LECTURE – 4b

MOBILE APPLICATION DEVELOPMENT

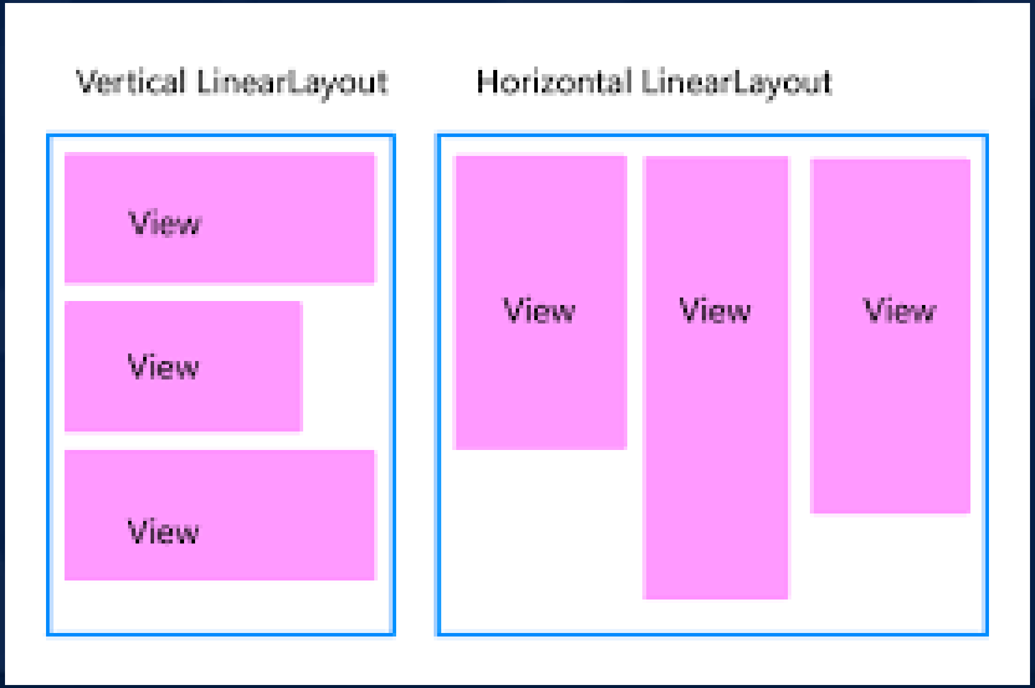
By: AbuBakar Ubaid

LAYOUTS

By: AbuBakar Ubaid

LINEARLAYOUT

- Good for smaller devices (like phones over Tablets) or when simple interface makes sense
- Layout in column (for Vertical) or row (for Horizontal) one after another child View objects
- Some Examples



LINEARLAYOUT

Good:

- Simple
- Know exactly how it will look on every device

Bad:

- Well for many interfaces too simple….

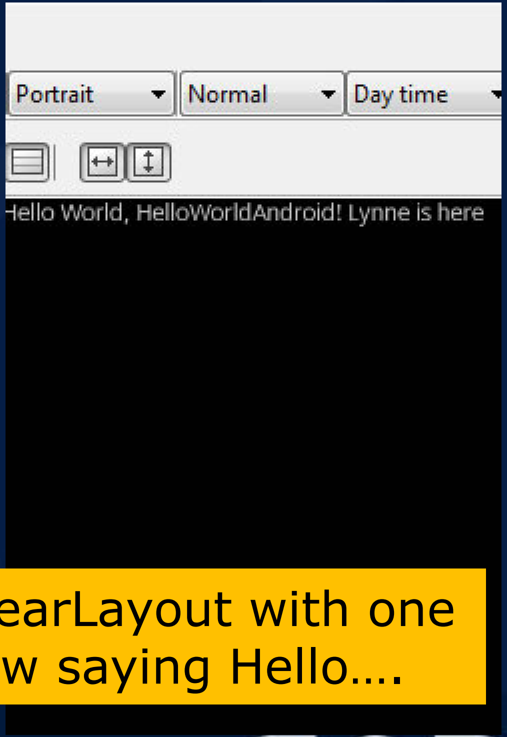
BUT → see next slide

- BUT, REMEMBER you can have a ViewGroup (another Layout) inside as a member of the LinearLayout to make a more COMPLEX interface
- ALSO can make more coplex

LinearLayout Very SIMPLE Example

- arranges by single column (vertical orientation)

```
<?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" >

 <Text View
      android:layout_width="fill_parent"
     android:layout_height="wrap_content"
     android:text="@string/hello"/>


</LinearLayout>
```

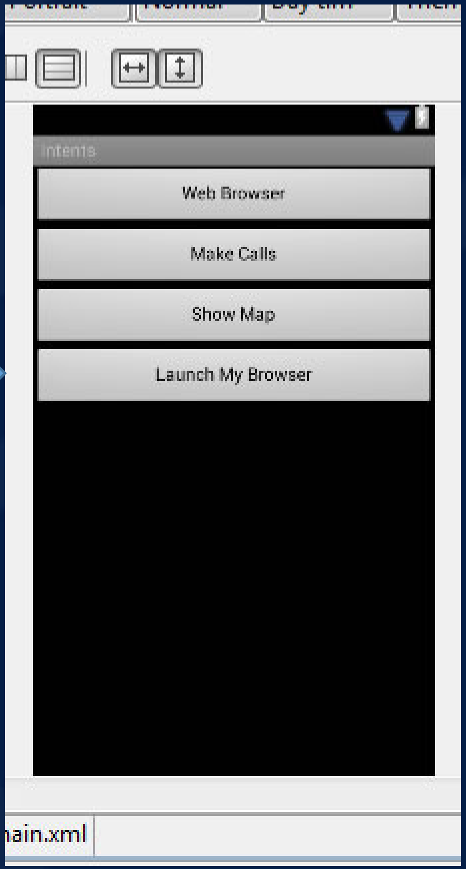


Hello World, HelloWorldAndroid! Lynne is here

VERY simple example – LinearLayout with one child View object, a TextView saying Hello….

LinearLayout Example 2

```
<?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" >

   <Button android:id="@+id/btn_webbrowser"
   android:layout_width="fill_parent"
   android:layout_height="wrap_content"
   android:text="Web Browser“
    android:onClick="onClickWebBrowser" />

   <Button android:id="@+id/btn_makecalls"
   android:layout_width="fill_parent"
   android:layout_height="wrap_content“
    android:text="Make Calls"
   android:onClick="onClickMakeCalls" />

   <Button android:id="@+id/btn_showMap"
   android:layout_width="fill_parent"
   android:layout_height="wrap_content“
    android:text="Show Map"
   android:onClick="onClickShowMap" />

   <Button android:id="@+id/btn_launchMyBrowser"
   android:layout_width="fill_parent"
   android:layout_height="wrap_content"
   android:text="Launch My Browser"
   android:onClick="onClickLaunchMyE

   </LinearLayout>
```



LinearLayout with 4 child View objects, all are buttons

LinearLayout attributes

- You can set either in XML or with set*() methods.

Xml

android:orientation="vertical"

code (ll is LinearLayout instance)

ll.setOrientation(VERTICAL);

Each View or ViewGroup can have its own set of attributes…but, some are very common

Attribute	Description
layout_width	specifies width of View or ViewGroup
layout_height	specifies height
layout_marginTop	extra space on top
layout_marginBottom	extra space on bottom side
layout_marginLeft	extra space on left side
layout_marginRight	extra space on right side
layout_gravity	how child views are positioned
layout_weight	how much extra space in layout should be allocated to View (only when in LinearLayout or TableView)
layout_x	x-coordinate
layout_y	y-coordinate

Another Option to get Complexity →What about Other Layouts

- **RelativeLayout** is good ---and ***can*** make your design EASIER

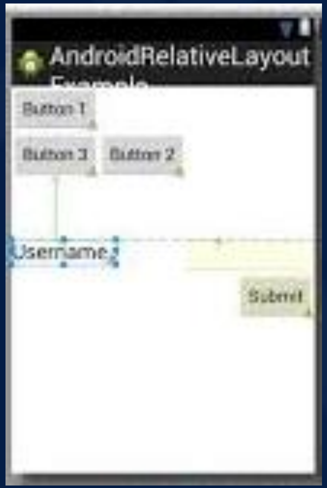
- ***Note: there is more than one way to use Layouts to create a look in an interface that is the same ---so, this in part is an art and in part how you think of things ---but, sometimes as we will see later some Layouts can be faster (especially when compared to nested layouts)***

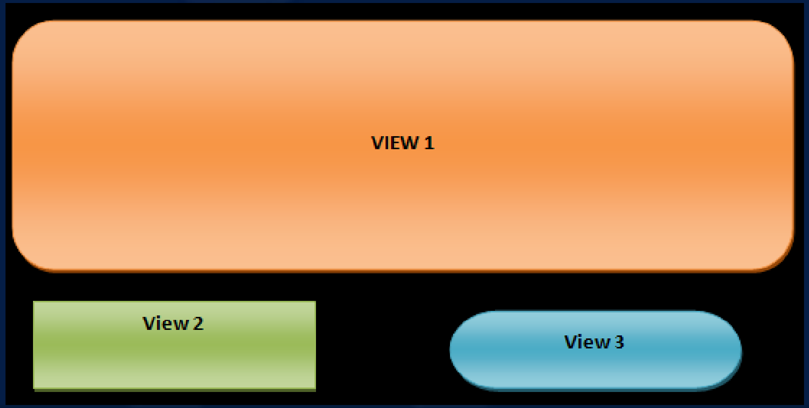
RelativeLayout

GOOD:

- Can give more complex interfaces
- Know what will look like on different sized devices
- Position relative to another position

CAUTION This is meant to be flat –meaning you don't want/need to nest RelativeLayouts in each other – sometimes may impact speed in rendering and some have reported problems.





RelativeLayout – how it works

Parameters in XML *(or can map to method calls in Java RelativeLayout class)*

- Position relative to Parent

android:layout_alignParentTop, android:layout_alignParentBottom, android:layout_alignParentLeft, android:layout_alignParentRight

VALUE = 'true' ---If "true", moves to that edge of Parent

android:layout_centerVertical

VALUE= "true" -- If "true", centers this child vertically within its parent.

- Position relative to another widget

android:layout_below, android:layout_above, android:layout_toLeftOf, android:layout_toRightOf

VALUE="resource ID of other widget" -- Positions the top edge of this view below/aboveof the view specified with a resource ID.

OR Positions the left edge of this view to the left/right of the view specified with a resource ID.

RelativeLayout – how it works

Example

```
<?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="match_parent"
  android:paddingLeft="16dp"
  android:paddingRight="16dp" >
  <EditText
    android:id="@+id/name"
    android:layout_width="match_pare
    android:layout_height="wrap_content"
    android:hint="@string/reminder" />
  <Spinner
    android:id="@+id/dates"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_below="@id/name"
    android:layout_alignParentLeft="true"
    android:layout_toLeftOf="@+id/times"
  <Spinner
    android:id="@id/times"
    android:layout_width="96dp"
    android:layout_height="wrap_content"
    android:layout_below="@id/name"
    android:layout_alignParentRight="true   />
  <Button
    android:layout_width="96dp"
    android:layout_height="wrap_content"
    android:layout_below="@id/times"
    android:layout_alignParentRight="
    android:text="@string/done" />
</RelativeLayout>
```

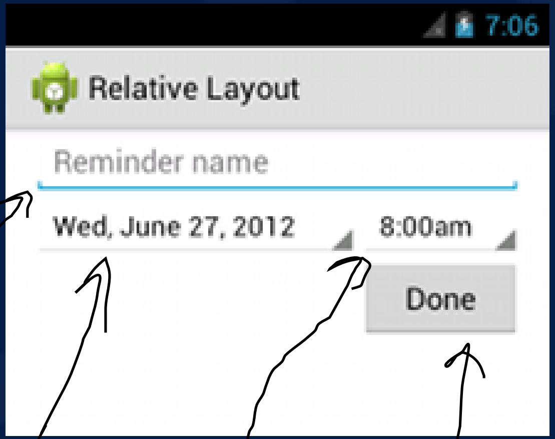
Says we have RelativeLayout that width and height match parent (which is the entire app screen)

1st View object in RelativeLayout will be at the top and is the EditText

2nd View object here is specified to be **below the 1st object** *EditText (id = name)* & **aligned to left of parent(app**) & **Left of** *the Button with id=times (see below)*

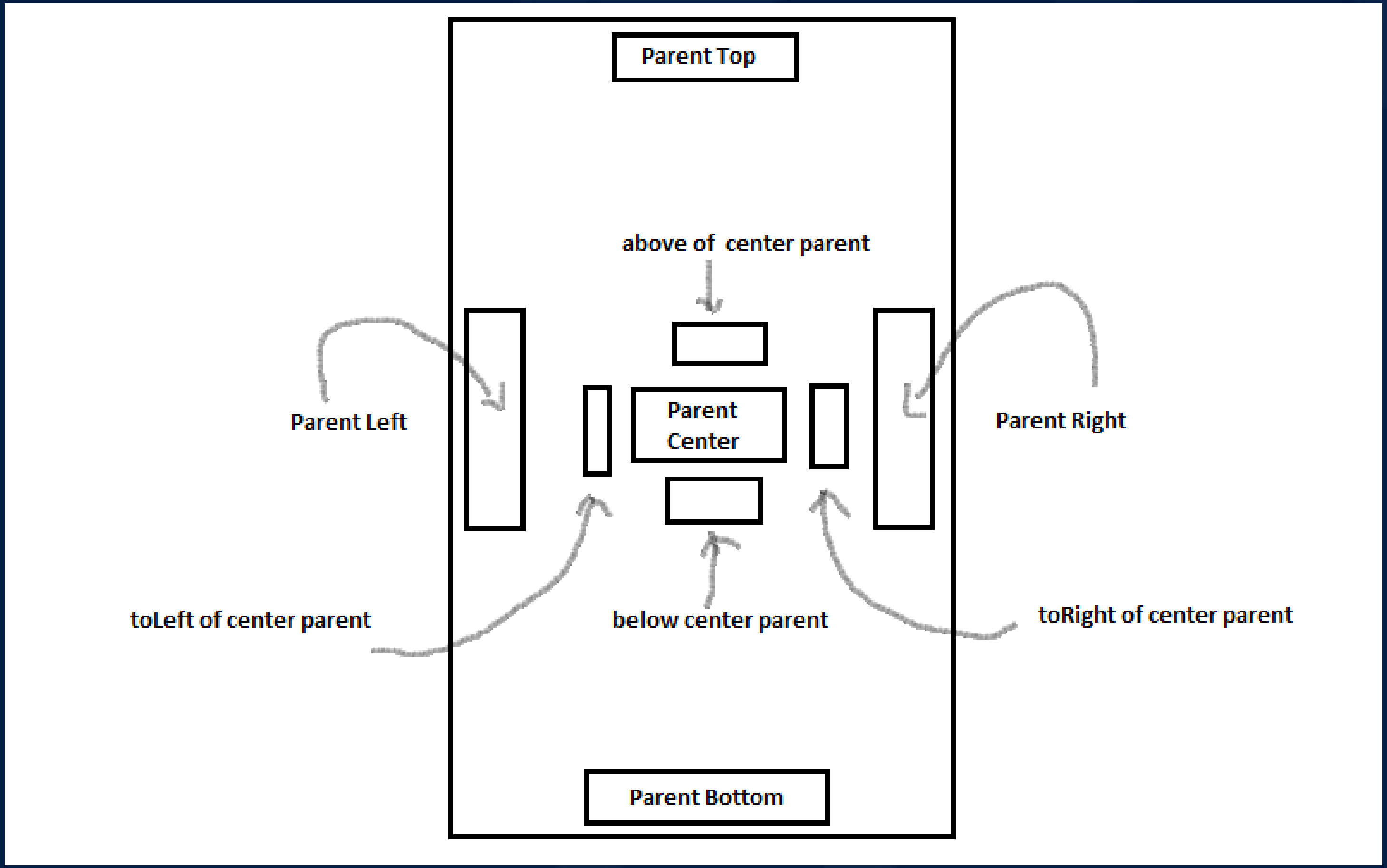
3rd View object here is specified to be **below the 1st object** *EditText (id = name)* & **aligned to left of parent(app**)

4th View object here is specified to be **below the 2nd object** *Spinner (id = times)* & **aligned to right of parent(app**)



More on RelativeLayout parameters

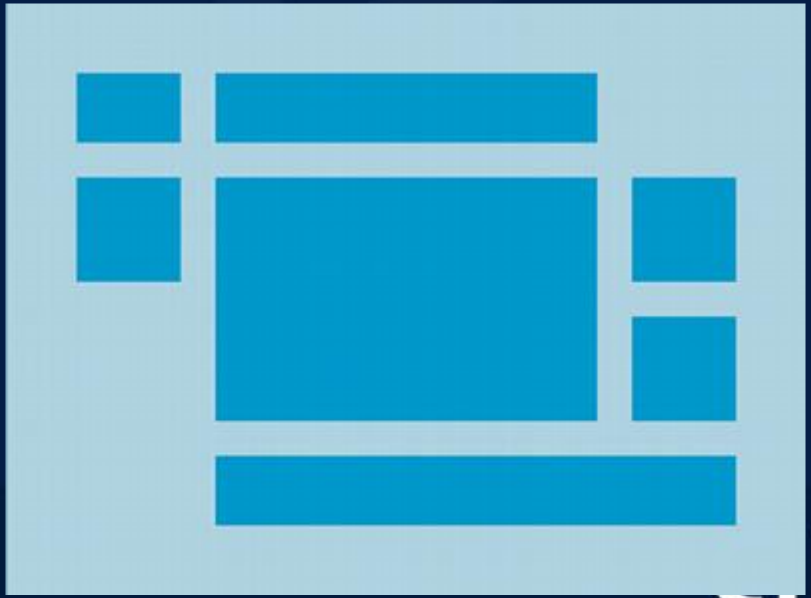
- Center Top Bottom of Parent



There are many other Layouts

- Look them up on Android Developer site
- They include: TableLayout (think a table), GridLayout, FrameLayout, and MORE!!

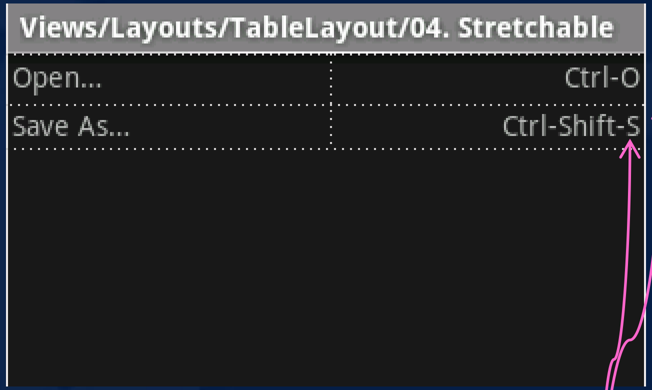
TableLayout



Read book and look at developer website to learn about others like TableLayout

TableLayout Example

```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:stretchColumns="1">
   <TableRow>
      <TextView
         android:text="@string/table_layout_4_open"
         android:padding="3dip" />
      <TextView
         android:text="@string/table_layout_4_open_shortcut"
         android:gravity="right"
         android:padding="3dip" />
   </TableRow>

   <TableRow>
      <TextView
         android:text="@string/table_layout_4_save"
         android:padding="3dip" />
      <TextView
         android:text="@string/table_layout_4_save_shortcut"
         android:gravity="right"
         android:padding="3dip" />
   </TableRow>
</TableLayout>
```

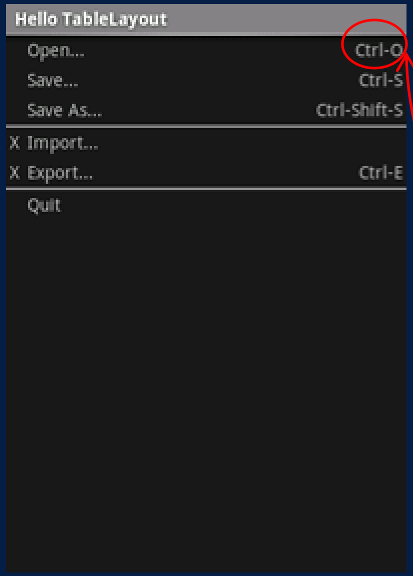

Views/Layouts/TableLayout/04. Stretchable

Open...	Ctrl-O
Save As...	Ctrl-Shift-S

This Table has 2 Rows

TableLayout example 2

- Here use gravity to move the 2nd item in row to the right



Hello TableLayout

```
Open...                Ctrl-O
Save...                Ctrl-S
Save As...       Ctrl-Shift-S
X Import...
X Export...            Ctrl-E
Quit
```

ONLY partial XML code

```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:stretchColumns="1">

   <TableRow>
      <TextView
         android:layout_column="1"
         android:text="Open..."
         android:padding="3dip" />
      <TextView
         android:text="Ctrl-O"
         android:gravity="right"
         android:padding="3dip" />
   </TableRow>

   <TableRow>
```

NOW CONTINUE ON FOR 2ND ROW

THANK YOU ☺