CS 580 Client-Server Programming
Fall Semester, 2012
Doc 10 Screen Sizes, Layouts, Dialogs
Sept 27, 2012

Screen Sizes

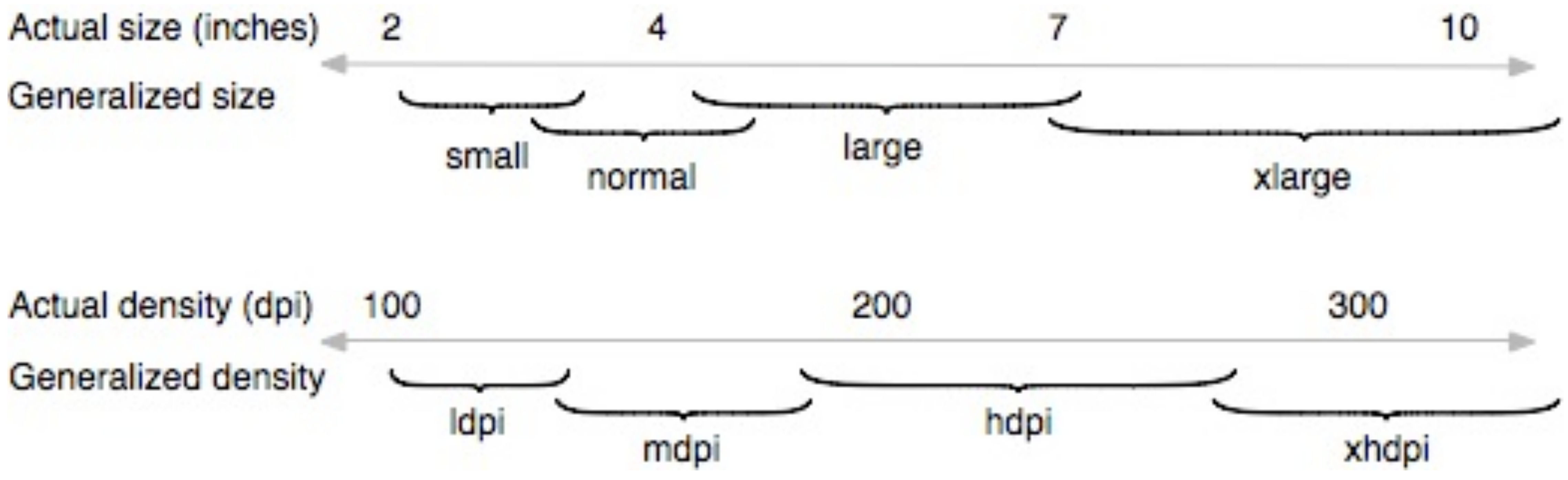
Multiple Screen Sizes (Chapter 25)

Pre Android 3.2

Screen Sizes - small, normal, large, and xlarge

xlarge screens are at least 960dp x 720dp
large screens are at least 640dp x 480dp
normal screens are at least 470dp x 320dp
small screens are at least 426dp x 320dp

Pixel Density - ldpi (low), mdpi, hdpi (high), xhdpi, tvdpi



Source:http://developer.android.com/guide/practices/screens_support.html

Supporting Multiple Screen Sizes

manifest file

- Can declare which sizes/densities the app supports

layouts & resources

- Different layout or resource files for different sizes/densit

Layouts & Resources

res/layout-large-port-mdpi-qwerty/main.xml
res/layout-normal-land-mdpi-nokeys/main.xml
res/layout-small/main.xml
res/layout-land/main.xml

File with same name in each directory

Android will pick the one that matches current situation

Manifest File

```
<supports-screens
        android:largeScreens="true"
        android:normalScreens="true"
        android:smallScreens="true"
        android:anyDensity="true"
    />
```

Options

```
android:resizeable=["true"| "false"]
android:smallScreens=["true" | "false"]
android:normalScreens=["true" | "false"]
android:largeScreens=["true" | "false"]
android:xlargeScreens=["true" | "false"]
android:anyDensity=["true" | "false"]
android:requiresSmallestWidthDp="integer"
android:compatibleWidthLimitDp="integer"
android:largestWidthLimitDp="integer"/>
```

Screen Sizes - Android 3.2+

res/layout-sw800dp-port
res/layout-sw800dp-land

Smallest Width (sw600dp)
- Smallest Width
- Does not change with device rotation

Available screen width (w720dp)
- Does change with device

Available screen height (h780dp)
- Does change with device

Directories allowed in res

animator/	XML files that define property animations.
anim/	XML files that define tween animations
color/	XML files that define a state list of colors
drawable/	Bitmap files
layout/	
menu/	XML files that define application menus
raw/	Arbitrary files to save in their raw form
values/	XML files that contain simple values
xml/	Arbitrary XML files

Qualifiers

MCC and MNC	
Language and region	en, fr, en-rUS
smallestWidth	sw<N>dp
Available width	w<N>dp
Available height	h<N>dp
Screen size	small, normal, large, xlarge
Screen aspect	long, notlong
Screen orientation	port, land
Dock mode	car, desk
Night mode	night, notnight
Screen pixel density (dpi)	ldpi, mdpi, hdpi, xhdpi, nodpi, tvdpi
Touchscreen type	notouch, stylus, finger
Keyboard availability	keysexposed, keyshidden, keyssoft
Primary text input method	nokeys, qwerty, 12key
Navigation key availability	navexposed, navhidden
Primary non-touch navigation method	nonav, dpad, trackball,wheel
Platform Version (API level)	V3, v4, etc

Source:http://developer.android.com/guide/topics/resources/providing-resources.html#AlternativeResources

Quantifier Order

Quantifiers must be used in order they are listed on previous slide

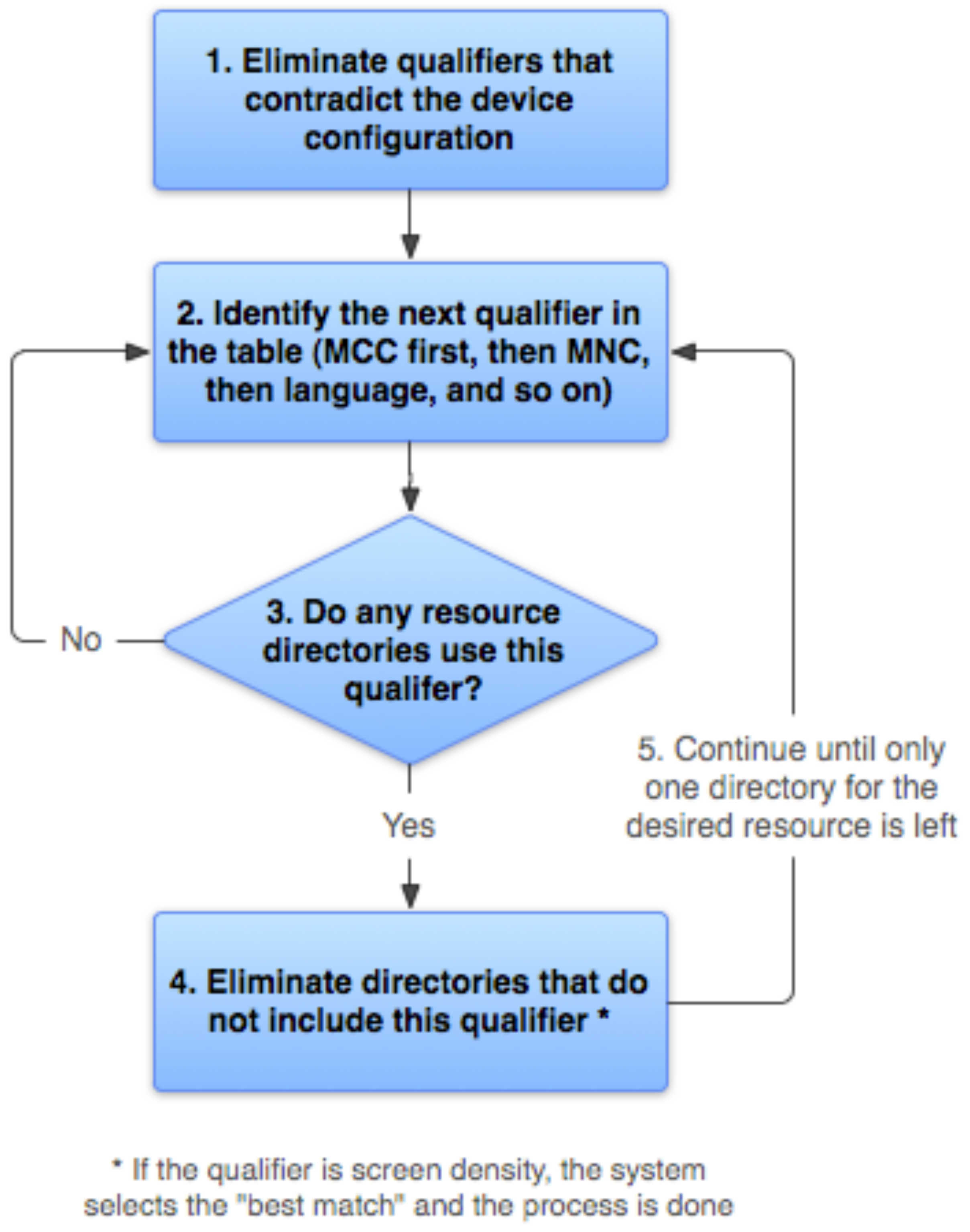
Legal

```
res/layout-large-port-mdpi-qwerty
```

Illegal

```
res/layout-large-mdpi-port-qwerty
```

Quantifier Match


1. Eliminate qualifiers that contradict the device configuration
2. Identify the next qualifier in the table (MCC first, then MNC, then language, and so on)
3. Do any resource directories use this qualifer?
No
Yes
4. Eliminate directories that do not include this qualifier *
5. Continue until only one directory for the desired resource is left
* If the qualifier is screen density, the system selects the "best match" and the process is done

Device

Locale = en-GB
Screen orientation = port
Screen pixel density = hdpi
Touchscreen type = notouch
Primary text input method = 12key

Resource Directories

drawable/
drawable-en/
drawable-fr-rCA/
drawable-en-port/
drawable-en-notouch-12key/
drawable-port-ldpi/
drawable-port-notouch-12key/

But this does not seem to work in all cases.

Layouts

Containers - LinearLayout

Important Properties/Concepts

- Orientation
- Fill Model
- Weight
- Gravity
- Padding

Orientation

android:orientation

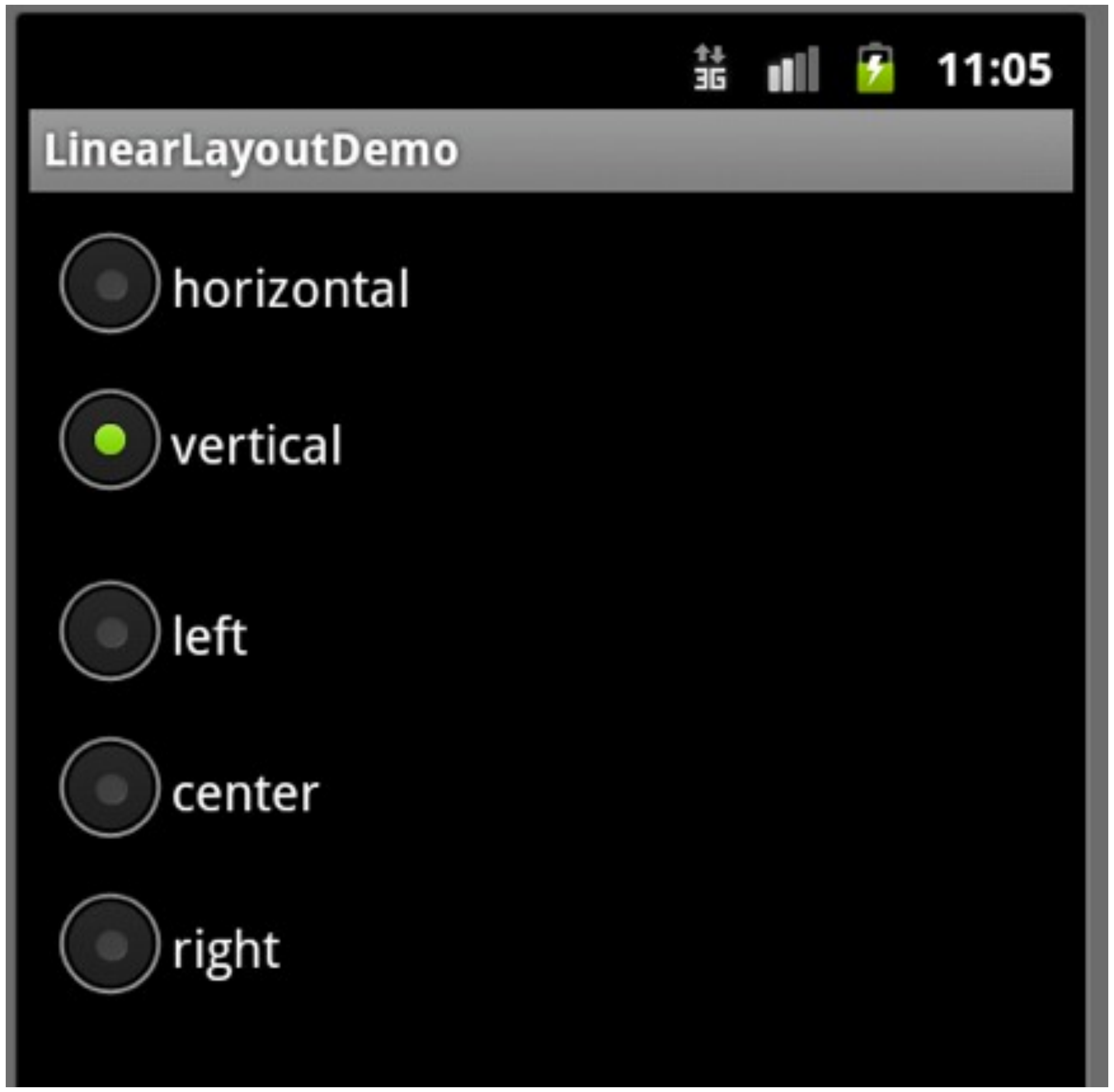
horizontal
- view is a row

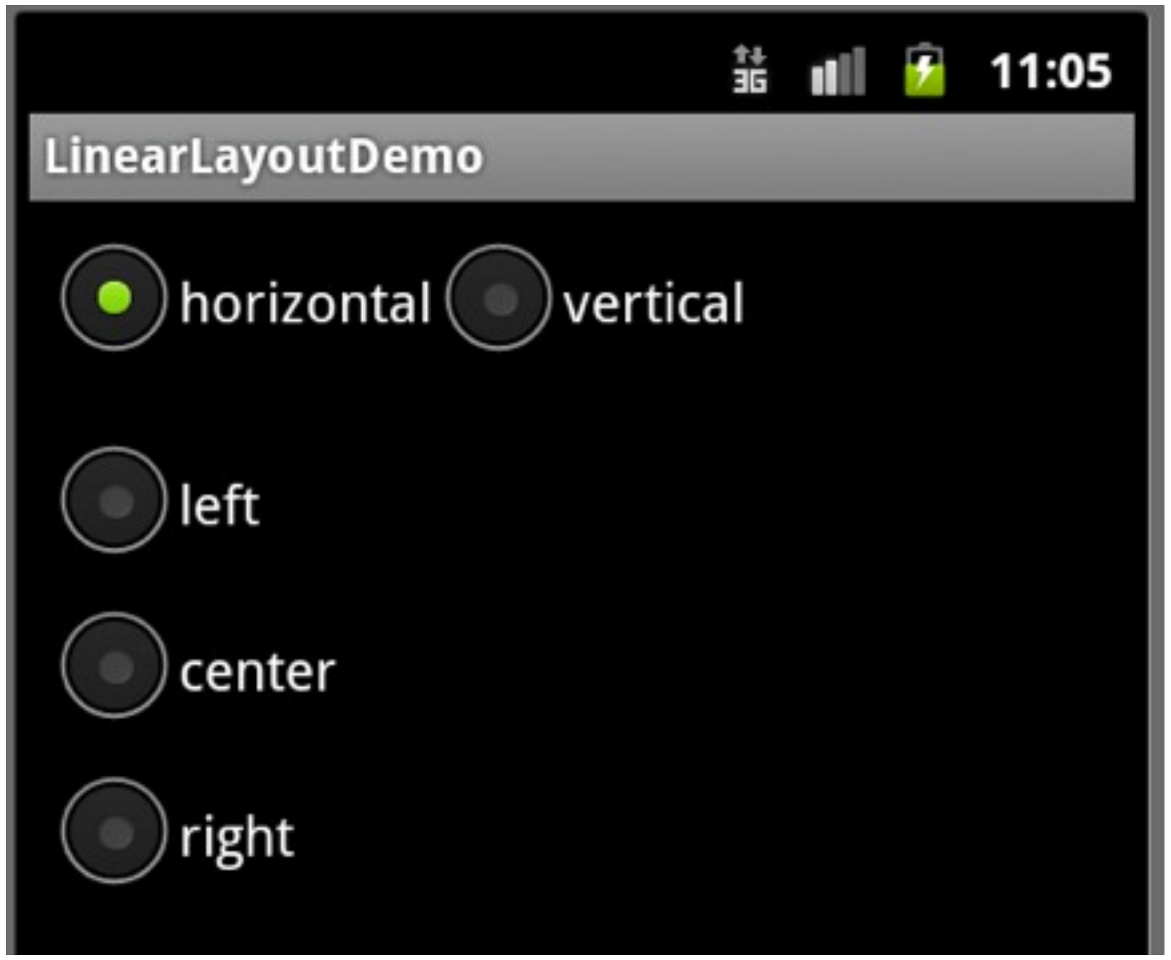
vertical
- view is a column

Change at runtime

```
setOrientation(LinearLayout.VERTICAL);

setOrientation(LinearLayout.HORIZONTAL);
```

Example


11:05
LinearLayoutDemo
horizontal
vertical
left
center
right


11:05
LinearLayoutDemo
horizontal
vertical
left
center
right

Gravity

android:layout_gravity
setGravity()

How do the subviews line up

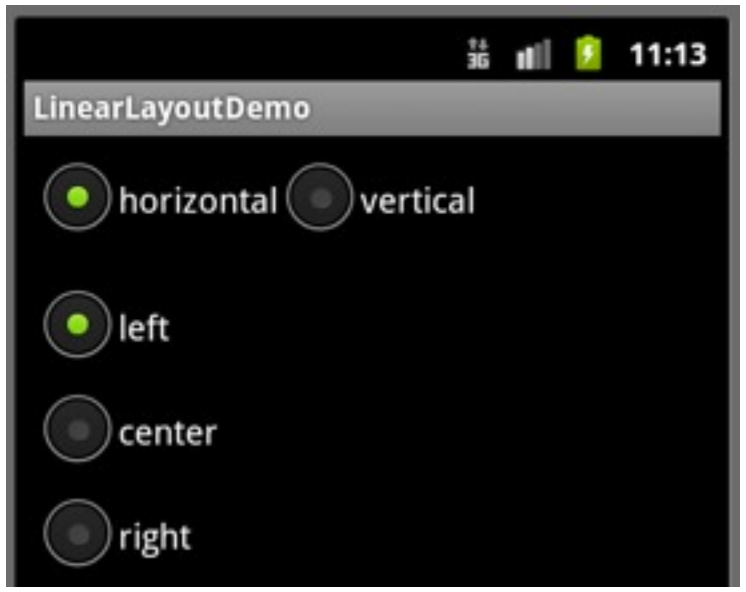
Values can be combined

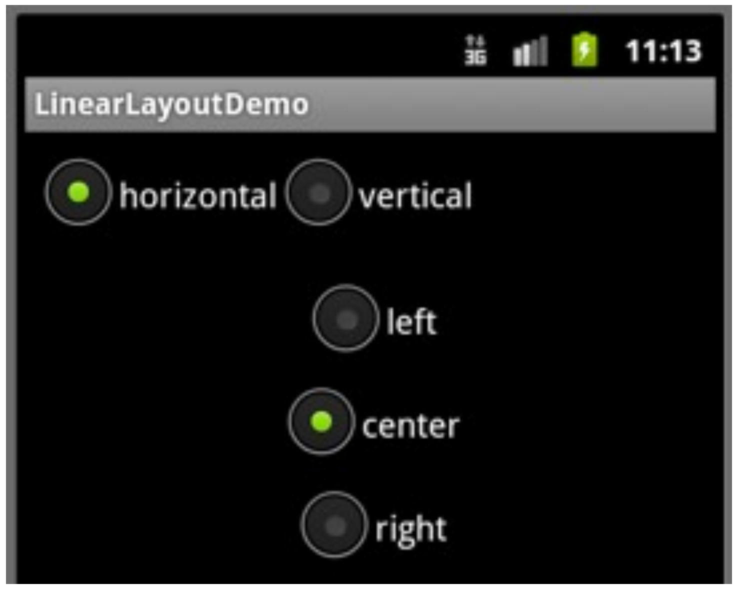
Values

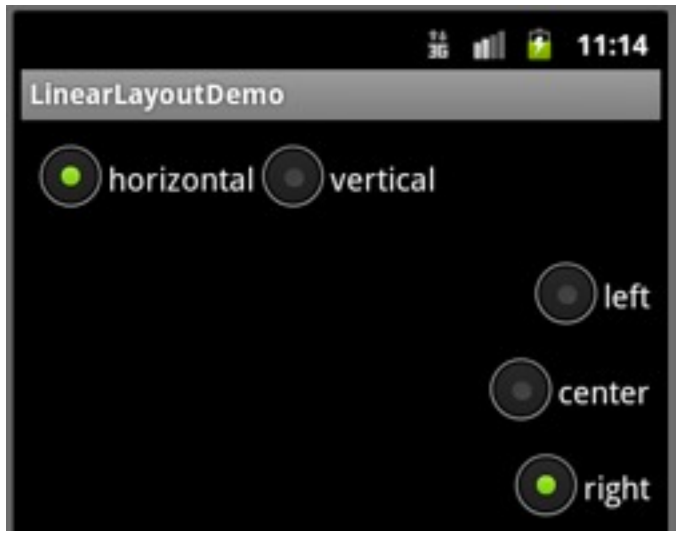
- top
- bottom
- left
- right
- center_vertical
- fill_vertical
- center_horizontal
- fill_horizontal
- center
- fill
- clip_vertical
- clip_horizontal
- start
- end

See http://developer.android.com/reference/android/widget/TextView.html#attr_android:gravity for descriptions of each.

Sample


11:13
LinearLayoutDemo
horizontal
vertical
left
center
right


11:13
LinearLayoutDemo
horizontal
vertical
left
center
right


11:14
LinearLayoutDemo
horizontal
vertical
left
center
right

Layout for examlpe

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:orientation="vertical"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        >
        <RadioGroup android:id="@+id/orientation"
                android:orientation="horizontal"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:padding="5dip">
                <RadioButton
                        android:id="@+id/horizontal"
                        android:text="horizontal" />
                <RadioButton
                        android:id="@+id/vertical"
                        android:text="vertical" />
        </RadioGroup>
```

```
        <RadioGroup android:id="@+id/gravity"
                android:orientation="vertical"
                android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:padding="5dip">
                <RadioButton
                        android:id="@+id/left"
                        android:text="left" />
                <RadioButton
                        android:id="@+id/center"
                        android:text="center" />
                <RadioButton
                        android:id="@+id/right"
                        android:text="right" />
        </RadioGroup>
</LinearLayout>
```

Activity source

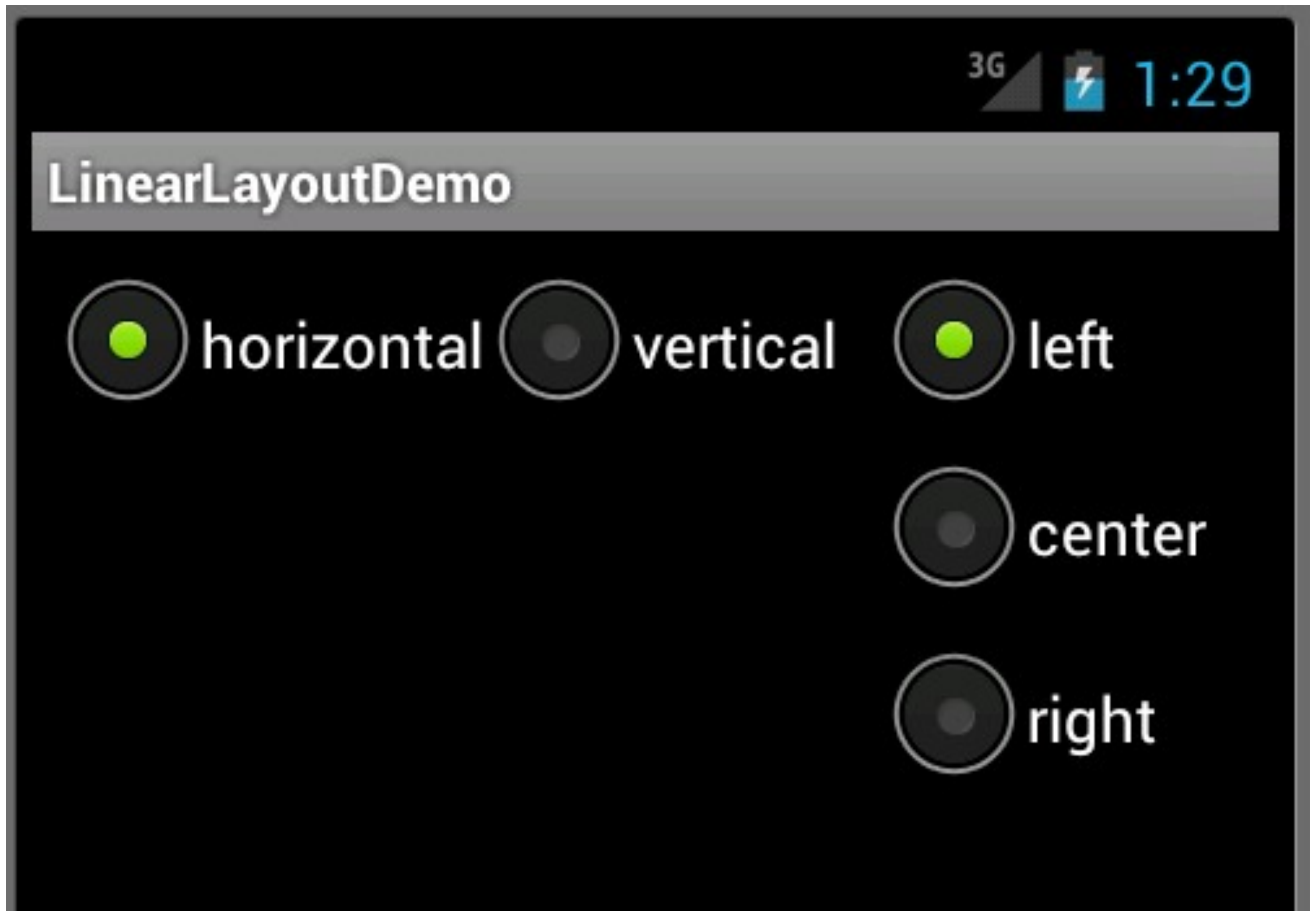
```
public class LinearLayoutDemo extends Activity
    implements RadioGroup.OnCheckedChangeListener {
    RadioGroup orientation;
    RadioGroup gravity;

    @Override
    public void onCreate(Bundle icicle) {
        super.onCreate(icicle);
        setContentView(R.layout.main);

        orientation=(RadioGroup)findViewById(R.id.orientation);
        orientation.setOnCheckedChangeListener(this);
        gravity=(RadioGroup)findViewById(R.id.gravity);
        gravity.setOnCheckedChangeListener(this);
    }
```

Source: Beginning Android 4, Grant Allen, Chapter 10

Activity source

```
    public void onCheckedChanged(RadioGroup group, int checkedId) {
        switch (checkedId) {
            case R.id.horizontal:
                orientation.setOrientation(LinearLayout.HORIZONTAL);
                break;
            case R.id.vertical:
                orientation.setOrientation(LinearLayout.VERTICAL);
                break;
            case R.id.left:
                gravity.setGravity(Gravity.LEFT);
                break;
            case R.id.center:
                gravity.setGravity(Gravity.CENTER_HORIZONTAL);
                break;
            case R.id.right:
                gravity.setGravity(Gravity.RIGHT);
                break;
        }
    }
}
```

Setting layout orientation



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

Fill Model

subviews supply
android:layout_width
android:layout_height

Specify

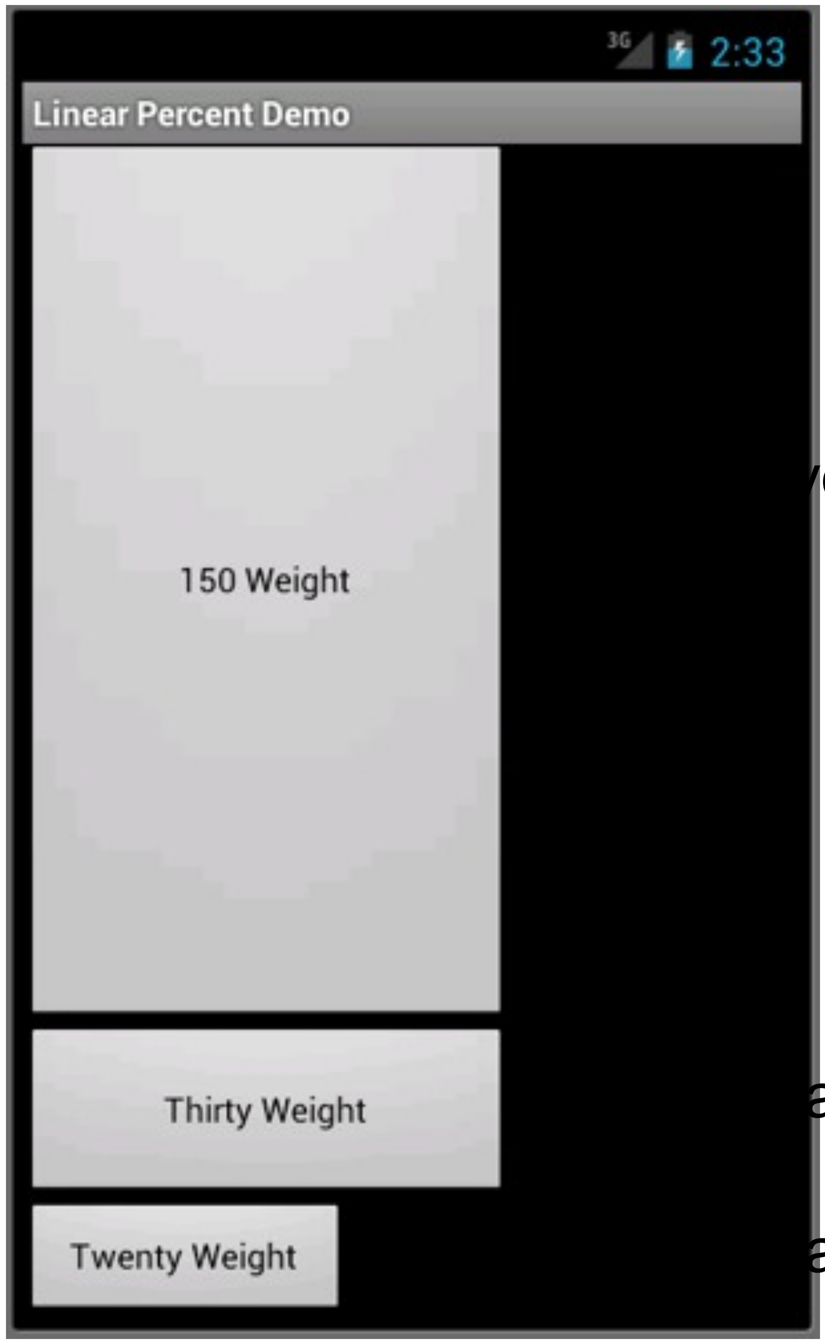
Exact number

wrap_content
- Big enough to enclose content + padding

fill_parent
- Big as parent minus padding
- SDK 7 and earlier

match_parent
- Big as parent minus padding
- SDK 8 and later
- Replaces fill_parent

Specifying Size of Widget



out_width="200sp"

ayout_width="200dip"

ayout_width="200px"

Units	
px	(pixels)
dp, dip	(density independent pixels
sp	(scaled pixels)
in	(inches)
mm	(millimeters)

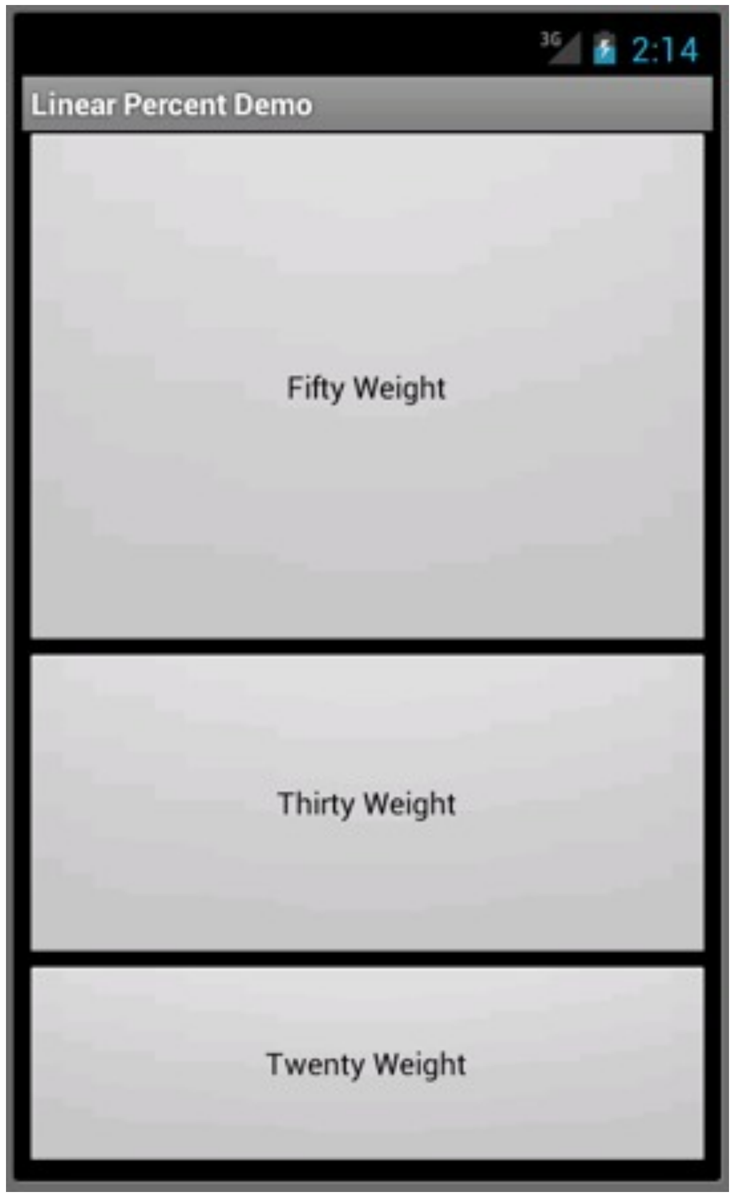
Weight

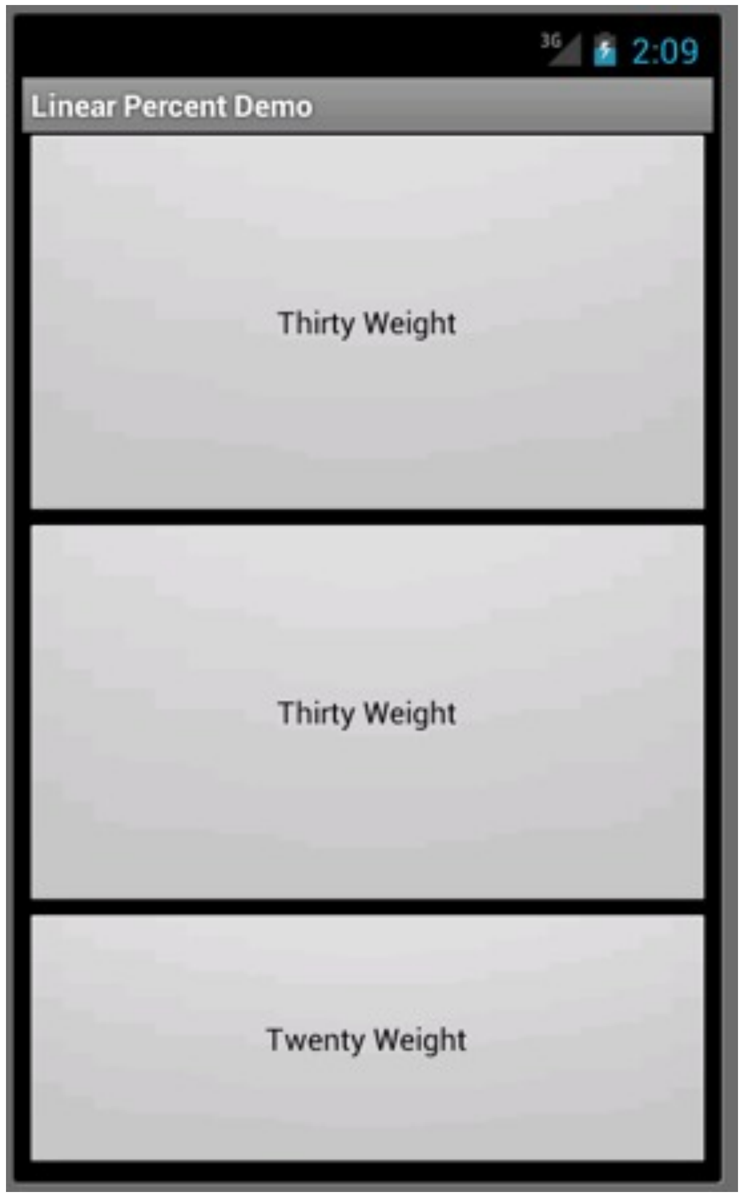
android:layout_weight

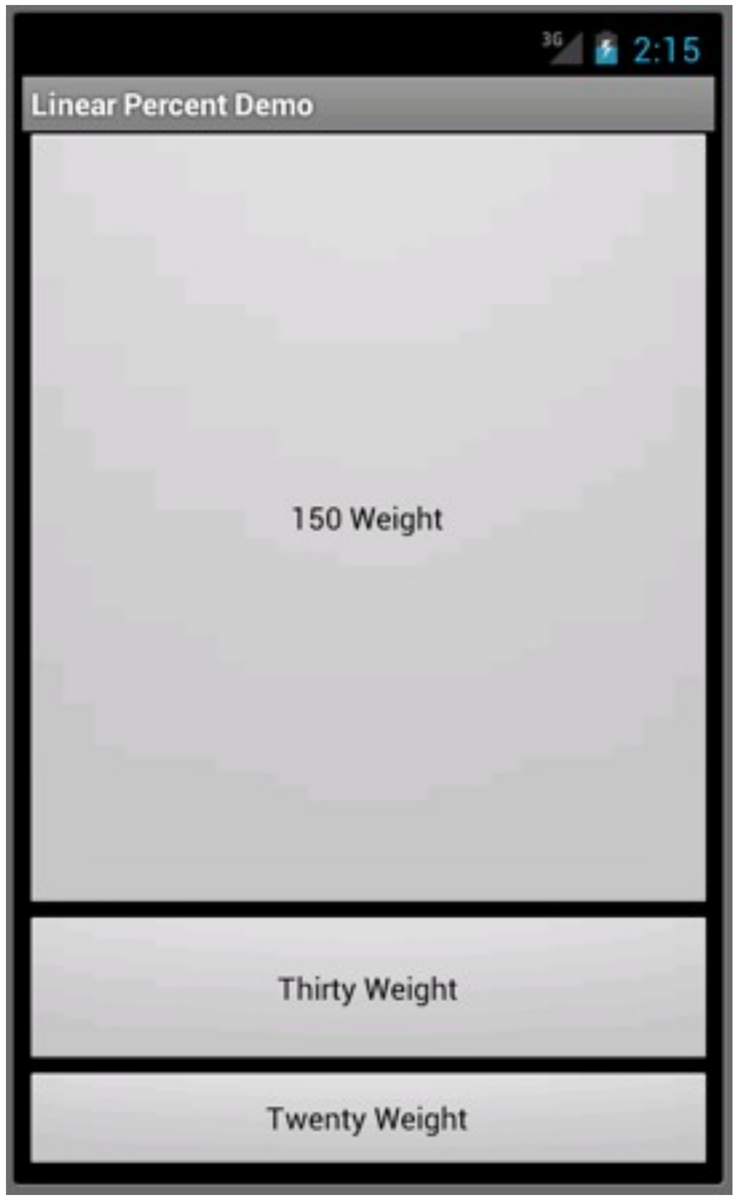
Relative weight of views to use in fill_parent

A view of twice the weight take twice the space

Example

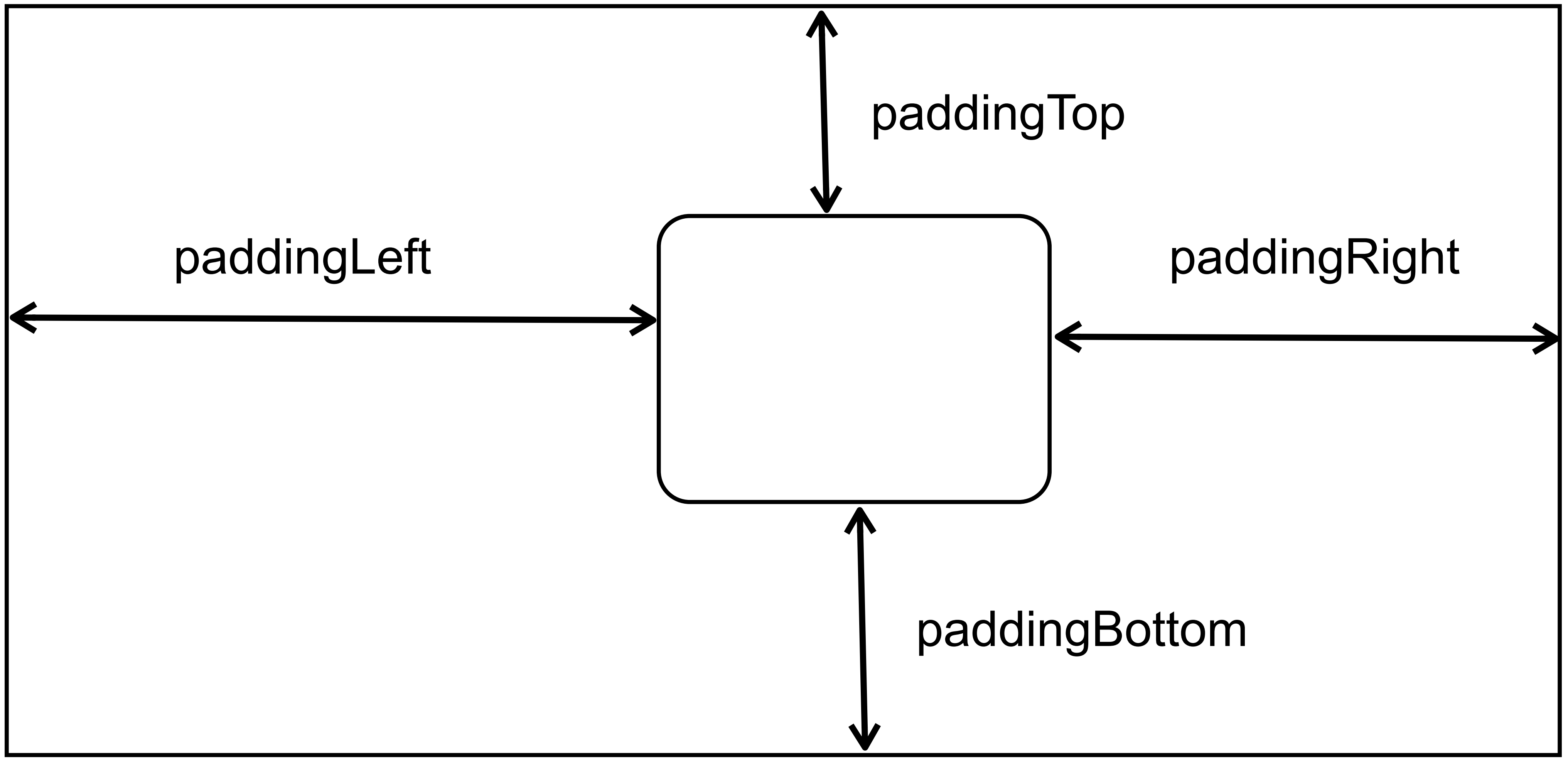

2:14
Linear Percent Demo
Fifty Weight
Thirty Weight
Twenty Weight


2:09
Linear Percent Demo
Thirty Weight
Thirty Weight
Twenty Weight


2:15
Linear Percent Demo
150 Weight
Thirty Weight
Twenty Weight

Padding

android:padding
android:paddingBottom
android:paddingLeft
etc
setPadding(int left, int top, int right, int bottom)
setPaddingLeft(int)
etc



in the xml padding can be in px, dp, dip, sp, in, mm
in methods it is in pixels

Relative Layout

Relative to parent

android:layout_alignParentTop:
android:layout_alignParentBottom:
android:layout_alignParentLeft:
android:layout_alignParentRight:
android:layout_centerHorizontal:
android:layout_centerVertical:

Relative Layout

have to give widget an id

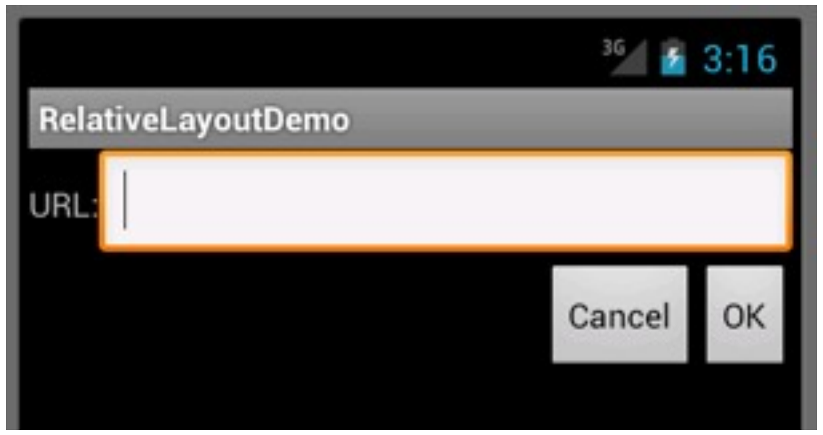
Relative to other widgets

must reference the id

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

android:layout_alignTop:
android:layout_alignBottom:
android:layout_alignLeft:
android:layout_alignRight:
android:layout_alignBaseline:

Example

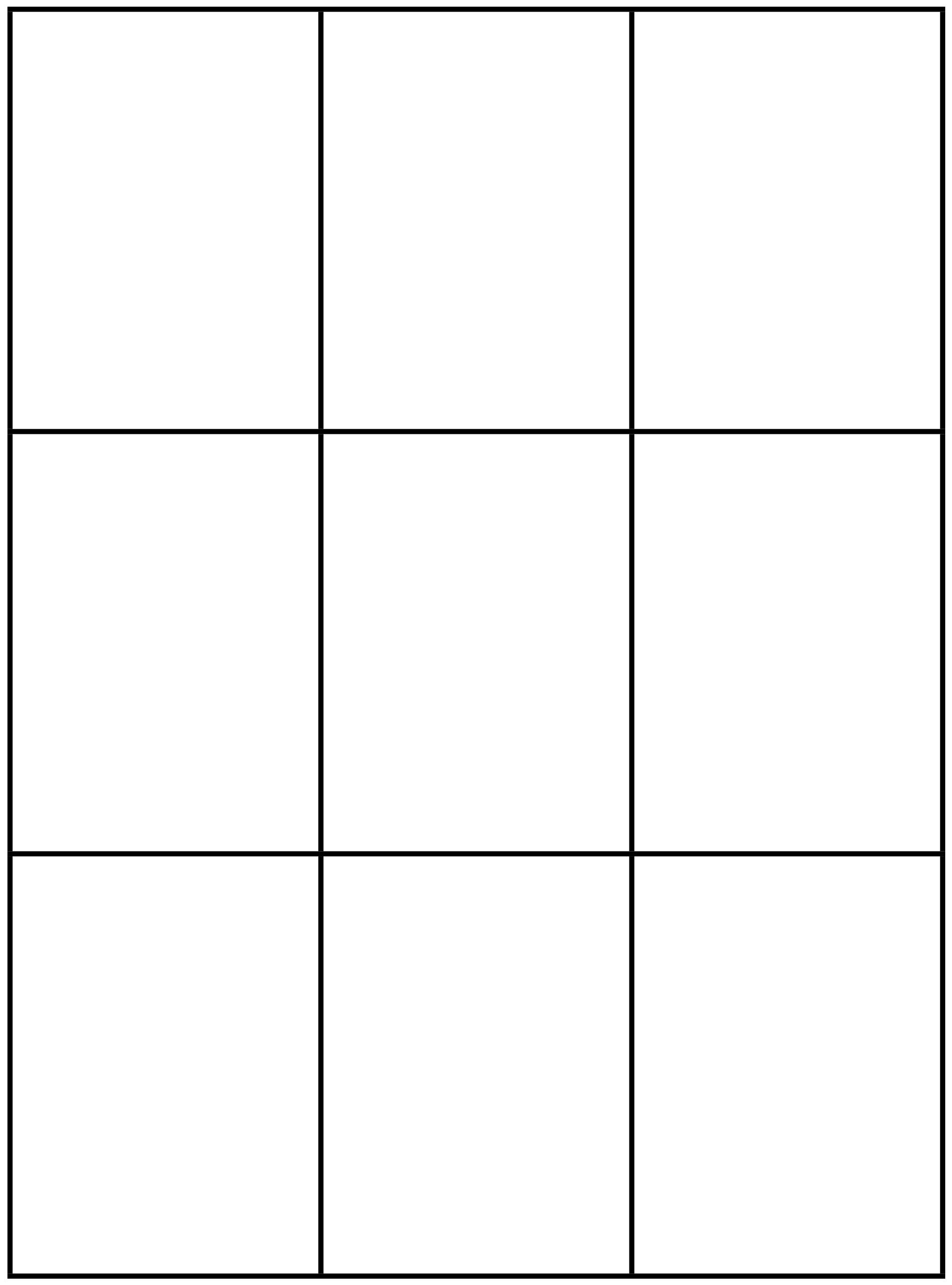


```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content">
    <TextView android:id="@+id/label"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="URL:"
        android:layout_alignBaseline="@+id/entry"
        android:layout_alignParentLeft="true"/>
    <EditText
        android:id="@id/entry"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_toRightOf="@id/label"
        android:layout_alignParentTop="true"/>
    <Button
        android:id="@+id/ok"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/entry"
        android:layout_alignRight="@id/entry"
        android:text="OK" />
```

Source: Beginning Android 4, Grant Allen, Chapter 10

Table View

Screen is divided into rows and columns



Creating Rows and Columns

```
<TableRow>
    <Button android:id="@+id/A"
        android:text="A" />

    <Button android:id="@+id/B"
        android:text="B" />

<Button android:id="@+id/C"
        android:text="C" />

</TableRow>
```

Each item in a row occupies a column

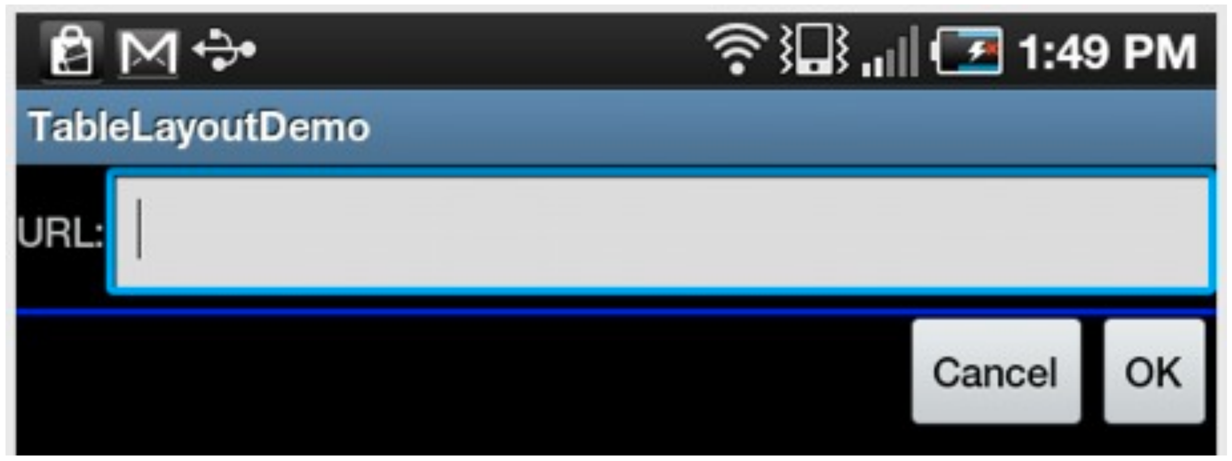
layout_span

```
<TableRow>
    <TextView
            android:text="URL:" />

    <EditText android:id="@+id/entry"              3 columns
        android:layout_span="3"/>

</TableRow>
```

Specifying the column

```
<TableRow>
    <Button android:id="@+id/cancel"
        android:layout_column="2"
        android:text="Cancel" />
    <Button android:id="@+id/ok"
        android:text="OK" />
</TableRow>
```

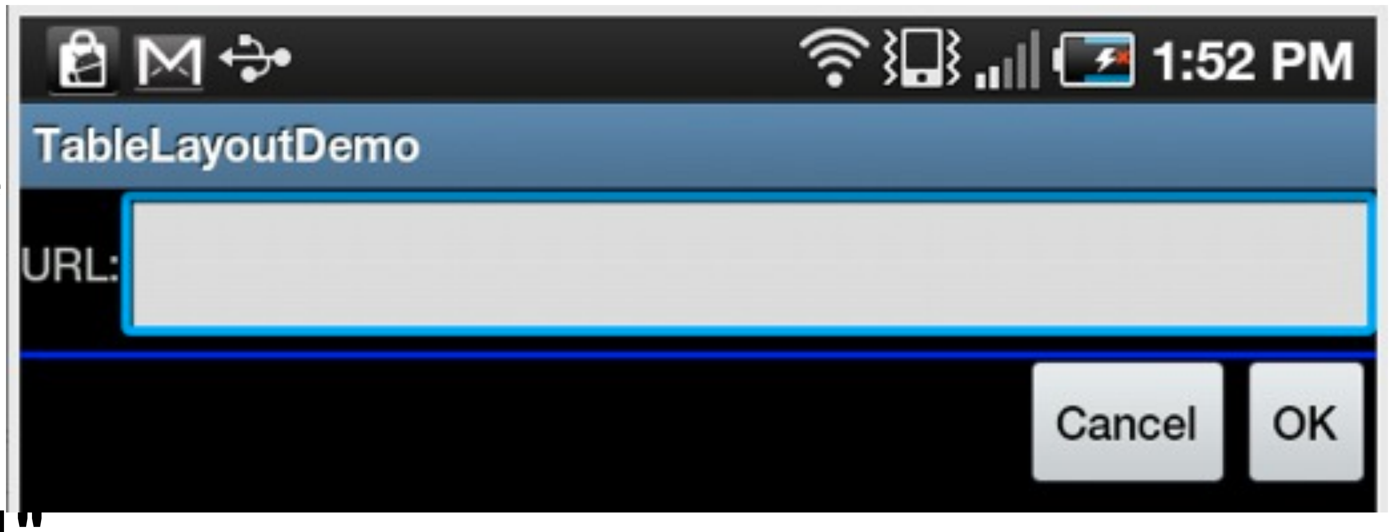
Example



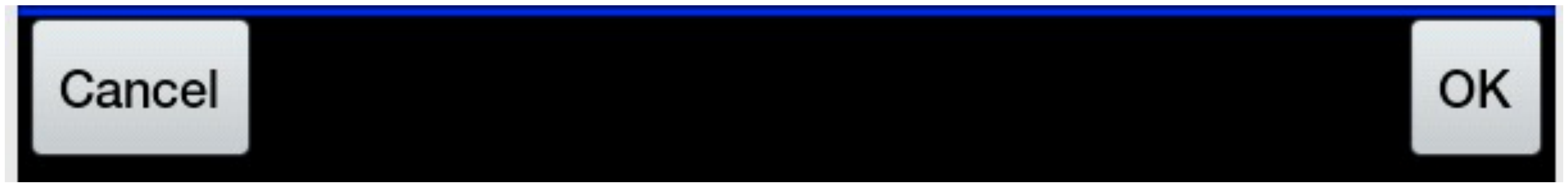
```
<TableLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:stretchColumns="1">
    <TableRow>
        <TextView
                android:text="URL:" />
        <EditText android:id="@+id/entry"
            android:layout_span="3"/>
    </TableRow>
    <View
        android:layout_height="2dip"
        android:background="#0000FF" />
    <TableRow>
        <Button android:id="@+id/cancel"
            android:layout_column="2"
            android:text="Cancel" />
        <Button android:id="@+id/ok"
            android:text="OK" />
    </TableRow>
</TableLayout>
```

Later items appear in later columns

```
<TableRow>
    <Button android:id="@+id/cance
        android:layout_column="2"
        android:text="Cancel" />
    <Button android:id="@+id/ok"
        android:layout_column="1"
        android:text="OK" />
</TableRow>
```

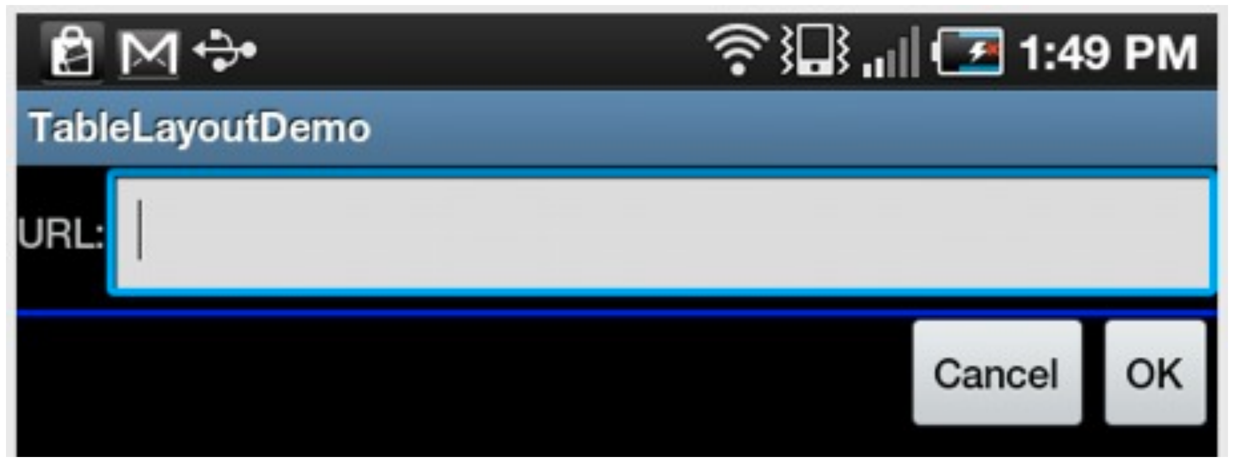


Can Skip Columns



```
<TableRow>
    <Button android:id="@+id/cancel"
        android:layout_column="0"
        android:text="Cancel" />
    <Button android:id="@+id/ok"
        android:layout_column="3"
        android:text="OK" />
</TableRow>
```

Stretch, Shrink, and Collapse



android:stretchColumns
android:shrinkColumns
android:collapseColumns

```
<TableLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:stretchColumns="1">
```

ScrollView, HorizontalScrollView

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content">
    <TableLayout
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:stretchColumns="0">
        <TableRow>
            <View
                android:layout_height="80dip"
                android:background="#000000"/>
            <TextView android:text="#000000"
                android:paddingLeft="4dip"
                android:layout_gravity="center_vertical" />
        </TableRow>
    </TableLayout>
</ScrollView>
```

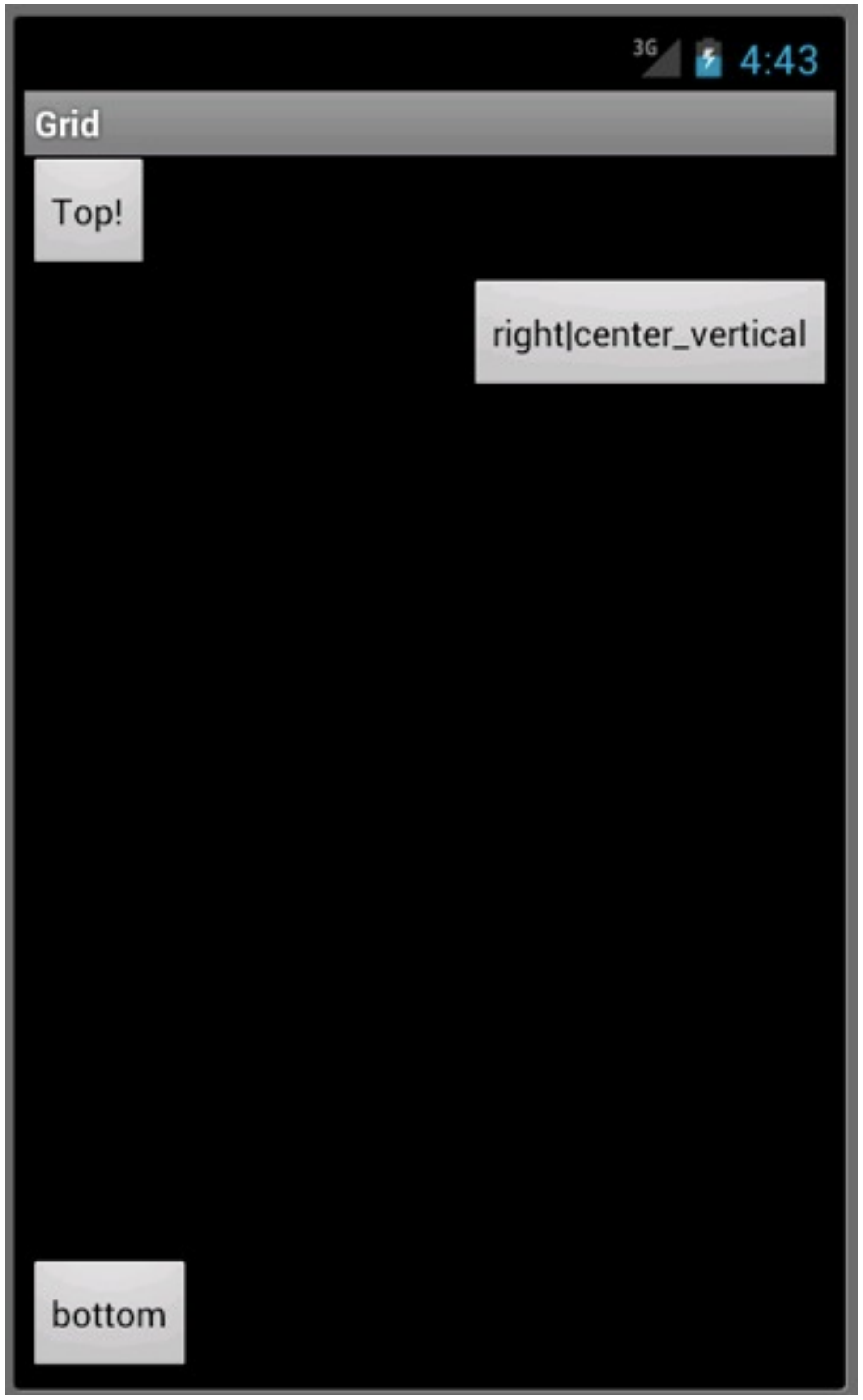
GridLayout

New in Android 4.0

Allows any number of rows and columns

Using Default Rows, Columns

```
<?xml version="1.0" encoding="utf-8"?>
<GridLayout
 xmlns:android="http://schemas.android.com/apk/res/android"
 android:orientation="vertical"
 android:layout_width="fill_parent"
 android:layout_height="fill_parent"
>
  <Button
    android:text="Top!"
    android:layout_gravity="top"
  />
  <Button
    android:text="right|center_vertical"
    android:layout_gravity="right|center_vertical"
  />
  <Button
    android:text="bottom"
    android:layout_gravity="bottom"
  />
</GridLayout>
```



Dialogs

Types of Dialogs

- AlertDialog
    - Can have buttons and checkboxes

- ProgressDialog
- DatePickerDialog
- TimePickerDialog
- Custom Dialogs

See http://developer.android.com/guide/topics/ui/dialogs.html

Activity.onCreateDialog(int)

```
static final int DIALOG_PAUSED_ID = 0;
static final int DIALOG_GAMEOVER_ID = 1;
```

Create dialogs in onCreateDialog

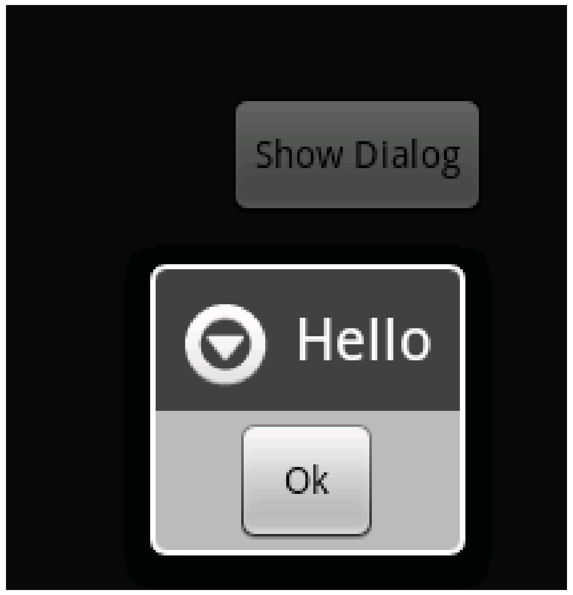
```
protected Dialog onCreateDialog(int id) {
    Dialog dialog;
    switch(id) {
    case DIALOG_PAUSED_ID:
        // do the work to define the pause Dialog
        break;
    case DIALOG_GAMEOVER_ID:
        // do the work to define the game over Dialog
        break;
    default:
        dialog = null;
    }
    return dialog;
}
```

showDialog(int)

To show a dialog in your activity call showDialog(int) which calls onCreateDialog the first time

```
showDialog(DIALOG_PAUSED_ID);
```

Creating an AlertDialog



Class DialogExample

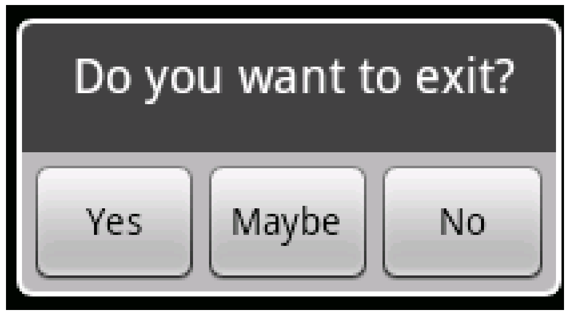
```
protected Dialog onCreateDialog(int id) {
        switch (id) {
        case SAMPLE_DIALOG_ID:
                AlertDialog.Builder builder = new AlertDialog.Builder(this);
                builder.setTitle("Hello").setPositiveButton("Ok",
                                new DialogInterface.OnClickListener() {
                                        public void onClick(DialogInterface dialog,
                                                        int whichButton) {
                                                DialogExample.this.finish();
                                                Toast.makeText(getApplicationContext(), "Good Bye",
Toast.LENGTH_SHORT).show();
                                        }
                                });
                return builder.create();
        default:
                return null;
        }
}
```

Three Buttons

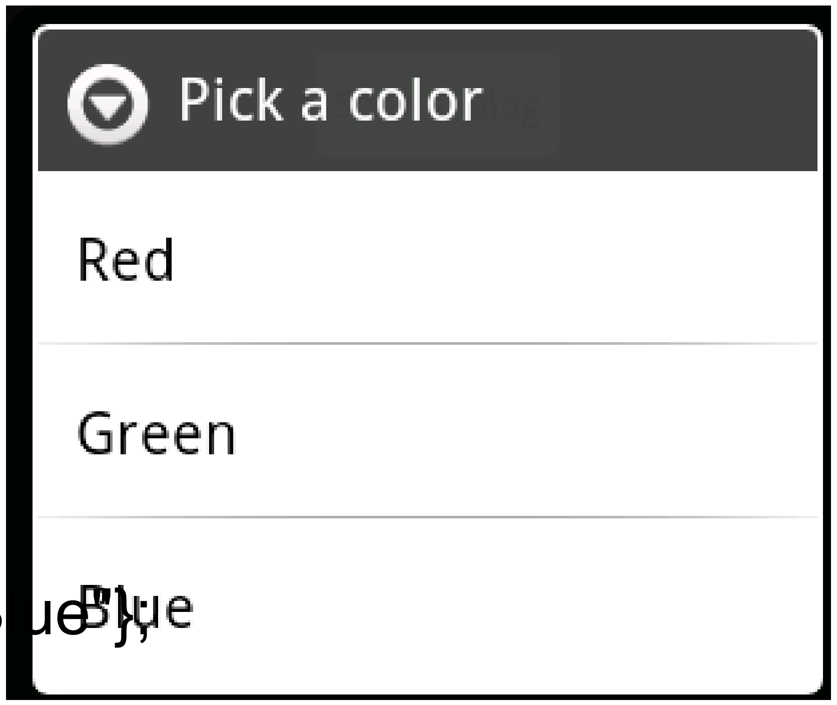
Positive	Can have only one of each
Negative	Button types have no meaing
Neutral	Positive can do what every you want

Three Button Example

```
AlertDialog.Builder builder = new AlertDialog.Builder(this);
builder.setMessage("Do you want to exit?")
        .setCancelable(false)
        .setPositiveButton("Yes",
                new DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface dialog,
                                        int whichButton) {
                                Toast.makeText(getApplicationContext(), "Good Bye",
                                        Toast.LENGTH_SHORT).show();
                                DialogExample.this.finish();
                        }
                })
                .setNegativeButton("No",
                new DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface dialog,
                                        int whichButton) {
                                dialog.cancel();
                        }
                })
                .setNeutralButton("Maybe",
                new DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface dialog,
                                        int whichButton) {
                                Toast.makeText(getApplicationContext(), "Make up your mind",
                                        Toast.LENGTH_SHORT).show();
                                DialogExample.this.showDialog(SAMPLE_DIALOG_ID); //Does not work
                        }
                });
return builder.create();
```

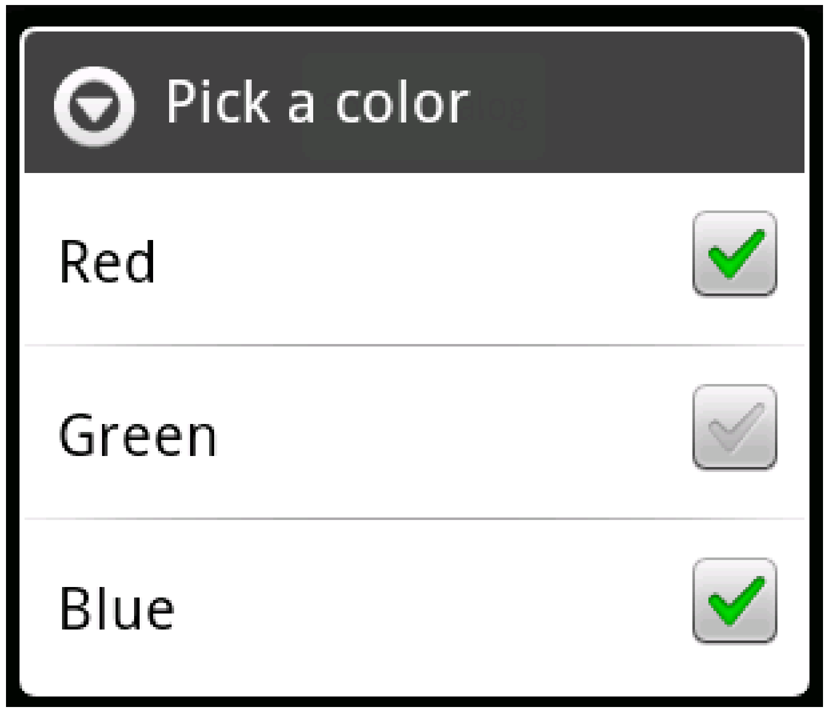


Lists



```
protected Dialog onCreateDialog(int id) {
        switch (id) {
        case SAMPLE_DIALOG_ID:
                final CharSequence[] items = {"Red", "Green", "Blue"};

                AlertDialog.Builder builder = new AlertDialog.Builder(this);
                builder.setTitle("Pick a color");
                builder.setItems(items, new DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog, int item) {
                        Toast.makeText(getApplicationContext(), items[item],
                      Toast.LENGTH_SHORT).show();
                    }
                });
                return builder.create();
        default:
                return null;
        }
}
```

MultiSelection



```
protected Dialog onCreateDialog(int id) {
	switch (id) {
	case SAMPLE_DIALOG_ID:
		final CharSequence[] items = {"Red", "Green", "Blue"};
		final boolean[] selected = {false, true, false};

		AlertDialog.Builder builder = new AlertDialog.Builder(this);
		builder.setTitle("Pick a color")
			.setMultiChoiceItems(items, selected, new DialogInterface.OnMultiChoiceClickListener() {
		    public void onClick(DialogInterface dialog, int item, boolean isChecked) {
			Toast.makeText(getApplicationContext(), items[item] + " isChecked " + isChecked,
		 Toast.LENGTH_SHORT).show();
		    }
		});
		return builder.create();
	default:
		return null;
	}
}
```

Thursday, September 27, 12

Should add a button to let the user exit. Also need keep track of which items are selected. Docs say that the Dialog has that information - dialog.getListView().isItemChecked(int position) or dialog.getListView().getCheckedItemPositions()