Mobile Programming

[week 03]

[Designing User Interface]

Hilmy A. T.

hilmi.tawakal@gmail.com

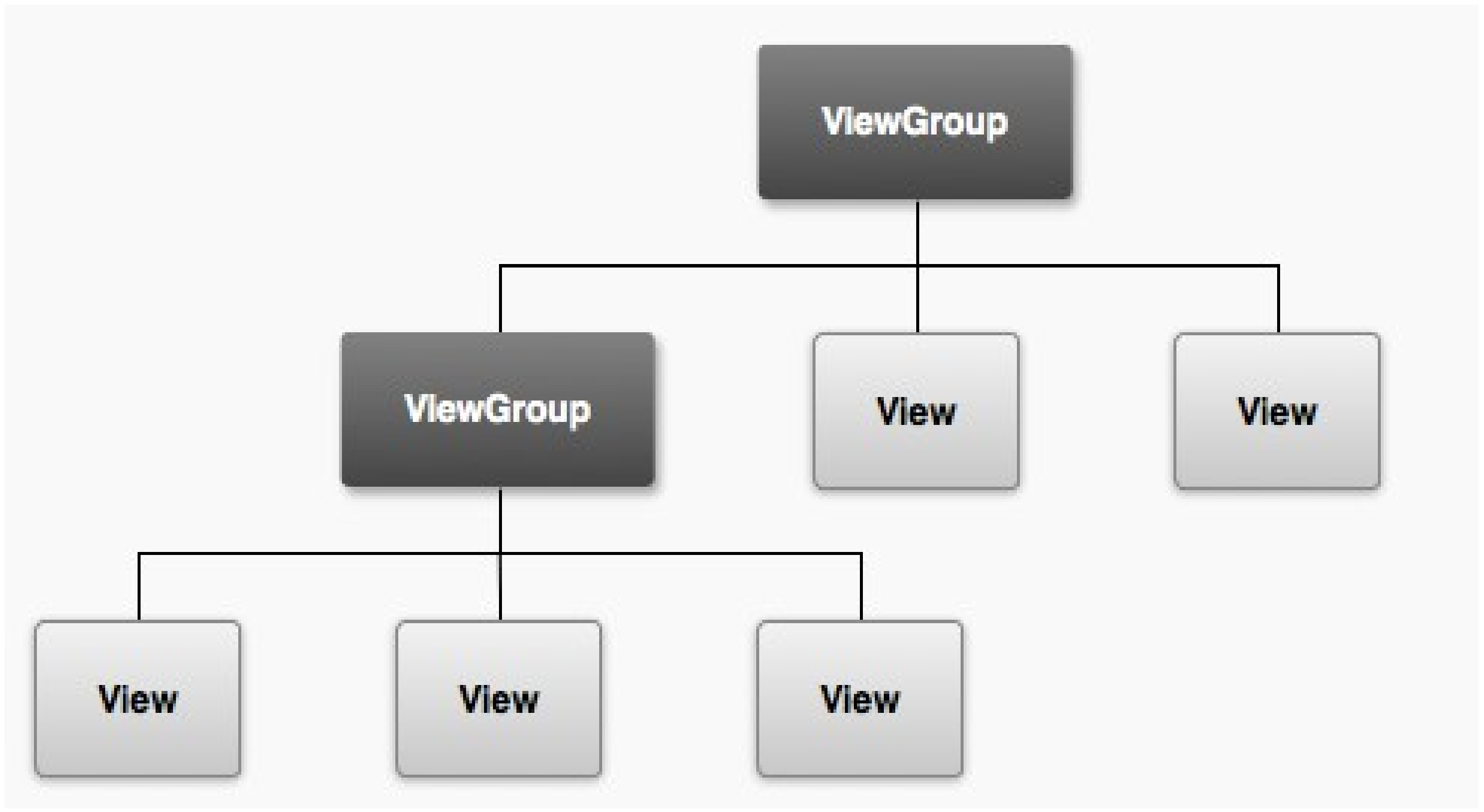
UI Overview

- All user interface elements in an Android app are built using View and ViewGroup objects.
- A View is an object that draws something on the screen that the user can interact with.
- A ViewGroup is an object that holds other View (and ViewGroup) objects in order to define the layout of the interface.

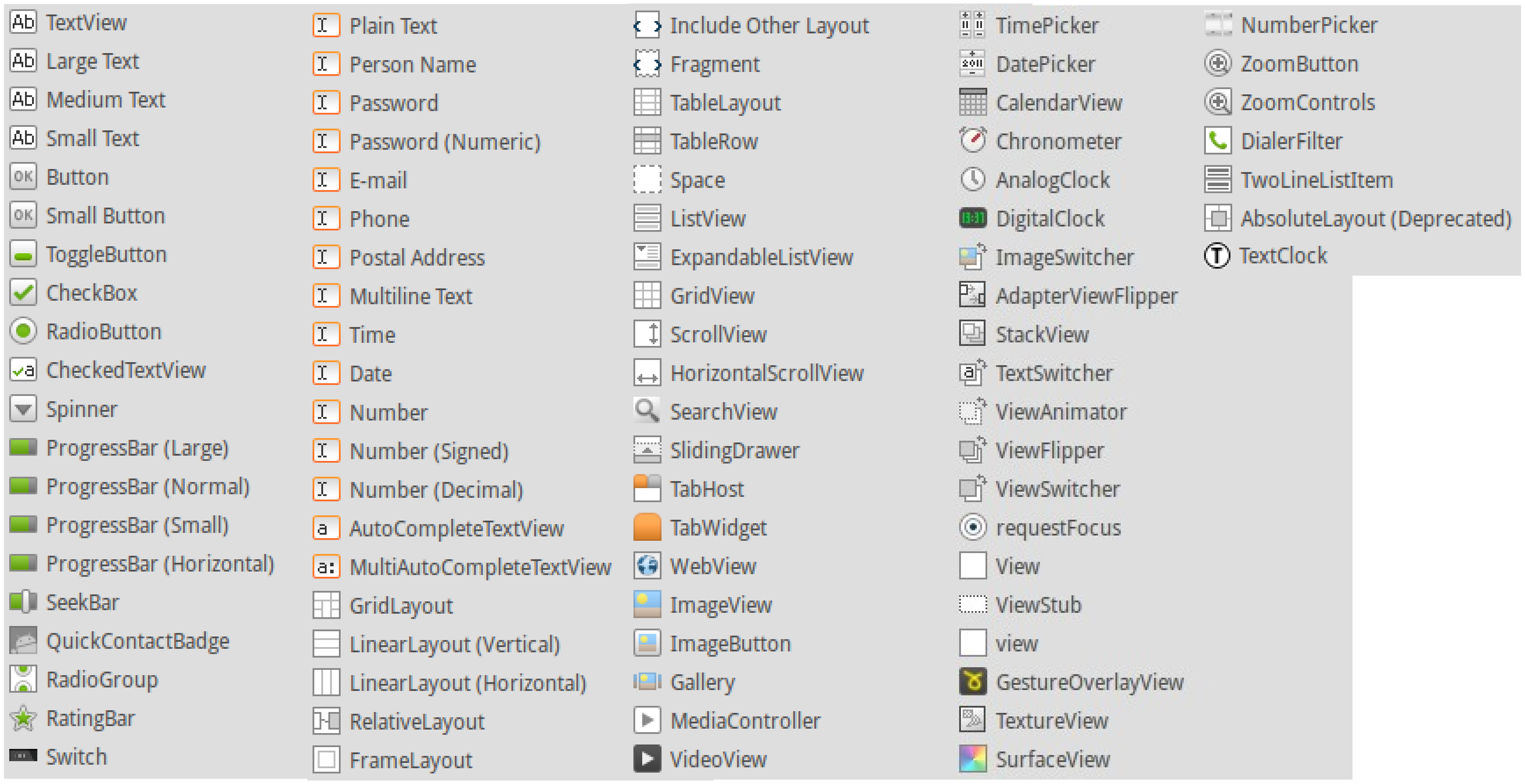
UI Layout

- The user interface for each component of your app is defined using a hierarchy of View and ViewGroup objects.
- Each view group is an invisible container that organizes child views.
- The child views may be input controls or other widgets that draw some part of the UI.

UI Layout



View & ViewGroup



TextView
Large Text
Medium Text
Small Text
Button
Small Button
ToggleButton
CheckBox
RadioButton
CheckedTextView
Spinner
ProgressBar (Large)
ProgressBar (Normal)
ProgressBar (Small)
ProgressBar (Horizontal)
SeekBar
QuickContactBadge
RadioGroup
RatingBar
Switch

Plain Text
Person Name
Password
Password (Numeric)
E-mail
Phone
Postal Address
Multiline Text
Time
Date
Number
Number (Signed)
Number (Decimal)
AutoCompleteTextView
MultiAutoCompleteTextView
GridLayout
LinearLayout (Vertical)
LinearLayout (Horizontal)
RelativeLayout
FrameLayout

Include Other Layout
Fragment
TableLayout
TableRow
Space
ListView
ExpandableListView
GridView
ScrollView
HorizontalScrollView
SearchView
SlidingDrawer
TabHost
TabWidget
WebView
ImageView
ImageButton
Gallery
MediaController
VideoView

TimePicker
DatePicker
CalendarView
Chronometer
AnalogClock
DigitalClock
ImageSwitcher
AdapterViewFlipper
StackView
TextSwitcher
ViewAnimator
ViewFlipper
ViewSwitcher
requestFocus
View
ViewStub
view
GestureOverlayView
TextureView
SurfaceView

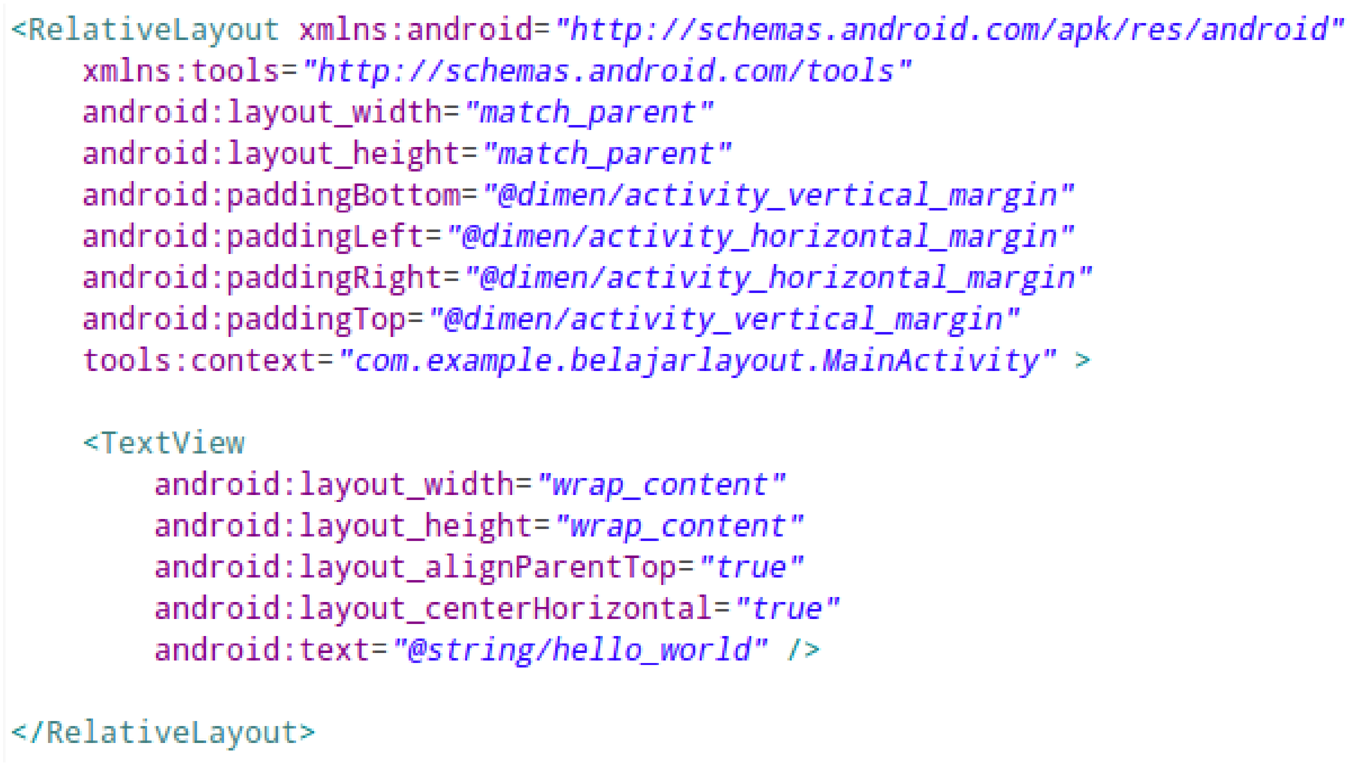
NumberPicker
ZoomButton
ZoomControls
DialerFilter
TwoLineListItem
AbsoluteLayout (Deprecated)
TextClock

Creating Layout

- To declare your layout, you can instantiate View objects in code and start building a tree.
- But the easiest and most effective way to define your layout is with an XML file.
- XML offers a human-readable structure for the layout, similar to HTML.

Creating Layout via XML

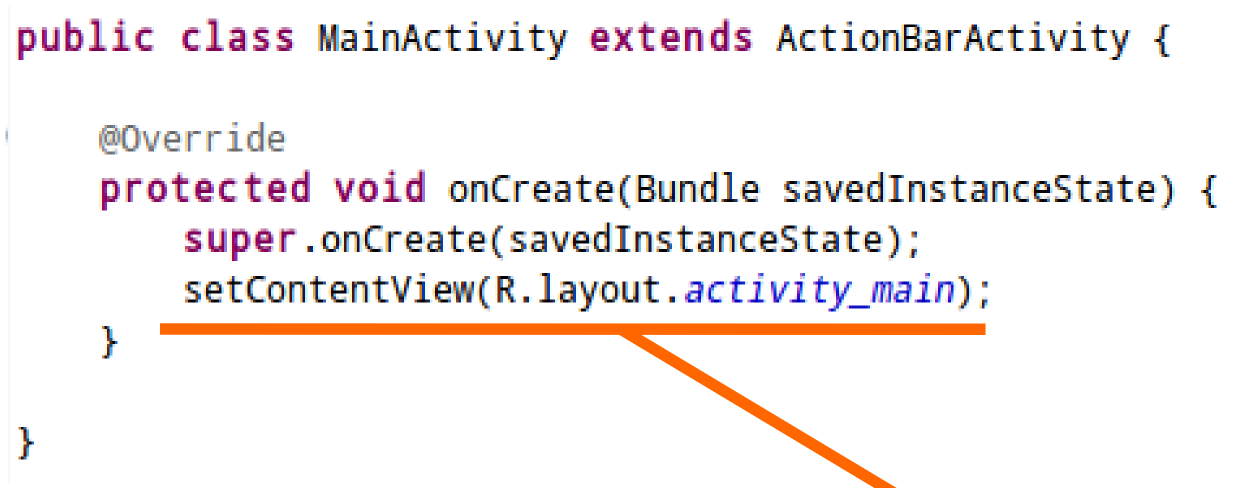
res/activity_main.xml



```
<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:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.belajarlayout.MainActivity" >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:text="@string/hello_world" />

</RelativeLayout>
```

Creating Layout via XML

MainActivity.java

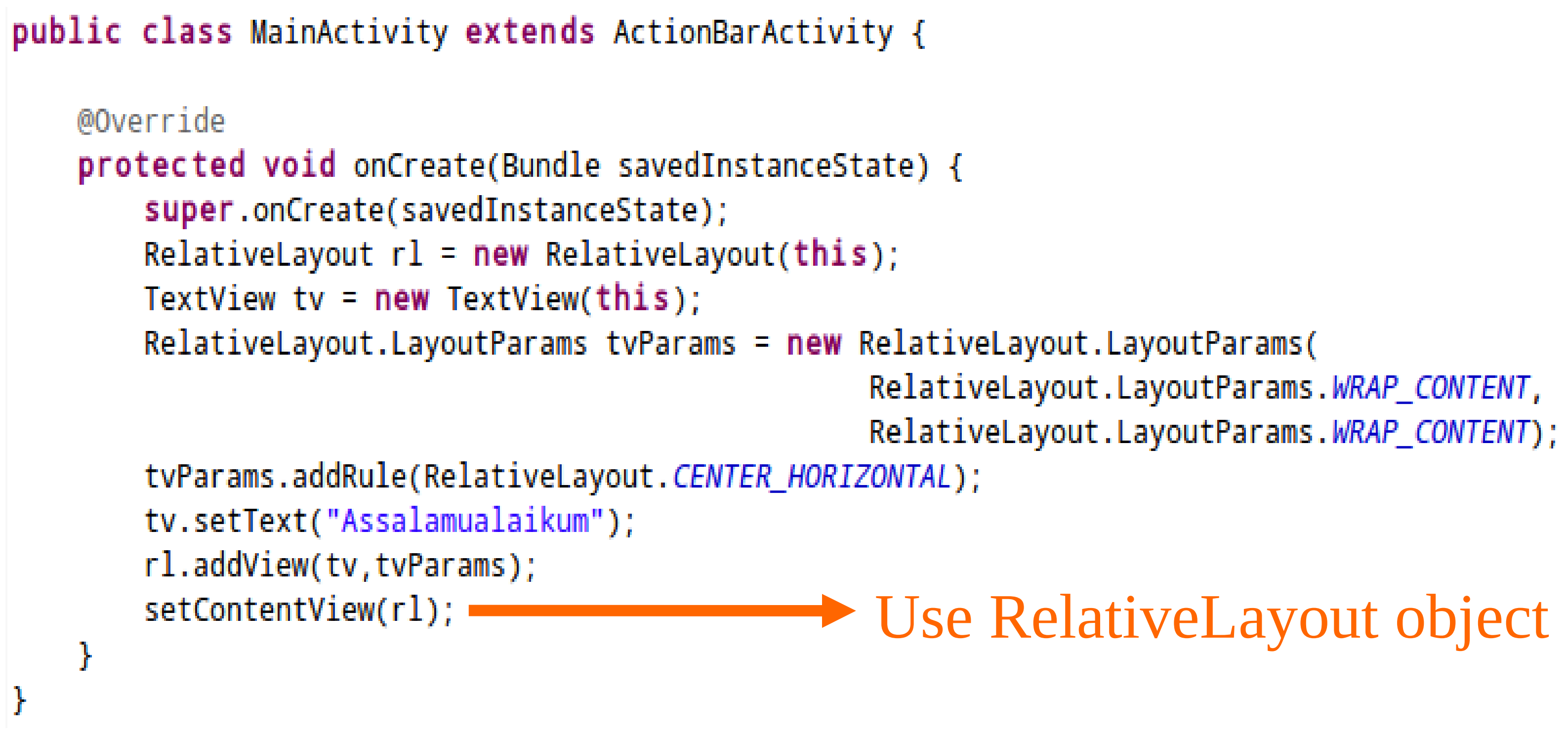


```
public class MainActivity extends ActionBarActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

}
```

Use activity_main.xml

Creating Layout via Java

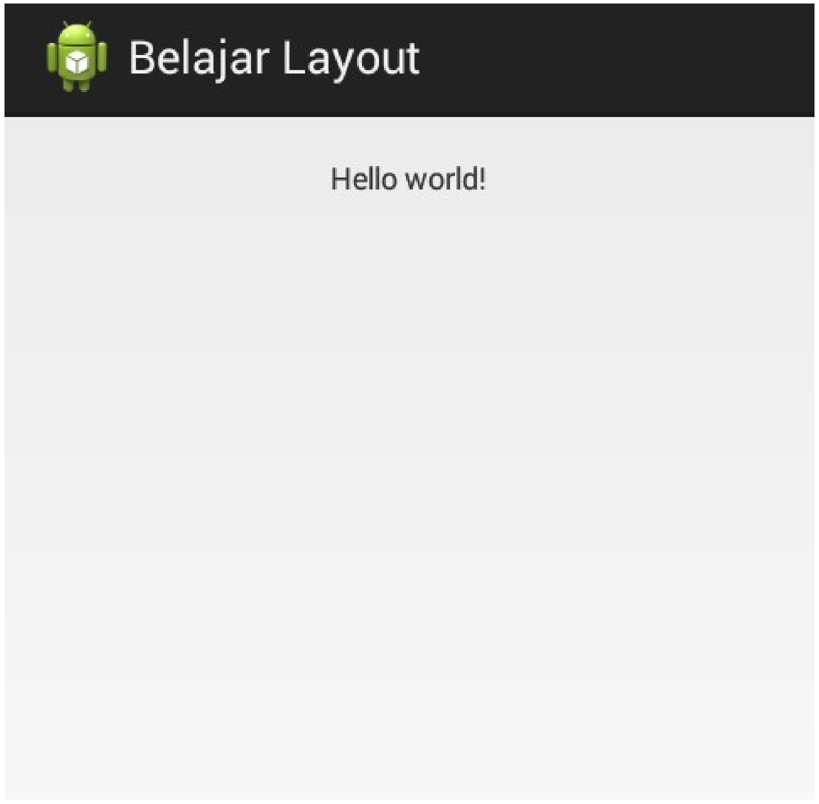
MainActivity.java



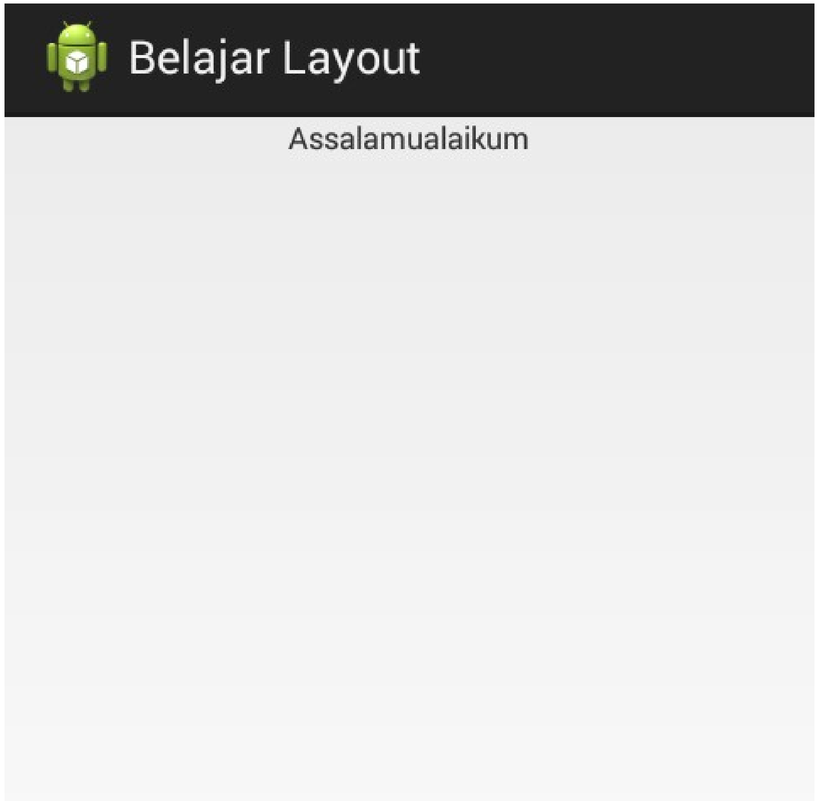
```
public class MainActivity extends ActionBarActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        RelativeLayout rl = new RelativeLayout(this);
        TextView tv = new TextView(this);
        RelativeLayout.LayoutParams tvParams = new RelativeLayout.LayoutParams(
                                                   RelativeLayout.LayoutParams.WRAP_CONTENT,
                                                   RelativeLayout.LayoutParams.WRAP_CONTENT);
        tvParams.addRule(RelativeLayout.CENTER_HORIZONTAL);
        tv.setText("Assalamualaikum");
        rl.addView(tv,tvParams);
        setContentView(rl);
    }
}
```

$\longrightarrow$ Use RelativeLayout object

Result


Belajar Layout

Hello world!


Belajar Layout

Assalamualaikum

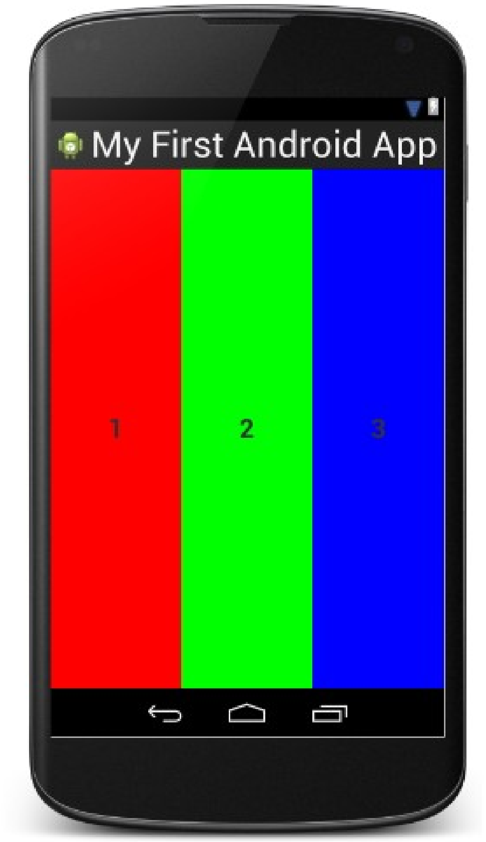
Commons Layout

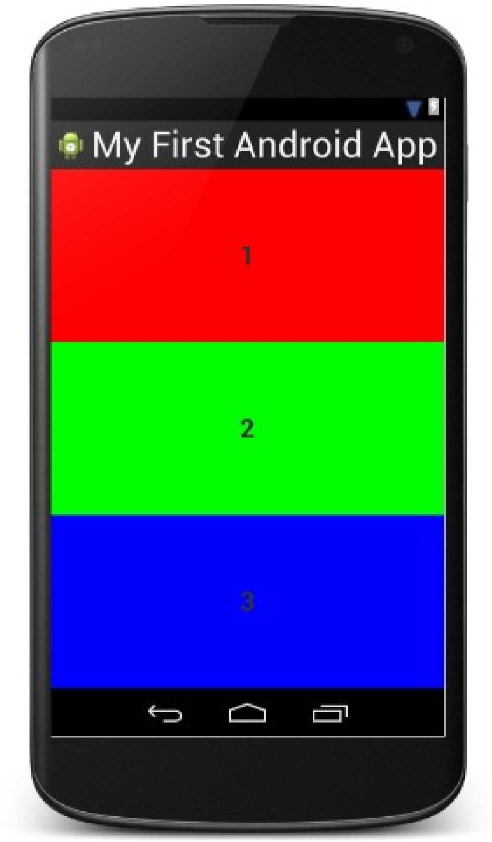
- Each subclass of the ViewGroup class provides a unique way to display the views you nest within it.
- Next are some of the more common layout types that are built into the Android platform.
- Note: Although you can nest one or more layouts within another layout to acheive your UI design, you should strive to keep your layout hierarchy as shallow as possible. Your layout draws faster if it has fewer nested layouts (a wide view hierarchy is better than a deep view hierarchy).

Linear Layout

- LinearLayout is a view group that aligns all children in a single direction, vertically or horizontally.
- It creates a scrollbar if the length of the window exceeds the length of the screen.
- You can specify the layout direction with the android:orientation attribute.

Linear Layout

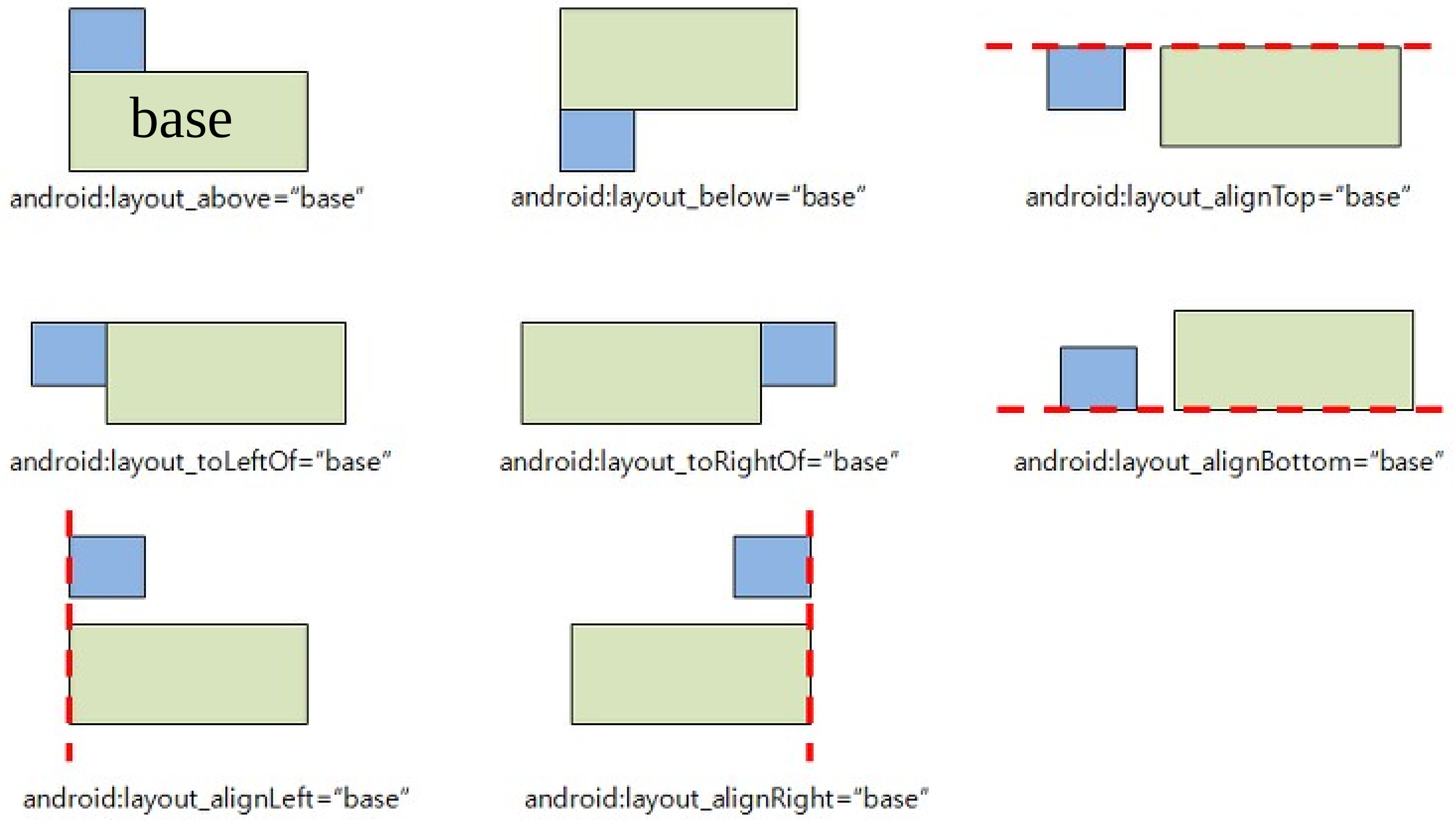




Relative Layout

- RelativeLayout is a view group that displays child views in relative positions.
- The position of each view can be specified as relative to sibling elements or in positions relative to the parent RelativeLayout area.
- A RelativeLayout is a very powerful utility for designing a user interface because it can eliminate nested view groups and keep your layout hierarchy flat, which improves performance.

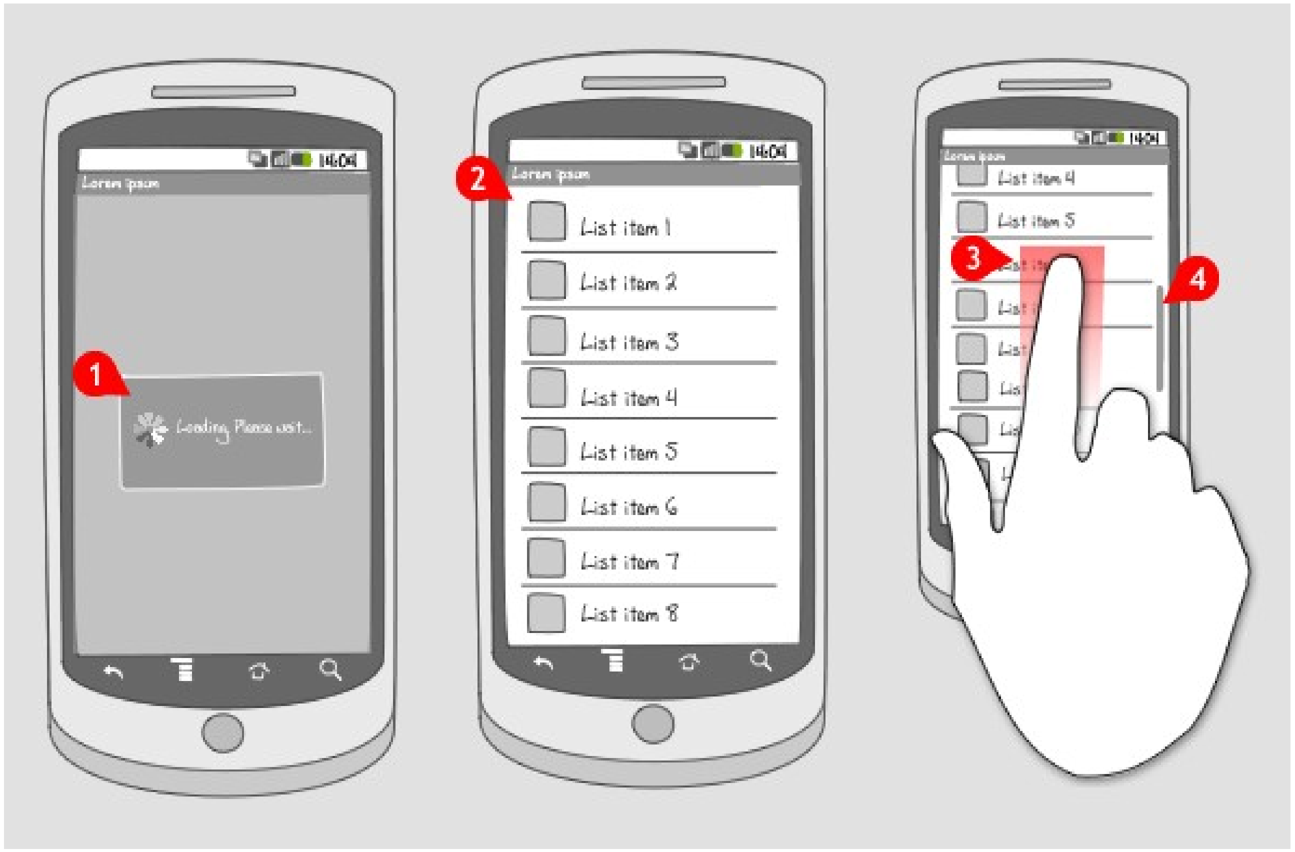
Relative Layout


base
android:layout_above="base"
android:layout_below="base"
android:layout_alignTop="base"
android:layout_toLeftOf="base"
android:layout_toRightOf="base"
android:layout_alignBottom="base"
android:layout_alignLeft="base"
android:layout_alignRight="base"

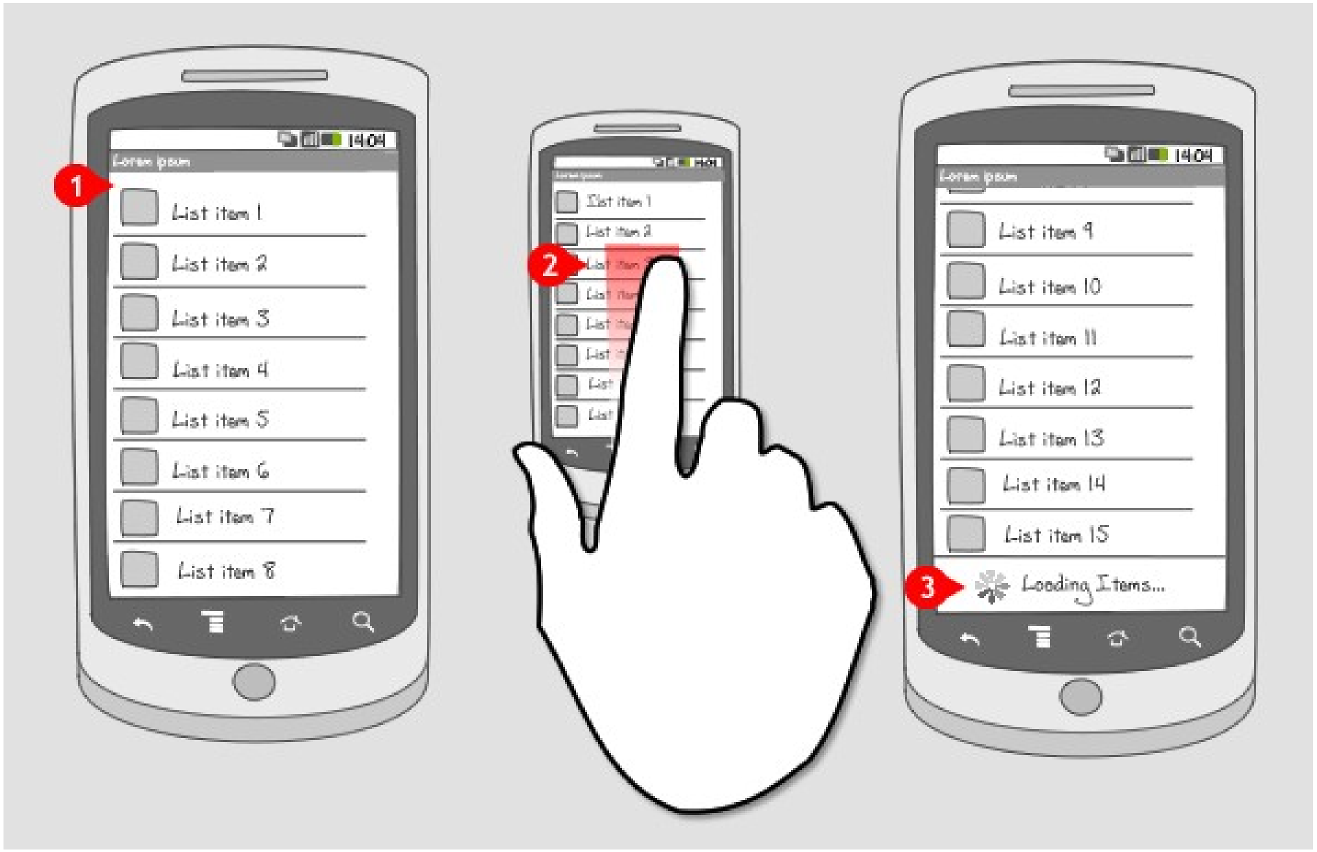
List View

- ListView is a view group that displays a list of scrollable items.
- The list items are automatically inserted to the list using an Adapter that pulls content from a source such as an array or database query and converts each item result into a view that's placed into the list.

Static List View

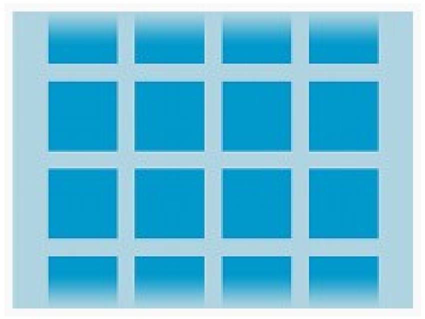


Dynamic List View

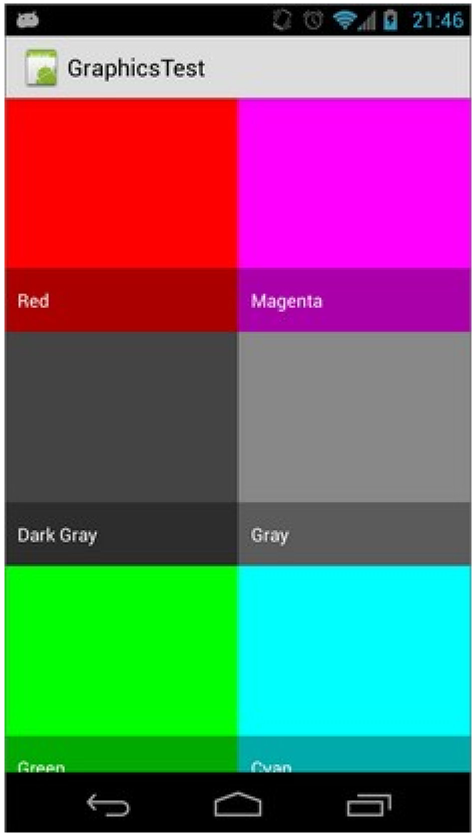


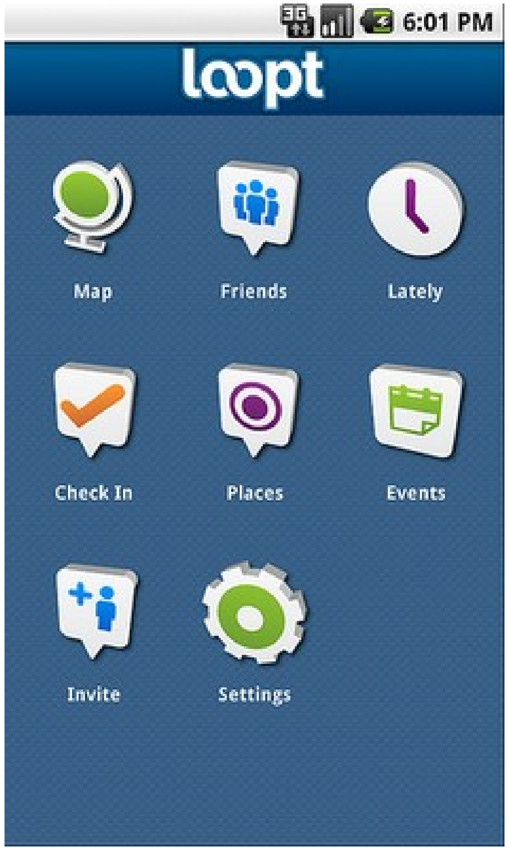
Grid View

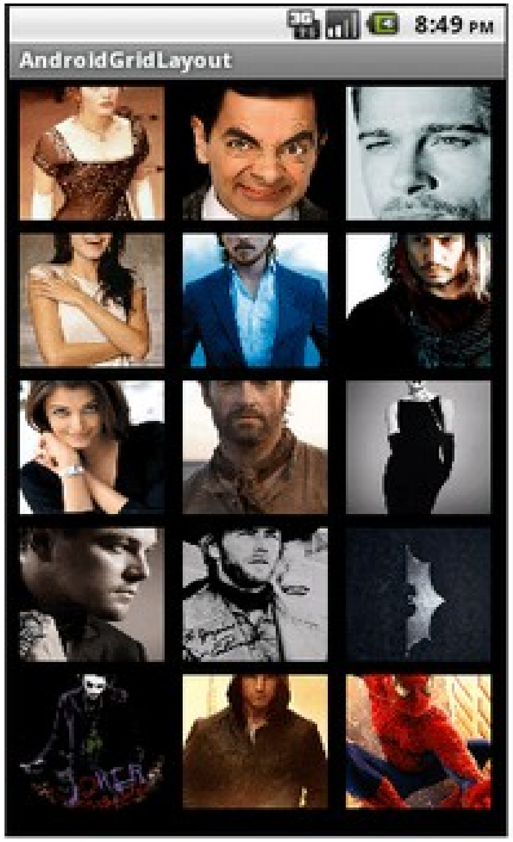
- GridView is a ViewGroup that displays items in a two-dimensional, scrollable grid.
- The grid items are automatically inserted to the layout using a ListAdapter.



Grid View

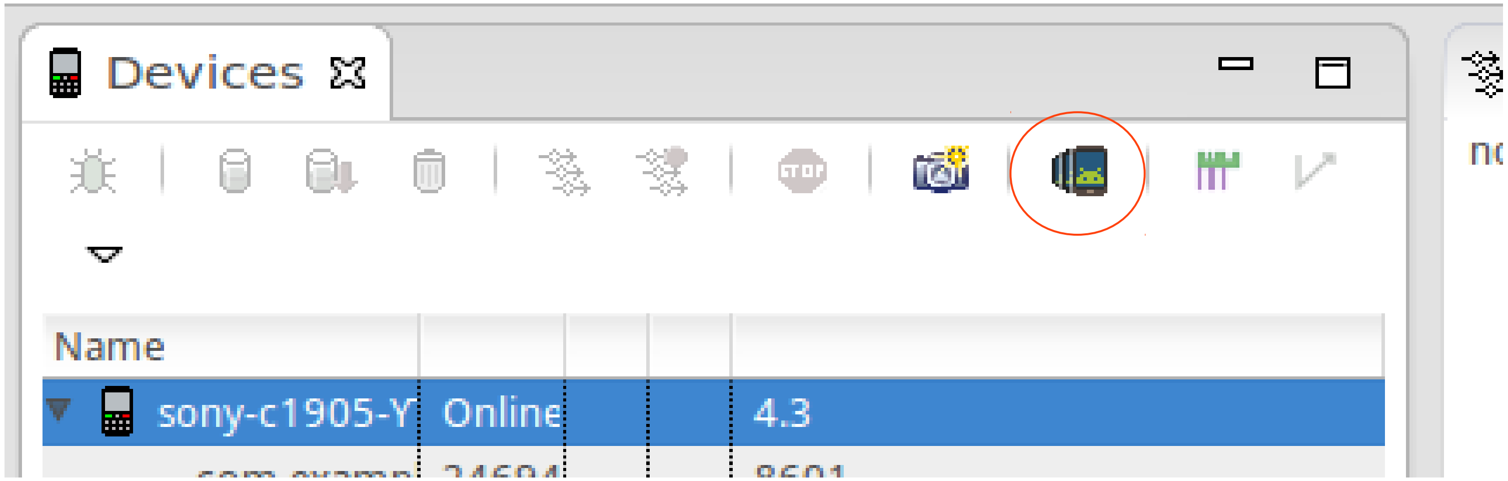

21:46
GraphicsTest
Red
Magenta
Dark Gray
Gray
Green
Cyan


6:01 PM
loopt
Map
Friends
Lately
Check In
Places
Events
Invite
Settings

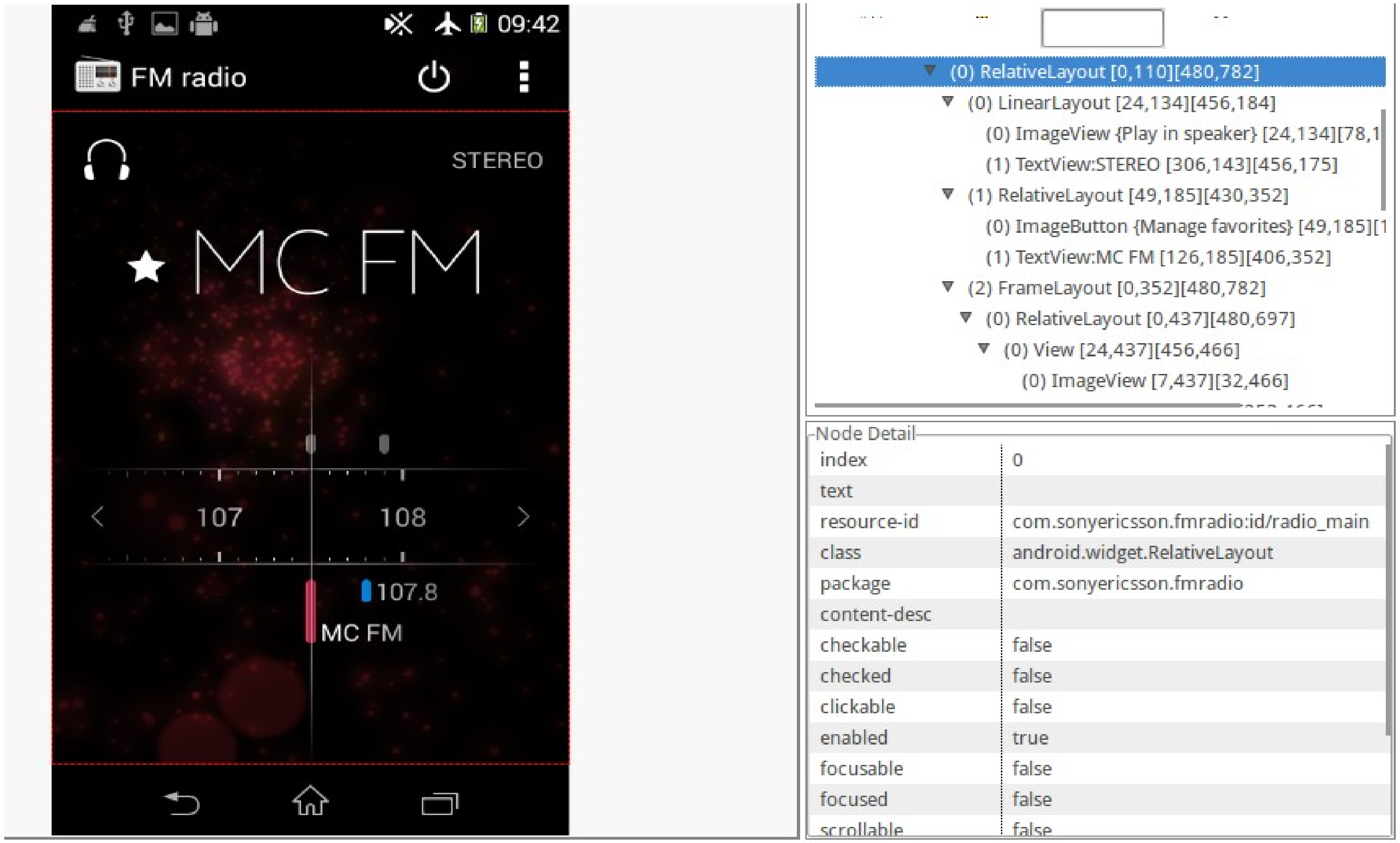

8:49 PM
AndroidGridLayout

Inspect View

You can analyze layout from application that run on your system using tools under DDMS perspective



Inspect View



```
▼ (0) RelativeLayout [0,110][480,782]
  ▼ (0) LinearLayout [24,134][456,184]
      (0) ImageView {Play in speaker} [24,134][78,1
      (1) TextView:STEREO [306,143][456,175]
  ▼ (1) RelativeLayout [49,185][430,352]
      (0) ImageButton {Manage favorites} [49,185][1
      (1) TextView:MC FM [126,185][406,352]
  ▼ (2) FrameLayout [0,352][480,782]
    ▼ (0) RelativeLayout [0,437][480,697]
      ▼ (0) View [24,437][456,466]
          (0) ImageView [7,437][32,466]
```

Node Detail

index	0
text	
resource-id	com.sonyericsson.fmradio:id/radio_main
class	android.widget.RelativeLayout
package	com.sonyericsson.fmradio
content-desc	
checkable	false
checked	false
clickable	false
enabled	true
focusable	false
focused	false
scrollable	false

Android Design Principles

- Enchant Me

  Android apps are sleek and aesthetically pleasing on multiple levels.

- Simplify My Life

  Android apps make life easier and are easy to understand.

- Make Me Amazing

  Android apps empower people to try new things and to use apps in inventive new ways.

Enchant Me

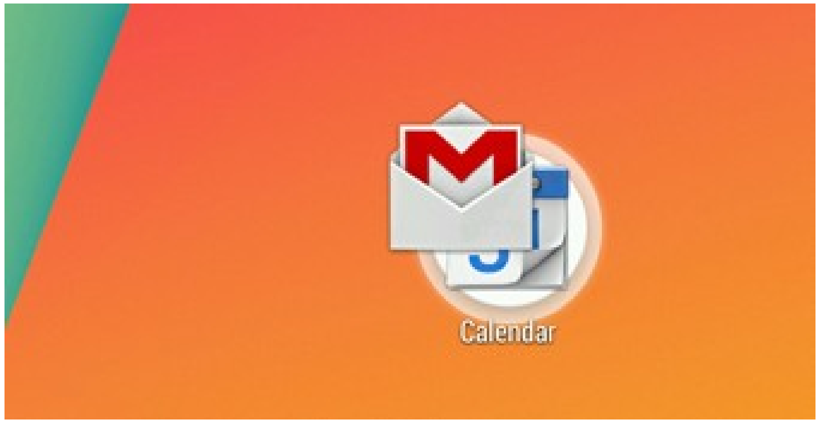
Delight me in surprising ways:

A beautiful surface, a carefully-placed animation, or a well-timed sound effect is a joy to experience. Subtle effects contribute to a feeling of effortlessness and a sense that a powerful force is at hand.

Enchant Me

Real objects are more fun than buttons and menus:

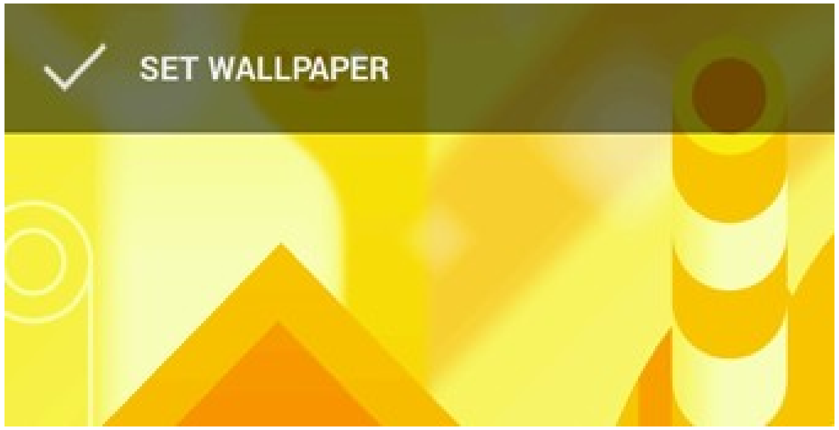
Allow people to directly touch and manipulate objects in your app. It reduces the cognitive effort needed to perform a task while making it more emotionally satisfying.



Enchant Me

Let me make it mine:

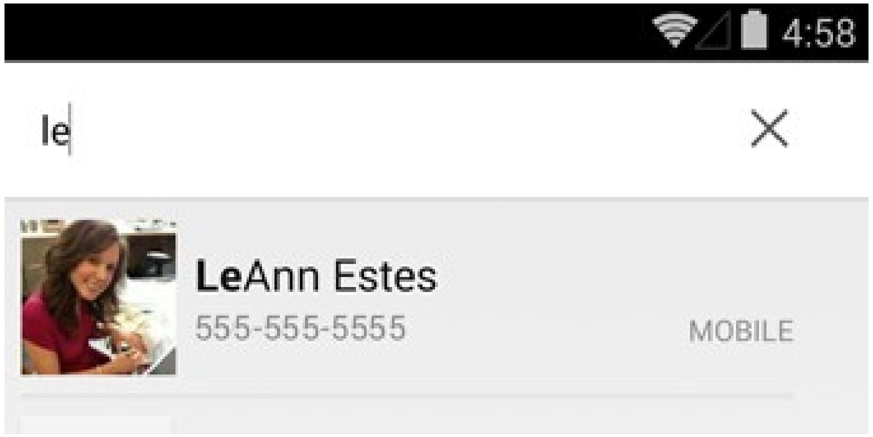
People love to add personal touches because it helps them feel at home and in control. Provide sensible, beautiful defaults, but also consider fun, optional customizations that don't hinder primary tasks.



Enchant Me

Get to know me:

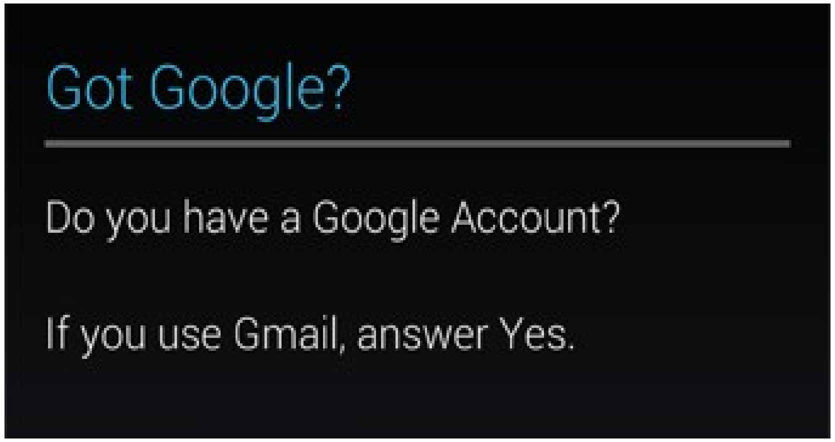
Learn peoples' preferences over time. Rather than asking them to make the same choices over and over, place previous choices within easy reach.



Simplify My Life

Keep it brief:

Use short phrases with simple words. People are likely to skip sentences if they're long.



Got Google?

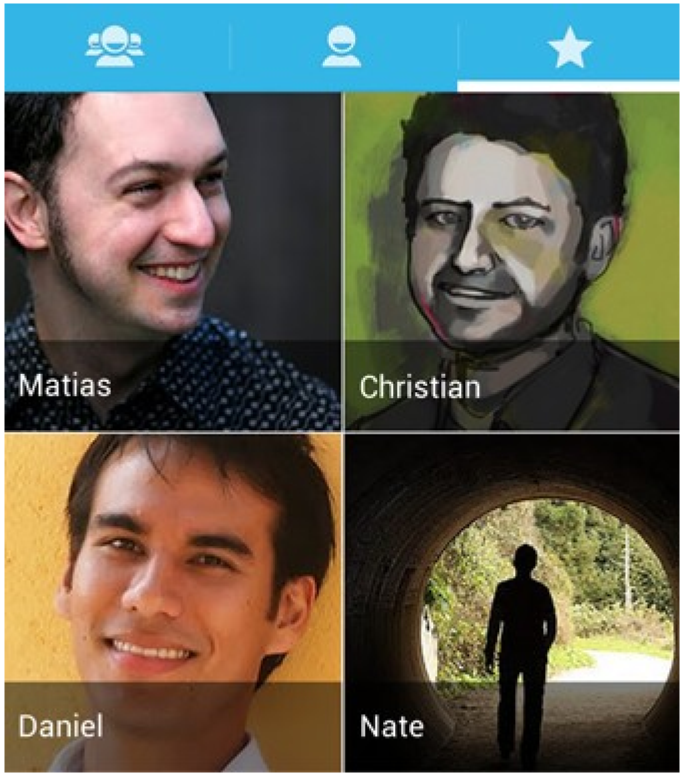
Do you have a Google Account?

If you use Gmail, answer Yes.

Simplify My Life

Pictures are faster than words:

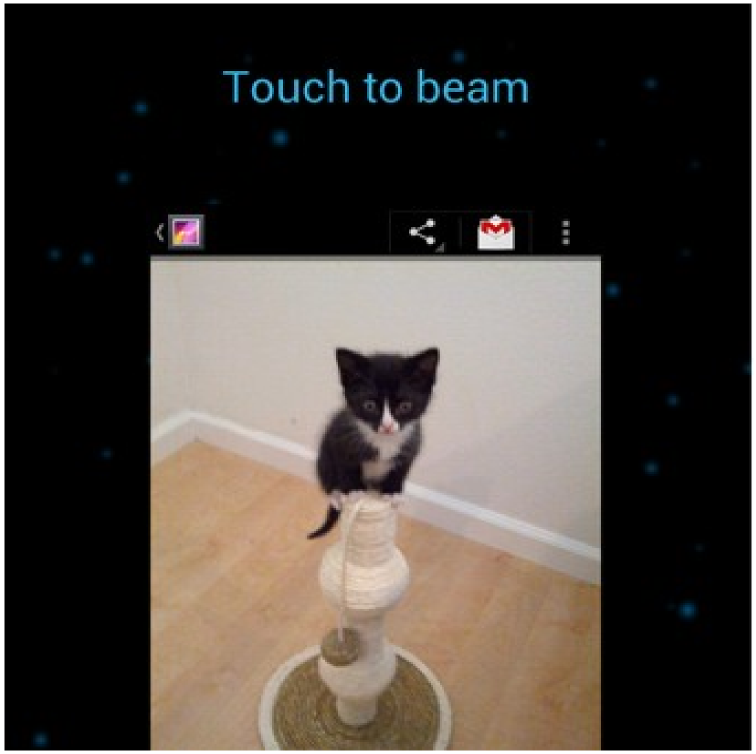
Consider using pictures to explain ideas. They get people's attention and can be much more efficient than words.



Simplify My Life

Decide for me but let me have the final say:

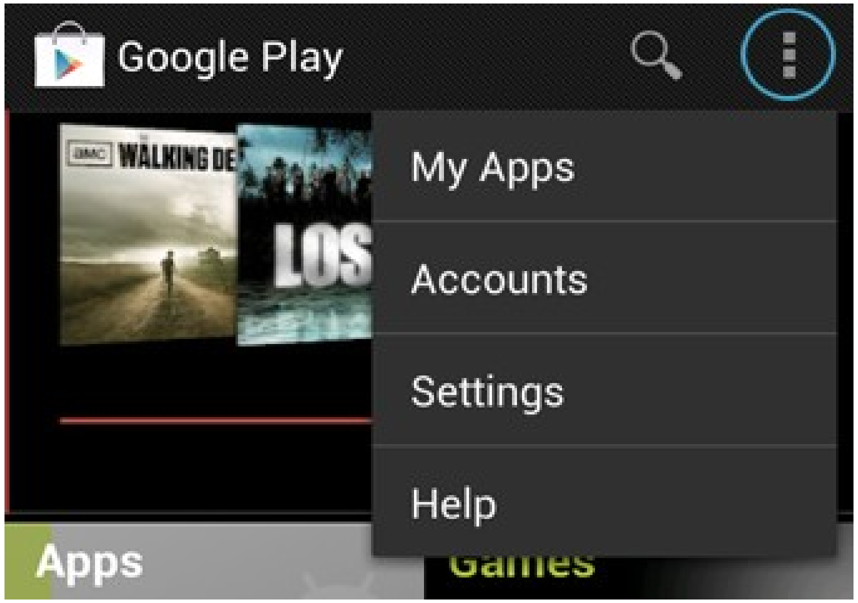
Take your best guess and act rather than asking first. Too many choices and decisions make people unhappy. Just in case you get it wrong, allow for 'undo'.



Simplify My Life

Only show what I need when I need it:

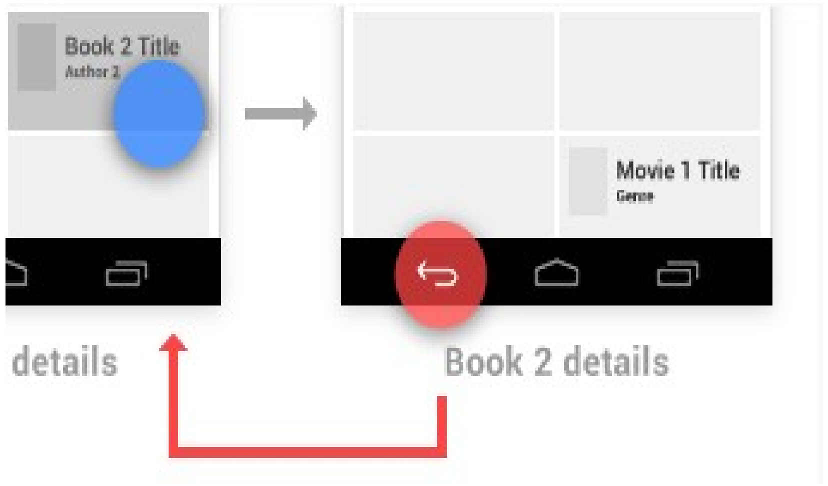
People get overwhelmed when they see too much at once. Break tasks and information into small, digestible chunks. Hide options that aren't essential at the moment, and teach people as they go.



Simplify My Life

I should always know where I am:

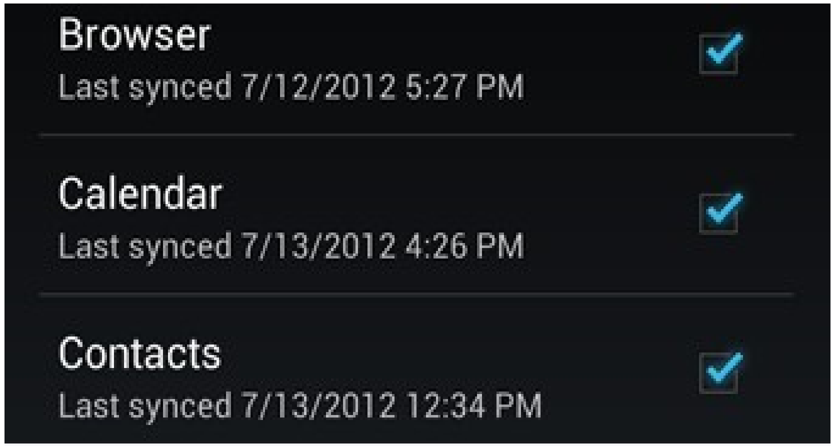
Give people confidence that they know their way around. Make places in your app look distinct and use transitions to show relationships among screens. Provide feedback on tasks in progress.



Simplify My Life

Never lose my stuff:

Save what people took time to create and let them access it from anywhere. Remember settings, personal touches, and creations across phones, tablets, and computers. It makes upgrading the easiest thing in the world.



Browser
Last synced 7/12/2012 5:27 PM ✓

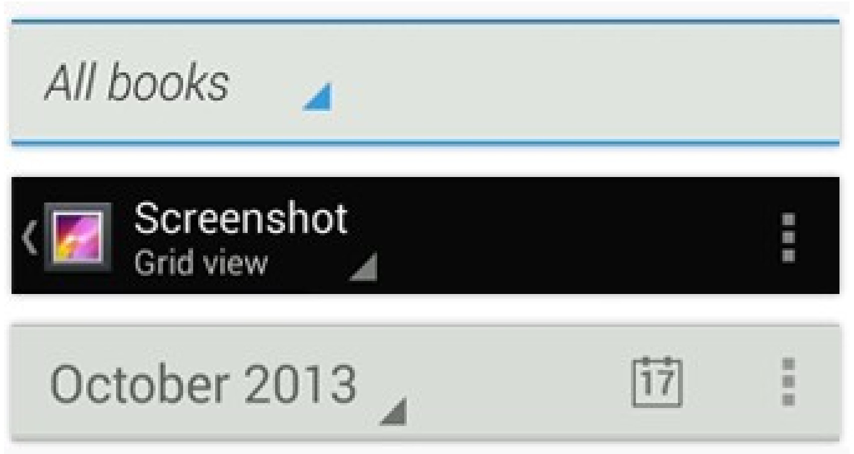
Calendar
Last synced 7/13/2012 4:26 PM ✓

Contacts
Last synced 7/13/2012 12:34 PM ✓

Simplify My Life

If it looks the same, it should act the same:

Help people discern functional differences by making them visually distinct rather than subtle. Avoid modes, which are places that look similar but act differently on the same input.



Simplify My Life

Only interrupt me if it's important:

Like a good personal assistant, shield people from unimportant minutiae. People want to stay focused, and unless it's critical and time-sensitive, an interruption can be taxing and frustrating.

Make Me Amazing

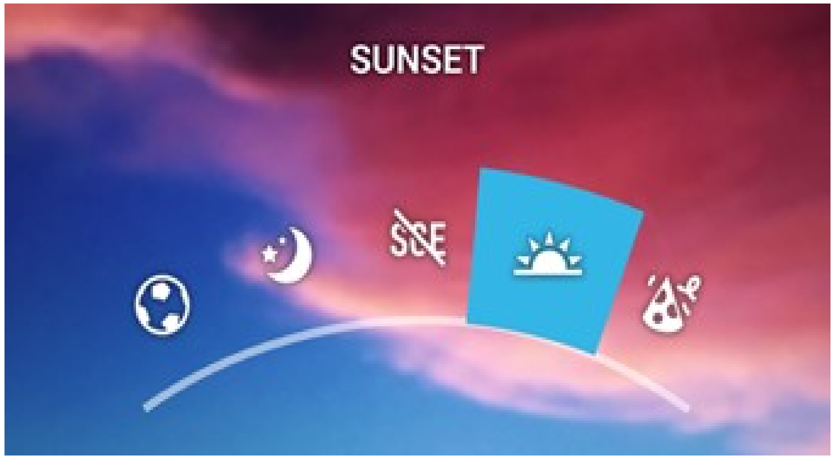
Sprinkle encouragement:

Break complex tasks into smaller steps that can be easily accomplished. Give feedback on actions, even if it's just a subtle glow.

Make Me Amazing

Do the heavy lifting for me:

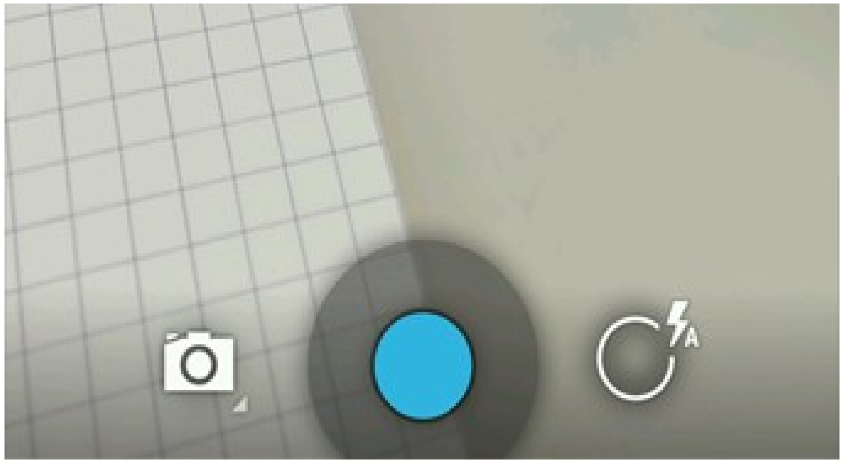
Make novices feel like experts by enabling them to do things they never thought they could. For example, shortcuts that combine multiple photo effects can make amateur photographs look amazing in only a few steps.



Make Me Amazing

Make important things fast:

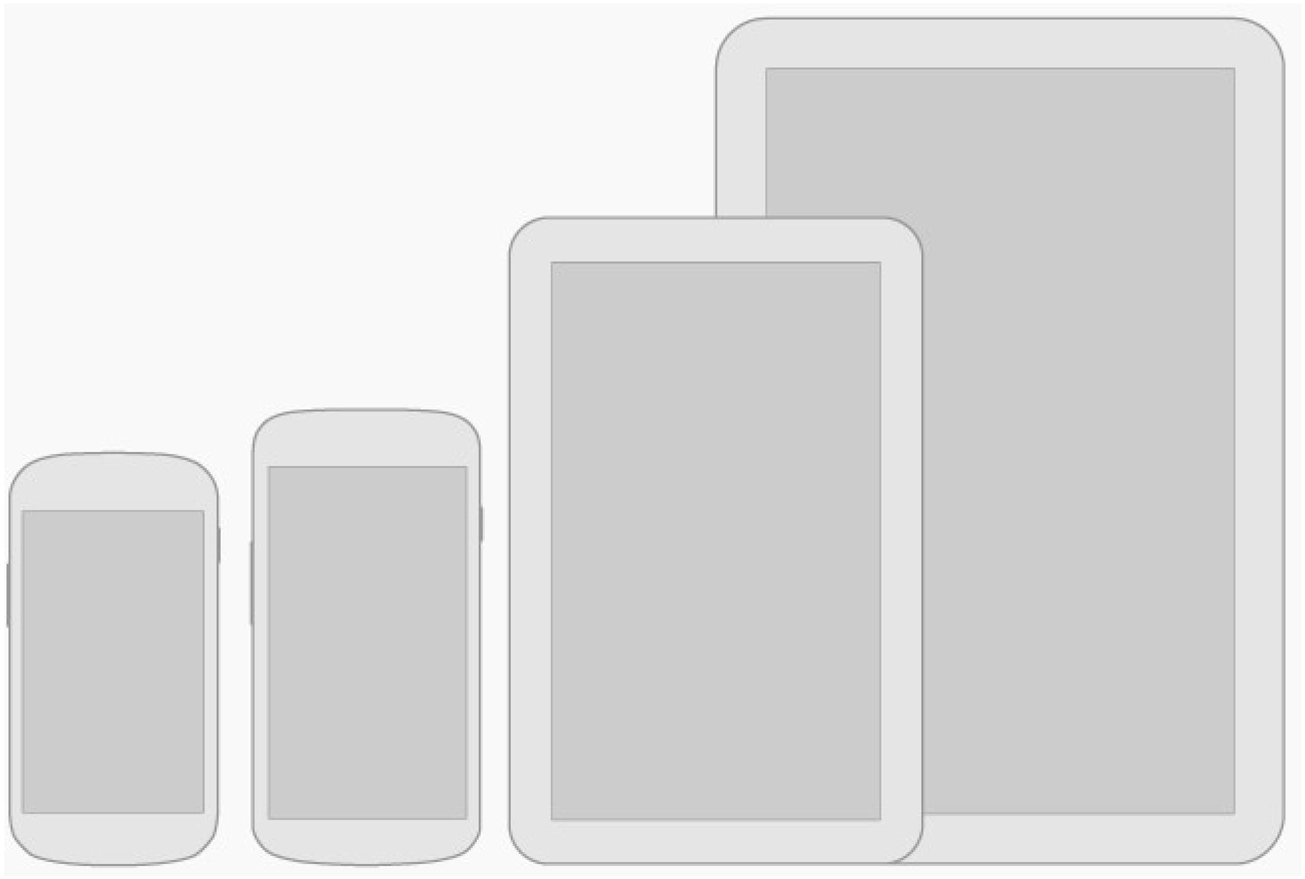
Not all actions are equal. Decide what's most important in your app and make it easy to find and fast to use, like the shutter button in a camera, or the pause button in a music player.



Devices and Displays

- Android powers hundreds of millions of phones, tablets, and other devices in a wide variety of screen sizes and form factors.
- By taking advantage of Android's flexible layout system, you can create apps that gracefully scale from large tablets to smaller phones.

Devices and Displays



Devices and Displays

- Be flexible

  Stretch and compress your layouts to accommodate various heights and widths.

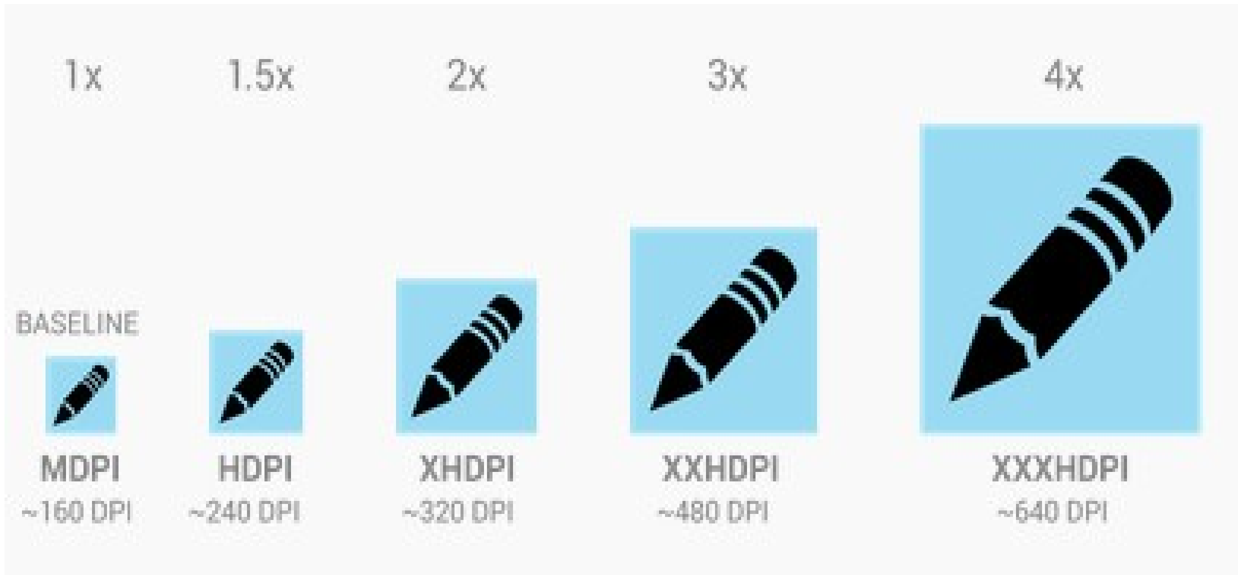
- Optimize layouts

  On larger devices, take advantage of extra screen real estate.

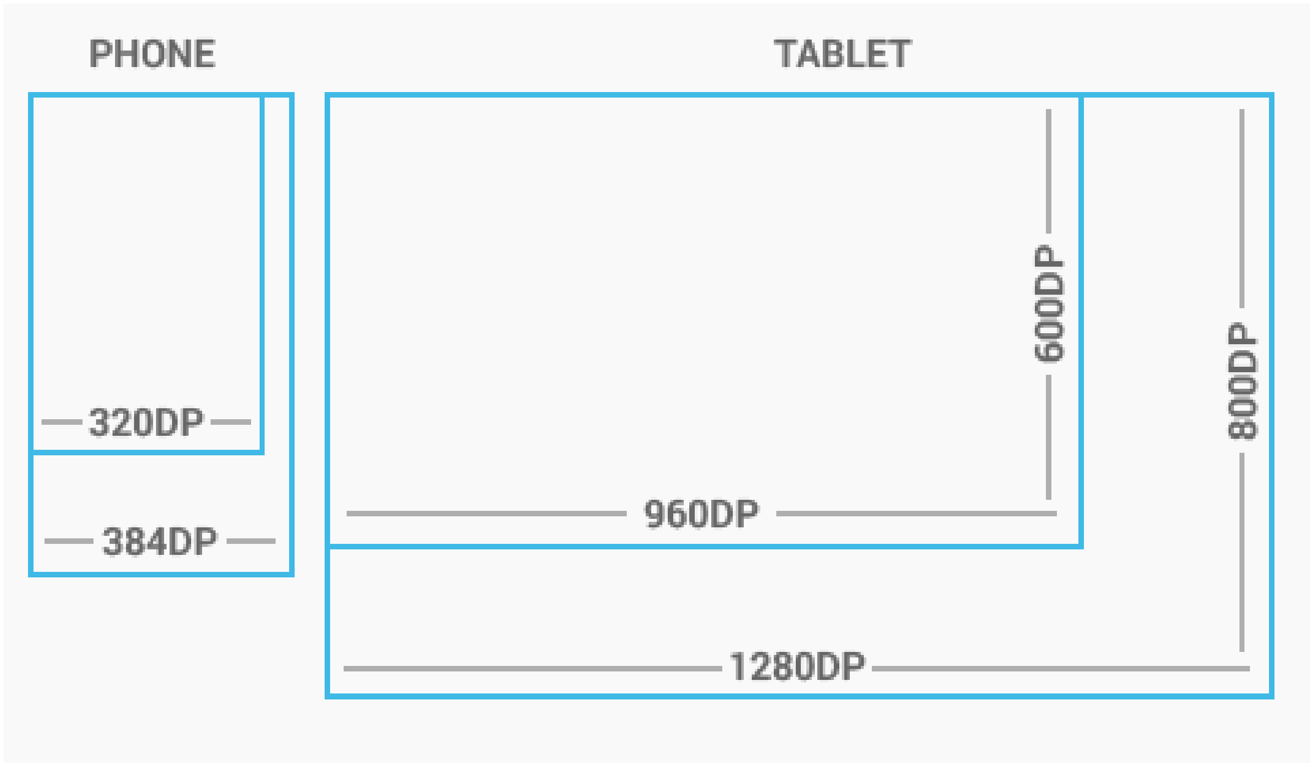
- Assets for all

  Provide resources for different screen densities (DPI) to ensure that your app looks great on any device.

Devices and Displays

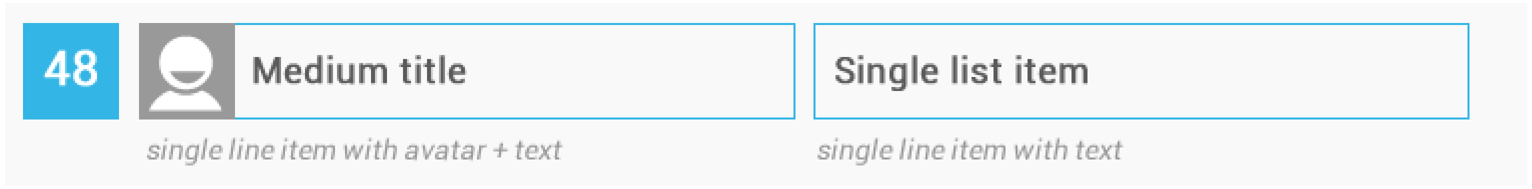


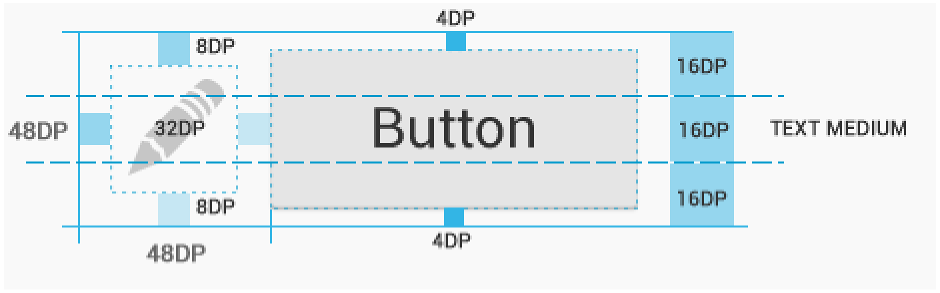
Metrics and Grids



Metrics and Grids

Touchable UI components are generally laid out along 48dp units.


48
Medium title
Single list item
single line item with avatar + text
single line item with text

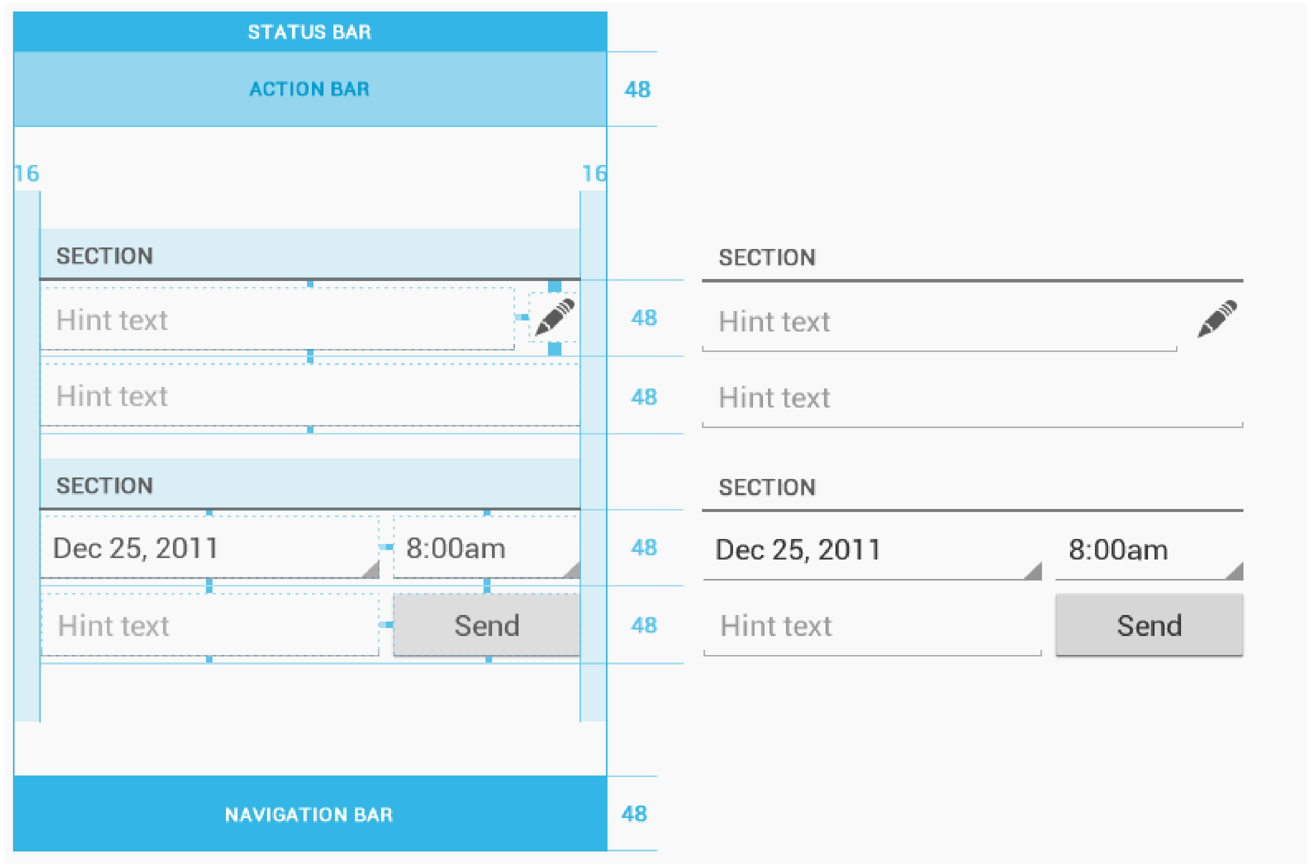


Metrics and Grids

Why 48dp?

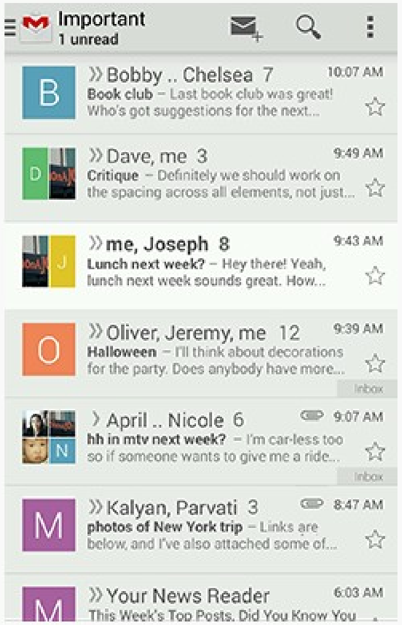
- On average, 48dp translate to a physical size of about 9mm (with some variability). This is comfortably in the range of recommended target sizes (7-10 mm) for touchscreen objects and users will be able to reliably and accurately target them with their fingers.
- If you design your elements to be at least 48dp high and wide you can guarantee that:
  - your targets will never be smaller than the minimum recommended target size of 7mm regardless of what screen they are displayed on.
  - you strike a good compromise between overall information density on the one hand, and targetability of UI elements on the other.

Metrics and Grids



Themes

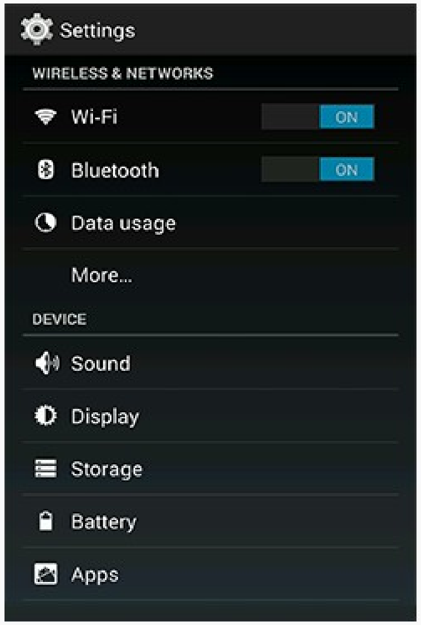
Themes are Android's mechanism for applying a consistent style to an app or activity.



Important
1 unread

B	Bobby .. Chelsea 7 — **Book club** – Last book club was great! Who's got suggestions for the next...	10:07 AM
D	Dave, me 3 — **Critique** – Definitely we should work on the spacing across all elements, not just...	9:49 AM
J	me, Joseph 8 — **Lunch next week?** – Hey there! Yeah, lunch next week sounds great. How...	9:43 AM
O	Oliver, Jeremy, me 12 — **Halloween** – I'll think about decorations for the party. Does anybody have more... (Inbox)	9:39 AM
N	April .. Nicole 6 — **hh in mtv next week?** – I'm car-less too so if someone wants to give me a ride... (Inbox)	9:07 AM
M	Kalyan, Parvati 3 — **photos of New York trip** – Links are below, and I've also attached some of...	8:47 AM
M	Your News Reader — This Week's Top Posts. Did You Know You	6:03 AM

Gmail in Holo Light.



Settings

WIRELESS & NETWORKS

- Wi-Fi: ON
- Bluetooth: ON
- Data usage
- More...

DEVICE

- Sound
- Display
- Storage
- Battery
- Apps

Settings in Holo Dark.