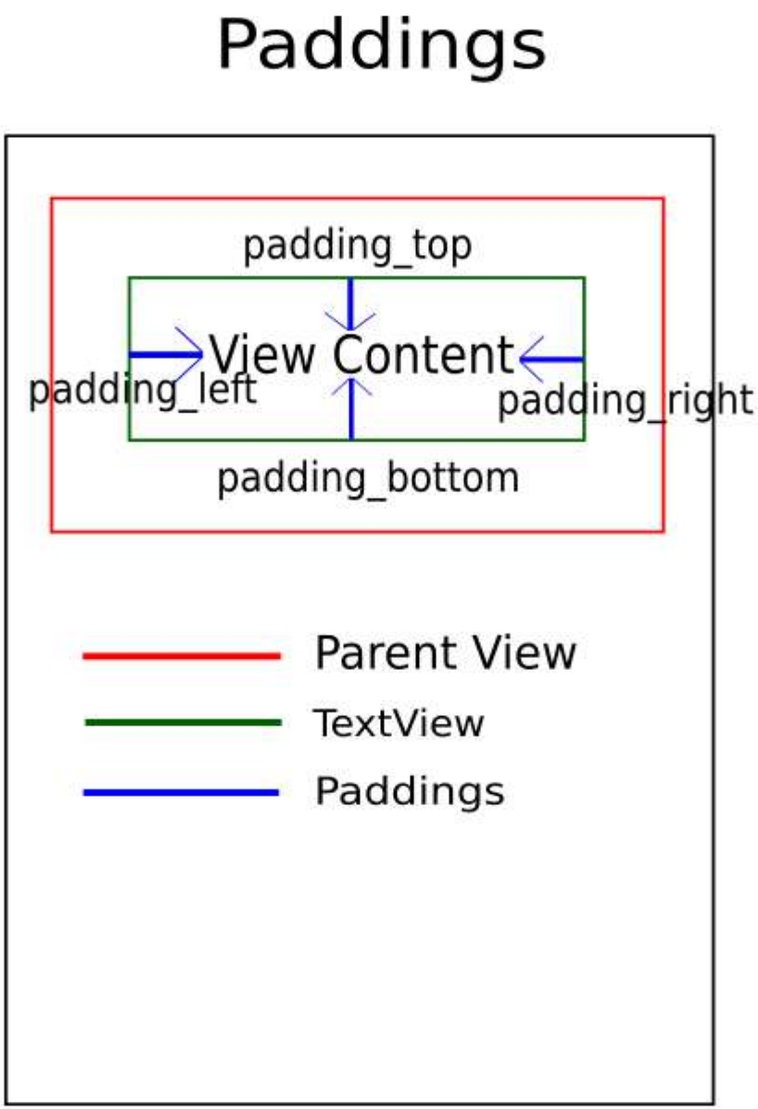
This is the minimum distance that a view has to maintain from its neighbouring views. Since there are four sides to any view, the four margins corresponding to them are "margin_left", "margin_right", "margin_top" and "margin_bottom". If the same margin is needed to be set on all sides, it can be set directly through "margin" property.



ANDROID UI CONTROLS

“padding”

The distance between the view’s outline and its content. Again similar to the “margin” property, “padding” too has “padding_left”, “padding_right”, “padding_top”, “padding_bottom” and the common padding to all sides can be set through “padding” property.



ANDROID UI CONTROLS

1. TextView:

A **TextView** displays text to the user and optionally allows them to edit it. A TextView is a complete text editor, however the basic class is configured to not allow editing.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <TextView
        android:id="@+id/textView1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginBottom="10dp"
        android:text="Welcome to Mobile App Development"
        android:textColor="#86AD33"
        android:textSize="20dp"
        android:textStyle="bold" />
</LinearLayout>
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.andr
oid.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_marginTop="10dp"
    android:orientation="vertical"
    android:padding="10dp">
    <TextView
        android:id="@+id/textView1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginBottom="10dp"
        android:text="Mobile App Development"
        android:textColor="#86AD33"
        android:textSize="20dp"
        android:textStyle="bold" />
    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="15dp"
        android:textAllCaps="true" />
    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text=" Mobile App Development"
        android:textStyle="bold"
        android:textColor="#fff"
        android:background="#7F3AB5"
        android:layout_marginBottom="15dp"/>
    <TextView
        android:id="@+id/textView4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:autoLink="email|web"
        android:text="Contact Admin" />
</LinearLayout>
```

MainActivity.java

```
package com.mmp.textviewexample;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        TextView tv =findViewById(R.id.textView2);
        tv.setText("Welcome to MMP");
    }
}
```

ANDROID UI CONTROLS

2. EditText:

EditText is a predefined subclass of TextView that includes rich editing capabilities. It is used to allow the user to enter or modify the text. While using **EditText** control in our android applications, we need to specify the type of data the text field can accept using **inputType** attribute.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical" >
   <EditText
      android:id="@+id/txtSub"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:hint="Subject"
      android:inputType="text"/>
</LinearLayout>
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://sche
mas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="40dp"
    android:orientation="vertical" android:id=
"@+id/linearlayout" >
    <EditText
        android:id="@+id/txtName"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="25dp"
        android:ems="10"
        android:hint="Name"
        android:inputType="text"
        android:selectAllOnFocus="true" />
    <EditText
        android:id="@+id/txtPwd"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:ems="10"
        android:hint="Password 0 to 9"
        android:inputType="numberPassword
" />
    <EditText
        android:id="@+id/txtEmai"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:ems="10"
        android:hint="Email"
        android:inputType="textEmailAddress
" />
    <EditText
        android:id="@+id/txtDate"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/editText3"
        android:ems="10"
        android:hint="Date"
        android:inputType="date" />
    <EditText
        android:id="@+id/txtPhone"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:ems="10"
        android:hint="Phone Number"
        android:inputType="phone"
        android:textColorHint="#FE8DAB"/>
    <Button
        android:id="@+id/btnSend"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="submit"
        android:textSize="16sp"
        android:textStyle="normal|bold" />
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/resultView"
        android:layout_marginTop="25dp"
        android:textSize="15dp"/>
</LinearLayout>
```

MainActivity.java

```
package com.mmp.edittextexample;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import org.w3c.dom.Text;

public class MainActivity extends AppCompatActivity {
    Button btnSubmit;
    EditText name, password, email, dob, phoneno;
    TextView result;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        name=findViewById(R.id.txtName);
        password =findViewById(R.id.txtPwd);
        email = findViewById(R.id.txtEmai);
        dob = findViewById(R.id.txtDate);
        phoneno=findViewById(R.id.txtPhone);
        btnSubmit = findViewById(R.id.btnSend);
        result = findViewById(R.id.resultView);
        btnSubmit.setOnClickListener(new View.OnClickListener() {
```

```
@Override
        public void onClick(View v) {
            if (name.getText().toString().isEmpty() || password.getText().toString().isEmpty() |
| email.getText().toString().isEmpty() || dob.getText().toString().isEmpty()
                    || phoneno.getText().toString().isEmpty()) {
                result.setText("Please Fill All the Details");
            } else {
                result.setText("Name -  " + name.getText().toString() + " \n" + "Password -
 " + password.getText().toString()
                        + " \n" + "E-Mail -  " + email.getText().toString() + " \n" + "DOB -  " + dob.g
etText().toString()
                        + " \n" + "Contact -  " + phoneno.getText().toString());
            }
        }
    });
  }
}
```

ANDROID UI CONTROLS

3. Button:

Button is a user interface control which is used to perform an action when the user click or tap on it.

Create Button in XML Layout File

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:orientation="vertical" android:layout_width="match_parent"
   android:layout_height="match_parent">
   <Button
      android:id="@+id/addBtn"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Add" />
</LinearLayout>
```

activity_main.xml

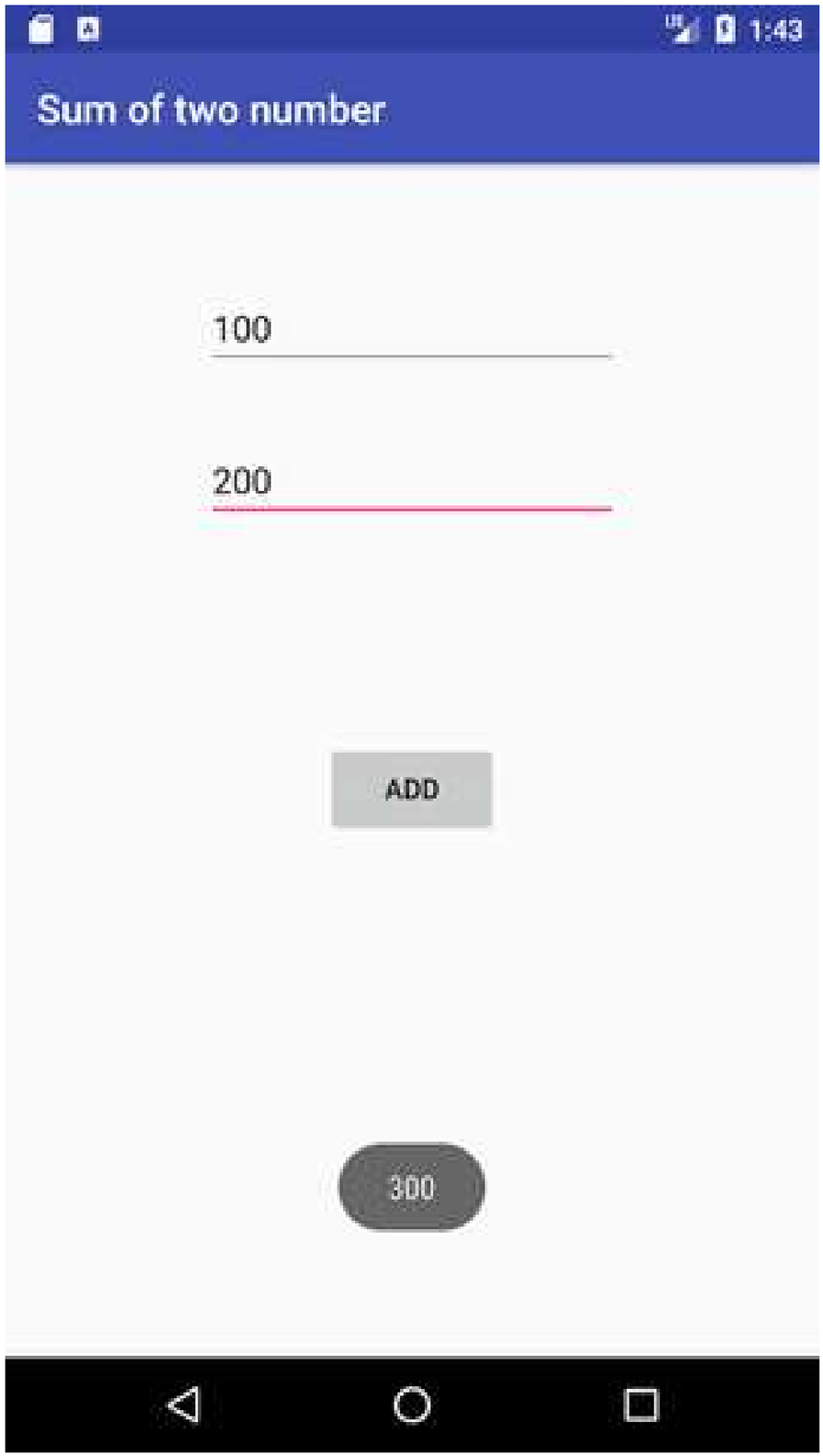
```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="example.javatpoint.com.sumoftwonumber.MainActivity">

    <EditText
        android:id="@+id/editText1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="61dp"
        android:ems="10"
        android:inputType="number"
        tools:layout_editor_absoluteX="84dp"
        tools:layout_editor_absoluteY="53dp" />
```

```
<EditText
    android:id="@+id/editText2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@+id/editText1"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="32dp"
    android:ems="10"
    android:inputType="number"
    tools:layout_editor_absoluteX="84dp"
    tools:layout_editor_absoluteY="127dp" />

  <Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@+id/editText2"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="109dp"
    android:text="ADD"
    tools:layout_editor_absoluteX="148dp"
    tools:layout_editor_absoluteY="266dp" />
</RelativeLayout>
```

MainActivity.java

```
package example.javatpoint.com.sumoftwonumber;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
    private EditText edittext1, edittext2;
    private Button buttonSum;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        addListenerOnButton();
    }
```

```
public void addListenerOnButton() {
    edittext1 = (EditText) findViewById(R.id.editText1);
    edittext2 = (EditText) findViewById(R.id.editText2);
    buttonSum = (Button) findViewById(R.id.button);

    buttonSum.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            String value1=edittext1.getText().toString();
            String value2=edittext2.getText().toString();
            int a=Integer.parseInt(value1);
            int b=Integer.parseInt(value2);
            int sum=a+b;
            Toast.makeText(getApplicationContext(),String.valueOf(sum), Toast.LENGTH_LONG).show()
;
        }
    });
  }
}
```


1:43
Sum of two number
100
200
ADD
300

ANDROID UI CONTROLS

3. Image Button

Image Button is a user interface control which is used to display a button with image to perform an action when user click or tap on it.

Generally, the Image button in android looks similar as regular Button and perform the actions same as regular button but only difference is for image button we will add an image instead of text.

Create ImageButton in XML Layout File

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/a
ndroid"
    android:orientation="vertical" android:layout_width="match_parent
"
    android:layout_height="match_parent">
    <ImageButton
        android:id="@+id/addBtn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/add_icon" />
</LinearLayout>
```

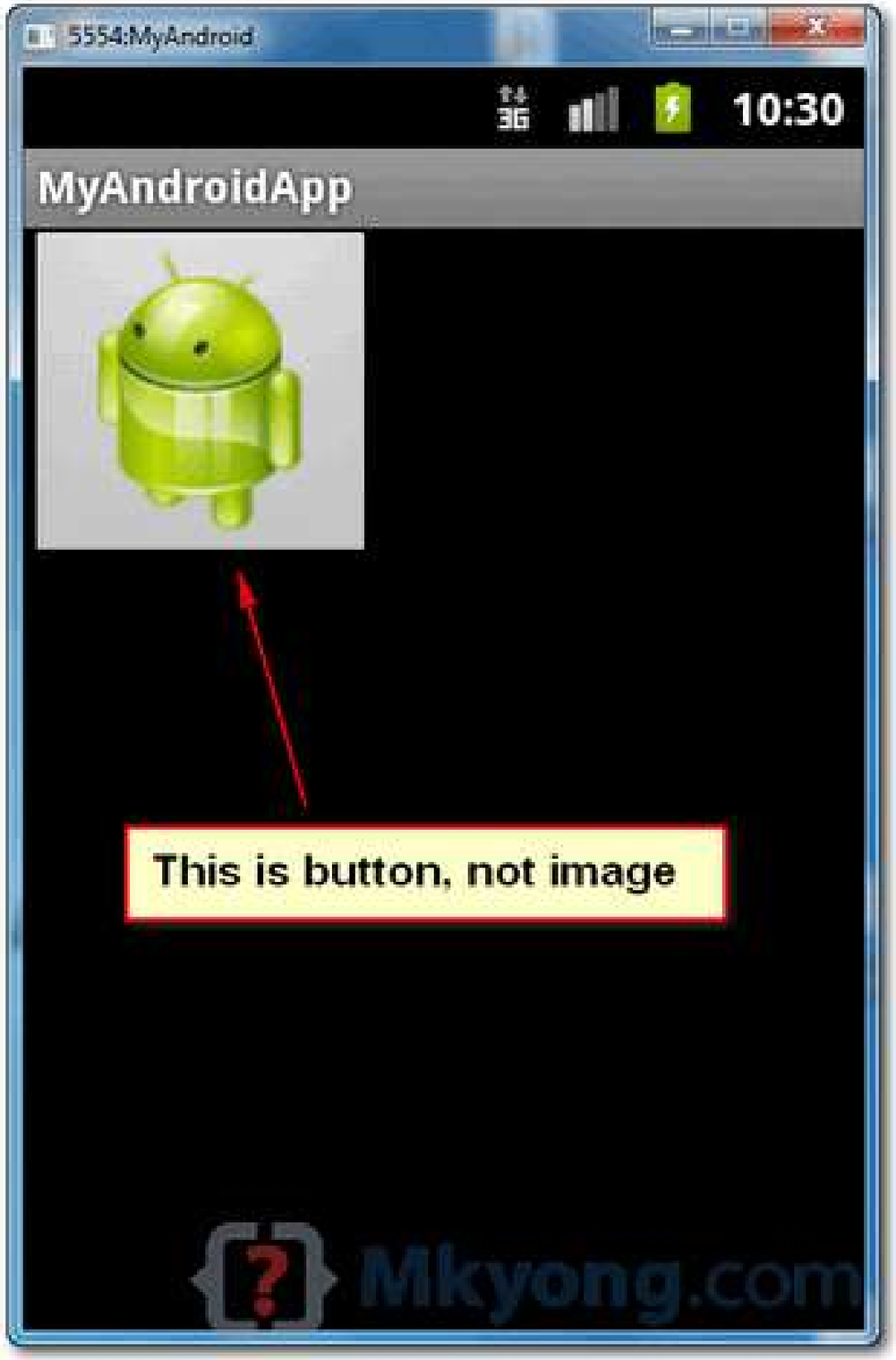
activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <ImageButton
        android:id="@+id/imageButton1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/android_button" />

</LinearLayout>
```

MainActivity.java

```
package com.mkyong.android;
import android.app.Activity;
import android.os.Bundle;
import android.widget.ImageButton;
import android.widget.Toast;
import android.view.View;
import android.view.View.OnClickListener;

public class MyAndroidAppActivity extends Activity {
        ImageButton imageButton;
        @Override
        public void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.main);
                addListenerOnButton();
        }
```

```
public void addListenerOnButton() {
        imageButton = findViewById(R.id.imageButton1);
        imageButton.setOnClickListener(new OnClickListener() {
                @Override
                public void onClick(View arg) {
                   Toast.makeText(getApplicationContext(),
                "ImageButton is clicked!", Toast.LENGTH_SHORT).show();
                }
        });
        }
}
```


5554:MyAndroid
10:30
MyAndroidApp
This is button, not image
Mkyong.com

ANDROID UI CONTROLS

4. Toggle Button

In android, **Toggle Button** is a user interface control which is used to display ON (Checked) or OFF (Unchecked) states as a button with a light indicator.

Create ToggleButton in XML Layout File

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res
/android"
    android:layout_width="match_parent" android:layout_height="mat
ch_parent">
    <ToggleButton
        android:id="@+id/toggle1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="100dp"
        android:layout_marginTop="120dp"
        android:checked="true"
        android:textOff="OFF"
        android:textOn="ON"/>
</RelativeLayout>
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="example.javatpoint.com.togglebutton.MainActivity">
    <ToggleButton
        android:id="@+id/toggleButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="8dp"
        android:layout_marginTop="80dp"
        android:text="ToggleButton"
        android:textOff="Off"
        android:textOn="On"
        app:layout_constraintEnd_toStartOf="@+id/toggleButton2"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
```

```
<ToggleButton
    android:id="@+id/toggleButton2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginRight="60dp"
    android:layout_marginTop="80dp"
    android:text="ToggleButton"
    android:textOff="Off"
    android:textOn="On"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

  <Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginBottom="144dp"
    android:layout_marginLeft="148dp"
    android:text="Submit"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintStart_toStartOf="parent" />
```

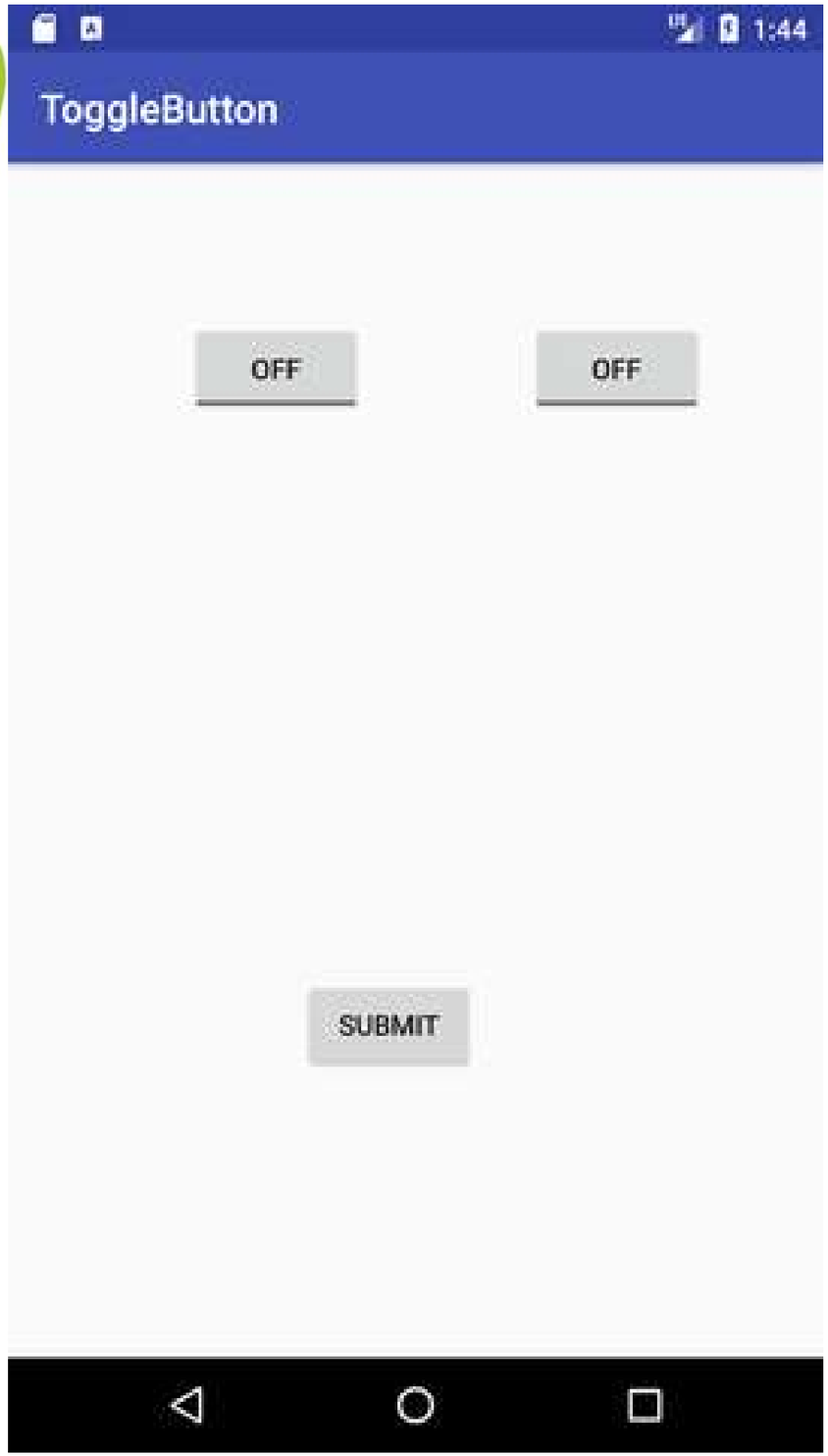


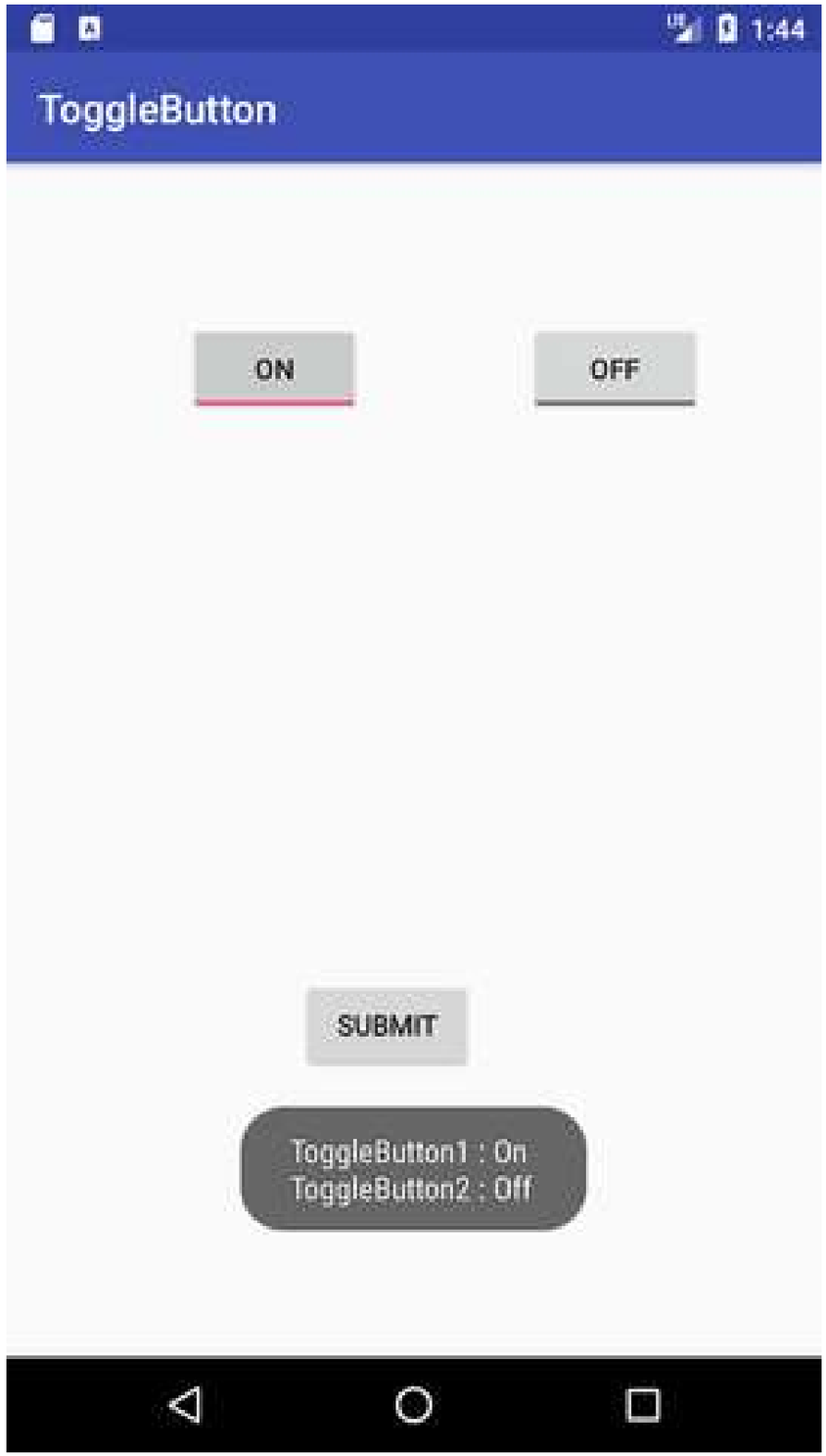
```
</android.support.constraint.ConstraintLayout>
```

MainActivity.java

```
package example.javatpoint.com.togglebutton;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;
import android.widget.ToggleButton;

public class MainActivity extends AppCompatActivity {
    private ToggleButton toggleButton1, toggleButton2;
    private Button buttonSubmit;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        addListenerOnButtonClick();
    }
```

```
public void addListenerOnButtonClick(){
    //Getting the ToggleButton and Button instance from the layout xml file
    toggleButton1=(ToggleButton)findViewById(R.id.toggleButton);
    toggleButton2=(ToggleButton)findViewById(R.id.toggleButton2);
    buttonSubmit=(Button)findViewById(R.id.button);

    //Performing action on button click
    buttonSubmit.setOnClickListener(new View.OnClickListener(){

        @Override
        public void onClick(View view) {
            StringBuilder result = new StringBuilder();
            result.append("ToggleButton1 : ").append(toggleButton1.getText());
            result.append("\nToggleButton2 : ").append(toggleButton2.getText());
            //Displaying the message in toast
            Toast.makeText(getApplicationContext(), result.toString(),Toast.LENGTH_LONG).sh
ow();
        }

    });

  }
}
```


1:44
ToggleButton
OFF
OFF
SUBMIT


1:44
ToggleButton
ON
OFF
SUBMIT
ToggleButton1 : On
ToggleButton2 : Off

ANDROID UI CONTROLS

5. Radio Button & Radio Button Group

In android, **Radio Button** is a two states button that can be either checked or unchecked and it cannot be unchecked once it is checked.

Create RadioButton in XML Layout File

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res
/android"
    android:layout_width="match_parent" android:layout_height="mat
ch_parent">
<RadioGroup
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">
    <RadioButton
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Java"
        android:checked="true"/>
</RelativeLayout>
```

Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Select your Subject ?"
        android:textStyle="bold"
        android:layout_marginLeft="10dp"
        android:textSize="20dp"/>

    <!-- add RadioGroup which contain the many RadioButton-->
```

```
<RadioGroup
    android:layout_marginTop="50dp"
    android:id="@+id/groupradio"
    android:layout_marginLeft="10dp"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content">

     <!-- In RadioGroup create the 1 Radio Button-->
     <!-- like this we will add some more Radio Button-->
    <RadioButton
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:id="@+id/radia_id1"
        android:text="DBMS"
        android:textSize="20dp"/>

    <RadioButton
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:id="@+id/radia_id2"
        android:text="C/C++ Programing"
        android:textSize="20dp"/>
```

```
<RadioButton
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:id="@+id/radia_id3"
        android:text="Data Structure"
        android:textSize="20dp"/>

    <RadioButton
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:id="@+id/radia_id4"
        android:text="Algorithms"
        android:textSize="20dp"/>
</RadioGroup>

<!-- add button For Submit the Selected item-->
```

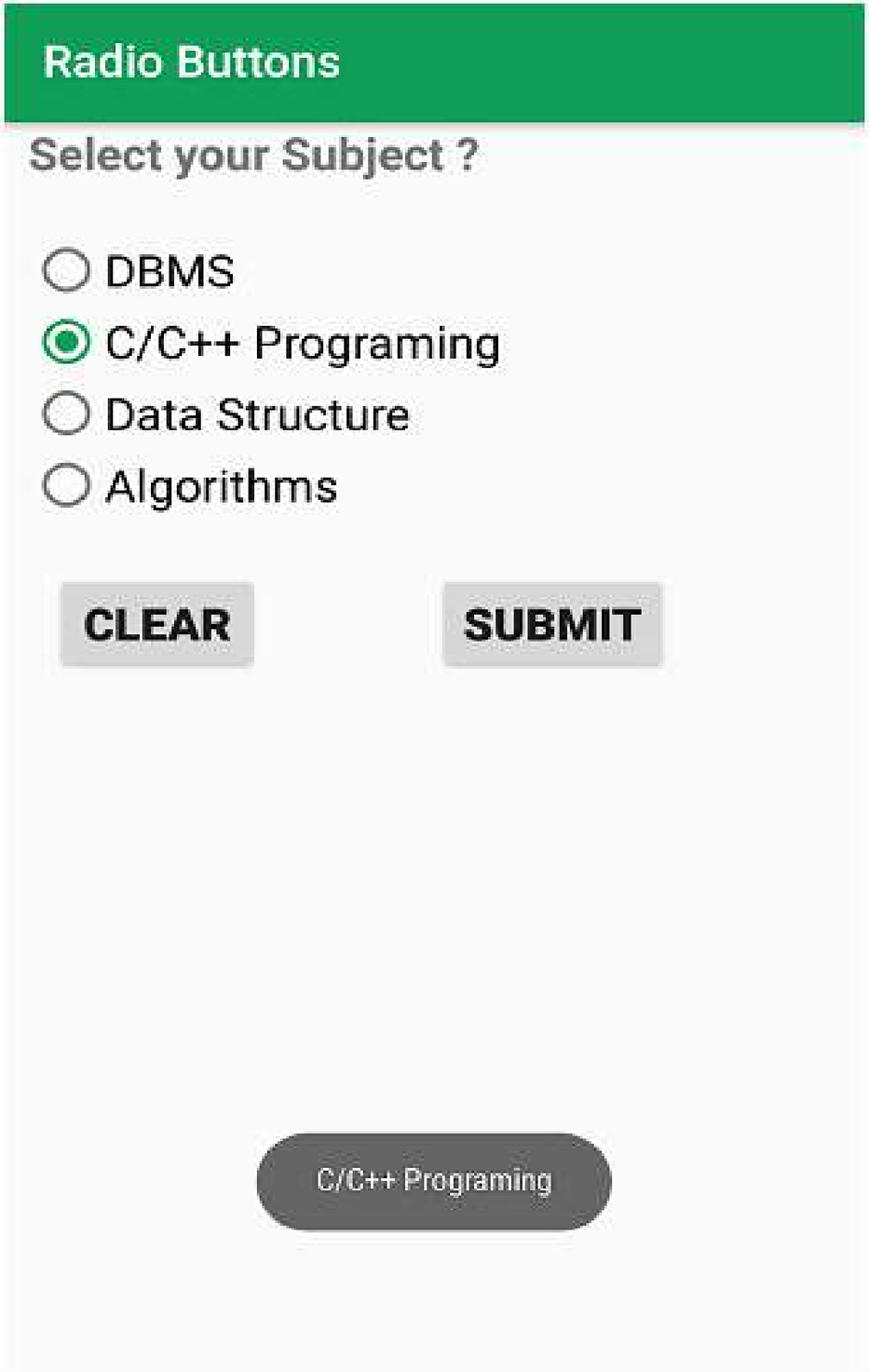
```
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Submit"
    android:id="@+id/submit"
    android:textStyle="bold"
    android:textSize="20dp"
    android:layout_marginTop="200dp"
    android:layout_marginLeft="180dp"
    />

  <!-- add clear button for clear the selected item-->
  <Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Clear"
    android:id="@+id/clear"
    android:textSize="20dp"
    android:textStyle="bold"
    android:layout_marginTop="200dp"
    android:layout_marginLeft="20dp"
    />

</RelativeLayout>
```

Activity.java

```
package org.geeksforgeeks.navedmalik.radiobuttons;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
    private RadioGroup radioGroup;
  Button submit, clear;

  @Override
  protected void onCreate(Bundle savedInstanceState)
  {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);

      submit = findViewById(R.id.submit);
      clear = findViewById(R.id.clear);
      radioGroup = findViewById(R.id.groupradio);

      radioGroup.clearCheck();
```

```
radioGroup.setOnCheckedChangeListener(
        new RadioGroup
            .OnCheckedChangeListener() {
                @Override
                 public void onCheckedChanged(RadioGroup group,
                                              int checkedId)
                {
                     RadioButton radioButton = group.findViewById(checkedId);
                }
            });

        submit.setOnClickListener(new View.OnClickListener() {

         @Override
```

```
public void onClick(View v)
        {
            int selectedId = radioGroup.getCheckedRadioButtonId();
          if (selectedId == -1) {
              Toast.makeText(MainActivity.this,"No answer has been
selected",Toast.LENGTH_SHORT).show();
          }
          else {
              RadioButton radioButton  = radioGroup.findViewById(selectedId);
              Toast.makeText(MainActivity.this,radioButton.getText(),Toast.LENGTH_SHORT).show();
          }
        }
      });
        clear.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v)
        {
            radioGroup.clearCheck();
        }
      });
    }
}
```


Radio Buttons
Select your Subject ?
DBMS
C/C++ Programing
Data Structure
Algorithms
CLEAR
SUBMIT
C/C++ Programing

ANDROID UI CONTROLS

6. CheckBox

Checkbox is a two states button that can be either checked or unchecked.

Create CheckBox in XML Layout File

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/androi
d"
   android:layout_width="match_parent" android:layout_height="match_par
ent">
<CheckBox
   android:id="@+id/chk1"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:checked="true"
   android:text="Java" /> </RelativeLayout>
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="example.javatpoint.com.checkbox.MainActivity">


    <CheckBox
        android:id="@+id/checkBox"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="144dp"
        android:layout_marginTop="68dp"
        android:text="Pizza"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
```

```
<CheckBox
    android:id="@+id/checkBox2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="144dp"
    android:layout_marginTop="28dp"
    android:text="Coffee"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/checkBox" />

<CheckBox
    android:id="@+id/checkBox3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="144dp"
    android:layout_marginTop="28dp"
    android:text="Burger"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/checkBox2" />
```

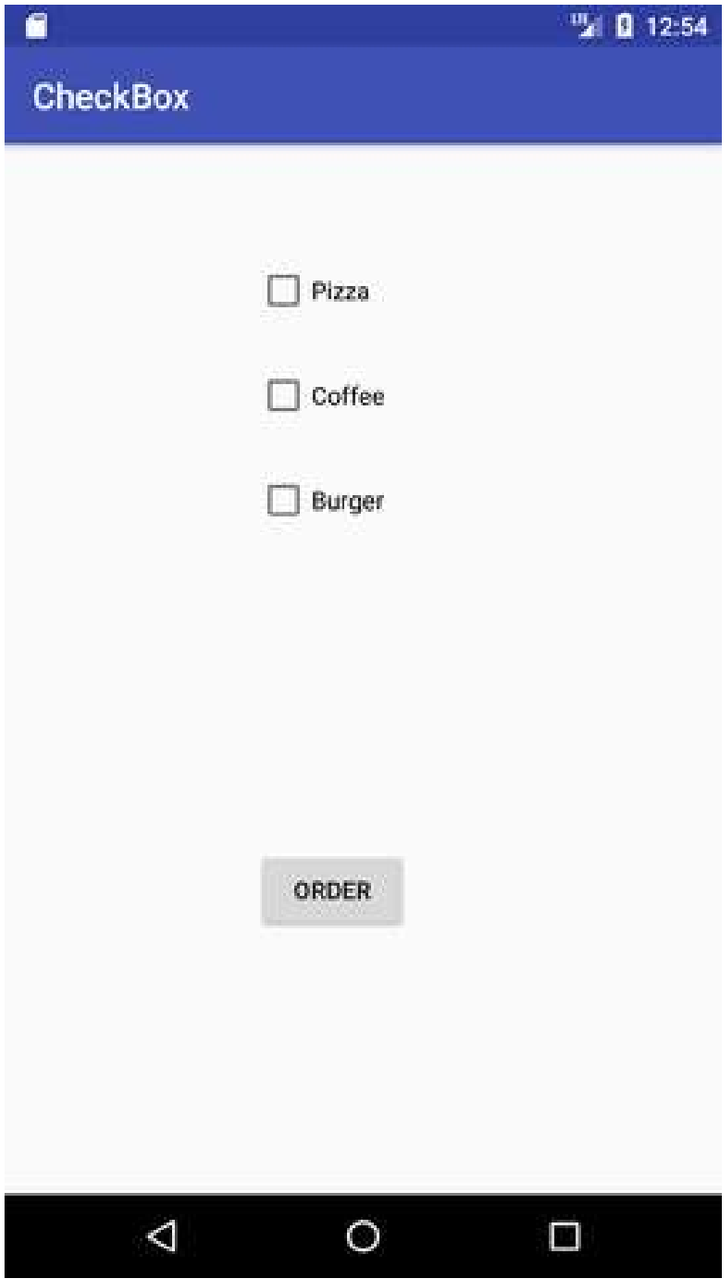
```
<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="144dp"
    android:layout_marginTop="184dp"
    android:text="Order"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/checkBox3" /
>

</android.support.constraint.ConstraintLayout>
```

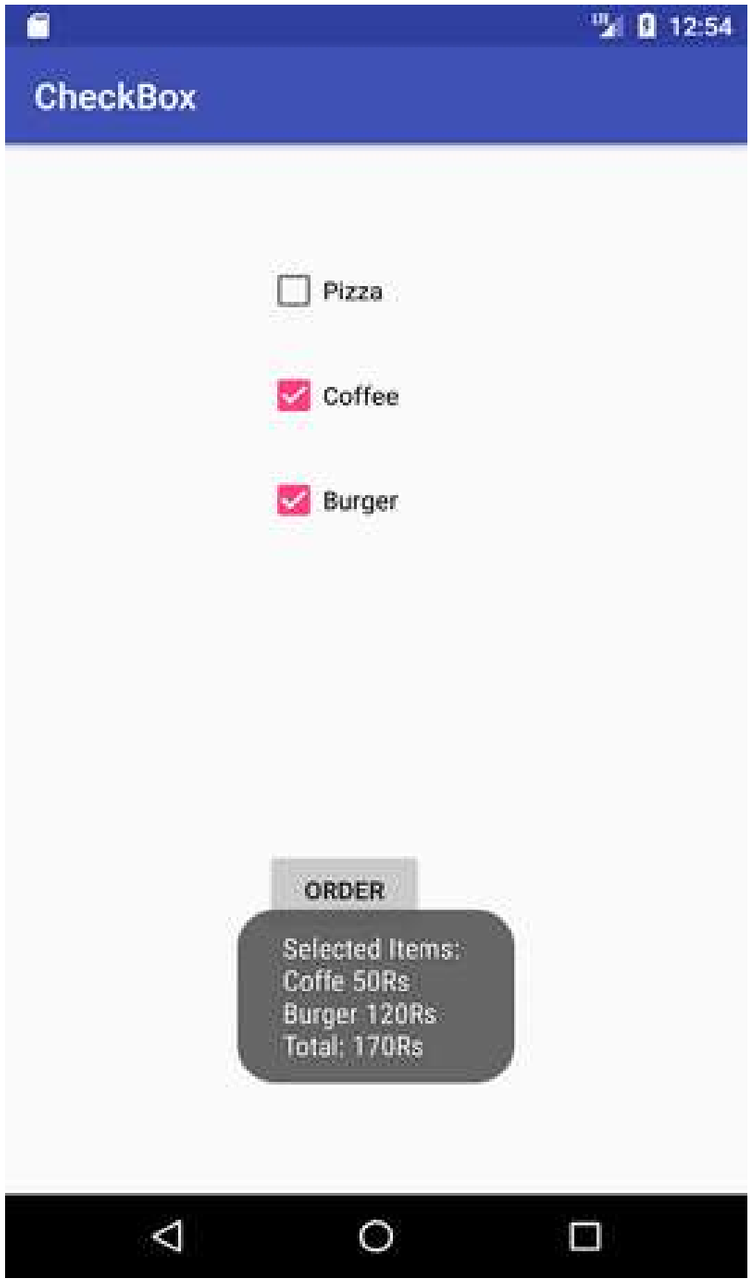
MainActivity.java

```
package example.javatpoint.com.checkbox;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
    CheckBox pizza,coffe,burger;
    Button buttonOrder;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        addListenerOnButtonClick();
    }
```

```
public void addListenerOnButtonClick(){
    //Getting instance of CheckBoxes and Button from the activty_m
ain.xml file
    pizza=findViewById(R.id.checkBox);
    coffe=findViewById(R.id.checkBox2);
    burger=findViewById(R.id.checkBox3);
    buttonOrder=findViewById(R.id.button);

    //Applying the Listener on the Button click
    buttonOrder.setOnClickListener(new View.OnClickListener(){

        @Override
        public void onClick(View view) {
            int totalamount=0;
            StringBuilder result=new StringBuilder();
            result.append("Selected Items:");
```

```
if(pizza.isChecked()){
            result.append("\nPizza 100Rs");
            totalamount+=100;
        }
        if(coffe.isChecked()){
            result.append("\nCoffe 50Rs");
            totalamount+=50;
        }
        if(burger.isChecked()){
            result.append("\nBurger 120Rs");
            totalamount+=120;
        }
        result.append("\nTotal: "+totalamount+"Rs");
        //Displaying the message on the toast
        Toast.makeText(getApplicationContext(), result.toString(), Toast.LENGTH_LONG).show();
        }

    });
  }
}
```


12:54
CheckBox
Pizza
Coffee
Burger
ORDER


12:54
CheckBox
Pizza
Coffee
Burger
ORDER
Selected Items:
Coffe 50Rs
Burger 120Rs
Total: 170Rs

ANDROID UI CONTROLS

7. Progress Bar

ProgressBar is a user interface control which is used to indicate the progress of an operation.

Create Android ProgressBar in XML Layout File

```
<ProgressBar
    android:id="@+id/pBar3"
    style="?android:attr/progressBarStyleHorizontal"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:minHeight="50dp"
    android:minWidth="250dp"
    android:max="100"
    android:indeterminate="true"
    android:progress="1" />
```

activity_main.xml

```
<RelativeLayout xmlns:androclass="http://schemas.android.com/apk/res/a
ndroid"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity" >

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="116dp"
        android:text="download file" />

</RelativeLayout>
```

MainActivity.java

```
package example.javatpoint.com.progressbar;

import android.app.ProgressDialog;
import android.os.Handler;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class MainActivity extends AppCompatActivity {
    Button btnStartProgress;
    ProgressDialog progressBar;
    private int progressBarStatus = 0;
    private Handler progressBarHandler = new Handler();
    private long fileSize = 0;
    @Override
```

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    addListenerOnButtonClick();
  }
  public void addListenerOnButtonClick() {
    btnStartProgress = findViewById(R.id.button);
    btnStartProgress.setOnClickListener(new View.OnClickListener(){
@Override
        public void onClick(View v) {
          // creating progress bar dialog
          progressBar = new ProgressDialog(v.getContext());
          progressBar.setCancelable(true);
          progressBar.setMessage("File downloading ...");
          progressBar.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL);
          progressBar.setProgress(0);
          progressBar.setMax(100);
          progressBar.show();
         progressBarStatus = 0;
          fileSize = 0;
```

```
new Thread(new Runnable() {
        public void run() {
            while (progressBarStatus < 100) {
                // performing operation
                progressBarStatus = doOperation();
                try {
                    Thread.sleep(1000);
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
                // Updating the progress bar
                progressBarHandler.post(new Runnable() {
                    public void run() {
                        progressBar.setProgress(progressBarStatus);
                    }
                });
            }
```

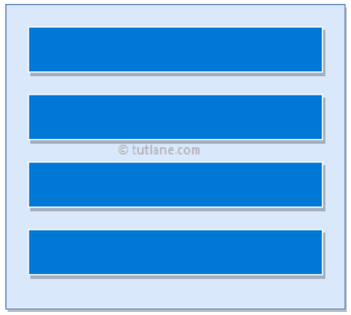
```
// performing operation if file is downloaded,
                if (progressBarStatus >= 100) {
                    // sleeping for 1 second after operation completed
                    try {
                        Thread.sleep(1000);
                    } catch (InterruptedException e) {
                        e.printStackTrace();
                    }
                    // close the progress bar dialog
                    progressBar.dismiss();
                }
            }
        }).start();
    }//end of onClick method
  });
}
```

```
// checking how much file is downloaded and updating the filesize
    public int doOperation() {
      //The range of ProgressDialog starts from 0 to 10000
      while (fileSize <= 10000) {
          fileSize++;
          if (fileSize == 1000) {
              return 10;
          } else if (fileSize == 2000) {
              return 20;
          } else if (fileSize == 3000) {
              return 30;
          } else if (fileSize == 4000) {
              return 40; // you can add more else if
          }
       /* else {
              return 100;
          }*/
      }//end of while
          return 100;
    }//end of doOperation
}
```


11:17
ProgressBar
DOWNLOAD FILE


11:18
ProgressBar
File downloading ...
40%
40/100

ANDROID UI CONTROLS



Adapter Views

1. ListView

In android, **ListView** is a **ViewGroup** which is used to display the list of scrollable of items in multiple rows and the list items are automatically inserted to the list using an **adapter**.

Listview is present inside Containers. From there you can drag and drop on virtual mobile screen to create it. Alternatively you can also XML code to create it.

Adapter: To fill the data in a ListView we simply use adapters. List items are automatically inserted to a list using an Adapter that pulls the content from a source such as an arraylist, array or database.

```
<ListView xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:tools="http://schemas.android.com/tools"

        android:id="@+id/simpleListView"

        android:layout_width="match_parent"

        android:layout_height="wrap_content" >

</ListView>
```

ANDROID UI CONTROLS

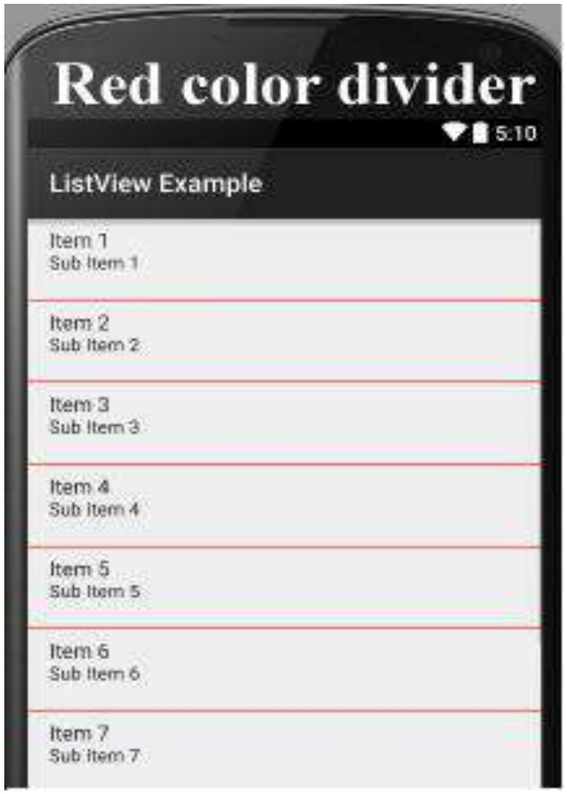
Attributes of ListView:

1. **id:** id is used to uniquely identify a ListView.

2. **divider:** This is a drawable or color to draw between different list items.

3. **dividerHeight:** This specify the height of the divider between list items. This could be in dp(density pixel),sp(scale independent pixel) or px(pixel).

4. **listSelector:** listSelector property is used to set the selector of the listView. It is generally orange or Sky blue color mostly but you can also define your custom color or an image as a list selector as per your design.



ANDROID UI CONTROLS

Adapter Views

Adapters Use in ListView:

ListView is a subclass of AdapterView and it can be populated by binding to an Adapter, which retrieves the data from an external source and creates a View that represents each data entry.

In android commonly used adapters are:

Array Adapter

Base Adapter

1.Array Adapter:

Whenever you have a list of single items which is backed by an array, you can use ArrayAdapter. For instance, list of phone contacts, countries or names. By default, ArrayAdapter expects a Layout with a single TextView,

2.Base Adapter:

BaseAdapter is a common base class of a general implementation of an Adapter that can be used in ListView. Base Adapter can be extended to create a custom Adapter for displaying a custom list item.

ANDROID UI CONTROLS

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<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"
   android:orientation="vertical">
   <ListView
      android:id="@+id/userlist"
      android:layout_width="match_parent"
      android:layout_height="wrap_content" >
   </ListView>
</LinearLayout>
```

ANDROID UI CONTROLS

MainActivity.java

```
package com.tutlane.listview;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.ArrayAdapter;
import android.widget.ListView;

public class MainActivity extends AppCompatActivity {
    private ListView mListView;
    private ArrayAdapter aAdapter;
    private String[] users = { "Suresh Dasari", "Rohini Alavala", "Trishika Dasar
i", "Praveen Alavala", "Madav Sai", "Hamsika Yemineni"};
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        mListView = (ListView) findViewById(R.id.userlist);
        aAdapter = new ArrayAdapter(this, android.R.layout.simple_list_item_1
, users);
        mListView.setAdapter(aAdapter);
    }
}
```

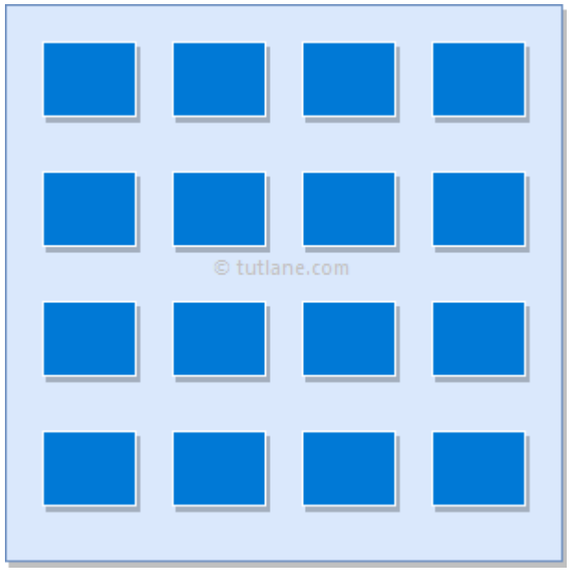

11:37
ListView
Suresh Dasari
Rohini Alavala
Trishika Dasari
Praveen Alavala
Madav Sai
Hamsika Yemineni
© tutlane.com

ANDROID UI CONTROLS

Adapter Views

2. GridView

In android, **Grid View** is a **ViewGroup** which is used to display items in a two dimensional, scrollable grid and grid items are automatically inserted to the gridview layout using a list adapter.



ANDROID UI CONTROLS

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<GridView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/gridView1"
    android:numColumns="auto_fit"
    android:gravity="center"
    android:columnWidth="50dp"
    android:stretchMode="columnWidth"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

</GridView>
```

ANDROID UI CONTROLS

MainActivity.java

```
import android.app.Activity;
import android.os.Bundle;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.GridView;
import android.widget.TextView;
import android.widget.Toast;
import android.view.View;
import android.widget.AdapterView.OnItemClickListener;

public class GridViewActivity extends Activity {

        GridView gridView;

        static final String[] numbers = new String[] {
                        "A", "B", "C", "D", "E",
                        "F", "G", "H", "I", "J",
                        "K", "L", "M", "N", "O",
                        "P", "Q", "R", "S", "T",
                        "U", "V", "W", "X", "Y", "Z"};
```

ANDROID UI CONTROLS

```
@Override
        public void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);

                setContentView(R.layout.main);

                gridView = (GridView) findViewById(R.id.gridView1);

                ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
                                android.R.layout.simple_list_item_1, numbers);

                gridView.setAdapter(adapter);

                gridView.setOnItemClickListener(new OnItemClickListener() {
                        public void onItemClick(AdapterView<?> parent, View v,
                                int position, long id) {
                          Toast.makeText(getApplicationContext(),
                                ((TextView) v).getText(), Toast.LENGTH_SHORT).show();
                        }
                });
        }
}
```

ANDROID UI CONTROLS

ANDROID UI CONTROLS

Adapter Views

3. ImageView

In Android, ImageView class is used to display an image file in application.

ANDROID UI CONTROLS

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<GridView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/gridView1"
    android:numColumns="auto_fit"
    android:gravity="center"
    android:columnWidth="50dp"
    android:stretchMode="columnWidth"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

</GridView>
```

ANDROID UI CONTROLS

MainActivity.java

```
import android.app.Activity;
import android.os.Bundle;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.GridView;
import android.widget.TextView;
import android.widget.Toast;
import android.view.View;
import android.widget.AdapterView.OnItemClickListener;

public class GridViewActivity extends Activity {

        GridView gridView;

        static final String[] numbers = new String[] {
                        "A", "B", "C", "D", "E",
                        "F", "G", "H", "I", "J",
                        "K", "L", "M", "N", "O",
                        "P", "Q", "R", "S", "T",
                        "U", "V", "W", "X", "Y", "Z"};
```

ANDROID UI CONTROLS

```
@Override
        public void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);

                setContentView(R.layout.main);

                gridView = (GridView) findViewById(R.id.gridView1);

                ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
                                android.R.layout.simple_list_item_1, numbers);

                gridView.setAdapter(adapter);

                gridView.setOnItemClickListener(new OnItemClickListener() {
                        public void onItemClick(AdapterView<?> parent, View v,
                                int position, long id) {
                          Toast.makeText(getApplicationContext(),
                                ((TextView) v).getText(), Toast.LENGTH_SHORT).show();
                        }
                });
        }
}
```

ANDROID UI CONTROLS


Good Luck!