

CS378 - Mobile Computing

User Interface Basics

User Interface Elements

- View
 - Control
 - ViewGroup
 - Layout
 - Widget (Compound Control)
- Many pre built Views
 - Button, CheckBox, RadioButton
 - TextView, EditText, ListView
 - Can be customized by extending and overriding onDraw()

XML UI Configuration

- Layouts can specify UI elements (provided and custom)
- res/layout
- "Design by Declaration"

Layouts

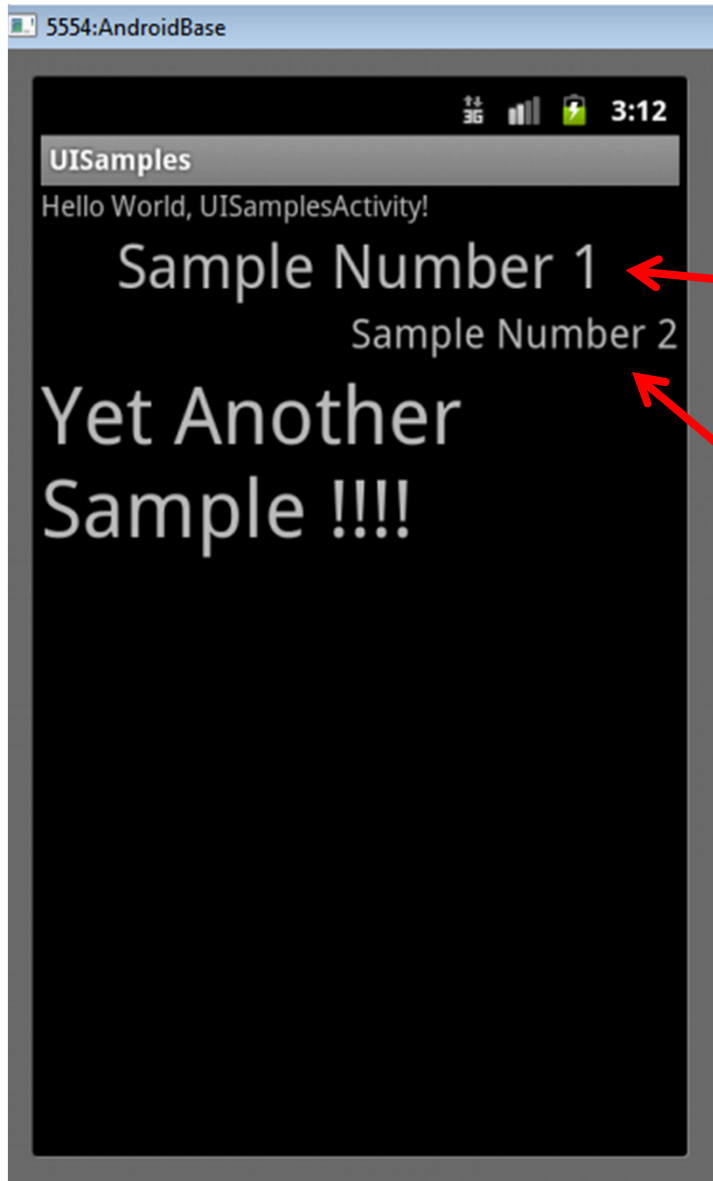
- Layouts are subclasses of ViewGroup
- FrameLayout
 - simplest type of layout object
 - fill with a single object (such as a picture) that can be switched in and out
 - child elements pinned to top left corner of screen and cannot be move
 - adding a new element / child draws over the last one

LinearLayout

- aligns child elements (such as buttons, edit text boxes, pictures, etc.) in a single direction
- orientation attribute defines direction:
 - android:orientation=*"vertical"*



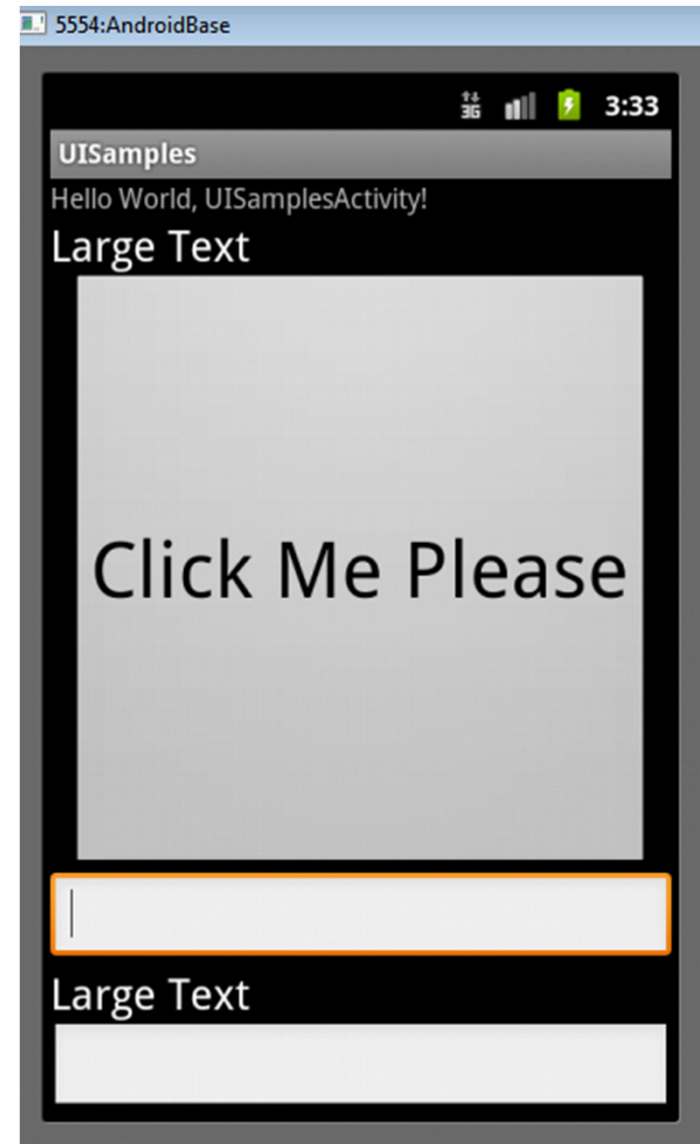
Gravity



- Child element's gravity attribute
– where to position in the outer container

Weight

- layout_weight attribute
 - "importance" of a view
 - default = 0
 - if set > 0 takes up more of parent space
- BTW, scale emulator Run -> Run Configurations -> target -> command line options "-scale 0.7"



Another Weight Examples

button and bottom
edit text weight of 2

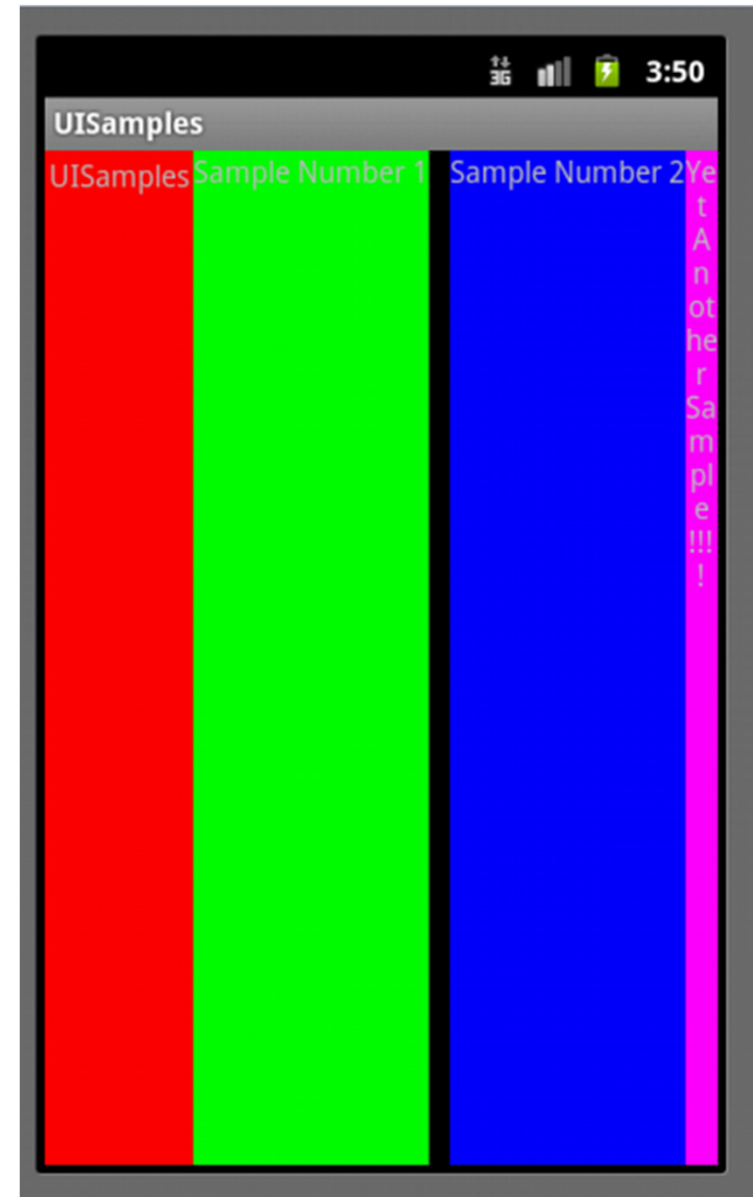


button weight 1 and
bottom edit text weight
of 2



LinearLayout - Horizontal Orientation

- padding
- background color
- margins



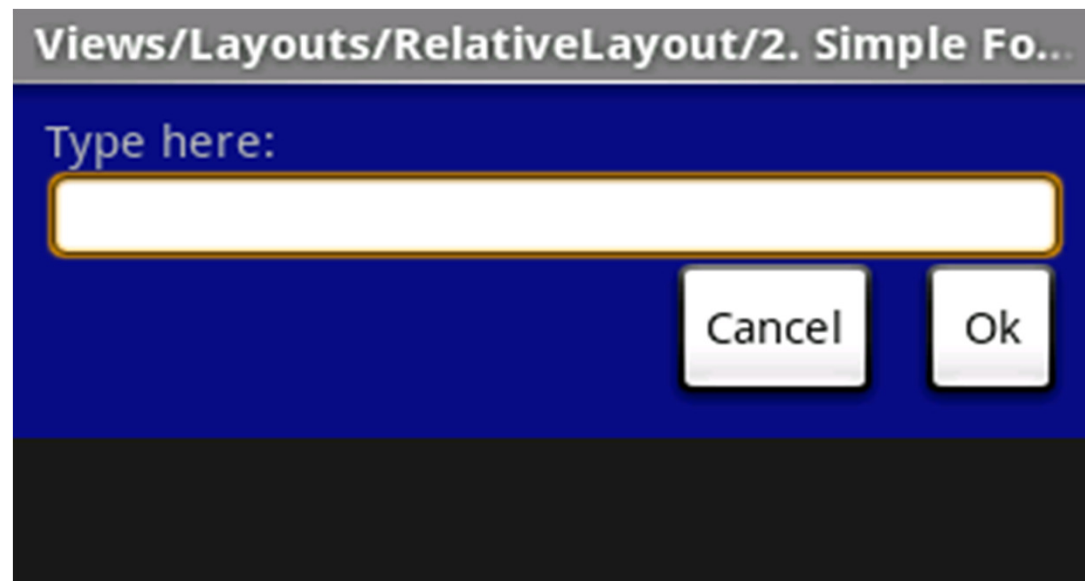
TableLayout

- rows and columns
- rows normally
TableRows
- TableRows contain
other elements such
as buttons, text, etc.



RelativeLayout

- children specify position relative to parent or to each other (specified by ID)
- First element listed is placed in "center"
- other elements placed based on position to other elements



RelativeLayout XML

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:background="@drawable/blue"
    android:padding="10px" >

    <TextView android:id="@+id/label"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Type here:" />

    <EditText android:id="@+id/entry"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:background="@android:drawable/editbox_background"
        android:layout_below="@id/label" />
```

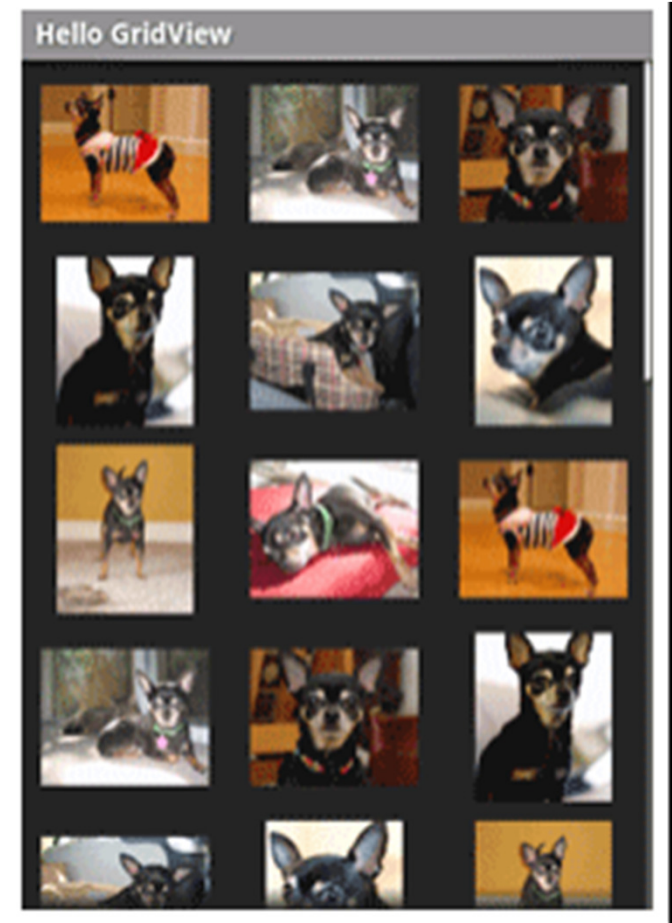
RelativeLayout XML

```
<Button android:id="@+id/ok"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/entry"
        android:layout_alignParentRight="true"
        android:layout_marginLeft="10px"
        android:text="OK" />

<Button android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_toLeftOf="@id/ok"
        android:layout_alignTop="@id/ok"
        android:text="Cancel" />
</RelativeLayout>
```

Other Layouts - GridView

- Two Dimensional Scrollable Grid
- Items inserted into layout via a ListAdapter



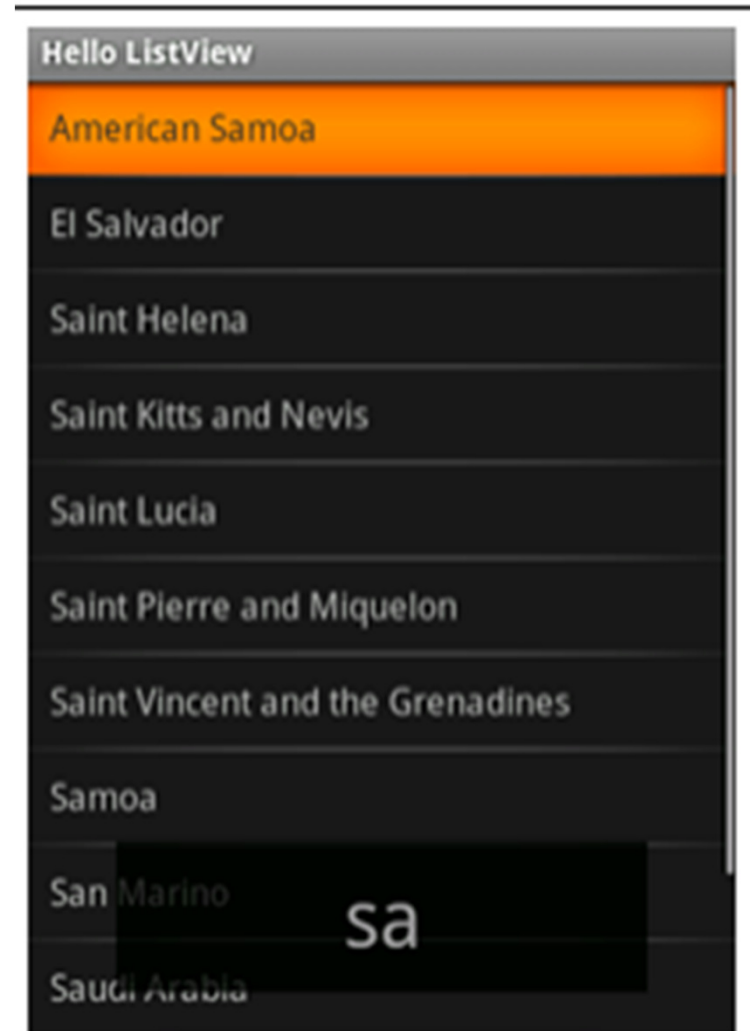
Other Layouts - TabLayout

- Uses a TabHost and TabWidget
- Swap between views in same activity or switch between different activities



Other Layouts - ListView

- Creates a list of scrollable items
- Items added via a ListAdapter as in GridView

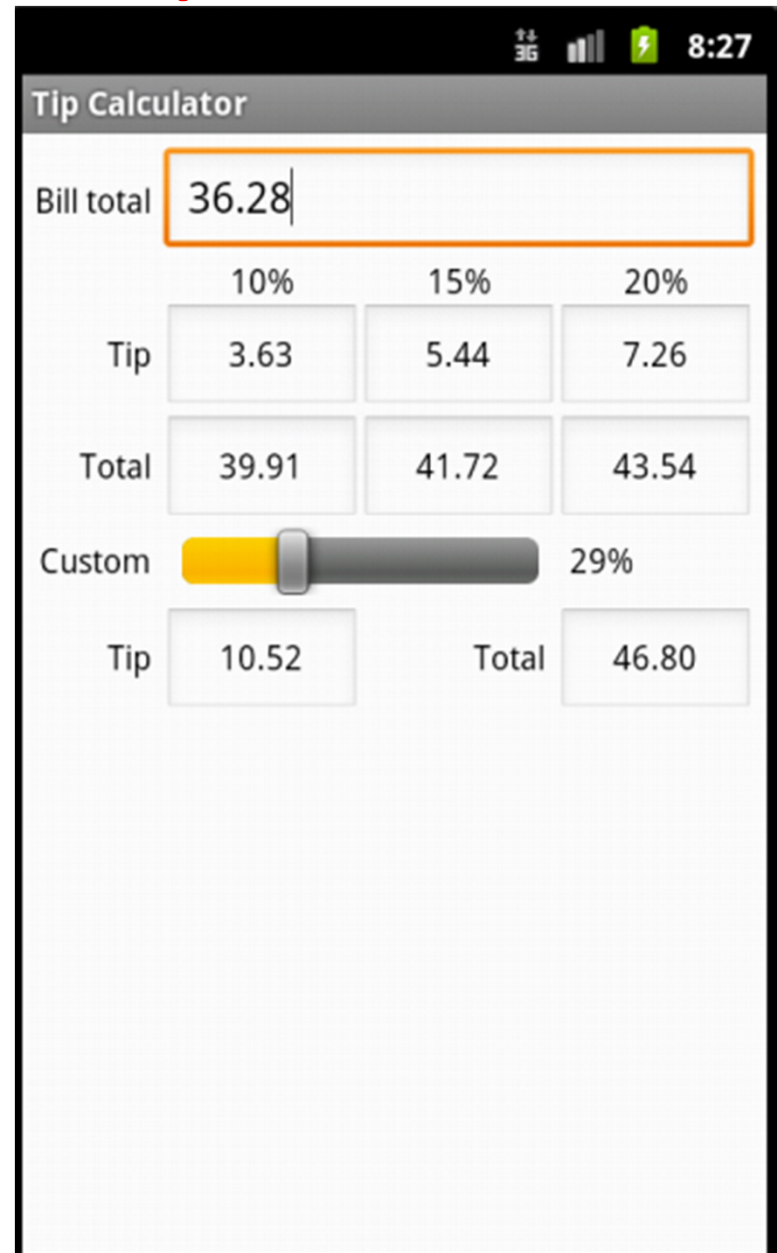


Other Views - Layouts

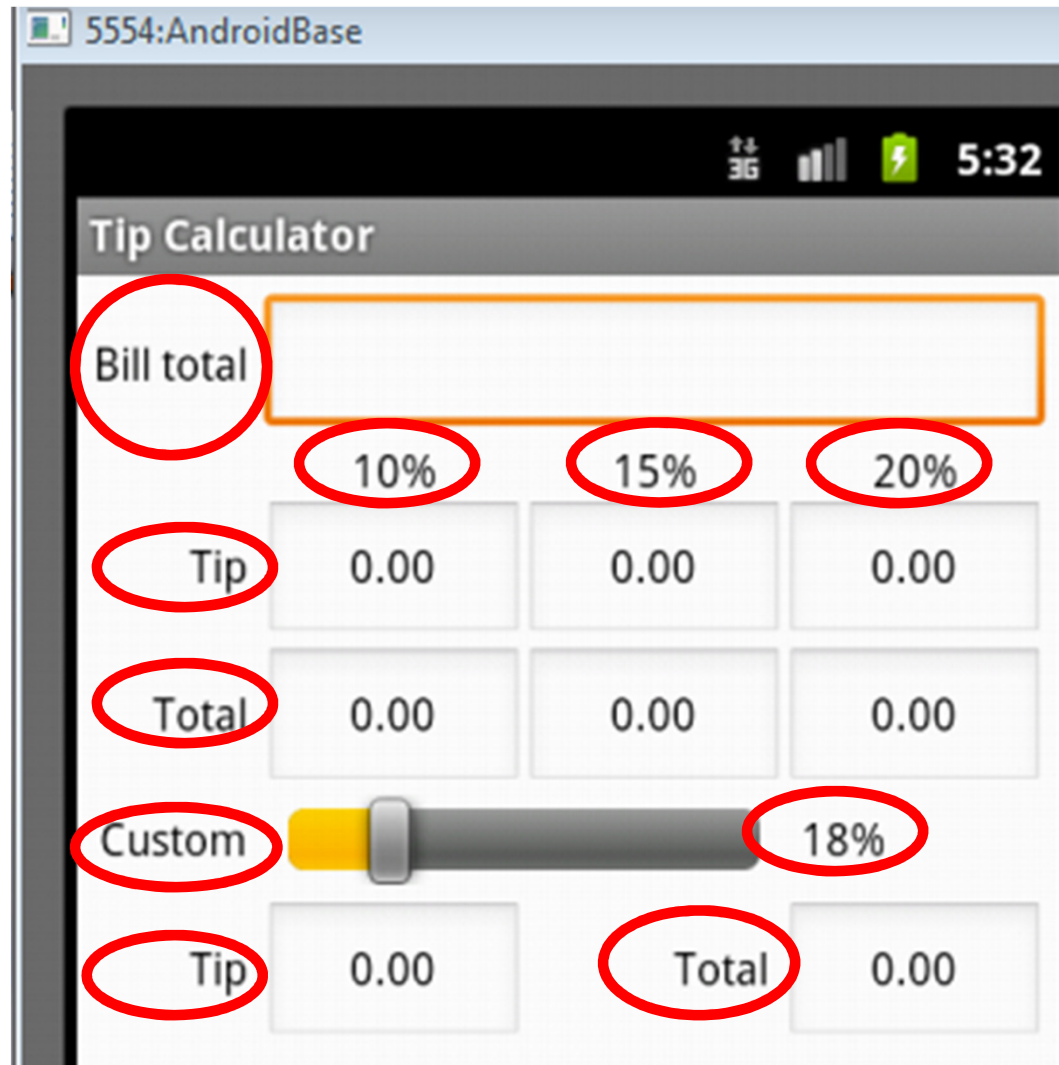
- Gallery
 - horizontal scrolling display of images from a list
- SurfaceView
 - provide access to a "drawing" surface.
Intended to draw pixels, not display other views / widgets

Concrete Example

- Tip Calculator
- What kind of layout to use?
- Widgets:
 - TextView
 - EditText
 - SeekBar



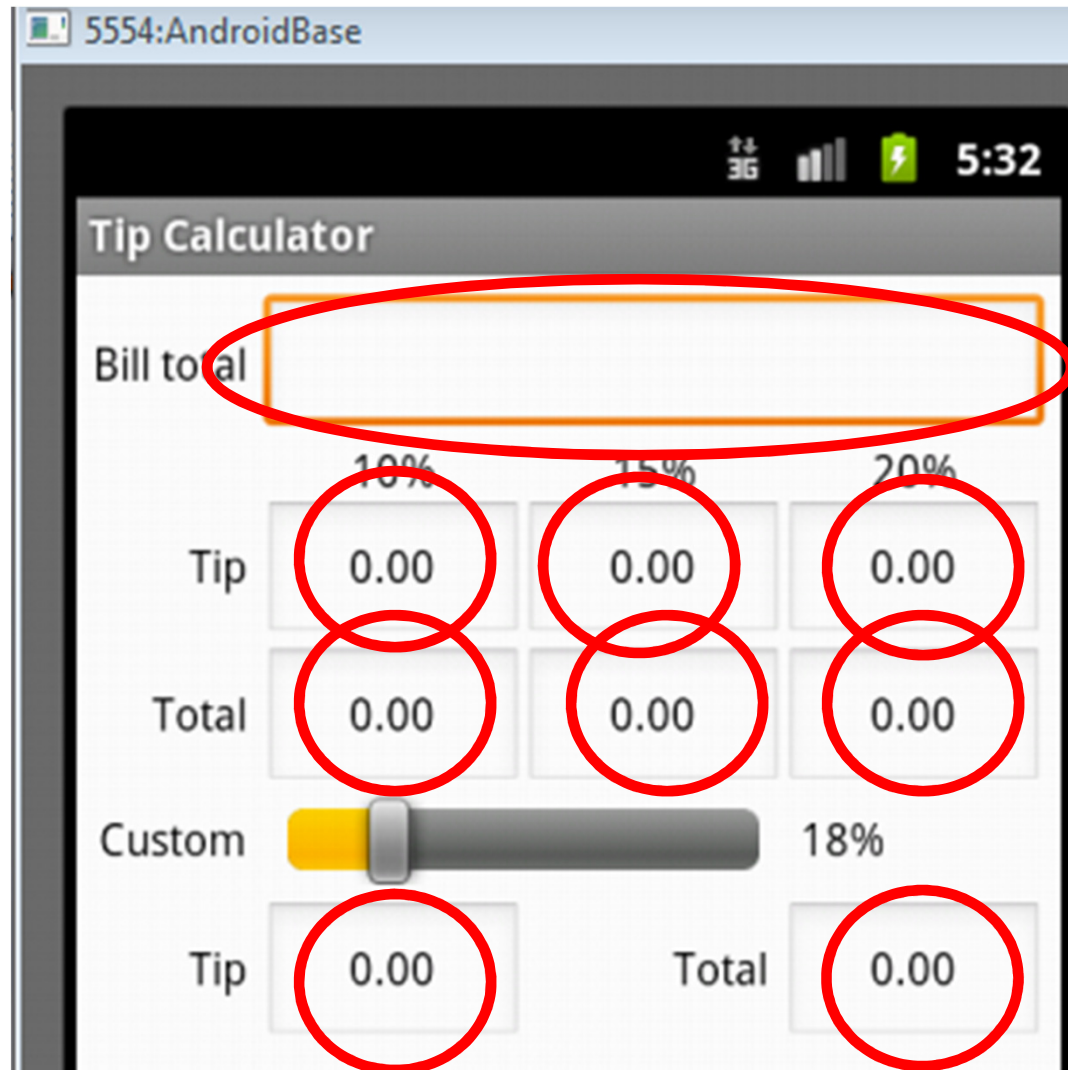
TextViews



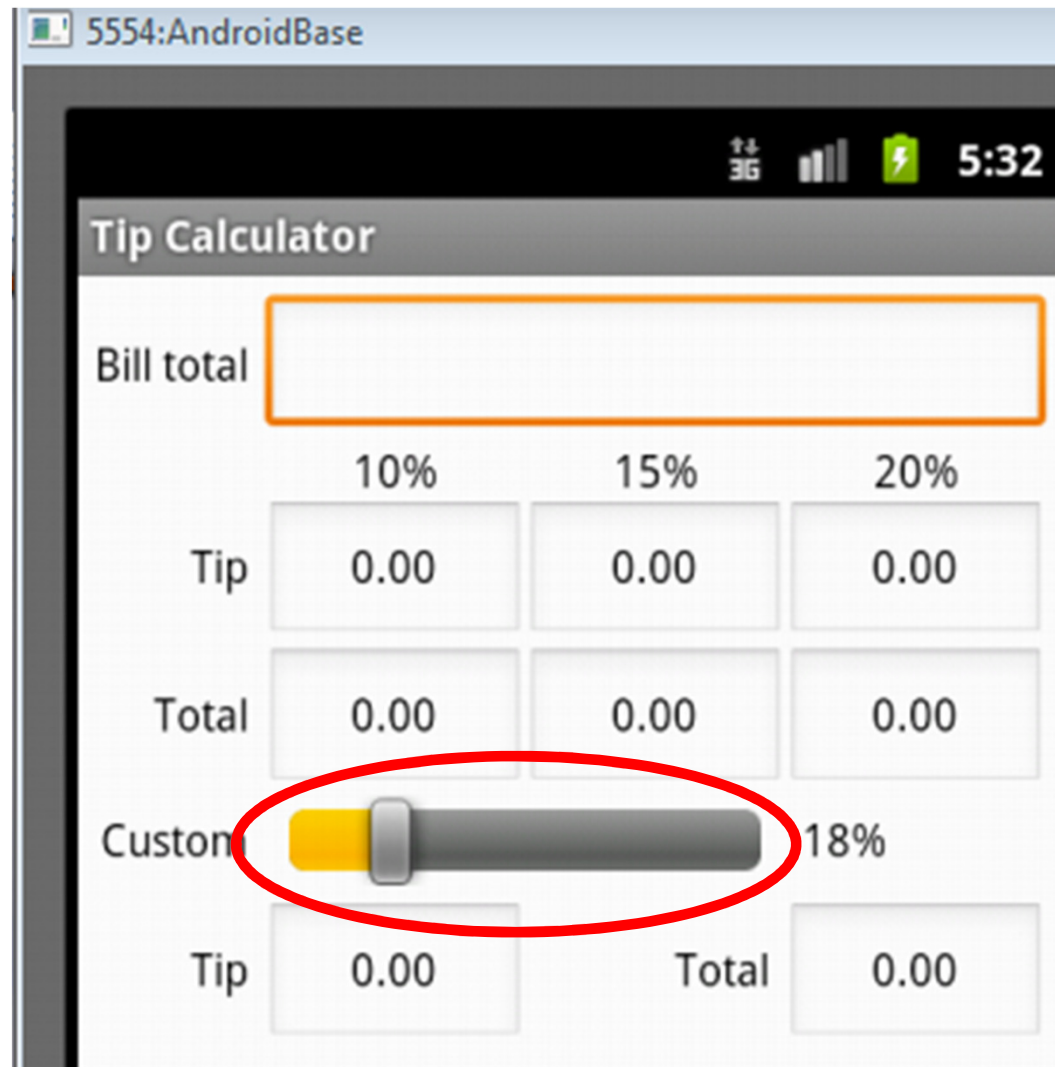
EditText

All but top
EditText are
uneditable

Alternative?
TextViews?



SeekBar



Layout

- TableLayout

row 0 → Bill total

row 1 → 10% 15% 20%

row 2 → Tip 0.00 0.00 0.00

row 3 → Total 0.00 0.00 0.00

row 4 → Custom 18%

row 5 → Tip 0.00 Total 0.00

Layout Attributes

```
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/tableLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#FFF"
    android:padding="5dp"
    android:stretchColumns="1,2,3" >
```

- android:background
 - #RGB, #ARGB, #RRGGBB, #AARRGGBB
 - can place colors in res/values/colors.xml

Color Resources

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <resources>
3     <color name="Cardinal">#C41E3A</color>
4     <color name="White">#FFFFFF</color>
5 </resources>
```

```
android:layout_width="match_parent"
android:layout_height="match_parent"
android:background="@color/White"
android:padding="5dp"
android:stretchColumns="1 2 3"
```

- Good Resource / W3C colors
 - <http://tinyurl.com/6py9huk>

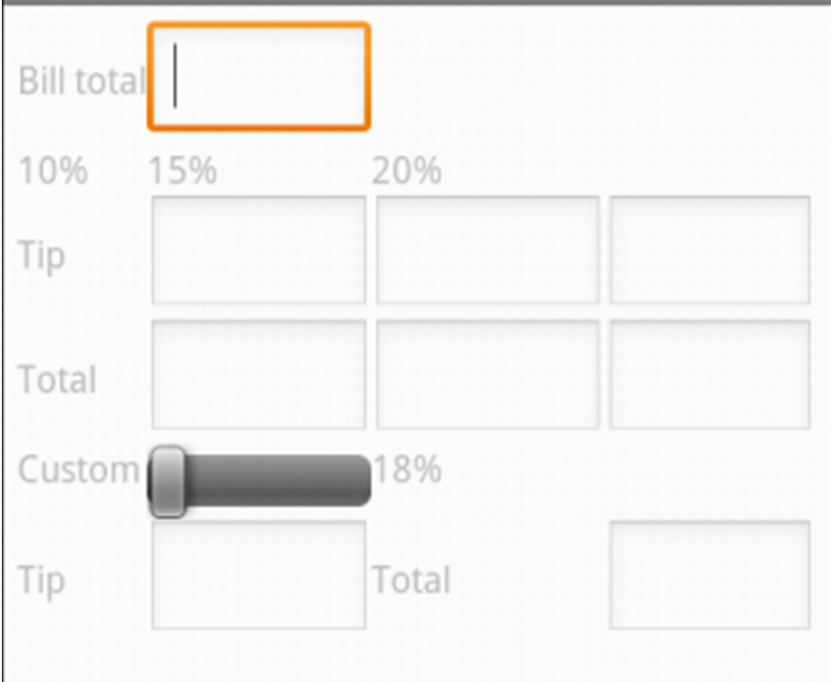
StretchColumns

```
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/tableLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#FFF"
    android:padding="5dp"
    android:stretchColumns="1, 2, 3" >
```

- columns 0 indexed
- columns 1, 2, 3 stretch to fill layout width
- column 0 wide as widest element, plus any padding for that element

Initial UI

- Done via some Drag and Drop, Outline view, and editing XML
- Demo outline view
 - properties



The image shows a user interface for a bill calculator. It features a 'Bill total' input field with an orange border. Below it are three radio buttons for tip percentages: '10%', '15%', and '20%'. The '15%' option is selected. There are three input fields for the tip amount, corresponding to the 10%, 15%, and 20% options. Below these is a 'Total' label and three more input fields. A 'Custom' option is also present, with a slider set to 18% and an input field for the custom tip amount. At the bottom, there are labels for 'Tip' and 'Total' next to input fields.

Option	Tip Amount	Total
10%		
15%		
20%		

Custom: 18%
Tip: Total:

Changes to UI

- Outline multiple select properties
 - all TextViews' textColor set to black #000000
- change column for %DD labels

```
android:text="10%"  
android:layout_column="1"  
android:textColor="#000000" />
```

- use center gravity for components

The screenshot shows a bill calculator app interface. At the top, there is a label 'Bill total' followed by a text input field. Below this, there are three columns for tip percentages: '10%', '15%', and '20%'. Each column has a corresponding 'Tip' label and a text input field. Below the '10%' column, there is a 'Total' label and a text input field. At the bottom, there is a 'Custom' label, a slider control set to '18%', and a 'Total' label with a text input field. The input fields are highlighted with orange borders.

Changes to UI

- change bill total and seekbar to span more columns
- gravity and padding for text in column 0
- align text with seekBar
- set seekBar progress to 18
- set seekBar focusable to false - keep keyboard on screen

```
<EditText  
    android:id="@+id/billEditText"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_span="3"  
    android:inputType="numberDecimal" >
```

Changes to UI

- Prevent Editing in EditText
 - focusable, long clickable, and cursor visible properties to false
- Set text in EditText to 0.00
- Change weights to 1 to spread out

The screenshot shows the 'MyTipCalc' app interface. At the top is a title bar with the text 'MyTipCalc'. Below it is a text input field labeled 'Bill total' with an orange border. Underneath the input field are three buttons labeled '10%', '15%', and '20%'. Below these buttons are two rows of buttons: 'Tip' and 'Total'. Each of these rows has three buttons corresponding to the 10%, 15%, and 20% percentages, all displaying '0.00'. Below the 10% and 15% buttons is a 'Custom' label followed by a horizontal slider. The slider has a yellow knob and is positioned at 18%. To the right of the slider is a label '18%'. At the bottom, there are two more buttons: 'Tip' displaying '0.00' and 'Total' displaying '0.00'.

Functionality

- onCreate instance variables assigned to components found via ids
- update standard percents:

```
private void updateStandard()
{
    for(int i = 0; i < NUM_PERCENTS - 1; i++) {
        double tip = currentBillTotal * tipPercents[i];
        double total = currentBillTotal + tip;
        tipEditTexts[i].setText(String.format("%.02f", tip));
        totalEditTexts[i].setText(String.format("%.02f", total));
    }
}

// end method updateStandard
```

Functionality - Saving State

- onSaveInstanceState
 - save BillTotal and CustomPercent to the Bundle
 - check for these in onCreate

```
// save values of billEditText and customSeekBar
@Override
protected void onSaveInstanceState(Bundle outState)
{
    super.onSaveInstanceState(outState);

    outState.putDouble(BILL_TOTAL, currentBillTotal);
    outState.putInt(CUSTOM_PERCENT, (int) (tipPercents[CUSTOM_INDEX] * 100));
} // end method onSaveInstanceState
```

Functionality Responding to SeekBar

- customSeekBarListener instance variable
- Of type OnSeekBarChangeListener

public static interface

SeekBar.OnSeekBarChangeListener

Public Methods	
abstract void	<code>onProgressChanged</code> (SeekBar seekBar, int progress, boolean fromUser) Notification that the progress level has changed.
abstract void	<code>onStartTrackingTouch</code> (SeekBar seekBar) Notification that the user has started a touch gesture.
abstract void	<code>onStopTrackingTouch</code> (SeekBar seekBar) Notification that the user has finished a touch gesture.

Create an Anonymous Inner Class

- Class notified when seek bar changed and program updates custom tip and total amount
- must register with the seekBar instance variable in onCreate!

```
// called when the user changes the position of SeekBar
private OnSeekBarChangeListener customSeekBarListener =
    new OnSeekBarChangeListener()
{
    // update tipPercents[CUSTOM_INDEX], then call updateCustom
    @Override
    public void onProgressChanged(SeekBar seekBar, int progress,
        boolean fromUser)
    {
        // sets tipPercents[CUSTOM_INDEX] to position of the Seek
        tipPercents[CUSTOM_INDEX] = seekBar.getProgress();
        updateCustom(); // update EditTexts for custom tip and to
    }
}
```

Functionality - Total EditText

```
public interface  
TextWatcher
```

Public Methods	
abstract void	<code>afterTextChanged (Editable s)</code> This method is called to notify you that, somewhere within <code>s</code> , the text has been changed.
abstract void	<code>beforeTextChanged (CharSequence s, int start, int count, int after)</code> This method is called to notify you that, within <code>s</code> , the <code>count</code> characters beginning at <code>start</code> are about to be replaced by new text with length <code>after</code> .
abstract void	<code>onTextChanged (CharSequence s, int start, int before, int count)</code> This method is called to notify you that, within <code>s</code> , the <code>count</code> characters beginning at <code>start</code> have just replaced old text that had length <code>before</code> .

- Another anonymous inner class
- implement `onTextChanged` to convert `s` to double and call update methods
- register with EditText for total in `onCreate()`!