UI Design and Development

+Roman Nurik
+Nick Butcher

Agenda

1. Designing for Android
2. Layouts and resources
3. Tablet considerations
4. System UI integration
5. Wireframing

Designing for Android

Design for...

- Touch
  - Interact primarily with your fingers
  - Expect direct manipulation
- Mobile
  - Often on the go
  - Often without network connectivity
- Heterogeneity
  - Different screen sizes and densities
  - Different hardware features
  - Different OS versions

Key principles

“Pictures are faster than words.”

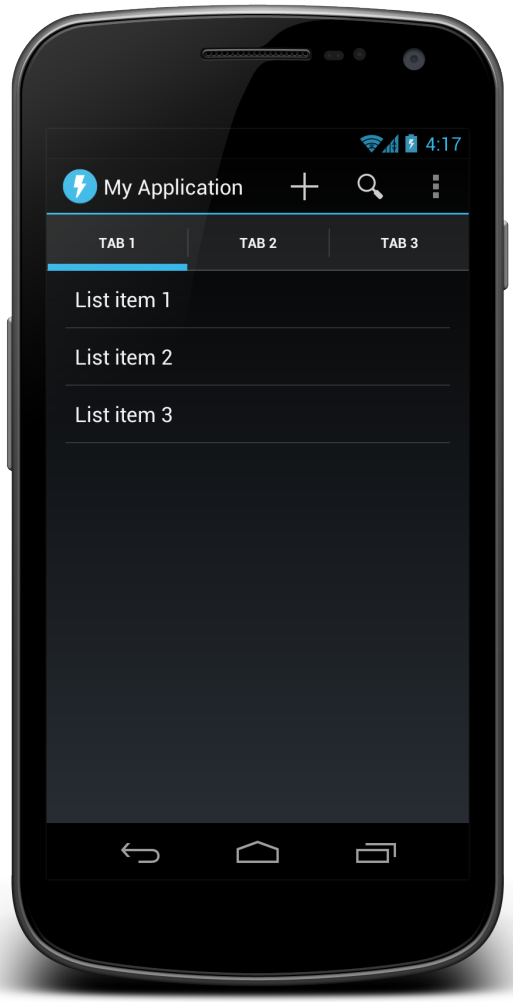
“Only show what I need when I need it.”

“Make the important things fast.”

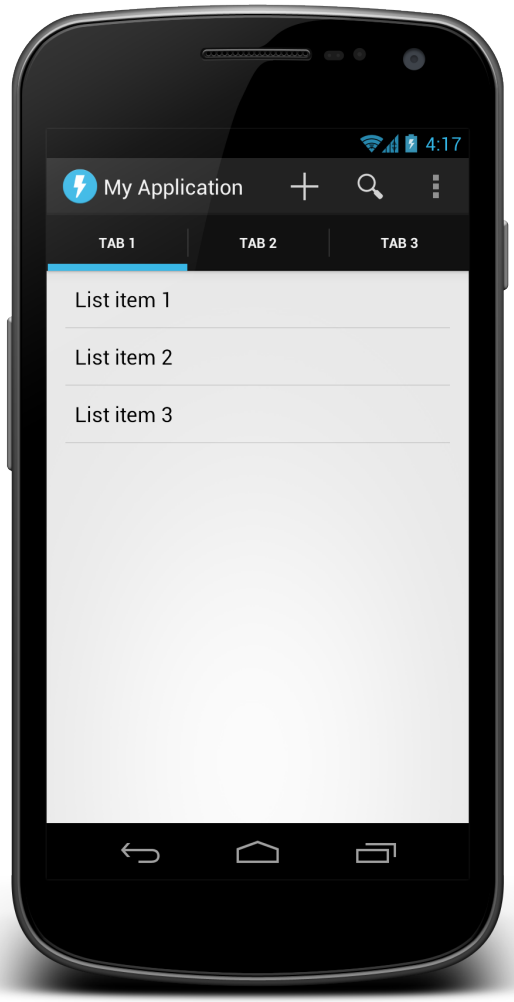
“Do the heavy lifting for me.”

“Holo” visual language

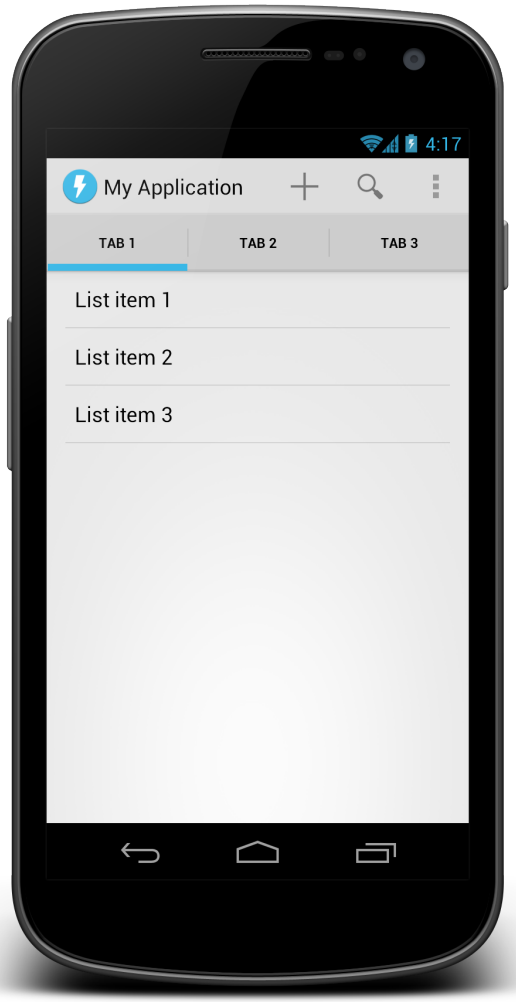
Holo variations



Dark



Dark Action Bar



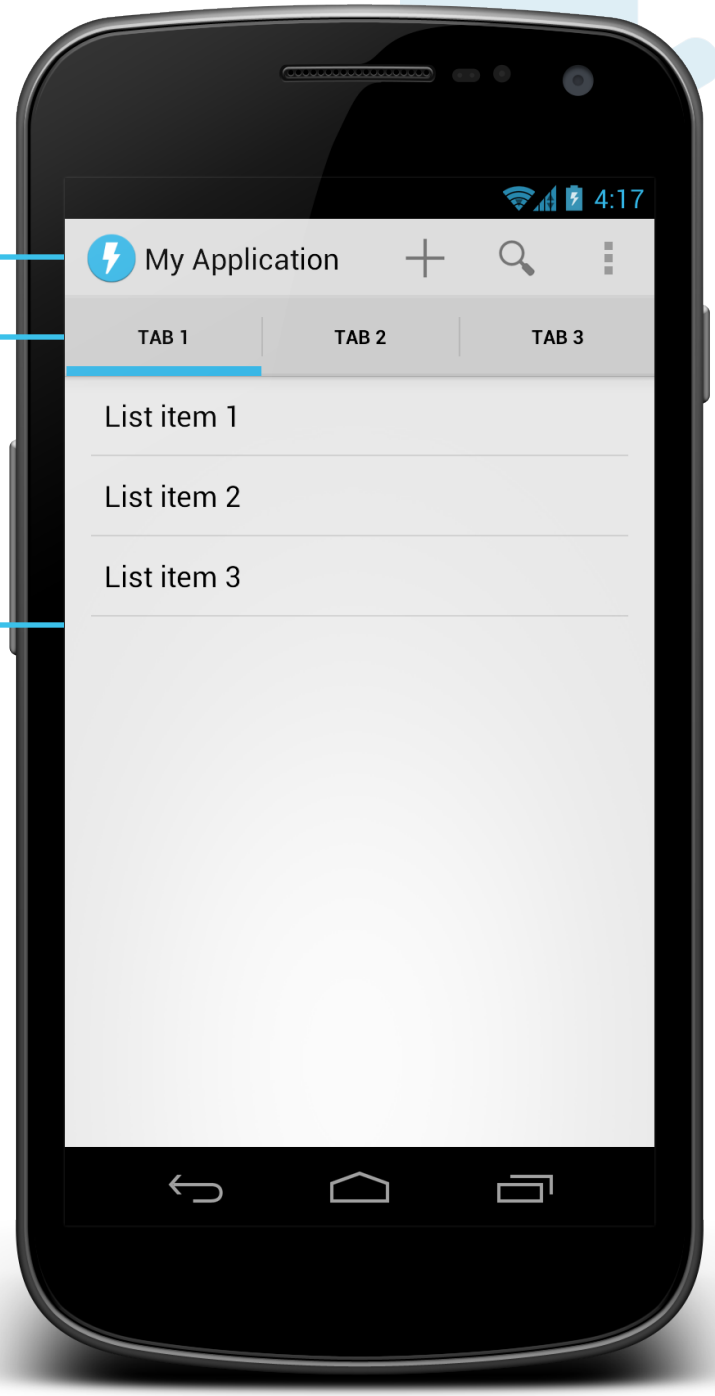
Light

Activity UI structure

Action bar

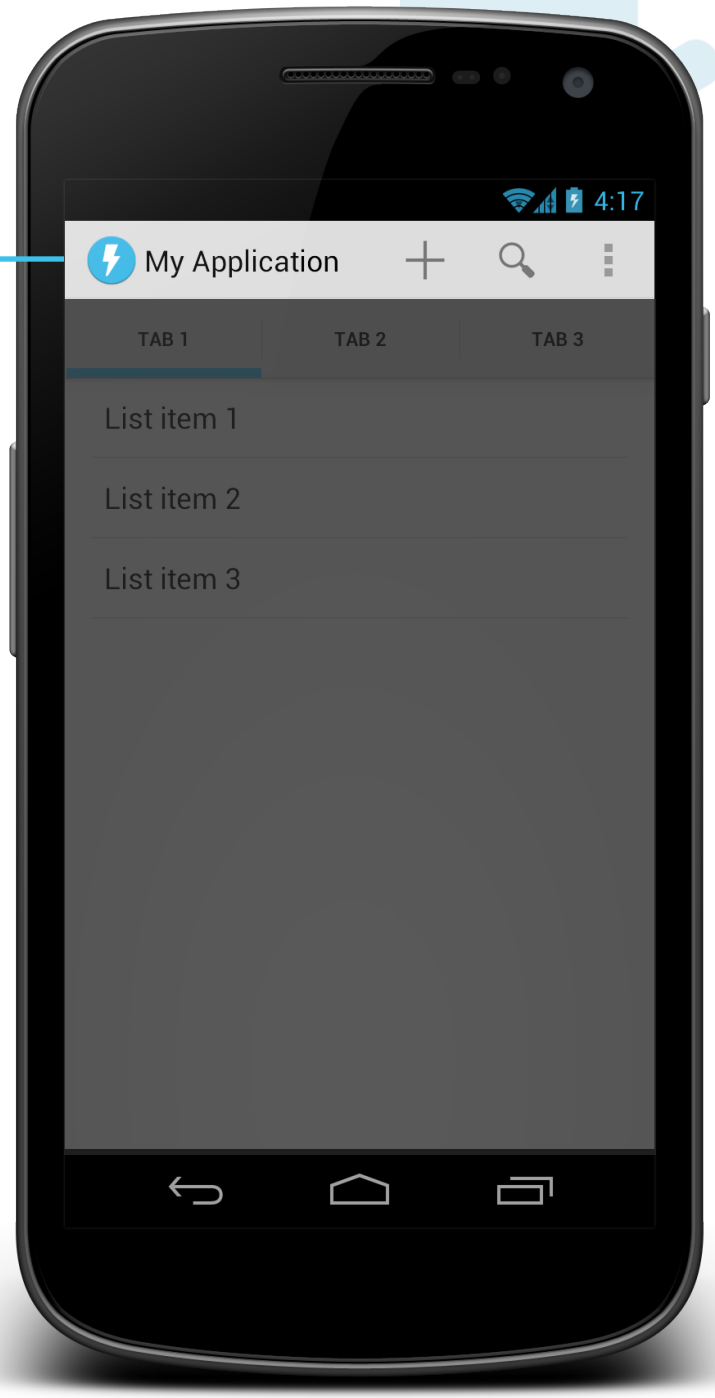
Tabs

Content
(activity layout)



Activity UI structure

Action bar



Action bar

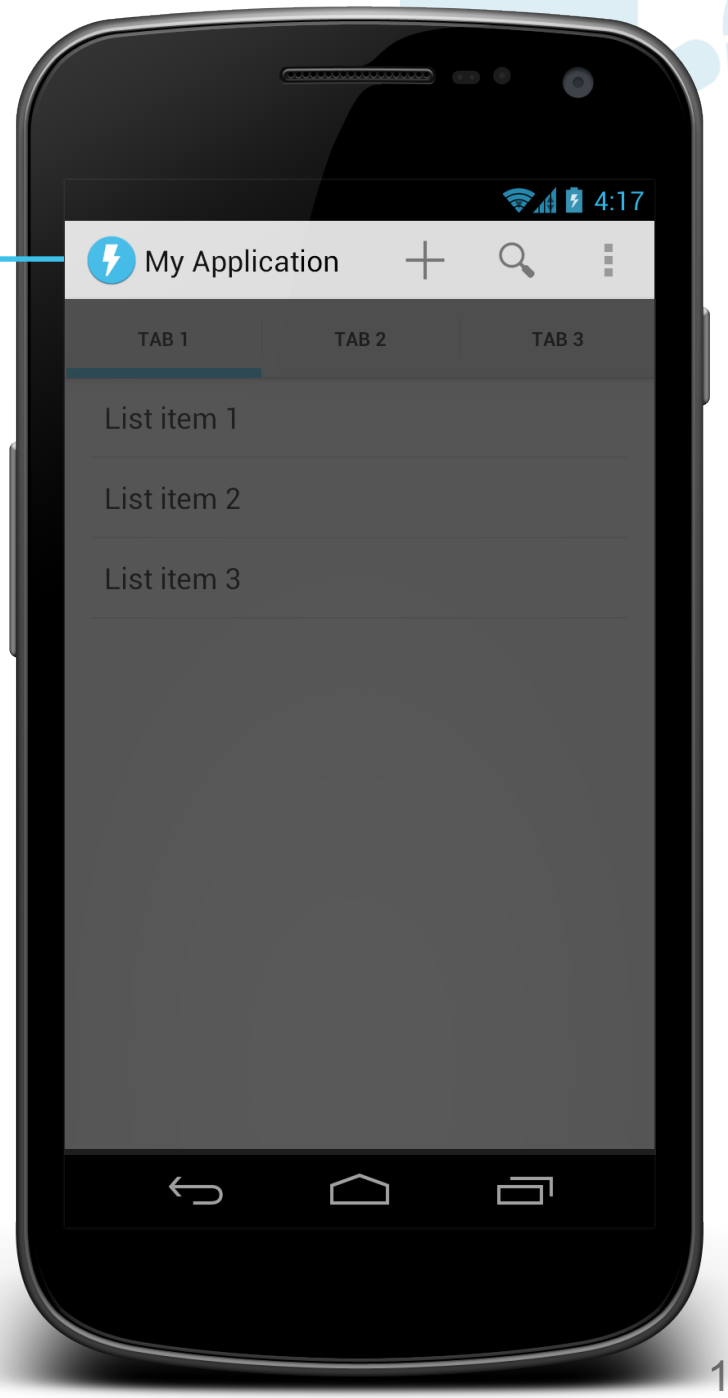
1. App icon and optional Up caret
2. View control (Title/tabs/dropdown)
3. Action buttons
4. Action overflow

Action bar

- Android 3.0 and above
- Automatically part of Holo themes
- Customize:
  - `getActionBar().setDisplayOptions()`
  - `getActionBar().setNavigationMode()`

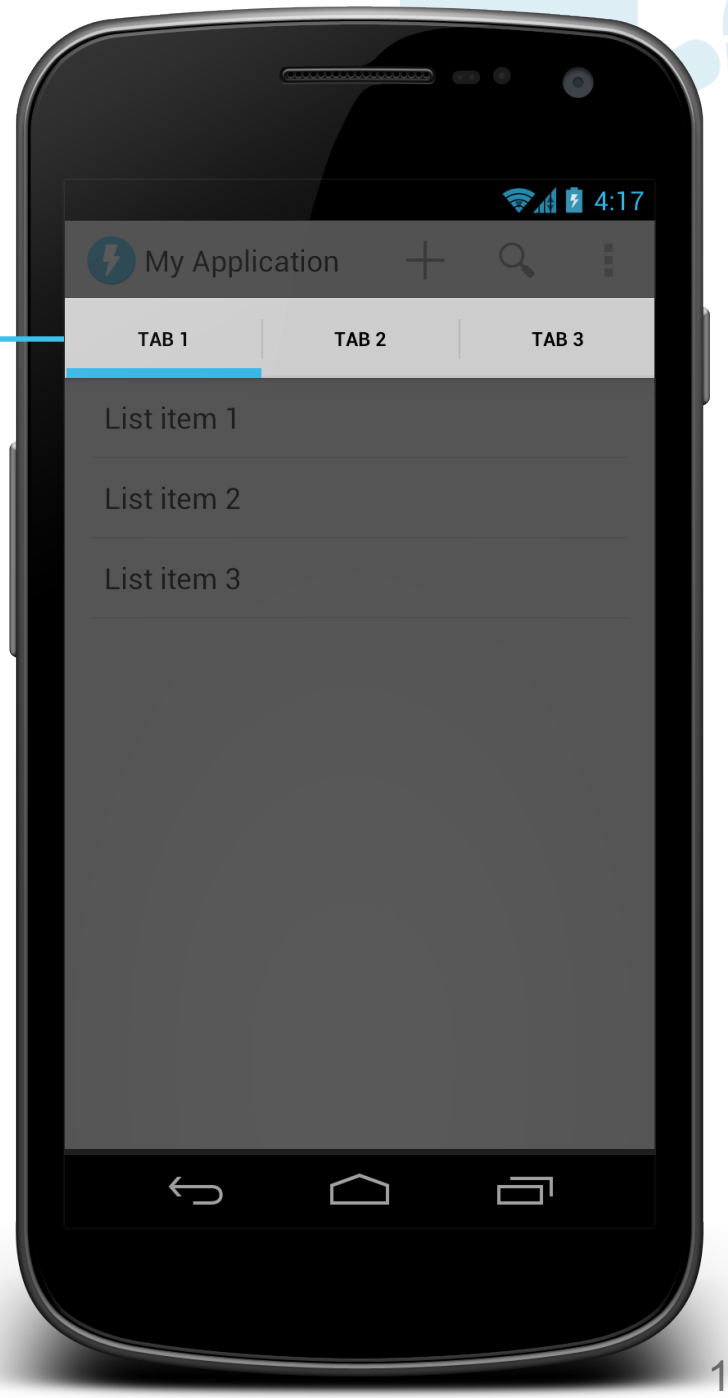
Activity UI structure

Action bar



Activity UI structure

Tabs



Tabs


You Tube
HOME
BROWSE
ACCOUNT
Google Music Event (11/16/11)
androiddevelopers
1 month ago | 57,360 views HD
39:34
200 million Android activations!
androiddevelopers
1 month ago | 146,372 views HD
00:35
Android Developer Story: Robot Invader
androiddevelopers
1 month ago | 59,618 views HD
02:38
Google and Samsung, a look at what's new from Android
androiddevelopers
2 months ago | 696,684 views HD
57:08
Introducing Galaxy Nexus. Simple, beautiful, beyond smart
androiddevelopers
2 months ago | 3,713,064 views HD
01:56
Calling All Ice Cream Sandwich Lovers
androiddevelopers
2 months ago | 801,726 views HD

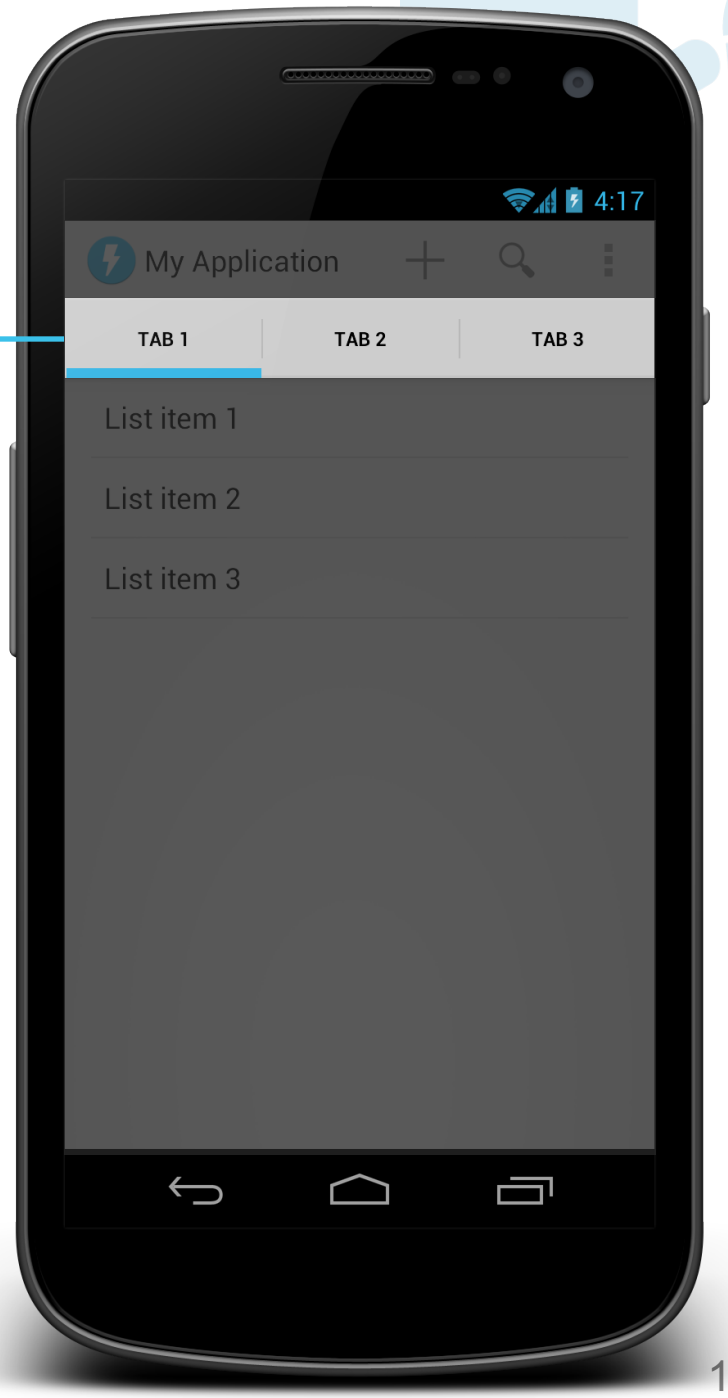
Tabs

- Part of the `ActionBar` APIs
- Usually gesture-enabled using `ViewPager`

```
getActionBar().setNavigationMode(NAVIGATION_MODE_TABS);

ActionBar.Tab tab = actionBar.newTab();
tab.setText(“Tab 1”);
tab.setTabListener(this);
getActionBar().addTab(tab);
```

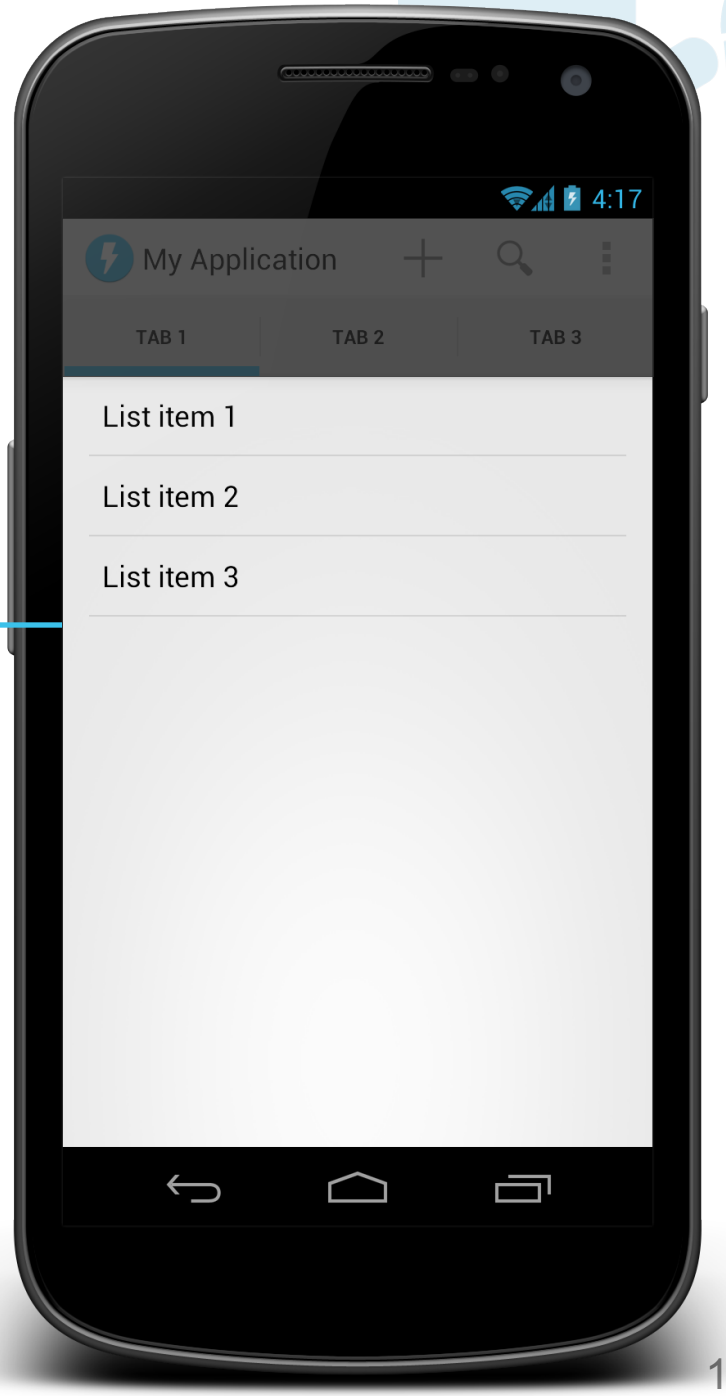
Activity UI structure

Tabs



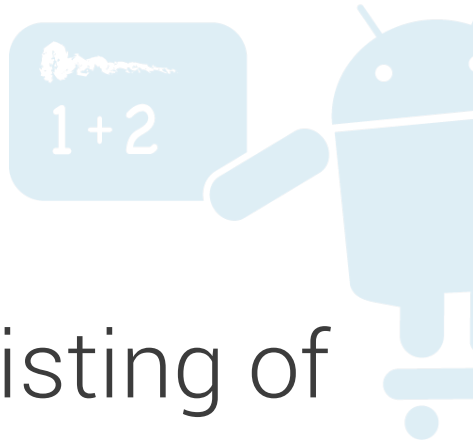
Activity UI structure

Content
(activity layout)



Layouts and resources

Layout system



- The UI for an activity is a tree consisting of view groups and views (leaf nodes), like HTML.

```
<view group>
    <view group>
        <view>
    <view group>
        <view>
        <view>
```

- Most commonly defined in XML under `res/layout/`.

Views and View Groups

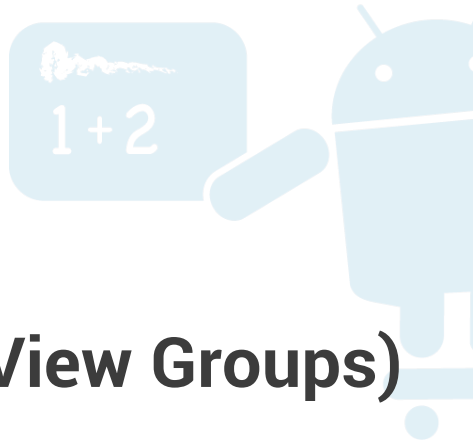
Views

- Reusable individual UI components
- Optionally interactive (clickable/focusable/etc.)
- Bare minimum functionality is to draw themselves

View Groups

- Ordered list of Views and View Groups
- In charge of positioning and sizing their child views and layouts
- Simple layouts and more complex groups (e.g. `ListView`)

Views and View Groups



Views

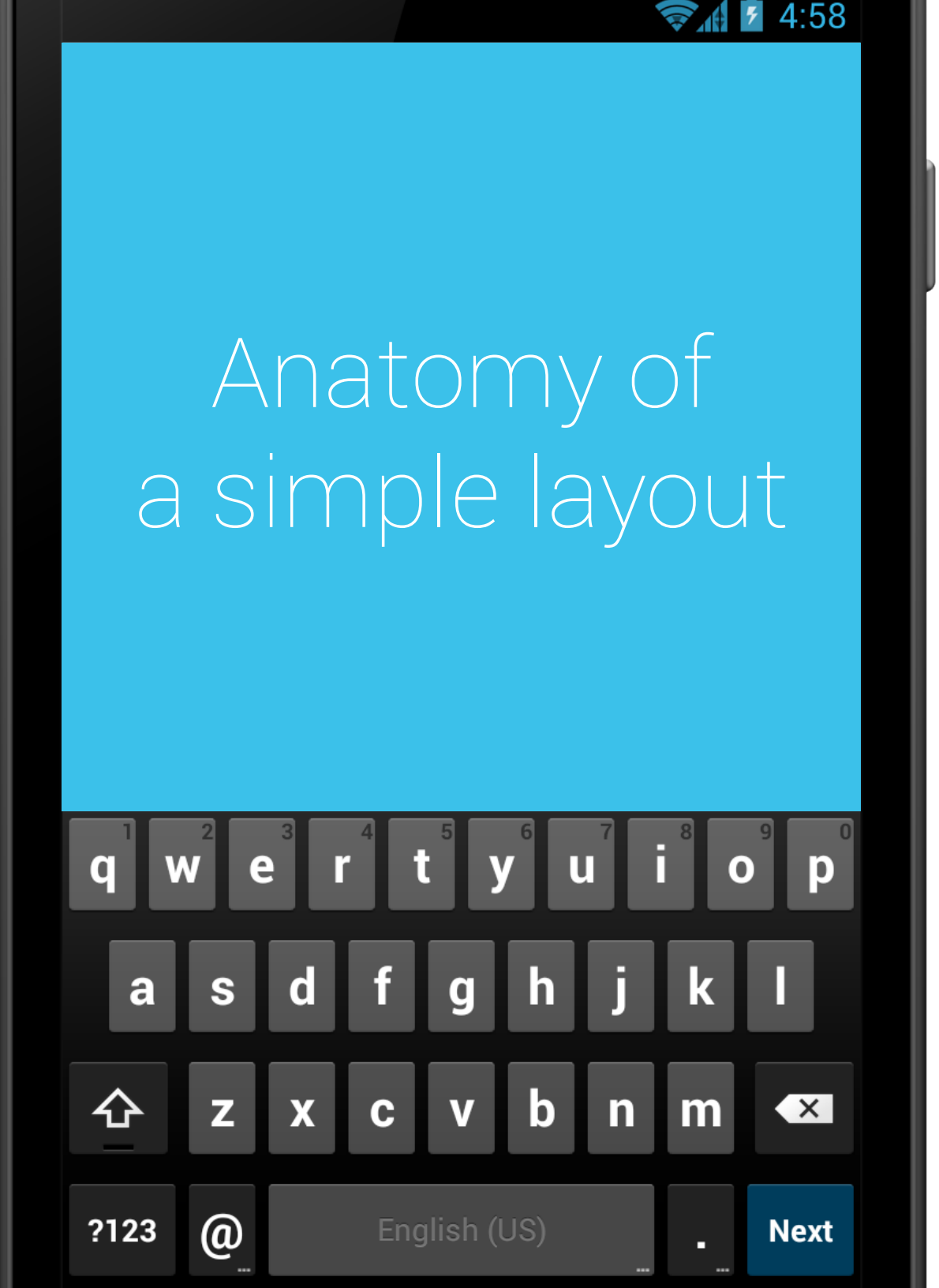
- TextView
- EditText
- Spinner
- ImageView
- Button
- WebView
- SurfaceView
- Your own custom views

Layouts (simple View Groups)

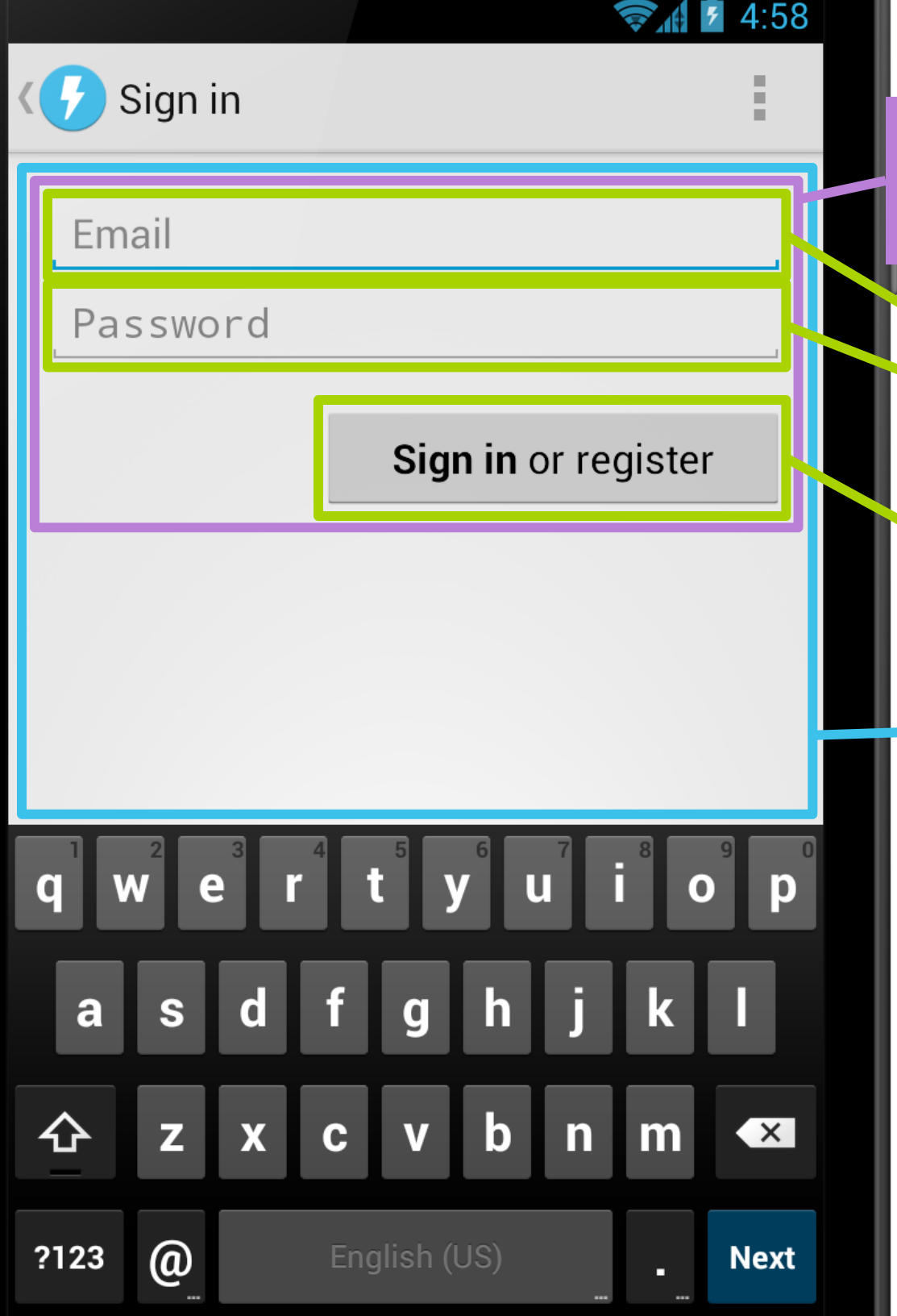
- FrameLayout
- LinearLayout
- RelativeLayout
- GridLayout
- Your own custom layouts

Complex View Groups

- ScrollView
- ListView


4:58
Anatomy of
a simple layout
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
?123 @ English (US) . Next

1+2



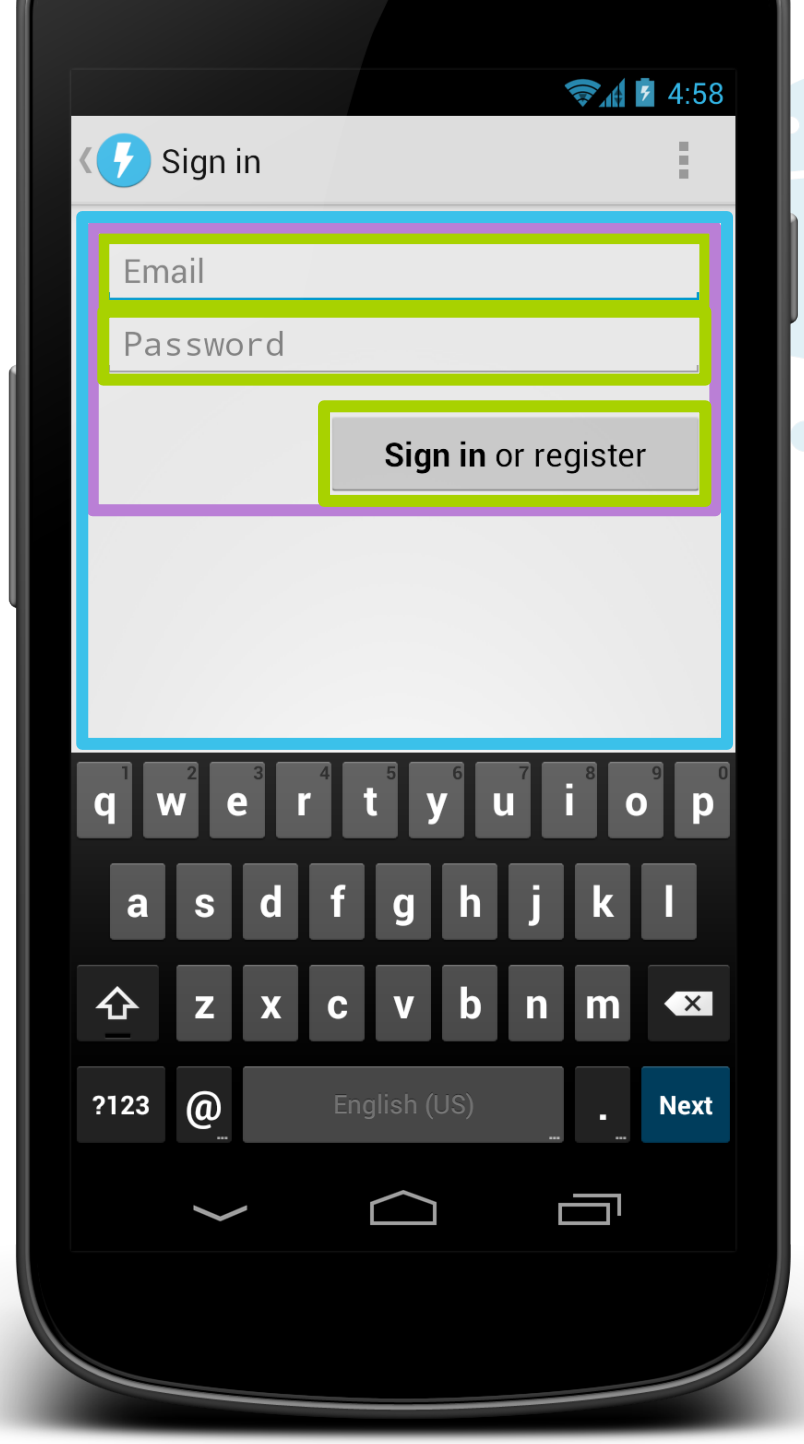
<LinearLayout
orientation="vertical">

<EditText>

<Button>

<ScrollView>

```
<ScrollView
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:padding="16dp">

        <EditText
            android:id="@+id/email"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="@string/prompt_email"
            android:inputType="textEmailAddress"
            android:singleLine="true" />

        <EditText
            android:id="@+id/password"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="@string/prompt_password"
            android:inputType="textPassword"
            android:singleLine="true" />

        <Button
            android:id="@+id/sign_in_button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="right"
            android:layout_marginTop="16dp"
            android:paddingLeft="32dp"
            android:paddingRight="32dp"
            android:text="@string/action_sign_in_register" />
    </LinearLayout>
</ScrollView>
```



Important layout attributes

`android:layout_weight`

- Children of LinearLayout

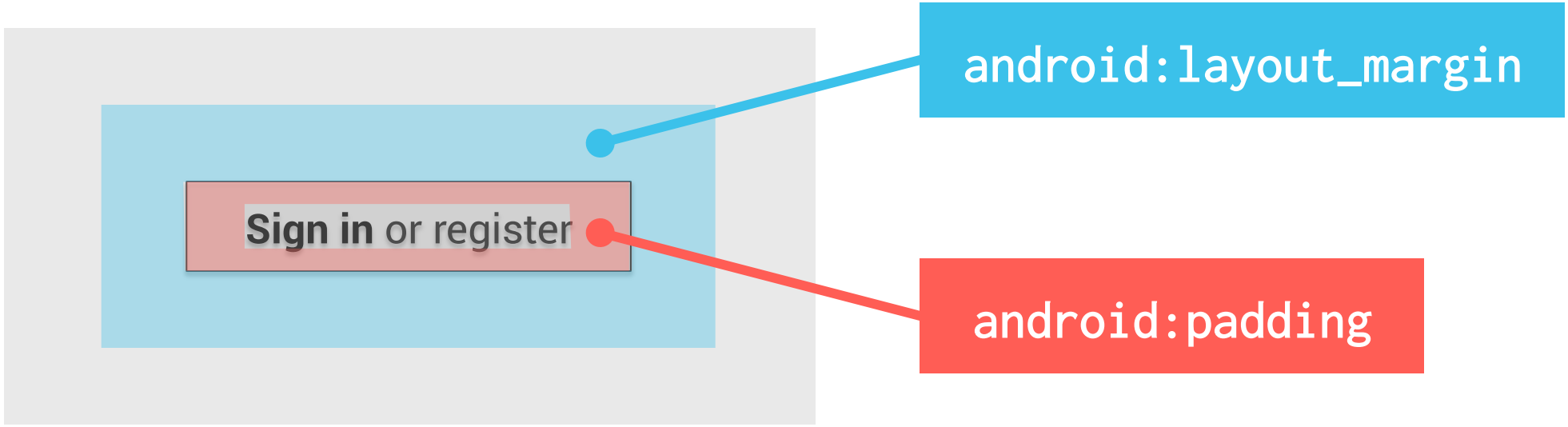
`android:layout_gravity`

- Children of FrameLayout, LinearLayout

`android:gravity`

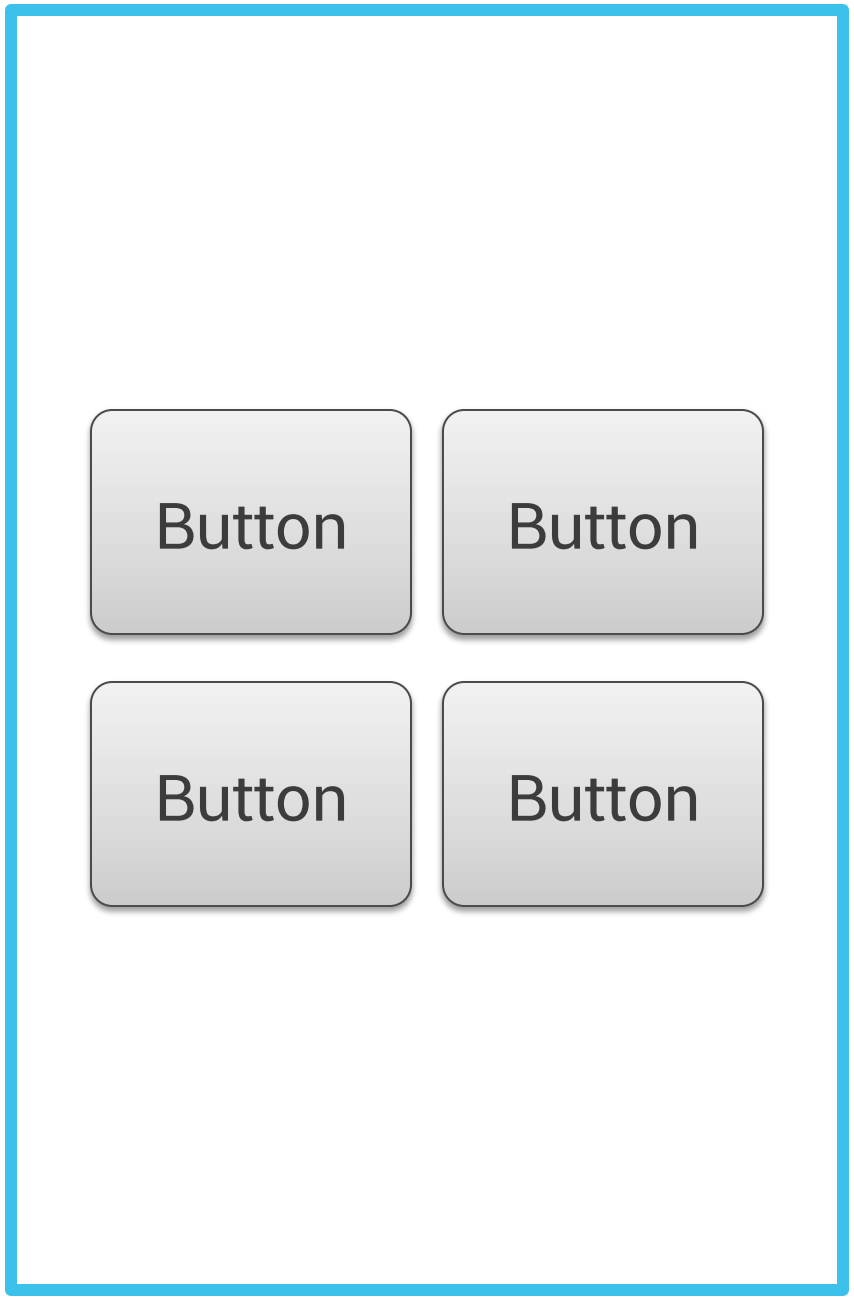
- TextView, LinearLayout

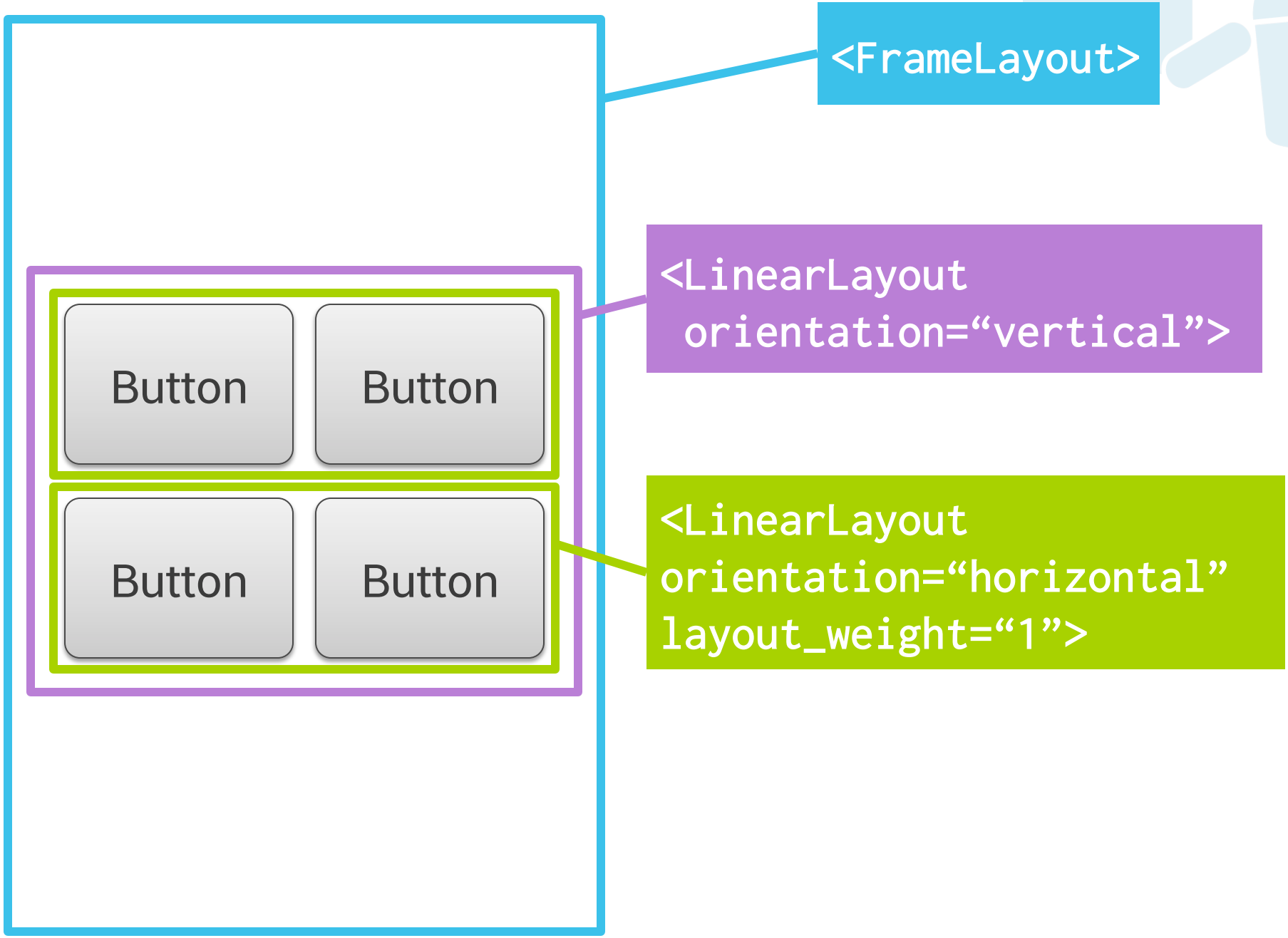
Margins and padding

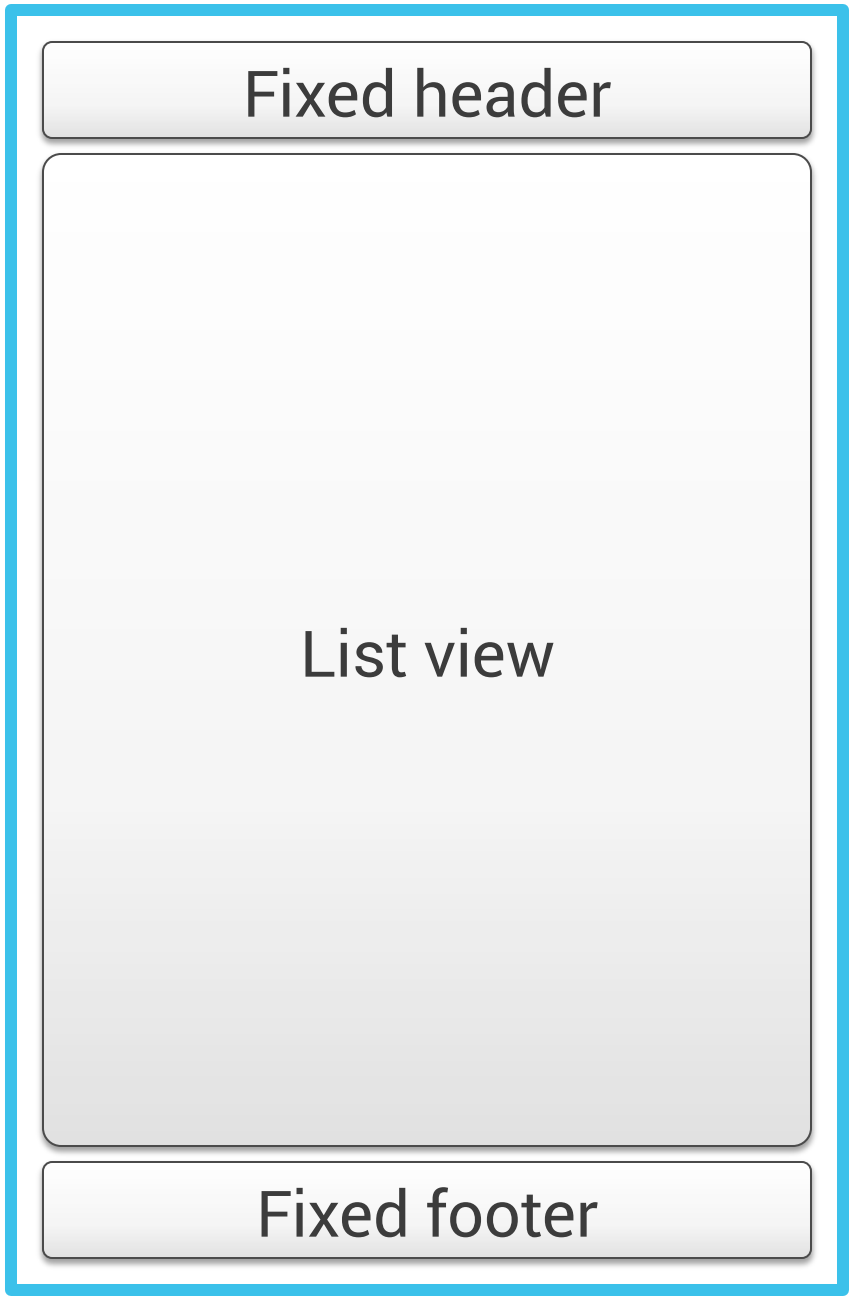


EXERCISE

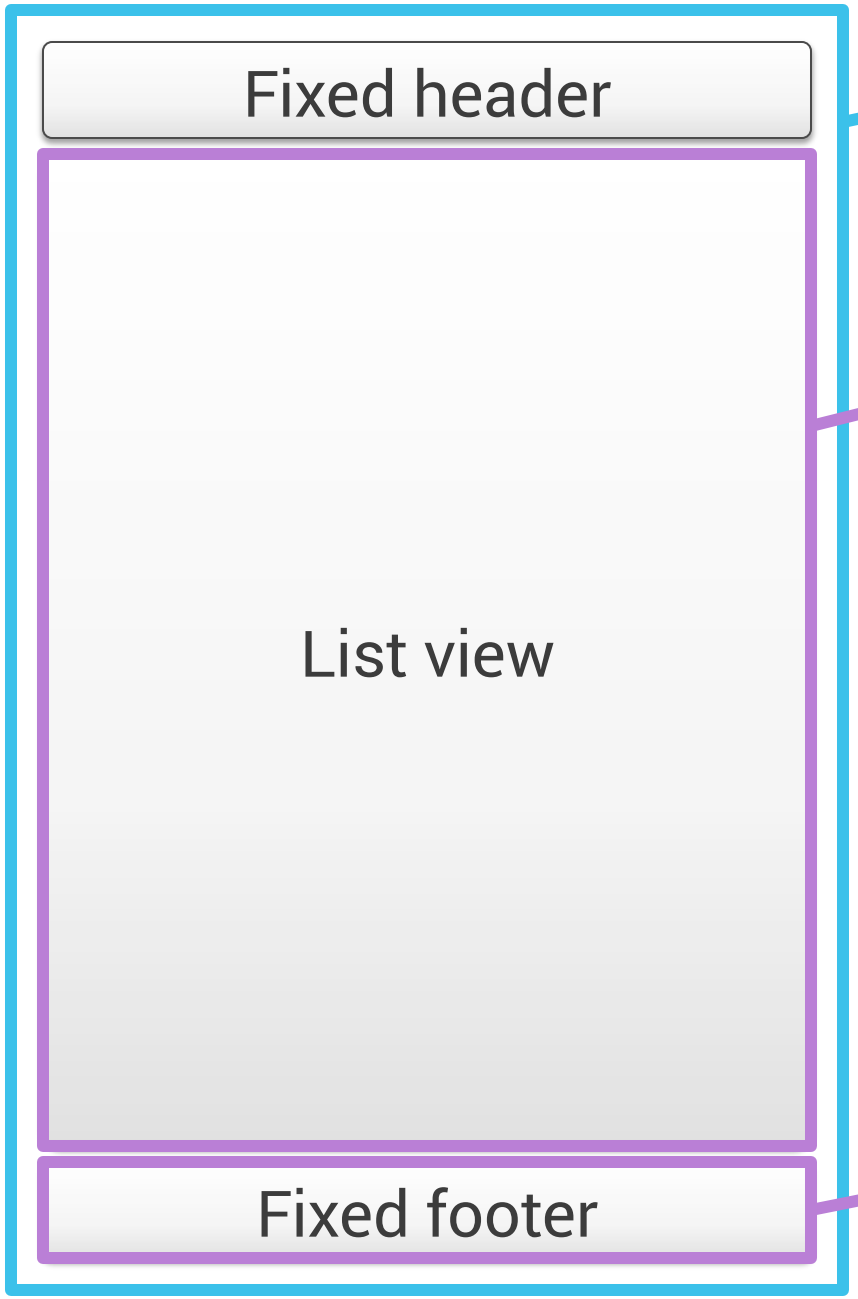
Code this layout


Button
Button
Button
Button


<FrameLayout>
<LinearLayout
orientation="vertical">
Button
Button
Button
Button
<LinearLayout
orientation="horizontal"
layout_weight="1">


Fixed header
List view
Fixed footer

1+2




```
<LinearLayout>
```

```
<ListView
 layout_weight="1">
```

```
layout_height="wrap_content"
```

App resources

- One universal app binary contains all resources
- System chooses at runtime which resources to use

res/

drawable	Drawable XML
drawable-xhdpi drawable-hdpi drawable-mdpi	PNGs, 9-patch PNGs, optimized for multiple densities
layout layout-land layout-large layout-large-land	Layout XML optimized for physical screen size and orientation
values	Strings, styles, themes, etc.
values-v11 values-v14	Styles, themes varying by API level
values-en values-fr values-ja	Strings XML localized for your target regions

Referencing resources

- A string in `res/values/strings.xml`
  - In Java: `R.string.hello`
  - In XML: `@string/hello`
- The system “edit” icon:
  - In Java: `android.R.drawable.ic_menu_edit`
  - In XML: `@android:drawable/ic_menu_edit`

Referencing resources


```
android.R.drawable.ic_menu_edit
```

```
@android:drawable/ic_menu_edit
```

Namespace (either `android` or blank)

Resource type

Resource name

Screen density and DIP units

DIP units keep things the
same physical size across any screen.

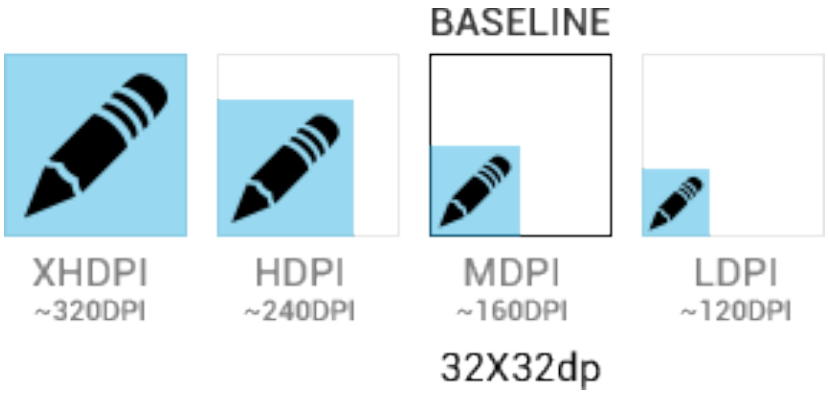
```
1 dip = 1 pixel  @ MDPI (160 dpi)

1 dip = 2 pixels @ XHDPI (320 dpi)
```

Q: What is the Nexus 7's screen resolution in DIPs if it's **1280x800 px** and **213dpi**?

A: ~960x600 dip

Screen density and DIP units

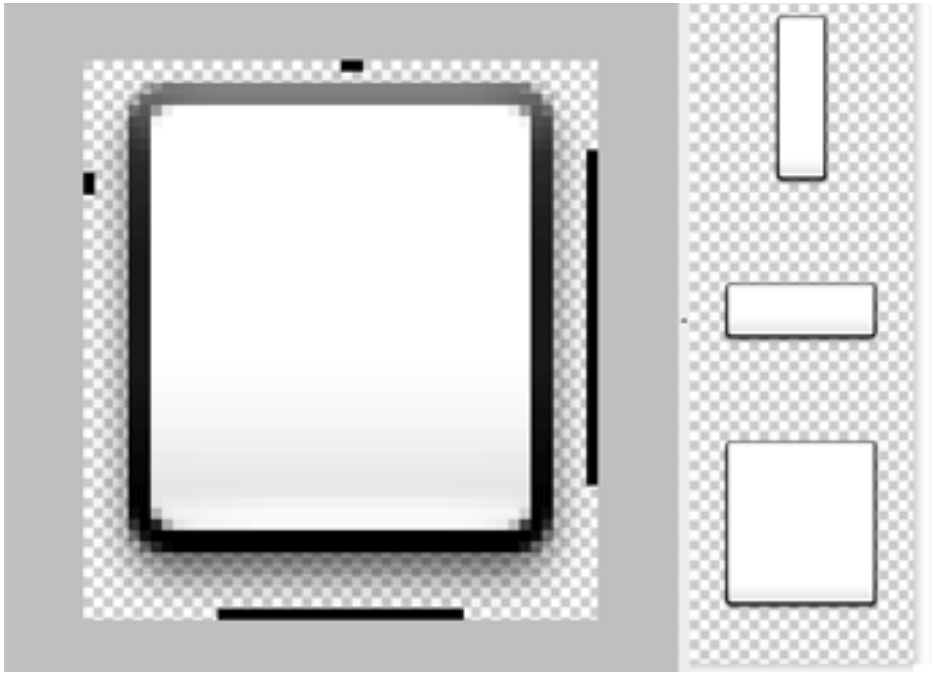
Icons and other PNG files should generally be provided for multiple densities



Key drawable types

- Bitmaps (.png)
- 9-patches (.9.png)
- State Lists (.xml)

9-patches – `foo.9.png`



- Border pixels indicate stretchable regions
- Make density-specific versions (`-xhdpi`)

State list drawables


drawable/


foo.xml

```
<selector>
    <item android:drawable="@drawable/foo_disabled"
          android:state_enabled="false" ... />
    <item android:drawable="@drawable/foo_pressed"
          android:state_pressed="true" ... />
    <item android:drawable="@drawable/foo_focused"
          android:state_focused="true" ... />
    <item android:drawable="@drawable/foo_default" />
</selector>
```

State list drawables

drawable-mdpi/


foo_default.png


foo_disabled.png


foo_focused.png


foo_pressed.png

drawable-hdpi/

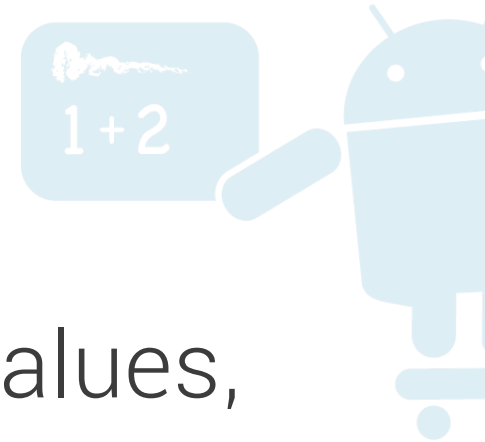

foo_default.png


foo_disabled.png


foo_focused.png


foo_pressed.png

Styles



Collections of layout XML attribute values, kind of like CSS

Instead of this...

```
<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:padding="4dp"
    android:text="1" />

<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:padding="4dp"
    android:text="2" />
```

Styles

...use this!

```
<TextView style="@style/MyText"
    android:text="1" />

<TextView style="@style/MyText"
    android:text="2" />
```

res/values/

styles.xml

```
<style name="MyText">
    <item name="android:padding">4dp</item>
    <item name="android:layout_width">match_parent</item>
    <item name="android:layout_height">wrap_content</item>
</style>
```

Themes

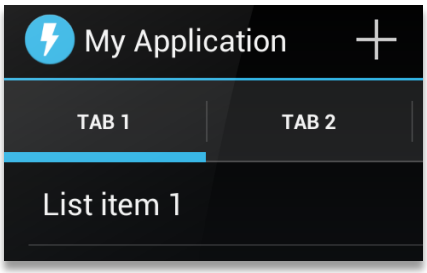
Are just styles that apply to activities (or the entire app)

AndroidManifest.xml

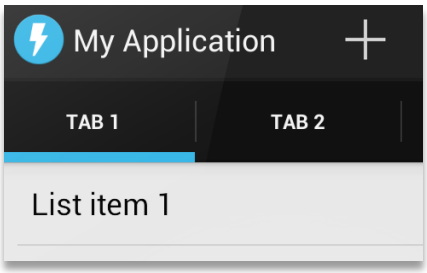
```
<application android:theme="@android:style/Theme.Holo">
    ...
</style>
```

You can extend the default themes!

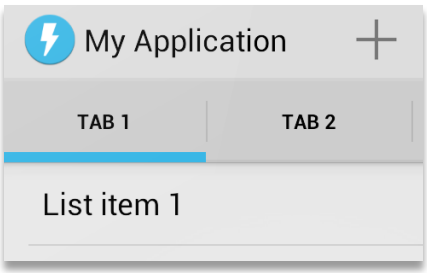
System themes



`@android:style/Theme.Holo`



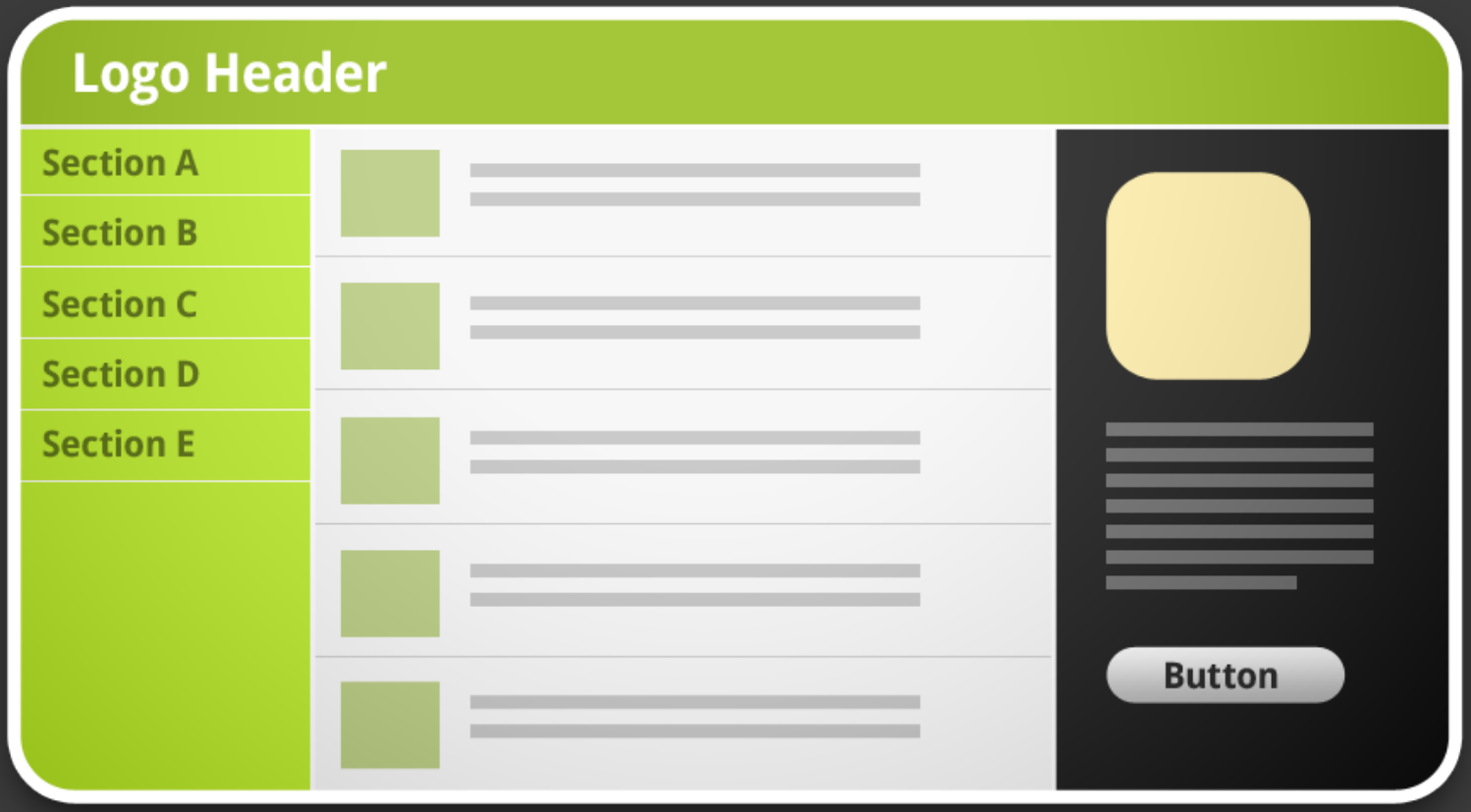
`@android:style/Theme.Holo.Light.DarkActionBar`



`@android:style/Theme.Holo.Light`

Tablet considerations

Information hierarchy and flow



Traditional desktop app or website

Information hierarchy and flow

Tablet or mini desktop app

Information hierarchy and flow

Mobile phone app

Fragments

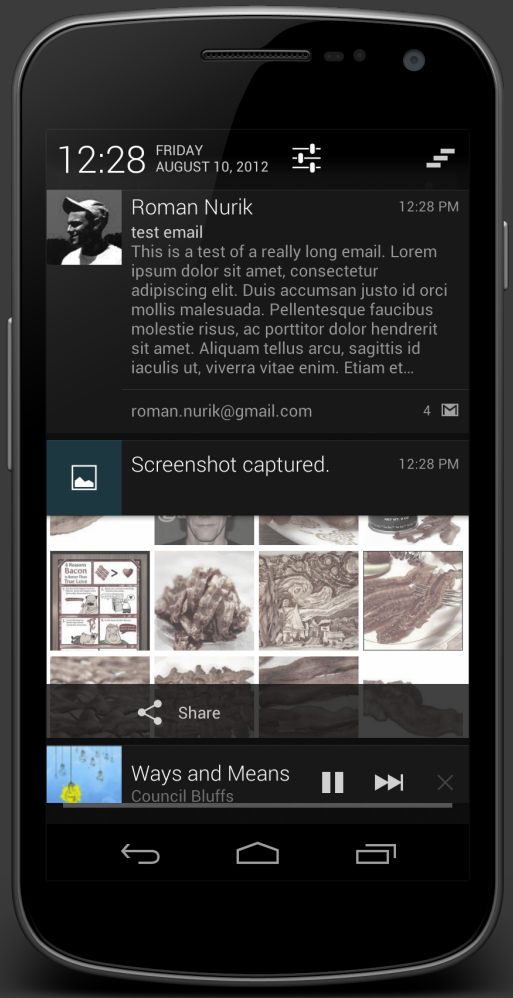
- Separate activities into UI or code **modules**, each have their own class
- Help with supporting phones + tablets
  - Each content pane is a fragment
  - Fragments split across activities
- `<fragment>` in layout XML
  - Automatically instantiates the given fragment
  - Acts as a placeholder for the fragment's view to be inserted in that part of the layout tree

Tablet resources

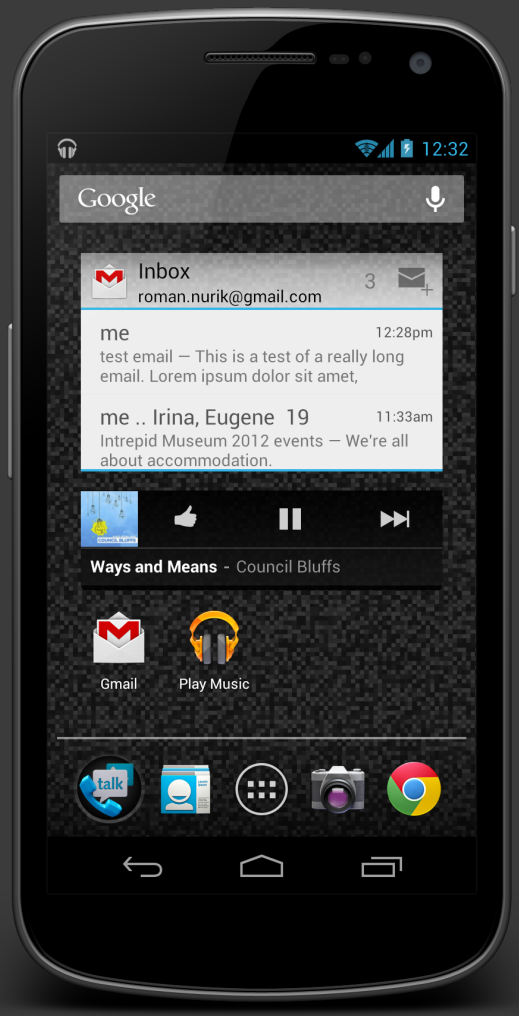
- Customize layouts for large screens
  - Includes `<fragment>` tags
- Incrementally increase font sizes, spacing, tweak styles

System UI integration

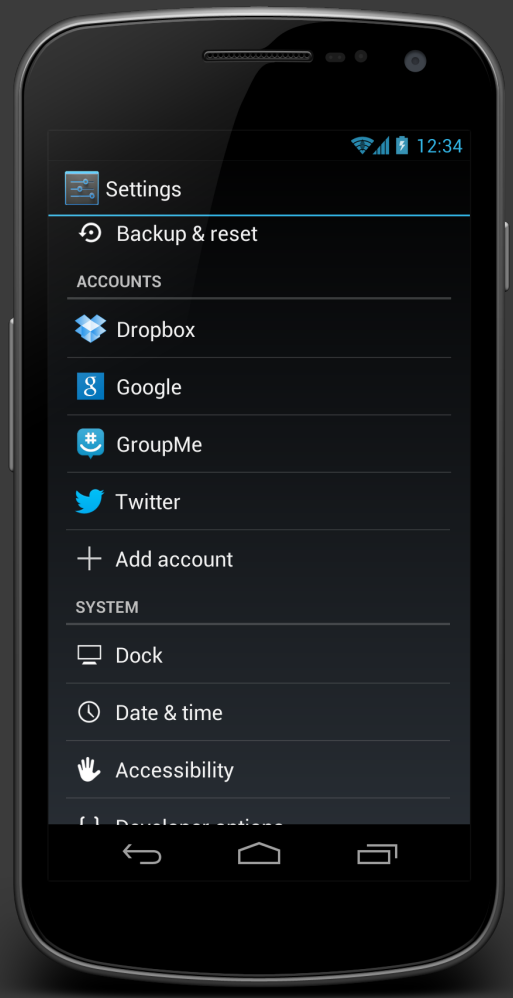
System UI integration



Notifications

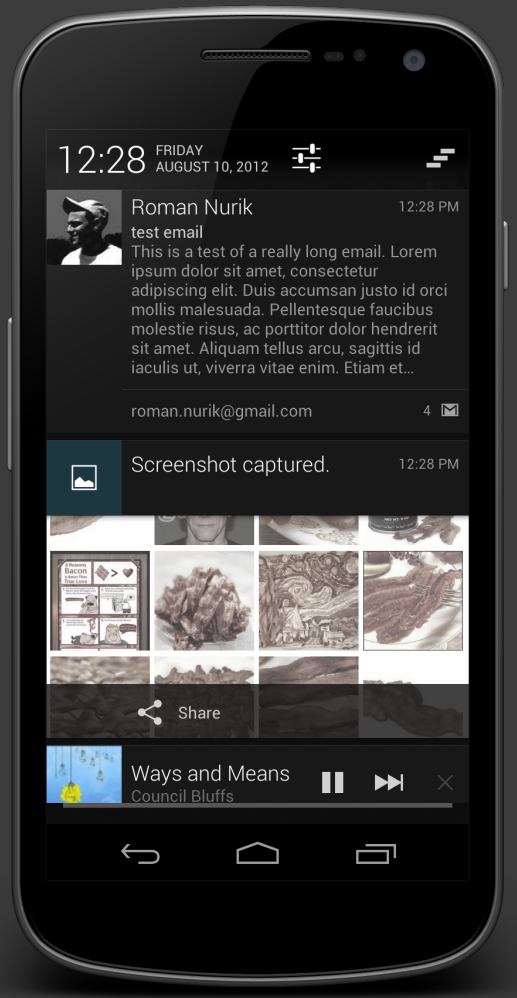


App Widgets

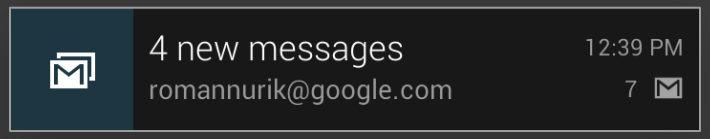


Accounts + Sync

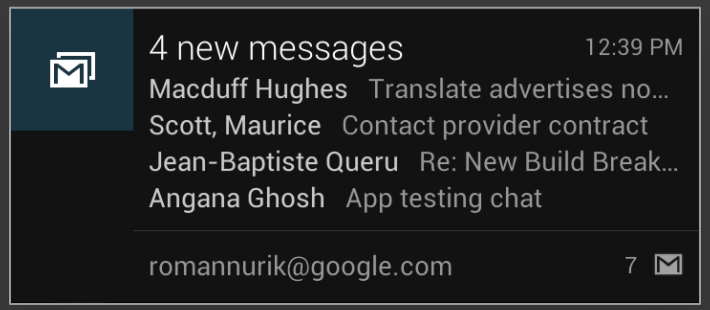
Notifications



Iconified



Collapsed



Expanded

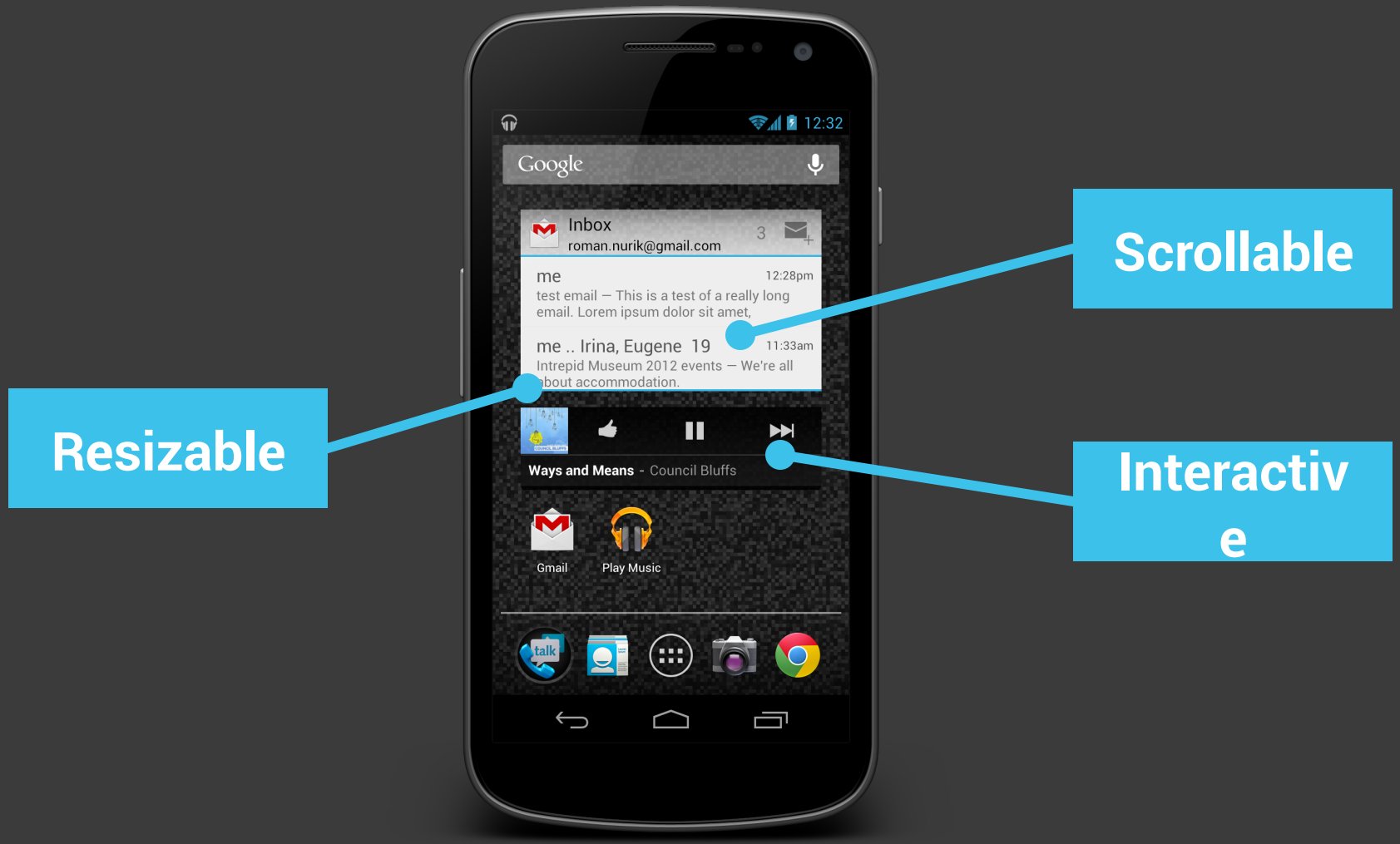
Android 4.1+

Notifications

Notifications

- Created with `NotificationManager` and `Notification.Builder()`
- Custom layouts with `RemoteViews`
  - Your layouts, used in a different process
- Interaction is handled entirely using intents

App widgets


12:32
Google
Inbox
roman.nurik@gmail.com
3
me
12:28pm
test email — This is a test of a really long email. Lorem ipsum dolor sit amet,
me .. Irina, Eugene 19
11:33am
Intrepid Museum 2012 events — We're all about accommodation.
Ways and Means - Council Bluffs
Gmail
Play Music
talk
Resizable
Scrollable
Interactive

App widgets

- Defined in your manifest as a receiver
  - Handles the `APPWIDGET_UPDATE` intent action
  - Metadata provided in `res/xml/widgetinfo.xml` and referenced in manifest
- Layout using `RemoteViews`
  - Your layouts, used in a different process
- Interaction is handled entirely using intents

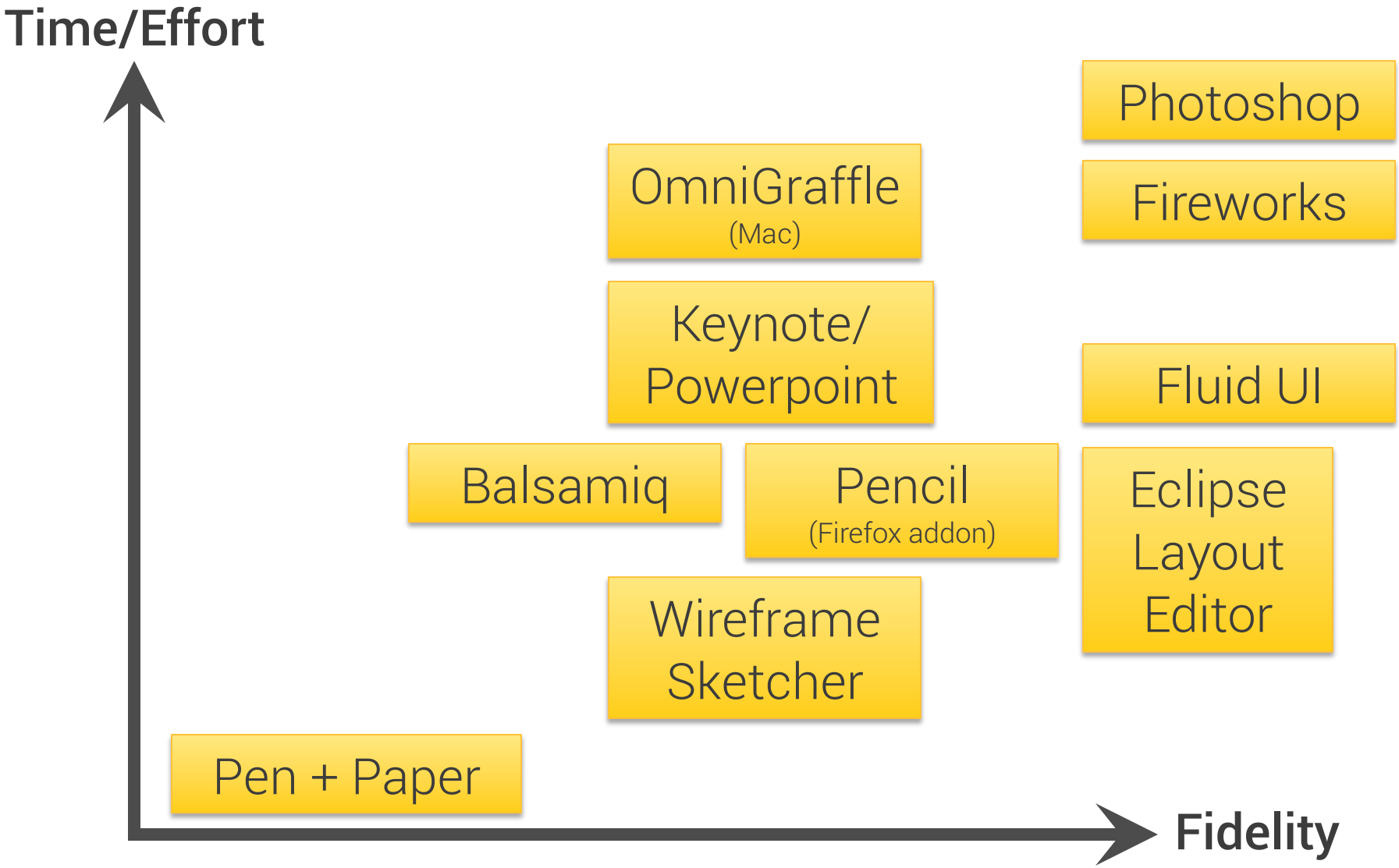
Wireframing

Why create wireframes?

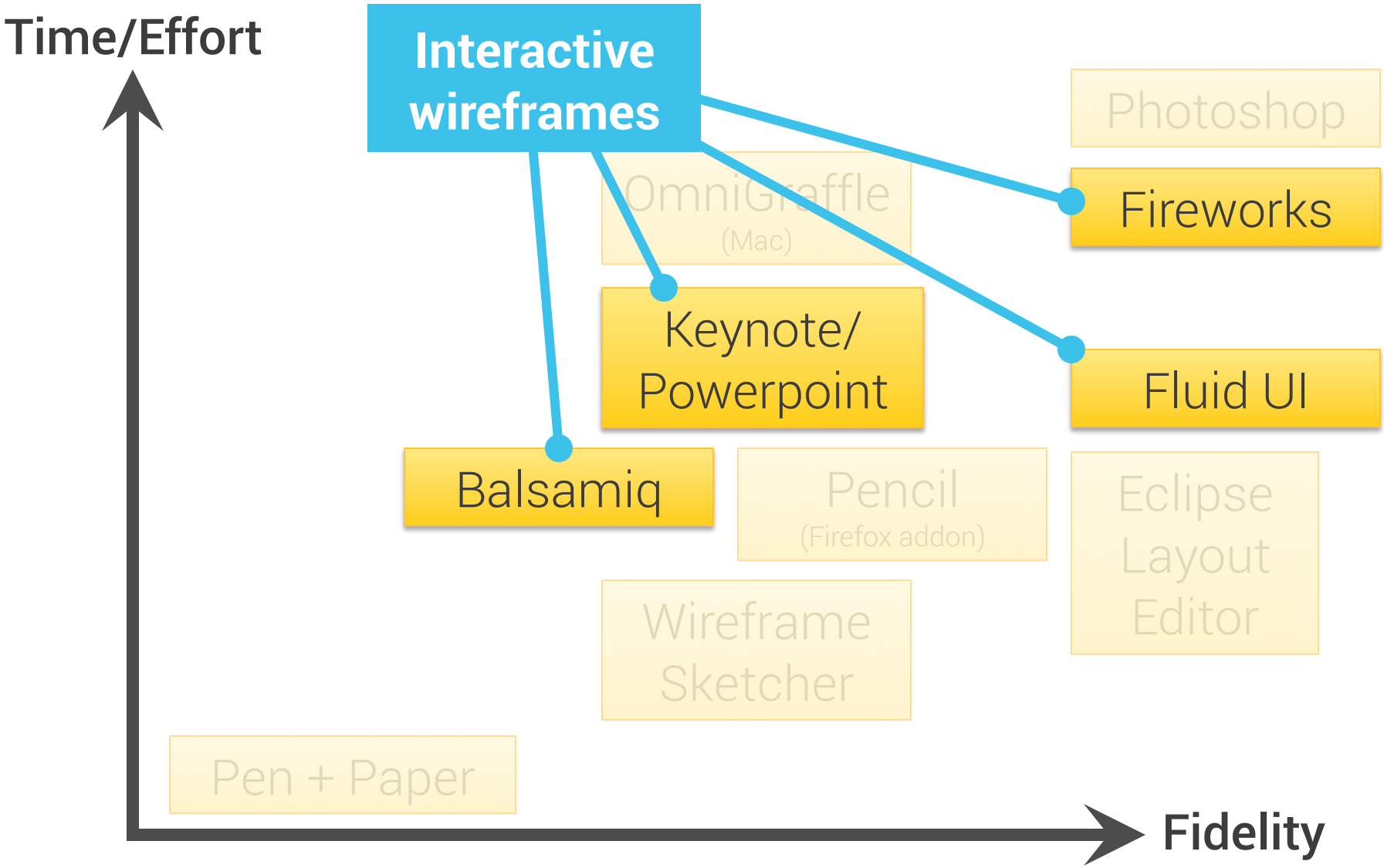
- Record your ideas and asses their real-world feasibility
- Test your your ideas and **rapidly iterate**
  - See which work and which don't, evolve them
- Map out user flow and activity diagrams
  - Re-arrange/add/remove interactions quickly
  - Scope UI complexity
  - Plan out intra-app "Intent-based API"

Wireframing before coding saves you time.

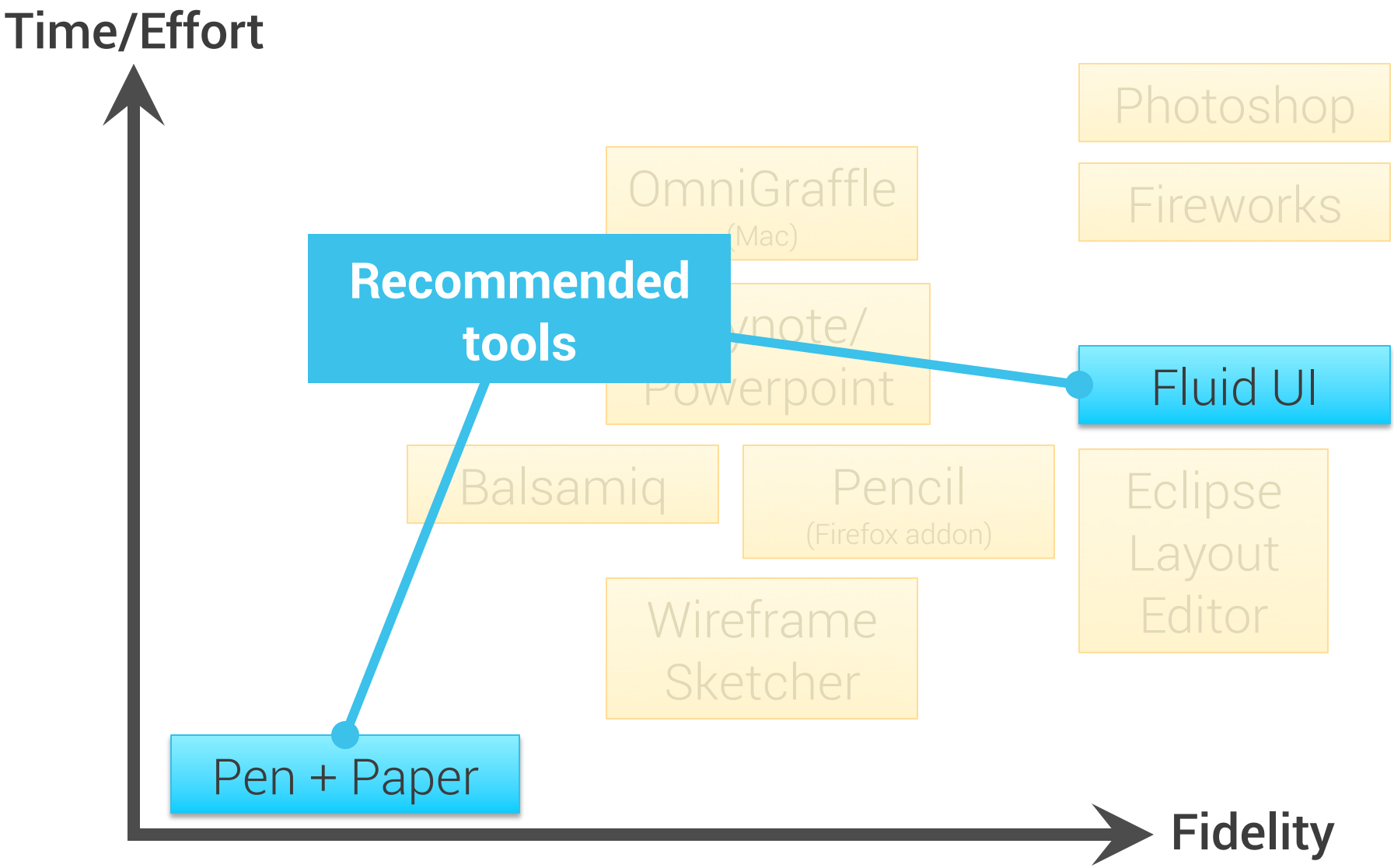
Wireframing tools



Wireframing tools



Wireframing tools



Always start with pencil and paper.

(or a whiteboard)


Google I/O
Sessions
Android
Android is the first free, open source, and fully customizable mobile platform.
Android Market for Developers
10:45am – 11:45am, Wed in Room 11
Building Aggressively Compatible Android Games
10:45am – 11:45am, Wed in Room 7
Memory management for Android apps
10:45am – 11:45am, Wed in Room 9
Android Office Hours
12:30pm – 3:00pm, Wed in Office Hours Space B
Designing and Implementing Android UIs for Phones and Tablets
12:30pm – 1:30pm, Wed in Room 11
Optimizing Android Apps with Google Analytics
12:30pm – 1:30pm, Wed in Room 9
Building Android Apps for Google TV
1:45pm – 2:45pm, Wed in Room 11
Optimizing Android Apps with Google Analytics
12:30pm – 1:30pm, Wed in Room 9
Summary
Notes
capabilities to improve the adoption and usability of Andriod Apps. This session covers best practices for tracking apps on mobile, TV and other devices. We'll also show you how to gain actionable insights from new tracking and reporting capabilities.
Speakers
Jim Cotugno
Jim is a software engineer on the Google Analytics Tracking Team with a focus on mobile tracking. Prior to Google, Jim developed operating systems, web applications and mobile applications.
Nick Mihailovski
Nick Mihailovski oversees Developer Relations for Google Analytics, working with clients to develop innovative measurement solutions. He has 6 years prior experience in the Web Analytics industry.
Philip Mui
4:25
Google I/O
Schedule
Map
Sessions
Starred
Sandbox
Bulletin
18 days, 16:05:46 until Google I/O
4:25

EXERCISE

Sketch a todo list app