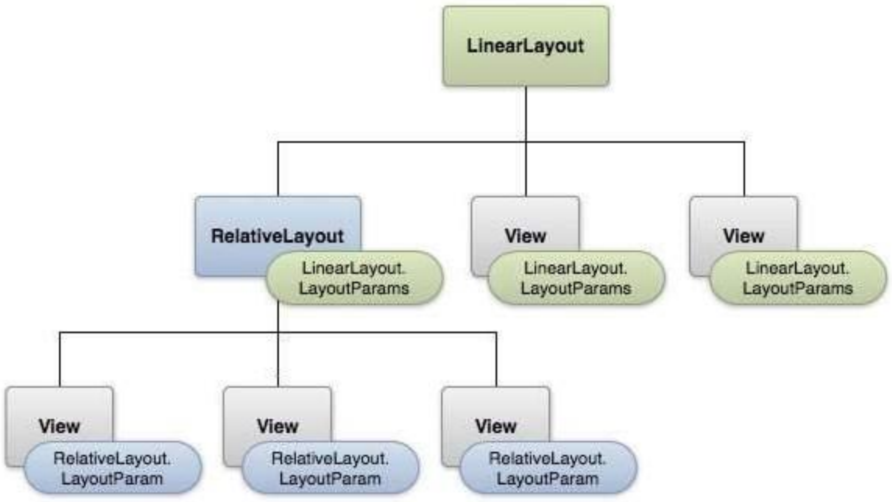
2.4 Screen Elements and Layouts

The basic building block for user interface is a **View** object which is created from the View class and occupies a rectangular area on the screen and is responsible for drawing and event handling.

View is the base class for widgets, which are used to create interactiveUI components like buttons, text fields, etc.

The **ViewGroup** is a subclass of **View** and provides invisiblecontainer that hold other Views or other ViewGroups and define their layout properties.

At third level we have different layouts which are subclasses of ViewGroup class and a typical layout defines the visual structure for an Android user interface and can be created either at run time using **View/ViewGroup** objects or you can declare your layout using simple XML file **main_layout.xml** which is located in the res/layout folder of your project.


LinearLayout
RelativeLayout
LinearLayout. LayoutParams
View
LinearLayout. LayoutParams
View
LinearLayout. LayoutParams
View
RelativeLayout. LayoutParam
View
RelativeLayout. LayoutParam
View
RelativeLayout. LayoutParam

This tutorial is more about creating your GUI based on layouts defined in XML file. A layout may contain any type of widgets such as buttons, labels, textboxes, and so on.

Following is a simple example of XML file having LinearLayout –

```
<?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" >

<TextView android:id="@+id/text" android:layout_width="wrap_content"
android:layout_height="wrap_content"android:text="This is a TextView" />
```

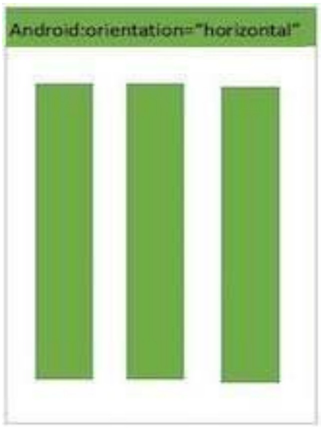
```
<Button android:id="@+id/button" android:layout_width="wrap_content"
  android:layout_height="wrap_content"android:text="This is a Button" />

  <!-- More GUI components go here -->

  </LinearLayout>
```

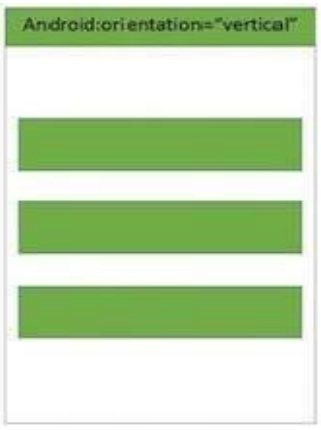
Android Layout Types

There are number of Layouts provided by Android which you will use in almost all the Android applications to provide different view, look and feel.

LinearLayout

LinearLayout is a view group that aligns all children in a singledirection, vertically or horizontally.





Relative Layout

Relative Layout is a view group that displays child views in relativepositions.

Android Relative Layout enables you to specify how child views are positioned relative to each other. The position of each view can be specified as relative to sibling elements or relative to the parent.

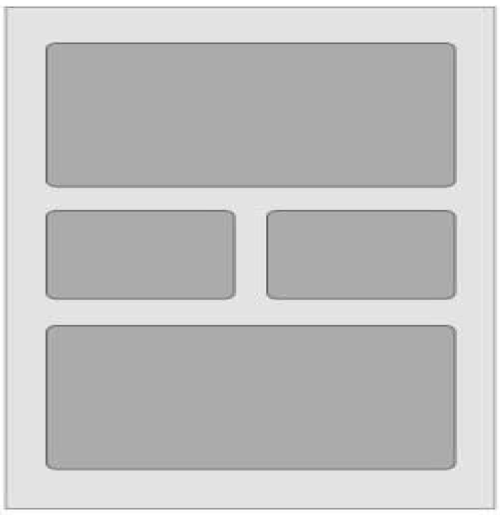
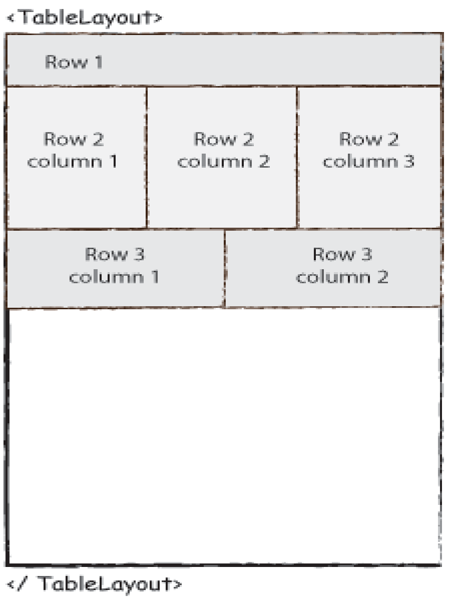


Table Layout

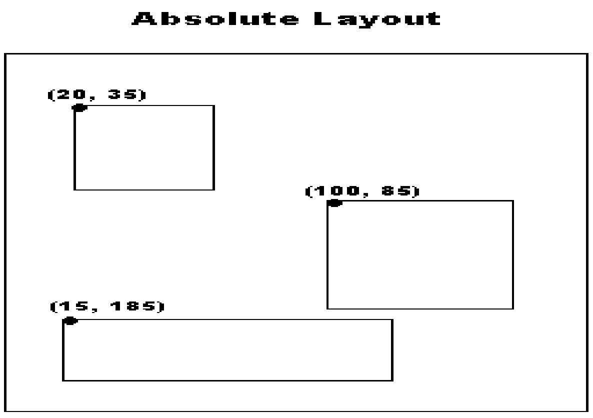
Android Table Layout going to be arranged groups of views into rows and columns. You will use the <TableRow> element to build a row in the table. Each row has zero or more cells; each cell can hold one View object.

Table Layout containers do not display border lines for their rows, columns, or cells.



Absolute Layout

An absolute layout lets you specify exact locations(x/y xcoordinates) of its children. Absolute layouts are less flexible and harder to maintain than other types of layouts without absolute positioning.

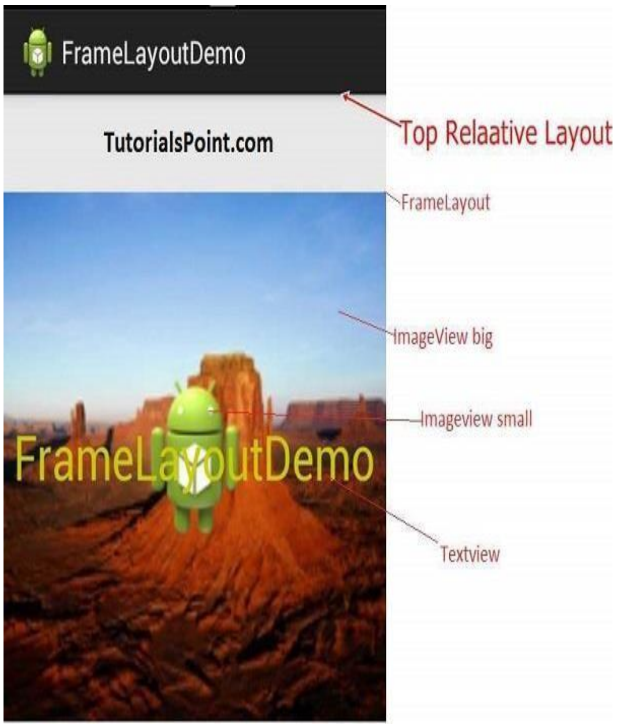


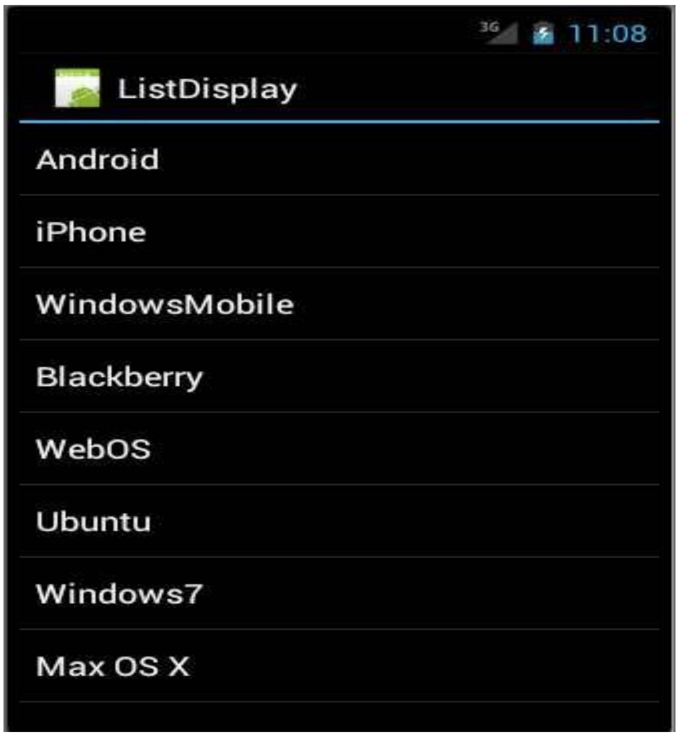
|

Frame Layout

Frame Layout is designed to block out an area on the screen to display a single item. Generally, FrameLayout should be used to hold a single child view, because it can be difficult to organize child views in a way that's scalable to different screen sizes without the children overlapping each other.

You can, however, add multiple children to a FrameLayout and control their position within the FrameLayout by assigning gravity to each child,





using the android:layout_gravity attribute.

ListView inserted to the list using an **Adapter** that pulls content from a sourcesuch as an array or database.

List View

An adapter actually bridges between UI components and the datasource that fill data into UI Component. Adapter holds the data and send the data to adapter view, the view can takes the data from adapterview and shows the data on different views like as spinner, list view, grid view etc.

The **ListView** and **GridView** are subclasses of **AdapterView** and they can be populated by binding them to an **Adapter**, which retrievesdata from an external source and creates a View that represents each data entry.

Android provides several subclasses of Adapter that are useful for retrieving different kinds of data and building views for an AdapterView (i.e. ListView or GridView). The common adaptersare **ArrayAdapter**,**Base Adapter**, **CursorAdapter**, **SimpleCursorAdapter**,**SpinnerAdapter** and **WrapperListAdapter**. We will see separate examples for boththe adapters.

GridView

Android **GridView** shows items in two-dimensional scrolling grid (rows & columns) and the grid items are not necessarily predetermined but they automatically inserted to the layout using a **ListAdapter**

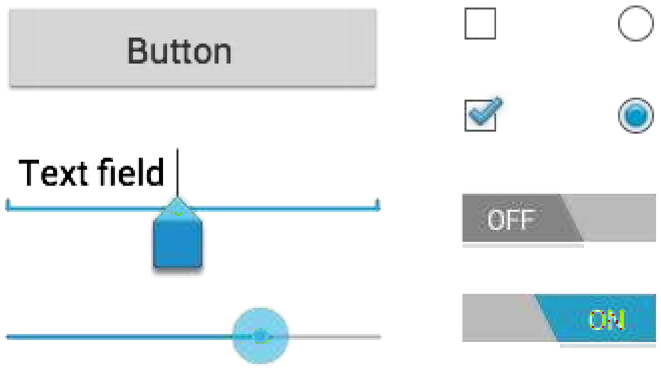
Informan
FEED
GRID

An adapter actually bridges between UI components and the datasource that fill data into UI Component. Adapter can be used to supplythe data to like spinner, list view, grid view etc.

The **ListView** and **GridView** are subclasses of **AdapterView** and they can be populated by binding them to an **Adapter**, which retrievesdata from an external source and creates a View that represents each data entry.

Screen Elements

Input controls are the interactive components in your app's user interface. Android provides a wide variety of controls you can use in your UI, such as buttons, text fields, seek bars, check box, zoom buttons, toggle buttons, and many more.


Button
Text field
OFF
ON