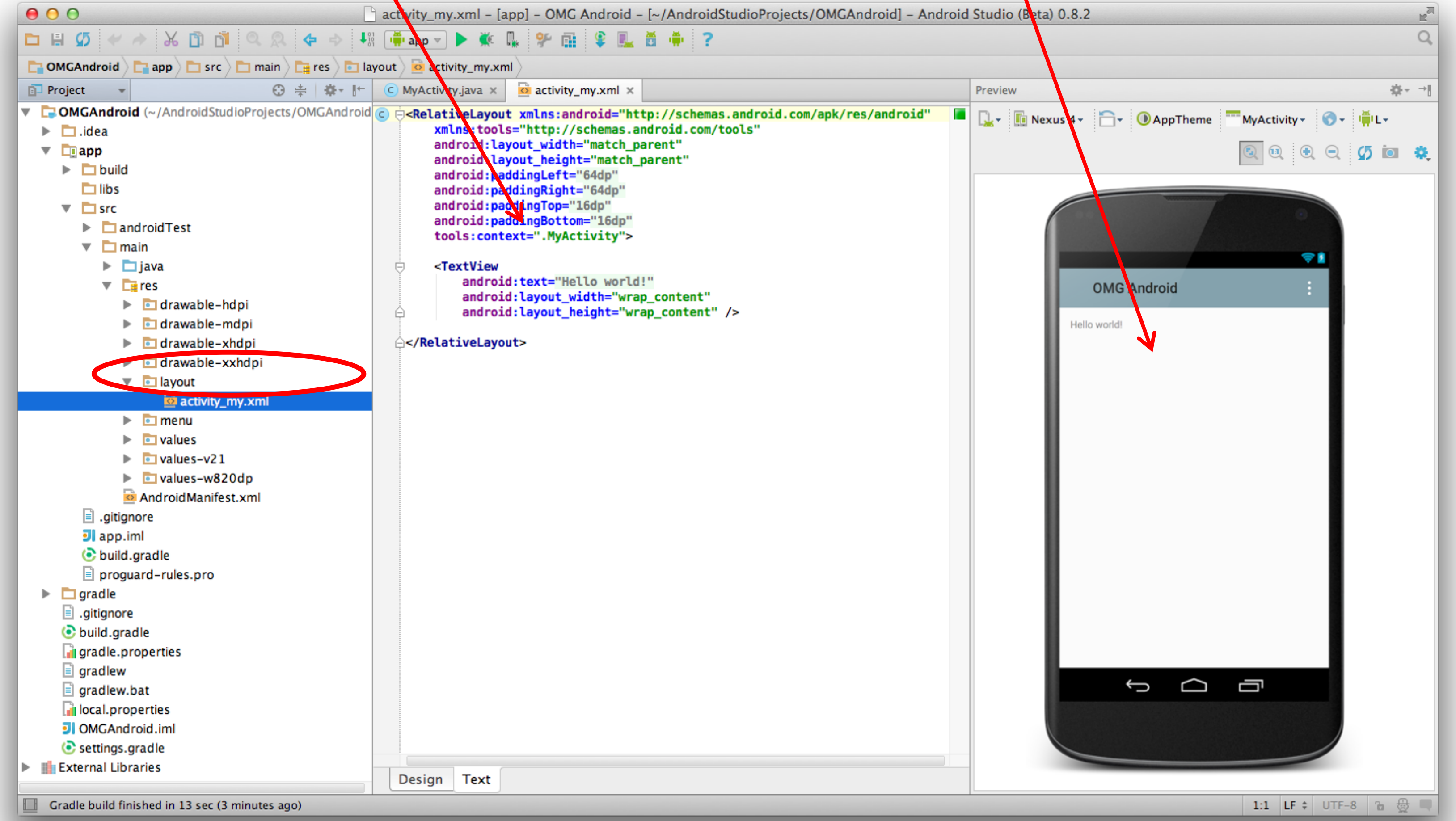
CS 528 Mobile and Ubiquitous Computing
Lecture 02a: Android UI Design

Emmanuel Agu

Editting in Android Studio

Recall: Editting Android

- Can edit apps in:
  - **Text View:** edit XML directly
  - **Design View:** or drag and drop widgets unto emulated phone



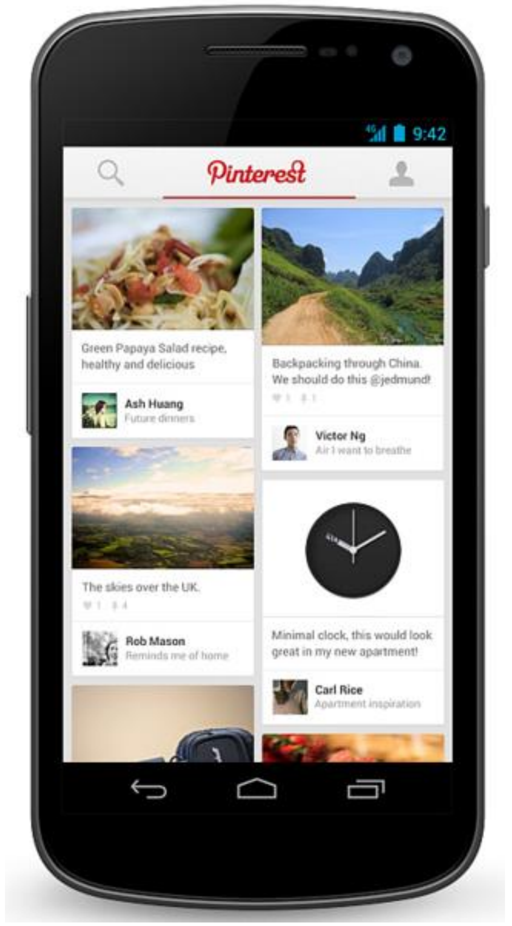
```xml
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="64dp"
    android:paddingRight="64dp"
    android:paddingTop="16dp"
    android:paddingBottom="16dp"
    tools:context=".MyActivity">

    <TextView
        android:text="Hello world!"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</RelativeLayout>
```

Android UI Design in XML

Recall: Files Hello World Android Project

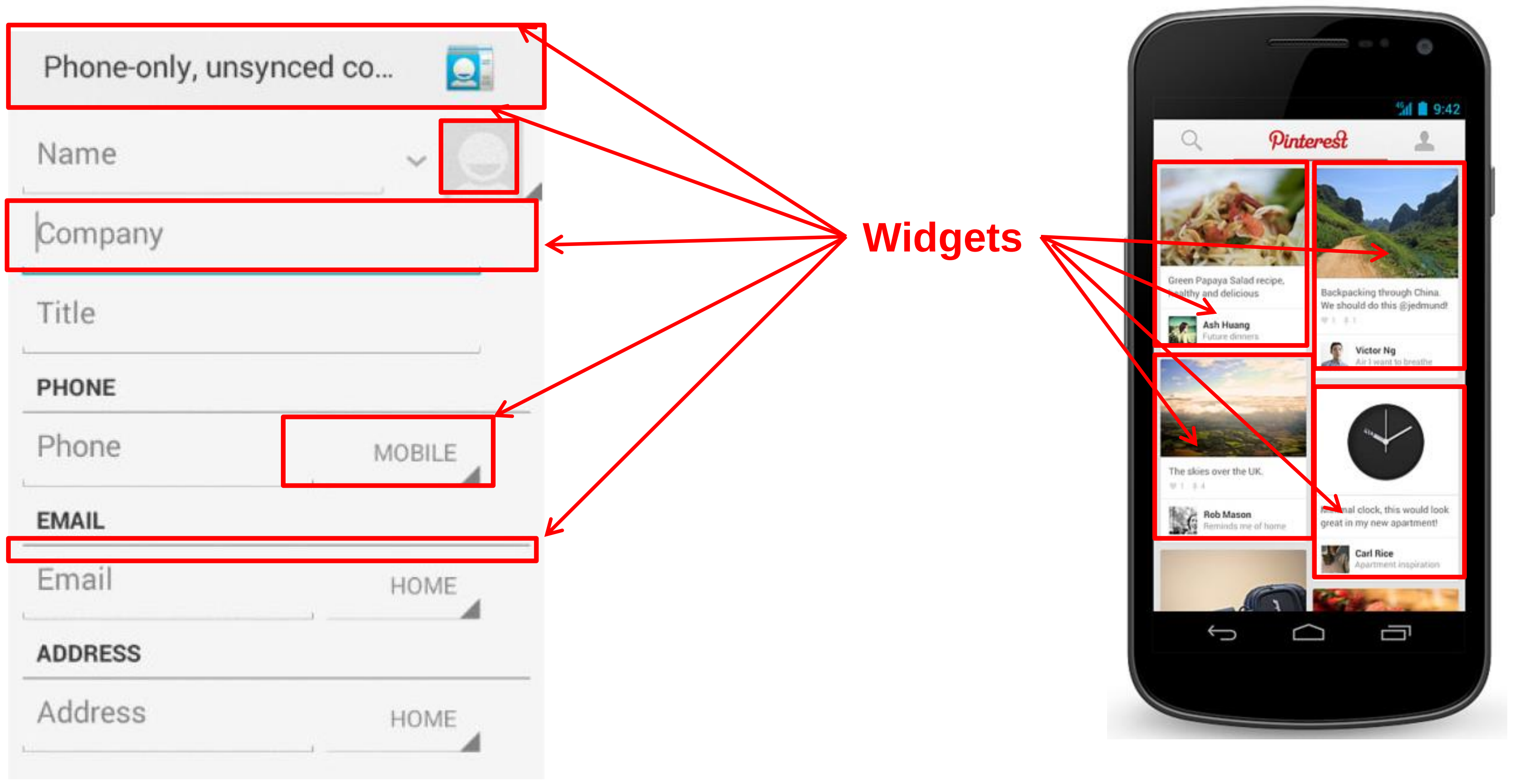
XML file used to design Android UI

- 3 Files:
  - **Activity_main.xml:** XML file specifying screen layout
  - **MainActivity.Java:** Java code to define behavior, actions taken when button clicked (intelligence)
  - **AndroidManifest.xml:**
    - Lists all app components and screens
    - Like a table of contents for a book
    - E.g. Hello world program has 1 screen, so AndroidManifest.xml has 1 item listed
    - App starts running here (a bit like main( ) in C), launching activity with a tag "LAUNCHER"


9:42
Pinterest
Green Papaya Salad recipe, healthy and delicious
Ash Huang
Backpacking through China. We should do this @jedmund!
Victor Ng
The skies over the UK.
Rob Mason
Minimal clock, this would look great in my new apartment!
Carl Rice

Recall: Widgets

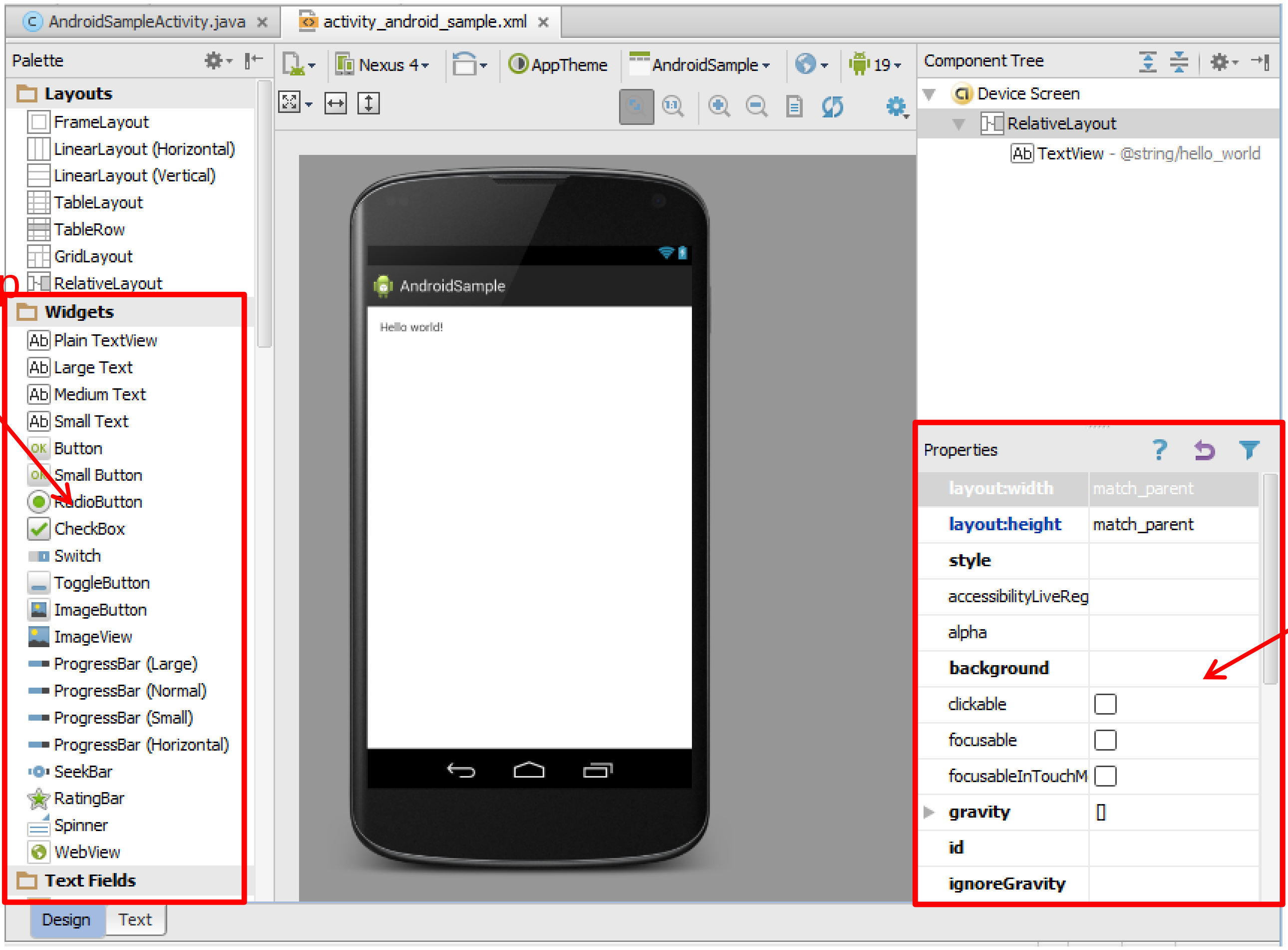
- ***Android UI design involves arranging widgets on a screen***
- **Widgets?** Rectangles containing texts, image, etc
- **Screen design:** Pick widgets, specify attributes (dimensions, margins, etc)



Recall: Design Option 1: Drag and Drop Widgets

- Drag and drop widgets in Android Studio Design View
- Edit widget properties (e.g. height, width, color, etc)

Drag and drop button or any other widget or view

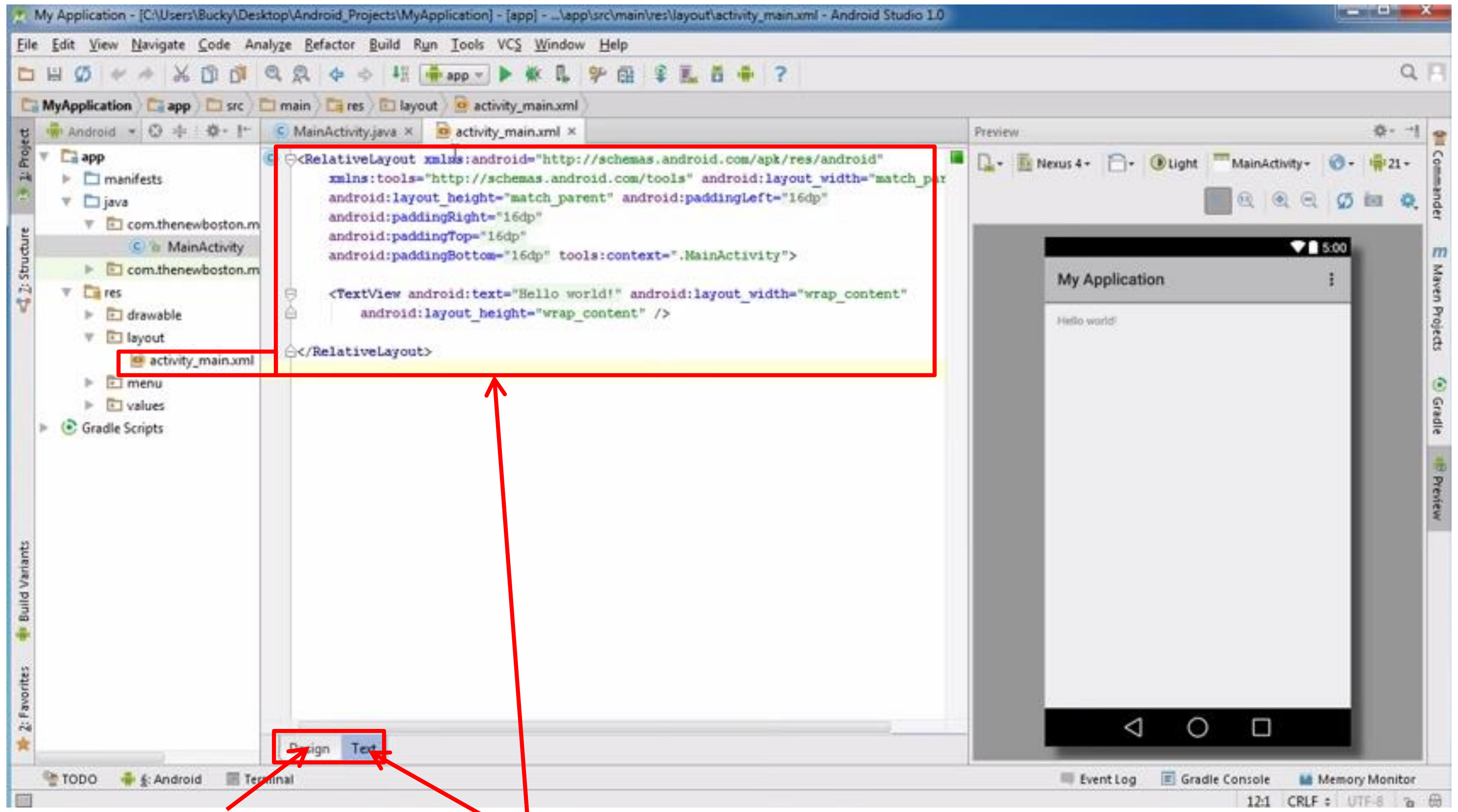


layout:width	match_parent
layout:height	match_parent
style	
accessibilityLiveReg	
alpha	
background	
clickable	☐
focusable	☐
focusableInTouchM	☐
gravity	[]
id	
ignoreGravity	

Edit widget properties

Recall: Design Option 2: Edit XML Directly

- **Text view:** Directly edit XML file defining screen (activity_main.xml)
- **Note:** dragging and dropping widgets in design view auto-generates corresponding XML in Text view



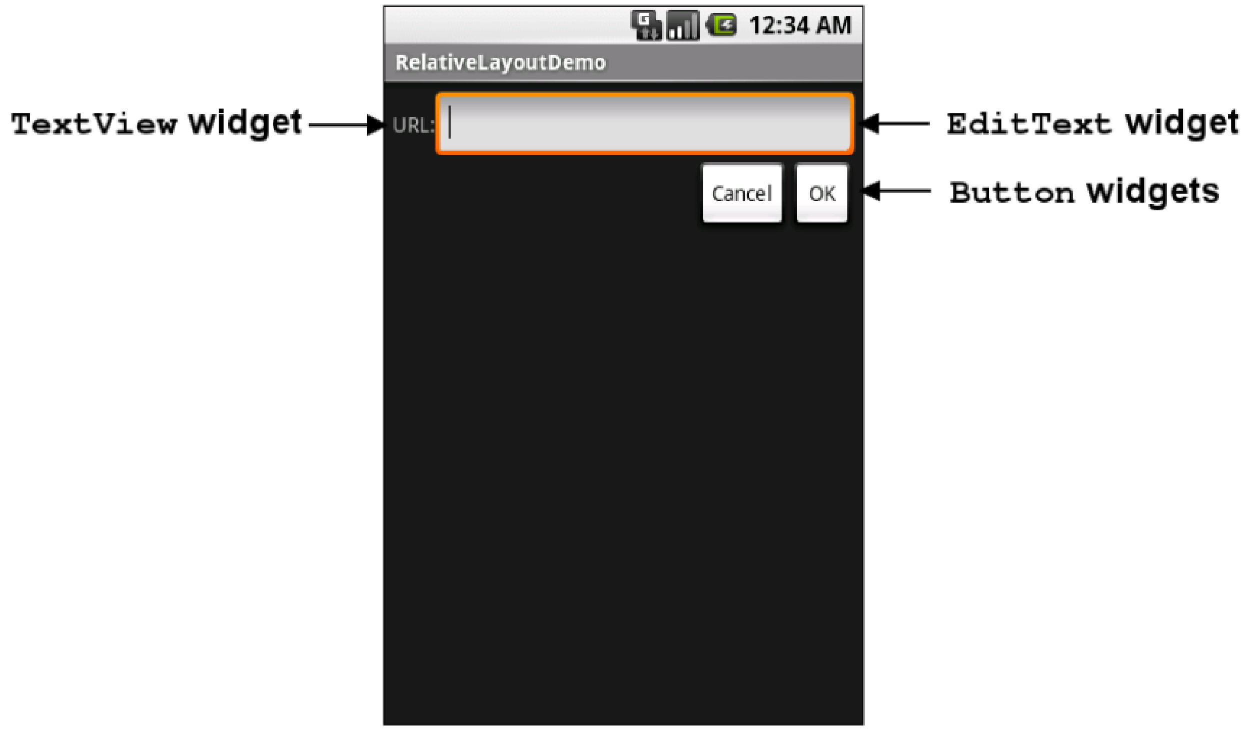
```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools" android:layout_width="match_par
    android:layout_height="match_parent" android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp"
    android:paddingBottom="16dp" tools:context=".MainActivity">

    <TextView android:text="Hello world!" android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

</RelativeLayout>
```

My Application - [C:\Users\Bucky\Desktop\Android_Projects\MyApplication] - [app] - ...\app\src\main\res\layout\activity_main.xml - Android Studio 1.0
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
MyApplication app src main res layout activity_main.xml
MainActivity.java
activity_main.xml
app
manifests
java
com.thenewboston.m
MainActivity
res
drawable
layout
menu
values
Gradle Scripts
Preview
Nexus 4
Light
MainActivity
21
My Application
Hello world!
Design
Text
TODO
6: Android
Terminal
Event Log
Gradle Console
Memory Monitor
Drag and drop widget
Edit XML

Android Widgets

Example: Some Common Widgets

- **TextView:** Text in a rectangle
- **EditText:** Text box for user to type in text
- **Button:** Button for user to click on



General Form of Widget Declaration

E.g. TextView, button, EditText, etc

<widget type

List of attributes (e.g. format, width, length, etc)

…………

………..

/>

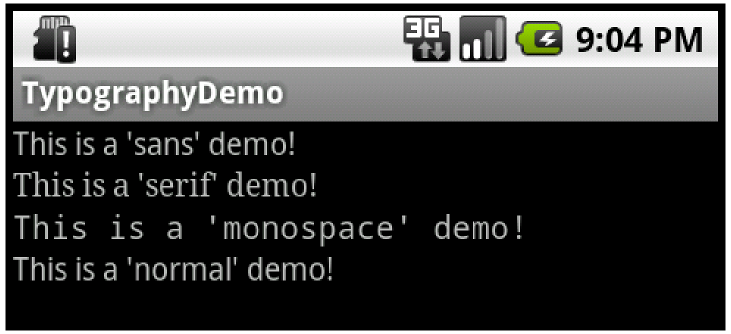
TextView Widget

- Text in a rectangle
- Just displays text, no interaction

XML code

```
<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="This is a 'sans' demo!"
    android:typeface="sans"
/>
```

TextView Widgets



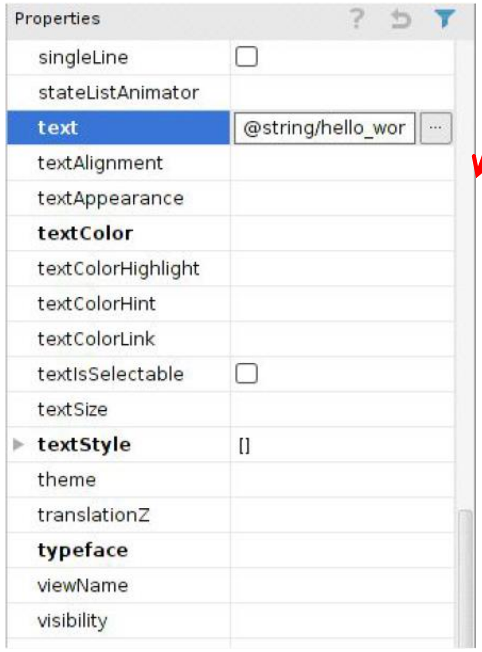
This is a 'sans' demo!
This is a 'serif' demo!
This is a 'monospace' demo!
This is a 'normal' demo!

- **Common attributes:**
  - typeface (android:typeface e.g monospace), bold, italic, (android:textStyle), text size, text color (android:textColor e.g. #FF0000 for red), width, height, padding, background color
  - Can also include links to email address, url, phone number,
    - web, email, phone, map, etc

TextView

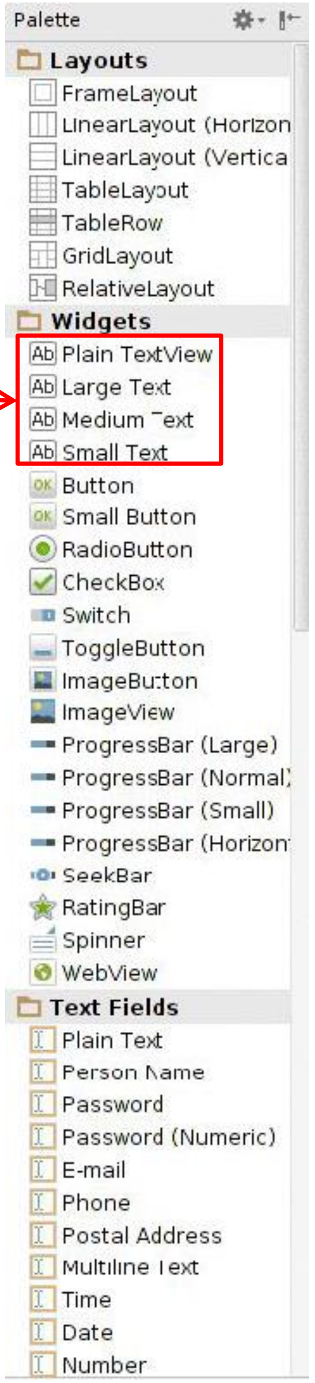
- TextView widget is available in widgets palette in Android Studio Layout editor
  - **Plain TextView, Large text, Medium text** and **Small text**

- After dragging Textview widget in, edit properties



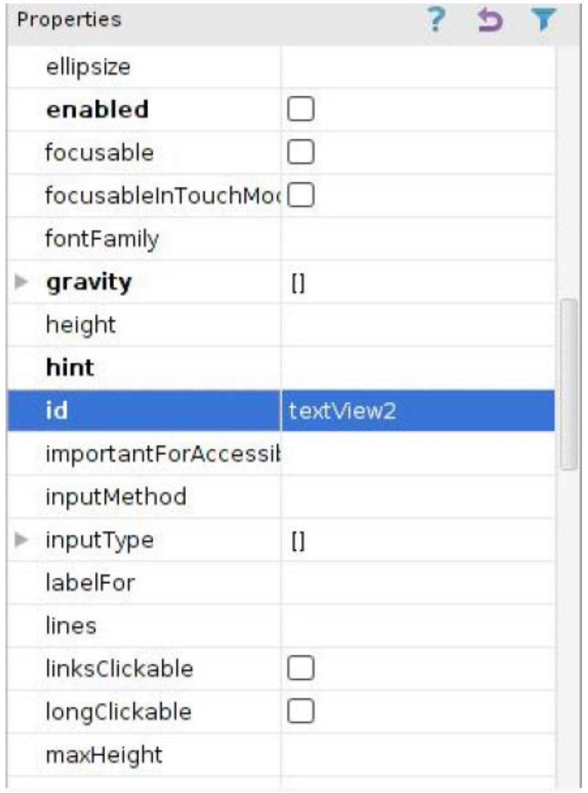
Properties

singleLine	☐
stateListAnimator	
text	@string/hello_wor
textAlignment	
textAppearance	
textColor	
textColorHighlight	
textColorHint	
textColorLink	
textIsSelectable	☐
textSize	
textStyle	[]
theme	
translationZ	
typeface	
viewName	
visibility	



Widget ID

- Every widget has ID, stored in **android:id** attribute
- Using Widget ID declared in XML, widget can be referenced, modified in java code (More later)

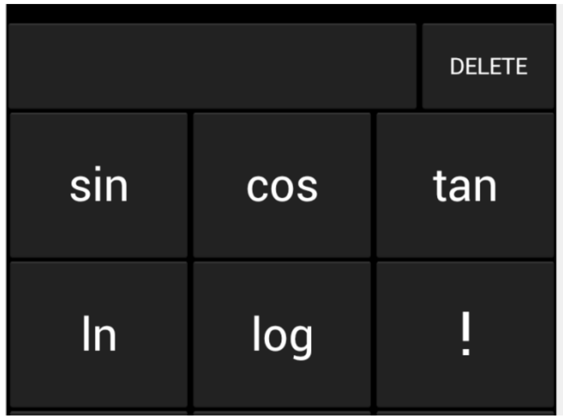


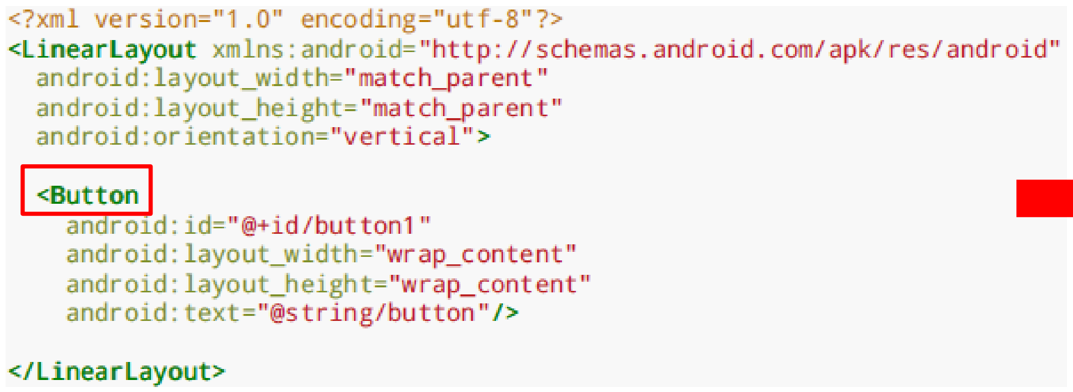
Properties

ellipsize	
enabled	☐
focusable	☐
focusableInTouchMo	☐
fontFamily	
gravity	[]
height	
hint	
id	textView2
importantForAccessil	
inputMethod	
inputType	[]
labelFor	
lines	
linksClickable	☐
longClickable	☐
maxHeight	

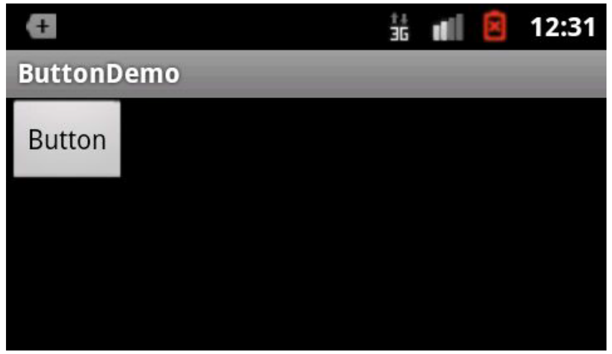
Button Widget

- Clickable Text or icon on a Widget (Button)
- E.g. “Click Here”
- Appearance can be customized
- Declared as subclass of TextView so similar attributes (e.g. width, height, etc)





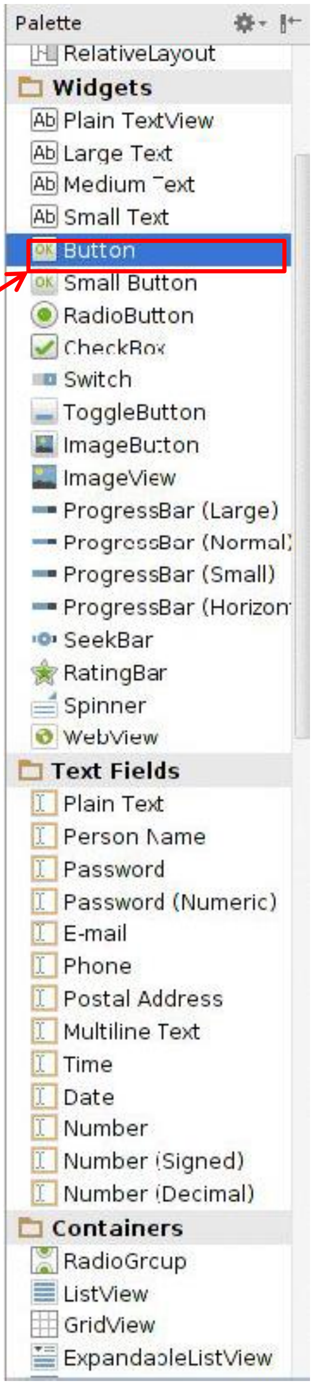
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical">

  <Button
    android:id="@+id/button1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/button"/>

</LinearLayout>
```



Button in Android Studio

- **Button** widget available in palette of Android Studio graphical layout editor

- Drag and drop button, edit its attributes



Responding to Button Clicks

- May want Button press to trigger some action
- How?

1. In XML file (e.g. Activity_my.xml), set android:onClick attribute to specify method to be invoked

2. In Java file (e.g. MainActivity.java) declare method/handler to take desired action

Activity_my.xml

```
<Button
  android:onClick="someMethod"
  ...
/>
```

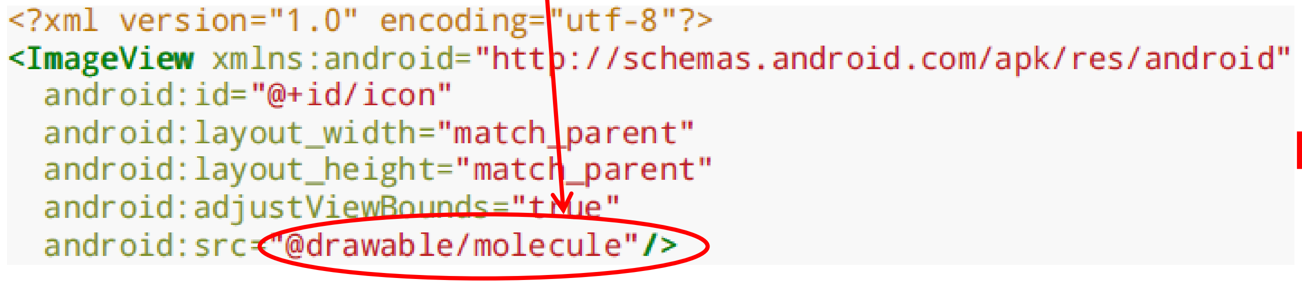
MainActivity.java

```
public void someMethod(View theButton) {
  // do something useful here
}
```

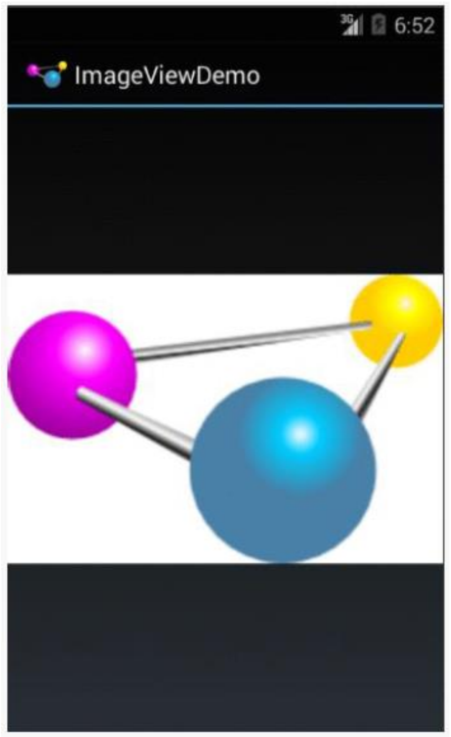
Embedding Images: ImageView and ImageButton

- **ImageView:** display image (not clickable)
- **ImageButton:** Clickable image

- Use **android:src** attribute to specify image source in **drawable** folder (e.g. **@drawable/icon**)



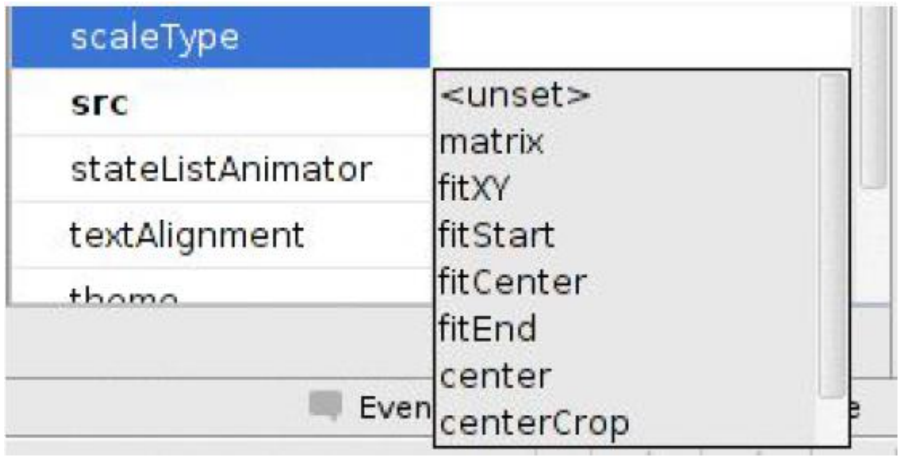
```
<?xml version="1.0" encoding="utf-8"?>
<ImageView xmlns:android="http://schemas.android.com/apk/res/android"
  android:id="@+id/icon"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:adjustViewBounds="true"
  android:src="@drawable/molecule"/>
```



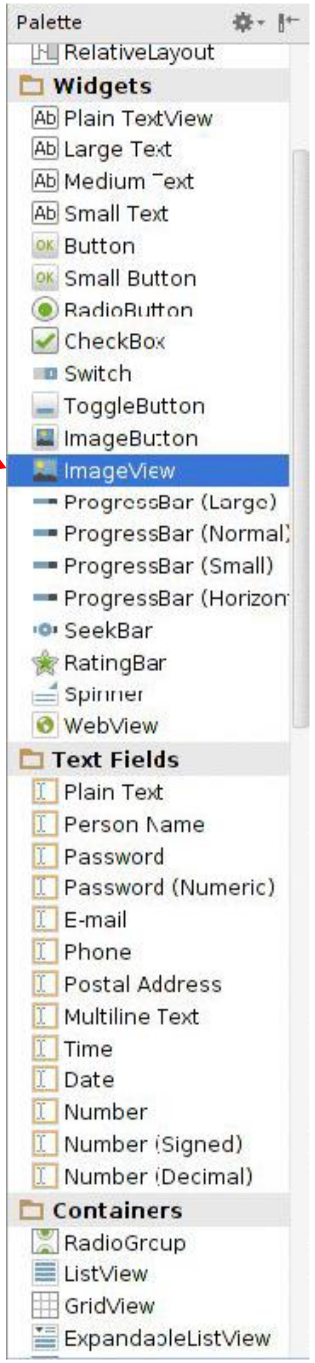
File molecule.png in drawable/ folder

ImageView in Widgets Palette

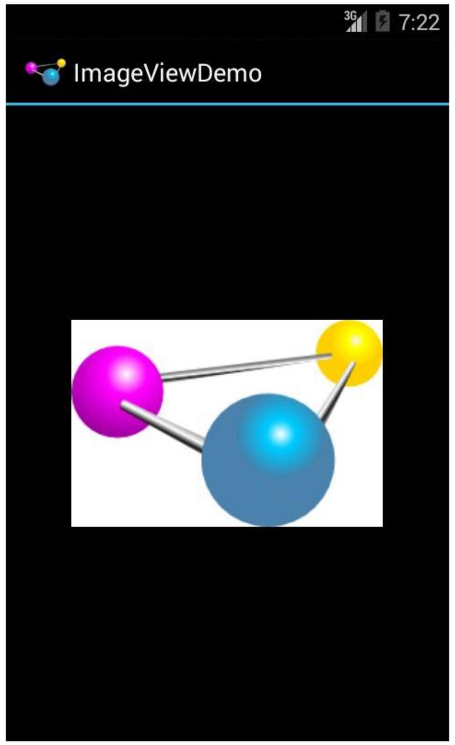
- Can drag and drop ImageView from Widgets Palette
- Use pop-up menus (right-click) to specify:
  - **src:** choose image to be displayed
  - **scaleType:** choose how image should be scaled

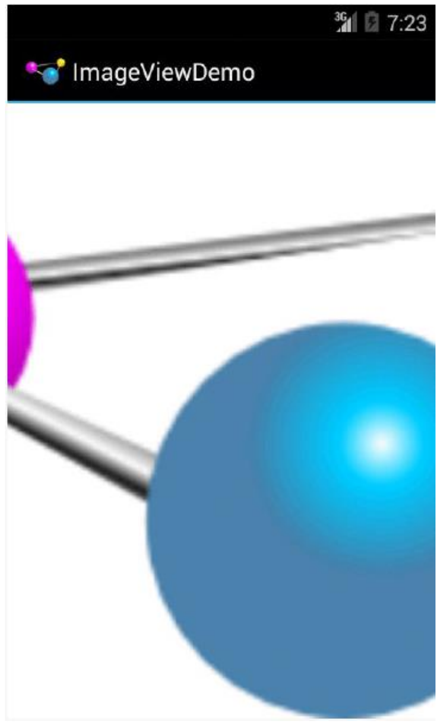

scaleType
src
stateListAnimator
textAlignment
theme

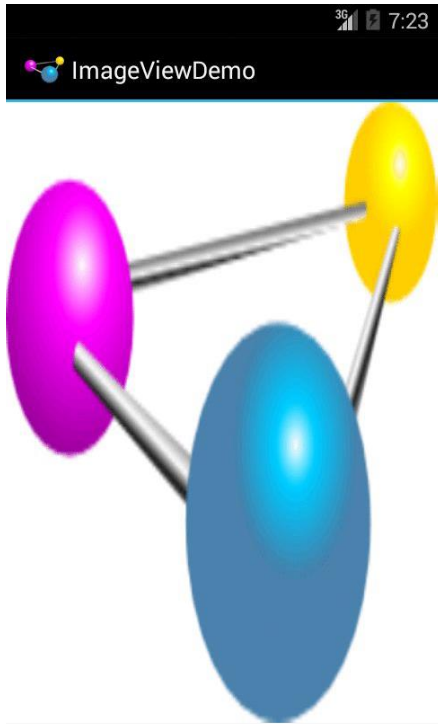
<unset>
matrix
fitXY
fitStart
fitCenter
fitEnd
center
centerCrop


Palette
RelativeLayout
Widgets
Plain TextView
Large Text
Medium Text
Small Text
Button
Small Button
RadioButton
CheckBox
Switch
ToggleButton
ImageButton
ImageView
ProgressBar (Large)
ProgressBar (Normal)
ProgressBar (Small)
ProgressBar (Horizon
SeekBar
RatingBar
Spinner
WebView
Text Fields
Plain Text
Person Name
Password
Password (Numeric)
E-mail
Phone
Postal Address
Multiline Text
Time
Date
Number
Number (Signed)
Number (Decimal)
Containers
RadioGroup
ListView
GridView
ExpandableListView

Options for Scaling Images (scaleType)







"center" centers image but does not scale it

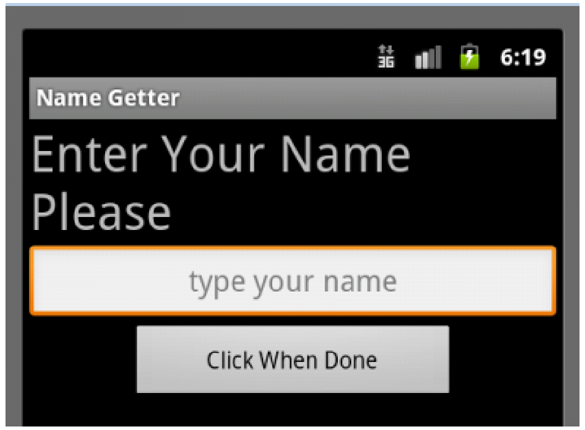
"centerCrop" centers image, scales it (maintaining aspect ratio) so that shorter dimension fills available space, and crops longer dimension

"fitXY" scales/distorts image to fit ImageView, ignoring aspect ratio

EditText Widget

- Widget with box for user input
- Example:

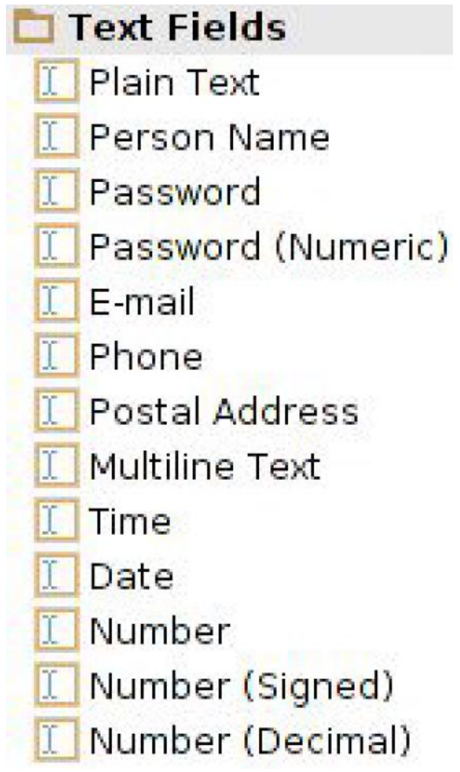
```
<EditText
        android:id="@+id/edittext"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:gravity="center"
        android:inputType="textPersonName"
        android:hint="type your name" />
```



- Text fields can have different input types
    - e.g. number, date, password, or email address
- **android:inputType** attribute sets input type, affects
    - What type of keyboard pops up for user
    - E.g. if inputType is a number, numeric keyboard pops up

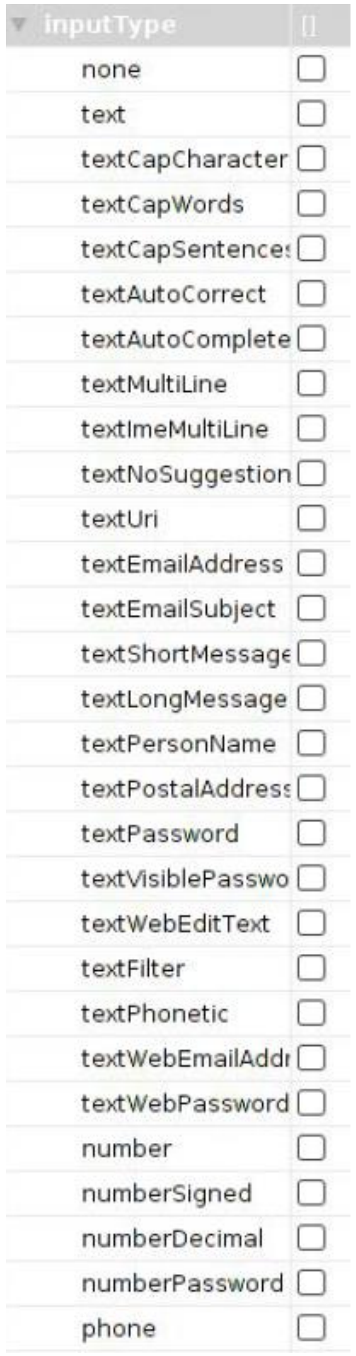
EditText Widget in Android Studio Palette

- A section of Android Studio palette has EditText widgets (or text fields)



Text Fields
- Plain Text
- Person Name
- Password
- Password (Numeric)
- E-mail
- Phone
- Postal Address
- Multiline Text
- Time
- Date
- Number
- Number (Signed)
- Number (Decimal)

Text Fields
Section of Widget palette

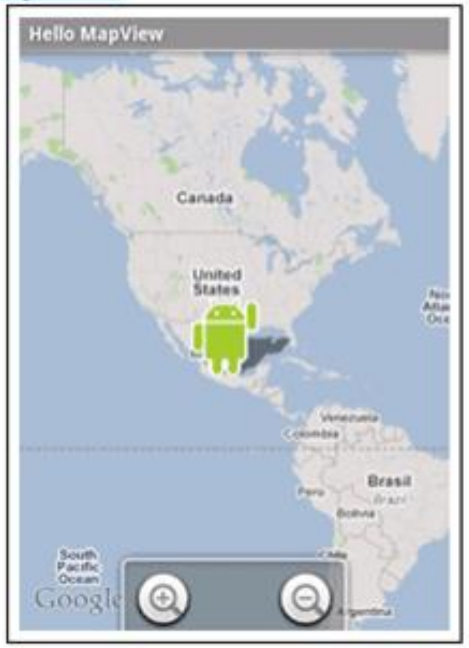


inputType	[]
none	☐
text	☐
textCapCharacter	☐
textCapWords	☐
textCapSentence:	☐
textAutoCorrect	☐
textAutoComplete	☐
textMultiLine	☐
textImeMultiLine	☐
textNoSuggestion	☐
textUri	☐
textEmailAddress	☐
textEmailSubject	☐
textShortMessage	☐
textLongMessage	☐
textPersonName	☐
textPostalAddress	☐
textPassword	☐
textVisiblePasswo	☐
textWebEditText	☐
textFilter	☐
textPhonetic	☐
textWebEmailAddr	☐
textWebPassword	☐
number	☐
numberSigned	☐
numberDecimal	☐
numberPassword	☐
phone	☐

EditText inputType menu

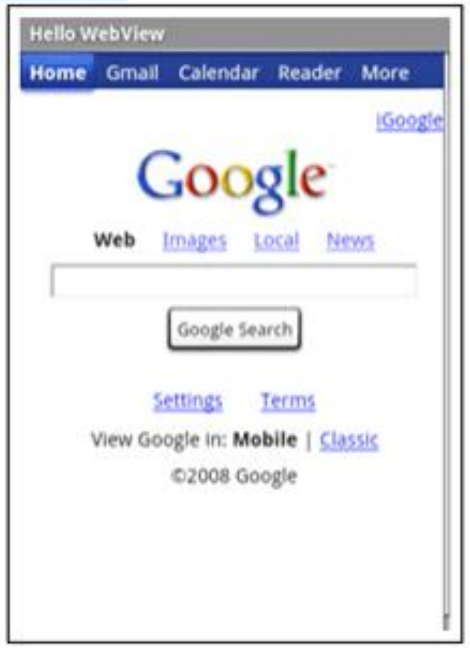
Some Other Available Widgets

MapView



Rectangle that contains a map

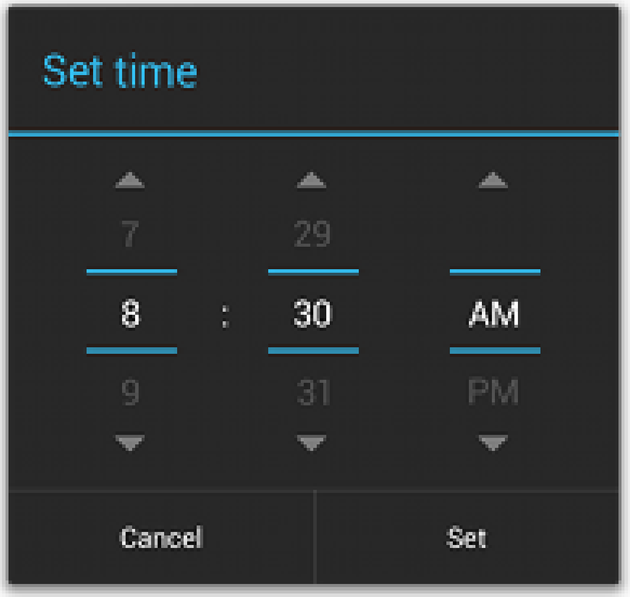
WebView



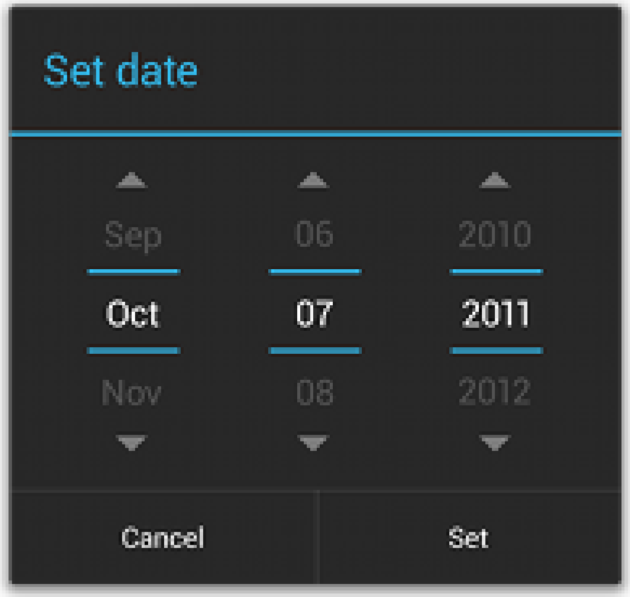
Rectangle that contains a web page

Pickers

- **TimePicker:** Select a time
- **DatePicker:** Select a date
- Typically displayed in pop-up dialogs (**TimePickerDialog** or **DatePickerDialog**)



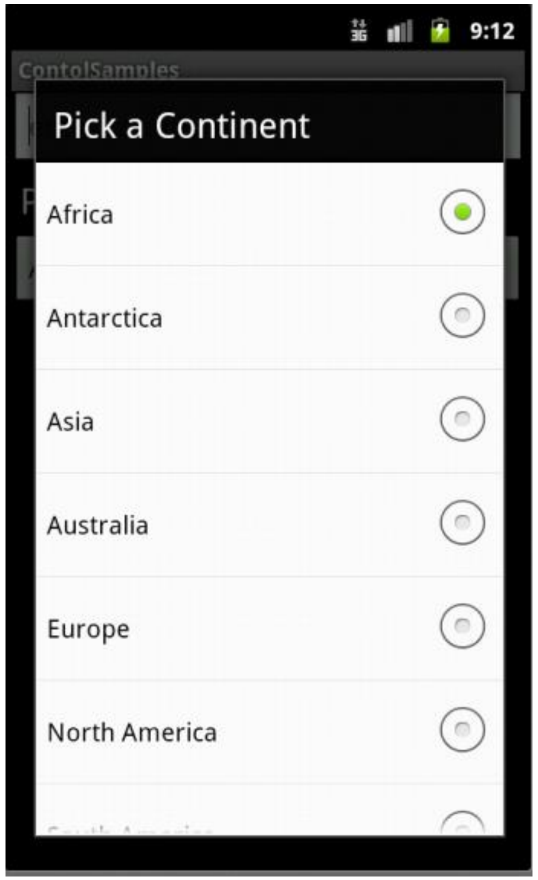
TimePicker



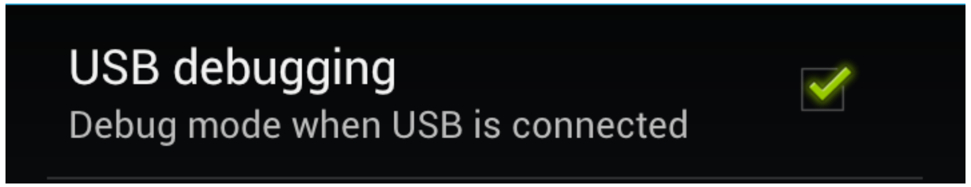
DatePicker

Spinner Controls

- user **must** select one of a set of choices



Checkbox

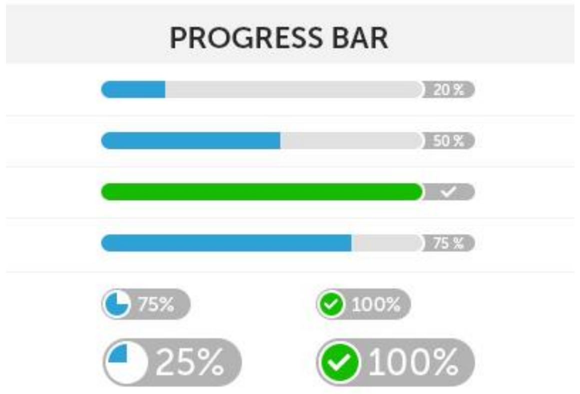

USB debugging
Debug mode when USB is connected

- Checkbox has 2 states: checked and unchecked
- XML code to create Checkbox

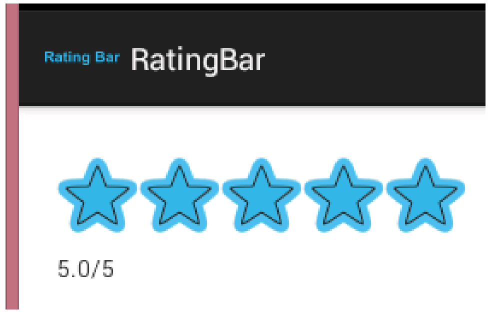
```
<?xml version="1.0" encoding="utf-8"?>
<CheckBox xmlns:android="http://schemas.android.com/apk/res/android"
  android:id="@+id/check"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:text="@string/unchecked"/>
```

Other Indicators & More Widgets

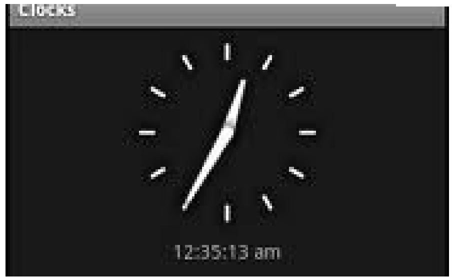
- ProgressBar



- RatingBar



- Chronometer
- DigitalClock
- AnalogClock

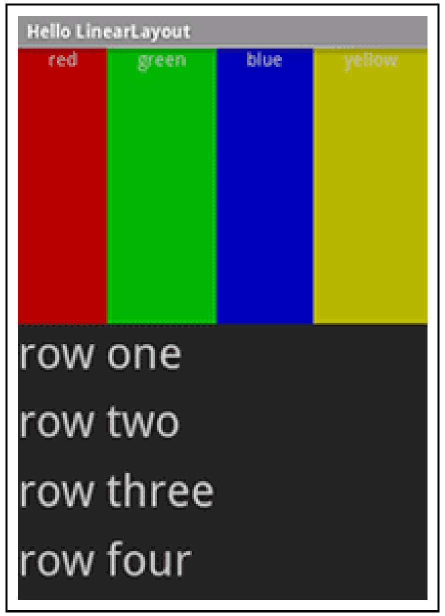


Android Layouts in XML

Android UI using XML Layouts

- Layout? Pattern in which multiple widgets are arranged
- Layouts contain widgets
- In Android internal classes, widget is child of layout
- Layouts (XML files) stored in **res/layout**

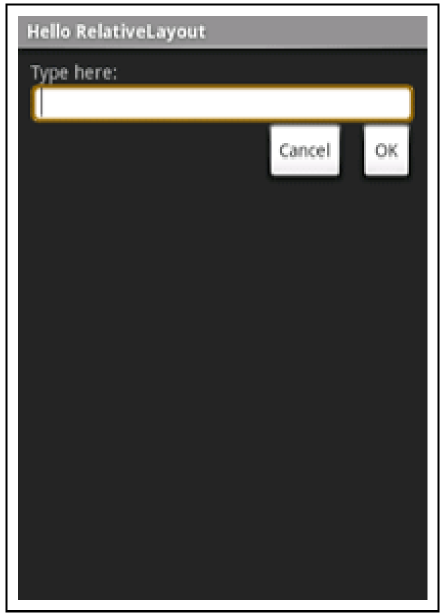
LinearLayout



Hello LinearLayout

row one
row two
row three
row four

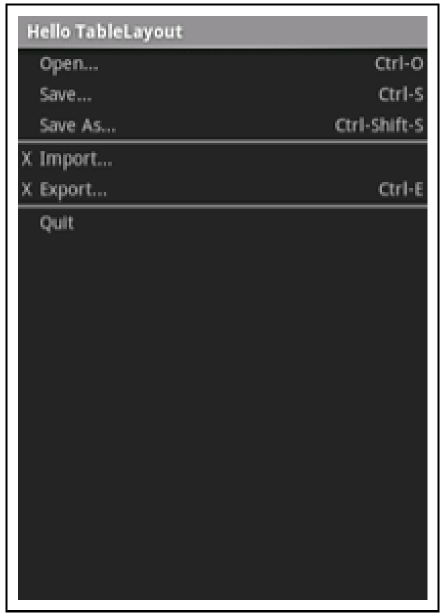
RelativeLayout



Hello RelativeLayout

Type here:

TableLayout



Hello TableLayout

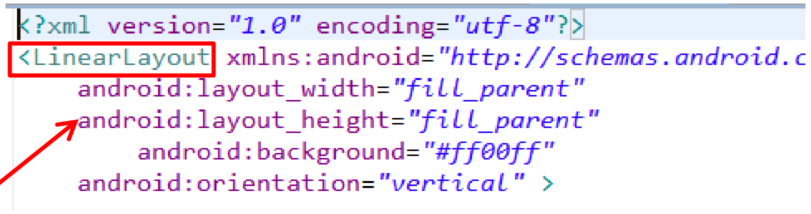
	Open...	Ctrl-O
	Save...	Ctrl-S
	Save As...	Ctrl-Shift-S
X	Import...	
X	Export...	Ctrl-E
	Quit	

Some Layouts

- FrameLayout,
- LinearLayout,
- TableLayout,
- GridLayout,
- RelativeLayout,
- ListView,
- GridView,
- ScrollView,
- DrawerLayout,
- ViewPager

LinearLayout

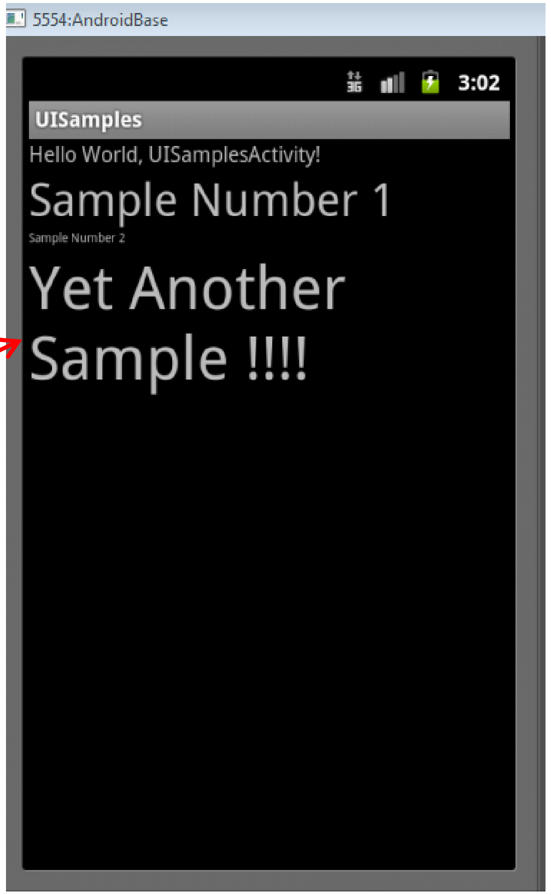
- aligns child elements (e.g. buttons, text boxes, pictures, etc.) in one direction
- Example:



```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.c
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
        android:background="#ff00ff"
    android:orientation="vertical" >
```

Layout properties

- orientation attribute defines direction (vertical or horizontal):
  - E.g. android:orientation="*vertical*"



5554:AndroidBase

3:02

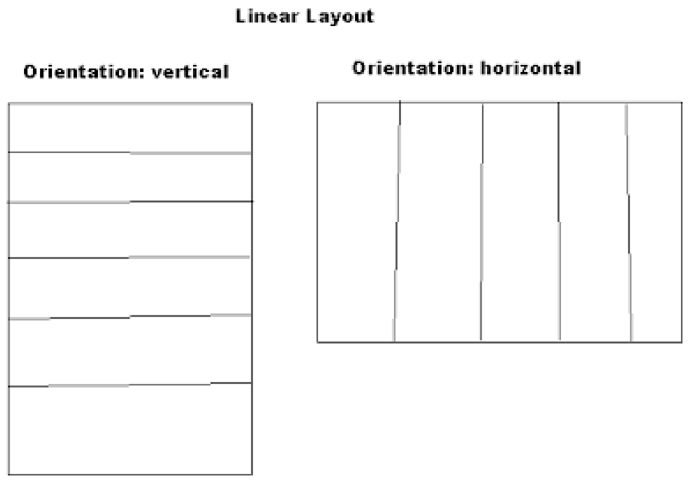
UISamples

Hello World, UISamplesActivity!

Sample Number 1

Sample Number 2

Yet Another Sample !!!!

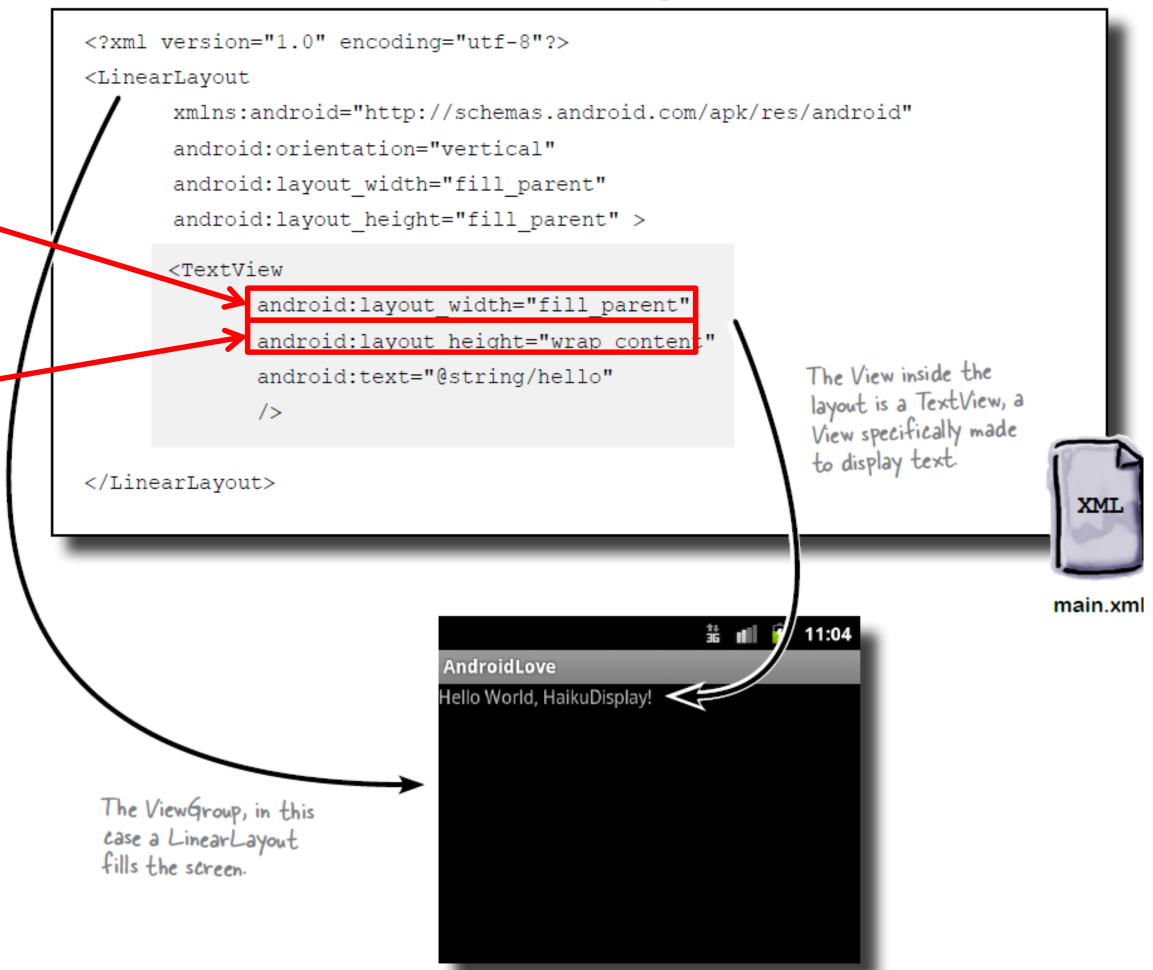


Layout Width and Height Attributes

- **wrap_content:** widget as wide/high as its content (e.g. text)
- **match_parent:** widget as wide/high as its parent layout box
- **fill_parent:** older form of **match_parent**

Text widget width should be as wide as Its parent (the layout)

Text widget height should Be as wide as the content (text)



```
<?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" >

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

</LinearLayout>
```

The View inside the layout is a TextView, a View specifically made to display text.
XML
main.xml
11:04
AndroidLove
Hello World, HaikuDisplay!
The ViewGroup, in this case a LinearLayout fills the screen.

<u>Hierarchy</u>

Screen (Hardware)

↑

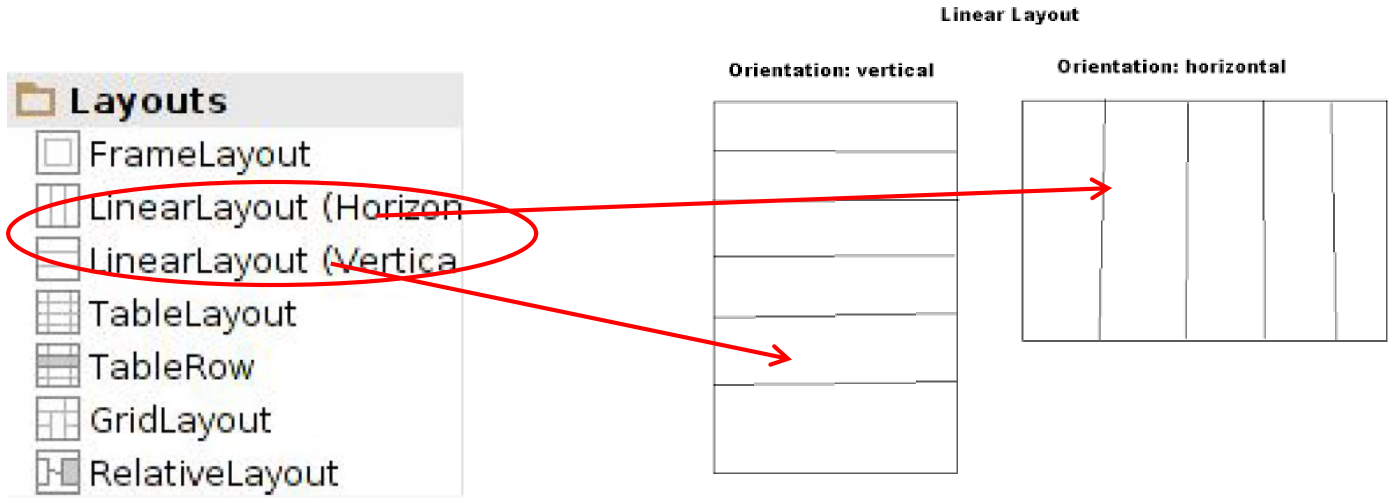
Linear Layout

↑

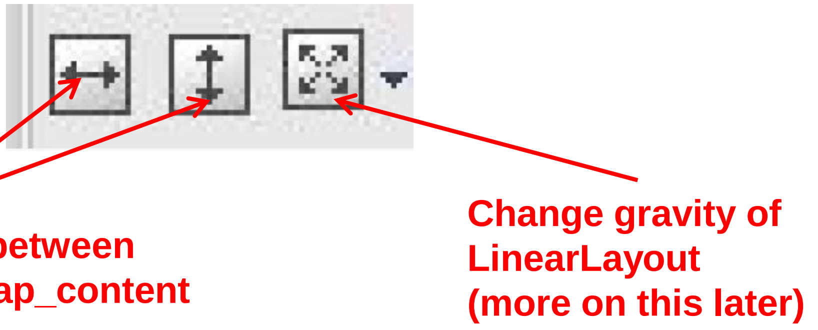
TextView

LinearLayout in Android Studio

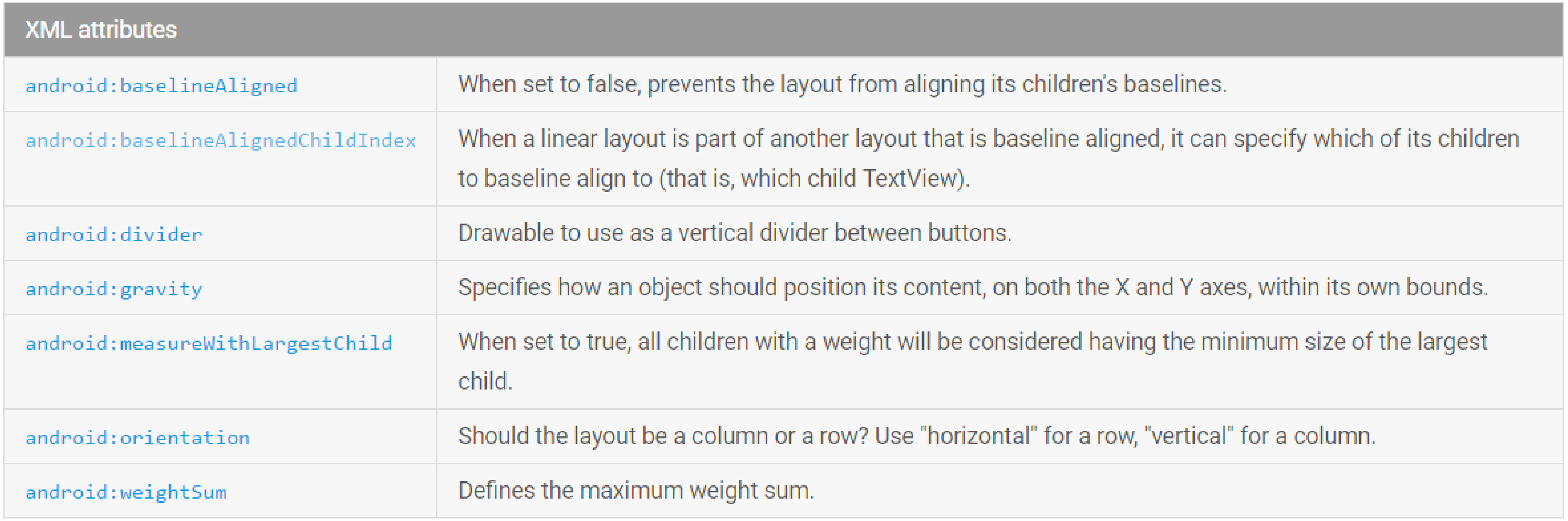
- LinearLayout in Android Studio Graphical Layout Editor



- After selecting LinearLayout, toolbars buttons to set parameters


Toggle width, height between match_parent and wrap_content
Change gravity of LinearLayout (more on this later)

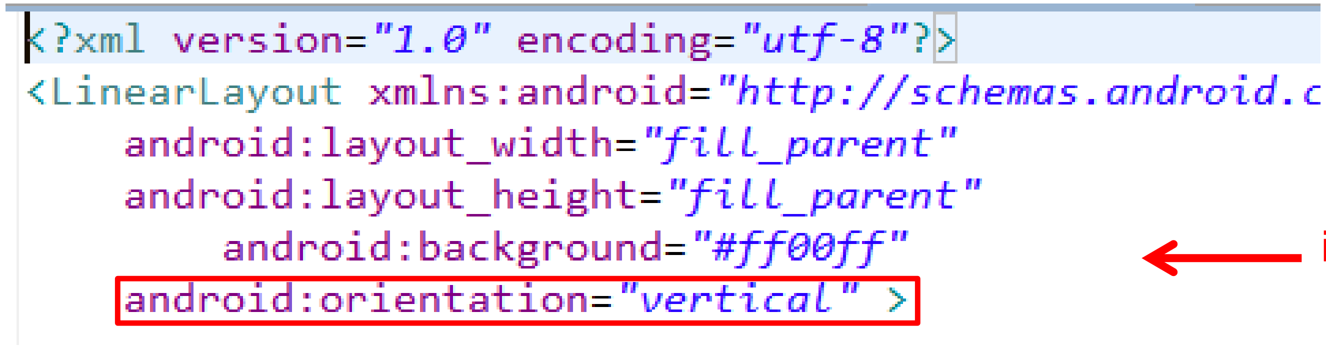
LinearLayout Attributes



XML attributes	
`android:baselineAligned`	When set to false, prevents the layout from aligning its children's baselines.
`android:baselineAlignedChildIndex`	When a linear layout is part of another layout that is baseline aligned, it can specify which of its children to baseline align to (that is, which child TextView).
`android:divider`	Drawable to use as a vertical divider between buttons.
`android:gravity`	Specifies how an object should position its content, on both the X and Y axes, within its own bounds.
`android:measureWithLargestChild`	When set to true, all children with a weight will be considered having the minimum size of the largest child.
`android:orientation`	Should the layout be a column or a row? Use "horizontal" for a row, "vertical" for a column.
`android:weightSum`	Defines the maximum weight sum.

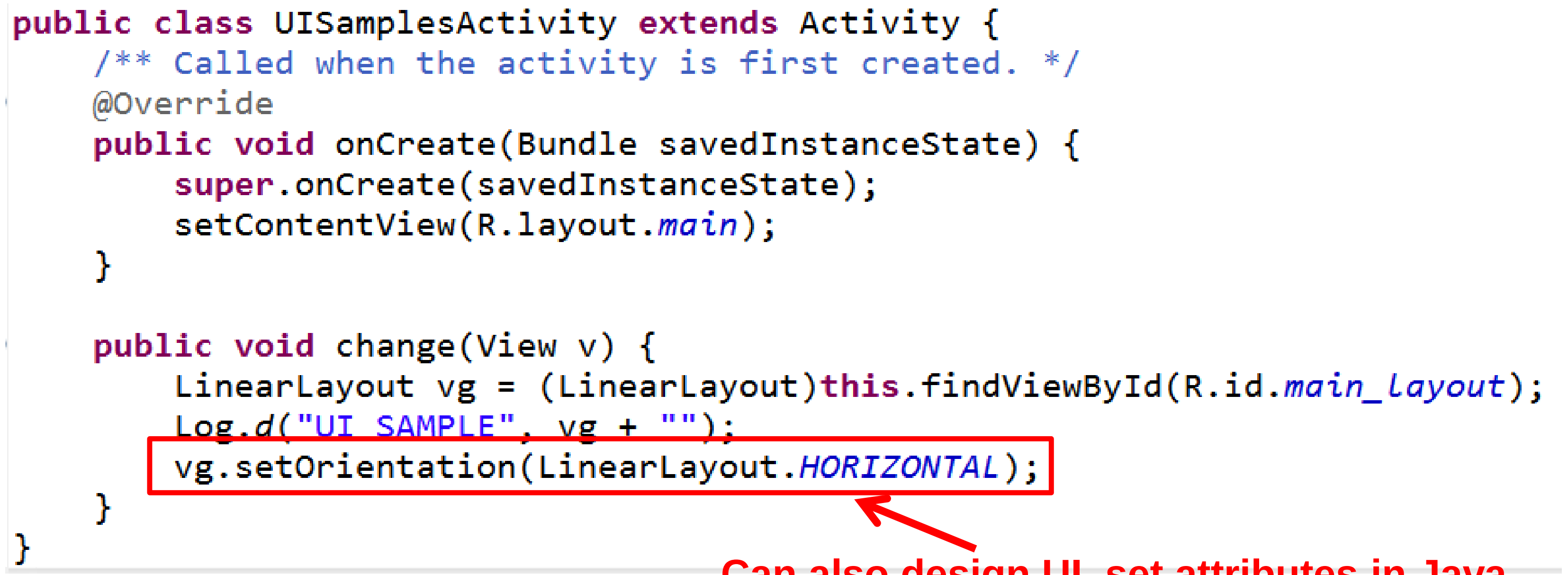
Ref: https://developer.android.com/reference/android/widget/LinearLayout

Setting Attributes



```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.c
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
        android:background="#ff00ff"
    android:orientation="vertical" >
```

in layout xml file



```java
public class UISamplesActivity extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }

    public void change(View v) {
        LinearLayout vg = (LinearLayout)this.findViewById(R.id.main_layout);
        Log.d("UI SAMPLE", vg + "");
        vg.setOrientation(LinearLayout.HORIZONTAL);
    }
}
```

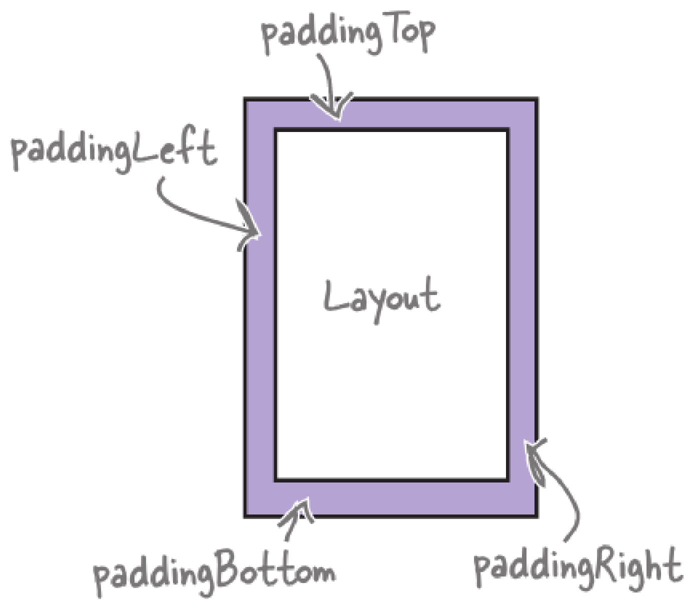
Can also design UI, set attributes in Java program (e.g. ActivityMain.java) (More later)

Adding Padding

- Paddings sets space between layout sides and its parent (e.g. the screen)

```
<RelativeLayout ...
    android:paddingBottom="16dp"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp">
    ...
</RelativeLayout>
```

Add padding of 16dp.

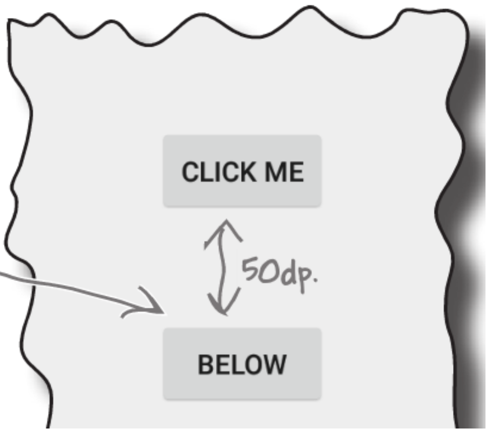


Setting Margins

- Can increase gap (margin) between adjacent widgets
- E.g. To add margin between two buttons, in declaration of bottom button

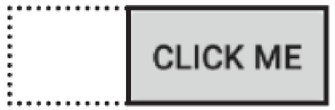
```
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/button_click_me"
        android:layout_below="@+id/button_click_me"
        android:layout_marginTop="50dp"
        android:text="@string/button_below" />
</RelativeLayout>
```

Adding a margin to the top of the bottom button adds extra space between the two views.

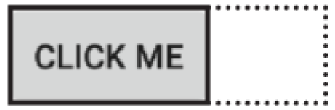


- Other options

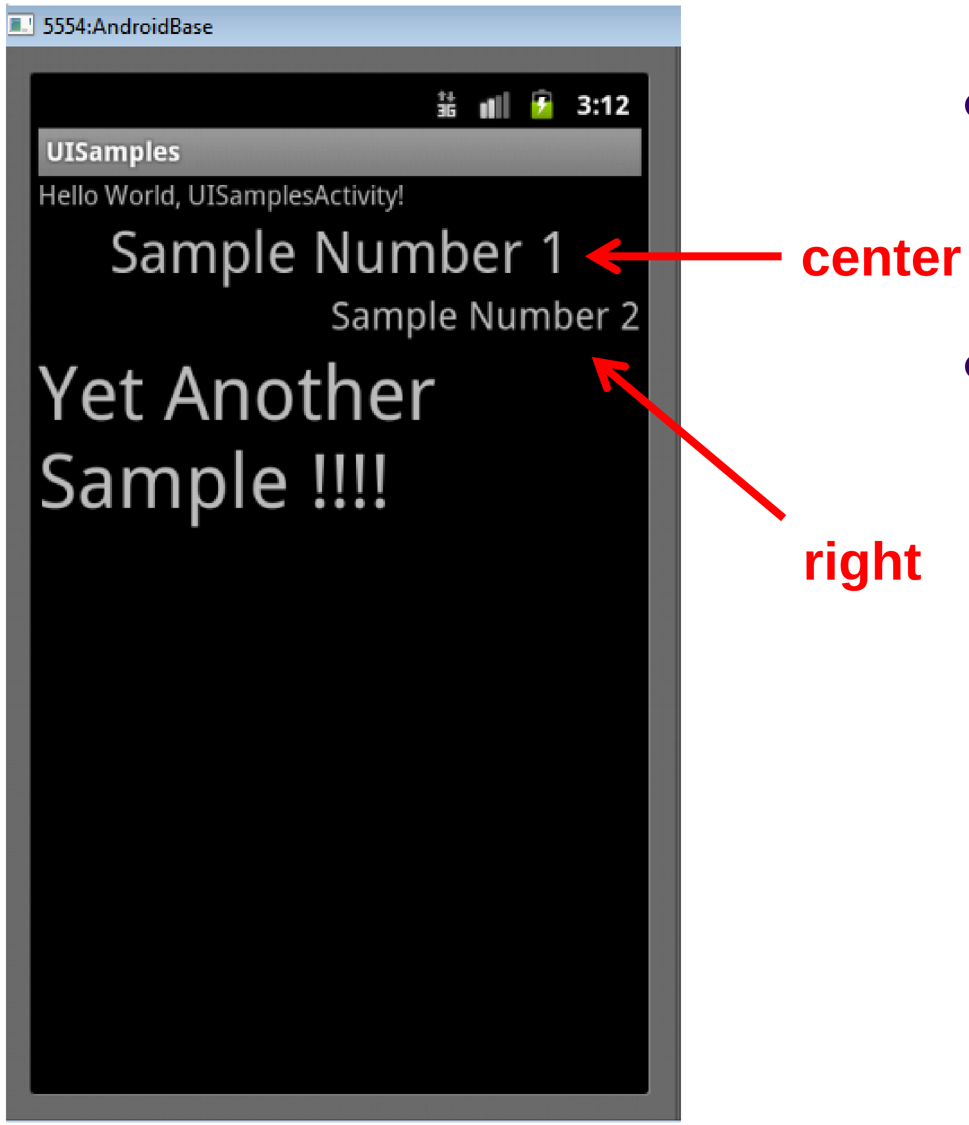
android:layout_marginLeft



android:layout_marginRight



Gravity Attribute


5554:AndroidBase

3:12

UISamples

Hello World, UISamplesActivity!

Sample Number 1

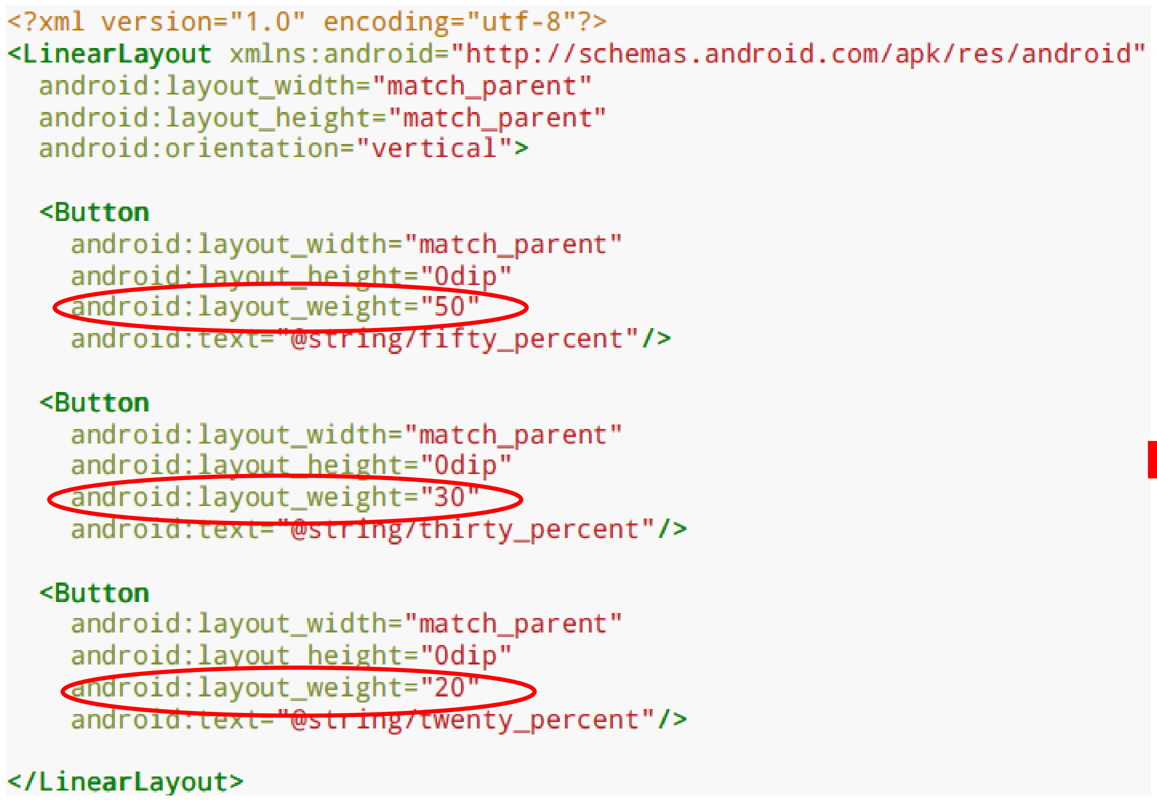
Sample Number 2

Yet Another Sample !!!!

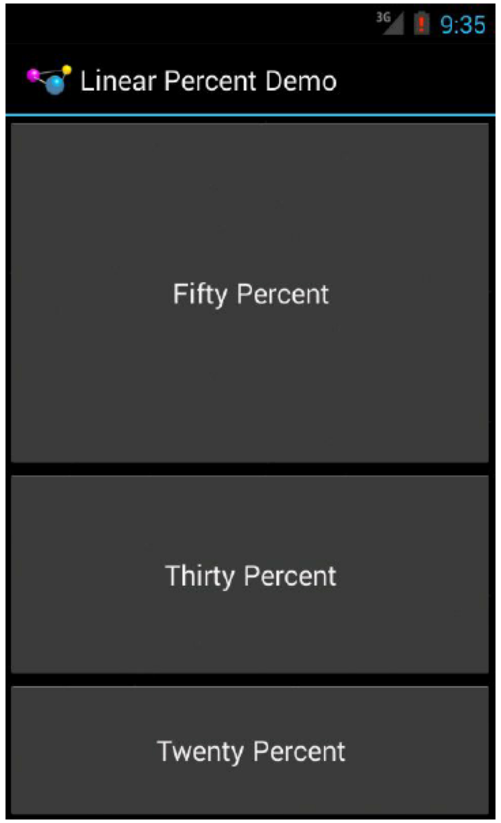
- By default, linearlayout left- and top-aligned
- Gravity attribute changes alignment :
  - e.g. android:gravity = “right”

Linear Layout Weight Attribute

- Specifies "importance“, larger weights takes up more space
- Can set width, height = 0 then
  - weight = percent of height/width you want element to cover



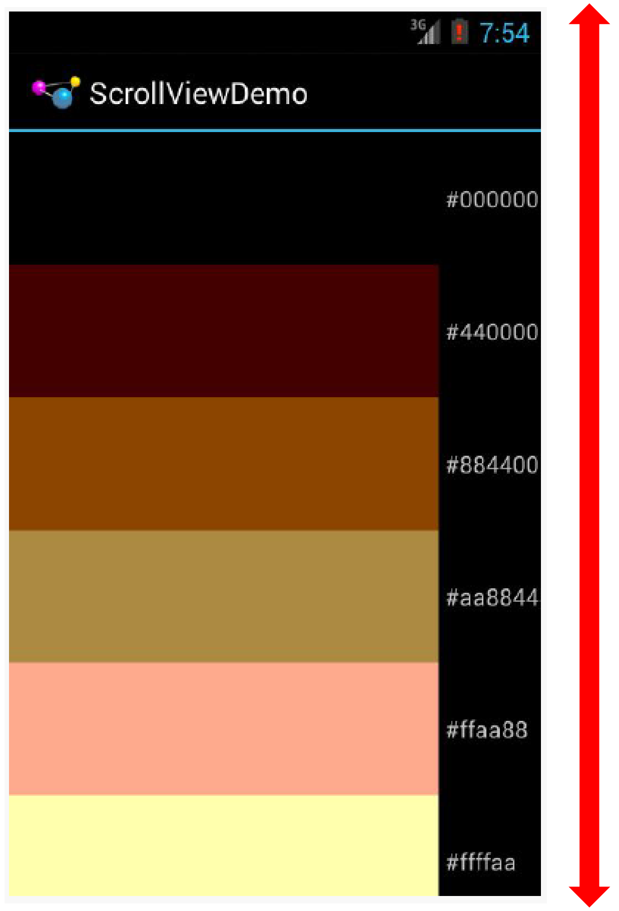
```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical">

  <Button
    android:layout_width="match_parent"
    android:layout_height="0dip"
    android:layout_weight="50"
    android:text="@string/fifty_percent"/>

  <Button
    android:layout_width="match_parent"
    android:layout_height="0dip"
    android:layout_weight="30"
    android:text="@string/thirty_percent"/>

  <Button
    android:layout_width="match_parent"
    android:layout_height="0dip"
    android:layout_weight="20"
    android:text="@string/twenty_percent"/>

</LinearLayout>
```



Scrolling

- Phone screens are small, scrolling content helps
- Examples: Scroll through
  - large image
  - Linear Layout with lots of elements
- Views for Scrolling:
  - **ScrollView** for vertical scrolling
  - **HorizontalScrollView**
- Rules:
  - Only one direct child View
  - Child could have many children of its own

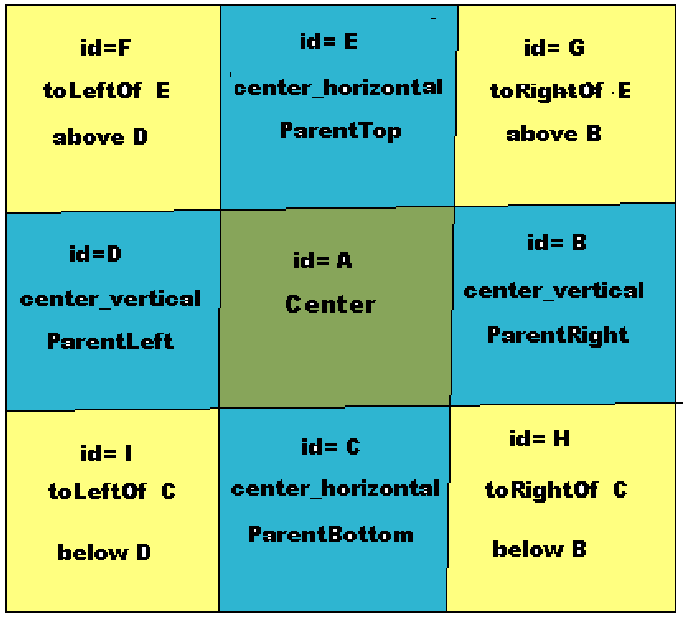
```
<ScrollView
    ...>
    <LinearLayout>
       ....
       <!-- you can have as many Views in here as you want -->
    </LinearLayout>
</ScrollView>
```

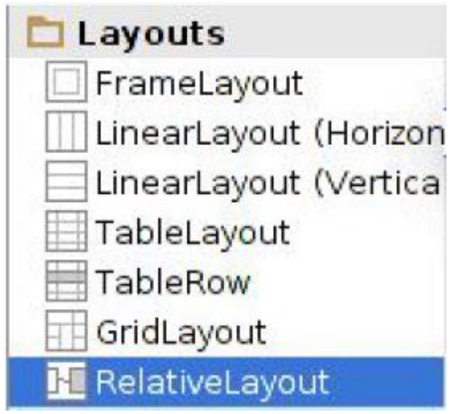


RelativeLayout

- First element listed is placed in "center"
- Positions of children specified relative to parent or to each other.

Relative Layout

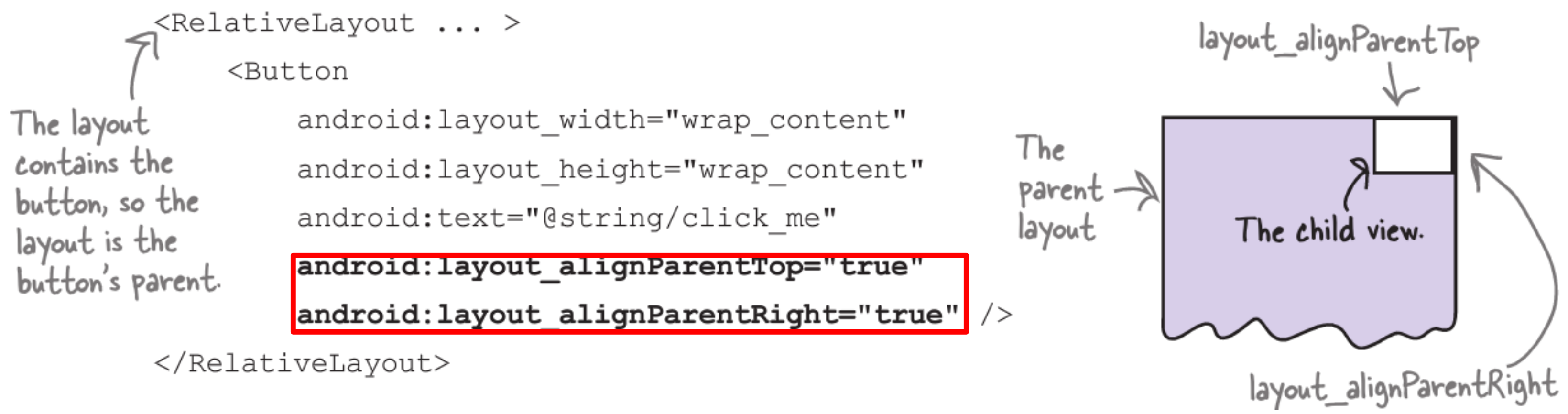




RelativeLayout available
In Android Studio palette

Positioning Views Relative to Parent Layout

- Position a view (e.g. button, TextView) relative to its parent
- Example: Button aligned to top, right in a Relative Layout

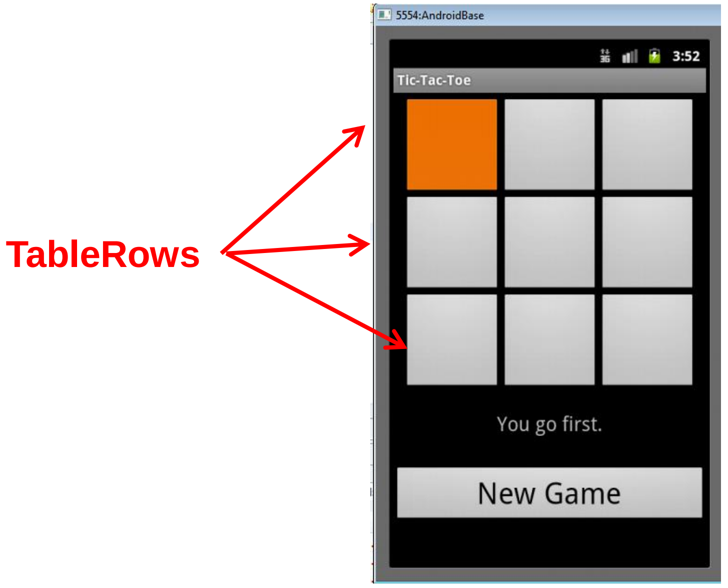


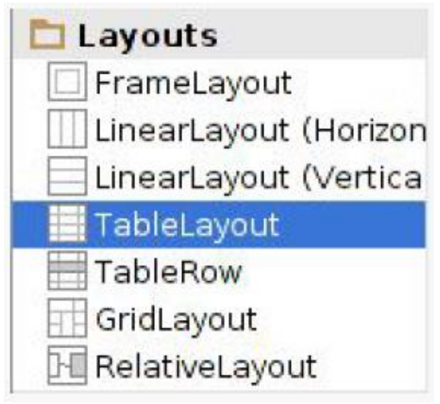
```
<RelativeLayout ... >
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/click_me"
        android:layout_alignParentTop="true"
        android:layout_alignParentRight="true" />
</RelativeLayout>
```

See Head First Android Development (2nd edition) page 169-220 for more examples

Table Layout

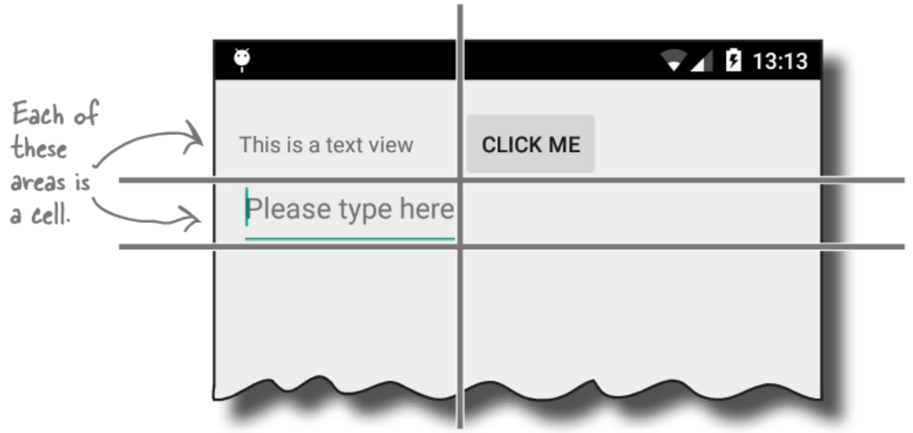
- Specify number of rows and columns of views.
- Available in Android Studio palette





GridLayout

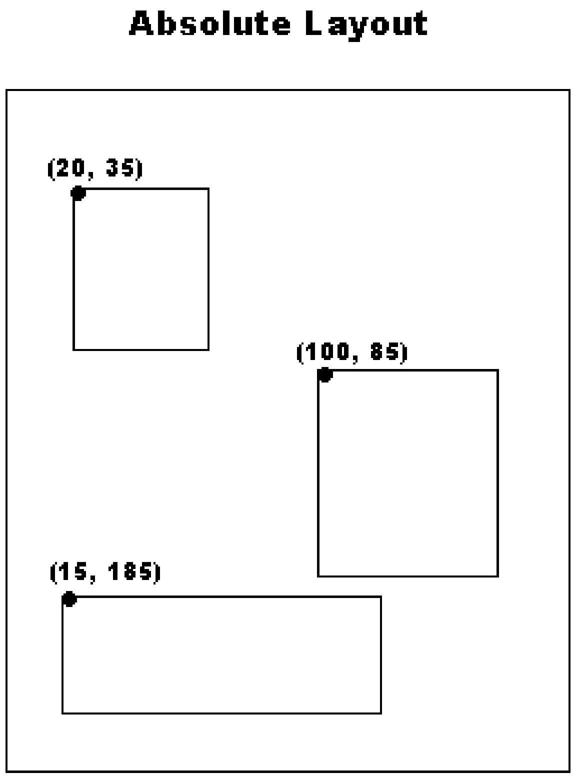
- In TableLayout, child views can span multiple columns only
- In GridLayout, child views/controls can span multiple rows **AND** columns



- See section “GridLayout Displays Views in a Grid” in Head First Android Development 2nd edition (pg 824)

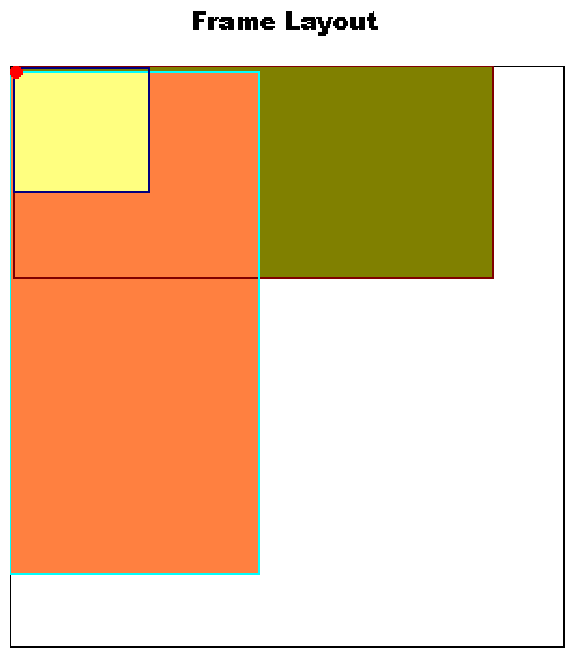
Absolute Layout

- Allows specification of exact x,y coordinates of layout’s children.

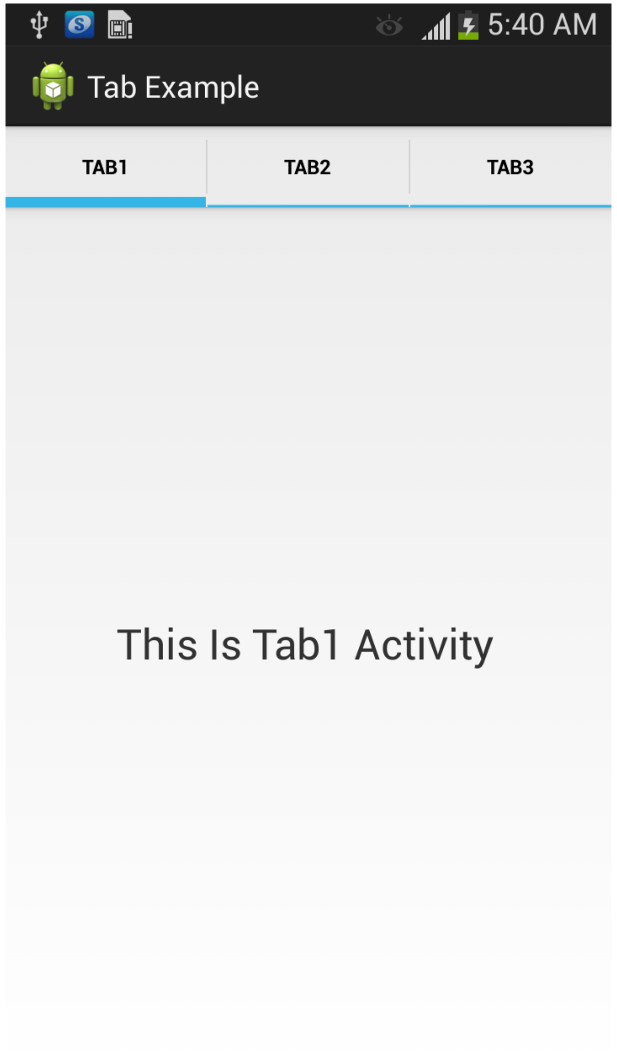


FrameLayout

- child elements pinned to top left corner of layout
- adding a new element / child draws over the last one



Other Layouts: Tabbed Layouts

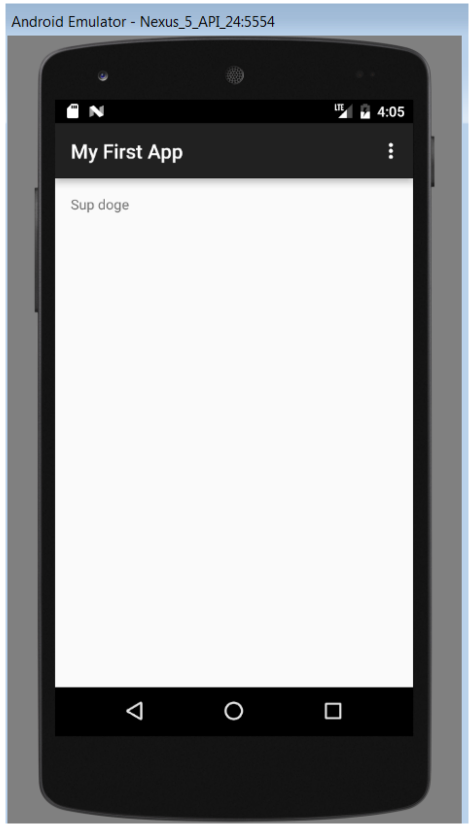


This Is Tab1 Activity

Android Example: My First App (Ref: Head First Android)

My First App

- Hello World program in Head First Android Development (Chapter 1)
- Creates app, types “Sup doge” in a TextView



References

- Busy Coder's guide to Android version 4.4
- CS 65/165 slides, Dartmouth College, Spring 2014
- CS 371M slides, U of Texas Austin, Spring 2014
- Android App Development for Beginners videos by Bucky Roberts (thenewboston)
- Head First Android
- Android Nerd Ranch, Third Edition

References

- Android App Development for Beginners videos by Bucky Roberts (thenewboston)
- Ask A Dev, Android Wear: What Developers Need to Know, https://www.youtube.com/watch?v=zTS2NZpLyQg
- Ask A Dev, Mobile Minute: What to (Android) Wear, https://www.youtube.com/watch?v=n5Yjzn3b_aQ
- Busy Coder's guide to Android version 4.4
- CS 65/165 slides, Dartmouth College, Spring 2014
- CS 371M slides, U of Texas Austin, Spring 2014