CSIS 3375 – 001

Priya Kandhadai

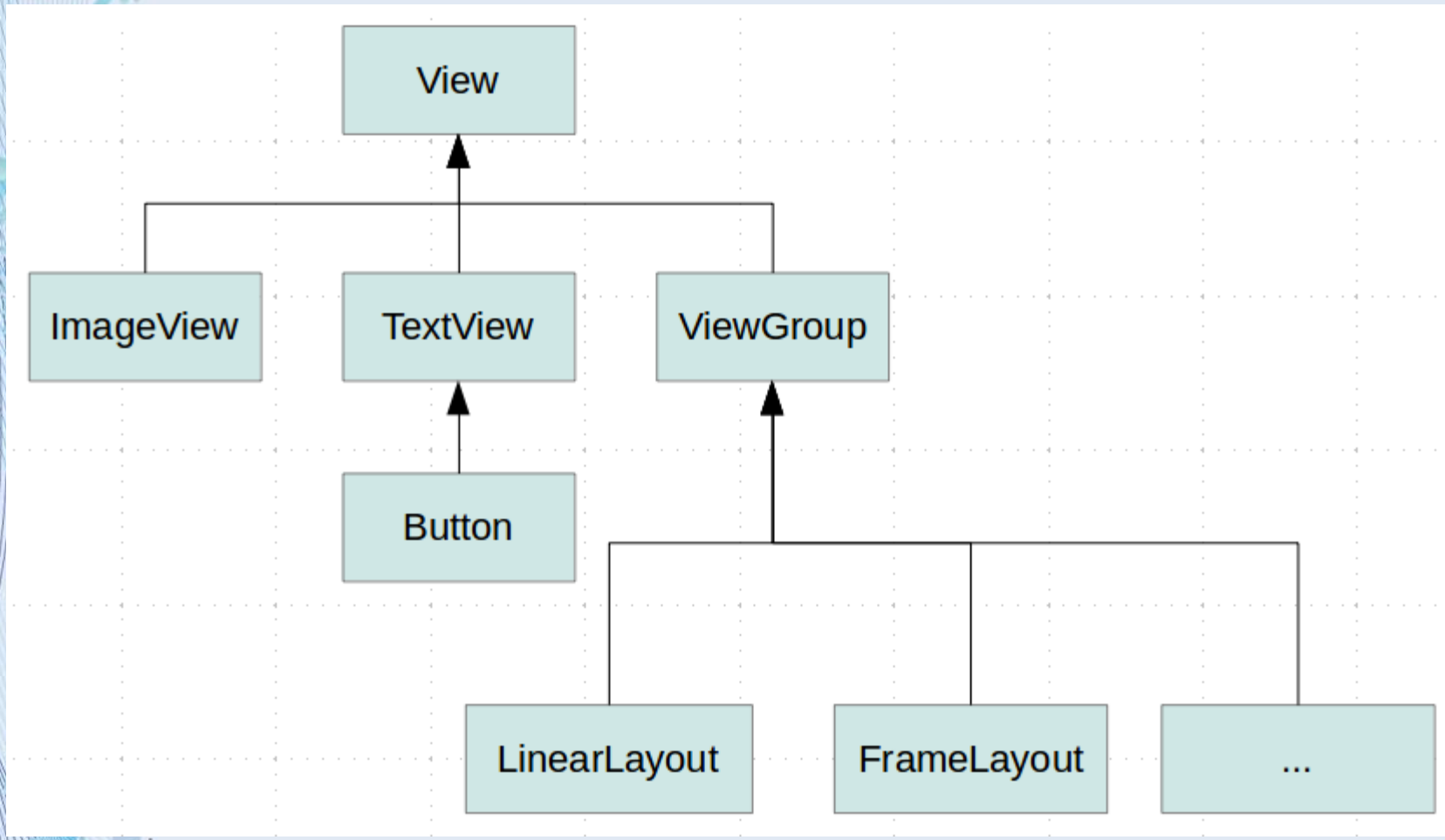
Building Android UI

Views in Android Building and Customizing UI

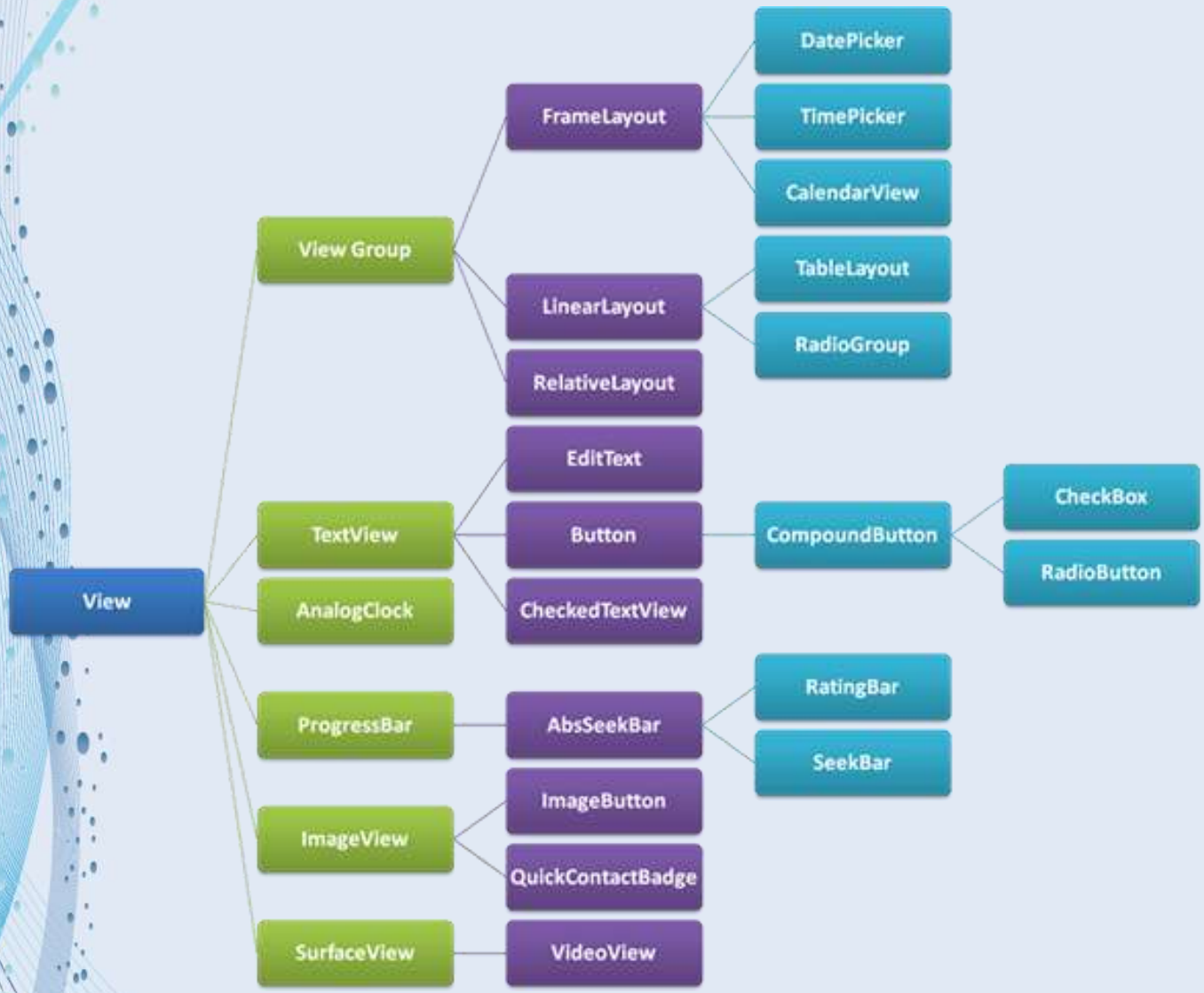
View in Android

- Extends java.lang.object
- Basic component of Android UI
- Two types: Views that stand alone, and views that are meant to group other views
- Views: has visual elements and can respond to events
- Every View: needs attributes from XML layout and JAVA code to set attributes and event handlers
  - Typically: XML layout for visual component
  - Java: event handlers, and change of visual based on events

View Class Hierarchy



Android Class Heirarchy



Source: codentrick.com

Some common stand alone views

- TextView
- EditText
- ImageView
- Button
- CalendarView
- AutoCompleteTextView
- CheckBox
- CheckedTextView
- CompoundButton
- DatePicker

View – common attributes

- android:id
  - Id for the view
  - Used in code
  - setId(int id)
- android:alpha
  - Alpha level defines transparency (0 – full transparent, 1 – fully opaque)
  - setAlpha(float)
- android:background
  - sets drawable to use as background
  - setBackgroundResource(int)

View Attributes and Methods

- android:contentDescription
  - Sets the description that describes the view (e.g., often used in imageview or imagebutton)
  - setContentDescription(CharSequence)
- android:padding
  - setPadding(int, int, int, int) – left, top, right, bottom
- android:focusable
  - Sets the focusable
  - setFocusable (int focusable): NOT_FOCUSABLE, FOCUSABLE, or FOCUSABLE_AUTO
- android: visibility
  - setVisibility(int): VISIBLE, INVISIBLE, or GONE.

TextView

- Multiline text: How to add newline breaks.
  - Add \n in XML layout
  - OR press shift+enter in translation editor to allow multiline string literal
- EditText: Extends TextView used for user input
  - Type of keyboard and data entered depends on EditText Type

TextView’s attributes

- drawableLeft, drawableRight, drawableTop, drawableBottom:
  - setCompoundDrawables (Drawable left, Drawable top, Drawable right, Drawable bottom)
- android:drawablePadding
  - The padding between the drawables and the text
  - setCompoundDrawablePadding(int)
  - 5dp
- android:gravity
  - Specifies how to align the text by the view's x- and/or y-axis when the text is smaller than the view.
  - Must be one or more (separated by '|') of the following constant values.

TextView’s attributes

- android:gravity
  - Specifies how to align the text by the view's x- and/or y-axis when the text is smaller than the view.
  - Must be one or more (separated by '|') of the following constant values
  - setGravity (int gravity) – view.setGravity(Gravity.CENTER | Gravity.BOTTOM);
- android:fontFamily
  - Sets the font family
  - setTypeface(Typeface.create("sans-serif-light", Typeface.NORMAL));

TextView’s attributes

- android:lineSpacingExtra, android:lineSpacingMultiplier
  - setLineSpacing(float extra, float multiplier)
  - textView.setLineSpacing(0.0f, 1.1f);
- android:lines
  - Sets exact number of lines
  - setLines(int)
- android:maxLines, android:minLines
  - setMaxLines(int), setMinLines(int)

TextView’s attributes

- android:text
  - Text for TextView - setText(CharSequence)
- android:textColor
  - Color for TextView: setTextColor(int)
  - textview.setTextColor(this.getResources().getColor(R.color.orange))
- android:textIsSelectable
  - setTextIsSelectable(boolean)
- android:textSize
  - setTextSize(int, float)
  - tv.setTextSize(TypedValue.COMPLEX_UNIT_SP, 18f);
- android:textStyle (use setTypeface() method)
  - BOLD, ITALIC, BOLD_ITALIC, NORMAL
  - textView.setTypeface(null, Typeface.BOLD);

EditText and Button

- Both Extend TextView
- Button for events
- EditText used to gather user input

ImageView and ImageButton

- ImageView
  - ScaleType
- ImageButton
  - Extends ImageView

Views for gathering input

- AutoCompleteTextView – EditText with suggestions as the user is typing
- CalendarView – this view lets you display dates to users for date input
- CheckBox
- CheckedTextView: text view that can be checked..typically used in a list view
- DatePicker

Views for gathering input

- NumberPicker
- RadioButton
- Spinner
- Switch
- TimePicker
- ToggleButton
- VideoView

Common View Events

- OnClickListener
- OnLongClickListener
- OnTouchListener: Uses GestureDetector
  Swipe left, right, up and down

Android Demo: Customizing Views & Custom Touch Events

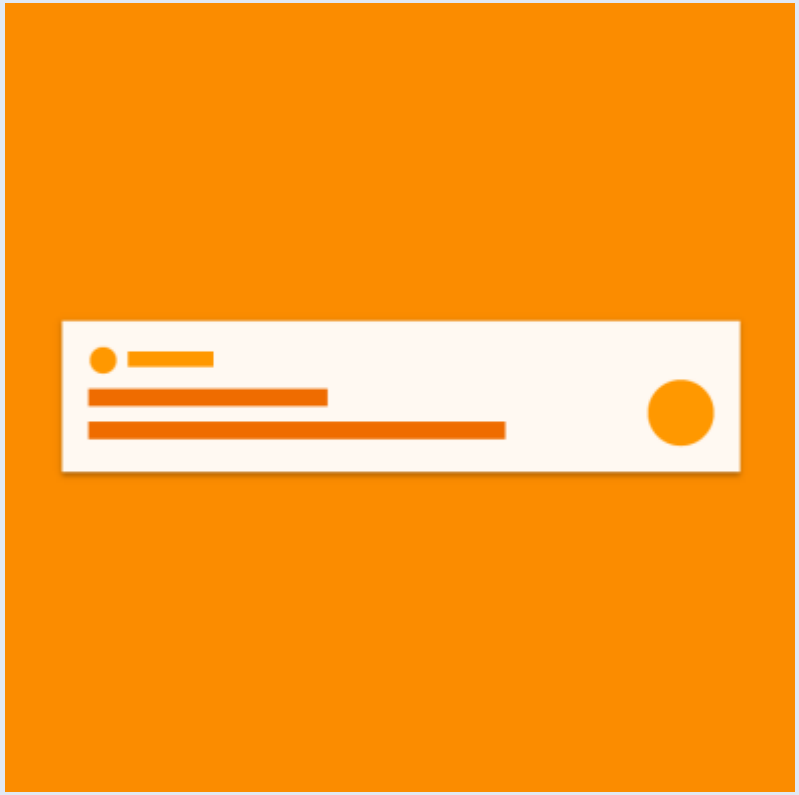
- Customizing TextViews
- Customizing ImageViews
- Working with Visibility, background, strike-through text, text color and text size
- Custom Touch Listener on Views
- Gestures: using gesture detector
  - Single Click, Double Click, Long Click
  - Left, Right, Up and Down Swipe

Android UI: System Bars

- System Bars:
  - Provide system relevant information
  - Android status bar: gives preview of notifications, phone settings information
    - Height: 24dp
  - Android bottom navigation bar
    - Height: 48dp

Android UI: Notifications

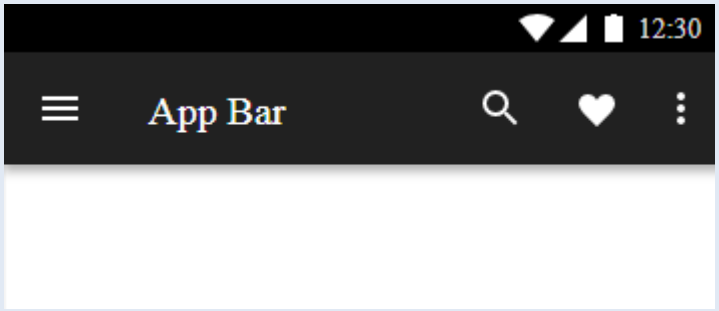
- Notifications: provide short, timely, and relevant information about your app when it’s not in use
  - Background task progress
  - Communications from other users
  - App-relevant reminders (sparse use)
- Snack bars and toasts



Android UI: App bar

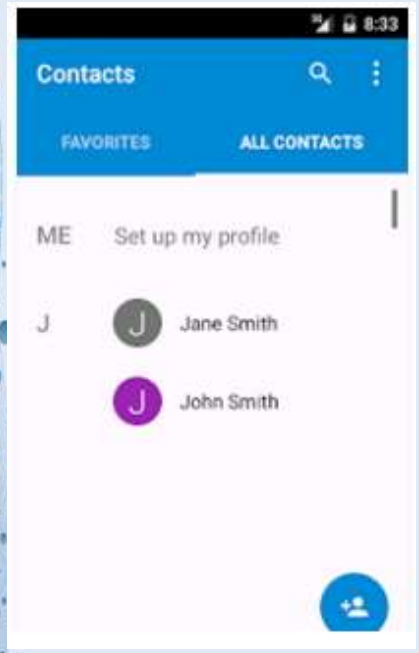
- App bar: A tool bar specific to the app that sits at the top of app
- Right below status bar
- Previously called action bar
- Standard height: 56dp on mobile devices
- May have another tool bar at the bottom (bottom tool bar)

- App bar may be hidden
  - Reading apps

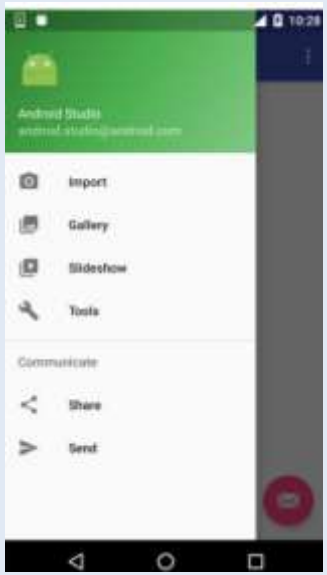


Android UI – Tabs and Navigation drawer

- Tabs: usually at top below app bar
  - No more than 2-3 tabs

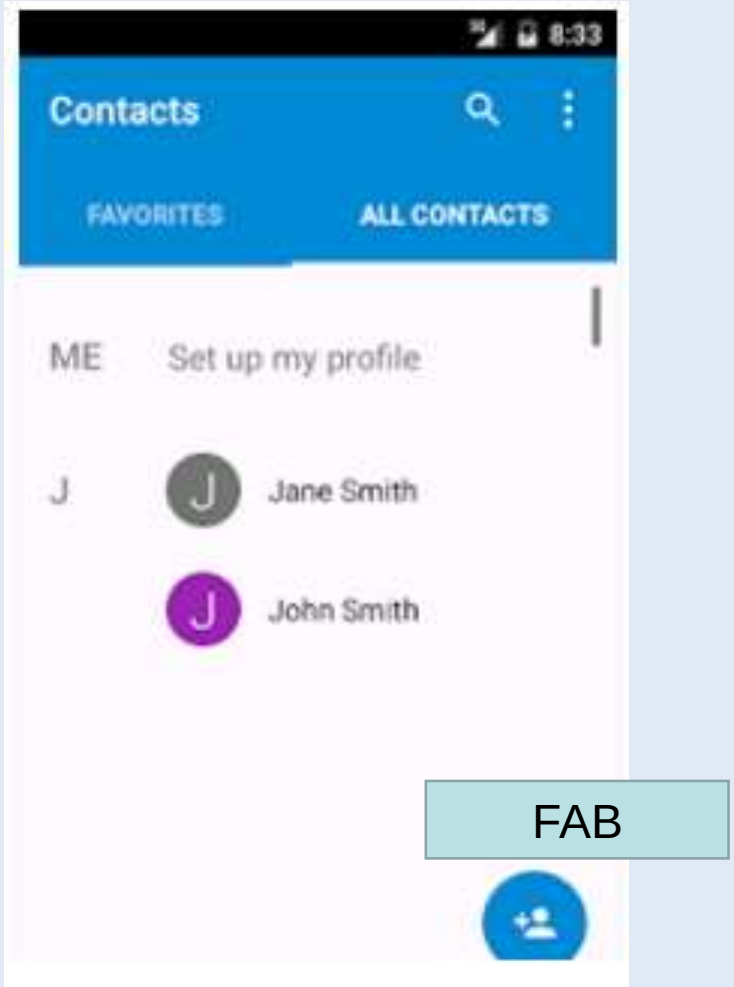


- Navigation drawer:
  - Several app sections
  - Full screen: sheet front of app bar

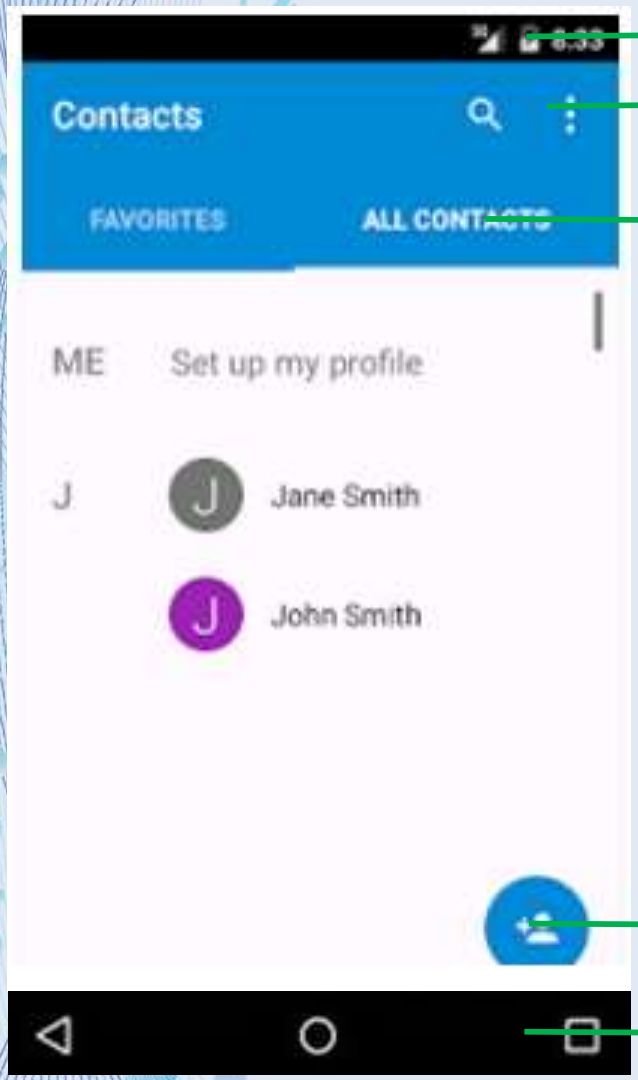


Android UI: FAB

- Floating Action Button
  - FAB
  - Typically accent color (secondary color)
  - Draw attention to important action
  - Eg: new event, new contact, new note in apps
  - Default: 56 X 56dp



Android UI



Status Bar

App Bar

Tabs

FAB

Navigation Bar

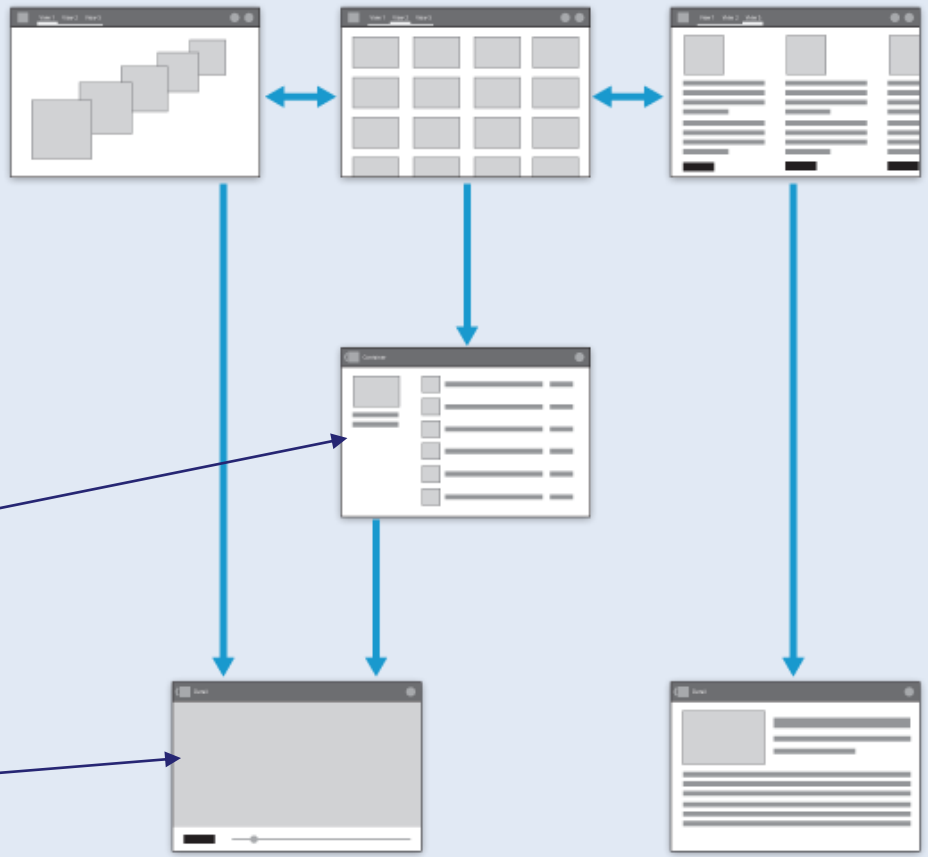
Newer Android Apps

- Avoid Menu Button
- Long Press denotes selection rather than context menu (right-click)

- Notifications: white or fully transparent icons.
  - Cross-app compatability especially important
- Avoid styles from other platforms
  - iOS and Android styles are drastically different

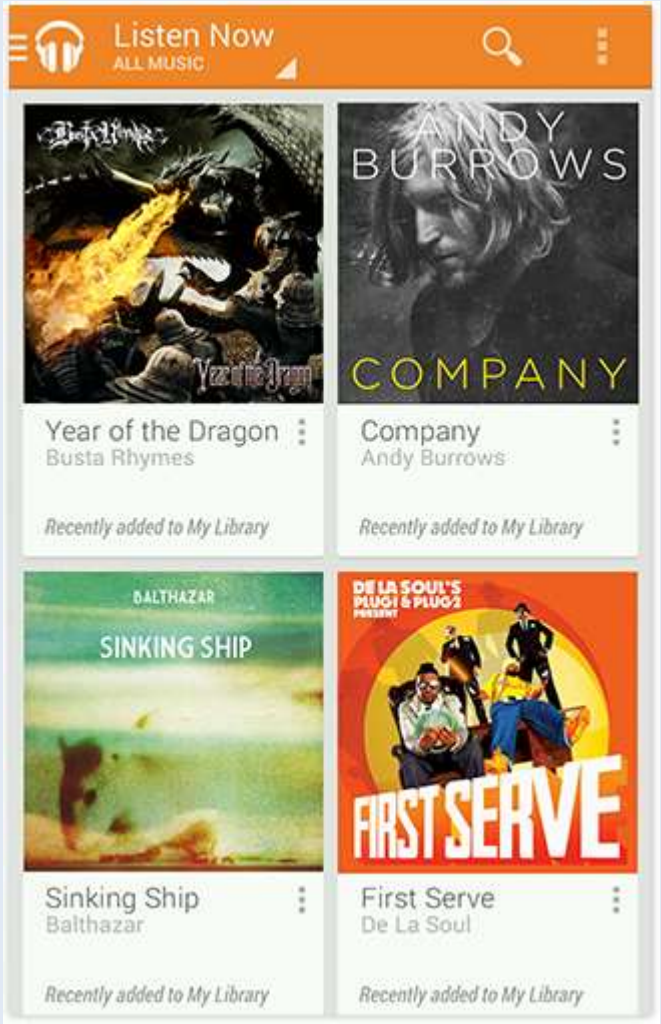
Basic Android Structure

- Top level views: different views that your app supports.
  - different representations of the same data
  - different functional facet of your app.
- Category views
  - drill deeper into your data.
- Detail/edit view: Consume or create data.



Top-Level

- Use bright and engaging layouts
- What do users of the app typically want to do?
- Identify different top-level views in your apps
- Add content to top-level view wherever possible
  - Engaging and fun!



App Bars

- App Bar: display your app's icon or title.
- If your top level consists of multiple views, add view controls to your app bar
- If your content is searchable, include the Search action in the app bar so people can cut through the navigation hierarchy.

- Identify functionality or utility of app bars in your app

Fixed Tabs

- Remains on the screen always at the top-level
- Allows multiple top-level views

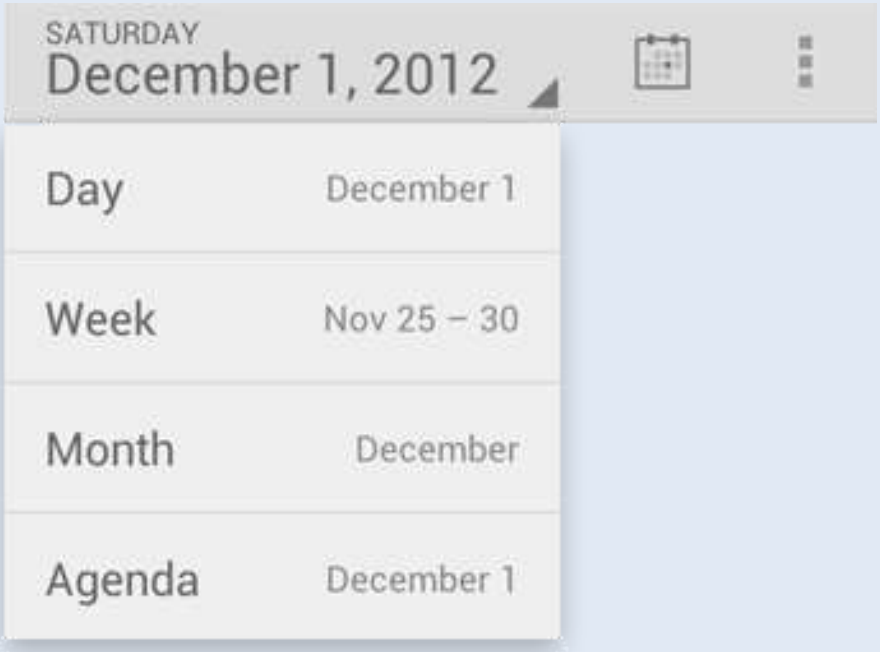
- User needs to switch between views frequently
- User needs to be made aware of the alternate views
- No more THAN three fixed tabs

Spinners

A spinner: drop-down menu

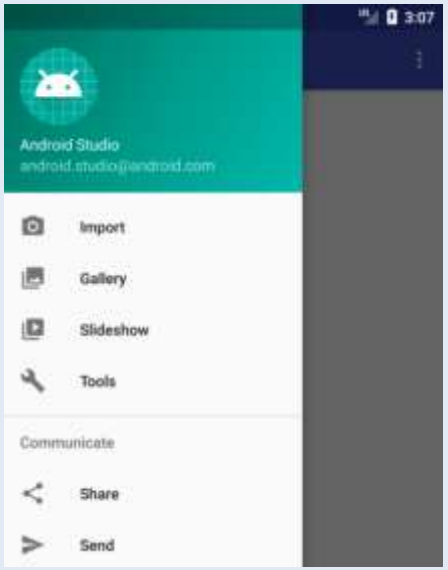
Use spinner in app bar if:

- Instead of dedicated fixed tab
- The user is switching between views
  - of the same data set: calendar events viewed by day, week, or month or
  - data sets of the same type: such as content for two different accounts



Navigation drawers

- **Use navigation drawers if**:
- You don’t want to waste navigation space
- You have a large number of top-level views.



- You want to provide direct access to screens on lower levels.
- You want to provide quick navigation to views which don't have direct relationships between each other.

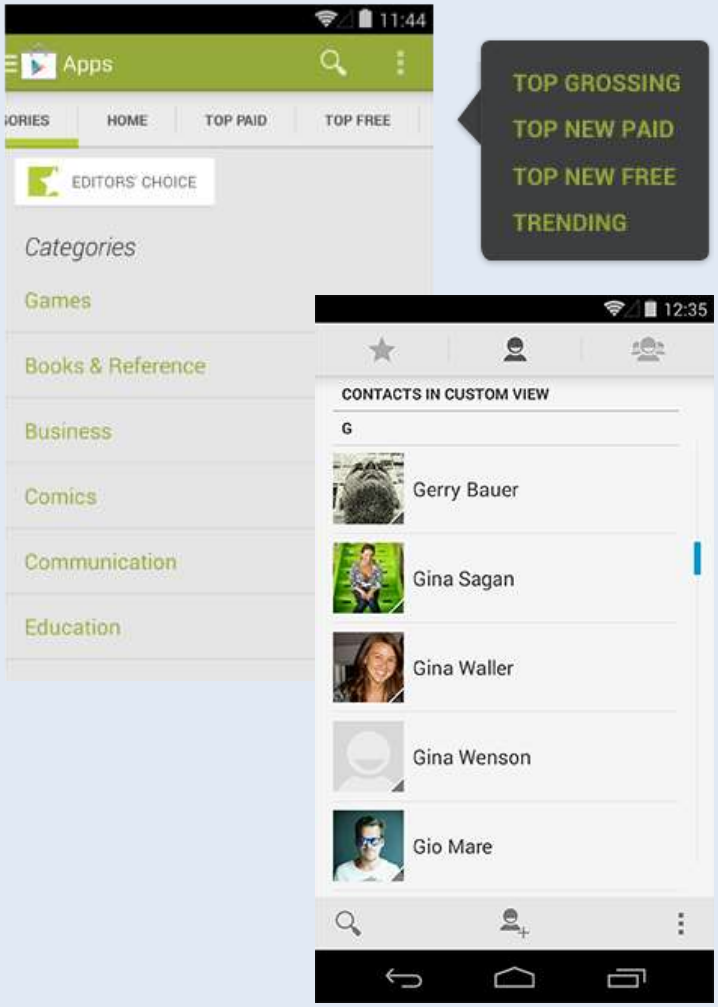
You have particularly deep navigation branches.

Top-level

- Use Tabs, Spinners and Navigation drawer appropriately
  - Don’t mix and match
  - Stick with one depending on your need

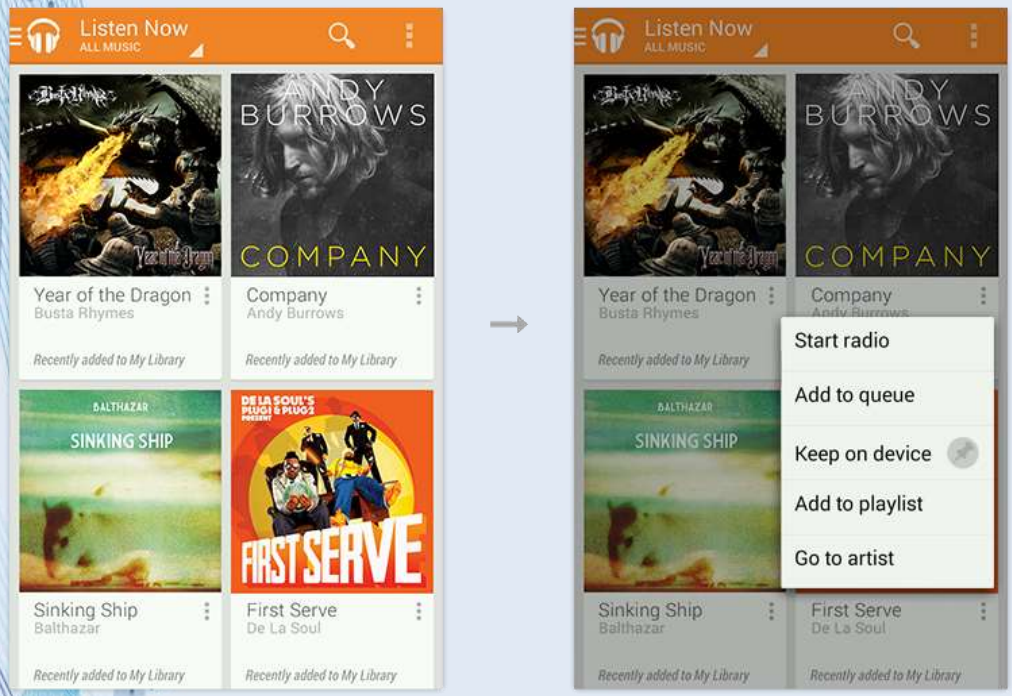
Category-View

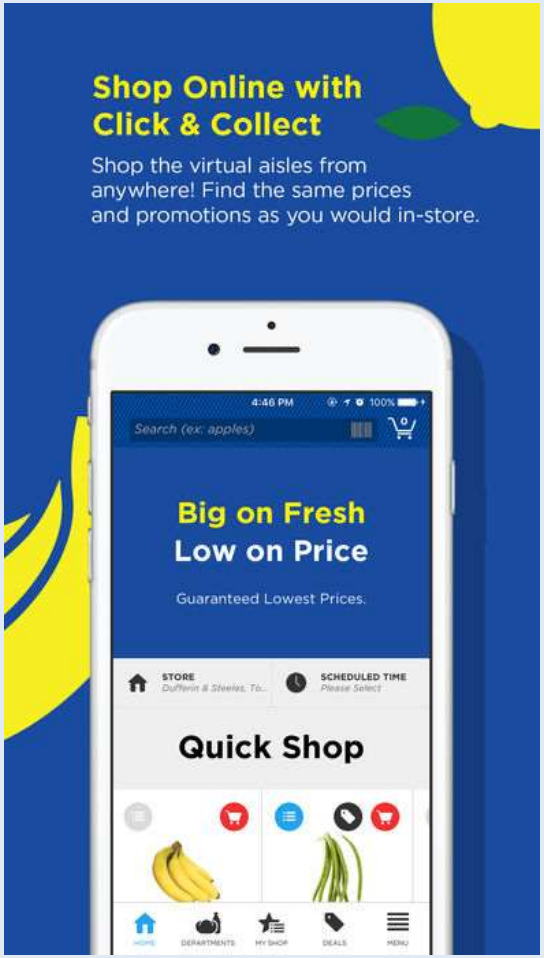
- Used in apps that have deep hierarchical structures
  - Use scrolling tabs for related categories
  - Use fixed tabs for unrelated categories



Allow cutting through hierarchies

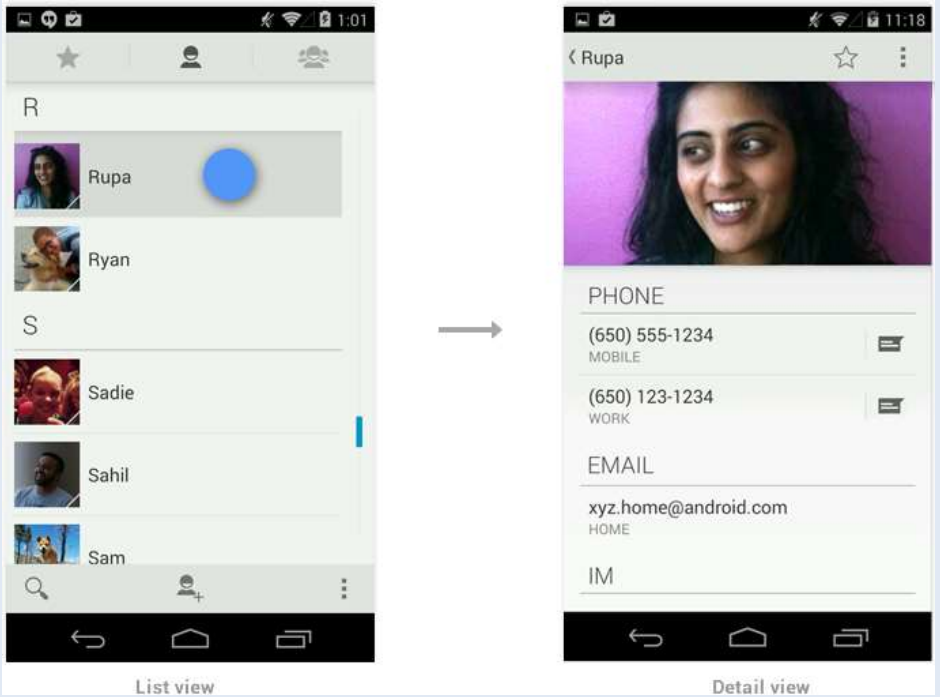
- Display prominent actions directly on the list items





Detail View

- Less is more
- Think about order of content processing to arrange layout
- Allow navigation between multiple detail views
  - Swipe to retrieve next item



Overall Android App Structure

- Find ways to display useful content on your start screen.
- Use app bars to provide consistent navigation.
- Maintain shallow hierarchies: use horizontal navigation and shortcuts
- Use multi-select to allow the user to act on collections of data.
- Allow for quick navigation between detail items with swipe views.

Android UI

- Basic Android UI components
  - What? When to use?
- Android UI structure
  - Overall layout
  - Three levels of Views
- Top-level vs. Category vs. Detail Views
  - What? When to use?
- Use of tabs, spinners, navigation drawers
- Design consideration for Android UI structure

Research Activity: Identify these in TWO apps of your choice

- Basic Android UI components: system bar, app bar, FAB, tabs, navigation drawer, notifications, android navigation bar, bottom navigation
- Android UI structure: Identify top-level, category and detail views
- Use of tabs, spinners, navigation drawers

Fixed Tabs

- Remains on the screen always at the top-level
- Allows multiple top-level views

- User needs to switch between views frequently
- User needs to be made aware of the alternate views
- No more THAN three fixed tabs