DOP: / /2023 DOS: / /2023

Experiment No:4

Title: Building Android applications User interfaces using various Views and Layouts.

Theory:

View:

View is the basic building block of UI(User Interface) in android. View refers to the android.view.View class, which is the super class for all the GUI components like TextView, ImageView, Button etc.

View class extends Object class and implements Drawable.Callback, KeyEvent.Callback and AccessibilityEventSource.

View can be considered as a rectangle on the screen that shows some type of content. It can be an image, a piece of text, a button or anything that an android application can display. The rectangle here is actually invisible, but every view occupies a rectangle shape.

The question that might be bothering you would be , what can be the size of this rectangle?

The answer is either we can set it manually, by specifying the exact size(with proper units) or by using some predefined values. These predefined values are match_parent and wrap_content.

match_parent means it will occupy the complete space available on the display of the device. Whereas, wrap_content means it will occupy only that much space as required for its content to display.

XML syntax for creating a View

Now, as we have explained earlier as well, to draw anything in your android application, you will have to sepcify it in the design XML files. And to add functionality we will create Java files.

```
<ViewName

  Attribute1=Value1

  Attribute2=Value2

  Attribute3=Value3

  AttributeN=ValueN

/>
```

Most commonly used Android View classes

Here we have some of the most commonly used android View classes:

- TextView
- EditText
- Button
- ImageView
- ImageButton
- CheckBox
- RadioButton
- ListView
- GridView

ViewGroup:

A ViewGroup is a special view that can contain other views. The ViewGroup is the base class for Layouts in android, like LinearLayout, RelativeLayout, FrameLayout etc.

In other words, ViewGroup is generally used to define the layout in which views(widgets) will be set/arranged/listed on the android screen.

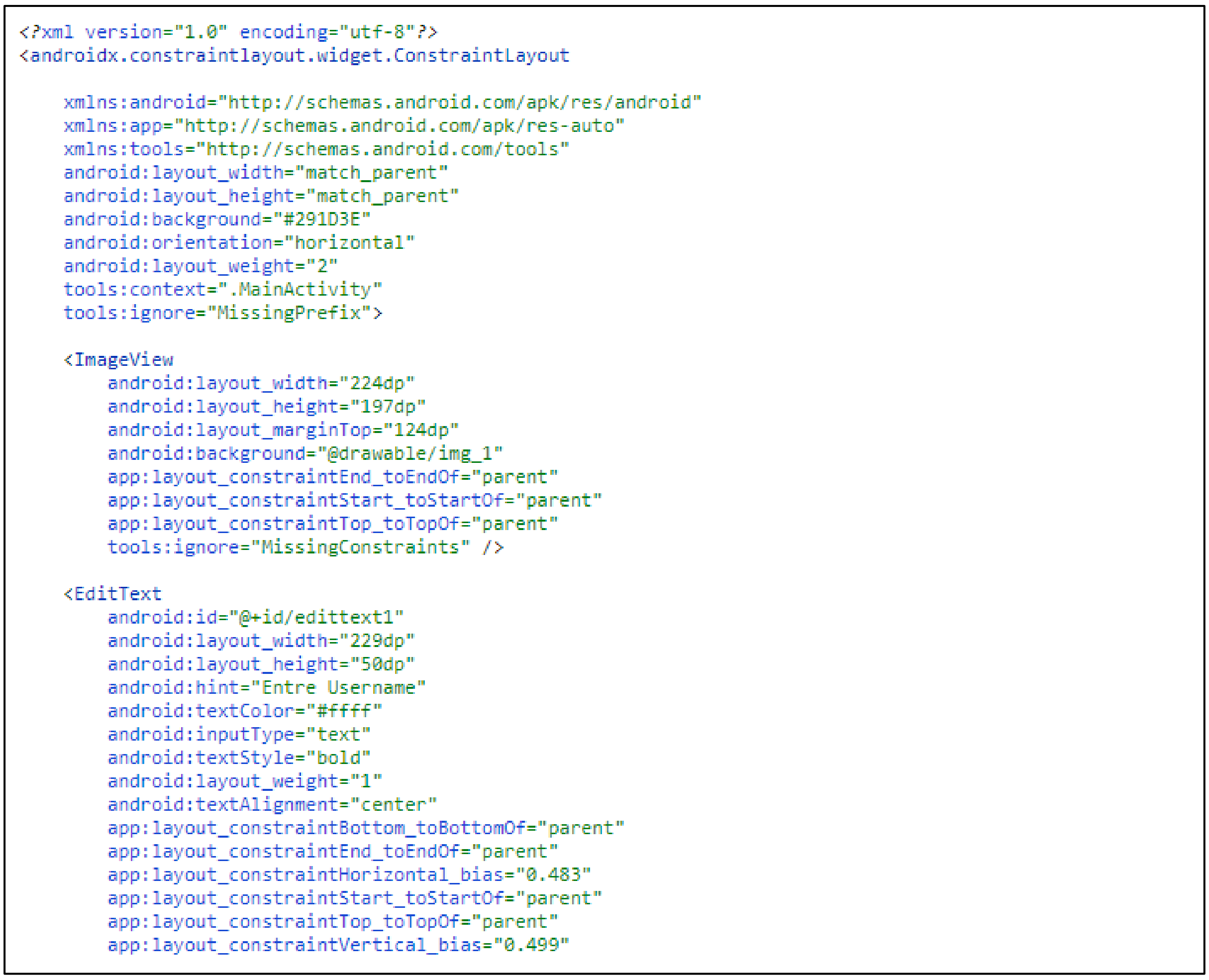
ViewGroups acts as an invisible container in which other Views and Layouts are placed. Yes, a layout can hold another layout in it, or in other words a ViewGroup can have another ViewGroup in it.

Most commonly used Android layout types

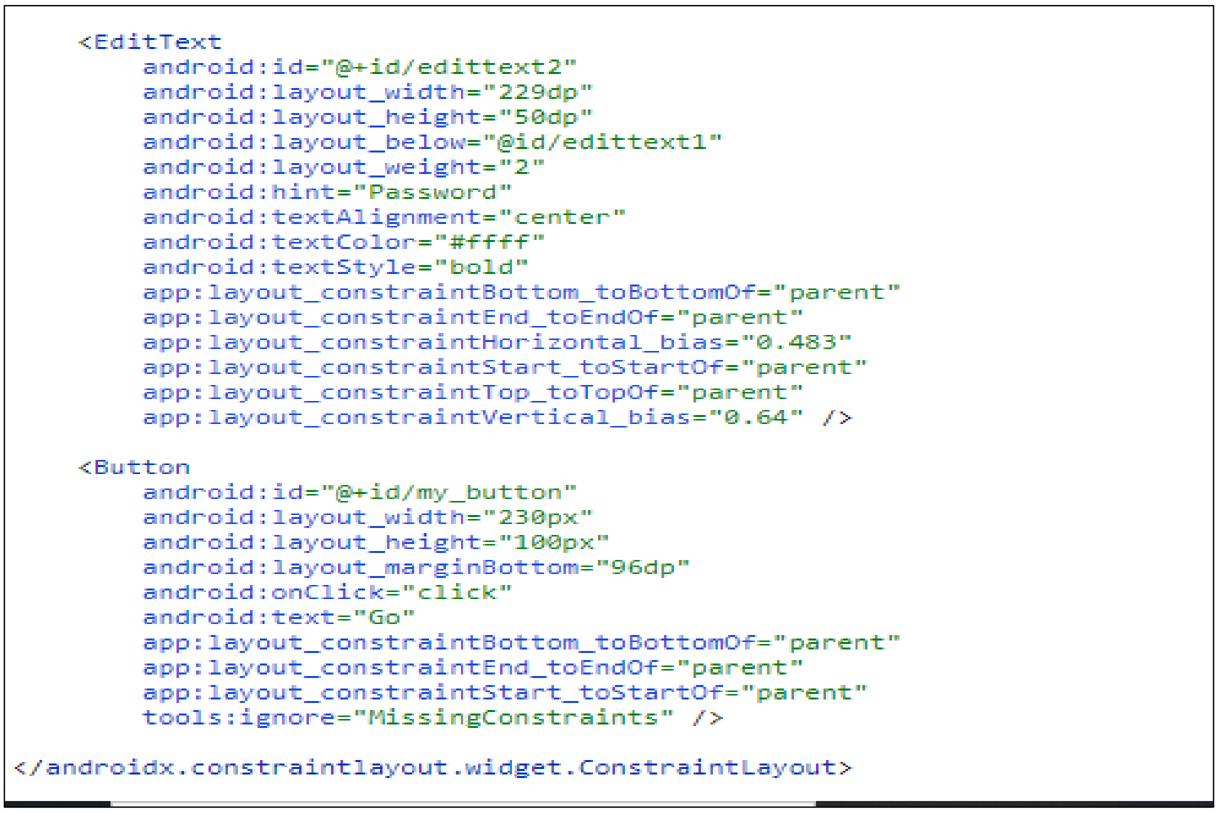
- LinearLayout
- RelativeLayout
- Web View
- Tabular layout
- ListView
- GridView

Input:

activity_main.xml:



```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.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"
    android:background="#291D3E"
    android:orientation="horizontal"
    android:layout_weight="2"
    tools:context=".MainActivity"
    tools:ignore="MissingPrefix">

    <ImageView
        android:layout_width="224dp"
        android:layout_height="197dp"
        android:layout_marginTop="124dp"
        android:background="@drawable/img_1"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        tools:ignore="MissingConstraints" />

    <EditText
        android:id="@+id/edittext1"
        android:layout_width="229dp"
        android:layout_height="50dp"
        android:hint="Entre Username"
        android:textColor="#ffff"
        android:inputType="text"
        android:textStyle="bold"
        android:layout_weight="1"
        android:textAlignment="center"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.483"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.499"
```



```xml
    <EditText
        android:id="@+id/edittext2"
        android:layout_width="229dp"
        android:layout_height="50dp"
        android:layout_below="@id/edittext1"
        android:layout_weight="2"
        android:hint="Password"
        android:textAlignment="center"
        android:textColor="#ffff"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.483"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.64" />

    <Button
        android:id="@+id/my_button"
        android:layout_width="230px"
        android:layout_height="100px"
        android:layout_marginBottom="96dp"
        android:onClick="click"
        android:text="Go"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        tools:ignore="MissingConstraints" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

activity_main.xml:

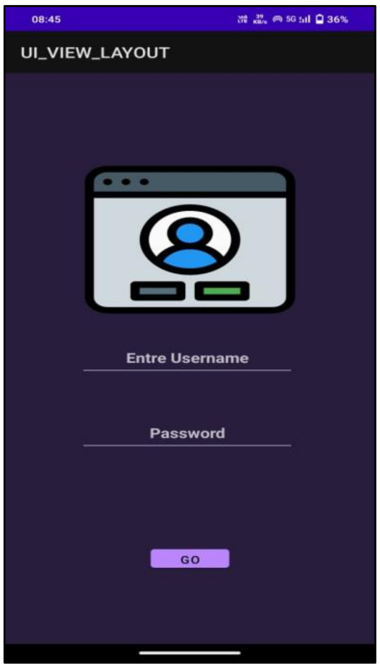
```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity2">

    <TextView
        android:id="@+id/received_value_id"
        android:layout_width="300dp"
        android:layout_height="50dp"
        android:layout_marginStart="40dp"
        android:layout_marginLeft="40dp"
        android:layout_marginTop="100dp"
        android:textSize="40sp"
        android:textStyle="bold" />
</RelativeLayout>
```

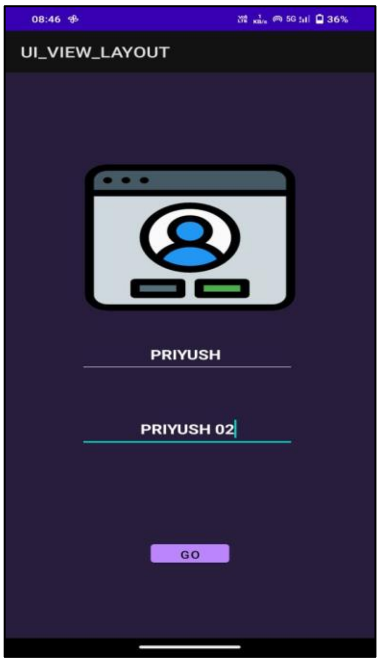
java

```
package com.example.ui_view_layout;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

public class MainActivity extends AppCompatActivity {
    EditText username;
    EditText passwaorld;
    Button button;


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

        username=findViewById(R.id.edittext1);
        passwaorld=findViewById(R.id.edittext2);
        button=findViewById(R.id.my_button);

        button.setOnClickListener(view ->{
            String str= username.getText().toString();
            String strr= passwaorld.getText().toString();
            Intent intent=new Intent(getApplicationContext(),MainActivity2.class);
            intent.putExtra("message_key",str);
            intent.putExtra("message_key",strr);
            startActivity(intent);

        } );

    }


}
```

```
package com.example.ui_view_layout;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.widget.TextView;

public class MainActivity2 extends AppCompatActivity {

    TextView recerview_msg;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main2);
        recerview_msg=findViewById(R.id.received_value_id);
        Intent intent=getIntent();
        String str=intent.getStringExtra("message_key");
        String strr=intent.getStringExtra("message_key");
       recerview_msg.setText(str);
       recerview_msg.setText(strr);

    }
}
```

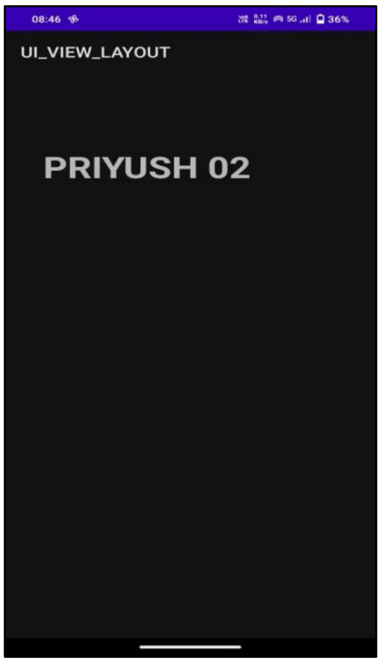
Output:


UI_VIEW_LAYOUT


UI_VIEW_LAYOUT

PRIYUSH

PRIYUSH 02


UI_VIEW_LAYOUT

PRIYUSH 02

Conclusion: - Hence successfully performed Building Android applications User interfaces using various Views and Layouts.